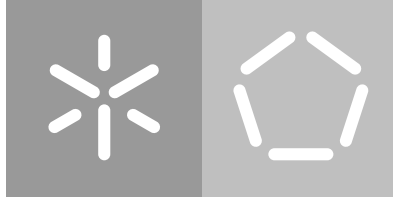




Universidade do Minho
Escola de Engenharia

Gianpaolo Gulletta

**From Human Motor Control to Human-like
Motion in Robotics: planning, learning and
controlling manipulation in Anthropomorphic
Robotic Systems**



Universidade do Minho

Escola de Engenharia

Gianpaolo Gulletta

From Human Motor Control to Human-like Motion in Robotics: planning, learning and controlling manipulation in Anthropomorphic Robotic Systems

Tese de Doutoramento

Doutoramento em Engenharia Eletrónica e de Computadores

Trabalho efetuado sob a orientação do(a)

Professora Doutora Estela Guerreiro Silva Bicho Erlhagen

Professor Doutor Wolfram Erlhagen

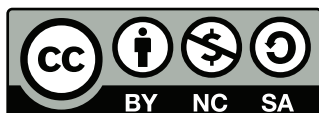
COPYRIGHT AND TERMS OF USE OF THIS WORK BY A THIRD PARTY

This is academic work that can be used by third parties as long as internationally accepted rules and good practices regarding copyright and related rights are respected.

Accordingly, this work may be used under the license provided below.

If the user needs permission to make use of the work under conditions not provided for in the indicated licensing, they should contact the author through the RepositoriUM of Universidade do Minho.

Licença concedida aos utilizadores deste trabalho



**Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International
CC BY-NC-SA 4.0**

<https://creativecommons.org/licenses/by-nc-sa/4.0/deed.en>

Acknowledgements

I would like to thank Prof. Dr. Estela Bicho and Prof. Dr. Wolfram Erlhagen for their scientific guidance and support during my years of research. Their vast expertise and, especially, their friendship were essential and decisive for the development of this project. My deepest gratitude goes to them.

I would also like to thank Prof. Dr. Eliana Oliveira da Costa e Silva and Prof. Dr. Maria Fernanda Pires Costa for their numerous and crucial scientific consultations concerning applied mathematics for optimization and the evolution of human-like motion planning in robotics. Both their theoretical and practical support revealed fundamental at the early stage of development of this project.

I am also very grateful to my friends and colleagues of the MARLab at the University of Minho: Luis Louro, Flora Ferreira, Paulo Vicente, Rui Silva, Carlos Faria, Tiago Malheiro, Toni Machado and Weronika Wojtak. Most of the hardware setup would have been impossible without their technical support and important recommendations. I feel blessed to have shared my workspace with such a smart, friendly and supportive team.

My special thanks also go to my colleagues of the Marie Curie NETT project. The challenging training, the interdisciplinary composition of the team and the amicable atmosphere encouraged me to embrace very different research directions without fear and prejudice.

This work was partly funded by the EU Project FP7 Marie Curie NETT-Neural Engineering and Transformative Technologies (ID 289146), the FCT PhD grant (ref. SFRH/BD/114923/2016) and the FCT Project UID/MAT/00013/2013.

STATEMENT OF INTEGRITY

I hereby declare having conducted this academic work with integrity. I confirm that I have not used plagiarism or any form of undue use of information or falsification of results along the process leading to its elaboration.

I further declare that I have fully acknowledged the Code of Ethical Conduct of the Universidade do Minho.

29/03/2021, Universidade do Minho
(Date) (Location)

A handwritten signature in black ink, reading "Gianpaolo Gulletta", written over a horizontal line.

(Gianpaolo Gulletta)

Resumo

Como os robôs estão começando a fazer parte do nosso dia a dia, eles devem ser capazes de cooperar de maneira natural e eficiente com os humanos para serem socialmente aceitos. A morfologia e os movimentos semelhantes aos humanos são frequentemente considerados recursos essenciais para interações intuitivas entre humanos e robôs, uma vez que permitem que colegas humanos prevejam facilmente os movimentos robóticos. Este projeto de doutorado diz respeito ao projeto e ao desenvolvimento de uma estrutura modular que pode transferir as características motoras humanas típicas de planejamento, aprendizagem e controle dos movimentos dos membros superiores para braços robóticos antropomórficos. A maior parte do trabalho apresentado diz respeito ao módulo de planejamento de movimento, que é capaz de gerar trajetórias livres de colisão com características cinemáticas humanas. Mais especificamente, o processo de planejamento leva em consideração uma hierarquia dependente da tarefa de restrições espaciais e posturais modeladas como funções de custo e, mais importante, aborda a prevenção de obstáculos de uma maneira naturalística. Além disso, a introdução de uma separação temporal de um movimento específico em fases fornece um certo grau de precisão humana, que pode ser necessária em tarefas de manipulação complexas. Para quantificar a semelhança humana, medidas de suavidade e regularidades de movimento cinemáticas, que são aplicadas em estudos de controle motor humano, são aqui adotadas para distinguir entre movimentos bem coordenados e prejudicados. Testes com dois dispositivos robóticos em diferentes tarefas pick-and-place mostraram a capacidade do planejador proposto de gerar trajetórias de membros superiores semelhantes às humanas com um custo computacional que é suficientemente pequeno para permitir interações fluentes entre humanos e robôs.

Na presença de pequenas perturbações esperadas do ambiente durante a execução de tarefas semelhantes, os mecanismos de aprendizagem podem aumentar a familiaridade com sequências semelhantes de ações, melhorando o desempenho do planejamento geral. Com o objetivo de mimetizar o ciclo de aprendizagem experiencial humano, o que permite ganhar familiaridade por meio da experiência em campo, também é proposto um aprendiz de movimento incremental adaptativo. Após sessões suficientes de treinamento, testes com movimentos de alcance demonstraram que este módulo é capaz de acelerar o

processo de planejamento e se adaptar a perturbações consistentes do cenário.

A fim de lidar com perturbações inesperadas do espaço de trabalho, como a mudança repentina do alvo da mão, os sinais de feedback permitem uma redefinição da trajetória dos membros superiores para a realização bem-sucedida da tarefa dada. Por este motivo, o estudo de controladores de loop de feedback também faz parte deste projeto de doutorado para rastrear movimentos planejados de membros superiores semelhantes aos humanos e reagir prontamente a eventuais perturbações imprevistas do espaço de trabalho inicial. A aplicação desses controladores em diferentes tarefas manipulativas demonstrou a capacidade de manter as características cinemáticas humanas das trajetórias desejadas na ausência de perturbações significativas. Quando o espaço de trabalho é perturbado, um movimento controlado pode se adaptar on-line e concluir com sucesso a tarefa dada, mas o nível de semelhança humana alcançado dificilmente pode ser discutido devido à falta de métricas válidas de comparação.

A separação e a interconexão dos módulos de planejamento, aprendizagem e controle compõem uma estrutura estruturada de geração de movimento semelhante ao humano para manipuladores antropomórficos. Esta divisão das diferentes funcionalidades sugere direções coordenadas de investigação para uma integração mais completa das habilidades motoras humanas em dispositivos robóticos centrados no homem.

Palavras-chave: Robótica cognitiva, robótica humanóide, planejamento de movimento, geração de movimento semelhante ao humano, evitação de obstáculos naturalista.

Abstract

As robots are starting to become part of our daily lives, they must be able to cooperate in a natural and efficient manner with humans to be socially accepted. Human-like morphology and movements are often considered key features for intuitive human-robot interactions since they allow human peers to easily predict robotic movements. This PhD project concerns the design and the development of a modular framework that can transfer typical human motor characteristics of planning, learning and controlling upper-limb movements to anthropomorphic robotic arms. The majority of the presented work regards the motion planning module, which is capable of generating collision-free trajectories with human-like kinematic features. More specifically, the planning process takes into account a task-dependent hierarchy of spatial and postural constraints modelled as cost functions and, importantly, addresses obstacles avoidance in a naturalistic manner. Additionally, the introduction of a temporal separation of a particular movement in phases provides a certain degree of human-like accuracy, which may be required in complex manipulative tasks. To quantify human-likeness, smoothness measures and kinematic movement regularities, which are applied in human motor control studies, are here adopted to distinguish between well-coordinated and impaired movements. Tests with two robotic devices in different pick-and-place tasks showed the ability of the proposed planner to generate human-like upper-limb trajectories with a computational cost that is sufficiently small to allow fluent human-robot interactions.

In presence of small expected perturbations of the environment during the execution of similar tasks, learning mechanisms can increase the familiarity with similar sequences of actions by improving the general planning performance. With the purpose of mimicking the human experiential learning cycle, which allows to gain familiarity through experience on the field, an adaptive incremental motion learner is also proposed. After sufficient sessions of training, tests with reaching movements demonstrated that this module is able to speed up the planning process and adapt to consistent perturbations of the scenario.

In order to cope with unexpected disturbances of the workspace, such as the sudden change of the hand target, feed-back signals allows a redefinition of a upper-limb trajectory for the successful accomplishment of the given task. For this reason, the study of feed-back loop controllers is also part of this PhD project

for tracking planned human-like upper-limb movements and promptly react to eventual unforeseen perturbations of the initial workspace. The application of these controllers on different manipulative tasks demonstrated the capability of maintaining the human-like kinematic characteristics of the desired trajectories in absence of significant perturbations. When the workspace is disturbed, a controlled motion can on-line adapt and successfully conclude the given task, but the level of achieved human-likeness can be hardly discussed due to the lack valid metrics of comparison.

The separation and the interconnection of the planning, learning and controlling modules compose a structured framework of human-like motion generation for anthropomorphic manipulators. This division of the different functionalities suggests coordinated directions of investigation towards a more complete integration of human motor skills into human-centred robotic devices.

Keywords: Cognitive robotics, humanoid robotics, motion planning, human-like motion generation, naturalistic obstacle-avoidance.

Contents

List of Figures	xiii
List of Tables	xxii
List of Algorithms	xxiv
I Introduction	1
1 Introduction	2
1.1 Towards the New Robotics	2
1.2 The perception of human-centered robots	3
1.3 Areas of research	6
1.4 The contribution of this PhD project	8
1.5 Thesis outline	10
2 State of the art	11
2.1 Motion planning in robotics	11
2.1.1 Overview on the inverse kinematics problem	12
2.1.2 Sampling-based planners	14
2.1.3 Optimization-based planners	17
2.2 Human-like arm motion generation in robotics	21
2.2.1 Overview of the search results	22
2.2.2 General insights on approaching human-likeness of arm motion	25
2.2.3 From humans to robots	30
2.2.4 Discussion	57
3 Human upper-limb motor behaviour	61

3.1	The posture-based motion planning model	61
3.1.1	Selection of a goal posture	63
3.1.2	Selection of a bounce posture	64
3.1.3	The duration of a movement	66
3.1.4	Kinematical analysis of the motion	67
3.2	Phases in a movement	68
3.3	Human-likeness evaluation of arm movements	72
3.4	Functional models of bimanual coordination	74
4	The hardware setup	76
4.1	The robots	76
4.1.1	ARoS	76
4.1.2	Sawyer	78
4.2	The Barrett Hand	80
4.3	Upper-limb formulation	81
II	Planning	82
5	Human-like upper-limb motion planning	83
5.1	The Human-like Upper-limb Motion Planner: concepts	83
5.1.1	Target posture selection	85
5.1.2	Trajectory selection	88
5.2	The Human-like Upper-limb Motion Planner: trajectory generation in sequential phases	93
5.2.1	Classification and segmentation of a movement	93
5.2.2	Transport phase	95
5.2.3	Approach phase	97
5.2.4	Retreat phase	99
5.3	Time parametrization	100
5.3.1	Phase period	100
5.3.2	Number of steps	101
5.3.3	Time step	101
5.4	The HUMP main algorithms	101
5.5	Adding hand straight-line motion capabilities	105
5.6	Planning human-like synchronous bimanual movements	109

5.6.1	Dual target posture selection	110
5.6.2	Dual bounce posture selection	113
5.7	Upgrading for collaborative robots: modelling the robot Sawyer	116
5.7.1	Modelling the body	117
5.7.2	Modelling the arm	119
6	Experiments and results	122
6.1	Experimental setup	122
6.2	Move movements in an obstacles-free scenario	125
6.2.1	The one-sixth power law fitting	126
6.3	Tests with the robot ARoS in scenarios in presence of obstacles	128
6.3.1	Toy vehicle assembly	128
6.3.2	Serving a drink	133
6.3.3	The user study	142
6.3.4	Naturalistic obstacle avoidance	142
6.3.5	Hand straight-line motion planning with ARoS	148
6.3.6	Human-like synchronous bimanual motion planning with ARoS	153
6.4	Human-like upper-limb motion planning with Sawyer	165
III	Learning	171
7	Adaptive learning of human-like upper-limb postures	172
7.1	Introduction	172
7.2	Warm starting interior-point methods	176
7.3	An adaptive Human-like Arm Posture Learner	179
7.3.1	Problem definition	180
7.3.2	A geometrical situation descriptor	181
7.3.3	Data collecting and updating	181
7.3.4	Locally weighted regression	183
7.3.5	Training	184
7.3.6	Online experiential learning	186
7.4	Experiments and results	187
7.4.1	Software setup	188
7.4.2	Results	189

IV Controlling	196
8 Tracking human-like upper-limb motion in the operational space	197
8.1 Introduction	197
8.2 The desired reference trajectories in the operational space	200
8.3 The controllers	205
8.3.1 Finger motion tracking	207
8.3.2 Null space motion	207
8.3.3 Analysis of the stability	214
8.4 Experiments and results	217
8.4.1 Software setup	217
8.4.2 Results	219
V Conclusions	229
9 Conclusions	230
9.1 Planning	230
9.2 Learning	235
9.3 Controlling	236
Bibliography	239
VI Appendices	256
A Kinematics	257
A.1 Arms direct kinematics of the robot ARoS	257
A.2 Arm direct kinematics of the robot Sawyer	260
A.3 Kinematics of the Barrett Hand	262
A.3.1 Direct kinematics of the fingers	262
A.3.2 Inverse kinematics of the fingers	264
B Modelization of the robot ARoS and of the objects in the workspace	267
C Mathematical formulation of the direct and the back-and-forth movements	271
C.1 Transport direct movements	271

C.2 Back-and-forth movements	273
--	-----

List of Figures

1.1	A picture from Schaal, 2007 illustrating the future involvement of the robots in our society: (a) assistive robots, (b) playmate robots in child education, (c) robots for mentoring and assistance in manipulation tasks, (d) robots that teach movement exercises, (e) personal robots for the elderly, (f) robots for surveillance and protection of children and adults. . . .	4
1.2	The uncanny valley (Duffy, 2003).	5
1.3	Research areas in human-centered robotics (Schaal, 2007).	7
2.1	An example of a PRM roadmap for a point robot in C -space = $\mathbb{R}^2$. Obstacles are in grey and the C_{free} is white. The number of nearest neighbours taken into consideration for connection is 3. Picture taken from Lynch and Park, 2017.	15
2.2	Three frames of the construction of an RRT rooted at the center of the C -space = $\mathbb{R}^2$ in absence of obstacles. Picture taken from LaValle, 1998.	16
2.3	ITOMP: conservative local bounds of a moving obstacle during the interval $I = [t_0, t_1]$ in the period of execution. The collision is anticipated and re-planning the trajectory avoids the dynamic obstacle. Picture taken from Park et al., 2012.	20
2.4	TrajOpt: illustration of a swept volume (Ericson, 2004). The volume covered by an object in two consecutive time steps is considered for continuous-time collision-avoidance. Picture taken from Schulman et al., 2014.	21
2.5	Pie chart showing the percentage of papers related to the databases.	23
2.6	Bar chart showing the distribution of the sources of publication. Numbers indicate the number of papers per each source.	23
2.7	Pie chart showing the types of experimental results	24

2.8	Humanoid robots that have been used to test human-like arm motion generation, namely: <i>ARoS</i> (Silva et al., 2008) (a), <i>iCub</i> (Sandini et al., 2008) (b), <i>ROMEO</i> (Tomić et al., 2018) (c), <i>NAO</i> (Gouaillier et al., 2008) (d), <i>ABB FRIDA</i> (Lamperti et al., 2015) (e), <i>HRP-2</i> (Kaneko et al., 2004) (f), <i>Rollin' Justin</i> (Fuchs et al., 2009) (g), <i>HERB 2.0</i> (Srinivasa et al., 2012) (h), <i>MAHRU</i> (You et al., 2008) (i), and <i>Simon</i> (Gielniak et al., 2013) (j).	24
2.9	A preliminary distinction of the included studies	29
2.10	Bar chart showing in which percentage principles of human-likeness have been used for development and assessment in the included studies.	30
2.11	The equivalence of the human and robot degrees of freedom with the world reference frame on the shoulder. (Artemiadis et al., 2010)	32
2.12	Two postures scored with RULA. (Zacharias et al., 2011)	42
2.13	Classification of the human-like arm movements that are generated by the reviewed motion planners.	44
2.14	Dual-arm manipulation and force control from capturing human motion. (Shin and Kim, 2015)	47
2.15	Concept of the UCM reference feedback controller (Togo et al., 2016).	56
3.1	Overview of the grasping model (Rosenbaum et al., 2001).	63
3.2	Superimposition of angular position and velocity profiles (Rosenbaum et al., 2001). (○) is the direct movement, (□) is the back-and-forth movement and (●) is the composite movement.	64
3.3	The composition of the direct and the bounce movements allows to avoid obstacles and the object to be grasped during prehension (Vaughan et al., 2006).	65
3.4	A simulated reach-and-grasp movement (Rosenbaum et al., 2001).	67
3.5	A simulated reach-and-grasp movement with a low-velocity phase (Rosenbaum et al., 2001).	68
3.6	(a) Power grip between the thumb and the other fingers. (b) Precision grip between the thumb and the index finger (Castiello, 2005).	68
3.7	State transition diagrams of pick-and-place tasks (Sing Bing Kang and Ikeuchi, 1994).	70
3.8	Kinematics of accurate human hand movements.	71
3.9	Motor program and parametric hierarchy of a bi-manual action (Schmidt et al., 1979).	74
4.1	The robot <i>ARoS</i> (Figure 4.1a), its body (Figure 4.1b) and its 7-DOFs anthropomorphic right arm (Figure 4.1c), which is equal to its left arm. The joint angles, the physical parameters and most significant human-like points along an arm are displayed.	77

4.2	The video system of the robot ARoS. A stereo camera system (Figure 4.2a), model STH-DCSG-VAR-C by Videre Design™, which is mounted on a Pan-Tilt unit (Figure 4.2b), model PTU-D46-17 by FLIR Motion Control Systems™. The image processing unit uses this hardware to retrieve type, position and orientation of the objects in the scene (Figure 4.2c).	78
4.3	The robot Sawyer (Figure 4.3a), its pedestal (Figure 4.3b) and its 7-DOFs anthropomorphic manipulator (Figure 4.3c). The joint angles, the physical parameters and most significant human-like points along an arm are displayed.	79
4.4	Parameters, nomenclature and angular displacement ranges of the Barrett Hand	80
5.1	A direct movement from the start to the target posture (5.1a) and a composite movement, which is the superimposition of a direct and a back-and-forth movement (5.1b). A back-and-forth movement is a movement from the start to a bounce posture and back (5.1b), which is selected to obtain a collision-free composite movement. Kinematics of a joint during a movement (5.1c).	84
5.2	Temporal segmentation of the movements.	94
5.3	The body of the robot Sawyer is basically constituted by its pedestal and the dimensions of the head are given by its screen and adjacent components. Figure 5.3a shows the representation of the body with an elliptic cylinder, while Figure 5.3b illustrates the ellipsoid wrapping the head and the pedestal modelled as a super-ellipsoid with shaping parameters $\epsilon_1 = 0.1$ and $\epsilon_2 = 1$	117
5.4	The distance between the center of a sphere and the surface of a super-ellipsoid is illustrated. To prevent collisions, the point E must be outside the super-ellipsoid and its distance from the point P must be greater than the radius of the sphere.	118
5.5	A 7-DOFs manipulator with offsets on the D-H frames 2,4 and 6 can be represented by a maximum of 14 spheres (Figure 5.5a). A 7-DOFs manipulator without these offsets can be represented with a maximum of 8 spheres (Figure 5.5b).	121
5.6	Model of the Sawyer manipulator with the position of the spheres	121
6.1	Sketch of the software architecture. The movement is planned using the target and grip type defined by the Cognitive System (which takes into account the intention of the action), the pose of the objects in the workspace given by the Vision System, and the initial conditions of the robotic arm and hand. Then, the Motion Planner provides a safe trajectory to the hardware for execution.	123

6.2	Sketch of the software modules in the simulated experimental setup. The V-REP environment simulates and provides all the necessary information concerning the humanoid robot, the objects in its workspace and the targets of the movements that are planned. The Motion Manager module elaborates this data and formulates a planning problem according to the user settings. The Motion Planner module runs a planning algorithm selected in a list of different available planners. Given the generated trajectory, the Motion Manager module can execute it with the simulated and the real robot.	125
6.3	3D-movement planned by the HUMP in absence of obstacles (NJS=24.8, NMU=1, solving time=567 ms).	126
6.4	One-sixth power law fitting: one session of obstacles-free move movements planned by the HUMP in the three-dimensional workspace. V denotes the tangential velocity of the hand, C and T the curvature and the torsion of the hand path, respectively. The average NJS of the planned movements is 27.04 with a standard deviation of 7.88, while the NMU is always 1.	127
6.5	Toy vehicle assembly.	129
6.6	Four snapshots that show the sequence of the movements planned by the HUMP in the toy vehicle assembly scenario. The complete video can be seen at the link https://youtu.be/GgQeSUV-6UA	130
6.7	Pick movement planned by the HUMP (NJS=1798.28, NMU=1, solving time = 2823 ms).	130
6.8	Place movement planned by the HUMP (NJS=15702.5, NMU=1, solving time = 2261 ms).	131
6.9	Move movement planned by the HUMP (NJS=18.06, NMU=1, solving time = 847 ms).	132
6.10	The complete task planned by the HUMP.	133
6.11	Serving a drink scenario.	134
6.12	Seven snapshots showing the sequence of the movements planned by the HUMP in the task of serving a drink. The complete video can be seen at the link https://youtu.be/_Zj1hZQrj5c	135
6.13	Pick movement planned by the HUMP in the drink serving task (NJS=6377.39, NMU=1, solving time=1225 ms).	136
6.14	Place movement 1 planned by the HUMP in the drink serving task (NJS=933.52, NMU=1, solving time=848 ms).	137
6.15	Pouring movement planned by the the HUMP in the drink serving task (NJS=5504.86, NMU=2, solving time=1603 ms).	138
6.16	Post-pouring movement planned by the HUMP in the drink serving task (NJS=199.65, NMU=2, solving time=620 ms).	139

6.17	Place movement 2 planned by the HUMP in the drink serving task (NJS=5566.22, NMU=1, solving time=1375 ms).	140
6.18	Move movement planned by the HUMP in the drink serving task (NJS=24.57, NMU=0, solving time=297 ms).	140
6.19	Retreat stage of the place movement 2 in Figure 6.15 plus the move movement in Figure 6.18 planned by the HUMP in the drink serving task (NMU=1).	141
6.20	Complete movement sequence of the drink serving task planned by the HUMP.	142
6.21	Spatial layout of the experimental setup. For details see the text. Obstacle locations O1 and O2 were only applied in combination with target T1, whereas obstacles locations O3 and O4 were combined with target T2.	143
6.22	Configuration T2O4 with a small (T2O4S) and a tall obstacle (T2O4T).	144
6.23	Hand velocity components and hand torsion of the manipulation in configuration T2O4TN.	145
6.24	Three snapshots that show the movement planned by the HUMP in the configuration T2O4TN. The complete video can be seen at the link https://youtu.be/Wlh_QIO-0Xg	145
6.25	Hand velocity components and hand torsion of the manipulation in configuration T2O4TY.	146
6.26	Three snapshots that show the movement planned by the HUMP in the configuration T2O4TY. The complete video can be seen at the link https://youtu.be/Wlh_QIO-0Xg	146
6.27	Hand velocity components and hand torsion of the manipulation in configuration T1O2TY.	146
6.28	Three snapshots that show the movement planned by the HUMP in the configuration T1O2TY. The complete video can be seen at the link https://youtu.be/Wlh_QIO-0Xg	147
6.29	Hand velocity components and hand torsion of the manipulation in configuration T1O1SY.	147
6.30	Pick-and-place from a narrow shelf scenario.	148
6.31	Four snapshots showing, with three different angles, the sequence of the movements planned by the HUMP in the task of picking and placing from a narrow shelf. The complete video can be seen at the link https://youtu.be/1dwj-HP0USI	149
6.32	Pick movement planned by the HUMP (NJS=25732.52, NMU=1, solving time = 8726 ms).	149
6.33	Place movement planned by the HUMP (NJS=14337.64, NMU=1, solving time = 3478 ms).	151
6.34	Move movement planned by the HUMP (NJS=35.31, NMU=1, solving time = 4813 ms).	152
6.35	The complete task planned by the HUMP.	153
6.36	Toy vehicle assembly: scenario for swapping two columns.	154
6.37	Four snapshots that show the sequence of the synchronous bi-manual movements planned by the HUMP in the toy vehicle assembly scenario. The complete video can be seen at the link https://youtu.be/Sj-nXmMg6U4	154

6.38	Synchronous bi-manual pick movement planned by the HUMP (right NJS=3798.96, left NJS=3952.60, Right NMU=1, Left NMU=1, solving time = 18269 ms).	155
6.39	Synchronous bi-manual place movement planned by the HUMP (right NJS=9696.65, left NJS=7217.45, Right NMU=1, Left NMU=1, solving time = 16310 ms).	156
6.40	Synchronous bi-manual move movement planned by the HUMP (right NJS=65.05, left NJS=20.30, Right NMU=1, Left NMU=1, solving time = 9899 ms).	157
6.41	The complete synchronous bi-manual task planned by the HUMP.	158
6.42	Serving a drink: moving a tray with two hands.	159
6.43	Four snapshots that show the sequence of the synchronous bi-manual movements planned by the HUMP in the serving a drink scenario. The complete video can be seen at the link https://youtu.be/Wf19IqvYsCA	159
6.44	Synchronous bi-manual pick movement planned by the HUMP (Right NJS=18974.93, Left NJS=22075.82, Right NMU=1, Left NMU=1, solving time = 14561 ms).	160
6.45	Synchronous bi-manual place movement planned by the HUMP (Right NJS=18933.08, Left NJS=14695.44, Right NMU=1, Left NMU=1, solving time = 6312 ms).	161
6.46	Synchronous bi-manual move movement planned by the HUMP (Right NJS=65.66, Left NJS=157.53, Right NMU=1, Left NMU=2, solving time = 6742 ms).	162
6.47	The complete synchronous bi-manual task planned by the HUMP.	163
6.48	Four snapshots that show the sequence of the synchronous bi-manual movements planned by the HUMP in the toy vehicle assembly scenario. When there is no adaptation to undesired contacts, the successful accomplishment of the task may be undermined. The complete video can be seen at the link https://youtu.be/ZgnVPnBzKoA	164
6.49	Four snapshots that show the sequence of the synchronous bi-manual movements planned by the HUMP in the toy vehicle assembly scenario. Misalignments of the base show the necessity of having an on-line adaptation mechanism. The complete video can be seen at the link https://youtu.be/w3J_F0J2LBg	165
6.50	The toy vehicle assembly scenario with the robot Sawyer.	166
6.51	Four snapshots that show the sequence of the movements planned by the HUMP for the simulated robot Sawyer in the toy vehicle assembly scenario. The complete video can be seen at the link https://youtu.be/QpEUiGuBpXU	167
6.52	Pick movement planned by the HUMP for Sawyer (NJS=1448.36, NMU=1, solving time = 7229 ms).	167
6.53	Place movement planned by the HUMP for Sawyer (NJS=2502.39, NMU=1, solving time = 3272 ms).	168

6.54	Move movement planned by the HUMP for Sawyer (NJS=22.89, NMU=1, solving time = 4120 ms).	168
6.55	The complete task planned by the HUMP for Sawyer.	169
6.56	Four snapshots of the sequence of the movements planned by the HUMP for the real robot Sawyer equipped with a gripper in the toy vehicle assembly scenario. The complete video can be seen at the link https://youtu.be/wJWujc-0I-Y	169
7.1	The experiential learning cycle (Kolb, 1984).	173
7.2	Illustration of trajectory prediction for planning in perturbed situations (Jetchev and Toussaint, 2013).	175
7.3	First derivative of the kernel computed at the mean distance $\bar{d}$ versus r . With values of r close to ± 0.6 , the slope of the Gaussian kernel is near its maximum negative value. . . .	184
7.4	A cyclic session of online experiential learning.	186
7.5	Illustration of one example of the considered reaching task. The manipulator has to reach the green target while avoiding collisions with the red obstacle. The proposed adaptive learning technique predicts efficient initial guesses for the selection of a human-like target configuration of the robotic arm.	187
7.6	Sketch of the software setup that includes online cyclic experiential learning. The Motion Manager module collects and updates the datasets necessary to build a concrete experience. The Motion Learner module applies memory-based learning on these experience to predict efficient initializations of the Motion Planner module. When trajectories are successfully planned, pairs of situations and solutions are added to the present experience and a new learning cycle can start.	189
7.7	Variation of the size of the datasets D_{cold} and D_{warm} with new incoming situations of the environment.	190
7.8	Un-regularized loss values (Equation 7.3.17) of the LWR_{Eucl} and of the LWR_{Opt} predictors on 10 sessions of on-line learning. Each session is composed by 10 new situations that update the datasets before the training starts.	190
7.9	The most 15 significant learned features for the task in Figure 7.5. Incremental feature extraction over 10 sessions of online experiential learning with 10 new situations per session. In superscript the component of the element is written: $x = \hat{x}$ axis, $y = \hat{y}$ axis, $z = \hat{z}$ axis, d is the L_2 norm, $r = roll$ angle, $p = pitch$ angle and $y = yaw$ angle.	192

7.10	Un-regularized loss values (Equation 7.3.17) of the LWR_{Eucl} and of the LWR_{Opt} predictors on 10 sessions of on-line learning. Each session is composed by 10 new situations that update the datasets before the training starts. The new situations differ from the others only for the target and the obstacle poses. The initial configuration of the robot remains constant. . . .	193
7.11	Illustration of one example of the considered reaching task. The manipulator has to reach the green target while avoiding collisions with the red obstacle. The main differences with the reaching task in Figure 7.5 refer to the initial configuration of the manipulator and to position and the orientation of the target.	195
8.1	Picture from Burdet et al., 2013 illustrating the schematic of human motor control.	198
8.2	Diagram of the feedback loop control in the operational space of the manipulator.	206
8.3	Diagram of the feedback loop control in the joints space of the robotic hand.	208
8.4	Distance between an obstacle and a point on the robotic arm. The point $P_d = P + D$ defines the distances $d = \ D\ $. Only the minimum distance is considered in the online obstacle avoidance mechanism.	210
8.5	Definition of the swivel angle α : the swivel angle is that angle between the reference plane (the sagittal plane) and the arm plane of motion, which contains the shoulder, the elbow and the wrist points of the manipulator.	212
8.6	Diagram of the feedback loop control for tracking the swivel angle in the null space of manipulation $\mathcal{N}(J)$. The angular accelerations $\ddot{\theta}_{op}$ are computed in the range space of the Jacobian $\mathcal{R}(J)$ by the controller illustrated in Figure 8.2. The angular velocities $\dot{\theta}_{null}$ are computed in the null space $\mathcal{N}(J)$ as formulated in Equation 8.3.19.	213
8.7	Scenarios with the robot ARoS performing two tasks for testing the proposed trajectory-tracking method. Tests with the real robotic system in these scenarios can be seen at link https://youtu.be/8BJK-qCiews	218
8.8	Sketch of the software setup that also includes feedback loop controlling. The Motion Controller module implements the proposed controllers and it is integrated into the Motion Manager as a separated thread, which is constantly updated with the latest environmental changes.	219
8.9	Qualitative comparison between the planned and the actual hand velocities in absence of perturbations in the workspace of the robot.	221
8.10	Errors during the unperturbed tracking of a human-like pick movement in the scenario in Figure 8.7a.	222
8.11	Desired and actual trajectories in the scenario in Figure 8.7a.	223
8.12	Effects of swivel angle tracking on the joints-space trajectory.	224

8.13	Three snapshots of picking the red column under perturbations. The red column changes position and orientation at $t = 2.5s$ and returns to its initial pose at $t = 4s$. Compared to the planned trajectory, the actual hand velocity shows more peaks in the <i>Transport</i> phase of the pick movement, which are due to the perturbations on the hand target.	226
8.14	Errors during the perturbed tracking of a human-like pick movement (Figure 8.13).	227
8.15	Desired and actual trajectories during the perturbations shown in Figures 8.13a,8.13b and 8.13c.	228
A.1	Right arm in home posture and reference frames assigned according to the Denavit-Hartenberg convention.	258
A.2	Left arm in home posture and reference frames assigned according to the Denavit-Hartenberg convention.	258
A.3	The Sawyer arm in home posture and reference frames assigned according to the Denavit-Hartenberg convention.	261
A.4	Joint angles and reference frames on the finger F_3 are displayed as an example of direct kinematics on the Barrett Hand.	263
A.5	Joint angles and distances d_{FF} and d_{FH} are displayed for the inverse kinematics of the Barrett Hand.	265
B.1	The right robotic arm (Figure B.1a) and the three fingers of robotic hand (Figure B.1b) are modelled as a set of spheres. The body of the robot ARoS is modelled as an elliptic cylinder (Figure B.1c).	268
B.2	The objects in the workspace are modelled as ellipsoids when they are considered as obstacles (Figure B.2a). When an object is being manipulated by the robot, it is modelled as a set of spheres (Figure B.2b).	269

List of Tables

4.1	Arm physical limits of the robot ARoS	77
4.2	Arm dimensions of the robot ARoS	77
4.3	Arm physical limits of the robot Sawyer	78
4.4	Arm dimensions of the robot Sawyer	78
4.5	Physical limits of the Barrett Hand	80
4.6	Dimensions of the Barrett Hand	80
6.1	Descriptive statistics of the six sessions, 95% confidence interval (CI) of the slope in 1/6-PL fitting.	127
6.2	Results over 100 trials of the pick movement planning problem in the toy vehicle scenario (Figure 6.6).	131
6.3	Results over 100 trials of the place movement planning problem in the toy vehicle scenario (Figure 6.6).	132
6.4	Results over 100 trials of the move movement planning problem in the toy vehicle scenario (Figure 6.6).	133
6.5	Results over 100 trials of the pick movement in the drink serving scenario (Figure 6.12).	136
6.6	Results over 100 trials of the place movement 1 in the drink serving scenario (Figure 6.12).	137
6.7	Results over 100 trials of the pouring movement in the drink serving scenario (Figure 6.12).	138
6.8	Results over 100 trials of the post-pouring movement in the drink serving scenario (Figure 6.12).	139
6.9	Results over 100 trials of the place movement 2 in the drink serving scenario (Figure 6.12).	139
6.10	Results over 100 trials of the move movement planning in the drink serving scenario (Figure 6.12).	140
6.11	Description of the symbols used in the experiments.	144
6.12	Results over 100 trials of the pick movement planning problem in the pick-and-place from a narrow shelf scenario (Figure 6.31).	150

6.13	Results over 100 trials of the place movement planning problem in the pick-and-place from a narrow shelf scenario (Figure 6.31).	151
6.14	Results over 100 trials of the move movement planning problem in the pick-and-place from a narrow shelf scenario (Figure 6.31).	152
7.1	Performance of predicting initializations to the optimization problem (5.1.4)–(5.1.7) on 500 situations uniformly samples from the task in Figure 7.5. The optimizer IPOPT is set to work in warm-start mode with a maximum number of iterations of 5000.	191
7.2	Performance of predicting initializations to the optimization problem (5.1.4)–(5.1.7) on 500 situations uniformly samples from the task in Figure 7.5. The initial configuration of the robot remains constant on the different situations. The optimizer IPOPT is set to work in warm-start mode with a maximum number of iterations of 5000.	194
7.3	Performance of predicting initializations to the optimization problem (5.1.4)–(5.1.7) on 500 situations uniformly samples from the task in Figure 7.11. The optimizer IPOPT is set to work in warm-start mode with a maximum number of iterations of 5000.	195
A.1	D-H parameters of the right arm	257
A.2	D-H parameters of the left arm	257
A.3	D-H parameters of the Sawyer arm	261
A.4	D-H parameters of the k^{th} finger	263
B.1	Positions of the points on the right arm and hand	269
B.2	Positions of the points on the left arm and hand	269

List of Algorithms

5.1	The HUMP main algorithm – Part 1	102
5.2	The HUMP main algorithm – Part 2	103
5.3	The HUMP approach phase algorithm	104
5.4	The HUMP retreat phase algorithm	105
5.5	The HUMP approach phase algorithm with straight-line motion	107
5.6	The HUMP retreat phase algorithm with straight-line motion	108
5.7	Modelling a manipulator with a set of spheres	120
A.1	Inverse kinematics of the Barrett hand	266
B.1	Generation of a set of spheres in a given object	270

Part I

Introduction

Chapter 1

Introduction

In this introductory chapter, the evolution of robotics in the last decade is briefly described in section 1.1 to outline the principal motivations behind this PhD project. Importantly, for a meaningful research guidance, emphasis is put on understanding how human-centred robotic devices are perceived by human interacting partners and the most recent corresponding user studies are delineated in section 1.2. The corresponding area of research is presented in section 1.3 and the main contributions of this project are outlined in section 1.4. Finally, this chapter ends with section 1.5 that summarizes how this dissertation is organized.

1.1 Towards the New Robotics

Since the 1950s, robotics has evolved in different manners with respect to the new fields of applicability and the level of integration in our society. The technological advances and the increasing necessities of the society have gradually led robotics from typical industrial to human-centered applications (Fukuda et al., 2001; Garcia et al., 2007; Gladden, 2019; Schaal, 2007; Sheridan, 2016). Particularly, in Fukuda et al., 2001, the authors identified three generations of robots that have featured automation and research in the recent past. The first generation was formed by robots without any computational or sensing capabilities, the second generation concerned devices with a limited computational power and the third generation includes the robots with artificial intelligent and diverse sensing capabilities. What started the growth of autonomous devices in our society was the success that industrial robots had in specific manufacturing sectors (e.g., in the automotive industry). This inaugurated new and untraditional fields in robotics research, such as medical robotics for the design of prosthetics devices, mobile wheeled or legged robots for transportation and micro-robotics (Fukuda et al., 2001). The industrial robots have been introduced in the productive chain in the 1960s and robotics research was only focused on them until the 1990s. More recently, the demand of the society to have devices to serve and assist human beings promoted

the arising of the service and personal robotics field (Garcia et al., 2007). This New Robotics, as it was called by Schaal, 2007, aims at realizing general purpose and intelligent devices instead of special purpose robots designed for particular tasks. The objectives of the New Robotics have been carried on by universities and research institutes towards the development of human-like robots, which will increasingly be present among us to take over unpleasant tasks and assist humans in unstructured environments (Schaal, 2007). The growing number of general artificially intelligent systems in our society has also forced the human-factors community to investigate the new role of humans interacting with robots.

New questions inevitably arise during the design of novel Human-Robot Interaction (HRI) or Collaboration (HCI) paradigms and often concern the authority and the autonomy level of these new intelligent devices (Sheridan, 2016). Ethical and anthropological issues have been recently considered by the government of Japan with the initiative called Society 5.0 (Gladden, 2019). The latter is expected to be a technologically post-humanized society where humans and robots coexist in the same environment and work to improve the quality of life by offering customized services to cope with various needs. The new social paradigm differs from the current Society 4.0 by the fact that robots will not only be passive tools as they are nowadays. On the contrary, they are expected to be active agents capable of proactive data collecting, making decisions and friendly behaving to maintain human beings at the core of the society, but not alone within it. Due to the process of post-humanization that is taking place, this new anticipated society is significantly dependent on the emerging transformative technologies that will impact the interactions among individuals as it has never happened so far. For example, a humanoid robot can be considered a personal helper designed to respond to the needs of human beings. Such an *artificial man* (Fukuda et al., 2001) could carry and manipulate objects for people with disabilities, could replace some cognitive functions or take part in the educations of children (see Figure 1.1). A robot with these features is human-centered because it is meant to interact, collaborate and work in unstructured environments with human beings. For this reason, research in robotics and artificial intelligence needs to be necessarily enriched by different scientific disciplines such as ethics, psychology, anthropology and neuroscience to simulate or mimic life-like activities and appearance (Fong et al., 2003; Gladden, 2019; Schaal, 2007; Wiese et al., 2017). These new abilities in robots significantly influence how they are perceived by interacting human partners.

1.2 The perception of human-centered robots

The increasing demand of human-robot interactions have also guided researchers through the study of particular psychological phenomena that occur in human interacting partners. For example, the natural tendency of human beings towards anthropomorphism has a substantial impact on designing human-centered



Figure 1.1: A picture from Schaal, 2007 illustrating the future involvement of the robots in our society: (a) assistive robots, (b) playmate robots in child education, (c) robots for mentoring and assistance in manipulation tasks, (d) robots that teach movement exercises, (e) personal robots for the elderly, (f) robots for surveillance and protection of children and adults.

devices. Anthropomorphism can be defined as the attribution of human characteristics to inanimate objects, animals and any other non-human agent in the attempt of rationalising their behaviours in given social contexts (Duffy, 2003). This cognitive process goes through the projection of natural intelligence on devices with capabilities and aspect accurately designed to be more familiar and facilitate interactions. For instance, such features often replicate voice, speech, visual and motion skills that are typical in humans. However, social robotics should embrace anthropomorphism by exploiting people's expectations of behaviours rather than necessarily trying to force to believe that a robot has real human capabilities (Duffy, 2003). With this knowledge, roboticists search for applications of optimal anthropomorphism to balance what people expect from robots and the humanization of technological aspects and capacities. Otherwise, the risk for designers is to encounter the uncanny valley phenomenon that is illustrated in Figure 1.2.

Figure 1.2 shows that the more closely a robot resembles the human being, the more it will increase its acceptance in our society until its appearance becomes so weird that it raises uncanny feelings (the uncanny valley). For this reason, designers of human-centered robots should adopt the minimal engineering solutions to realize artificial agents capable of social interactions (Strait et al., 2017). The uncanny valley phenomenon demonstrates that incorporating high level humanized features in robots evokes behavioural

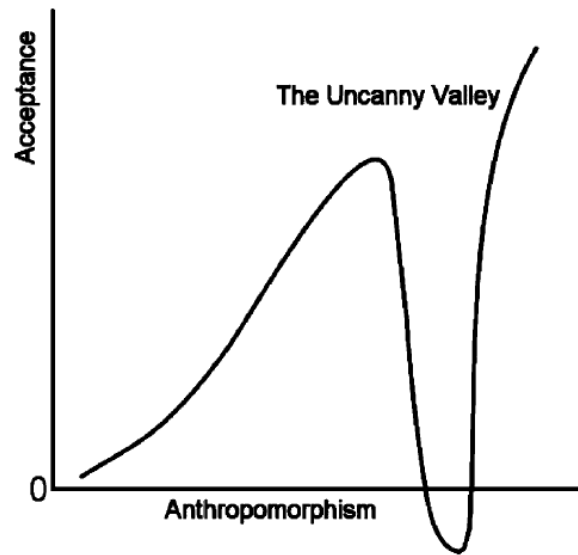


Figure 1.2: The uncanny valley (Duffy, 2003).

expectations that may not be respected. This cognitive process is also called the "violation of expectation hypothesis" and it usually occurs when assumptions on anthropomorphic robotic systems are not maintained (Wang et al., 2015). Wang et al. (2015) also introduced a "dehumanization hypothesis" that is not necessarily in contrast with other theories behind the presence of the uncanny valley. According to this hypothesis, weird feelings may arise when people dehumanize anthropomorphized agents. Any feature of human replicas, like voice or motion, may reveal a mechanistic nature that leads to dehumanization and consequently to uncanny impressions. In other words, a dehumanized human replica becomes a "robot-like human" who lacks humanness and looks bizarre to human observers (Wang et al., 2015).

In order to understand the real implications of the uncanny valley on possible humanoid agents, Strait et al. (2017) implemented precise methodological practices for the assessment of people's aversion to social robots. In specific, naïve subjects were given 60 pictures of robots with different levels of typicality to measure their discomfort generated after observation. Results confirmed the existence of the uncanny valley and demonstrated that high human-like agents generate more discomfort than prototypical robots. Therefore, the design of human-centered robots has to be focused on prototypical devices that increase people's comfort and willingness of interaction.

In general, studies have revealed a very positive attitude towards robots and the idea of being surrounded by them in different personal and societal contexts (Ray et al., 2008; Welfare et al., 2019). People commonly expect from robots a very pragmatic daily help in domestic, entertaining and educational applications. Moreover, having intelligent devices that perform repetitive or dangerous tasks is desirable, because safety and monotonousness seem to be mostly important issues in industrial settings. Particularly,

robotics in the upcoming industry 5.0 is expected to augment the capabilities of human workers instead of replacing them. The current industry 4.0 has been featured by robots designed for a complete automation of work flows, while the industry 5.0 will include robots intended to satisfy the necessities of human co-workers for the augmentation of productivity and efficiency of the entire company (Welfare et al., 2019). Therefore, in this new human-centered robotics, augmentation will gradually replace automation with the introduction of novel artificially intelligent devices that enhance collaboration in shared workspaces. For instance, the fundamental principles of motor interpretation behind successful human-human joint actions can be used when adopting motion planning strategies (Bicho et al., 2011a; Bicho et al., 2011b; Glasauer et al., 2010; Sebanz et al., 2006). Koppenborg et al. (2017) showed that the predictability and the velocity of the motion in robotic manipulators significantly influence the performance of the collaboration with human partners. High speed movements specifically increased anxiety and risk perception of human co-workers, which, consequently, act unpleasantly and inefficiently. Therefore, time parametrization also plays an essential role in any motion planning process that is meant for human-robot interaction. Body motion in robotics has also been recently considered a language to communicate emotional states of social robots (Tanizaki et al., 2018). This study revealed that social robots for educational support had more significant impressions on learners with body language than without it. However, movements of robots without a social role are also interpreted as social cues by human observers (Erel et al., 2019). Therefore, there is an automatic and implicit cognitive process for the attribution of mental states in robots to anticipate their behaviours, which is independent on their assigned purposes. For these reasons, designers of human-centered robots and roboticists of different areas of research cannot skip accurate strategies of motion planning to obtain socially acceptable interactions with humans.

1.3 Areas of research

Schaal (2007) defined three main research areas in the New Robotics, which are illustrated in Figure 1.3. The *action*, *interaction* and *engagement* research areas are significantly human-oriented to produce positive experiences in human users/partners. The *action* category concerns the studies of motor control modelling and execution for humans and robots. The *interaction* area is the basic research to enhance human-robot and robot-robot interactions for the accomplishment of a common task. The *engagement* area regards the study of the psychological and cognitive aspects that are considered when robots are operating with humans (Schaal, 2007).

Interaction and *engagement* with robots have been often investigated in the attempt of replicating communication mechanisms that are typical among human beings (McColl et al., 2016; Rossi et al., 2017; Wiese et al., 2017). Interaction and cooperation of human partners are encouraged with robots that

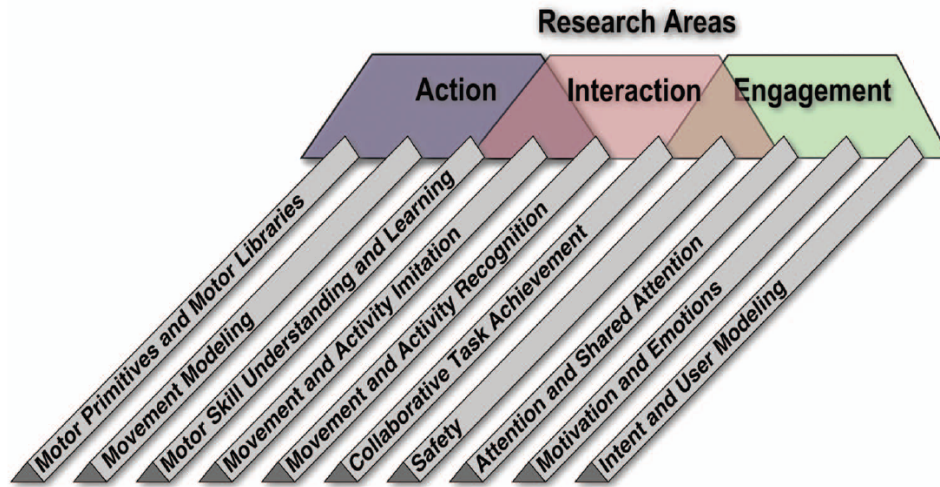


Figure 1.3: Research areas in human-centered robotics (Schaal, 2007).

can use communication modalities that are natural among human beings. Such devices are called *social robots* because they are designed for a successful integration in particular societal contexts (McColl et al., 2016; Rossi et al., 2017). This integration might start with the detection of human affects, which identify the complex combination of emotions, moods and attitude that feature each person (McColl et al., 2016). This automatic detection uses the different signals of humans (e.g., facial expressions, body language, etc.) to identify models representing, for instance, happy or sad affects. Robots capable of recognizing affects can respond and adapt their actions accordingly. The benefits are mutual because engagement of human partners and the performance of the robot during the execution of collaborative tasks increase significantly (McColl et al., 2016). However, automatic human affects detection can be seen as an element that concerns sociality among other user profiling techniques (Rossi et al., 2017). With the term profiling, researcher indicate endowing robots with the capability of having a human-oriented perception of their intentions, motor capabilities and non-verbal communication. Then, such robots can adapt their behaviours on physical, cognitive and social levels in a way that is customized for particular human partners who are individualized as long-term companions.

The social relevance and integration of robots are intensified when they are perceived as intentional agents like human beings are (Wiese et al., 2017). Action understanding and inference about internal states are phenomena that activate determined areas of the brain during interactions among humans. For this reason, Wiese et al. (2017) suggested a closer collaboration between neuroscientists and roboticists for the design of robots with advanced cognitive, motor and reasoning skills capable of activating those brain areas on human partners. In particular, action understanding is triggered when motion of a robot is biologically inspired due to its ability to express intentionality and to maintain non-verbal social connection with human observers (Wiese et al., 2017).

The *action* research area has recently progressed towards the removal of fences and of any kind of separation between humans and robots involved in joint tasks (Billard and Kragic, 2019; De Momi et al., 2016; Dragan and Srinivasa, 2014; Kanazawa et al., 2019). In the last decades, manipulation in robotics has reached complex levels of dexterity and soft skills that, however, are not sufficiently advanced to cope with any necessities in real-world collaborations with human co-workers (Billard and Kragic, 2019). The minimization of the risks and the mitigation of eventual damages to humans are the priorities for any robotic application and especially during coordinated manipulation tasks. High quality of perception ensures the localization of the objects during pre-shaping, while planning and control are necessary to guarantee a stable grasp that should be selected according to specific intentions. Instead of control approaches, learning dexterous manipulation from human data is often applied to replicate human skills and adapt to unfamiliar objects. However many learning techniques require a significant amount of data for training, which are often collected with laborious trial-and-error experiments (Billard and Kragic, 2019).

Like in human beings, the quality of manipulation is linked to the generation of arm motion that has recently guided researchers to user-oriented solutions. For instance, motion planning in manipulators can optimize human-robot collaboration for an efficient and safe sharing of the same workspace with a human co-worker (Kanazawa et al., 2019). By learning regularities of human motion, the generation of a trajectory becomes human-aware to avoid waste of time and increase safety during the accomplishment of a collaborative task. Dragan and Srinivasa (2014) introduced the notions of legibility and predictability of robotic arm movements to address fluent human-robot interactions driven by the perception of intentionality. Specifically, legibility is the property of a movement to be readable on its final goal: it is crucial in a workspace shared with humans and in presence of multiple targets. Predictability features a movement with the ability of being anticipated given a final goal: it is fundamental when both the interacting partners are considering a single target. De Momi et al. (2016) reproduced human-like arm motion from a database of human collected data to enhance collaboration with humans in hand-over tasks. Results demonstrated that biological kinematic regularities were respected and that the proposed scheme was qualitatively assessed by human naïve observers. The project of this dissertation is also part of the *action* area of research and also aims at transferring the biological kinematic regularities of humans into humanoid robots with the ultimate goal of obtaining efficient human-robot interactions.

1.4 The contribution of this PhD project

Many studies have demonstrated that human subjects are very sensible and influenced during the observation of biologically-inspired or human-like movements generated by humanoid agents, independently to

their human or robotic nature (Chaminade et al., 2005; Hugues et al., 2016; Kupferberg et al., 2011). Additionally, it has recently been proven that the application of natural human movement facilitates physical human-robot interaction (Maurice et al., 2018; Noohi et al., 2016).

Human-centred robots will operate autonomously in partially unknown or changing environments, like any living being does. Therefore, learning and controlling capabilities must be designed and implemented to adapt and respond to dynamic situations (Floreano et al., 2014; Schaal, 2007). Specifically, learning techniques can be used to simulate the predictive component of motor planning in the central nervous system, while feedback control can be used to simulate the reflex action in volitional movements for the accomplishment of an intended action (Schwartz, 2016).

The studies that compose the present project are part of the *action* research area showed in Figure 1.3. The main contribution is threefold: planning, learning and controlling human-like upper-limb motion in robotics. Each area of development is motivated by the fact that they can be identified in the general schematic of human motor control. This means that robotics can take advantage of this knowledge by applying human motor behaviours (algorithms) to anthropomorphic devices. This synthesis at a system level has been called *human robotics* by Burdet et al. (2013).

The main objective is the generation of smooth, fluent and collision-free movements for anthropomorphic robotic arms that captures important characteristics of human upper-limb motions. Most importantly, the generated movements must allow a human partner to interpret the robot's underlying motor intention. With this purpose, a goal-directed movement planner is firstly developed. It takes inspiration from observed regularities in studies of human upper-limb movements and on corresponding models that have been proposed (Burdet and Milner, 1998; Burdet et al., 2013; Meulenbroek et al., 2001; Milner, 1992; Rosenbaum et al., 1999; Rosenbaum et al., 2001; Rosenbaum et al., 2006; Rosenbaum et al., 2009; Vaughan et al., 2006). This motion planner is part of the feed-forward control policy that is characterized by the ability of prediction when adapting to known situations. This specific capability is also part of this project and it is simulated by a learner module, which allows a robot to predict the trajectory of a movement (Schaal, 2007; Schwartz, 2016).

In order to cope with unpredictable perturbations that may occur in dynamic environments, the third contribution is the realization of a feed-back loop controller, which endows a robot with the capability of avoiding moving obstacles and following moving targets while maintaining the human-like characteristics of the motion (Floreano et al., 2014; Schaal, 2007; Schwartz, 2016).

1.5 Thesis outline

This dissertation is organized by the following Parts.

Part I introduces the context of investigation wherein this PhD project is developed and is formed by the following chapters:

- chapter 2 describes a review on the most recent techniques of motion planning in robotics with particular attention to human-like arm and hand motion generation;
- in chapter 3 the principles of human upper-limb motor behaviour that inspired this project are collected and briefly discussed to understand the psychological motivations behind the developed techniques;
- in chapter 4 the robotic devices that were used for experimentation are shortly introduced.

In Part II, the planning module is introduced in two chapters:

- chapter 5 proposes a novel motion planner for the generation of human-like collisions-free upper-limb movements for applications in manipulative tasks;
- in chapter 4 the general software setup is described and the results obtained from testing the proposed motion planner are collected and discussed.

Part III is composed by chapter 7 that introduces a module for posture learning for assisting of the planning module in presence of repetitive similar situations that the robot may encounter. In Part IV, chapter 8 presents the design of a trajectory tracking system capable of replicating given human-like kinematic features and of adapting the motor behaviour when perturbations of the workspace occur. Finally, Part V introduces chapter 9 that separately draws conclusions and provides directions of future investigations regarding the proposed planning, learning and controlling techniques.

Chapter 2

State of the art

This chapter provides a comprehensive state of the art concerning global methods for motion planning with particular interest in the generation of human-like movements. In section 2.1, some of the most popular motion planners in robotics are described to provide an exhaustive overview about planning for robotic manipulators. In section 2.2, a systematic literature review concerning human-like arm motion generation in robotics is presented. Here, the selected publications are analysed according to different aspects: source of publication, experimental platforms, basic principles and features of the planning method and metrics used to assess human-likeness of the generated motion. The chapter ends with a discussion concerning the limitations of the current literature and the directions that this project takes to overcome them.

2.1 Motion planning in robotics

In robotics, the motion planning problem consists in finding a *path* with a proper time parametrization to connect two points in the configuration space of a robot (C -space). In particular, a feasible *path* belongs entirely to the free C -space (C_{free}), which is the set of valid configurations that do not collide with obstacles in the workspace and do not exceed the joint limits of the robot (Lynch and Park, 2017). For example, a configuration space of a robotic manipulator with D degrees of freedom (DOF) can be its joint space and a configuration of its joints $\theta \in C_{free} \subset \mathbb{R}^D$ represents a point in this space. A *path* describes the geometry of a motion from an initial configuration θ_0 to a goal configuration θ_g , which serves the objective of the planning problem (e.g., grasping an object). Often, the goal configuration θ_g is provided by analytical or numerical inverse kinematics (IK) procedures, which take place before the generation of a valid *path* (see section 2.1.1). Formally, a *path* $\theta(s)$ maps a parameter $s \in [0, 1]$ into the space of robot's configuration $\Theta \subseteq C_{free}$ (Equation 2.1.1).

$$\theta(s) : [0, 1] \rightarrow \Theta \quad (2.1.1)$$

In other words, while the s parameter increases from 0 to 1, the robot moves along the path $\theta(s)$ from the start configuration θ_0 to the goal θ_g in N steps that discretize the interval $[0, 1]$.

Scaling with respect to time $t \in [0, T]$ the parameter $s(t)$, the duration of the motion T is defined for the path $\theta(s)$. Formally, the parameter s is scaled to map time in the interval $[0, T]$ into the interval $[0, 1]$ (Equation 2.1.2).

$$s(t) : [0, T] \rightarrow [0, 1] \quad (2.1.2)$$

A path $\theta(s)$ (Equation 2.1.1) with a time parametrization $s(t)$ (Equation 2.1.2) defines a trajectory $\theta(s(t)) = \theta(t)$ in Equation 2.1.3.

$$\theta(t) : [0, T] \rightarrow \Theta \quad (2.1.3)$$

A trajectory $\theta(t)$ provides information about velocities and accelerations that well define a *state* of a robot $x(t) = (\theta(t), \dot{\theta}(t), \ddot{\theta}(t)) \forall t \in [0, T]$, if $\theta(s)$ and $s(t)$ are at least twice differentiable (Lynch and Park, 2017).

Several methods for solving the inverse kinematics and the motion planning problems are already implemented and available for the robotics community (e.g., the MoveIt! library¹). Time parametrizations can be different for a given path and are often computed independently to the spatial description (Lynch and Park, 2017). In section 2.1.1, an overview of the most recent commonly used inverse kinematics solvers is provided, while sections 2.1.2 and 2.1.3 describe two categories of path planners that have been widely used in many robotics applications: sampling-based and optimization-based planners.

2.1.1 Overview on the inverse kinematics problem

For a generic open chain with forward kinematics $T(\theta)$, the inverse kinematics problem can be formulated as finding a homogeneous transformation matrix $X \in SE(3)$ (Equation 2.1.4), such that $X = T(\theta)$ (Lynch and Park, 2017).

$$SE(3) = \left\{ \begin{bmatrix} R & p \\ 0 & 0 & 0 & 1 \end{bmatrix} \mid p \in \mathbb{R}^3, R \in \mathbb{R}^{3 \times 3}, R^T R = I, |R| = 1 \right\} \quad (2.1.4)$$

In general, analytical inverse kinematics can be applied when there is only one solution of the inverse kinematics problem. For instance, the configurations of a manipulator with 6 degrees of freedom (DOF), which correspond to a given pose (position and orientation) of its end-effector in a three-dimensional workspace, can be computed analytically. However, numerical inverse kinematics is necessary when there is not an analytical solution to the inverse kinematics problem. Moreover, when an analytical solution exists, numerical procedures improve accuracy and overcome issues related to imprecise modellizations of the

¹<http://moveit.ros.org/>

physical robot. The numerical approach is based on the Newton-Raphson method that is widely used for non-linear root-finding and proceeds iteratively as it is described by Equation 2.1.5 (Lynch and Park, 2017).

$$\boldsymbol{\theta}^{k+1} = \boldsymbol{\theta}^k + \mathbf{J}^{-1}(\boldsymbol{\theta}^k) \cdot (\mathbf{x}_d - \mathbf{f}(\boldsymbol{\theta}^k)) \quad (2.1.5)$$

$\boldsymbol{\theta}^k, \boldsymbol{\theta}^{k+1} \in \mathbb{R}^D$ are the configurations of the manipulator at the iterations k and $k + 1$, respectively. The initial guess $\boldsymbol{\theta}^0$ is provided to start the algorithm (it is usually the analytical solution when it exists). $\mathbf{x}_d \in \mathbb{R}^M$ is the desired pose of the end-effector, $\mathbf{f}(\boldsymbol{\theta}) : \mathbb{R}^D \rightarrow \mathbb{R}^M$ is the pose of the end-effector and $\mathbf{J}(\boldsymbol{\theta}) \in \mathbb{R}^{M \times D}$ is the Jacobian matrix of the manipulator evaluated at the configuration $\boldsymbol{\theta}$. The algorithm stops and a solution is returned when the error $\|\mathbf{x}_d - \mathbf{f}(\boldsymbol{\theta})\|$ becomes smaller than a given threshold. However, if the Jacobian matrix $\mathbf{J}$ is either not square or in a singular configuration, becomes not invertible. In this case, $\mathbf{J}^{-1}$ in Equation 2.1.5 is replaced by the Moore-Penrose pseudo-inverse $\mathbf{J}^+$ that is given in Equation 2.1.6 when $\mathbf{J}$ is full rank (Lynch and Park, 2017).

$$\mathbf{J}^+ = \begin{cases} \mathbf{J}^\top (\mathbf{J}\mathbf{J}^\top)^{-1}, & D > M \text{ (right inverse)} \\ (\mathbf{J}^\top \mathbf{J})^{-1} \mathbf{J}^\top, & D < M \text{ (left inverse)} \end{cases} \quad (2.1.6)$$

In the robotics community, numerical IK algorithms based on the Newton-Raphson method are commonly found implemented in the Orocos Kinematics and Dynamics Library (KDL)². The functionalities of this library are widely used in different ROS packages (Quigley et al., 2009) and by the MoveIt! library for motion planning¹. The simplicity of these IK solvers is behind their popularity, but it is also the reason of well known issues concerning, for example, the convergence failures in presence of joint limits, the inactivity of the algorithm when the search cannot overcome local minima and poor support for tolerances in Cartesian poses (Beeson and Ames, 2015). In the attempt of mitigating such issues, different alternatives have been proposed and tested with statistically valid methods. For instance, Beeson and Ames (2015) provided quantitative comparisons of some of these methods and introduced a more efficient approach to solve the inverse kinematics problem. A simple enhancement of the basic inverse Jacobian solver can be achieved by restarting the algorithm with a random seed when a local minimum is detected. The resulting KDL algorithm with random restarts (KDL-RR) also minimizes the function $\|\mathbf{x}_d - \mathbf{f}(\boldsymbol{\theta})\|$, but it can start again with another guess if insufficient progresses occur in too many iterations.

In many practical application, joint limits have to be respected to avoid infeasible configurations and damages. However, it is known that the KDL IK solvers suffer of a high rate of convergence failure in presence of joint limits, because the inverse Jacobian updating rule can push a joint towards its limit in the attempt of reducing the error in the operational space of the manipulator (Beeson and Ames, 2015). Being strongly non-linear, the IK problem is expected to be efficiently solved by sequential quadratic programming (SQP)

²<https://orocos.org/>

that can better handle hard constraints such as joint limits. The SQP problem can be approached by optimizers with search methods that efficiently handle non-smooth search spaces such as the configuration space of a manipulator with joint limits. For instance, the Broyden-Fletcher-Goldfarb-Shanno (BFGS) iterative algorithm is one of these methods that can perform better by approximating the Hessian of the Lagrangian function in the optimization problem (Nocedal and Wright, 1999).

In Beeson and Ames, 2015, variants of the SQP method were compared in the resolution of the IK problem to assess the quality of different objective functions. While a simple and intuitive minimization in the joint space was firstly considered, different formulations of the error in the operational space demonstrated to reach superior results. For example, dual-quaternions (Kenwright, 2012) were used to have a compact representation of the error in position and orientation (SQP-DQ), while the sum of squares (SQP-SS) and the L2-norm (SQP-L2) of the 6-dimensional operational error vector were selected for their simplicity. Results showed a high success rate for SQP-SS that was also featured by a significant solve time for certain configurations. On the contrary, KDL-RR was featured by a low solve time even though it did not have a high success rate. These findings inspired Beeson and Ames (2015) for proposing the TRAC-IK solver that runs in parallel SQP-SS and KDL-RR and stops once one of the two algorithms returns a valid solution. The experiments on different robotic platforms proved that the proposed IK strategy clearly outperformed the others in terms of success rate and solve time in different conditions and tolerances.

2.1.2 Sampling-based planners

This class of planners relies on sampling the C -space with random or deterministic functions in the attempt of constructing graphs or trees that can be searched to solve specific planning queries (Lynch and Park, 2017). Sampling-based methods are commonly featured by the same conceptual items and, for this reason, a stable implementation of these algorithms can be found in a shared framework: the Open Motion Planning Library (OMPL) (Sucan et al., 2012). The OMPL is a popular C++ library in the robotics community that implements many state-of-art sampling-based planners and allows the integration of new ones. Additionally, the OMPL is used by the MoveIt! library for motion planning¹ and can be interfaced with ROS packages. In many robotics applications, two significant families of sampling-based algorithms for motion planning have been successful: the Probabilistic Roadmaps (PRM) and the Rapidly-exploring Random Trees (RRT) with their variants.

A complete and experimentally supported version of the PRM planner was introduced by Kavraki et al. (1996). The main planning process needs a first learning phase for the construction of a graph in the C_{free} , which is then searched for a valid path from a start to a goal configuration in a subsequent query phase. The algorithm builds a roadmap in the collision-free configurations space of the robot to take knowledge of the environment with stationary obstacles. The roadmap consists of configurations (nodes of the

graph) sampled from the C_{free} with uniform probability distribution that are connected to maximize the coverage of the configuration space. Every time a configuration is sampled, a connection with a number of nearest configurations is attempted by a local planner. In Figure 2.1, for example, a bi-dimensional

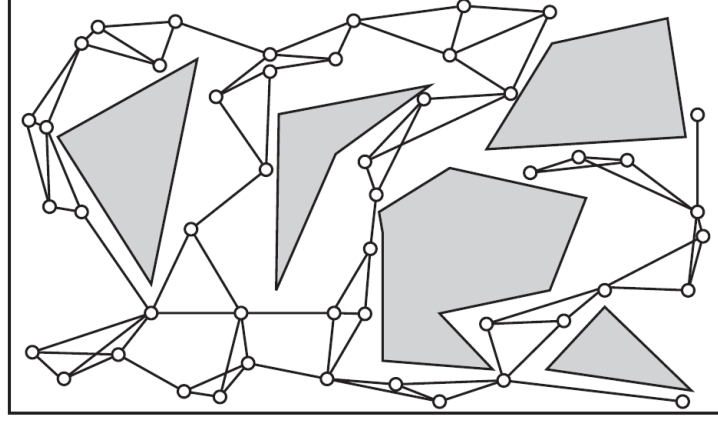


Figure 2.1: An example of a PRM roadmap for a point robot in $C\text{-space} = \mathbb{R}^2$. Obstacles are in grey and the C_{free} is white. The number of nearest neighbours taken into consideration for connection is 3. Picture taken from Lynch and Park, 2017.

probabilistic roadmap with 3 nearest neighbours connections for each sample is illustrated. Even though it is not a strict requirement, a straight-line local planner is used because it is simple, deterministic, very fast and provided good results during the learning phase. The collision checking algorithm is performed on each sampled configuration and on the local paths after their discretization in an arbitrary number of configurations. When a local path is collision-free, then it forms an edge for the graph and it is added to the roadmap. The learning phase also consists of an expansion strategy that increases the number of samples in difficult regions with narrow passages of the C_{free} to enhance the connectivity of the graph. In the query phase, the start and the goal configurations in the C_{free} are first connected by the local planner to two samples of the the graph constructed in the previous learning phase. Then, a valid path between these two samples is searched on the graph and returned with the components that do not belong to the graph.

The PRM method may require the connection of thousands configurations (or states) of a robot to find a solution, which is poorly efficient or even impractical for many applications with non-holonomic constraints. With the aim of addressing such problems, LaValle (1998) presented the Rapidly-exploring Random Trees (RRT) for path planning. The RRT method builds a tree (a directed graph) rooted in the initial state of the robot and grows with a given number of vertices. At the beginning the tree is empty and, in each iteration, a sample of the state space of the robot is selected following an uniform probability distribution. Then, the nearest state of the tree is searched and given to a local planner for moving towards a new state. The local planner discretizes the motion control described by the kinematic or dynamic model of the robot and

provides the control signal to move accordingly with given constraints. If the motion is valid (collision-free), then the new state becomes a new vertex of the tree and the edge from the nearest state to the new one is added to it. An example of a bi-dimensional RRT rooted in the center of the $\mathcal{C}$ -space and built assuming any bounded velocity is illustrated in Figure 2.2. When samples are uniformly distributed, the expansion of the

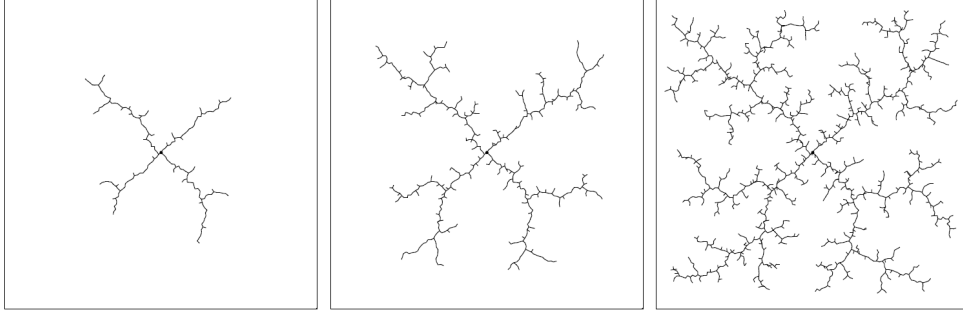


Figure 2.2: Three frames of the construction of an RRT rooted at the center of the $\mathcal{C}$ -space $= \mathbb{R}^2$ in absence of obstacles. Picture taken from LaValle, 1998.

tree is biased towards unexplored portions of the collision-free state space, hence ensuring an extensive coverage. Compared to the PRM, the RRT planner is generally faster because each iteration requires the search of a single nearest neighbour instead of a set of neighbours around a random sample. Moreover, the RRT generally maintains a connected structure with a smaller number of edges (LaValle, 1998).

Specifically designed for a reliable and quick single-query path planning, Kuffner and LaValle (2000) introduced RRT-Connect, which is also sometimes called bi-directional RRT. The RRT-Connect algorithm is based on the combination of a two rapidly-exploring random trees with simple greedy heuristic. The planning starts with the initialization of two trees: one rooted in the initial configuration and the other one in the goal configuration inside the space $\mathcal{C}_{free}$. Then, a sample in $\mathcal{C}_{free}$ is selected and one tree is extended towards it generating a new configuration according to the constraints of local planning of the basic RRT algorithm. Next, a "Connect" function iteratively tries to connect this new configuration to the nearest vertex belonging to the other tree until either the connection is made or an obstacle is reached. When the connection is made, the path that results from the combination of the two trees is returned. Otherwise, the role of the trees is swapped and another iteration starts. The outcome is the exploration of the $\mathcal{C}_{free}$ in the attempt of finding a configuration connecting the two trees that grow towards each other. A theoretical analysis demonstrated that RRT-Connect, like PRM and RRT, is probabilistically complete, therefore, if a solution exists, the probability that the algorithm finds it approaches one as more time is spent. Moreover, experiments on different applications showed that RRT-Connect is generally faster than RRT especially in uncluttered environments (Kuffner and LaValle, 2000).

Addressing optimality in sampling-based methods, Karaman and Frazzoli (2011) modified the PRM and RRT algorithms to introduce their asymptotically optimal counterparts: PRM* and RRT*, respectively.

A theoretical analysis of these algorithms demonstrates their asymptotic optimality (and their consequent probabilistic completeness) according to some cost of the planned path that approaches its minimum as the number of samples increases. The major modifications concern the radius used in the search of near neighbours around a sample of the C_{free} in the attempt of expanding the graph related to the planned path. This radius is suggested to be a function of the sample dispersion, which is $O\left((\log(n)/n)^{1/D}\right)$ for a set of n independent and uniformly sampled points, where D is the dimensionality of the C_{free} (Karaman and Frazzoli, 2011). Generally speaking, the graph is mapped with a number of connections per iteration that is proportional to $\log(n)$ to ensure asymptotically optimality and computational efficiency of PRM* and RRT*. While the graph defines the roadmap returned by the PRM* algorithm, the optimal tree of the RRT* is a subset of it. The extraction of this tree is driven by searching the minimal cost path in the neighbourhood of a new sample given by the local planning strategy. In each iteration, this search starts by assuming that the minimal cost is provided by the nearest sample in connection with the new one, but this assumption is not always true because it depends on the selected cost function. Then, when the optimal path is found, the set of edges composing the resulting tree is rewired accordingly. Experiments on the minimization of the path length showed the asymptotic optimality of PRM* and RRT* on thousands of iterations. Moreover, it was proved that adding asymptotic optimality on probabilistically complete path planners increased their time complexity only of a constant factor (Karaman and Frazzoli, 2011).

2.1.3 Optimization-based planners

A motion planning problem can also be formulated as a non-linear optimization problem with equality and inequality constraints. Optimization problems can be solved by gradient-based methods, such as sequential quadratic programming, or by non-gradient methods, such as simulated annealing (Lynch and Park, 2017). In general, these methods do not always guarantee to find a feasible solution when one exists and gradient-based methods usually provide a locally optimal solution starting from a meaningful initial guess. However, optimization-based planners are widely used in the robotics community and several techniques are integrated to overcome these notorious issues.

Zucker et al. (2013) introduced the Covariant Hamiltonian Optimization for Motion Planning (CHOMP), which uses gradient-based techniques that are invariant to the parametrization of a trajectory. This optimization method is based on the maximization of two objective functionals corresponding to the properties of smoothness and obstacles-avoidance in a trajectory. The functional of smoothness concerns the dynamics of a trajectory and it is formulated to minimize quantities across a trajectory, such as the squared velocity norm over the time interval of execution. The functional of obstacle-avoidance is invariant to the time-scaling of the trajectory and uses cost functions evaluated in the operational space of the robot (typically with 2/3 dimensions) to maintain the complexity of the algorithm reasonably low.

The CHOMP takes the properties of covariance to re-parametrization in the Riemannian geometry (Hasani, 1999) to introduce a Riemannian metric in the manifold of trajectories. This metric defines operations of norm and inner product for trajectories to measure the same physical quantities in presence of different geometrical representations. This property endows the planner with a gradient descent technique that iteratively searches the optimal trajectory with perturbations of small energy in the manifold of trajectories (Zucker et al., 2013).

Like in many gradient-based optimization method, the deterministic version of CHOMP can be trapped in local minima of the objective functional. However, sampling from a distribution of trajectories can solve this problem. For this reason, the gradient information can be augmented by methods like the Hamiltonian Monte Carlo (HMC) algorithm (Brooks et al., 2010). This technique allows CHOMP to better explore the space of solutions and makes the planner probabilistically complete. In particular, the HMC method is featured by a Hamiltonian functional that results from the sum of the objective functional, which is considered the potential energy of the physical system, and the kinetic energy computed from the trajectory momentum. In each step, a trajectory momentum is sampled randomly from a Gaussian probability distribution that associates high probability to small values of the kinetic energy. Then, a trajectory is numerically integrated from its momentum and it is accepted as a new step in the optimization process if its Hamiltonian is sufficiently small. Importantly, the HMC technique is also made covariant to re-parametrization of the trajectory by re-defining the Hamiltonian functional under the considered Riemannian metric. Tests of the deterministic version of CHOMP on different tasks showed a fast convergence and a good quality of the path. However, the performance often improve considerably with the HMC sampling, especially in difficult environments that are cluttered around the goal configuration.

Although the Hamiltonian Monte Carlo variations of CHOMP can be decisive for the success of the planning process, it might be of problematic application for the tuning of many parameters (Kalakrishnan et al., 2011). With the purpose of mitigating this complication and overcoming local minima issues that are typical in gradient-based methods like CHOMP, Kalakrishnan et al. (2011) introduced a Stochastic Trajectory Optimization for Motion Planning (STOMP). The STOMP is a non-gradient optimization method that can minimize non-smooth and non-differentiable cost functionals. This planner iteratively minimizes a state-dependent cost to address obstacle-avoidance with smooth trajectories that are discretized over a given interval of time. The planning algorithm starts by randomly sampling a set of trajectories around a given initial trajectory from a normal probability distribution with a proper covariance matrix, so that picking moderately accelerated trajectories is encouraged. Then, the probability of having each of these trajectories is computed to be exponentially inversely proportional to their cost, which refers to the avoidance of obstacles and joint limits. Finally, a new trajectory results from a probability-weighted convex combination

of these trajectories and the new corresponding cost is evaluated for convergence. Simulations and experiments on manipulation tasks demonstrated the feasibility of the trajectories planned by the STOMP and generally revealed a better success rate when compared to CHOMP (Kalakrishnan et al., 2011). Being local optimizers, STOMP and CHOMP can benefit from a meaningful initialization that can be generated by trajectory libraries of solutions in similar situations.

Another stochastic derivative-free optimization method for motion planning was introduced by Park et al., 2012 to allow fast re-planning in dynamic environments. Specifically, Park et al. (2012) proposed an Incremental Trajectory Optimization for Motion Planning (ITOMP) that interleaves the processes of planning and execution in parallel threads to continuously acquire knowledge about moving obstacles and avoid them. Like in STOMP and in CHOMP, the optimization process aims at enhancing smoothness of the trajectory and minimizing the cost associated with obstacles-avoidance and additional constraints. While knowledge concerning static obstacles can be acquired by signed Euclidean Distance Transform (EDT) (Kalakrishnan et al., 2011; Zucker et al., 2013), necessities for fast re-planning lead to the computation of conservative local bounds for dynamic obstacles. This representation allows to integrate the measures of position and orientation with errors coming from the sensors and offers a quick solution for the detection of collisions with moving objects. For instance, Figure 2.3 illustrates the avoidance of a dynamic obstacle after the anticipated collisions with its conservative local bounds. A scheduling module synchronizes motion planning and execution according to the goal settings and the monitored information from the environment. Planning and execution time are budgeted and refer to limited time intervals and not to the entire duration of the motion. Due to a limited time budget, the planner may return a sub-optimal solution, which is executed only if it is collisions-free. However, the consecutive planning stage starts optimizing from the latest returned solution, therefore, the trajectory is incrementally optimized within the given budget of time. Simulations of reaching movements demonstrated the efficacy of ITOMP in dynamic workspaces. It was also showed how the performance of the planner predictably decreases with the raise of the sensory noise, the number of dynamic obstacles and the magnitude of their speed.

Schulman et al. (2014) introduced a trajectory optimization solution for motion planning among obstacles with a low expensive collision-avoidance that can also guarantee continuous-time safety. This algorithm is implemented in a software package called TrajOpt and approaches trajectory optimization, which is a non-convex constrained optimization problem, with sequential quadratic programming (SQP) (Nocedal and Wright, 1999). The optimization problem is solved by a sequence of convex optimization problems that iteratively approximate the original problem inside the boundaries of a trust region. Thus, the objective function, which encourages the minimization of the path length, is locally approximated with a quadratic function and the constraints are locally approximated to affine functions. The obstacles-avoidance mechanism penalizes collisions according to the notion of minimum translation distance (Ericson, 2004), which

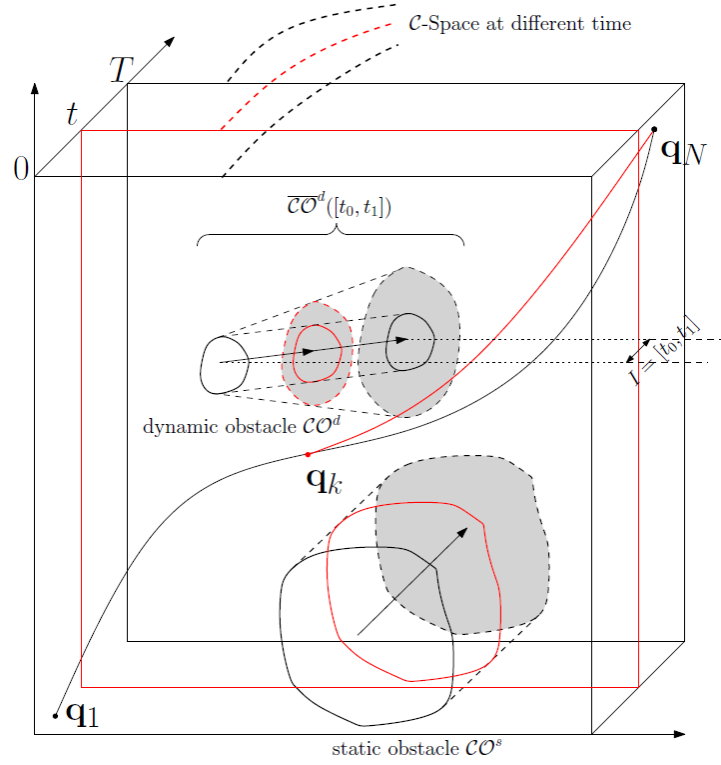


Figure 2.3: ITOMP: conservative local bounds of a moving obstacle during the interval $I = [t_0, t_1]$ in the period of execution. The collision is anticipated and re-planning the trajectory avoids the dynamic obstacle. Picture taken from Park et al., 2012.

allows a simple formulation of the collision constraints without a specific shaping of the obstacles in the workspace. Additionally, safety of the motion in continuous-time is ensured by considering swept volumes among way-points of the trajectory in the configuration space (see Figure 2.4). Experiments showed that continuous-time safety only adds a moderate computational burden to the discrete version of TrajOpt. The proposed collision-avoidance system showed its efficiency in different tasks in comparison with other planners. In general, TrajOpt outperformed different sampling-based planners and CHOMP in terms of success rate, planning time and quality of the solution. However, similarly to many optimization-based planners, the performance of TrajOpt is heavily dependant on the trajectory initialization because the optimization process provides a locally-optimal solution. Therefore, initializing the planner with trajectories passing through obstacles does not guarantee a collisions-free solution because the constraints for obstacle-avoidance of the motion planning problem are non-convex. For this reason, such methods often require strategies for re-initialization (Schulman et al., 2014).

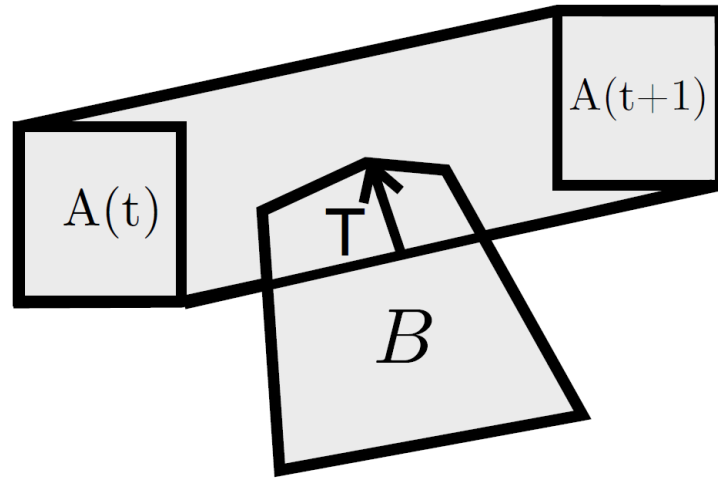


Figure 2.4: TrajOpt: illustration of a swept volume (Ericson, 2004). The volume covered by an object in two consecutive time steps is considered for continuous-time collision-avoidance. Picture taken from Schulman et al., 2014.

2.2 Human-like arm motion generation in robotics

The aim of this section is to provide a comprehensive background of human-like arm motion planning for robots by systematically reviewing the most recent literature. The content of this section has also been published in Gulletta et al., 2020. An extensive description of different approaches and techniques is essential for the identification of the current issues and the definition of new activities of investigation. The design of this review is systematic to offer the possibility of reproducing it and it was led by particular research questions. For instance, special attention was put on the criteria used for the evaluation of human-likeness of the generated movements, on the tools and robotic devices used for experimentations, on the physical nature of the applied methodologies and on obstacle-avoidance. The reliability, the performance, advantages and disadvantages are analysed and generally discussed to summarize the modern state of the art in the generation of human-like arm motion.

Being sources that include the vast majority of peer-reviewed publications on human-robot interaction and general robotics, the ACM Digital Library, the IEEEExplore, the Scopus and the Web of Science were searched. The search terms that were used in the advanced boolean method are:

(TITLE-ABS-KEY (arm) OR TITLE-ABS-KEY (upper-limb)) AND (TITLE-ABS-KEY (human-like) OR TITLE-ABS-KEY (legible) OR TITLE-ABS-KEY (predictable)) AND (TITLE-ABS-KEY (motion) OR TITLE-ABS-KEY (movement) OR TITLE-ABS-KEY (trajectory)) AND (humanoid OR robot) AND (planning OR generation) AND PUBYEAR > 2005 AND (LIMIT-TO (LANGUAGE , "English")) AND (LIMIT-TO (SRCTYPE , "p") OR LIMIT-TO (SRCTYPE , "j")).

The sources published in peer-reviewed conferences or journals were included, while unpublished or

non-peer-reviewed manuscripts, book chapters, posters, and abstracts were excluded. Moreover, anything published before 2006 was also excluded because state of the art might not reflect the capabilities that robots have nowadays. For the sake of completeness, papers proposing methodologies tested only on simulated frameworks were included, as well as on real robotic platforms. The focus is in those methods and techniques capable of generating human-like arm movements on robots (either humanoid or not). The minimum requirement featuring these planners is the capability of producing arm and hand trajectories showing typical human-like characteristics, which have been widely analysed and described in psychology and neurophysiology (see, e.g., Rosenbaum et al., 2001).

In section 2.2.1, an overview of the search results is described, while section 2.2.2 outlines the general characteristics of approaching human-likeness of robotic arm motion in the proposed methods. In section 2.2.3, the included studies are analysed and classified according to the adopted human motor principles, the type of the generated movements and distinctive functional properties. section 2.2.4 summarizes the review findings, limitations and challenges that are addressed by this PhD project.

2.2.1 Overview of the search results

The original search returned 156 results. There were 54 publications left after removing the duplicates across the databases and applying the inclusion and exclusion criteria. The included papers were published starting from 2006 and, therefore, concern the most recent research in the topic. Figure 2.5 illustrates how the publications are shared among the databases of search. Scopus with 61.11% of papers is clearly the primary source of knowledge. It is followed by IEEEExplore follows with 22.22%, Web of Science with 11.11% and ACM Digital Library with 5.56%.

The distribution of the publication venues is as follows (Figure 2.6): The International Conference on Intelligent Robots and Systems (IROS) was the largest provider with six studies. It is immediately followed by the International Conference on Robotics and Automation (ICRA) and the International Conference on Robotics and Biomimetics (ROBIO) with five papers each. Then, the International Journal of Humanoid Robotics (IJHR), the International Conference on Humanoid Robots (Humanoids), the International Conference on Numerical Analysis and Applied Mathematics (ICNAAM) and the journals of Autonomous Robots (AURO), Robotics and Autonomous Systems (RAS) and Intelligent and Robotic Systems (JINT) provided two papers each. The majority of sources delivered one paper. Among them, there is the International Journal of Robotics Research (IJRR), the Journal of Computational Science (JCS) and Frontiers in Psychology (FPsy).

Before describing more in detail the selected studies, an overview concerning the experimental equipment is presented. The types of results were classified according to the robotic systems that were used to validate the proposed methods of human-like arm motion generation. This preliminary distinction is

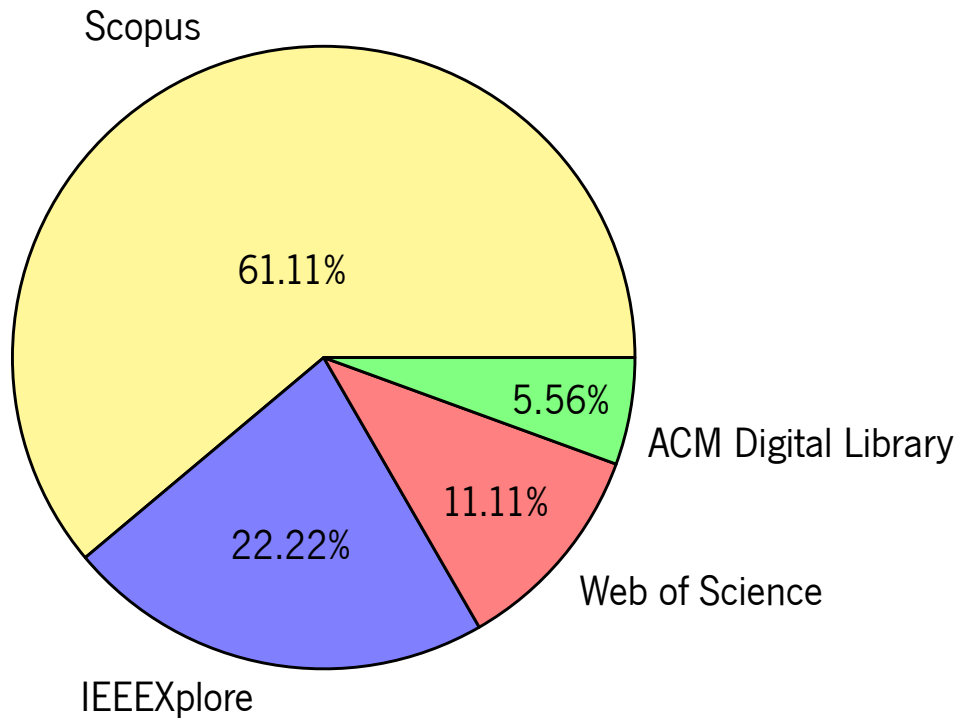


Figure 2.5: Pie chart showing the percentage of papers related to the databases.

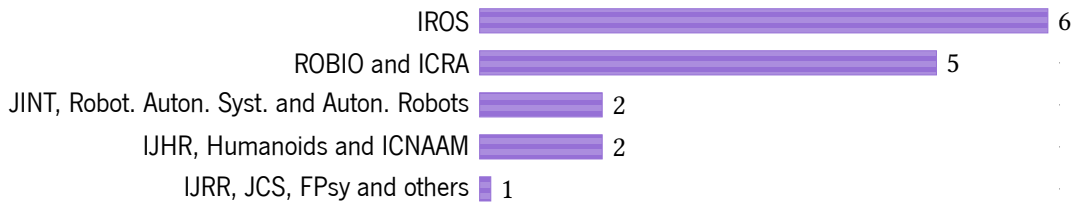


Figure 2.6: Bar chart showing the distribution of the sources of publication. Numbers indicate the number of papers per each source.

important to deduce implications regarding their level of applicability. For this purpose, the pie chart in Figure 2.7, illustrates the percentage of papers describing methods that were tested on robotic manipulators (37.03%), on humanoid robots (35.19%), and on simulations only (27.78%).

Robotic manipulators and, in second place, humanoids are arguably the most preferred tool for testing and showing the level of human-likeness provided by different generation techniques. A humanoid robot can provide a high level of anthropomorphism that can be used to encourage interactions with human partners (Duffy, 2003; Strait et al., 2017). Figure 2.8 shows the anthropomorphic robotic systems that were utilized in the included studies. *ARoS* (Silva et al., 2008) (Figure 2.8a) is an anthropomorphic robotic system composed by a static torso, two 7-DOFs lightweight Amtec arms, each one equipped with a three-fingered Barrett hand, a 2-DOFs neck, and a stereo vision system. *iCub* (Sandini et al., 2008) (Figure 2.8b),

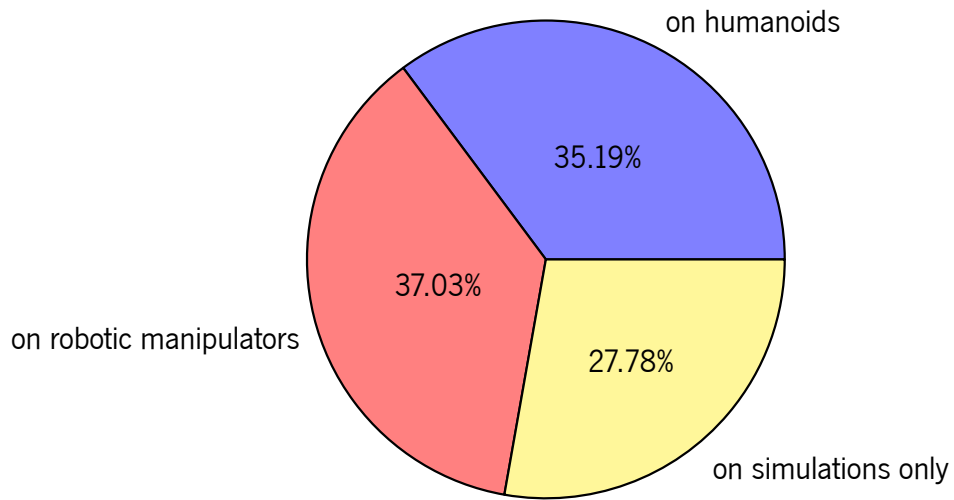


Figure 2.7: Pie chart showing the types of experimental results

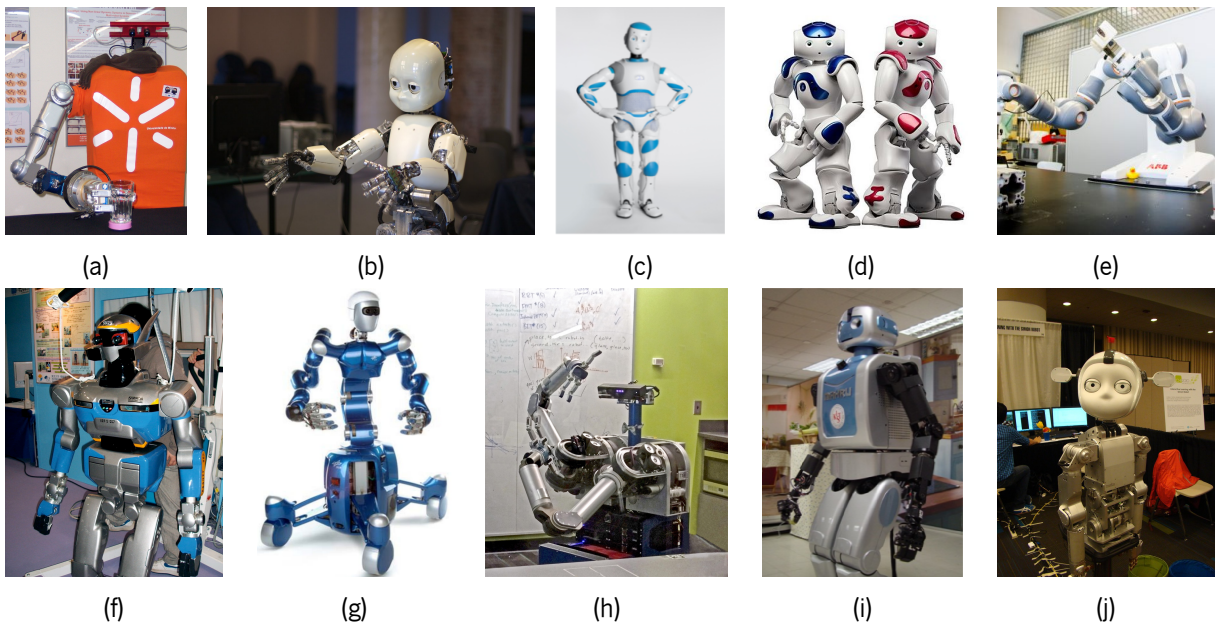


Figure 2.8: Humanoid robots that have been used to test human-like arm motion generation, namely: *ARoS* (Silva et al., 2008) (a), *iCub* (Sandini et al., 2008) (b), *ROMEO* (Tomić et al., 2018) (c), *NAO* (Gouaillier et al., 2008) (d), *ABB FRIDA* (Lamperti et al., 2015) (e), *HRP-2* (Kaneko et al., 2004) (f), *Rollin' Justin* (Fuchs et al., 2009) (g), *HERB 2.0* (Srinivasa et al., 2012) (h), *MAHRU* (You et al., 2008) (i), and *Simon* (Gielniak et al., 2013) (j).

with 30 DOFs in its upper torso, was designed to have a high degree of manipulation: each hand has 9 DOFs, and the two arms have 7 DOFs each. Additionally, it has cameras, microphones and position/force sensors that permit it to obtain audiovisual information and to safely interact with the environment. *ROMEO* (Tomić et al., 2018) (Figure 2.8c) has a total of 37 DOFs, which include 7 DOFs per arm, 6 DOFs per leg, 2 DOFs for each eye, 1 DOF for each foot, 2 DOFs for the neck, 2 DOFs for the head and 1 DOF for the spine. *NAO* (Gouaillier et al., 2008) (Figure 2.8d) is a humanoid robot designed to adopt specific behaviors according to the needs of the user. It has a total of 25 DOFs of which 14 DOFs are in the upper body, with two 5-DOFs arms, each one equipped with a 1-DOF hand. Designed for assembly applications, *ABB FRIDA* (Lamperti et al., 2015) (Figure 2.8e) is an upper-torso humanoid robot with two 7-DOFs arms, each one equipped with a servo gripper for small-part handling. The *HRP-2* robot (Kaneko et al., 2004) (Figure 2.8f) is a humanoid robot equipped with two legs and two 6-DOFs arms, each one equipped with grippers. It has been designed for cooperative works with humans and has three Charge-Coupled Device (CCD) cameras mounted in the head. *Rollin' Justin* (Fuchs et al., 2009) (Figure 2.8g) is a humanoid robot with 43 DOFs in the upper-body and an omnidirectional torso driven by 4 wheels. *HERB 2.0* (Srinivasa et al., 2012) (Figure 2.8h) is a humanoid robot specifically designed to operate with human partners in domestic environments. It is equipped with wheels, which allow the robot to navigate indoors, and with two 7-DOFs arms, each one equipped with a three-fingered Barrett hand. It also has an array of four laser scanners that provides clouds of 3D points in the surrounding environment. *MAHRU* (You et al., 2008) (Figure 2.8i) is a network-based humanoid robot with 35 DOF, including 6-DOFs for each arm and 4-DOFs for each hand. It also has two 6-DOFs legs, a stereo camera is mounted on the head and a microphone and four force sensors allow the robots to physically interact with the environment. *Simon* (Gielniak et al., 2013) (Figure 2.8j) is an upper-torso humanoid robot with two DOFs on the torso, two 7-DOFs arms, each one equipped with a four-fingered hand, a microphone and cameras for a stereo vision.

2.2.2 General insights on approaching human-likeness of arm motion

The procedures of human-like arm motion generation can be classified according to primary features that offer general but important insights on concepts, ideas and choices that are behind the genesis of a motion. A first classification addresses how the knowledge regarding the workspace of a robot is used to plan a path. A method is called global when the information of the entire workspace is processed off-line before the execution of a movement. On the contrary, local methods plan a motion on-line by taking into account portions of the workspace that are of immediate interest. Typically, these two approaches concern the implementation of strategies for collisions avoidance when it is considered. A second classification focuses on the physical nature of the variables that are involved in the motion generation process. Some methods

are based on kinematic variables only (position, velocity and acceleration), while others also consider dynamical quantities (forces and torques). A proper selection of these physical variables defines the state space of a robot and might be driven by the expected interaction with the environment or with a human partner. However, the choice of considering dynamics, instead of kinematics only, is often influenced by biological motor principles when human-like motion generation is supported by discoveries in neuroscience. Regularities of human upper-limb motion can also be replicated on different levels concerning the space where a robot acts. Some applications may constraint the dimensionality of this space that consequently affects the complexity of a planning algorithm. For these reasons, a third classification concerns the space in which the generation of a motion takes place: it could be the operational space or the joints space of manipulation (Siciliano et al., 2009).

For example, Chen et al. (2018) introduced a technique to reduce the dimensionality of the joints space for human-like dual-arm motion planning. This approach wants to replicate the demonstrations of human operators acting on common daily tasks. In particular, the replica of a bimanual task of pouring water from a cup to another one was performed. The position and the orientation of human hands were mapped into the joints space of two 6-DOFs manipulators by inverse kinematics. Then, the configuration space (i.e., the joints space) is reduced to minimize the computational burden of the search of a collision-free path that is provided by the RRT-Connect algorithm (Kuffner and LaValle, 2000). Therefore, this approach works in the joints space of a manipulator; it runs off-line and globally deals with the workspace of the robot. Dynamical variables of the motion were not considered because Chen et al. (2018) focused on the study of the non-linear relationship of the joints positions in the attempt of extracting this particular aspect of the human upper-limb motion.

Similarly, a global method for human-like bimanual motion planning was presented by Silva et al. (2015). The proposed planner is optimization-based, it does not depend on task-specific recorded human data and works in the joints space of the humanoid robot *ARoS* (Figure 2.8a) equipped with two 7-DOFs arms. The design of the planning strategy takes inspiration from typical kinematic features observed in human arm and hand movements, such as the minimum jerk model (Flash and Hogan, 1985) and the posture-based model for motion planning (Rosenbaum et al., 2001). Goal reaching and obstacle avoidance are guaranteed by the selection of a goal and a bounce postures, which are computed through the minimization of the angular jerk of the motion. Then, asynchronous bimanual tasks can be performed after the settings of intermediate hand goals in the operational space. This method only leverages on kinematic variables to generate the human-like kinematic regularities detected on human healthy subjects. Even though the planning process takes place in the joints space, human-like kinematic features, like a bell-shaped velocity profile (Rosenbaum et al., 2001), also emerge in the operational space of the robot.

An example of local planning in the operational space is described by Rano and Iossifidis (2013), who

presented an attractor dynamics approach for human-like hand motion generation. The authors considered the spherical coordinates (the heading directions and the velocity vector) of an end-effector to model human arm motion of reaching movements with a robotic manipulator. The proposed dynamical systems are based on attractive vector fields in the vicinity of a goal for the end-effector and repulsive vector fields for the obstacles in the workspace. Importantly, the attractor dynamics were properly designed to replicate the planarity characteristic of human hand motion, which is also observed in scenarios cluttered with obstacles (Grimme et al., 2012). In other words, a straight-line solution for the trajectory of the end-effector was analytically determined in obstacles-free scenarios where expectations on planarity were respected. However, to prevent unwanted collisions with obstacles, the trajectory could not maintain small values of torsion (i.e., good planarity) because the repulsion effects are significant only in the proximity of the obstacles and poor on other regions of the workspace (Rano and Iossifidis, 2013).

A hybrid joints/operational-space approach based on dynamic movement primitives (Ijspeert et al., 2013) was proposed by Lauretti et al. (2019) to enhance functional anthropomorphism in redundant manipulators. In specific, Cartesian and joints-related characteristics are extracted from recorded human upper-limb movements during the execution of reaching tasks. The parameters of a Cartesian and of a joint dynamical movement primitives (DMPs) are learnt from the recorded demonstrations and used to compute compatible desired trajectories in the operational and in the joints spaces. These references are coupled into an inverse dynamics controller that prioritizes operational-space constraints, while exploits the redundant variables of manipulation to replicate human recorded motion. Experiments with a 7-DOFs robotic arm and comparisons with classical Cartesian DMPs approaches demonstrated that the proposed hybrid method could provide acceptable accuracy in positioning and orientating the end-effector, higher levels of anthropomorphism within given physical joint limits and successful obstacle-avoidance.

Reaching movements were also generated for modelling human hand motion in relation with choosing a target through visual attention (Strauss and Heinke, 2012). A simple 2-DOFs robotic arm was sufficient to test the effects of biasing motor control towards an incorrect target on the kinematics of the resulting end-effector trajectory. In specific, Strauss and Heinke (2012) presented a local method for planning trajectories in the operational space that takes advantage of the Dynamical Neural Fields (DNFs) theory (Erlhagen and Schöner, 2002) to implement the detection of a target and control the robot in a human-like manner. First tests concerned single-target experiments to assess the appearance of typical kinematic characteristics of human hand reaching movements: namely, a straight-line path and a bell-shaped velocity profile (see, e.g., Jeannerod, 1984). Then, the effects of pre-activations of the DNFs for target detection were studied to simulate human visual attention in choosing the correct target in presence of multiple ones. In conformity with observations in humans, pre-activations generally produced curved paths with a longer duration and a prompt start at the beginning of the motion (i.e., a smaller initial latency).

Dynamic models that simulate the capabilities of the human arm have also been used to generate biologically-inspired arm motion (Fu et al., 2012; Täix et al., 2013). For example, Fu et al. (2012) investigated how the motor synergies hypothesis behind the motion of human arms can improve planning reaching movements for a planar manipulator. Specifically, motor synergies are responsible for the activation of a group of muscles in a coordinated manner. Therefore, approaching control can be simplified by managing fewer variables corresponding to the synergies of a motion. The general planning strategy is formulated with a framework for optimal control that minimizes the contribution of the control activation inputs acting on the joints of the manipulator. Fu et al. (2012) proposed the extraction of motor synergies from control inputs to solve the optimization problem in a lower dimensionality and, consequently, decrease the computational burden of the algorithm. Experiments on a 3-DOFs planar human-like arm moving on a bi-dimensional workspace demonstrated how the reduction of dimensionality generally improved the performance of planning by providing a higher success rate and a lower cost of the objective function spending less computation time. Further investigations on the properties of motor synergies showed their capability to adapt for the optimization of specific criteria, such as energy efficiency, jerk and smoothness of the resulting trajectory while achieving the intended goal of a movement.

A neurologically-inspired global method that accounts for the dynamics of a 6-DOFs arm was proposed by Täix et al. (2013). Reaching movements are generated by the application of six pairs of virtual antagonist muscles, which act like a second-order low-pass filter for a control signal and produce two forces in opposition. The scaled difference of these forces provides the torque that is applied to each degree of freedom of the manipulator. The trajectory of a movement results from the minimization of the energy related to the control signals over a given duration, which is formulated as a non-linear constrained optimization problem. Experiments on single-arm reaching with the robot *HRP-2* (Figure 2.8f) showed that different position goals in a three-dimensional workspace could be successfully achieved and that the proposed strategy is capable of replicating typical human-like features, such as a quasi-rectilinear path in the operational space and a bell-shaped velocity profile of the wrist. In the attempt of reducing the computational effort, Täix et al. (2013) also proposed an approach based on Principal Component Analysis (PCA) to extract movement primitives from a database of reaching movements obtained by selecting evenly distributed goal points in the workspace. The concept of motor primitives finds a neurological basis in studies of the human Central Nervous System and allows an approximation of complex movements by the linear combination of different elementary motion components. The selection of these primitives permits a dimensionality reduction of the original optimization problem with a consequent improvement of the performance. Results demonstrated that a proper generalization from the database of reaching movements is characterised by the parameters for the linear combination when the end-effector is constrained to track a minimum jerk trajectory. These findings are also in support of the biological-like structure of the proposed method that can well reproduce

human-like kinematic features in the operational space.

Figure 2.9 illustrates a classification of the included studies concerning how the available workspace is approached, the physical nature of the variables that are involved in the planning process and the space of configurations. There is a clear predominance of global methods acting in the operational space and

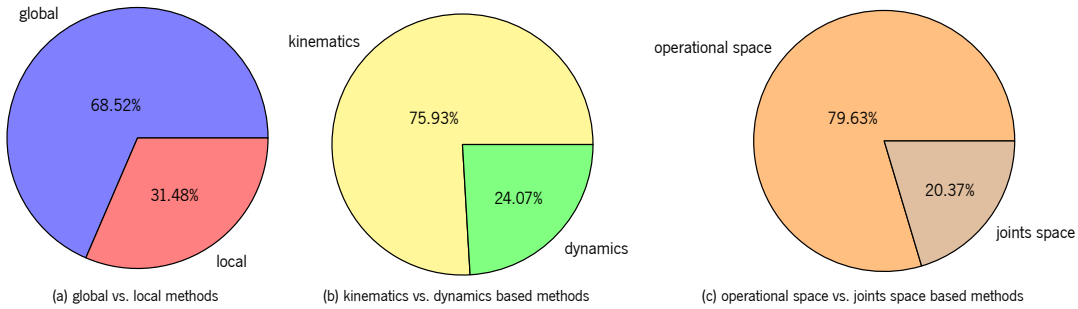


Figure 2.9: A preliminary distinction of the included studies

physically operating with the kinematic variables of a robotic device. Figure 2.9a shows that 68.52% of the included studies proposed global planning strategies, while only 31.48% of them treated the workspace locally. Modelling the workspace of a robot as a unique static environment generally simplifies the planning algorithms, permits taking advantage of the same assumptions in different portions of the scenario and can generally replicate more loyal human-like regularities of the motion. On the contrary, local methods are generally more complex and can normally operate in dynamic scenarios accounting for on-line obstacles avoidance. However, the risk of generating motion in local portions of the workspace is to lose the efficacy in replicating typical human arm motion features.

As it can be seen in Figure 2.9b, 75.93% of the reviewed studies introduced techniques that operate with kinematic variables, while the 24.07% of them also considered the dynamics of manipulation. These findings are mainly due to the requirements of simplicity of the proposed solutions, and to the fact that most of the studies on human arm motion have been conducted on a kinematic level of the analysis. However, dynamic variables are necessarily introduced in biologically-inspired solutions that aim at replicating the muscles activities in humans. Moreover, the dynamics of a robotic agent is also needed in applications when specific physical interactions with the environment are required.

The vast majority of the included studies presented solutions that operate in the operational space (79.63%) and only 20.37% of them in the joints space of manipulation (Figure 2.9c). Generating motion in the operational space generally means searching for a trajectory in a space with lower dimensionality than in the joints space. Additionally, patterns on human movements have been principally analysed by considering the hand motion, which reflects properties on the end-effector trajectory in robotics. However, techniques that exclusively perform in the operational space may discard shreds of evidence of human arm movements that arise on different degrees of freedom, such as joint synchrony and a bell-shaped angular velocity

profile (Breteler and Meulenbroek, 2006; Rosenbaum et al., 1995; Wada and Kawato, 2004). A largely observed disadvantage of working in the joints space is related to a higher dimensionality of the search space that significantly influences the computational expense of the proposed solutions. For this reason, many authors introduced techniques of dimensionality reduction to achieve better performance without degrading the intended level of human-likeness.

2.2.3 From humans to robots

2.2.3.1 Applied principles of human motor control

In many and different ways, the included studies have considered human motor control in the construction of the proposed methods. For this reason, it is important to have specific insights into the principles and techniques that transfer human motion into robotics and assess the human-likeness of the generated movements. In Figure 2.10, the most relevant principles and metrics of human-like motion in robotics are illustrated. In this section, robotic applications are analysed according to the popularity and originality of the human motor consistencies that support the proposed solution to the human-like motion generation problem.

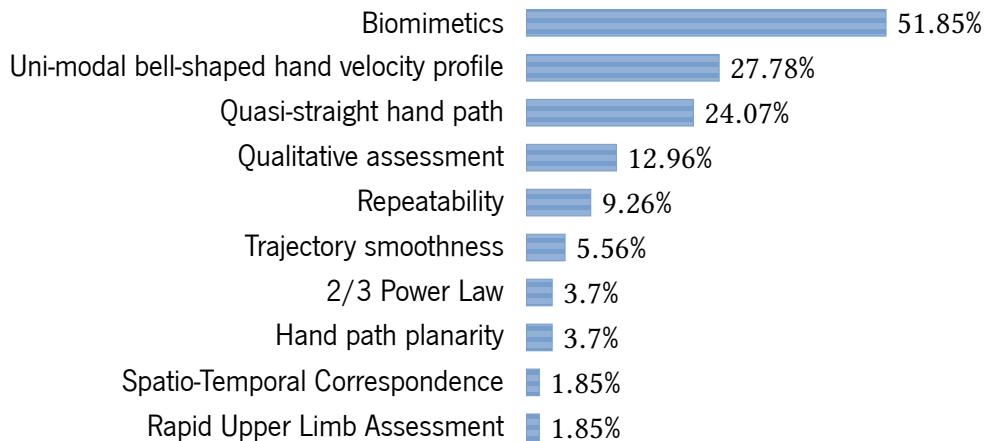


Figure 2.10: Bar chart showing in which percentage principles of human-likeness have been used for development and assessment in the included studies.

Biomimetics: In the majority of the reviewed articles, human-like arm motion generation has been approached with techniques of biomimetics (51.85%). This particular set of solutions is based on the extraction of bio-markers and of physical regularities from recorded movements with the ultimate purpose of mimicking arm trajectories. For instance, Kim et al. (2006) proposed a technique for deriving values of the

elbow elevation angle from series of recorded human reaching movements in a three-dimensional environment. The *elbow elevation angle* (also called *swivel angle* or simply *elbow angle*) is the angle between the sagittal body plane and the plane formed by the triangle shoulder-elbow-wrist (Kim et al., 2006; Zanchettin et al., 2011). This angle features a human arm posture uniquely once the position and the orientation of the wrist are given. For this reason, it represents the key for transferring motion from human records to the humanoid robot MAHRU (Figure 2.8i) that is equipped with two 6-DOFs arms. Then, a trajectory is computed in the joints space by analytical inverse kinematics that considers six holonomic constraints concerning the position of the wrist, the hand palm direction and the *elbow elevation angle*, which encodes the human-likeness of a movement. Experimental results showed the similarity with the recorded human motion and the possibility of approaching real-time applications with the proposed solution. The latter was also upgraded with further functionalities that improve the communication with a human partner that might be moving in front of the robot (Kim et al., 2007). Specifically, Kim et al. (2007) defined the concepts of motion plane, which is the closest plane where the wrist trajectory belongs to, and of motion centre, which is the centroid of the wrist trajectory on the motion plane. Changing the direction of the movement is possible by aligning the normal vector to the motion plane applied in the motion centre with the desired direction that keeps attention on a human observer. Additionally, the *elbow elevation angle* and the magnitude of the wrist position are properly scaled to maintain human-like characteristics on the modified movement.

Artemiadis et al. (2010) proposed a biomimetic approach with capabilities of generalization for solving the inverse kinematic problem of redundant manipulators. Particularly, the human arm inter-joints dependency was studied on sets of recorded human reaching movements and probabilistically modelled by a Bayesian Network (Bishop, 2006). A human arm movement is firstly analysed by a system of motion detection through trackers positioned on the shoulder, the elbow and the wrist of the right arm. Then, five joints are identified and mapped onto a 7-DOFs robotic manipulator. An illustration of the human-robot mapping concept is sketched in Figure 2.11. The analysis is focused on positioning the wrist in a three-dimensional workspace with a desired roll orientation angle. Therefore, the inverse kinematic problem is redundant because 4 desired variables have to be set by a 5-DOFs arm. The given degree of redundancy is used in a closed-loop inverse kinematics to maximize the probability of having human-like configurations that resemble those ones collected on humans. For this reason, a tree model is trained from the recorded data to extract the mutual information between couples of joints and define conditional probability distribution functions by fitting Gaussian Mixture Models (GMMs) on the given dataset. Being a weighted sum of Gaussian distributions, the GMMs can efficiently represent complex and non-smooth probability distributions (Bishop, 2006). The inverse kinematics algorithm account for the pseudo-inverse of the Jacobian matrix of the manipulator to compute velocities of the joints. Additionally, the joint velocities projected onto the null

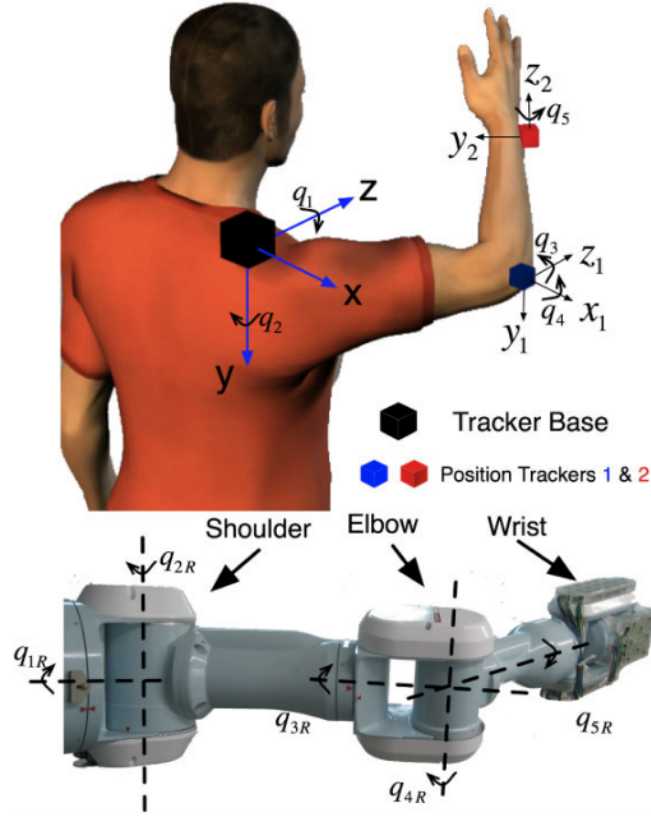


Figure 2.11: The equivalence of the human and robot degrees of freedom with the world reference frame on the shoulder. (Artemiadis et al., 2010)

space of the Jacobian matrix are not only used to avoid physical limits and singularities (Siciliano et al., 2009), but also to enhance the human-likeness of a movement (a secondary task) without compromising the correct location of the end-effector (a primary task). Observations on the angular position of the joints demonstrated the capability of imitating human arm reaching movements. Most importantly, the method was also able to use the learned inter-joints dependencies to generate new human-like movements that were not available among the off-line collected data. This relevant aspect provides a certain degree of generalization that might be essential in many practical applications with uncertainties on the environment. Mimicking reaching and grasping movements was also addressed by Zhao et al. (2014) through the minimization of the total potential energy of a human arm. The considered potential energy is the sum of a gravitational component, which is derived from the physical parameters of the arm, and of an elastic component that is obtained from acquisitions of human motion data on different subjects. Considering the planning of reaching movements, the minimization is once again permitted by the application of the Gradient Projection Method (GPM) that accounts for exploiting the null space of the Jacobian matrix (Siciliano et al., 2009). The position of the end-effector is assumed to reflect a minimum jerk trajectory with null boundary conditions, which has been observed in human reaching motion (Flash and Hogan, 1985).

Moreover, Zhao et al. (2014) extended this solution to grasping movements by the introduction of a novel human-like arm motion planner that combines the advantages of the Gradient Projection Method in redundant manipulators and the RRT algorithm (LaValle, 1998) (a GPM-RRT planner). A sampling-based planner like RRT allows to define a trajectory for the end-effector orientation, which is unknown a priori, and, most importantly, permits biasing the construction of a tree with optimal configurations provided by the GPM. Notably, in the case of grasping, further human data needed to be acquired to quantify how determined hand orientations affect wrist discomfort, which is defined as the distance from a comfortable wrist posture. Therefore, when the robotic arm acts on grasping minimizes not only its potential energy but also its wrist discomfort to mimic as close as possible human demonstrations. Qualitative observations on the human and on the robotic *swivel angle* trajectories showed the validity of the proposed planners in comparison with non-human-like alternatives.

Liarokapis et al. (2017) introduced an optimization-based technique of human-robot arm mapping that is founded on the principles of functional anthropomorphism (Duffy, 2003). The priority is the successful accomplishment of a task in the operational space, while human-like prerequisites of the motion are satisfied. A magnetic motion capture system provides the positions of shoulder, elbow and wrist from random reaching movements of different participants. Then, anthropomorphism can be transferred to complex robotic arm-hand platforms by the minimization of dissimilarity metrics, which represent different geometrical distance with human recorded motion. In particular, a simple sum of distances from the recorded human arm positions and the joints positions on a robotic manipulator revealed to be the most efficient metric. The proposed solution is formulated with a simple non-linear constrained optimization problem that guarantees a low computational cost and applicability on chains of arbitrary kinematics. The objective function of the optimization problem concerns the distance with the goal pose for a selected fingertip of the robotic hand, which replicates the human hand thumb during mimicking. Only the forward kinematics of the arm and of the fingers is used to determine end-effector and phalanxes poses, respectively. Additionally, constraints on anthropomorphism are softly added to the objective function through parameters that weight the influence of human postures in the accomplishment of a task. Comparisons with the classical inverse kinematics and simple joint-to-joint mapping showed that the proposed method was the only capable of achieving strong similarity with recorded human postures while successfully reaching goals in the operational space. Although the significant non-linearities in the formulation of the optimization problem, the proposed method of human-robot mapping revealed to be suitable for tele-operation and general applications with real-time requirements.

One of the advantages of human-like arm motion planning based on biomimetic principles is the possibility of customizing movements on the characteristics of human operators. Expressing anthropomorphism by mimicking humans is arguably the most intuitive and direct way that roboticists have at

their disposal, and probably, for this reason, biomimetic approaches are mainly preferred. Additionally, an efficient motion mapping from humans permits remote tele-manipulation in real-time when there is the need, for instance, to operate in dangerous or inaccessible environments. On the other hand, biomimetics methods require accurate and expensive motion capture systems that cannot always be available, and complex post-processing frequently due to noisy signals and adaptation on different embodiments. Moreover, these methods are often tailored for the recorded human movements or specific manipulation tasks, even though some effort on generalization has been carried on by applying techniques of machine learning. A learned model often results very dependent on a task or a set of movements that compose a training dataset and, therefore, represent patterns that cannot be generalized on any human-like upper-limb behaviour. Moreover, while biomimetics focuses on patterns of human motion in the joints space to encapsulate human-likeness for mimicking, kinematic analysis of the human hand is often neglected in the generation of end-effector trajectories. However, many studies on human reaching and manipulation have shown specific regularities in the operational space (Jeannerod, 1984; Pollick et al., 2009; Rosenbaum et al., 2001).

Kinematic assessment of the end-effector trajectory: It is state of the art that the human hand is typically featured by a quasi-straight path and a uni-modal bell-shaped velocity profile during reaching and prehension (Jeannerod, 1984; Morasso, 1981; Rosenbaum et al., 2001). For this reason, many of the proposed solutions have taken inspiration from these psychological pieces of evidence to generate human-like arm motion. Specifically, the 27.78% and the 24.07% of the included studies, respectively, considered the hand velocity profile and the shape of the hand path to synthesize and assess human-likeness. For example, Arimoto and Sekimoto (2006) introduced a virtual spring-damper hypothesis for three-dimensional reaching movements of a 5-DOFs robotic manipulator that is driven by the Euclidean distance between the initial and the goal end-effector positions. This method extends a virtual spring hypothesis for bi-dimensional reaching movements and avoids the application of any inverse kinematics or dynamics to resolve the redundancy of the problem. Taking inspiration from the physiology of the human arm motion, Arimoto and Sekimoto (2006) proposed to directly generate torque controls by the parametrization of stiffness and damping factors belonging to a virtual spring-damper system that connects the initial and the goal positions of the end-effector. Additionally, control signals are featured by passive damping on the joints of the manipulator and are influenced by the effect of the gravity force. Interestingly, the latter is estimated according to studies in handwriting control. The kinematic analysis of the generated reaching movements showed quasi-straight paths and single-peaked velocities profiles in the operational space. Acting on stiffness, different end-effector velocity peaks can be produced, while the shape of the end-effector path remains invariant.

Atawnih et al. (2014) introduced a control scheme that does not depend on the dynamic model of a manipulator and, therefore, does not suffer of the corresponding uncertainties. The proposed solution takes advantage of a prescribed performance control that allows to impose bounds on the evolution of the operational-space error over a given period of time. These bounds are set to reflect the minimum jerk principle of human arm motion in order to replicate the kinematic characteristics of a hand during reaching: a quasi-straight path and a single-peaked bell-shaped velocity profile. The control design provides torques on the joints by setting their angular velocity deviations from reference joints-space velocities, which are dependent on the evolution of the operational-space error through the pseudo-inverse of the Jacobian matrix. Moreover, the addition of active compliance to external forces on the end-effector increases safety and offers a major degree of human-likeness for physical interaction. A desired impedance between an external force and the resulting displacement of the end-effector is modelled and integrated into the proposed control scheme. Simulations with a 5-DOFs arm and real experiments with a 7-DOFs robotic manipulator demonstrated the capability of the system to reach a given operational-space target in a required period of time. The produced end-effector trajectories are replicable in the same conditions and are featured with human-like quasi-straight paths and bell-shaped velocities profiles. Additionally, the manifestation of external forces is successfully absorbed by the motion in stationary and moving conditions. A technique for time adjustment of the end-effector velocity was presented by Kashima and Hori (2016). Taking inspiration from observations of delays in human arm motion between the elbow and the shoulder, the authors proposed a modified minimum angular jerk principle for the optimal selection of this delay. This inter-joints time adjustment is designed to obtain exact symmetric end-effector velocities over a given duration of a reaching movement. This trajectory generation method was tested on a 4-DOFs simulated arm acting on a three-dimensional operational space, and it was compared with recorded human reaching movements and classical minimum jerk models. Investigation on short and lengthy distances from a goal position showed that the proposed trajectory formulation is featured by a symmetric uni-modal bell-shaped hand velocity profile and can replicate different curvatures of the human hand better than traditional methods that minimize the angular or the hand jerk.

The construction of human spatio-temporal features for an end-effector trajectory has also inspired a combination of potential fields to generate of self-reaching motion (Chen et al., 2016). In this particular study, Chen et al. (2016) only focus on highly curved self-reaching movements that transport the hand from a downward resting pose to a point of the upper body with the purpose of mimicking daily self-caring activities, like, for example, feeding or cleaning. The method is formulated in the operational space of manipulation and defines the composition of two forces: one exerted by an attractive potential field towards the target pose and another one by a repulsive potential field that prevents the formation of arm configurations in

the mid area between the initial and the target poses. The design of the attractive force is intended to replicate an initial accelerating phase and a subsequent deceleration as the distance with the goal becomes smaller, while the repulsive force is meant to push away the end-effector from a straight-line trajectory and generate high curvatures instead. Inspired from analogies with neuroscience, Chen et al. (2016) derived the intrinsic coordinates of the motion (i.e., the variables in the joints space) starting from the corresponding extrinsic coordinates (i.e., the variables in the operational space). In specific, a multivariate quadratic regression was applied to a small set of recorded human self-reaching arm movements to discover the relation underlying intrinsic and extrinsic variables for singular self-reaching tasks. This off-line procedure allows a real-time derivation of a joints-space trajectory from a end-effector trajectory that is generated by the composite potential field forces. Simulations and tests with a 7-DOFs robotic arm demonstrated the feasibility of the proposed solution and the generation of highly curved end-effector paths with uni-modal bell-shaped velocities. Moreover, Chen et al. (2016) highlighted the fact that the method is also applicable for producing quasi-straight hand paths by acting on a curvature regulation factor that can predict a task-related average human hand curvature.

An accurate replica of human hand positions and velocities during arm movements is strongly motivated by clear shreds of evidence in neuroscience and psychology. For this reason, many of the proposed human-like arm motion generation methods can synthesize these kinematic features by taking inspiration from different principles of the human upper-limb movements. Additionally, these techniques often take advantage of controlling schemas, which find applications when real-time requirements are needed. However, planning movements with kinematic constraints in the operational space is often intended for specific tasks and for reaching movements only, which transport the end-effector by point-to-point trajectories that ignore manipulation in the workspace. This is arguably due to the inevitable constraints that manipulation introduces and that may negatively affect the kinematic analysis of human-likeness.

Qualitative assessment: A qualitative assessment for the synthesis and the evaluation of human-likeness of robotic manipulation was also taken into consideration by the 12.96% of the included studies. For instance, Dragan and Srinivasa (2014) proposed an arm motion planning method that integrates the inferences of human observers to mathematically formalize principles of the action interpretation theory. Specifically, Dragan and Srinivasa (2014) introduced the notions of legibility and predictability as opposite properties of the motion that stem from inferences of an observer. In a scenario with multiple potential goals for a movement, a motion is predictable when, given a goal, it matches human expectations. On the contrary, a motion is legible when permits an observer to quickly and confidently infer the correct goal. These concepts are translated into trajectory functionals for a constrained optimization that can plan predictable or legible arm movements in robots. Although the goal of the authors is not to explicitly generate

human-like arm movements, the latter may be the outcome if this matches the expectations of a human observer. Legible movements, in particular, are intended to show the underlying intention of an action in ambiguous environments (with multiple potential goals) that might be shared with human partners. Thus, the integration of human inferences into robotic manipulation may significantly improve human-robot collaboration. Experiments with the robot *HERB* (Figure 2.8h) demonstrated the feasibility of the proposed planners and the exposition of the robot motion to human volunteers fairly supported the formulations of predictability and legibility in the domain of robotics.

The support from subjective ratings of robotic movements has also been used in combination with quantitative metrics of evaluation. For example, the trajectory planner described by De Momi et al. (2016) finds support in different validating options. In specific, De Momi et al. (2016) introduced a neural network based approach for robot-human handover tasks, which involves four independent neural networks (NNs) trained to learn the features of real human reaching movements towards evenly distributed goal positions in a three-dimensional space. After the training, the proposed solution is capable of predicting a human-like end-effector trajectory and its corresponding time of execution. Then, inverse kinematics is applied to transfer the motion to a 7-DOFs robotic manipulator. A qualitative assessment of the proposed solution was performed by recruiting volunteers for classifying an observed robotic movement as "biologically-inspired" or as "non biologically-inspired". In order to focus the attention on the end-effector and not on the motion of joints, the robot was covered and participants wore soundproof headset that mitigated the motor noise. Experimental results of this subjective ratings and of quantitative evaluations of human-likeness were in support of the proposed approach.

Collecting and analysing qualitative opinions of naïve subjects concerning the intention and the biological origin of a motion is certainly an easy and immediate manner of receiving descriptive feedbacks. However, the intrinsic subjectivity of such an analysis is the major drawback that induces the researchers to not only rely on it. Additionally, a qualitative assessment may also be misled by embodiment factors of the moving agent, which might distort the opinion of a human observer and contaminate the results. For these reasons, the number and the type of participants as well as the experimental protocol design are essentially important in user studies for the assessment of human-like arm movements in robotics.

Repeatability: The consistency and repeatability of robotic arm motion have been considered important human-like properties by the 9.26% of the included studies (Figure 2.10). For instance, Koskinopoulou and Trahanias (2016) presented an observation-based imitation framework capable of replicating human arm movements, as well as producing novel (unseen) actions. The proposed solution overcomes common correspondence issues that might affect standard learning from demonstration techniques by the employment of intermediate latent space representations. Specifically, human actions are visually recorded,

while robotic actions are kinesthetically taught. Then, these high dimensional spaces of action are reduced to bi-dimensional latent spaces by non-linear techniques of principal components extraction. During the training phase, these latent spaces are associated by the minimization of the errors between human and robotic trajectories pairings. After training, the reproduction of human movements is performed by inversely projecting the robotic latent space back onto its high dimensional action space in the respect of physical joints limits. Experimental results with a 6-DOFs robotic manipulator showed the feasibility of the described imitation framework on writing and pick-and-place tasks. The ability to consistently generate similar arm motions on the basis of human repetitions was quantitatively assessed on the reduced latent space, as well as on the action space, of the robot. The detection of irrelevant operational-space errors also exhibited the capability of repeating arm trajectories when comparable human trials are observed.

Methods that enhance repeatability during imitation can certainly generate human-like arm movements. However, there is a relevant risk of ignoring other characteristics that are typical of human upper-limb motion. While providing repeatable features can simplify observation-based learning and intensify predictability, it can also seriously undermine human-like generalization capabilities. Similarly to biomimetic approaches, expensive equipments and complex post-processing might also be required to record human demonstrations and show consistency on different kinematic levels (e.g., velocity, acceleration, etc.).

Trajectory smoothness: While in classical motion planning techniques the smoothness of a motion is measured by the sum of squared velocities or accelerations along a trajectory (Kalakrishnan et al., 2011; Zucker et al., 2013), human-like demands require jerk-based metrics. Only the 5.56% of the included studies considered such a validation to assess human-likeness with the generation of limited jerky movements. For instance, the movements generated by the solution from De Momi et al. (2016) were also evaluated through a measure of jerk on the principal plane of motion. Specifically, the sum of the squared norms of the jerk vectors in the path is normalized along the total time of execution and the length of the trajectory. This metric offers an independent index of smoothness for the detection of biological behaviours during the analysis of generic reaching movements (Chang et al., 2005).

The maximization of human-like trajectory smoothness was the foundation for the minimum-jerk Cartesian controller presented by Pattacini et al. (2010). The proposed solution takes inspiration from a multi-referential dynamical systems approach that takes advantage of the combined effects of a joints-space and of an operational-space controllers. Then, a coherence constraint mitigates the influence of each controller to provide robustness against singularities and physical joint limits. Additionally, an interior point method is applied to find the joints-space references for these controllers by the inversion of the kinematics from a goal pose in the operational space. The formulation of the control laws is based on the design of a linear time invariant (LTI) third order system that fairly approximates the smoothness of the ideal minimum-jerk

model, but suffers of a slower convergence on a steady state. Experiments showed the capability of reproducing quasi-straight paths and bell-shaped velocities of the end-effector trajectories and the avoidance of significant initial accelerations that are typical of classical controllers. Importantly, a smoother motion generation was also demonstrated by the relative measure of the squared Cartesian jerk over the period of execution.

The measure of the jerk of the motion provides a compact and intuitive manner to validate the smoothness of human-like arm trajectories. However, in many of the included studies this metric was totally ignored. It is also important the fact that a measure of the jerk cannot be the only tool of assessment, but it must be included in a multiple validation system. Otherwise, a null jerk motion, which produces a triangle-shaped velocity profile, might be wrongly associated with the smoothest human-like trajectory that can be ever generated.

2/3 Power Law: The two-thirds power law (2/3-PL) formulates the non-linear relationship that exists between the wrist velocity and curvature of human planar reaching movements (Lacquaniti et al., 1983; Wann et al., 1988). This kinematic characteristic was considered by only the 3.7% of the included studies as a valid reference of human-likeness in robotics. For instance, the 2/3-PL was part of the set of metrics that De Momi et al. (2016) used to support and consolidate the biological origin of the generated arm movements.

The two-thirds power law was also used for determining its impact on the perception of robotic motion in comparison with other levels of anthropomorphism (Hugues et al., 2016). In specific, Hugues et al. (2016) designed a virtual car assembly scenario with different robotic platforms for the establishment of a small user study with naïve participants. Geometric inverse kinematics procedures on a given set of robots were modified to provide three parameters of anthropomorphism: namely, a base inertia, an end-effector inertia and the two-thirds power law. While inertia on the base and on the end-effector was simulated by spring-damper systems, the 2/3-PL was respected by forcing the end-effector velocity in accordance to the curvature of its path. A simple protocol was set up by questioning on singular perceived sensations and feelings, while a simulated robotic manipulator was watched moving its end-effector on a circling planar path. Results showed that the most influencing criterion was the end-effector inertia, which positively impacts on the perceived flexibility, naturalness and human-likeness of the generated motion. Consequently, these movements encouraged safety, less stress and willingness to work with a robotic partner. On the contrary, the impact of the 2/3-PL on the perceived motion was mostly neutral.

In general, the application of the two-thirds power law in the generation of biologically-inspired arm movements is relatively simple and can be eventually used to replicate common human tasks, like hand writing. On the other hand, the assessment of human-likeness in robotic motion may require the planning

and recording of a significant number of arm movements with the corresponding kinematic features. The 2/3-PL is only exhibited by bi-dimensional human trajectories, but it was showed that introduces systematic errors in correlation with the overall shape of the path (Pollick et al., 2009).

Hand path planarity: The planarity of the hand path is a characteristic of human upper-limb movements that also occurs in the presence of obstacles.(Grimme et al., 2012) For this reason, the 3.7% of the included studies took operational-space planarity into consideration of human-like arm motion generation (Figure 2.10). It can be used to assess human-likeness, simplify arm motion planning and mimic human arm postures. For instance, Xu and Ding (2017) presented a method for bi-manual transporting tasks that constrains the motion of the end-effectors to lay on separated planes and utilizes synergies to reduce the dimensionality of the search for a feasible movement. In specific, the definition of planes of motion in the operational space allows the control of the wrists by two virtual 2-DOFs manipulators that are attached on a torso-like structure. From an anthropomorphic arm, a virtual manipulator is obtained by restraining a wrist to move along different possible shoulder-wrist lines, which can draw planar curves in the operational space. Positional constraints between the left and the right arms permit to find an analytical solution to the translation planning problem. While translations are planned, orientations are consequently stabilized to maintain a constant rotation matrix between the left and the right wrists and to keep contact with the transported object. Simulations demonstrated the feasibility of the proposed motion planner for bi-manual transportation tasks. Moreover, comparisons with conventional inverse kinematics methods showed more natural end-effector trajectories and the avoidance of awkward arm postures.

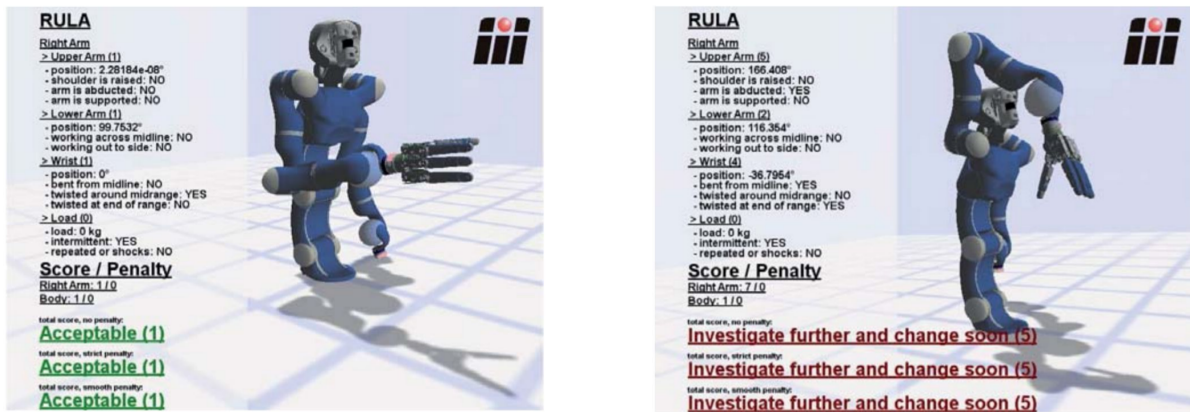
On the one hand, planar human hand paths in reaching and transporting tasks can be used to significantly simplify the arm motion generation problem to show human-like features. On the other hand, this simplification might only result valid on specific scenarios and movements. Moreover, a serious reduction of the search-space dimensionality might negatively affect other important aspects that are typically observed on human arm movements. For this reason, planar operational-space trajectories should be analysed in connection with other human-like metrics and should preferably emerge from independent applied human motor control principles.

Spatio-Temporal Correspondence (STC): Taking inspiration from the chaos theory, Gielniak et al. (2013) introduced the Spatio-Temporal Correspondence (STC) as a tool to synthesize and evaluate human-like motion. Similarly to biomimetics, the proposed algorithm starts from recorded human trajectories and brings together three procedures into a pipeline to generate human-like robotic movements. The first step produces human-like motion through the maximization of the correspondence in space and time with a reference trajectory extracted from human goal-directed activities. This correspondence emulates

the local couplings of connected degrees of freedom that in biological structures like humans are exerted by groups of muscles. The attempt of coordinating a reference trajectory through spatial and temporal shifts is realized by the minimization of a measure of dissimilarity, which has been generally used for chaotic signals: the Spatio-Temporal Correspondence. After this step, the resulting human-like trajectory is subjected to optimal control (Lewis et al., 2012) to produce a certain degree of variance for the motion. This second step is motivated by the fact that humans rarely replicate the same movement twice, even when the same task is performed. Adding variance around the reference trajectory (from the previous step) preserves the original human-like characteristics and, most importantly, respects the constraints of the motion in the joints space. Then, the optimal control provides a variant of the reference trajectory within the physical limits of the joints and respects any eventual synchrony. The third step projects the torque control commands onto the null space of the Jacobian that maps forces on the end-effector to the torque on the joints. This projection guarantees that the variant joints space trajectory does not compromise the operational space constraints of the reference trajectory, like holding a cup or pointing an object. Several tests with the humanoid robot *Simon* (Figure 2.8j) demonstrated that mapped motion from humans is perceived more natural, and its intention is easier to identify when it is coordinated with the STC method. Further experiments showed the capability of the proposed method of recovering human-likeness even in the presence of imperfect embodiments (e.g., any agent kinematically or dynamically different than the original one) or of lost information during mapping (e.g., down-sampling of the trajectory signal). Adding variance to the generated human-like trajectories did not degrade the ability to recognize motion's intentions by human observers, and the variants exhibited the capacity of respecting both operational and joints space constraints. However, as expected, the performance and the ability to generate human-like variants of the proposed solution decrease as the number of constraints increases.

The Spatio-Temporal Correspondence is a versatile tool that can be used either to generate human-like arm movements or to quantitatively and objectively evaluate human-likeness. Another advantage is that the proposed solution is not heavily data-dependent because only one input trajectory is necessary for the generation of many possible variants around it. However, like for biomimetics approaches, there is the need for recording task-oriented human movements that often require the application of expensive equipments. This also means that for each constraint in the operational space (end-effector position and orientation), an input reference movement is needed and one of its generated variants has to be appropriately selected. Notably, the proposed method also risks undermining fluent real-time interactions with human partners because two optimization processes in the pipeline (first and second steps) may significantly slow down the generation of the final human-like motion.

Rapid Upper Limb Assessment (RULA): Advances in ergonomics research have been an inspiration for novel human-like manipulation techniques that can also improve the performance of traditional motion planners in robotics. Zacharias et al. (2011) proposed a solution that takes advantage of the RULA (Rapid Upper Limb Assessment) criterion to generate human-like configurations and towards a determined end-effector pose in the operational space of a robot. In ergonomics, the RULA criterion scores conditions of stress and strain of human arm postures on a range from 1 to 7 with lower scores identifying more natural and comfortable configurations (Figure 2.12). The proposed path planner is equipped with knowledge



(a) Human-like posture with RULA score 1

(b) Non-human-like posture with RULA score 5

Figure 2.12: Two postures scored with RULA. (Zacharias et al., 2011)

concerning the reachability of the workspace in a human-like manner. This knowledge is generated off-line and depends on the robotic agent that is applied for manipulation. In this particular study, Zacharias et al. (2011) conducted simulations and tests with the robot *Rollin' Justin* that is illustrated in Figure 2.8g. The dexterous workspace is modelled by a capability map that is composed of discrete subregions where the end-effector can reach determined positions with different orientations. Each subregion includes spheres with a certain index of reachability, which measures the success rate in finding inverse kinematics solutions that place the end-effector onto uniformly distributed points on these spheres. Then, the restriction of the capability map to those robotic postures that possess acceptable RULA scores (usually no more than 4) provides a portion of that workspace that intuitively manifests the preferences of human manipulation. Given this restricted capability map, a RULA-driven inverse kinematics process (RulaIK) is proposed for reachable goal positions and orientations. While standard IK solvers depend on randomly selected initial arm configurations, the RulaIK iteratively try to solve the inversion of the kinematics starting from human-like postures with limited RULA scores that interest the region in the vicinity of a goal. Compared with standard methods, the application of the RulaIK demonstrated to generate awkward postures with a significantly lower frequency. Moreover, the computed restricted capability map is also used to bias

the sampling of a bi-directional RRT planner (Kuffner and LaValle, 2000) by selecting arm postures in a preferred range of RULA scores and that are feasible configurations in the region between the start and the goal postures. Tests on pick-and-place tasks in comparisons with standard methods exhibited a faster convergence and smaller mean RULA scores over the resulting trajectories when the RulalK was used in combination with RULA-biased sampling for path planning. Interestingly, a similar but slightly worse performance was achieved when the RulalK was used in combination with uniform random sampling for path planning. This result shows the significant influence of a human-like goal configuration on the planning time and on the overall ergonomics of a trajectory.

RULA-driven arm motion planning provides human-like arm trajectories that can also be perceived natural because they are acting on a region of the workspace that is usually preferred by human operators. However, typical human time parametrization is ignored and, for this reason, the proposed planner might negatively affect human observations of a resulting trajectory. Although a faster convergence is achieved by the mean of RULA-biased IK and sampling, the total planning time is not compatible with the requirements of a fluent human-robot interaction. Additionally, the construction of a capability map with restriction on RULA configurations is robot-dependent and also adds a level of complexity that has to be negotiated with the level of accuracy in the discretization of the workspace.

2.2.3.2 Classification of the generated human-like arm movements

In this section, more insights regarding the type of movements that can be generated by the reviewed solutions are provided. Specifically, a proper classification is necessary to understand better the degree of applicability of the proposed techniques and how they should be upgraded for a wider generalization in different tasks. As the Figure 2.13 illustrates, 87.04% of the included studies addressed human-like single-arm motion generation, while only 18.52% of them proposed techniques for dual-arm motion planning. This outcome certainly depends on the robotic devices that are available for experimentation and on the tasks being addressed.

Featured by a limited degree of complexity, reaching (sometimes also called point-to-point) movements have been the focus of the 74.07% of the included studies. This category of movements generally transports an end-effector from one point to another point of the Euclidean space $SE(3)$ without the engagement of any particular manipulation of objects. On the contrary, picking and placing actions of the presented planners concern manipulation of rigid objects in the workspace of a robot. In particular, 37.04% of the reviewed papers addressed the generation of human-like pick movements, while 20.37% of them considered placing (Figure 2.13). Interestingly, most of the placing actions were part of the planning and execution of more general pick-and-place tasks.

A more specialized branch of robotics research in human-like arm motion generation has investigated

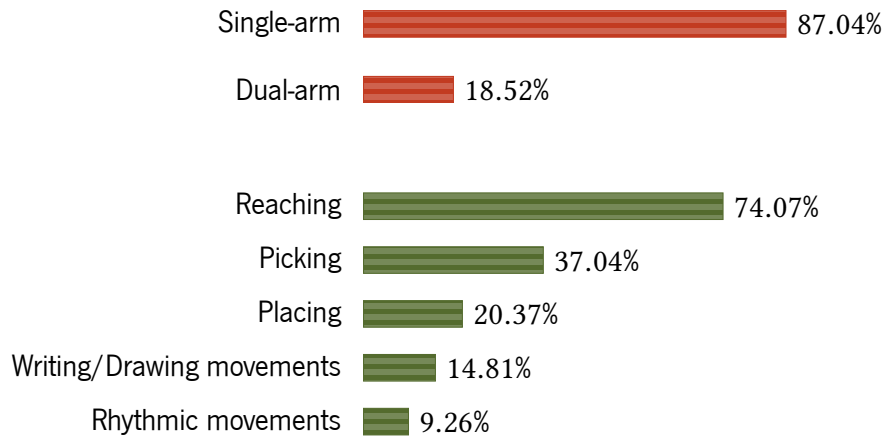


Figure 2.13: Classification of the human-like arm movements that are generated by the reviewed motion planners.

planar movements for writing/drawing applications and the reproduction of rhythmic trajectories, which regards the 14.81% and the 9.26% of the included studies, respectively (Figure 2.13). The latter areas of research attract particular attention for their originality and vast unexplored future work.

For instance, Caggiano et al. (2006) presented a biomimetic approach to distribute human-like mobility to a 7-DOFs virtual manipulator in drawing tasks. The trajectory planning is vision-based and reproduces in the operational-space human draws with similar movements. Writing and drawing tasks generally force humans to maintain the elbow, the wrist and the hand on the same plane of motion to compensate the effects of gravity and minimize the energy consumption during execution. Caggiano et al. (2006) take inspiration from these biomimetic evaluations to develop a simple closed-loop inversion kinematics algorithm that weights the contribution of the joints and regulates the motion projected onto the null space of the Jacobian matrix. The algorithm is designed in three layers to keep the elbow and the wrist of the robotic arm at the desired heights observed during human trials, while avoiding influences on the end-effector that follows a given drawing pattern. More emphasis is also given to the first four joints of the manipulator because more mobility was observed in human writing. Experiments on different case studies exhibited qualitative similarities in both the operational and joints spaces.

Li et al. (2013) introduced a trajectory planning method that can test different hypotheses concerning the role of the human Central Nervous System (CNS) in handwriting and drawing. Human actions seem to be represented in the CNS in more abstractive terms than with specific commands to the muscles (Morsella et al., 2009). For this reason, handwriting and drawing are encoded in terms of strokes in positions and directions without any specific motor command (Li et al., 2013). Specifically, three methods for imitating human handwriting and drawing are described: basic strokes, Bezier curves and non-gradient numerical optimization methods. The basic stroke method addresses the writing of alphanumeric characters with

two basic elements (or strokes): circular arcs and line segments. Both of them are described analytically, and the inverse kinematics of a 2-DOFs manipulator is computed. The time parametrization of the resulting path is adaptable to different velocities profiles that can also simulate neurophysiological injuries and diseases. Concerning the Bezier curve method, Li et al. (2013) took inspiration from computer graphics for the definition of a particular parametric curve, i.e. the cubic Bezier. This curve depends on four control points that are recorded after the visual acquisition of the image that has to be drawn. Then, a decoding process loads the saved parameters into a piecewise Bezier function that reproduces the handwriting. Finally, a trapezoidal speed profile is defined, and the end-effector motion is translated into specific joints-space commands. A variation of the Bezier curve method provides the non-gradient numerical optimization method that takes inspiration from vehicle guidance. Simulations with a three-links manipulator demonstrated the feasibility of the proposed trajectory planners in drawing complex figures mixed with alphanumeric characters.

Yang et al. (2010) introduced a biologically-inspired method for the generation of cyclic (or rhythmic) arm movements that resemble the stability properties of animal and human motion. In specific, the neural oscillatory circuits on the spinal cord, which are known as Central Pattern Generators (CPG), are formulated into artificial neural oscillatory networks and combined with a virtual force inducer (VFI) acting in the operational space of a manipulator. The VFI interacts with a series of oscillators that operate on each joint influencing the corresponding torque. Particularly, each neural oscillator tracks the joint sensory signal feedback to exhibit an adaptive behaviour during interactions with the environment (the phenomenon of “entrainment”). The control scheme accounts for the joints space dynamical model of a multi-DOFs manipulator, which is featured by a PD controller in the operational space that follows a desired trajectory and by the scaled contribution of the rhythmic commands from the neural oscillators. Interestingly, the proposed scheme can achieve cyclic pattern generation without calculating the inverse dynamics and confronting the ill-posed problem of redundant systems. Simulations and experiments with planar and multi-DOFs manipulators demonstrated the capability of the proposed solution of generating smooth reaching movements, of producing cyclic repeatability of the joint positions and of adapting to unexpected environmental changes.

Being a relatively recent and trending topic of research, techniques of human-like dual-arm motion and manipulation are of particular interest. Some authors presented algorithms designed for single- and dual-arm applications (Gielniak et al., 2013; Lamperti et al., 2015; Rosado et al., 2013), while others only addressed bi-manual planning problems (Gulletta et al., 2015; Shin and Kim, 2015; Suarez et al., 2015). For instance, Rosado et al. (2013) proposed a low-cost motion capture method for transferring single- and dual-arm motion from humans to upper-body humanoid systems. The proposed motion capture system is Kinect-based and addresses the extraction of a human body skeleton through the recording of Cartesian

points on the torso and the upper limbs. Then, the captured information are optimized to reduce errors on measuring the dimensions of the human arms. Given the positions of the shoulders, elbows and wrists, a geometrical kinematic mapping transfers the motion to a simulated hybrid humanoid platform, which is equipped with a 4-DOFs right arm and a 7-DOFs left arm that replicate the complex mobility of a human shoulder. The human-robot mapping is then post-processed by basic filtering and interpolating techniques, which are evaluated to quantify the smoothness of the resulting joints-space trajectories. Experiments on transferring rhythmic and discrete dual-arm motion showed the feasibility of the proposed method and the consistency of the reconstruction algorithm.

Lamperti et al. (2015) proposed a solution for resolving the redundancy of an anthropomorphic dual-arm robotic system through the minimization of muscular effort. With two 7-DOFs arms, the redundant variables raise from 1 in uni-manual motion to 8 in bi-manual motion when there are constant rules of inter-limbs co-operation with one object. The right hand posing problem is solved by minimizing the muscular activation that supports a manipulator against gravity with respect to given operational-space constraints. The right arm configuration is uniquely identified by the hand pose and by an optimal swivel angle that minimizes muscular effort. A swivel angle, indeed, is the angle between the sagittal body plane of motion and the plane projected by the triangle wrist-elbow-shoulder, which is a key factor in featuring the human-likeness of a posture (Zanchettin et al., 2011). Then, the left hand posing problem is solved by considering the relative translational and rotational constraints with a given right hand pose. Similarly, the definition of the left arm posture is completed by the left swivel angle that minimizes the muscular effort of the left robotic arm. Addressing many samples of different dual-arm tasks, Lamperti et al. (2015) fitted the data that relate arm configurations (14 variables) to the eight redundant variables that define the right hand pose, the right and the left swivel angles. Under the limits of the selected tasks, the fitting algorithm improves the performance of the proposed dual-arm redundancy solver because approximates the redundant variables by linear combinations of polynomial basis functions. Experiments with the humanoid robot *ABB FRIDA* (Figure 2.8e) showed the necessity of scaling the fitted functions according to the dimensions of the robotic manipulators. Results demonstrated the ability of the proposed method to generate human-like dual-arm movements when compared with those ones of a simulated bi-manual human kinematic model, which has spherical joints on the shoulders and the wrists.

In continuation of our previous studies (Costa e Silva et al., 2013), Gulletta et al. (2015) proposed a method of bi-manual synchronous motion planning for anthropomorphic robots. This synchronous dual-arm motion planner takes also inspiration from evidences of human motor control in the execution of complex bi-manual tasks. Indeed, it was demonstrated that the synchrony of the upper-limbs trajectories is supported by synergies of the muscles that are controlled as a singular functional unit (Bingham et al., 2008;

Hughes et al., 2011). A minimum angular jerk trajectory is proposed for the arms of a generic anthropomorphic platform and tested with the robot *ARoS* (Figure 2.8a). The trajectory depends on a goal dual-arm posture that allows the correct accomplishment of a task and on a bounce dual-arm posture that guarantees a collision-free path in presence of obstacles. Non-linear constrained optimization problems are formulated in AMPL (Fourer et al., 1989) and solved by IPOPT (Wachter and Biegler, 2005) to select goal and bounce dual-arm postures that minimize the overall angular jerk. Simulations and tests in assembly tasks demonstrated that the proposed planner generates feasible movements for humanoid devices and that can well perform in fluent human-robot interactions.

Shin and Kim (2015) used imitation learning to design and test a mimicking method for reaching, grasping and manipulating objects with the arms of a humanoid robot (Figure 2.14). The proposed method of human-

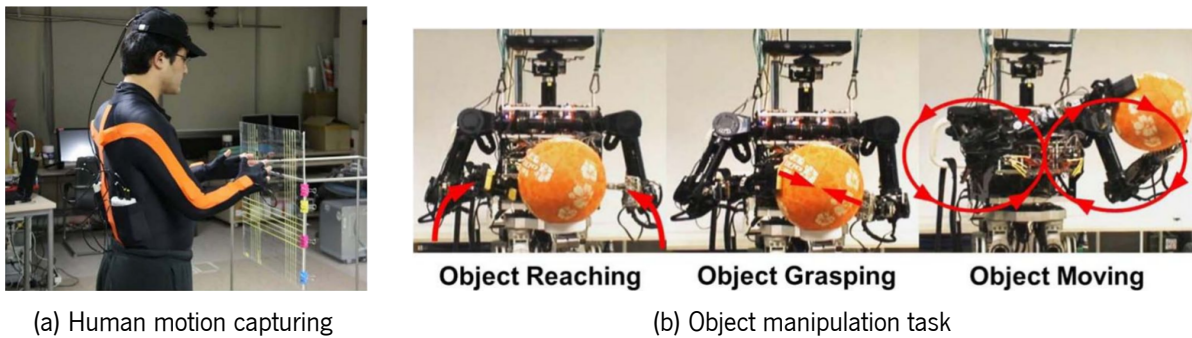


Figure 2.14: Dual-arm manipulation and force control from capturing human motion. (Shin and Kim, 2015)

like dual-arm motion generation is based on the acquisition of human motion data by a camera-less inertial motion capturing system (Figure 2.14a). The imitation is performed on the acquisition of object reaching and moving movements, while grasping is achieved by adjusting the distance between the end-effectors in consideration of the object's radius (Figure 2.14b). Dual-arm reaching movements are planned by the extraction of the principal components from the recorded human motion data related to the shoulders and to the thumbs, middles and pinkies fingers. The positions of the elbows are indirectly computed by the elbow elevation angle algorithm (Kim et al., 2006). Then, a Lagrangian optimization problem is solved to generate a new dual-arm trajectory that connects the recorded initial and final positions and closely adheres to the mean of the captured human reaching trajectories. Similarly, the object moving movements are planned after a reduction of the involved variables that only consider the position of the shoulders and the pose of the manipulated object, which permits to reconstruct the coordinates of the end-effectors. Then, the planned movements feed the reference of a virtual dynamics model (VDM) controller (Shin et al., 2012) to provide an accurate trajectory tracking and add compliance in the operational space of the robot. The human-like characteristics of the planned and controlled movements were assessed by quantitative and qualitative comparisons with the captured human trajectories.

Through the study of synergies in dual-arm manipulation tasks, Suarez et al. (2015) designed a dual-arm human-robot mapping system that is based on the extraction of principal movement directions (PMDs) to simplify sampling-based motion planning in the joints space. Specifically, the positions and the orientations of the hands of a human operator are firstly recorded and sampled during the execution of an assigned assembly task. Then, these data are transformed into the operational space of a humanoid robot with two 6-DOFs manipulators equipped with a 16-DOFs hand each. Subsequently, the samples are translated into the joints space of the robot and the values of the joints are properly filtered to minimize their variance. The synergies of a particular task are then extracted as principal components of the dual-arm configurations that are the result of the mapping process. These principal components were called principal movement directions (PMDs) and represent the recorded dispersion of the samples. After the selection of feasible and collision-free dual-arm goal configurations, more instances of the RRT-Connect algorithm (Kuffner and LaValle, 2000) are launched in a PMDs-subspace that reduces the dimensionality of the original search space (i.e., the dual-arm joints space). Experiments with assembly and free-movement tasks showed a significant increase of the planning computational performance when a minimal number of PMDs, which are sufficiently representative of the task-related human variance, is adopted.

In the same direction of research, Garcia et al. (2018) proposed a synergies-based partition of the operational space for dual-arm reaching and grasping with a humanoid robot mounted on a mobile platform. The synergies represent a human-like coordination between the translational movements and the dual-arm motion of the robot that is approaching a region of the workspace for grasping two cylindrical objects on a table. Human motion is captured and used as reference for transferring human-likeness, which is maximized by an optimization-based human-robot mapping that ensures the correct placement of the end-effectors for a successful grasping. Then, principal components analysis is run on the space of configurations for the torso and the joints to extract a minimum number of synergies that significantly comprise the variance of the samples. This analysis permits to partition the torso translational space of the robot according to similarities of the captured dual-arm movements and, consequently, reduce the dimensionality of the planning search space for new joints-space states. For this reason, the proposed motion planner modifies the original version of the RRT-Connect algorithm (Kuffner and LaValle, 2000) to take advantage of the acquired knowledge and considerably decreases the required planning time and iterations.

The vast majority of the included studies presented methods for the human-like generation of single-arm reaching movements, while poor attention has been put on drawing and rhythmic motion. More recently, more complex robotic applications of pick-and-place and dual-arm manipulation has started to attract the interest of an interdisciplinary research that takes inspiration from biological and behavioural findings. Although more generic tools of arm motion generation that can offer human-like solutions for different classes of robotic movements are often auspicious, it emerged that research seems to be focused on

narrow sets of applications. For instance, while more advanced solutions might potentially integrate simpler utilizations with few modifications, the techniques for planning human-like writing/drawing and rhythmic movements seem to be an isolated branch of investigation that can hardly be merged with more classical methods for reaching, picking and placing. Most of the proposed methods for human-like bi-manual manipulation could work for uni-manual applications too with the change of few design assumptions, but the opposite can unlikely be stated. Additionally, the great part of the included human-like dual-arm motion planners depends on biomimetic measures and on databases of human captured motion data that strongly limit task generalization.

2.2.3.3 Obstacles avoidance in human-like arm motion generation

Endowing robots with the capability of avoiding and preventing collisions with obstacles in the workspace is a necessary skill in scenarios that are shared with humans for interacting purposes (Fukuda et al., 2001; Schaal, 2007; Sheridan, 2016). However, only the 27.78% of the reviewed papers addressed this particular issue. For example, Rosell et al. (2011) proposed a human-like sampling of the configurations of a robotic arm-hand system that also improve the efficiency of a PRM-based graph building (Kavraki et al., 1996) in hand pre-shaping movements during grasping. Specifically, the hand motion of a human operator is captured during the execution of unconstrained movements and then mapped onto a robotic anthropomorphic hand to extract the inter-fingers coordination by a principal component analysis. Rosell et al. (2011) also took into consideration human-like requirements that may regulate the palm orientation constantly facing an object to be grasped during prehension. These considerations, and the principal components of the fingers' motion, permit to solve a planning query between two feasible arm-hand configurations with reduced dimensionality. Simulations and tests with a 6-DOFs arm equipped with a four-fingered robotic hand demonstrated the feasibility of the presented virtually constrained planner. Additionally, comparisons with standard PRM planning showed that the proposed method can provide sequences of human-like postures with a higher success rate in a minor computational time.

Assuming a pre-determined mental image of a final arm posture in human goal-directed movements, Xie et al. (2011) introduced the hypothesis of the "Target Arm Pose" (TAP) in human-like arm motion planning for robotic manipulators. The TAP hypothesis determines final arm configurations that maximize dexterity by seeking for a high Jacobian condition number in the inversion of the kinematics. A detailed analysis of the obstacles in the workspace and the proper selection of intermediate TAP configurations facilitate the generation of collision-free human-like arm movements. While the human motion of the fingers is mimicked by the minimum jerk model, the inversion of the kinematics at joint jerk level (IK-JJL) generates reaching movements by tracking an end-effector trajectory, which respects the boundary conditions with given TAPs and avoids the physical limits of the joints. In presence of obstacles, the TAP hypothesis for

intermediate via postures is combined with the RRT-Connect algorithm (Kuffner and LaValle, 2000) to generate collision-free paths. Simulation and tests showed the feasibility of the IK-JJL in planning reaching movements in obstacles-free scenarios. Moreover, the combination of the TAP hypothesis with the RRT-Connect algorithm improved the performance of the latter in planning reaching and grasping movements through narrow passages.

Liu et al. (2017) proposed an analytical inverse kinematics algorithm for anthropomorphic manipulators to map human-like configurations from recorded human arm motion data. In particular, the shoulder, elbow, wrist and palm points are operational-space key positions for a task-constrained human-robot mapping. While the shoulder positions is fixed at the base of a robotic arm and the palm/end-effector positions are dictated by the task, there is some flexibility in mapping the elbow and the wrist positions because only a grasping direction constraints the end-effector orientation. The proposed human-robot configuration correspondence solves the inverse kinematics problem with respect to the link lengths and the joints limits of the manipulator. In specific, the wrist key position is constrained to be on the line connecting the elbow and the palm positions (wrist-elbow-in-line) so that all the key positions lie on the same reference plane. Then, the obtained posture is iteratively adjusted if any of the joints limits is surpassed. This adjustment procedure starts by circularly moving the wrist position to obtain a feasible elbow joint angle. Next, the elbow key position is initially chosen to be the closest to the captured human elbow that lies on the reference plane. If any joints-space infeasibility is detected, then the elbow position is iteratively moved on an arc that defines different swivel angle values. The simulations with a 7-DOFs redundant arm demonstrated that the proposed IK method provides a high end-effector accuracy that slightly decrease in motion tracking but can show sufficiently rigorous human-likeness, which is manifested by the detected similarities with the human swivel angle. Interestingly, the human-like performance does not degrade when simple avoidance of a spherical obstacle is considered. The elbow key position is maintained, while the flexibility of the wrist is exploited to adjust the robot configuration and avoid collisions.

Although preventing self-collisions and avoiding obstacles in the workspace of a robot is a priority of human-centred robotics, the reviewed techniques of human-like arm motion generation rarely address it. Most of the proposed solutions for obstacles avoidance work under assumptions that significantly simplify a planning problem and take advantages of sampling-based motion planners that can surely provide collisions-free paths, but, on the other hand, require smoothing procedures and neglect typical human-like time parametrizations. Additionally, experiments on collisions prevention often ignore assessing kinematic human-like features that characterize human arm movements (Grimme et al., 2012).

2.2.3.4 Optimization, learning and control in human-like arm motion generation

Human voluntary movements are the physical linkage that the brain uses to communicate with the external world (Schwartz, 2016). The movement generation system embodies the intentionality of a subject as the desire to change a portion of the surrounding environment, and the prediction of an action, which is based on learned internal models. The sensations and changes resulting from the execution of a movement feed-back to the nervous system that might adjust the motion generation mechanisms. There is a complete specification of upper-limb trajectories before the beginning of the action, but there is also a continuous motion redefinition as an arm moves (Schwartz, 2016). This implies that planning and controlling arm trajectories can be considered independent and interconnected stages of human-like arm motion generation (Burdet et al., 2013). The planning stage of volitional movements, in particular, is featured by optimality requirements when accurate prediction is necessary to satisfy the specifications of goal-directed actions (Flash and Hogan, 1985; Uno et al., 1989). As part of a more general feed-forward control policy, human motor planning also evokes “internal models” that are constructed by learning from prior knowledge and offer more precise expressions of an intended action (Schwartz, 2016). Being part of a combined human motor framework, it is important to analyse optimization, learning and control techniques that have been proposed as solutions for human-like arm motion generation.

In the last decades, optimality principles and models have been studied and empirically developed under biological and psychological perspectives as a manifestation that features the evolution of the human motor system (Engelbrecht, 2001; Ivaldi et al., 2012; Todorov, 2004). The 37.04% of the reviewed papers present different optimization-based solutions that consider the minimization of typical costs in human upper-limb movements. For instance, Albrecht et al. (2011) applied bilevel optimization to imitate human reaching movements that were tracked and clustered during the execution of daily kitchen routines. In specific, a human-like arm trajectory is obtained by optimizing on two levels to address the replication of a wide range of movements. The resolution of a low-level problem comprehends the minimization of a weighted combination of three cost functions that have been experimentally observed in humans: a hand jerk cost (Flash and Hogan, 1985), an angular jerk cost (Rosenbaum et al., 1995) and a torque change cost (Uno et al., 1989). The resolution of a high-level problem finds the correct combination of the weights to minimize the distance between the optimal solution and the recorded human data in a single cluster. The availability of weighting factors permits adapting the cost according to the diverse tasks that identify the clusters. Simulations and experiments with the robot *iCub* (Figure 2.8b) demonstrated that the proposed method can correctly imitate human reaching movements better than the single optimal criteria alone, even though similarities can be limited by the embodiment of a robotic device.

Xia et al. (2014) introduced an analytical method to identify human-like swivel angles in the feasibility regions of anthropomorphic robotic manipulators. Starting from the human-like relationship between the

swivel angle and given end-effector poses for 7-DOFs arms (Zanchettin et al., 2011), the redundancy of a posture can be solved by optimization when feasible regions of swivel angles values are computed. Xia et al. (2014) found these regions that constrain an optimal human-like arm configuration to respect the physical joint limits of an anthropomorphic manipulator. Given a desired trajectory of the end-effector, a feasible optimal swivel angle allows the analytical generation of a human-like joints-space trajectory in a closed form. If the optimization fails and a feasible solution cannot be found, then an optimal swivel angle is heuristically computed along the desired operational-space path. Simulations showed that the proposed method can provide an optimization-based human-like redundancy resolution for 7-DOFs manipulators, which can also deal with infeasible solutions in the maximum reachable workspace of a robot.

In our previous studies, a human-like arm motion planning that minimizes the angular jerk of collisions-free trajectories was described (Gulletta et al., 2015; Silva et al., 2011; Silva et al., 2015). Taking inspiration from psychological findings in human prehension, (Rosenbaum et al., 2001) the proposed method is based on the resolution of two non-linear constrained optimization problems. One for the selection of a goal posture, which ensures a correct collisions-free posing of an end-effector, and another for the selection of a bounce posture, which guarantees naturalistic obstacles-avoidance on the entire generated path. Then, the selected goal and bounce postures are integrated to compose a minimum angular jerk trajectory that endows a humanoid robot with the capability of manipulating objects in scenarios cluttered with obstacles. The goal and bounce postures selection problems are formulated in AMPL (Fourer et al., 1989) and are solved by IPOPT (Wachter and Biegler, 2005), which implements an interior-point filter (Nocedal and Wright, 1999) for large-scale non-linear optimization. Simulations and experiments on assembly tasks with the robot *ARoS* (Figure 2.8a) demonstrated the feasibility of the proposed solutions in generating human-like single-arm movements (Silva et al., 2011), as well as bi-manual asynchronous (Silva et al., 2015) and synchronous (Gulletta et al., 2015) movements.

Training and learning models and processes in human arm-hand motion can describe how adult motor behaviour emerge (Burdet and Milner, 1998) and how the information of actions can be memorized and stored in internal models for generating prehension and manipulation (Rosenbaum et al., 1995; Rosenbaum et al., 2001). The 22.22% of the include studies introduced techniques for learning and generalizing from the observation of human patterns of motion during the execution of particular tasks. For instance, Park and Kim (2010) described an optimal database construction of human re-targeted motion to learn arm motion primitives for a real-time human-like trajectory planning. Particularly, an Evolutionary Algorithm (EA) is applied as a global optimizer to iteratively select minimal joint torques trajectories among a collection of captured human motion data. The latter are clustered according to task-related conditions, which need to be respected for the correct accomplishment of an intended action. Within these conditions, the principal components of motion are extracted for a dimensionality reduction and for reconstructing new

evolved movements that locally minimize the joint torque along the movement duration. Experiments with the humanoid *MAHRU* (Figure 2.8i) demonstrated that the proposed method is capable of learning minimal torque arm trajectories and generate human-like motion in real-time by reconstructing a movement through its principal components under given boundary conditions. The general performance decreases as the size of the clusters in the database increases because the selected principal components may not fully describe the variance of the captured motion.

Taking inspiration from human adaptation to perturbations in reach-to-grasp movements, Shukla and Billard (2012) introduced a coupled dynamical system to control the arm-hand interdependency during task execution. In specific, a Programming by Demonstration approach (Billard et al., 2008) is followed to learn the explicit coupling between the hand and the fingers motion during unperturbed and perturbed grasping actions. Human movements from different subjects were recorded, transferred to a robotic embodiment and modelled as Gaussian Mixture Models (Bishop, 2006) for estimating operational-space position and velocities, which maximise the likelihood of the generated demonstrations. In particular, three joint distributions are learned to form the proposed coupled dynamical system (CDS): one encodes the dynamics of the hand transportation, one encodes the fingers motion and one encodes the state of the fingers inferred from the hand position. The norm of the distance between the hand position and its target represents the coupling function to deduce the state of the fingers, which activates a controller to update the values of the finger joints. When perturbations occur, tuning the controlling parameters can regulate the speed and the amplitude of hand reopening, which can either adhere to human-like requirements or surpass human physical limitations to cope with fast adaptation demands. Simulations and tests with the robot *iCub* (Figure 2.8b) showed the necessity of having an explicit arm-hand coupling for a successful grasping when, for example, a target suddenly changes after the onset of the motion and when a grip type needs to be substituted.

A deep neural network approach was proposed by Su et al. (2019) to optimize the redundancy of anthropomorphic manipulators and reproduce the human-like features of human recorded motion. In particular, target poses and swivel angle motion are mapped from the given demonstrations with a network of five hidden modules, which apply convolution, a drop-out module to prevent over-fitting and a regression layer. After training, the regression model can predict human-like swivel angle kinematics from novel target end-effector poses. Then, the predicted swivel angle motion is fed to an inverse kinematics controller to reproduce human-like elbow positions through the redundant variables of an anthropomorphic manipulator. Simulations with a 7-DOFs robotic arm showed the capability of replicating human-like postures in tracking tasks. Additionally, compared with other artificial neural network based models, the proposed learning approach exhibited a higher reconstruction accuracy, a less predicting time and a greater noise robustness.

A human-like task planning framework for arm motion generation with decision-making capabilities was proposed by Zhao and Wei (2017). Human reaching and grasping movements were captured to extract variations of the motion on different initial positional conditions of the hand. Then, a hierarchical planning strategy is defined in relation with these conditions that may trigger three different human-like arm planning procedures. Assuming target arm postures (Xie et al., 2011), the minimization of the total potential energy of the arm is achieved for the generation of simple reaching movements (Zhao et al., 2014). When distance conditions with the target are met, a second planner provides grasping movements by also considering minimal displacements of the elbow positions in the robotic trajectory. Moreover, a third planning solution balances the TPE minimization with minimal wrist angles displacements when orientation conditions with the target are met. The framework is endowed with automatic decision-making as a result of the training of Bayesian networks (Bishop, 2006) to describe the variabilities of the recorded human motion data. The Bayesian networks provides a probabilistic graphical representation of the three possibly adopted planning strategies on the basis of conditional dependencies, which are derived from the analysis of the mutual influencing information across the motion states in the hierarchy. Experiments with a 7-DOFs robotic arm showed similarities with human motion in static and dynamic situations and validated the propose automatic decision algorithm in predicting the recognition of a motion state change.

Studies in humans have shown that there is a complete trajectory specification before the onset of, for example, reaching and drawing movements. However, this specification continues throughout the motion and shows adaptations to unpredicted task-related variations (Schwartz, 2016). Although visual information and, more generally, sensory feedbacks necessarily close a controlling loop, some evidences on humans concerning fast motion adaptation suggest the existence of feedback-free internal controlling models that emerge from experience (Flozano et al., 2014; Schwartz, 2016). However, accuracy requirements seem to be achieved by visually detecting a hand moving, which suggests motion modifications based on physical and environmental sensations (Schwartz, 2016).

The 29.63% of the included studies introduced closed-loop controllers to cope with fast unpredicted changes that may occur in the workspace of a robot and to allow a compliant physical interaction with the environment. For instance, starting from the Virtual Spring-Damper controller introduced by Arimoto and Sekimoto (2006), Bae et al. (2011) took inspiration from the human muscle tension effect under gravity to increase energy efficiency and conform to a more repeatable human-like motion. Specifically, a torsional spring effect in the joints-space is added to the previous proportional derivative closed-loop controller, which ensures a correct trajectory tracking in the operational space. Feeding back a joints-space signal permits to privilege arm configurations, which are usually adopted by humans to minimize the tension of the muscles under gravity. Experiments with a 7-DOFs redundant manipulator revealed that the

introduction of the muscle tension effect can reduce the average power consumption and trigger joints-space repeatability, which would otherwise be lost after a few repetitions. Moreover, it was shown that the proposed human-like controller is featured by compliant behaviours because it can promptly absorb external disturbances and maintain good operational-space tracking capabilities and joints-space repeatability. In the trail of the virtual spring-damper hypothesis, Bhattacharjee et al. (2011) proposed a controller to simulate human-like arm-trunk regularities during the execution of reaching and pointing movements. For modelling purposes, arm inertial and mass properties in human healthy subjects are transferred into a 4-DOFs manipulator. The base joint is exclusively considered as a human-like trunk in bending motion, while the other joints execute reaching tasks on the sagittal plane of motion. In order to reflect human muscles characteristics, the control law is featured by sinusoidal time-varying joint damping and joint muscle stiffness, which ensure a smooth convergence of the end-effector towards its target with human-like quasi-straight paths and symmetrical bell-shaped velocity profiles. The resulting control signals are also low-pass filtered to simulate the delays of brain-muscles communications. Results on simulated environments demonstrated that the proposed controller can replicate the kinematic characteristics of human arm-trunk reaching motion and imitate human-like arm self-motion, which emerges from the null-space velocities of the joints.

Stefanovic and Galiana (2014) introduced a spinal-like regulator for replicating behavioural patterns that are typical in volitional reaching movements. In particular, this approach finds its foundations on the imitation of the centres of pattern generation that exist in the spinal cord and couple brain activity with the spinal motor areas. For this reason, the proposed controller is based on spinal-like circuits, which can exhibit intelligent functions that resemble internal models and reflexive reactions to external perturbations. The applicability of the proposed biological-like controller to multi-joint artificial limbs is tested using a two-links simulated robotic arm, which executes planar reaching tasks on different ranges of the available workspace. The results showed the capability of the proposed method of scaling the operational-space trajectories on spatial and temporal levels to fit with human-like arm motion requirements. In specific, the inherent spinal gains of the controller can be tuned to change the end-effector path curvatures and speeds to generate quasi-straight trajectories with bell-shaped velocity profiles. Additionally, the regulator can successfully absorb pulse and sustained perturbations and adaptively track moving targets to return and maintain a human-like trajectory profile.

With the purpose of mimicking the human joint coordination at a behavioural level, Togo et al. (2016) took inspiration from the uncontrolled manifold (UCM) concept (Scholz and Schöner, 1999) to introduce a feedback controller capable of reproducing the trial-to-trial joints-space variance of reaching movements. This variance of redundant motor elements (i.e., joint angles) is quantified by a UCM component, which is orthogonal to another component (ORT) that directly affects the performance variables (i.e., coordinates in

the operational space of a manipulator). The proposed “UCM reference feedback controller” regulates the task-relevant elements (the ORT component) in respect of the minimum intervention principle introduced by Todorov and Jordan (2002) (Figure 2.15). A one-dimensional target tracking task with a simulated

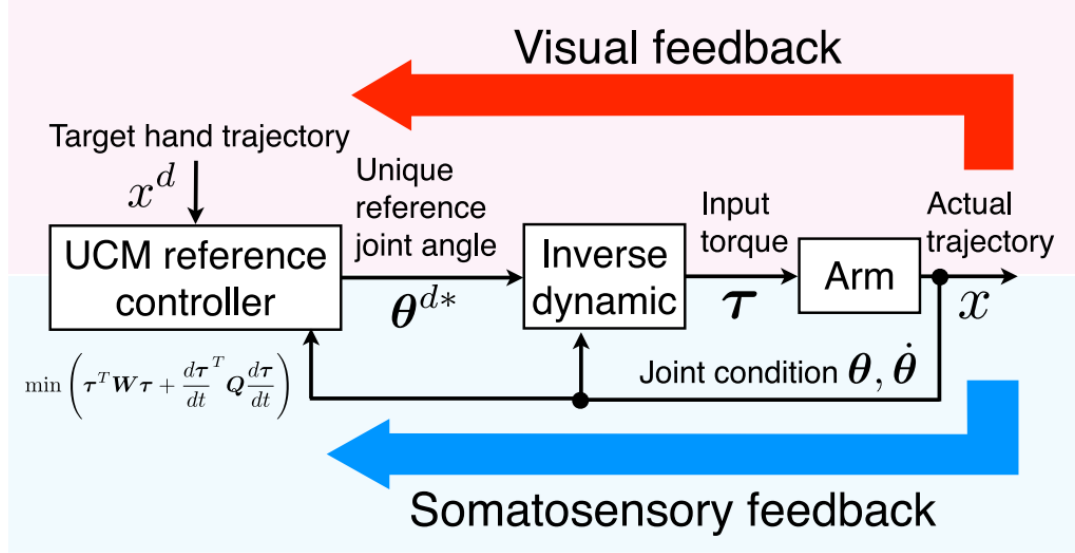


Figure 2.15: Concept of the UCM reference feedback controller (Togo et al., 2016).

3-DOFs arm is addressed for tests and comparisons with captured human reaching motion. The minimum jerk criterion with null boundary conditions (Flash and Hogan, 1985) is used to generate human-like operational-space trajectories, which are then tracked by the proposed human-like regulating method. In conformity with the minimum torque change principle of human reaching movements (Uno et al., 1989), a task-constrained optimal problem is analytically solved to provide a reference configuration and, through inverse dynamics, a torque signal for the arm. Visual information of the actual trajectory is fed back to the controller with added normal distributed noise that simulates human imperfections in the visual perception of moving targets. Additionally, the proposed controller is supplied with a somato-sensory feedback for internally monitoring the joint position and velocities during motion. Normal distributed noise signals are also added to the joints torque signals and to the optimal configuration references to simulate signal-dependent noise of motor commands and the influence of visual perception uncertainties on the variability of the joints angles, respectively. The efficacy and the capability of reproducing human-like joints coordination of the proposed uncontrolled manifold regulator were statistically assessed by simulations and comparisons with captured human reaching movements.

From the analysis of the included studies emerges a wide variety of optimization, learning and controlling methods that have been fairly applied for human-like arm motion generation. However, a common motor framework capable of an efficient modular integration seems to be still missing, even though

roboticists have often referred to the genesis of movements as a linked functional decomposition of these technical features (Burdet et al., 2013; Schaal, 2002). The proposed human-like arm motion planning and learning methods are generally based on task-related human captured data. They involve optimization techniques on a global level in the attempt of clustering motion patterns, and also on a local level in respect to human motion minimization principles. Learning approaches take advantages of dimensionality reduction mechanisms to simplify the generation process and extract synergies for a motion reconstruction in unseen environmental situations. Additionally, decision-making systems have also been presented as the result of learning from captured human data to select the most appropriate movement planning method. Although collections of human motor features guarantee an authentic source of knowledge and an unequivocal tool of comparison, these optimization and learning techniques may suffer from a lack of generalization because of internal feed models that store experience in a categorical and human-dependant manner. Controlling human-like trajectories have also been studied to optimize the input motor commands of robotic manipulators. The reviewed controllers are of biological and behavioural inspiration and have been tested with devices of few degrees of freedom. Therefore, their application on complex human-like redundant circumstances remains unaccounted. Additionally, the regulated trajectories are often generated on-line without any given reference from prior planning stages, but the human Central Nervous System seems to generate optimized motion outputs close to a final motor solution before the onset of a movement (Schwartz, 2016; Togo et al., 2016). Learning of control parameters and internal models are also absent in the analyzed human-like arm motion regulators, despite the fact that training and practice find neuroscientific and psychological bases in planning and controlling voluntary movements (Kawato et al., 1987; Kawato, 1999).

2.2.4 Discussion

A literature review on the most recent techniques of human-like arm motion generation has been presented. The analysis included 54 papers that were firstly classified according to the sources of knowledge. The vast majority of the papers was found in Scopus, immediately followed by the IEEEExplore, Web of Science, and, with minor contribution, ACM Digital Library. The equipment of the included studies was roughly equally composed by simulators, robotic manipulators and anthropomorphic platforms. The latter devices are of particular impact on motion planning solutions because provide a significant level of anthropomorphism that can positively influence interactions with humans (Duffy, 2003; Strait et al., 2017). The large majority of the reviewed papers proposed global methods that operate with kinematic variables and address the generation of trajectories in the operational space of robotic manipulators. This result might be due to the fact that a globally-considered static environment can provide solutions in a low dimensional space and a more accurate match with kinematic human-like features (Milner, 1992; Morasso, 1981; Rosenbaum et al.,

2001). However, such methods might be reductive and risk to ignore human-like characteristics of the joints space, such as synchrony and bell-shaped angular velocity profiles (Breteler and Meulenbroek, 2006). Dynamics variables are often the focus of biologically-inspired solutions, which have been poorly addressed due to the difficulties that arise in transferring the activities of the muscles into robotic manipulators. Local human-like arm motion generation methods have also been scarcely proposed, but they can introduce the capability of simulating human reflexes and react to changing scenarios during the execution of a movement. However, these techniques only partially consider the workspace and, therefore, may fail to achieve human-likeness in complex tasks.

Roboticists have considered a variety of human motor control principles as inspirational and evaluational tools for proposing techniques of human-like arm motion generation. The included studies showed that researchers have mainly focused on biomimetic and human-robot mapping approaches, which can ensure a strong resemblance with human motion that is acknowledged as an objective reference of an action. In particular, human-robot transferring techniques can guarantee real-time execution of the movements, which is essential during tele-operation. However, most of these methods need expensive and complicated equipment for capturing human movements, databases to collect them and post-processing for features extraction. Additionally, these techniques generally lack of generalization because usually work for a limited range of goal-directed tasks. Similar issues are also experienced by learning-based methods to teach specific tasks to a robot by imitation or training. Although learning from observations ensures a stable repeatability and a degree of adaptation to external perturbations, generalization is seldom achieved. The kinematics of the obtained end-effector trajectories have often been compared to the hand path and hand velocity profile of human reaching to show similarities and divergences between robotic and human movements. Most of these particular solutions were developed for single-arm motion and based on human-like optimization principles that minimize energy under tasks-related constraints. Manipulation in the workspace of a robot is often neglected when kinematics is assessed because of the inevitable complications that pick-and-place tasks would import into the analysis of human-likeness. The majority of the proposed methods were not tested with qualitatively assessments to understand the perception of human observers. However, there are many scientific shreds of evidence demonstrating that, for pleasant human-robot interactions and collaborations, the movements of the robot have to be perceived as natural, predictable, and showing the intention of the underlying action without any verbal communication (Bicho et al., 2011b; Bisio et al., 2014; Dragan et al., 2013; Kupferberg et al., 2012).

The smoothness of a trajectory has often been measured in relation with the third derivative of position, i.e., the jerk, which is a compact and intuitive index of human-likeness. However, smoothness is not exclusively connected with jerk, but it must be assessed more comprehensively with other validation systems, such as, for example, the number of movement units (Chang et al., 2008). The application of the two-third

power law (2/3-PL) is relatively simple for the generation of biologically-inspired planar movements, like writing or drawing. However, the 2/3-PL has shown to introduce systematic errors on the complete shape of a trajectory and needs to be extended to explain regularities of three-dimensional reaching movements (Pollick et al., 2009). Although path planarity in the operational space may significantly simplify a planning problem, this human motor principle should emerge from other independent human-like motor principles instead of being a constraint of the generation process. Otherwise, there is a risk of complicating a planning problem and decrease the overall performance. The repeatability of motion on imitating observed human movements can certainly increase predictability and simplify learning, but it can seriously undermine generalization on different tasks. With two optimization processes in sequence, the Spatio-Temporal Correspondence (Gielniak et al., 2013) can negatively affect real-time interactions with humans, but it is a versatile tool that can be used to either assess or generate human-like body trajectories from one single sample. Human-like arm motion has also been achieved by the RULA-driven technique (Zacharias et al., 2011), which introduced an upper limb assessment of the inverse kinematics and the sampling of the search space to enhance ergonomics, but ignored human-like time parametrizations.

Most of the included studies addressed single-arm reaching movements, while human-like dual-arm applications are emerging and pick-and-place tasks still deserve more attention. The simple nature of motion without manipulation and the vast literature concerning the psychology of human reaching have probably encouraged roboticists towards this direction of investigation. Replicating human writing, drawing, and rhythmic movements revealed to be a small research branch, which seems to be isolated because not of easy integration with more classical methods of picking, placing, and reaching motion generation. The prevention of self-collisions and the avoidance of obstacles in the workspace of a robot were rarely considered in the reviewed papers, even though these are necessary features for interacting devices in human-centered environments. Additionally, many of the proposed solutions ignored a human-like kinematic analysis during the avoidance of obstacles and addressed collisions prevention under simplifying assumptions, which reduce their range of applicabilities.

Roboticists have usually defined the genesis of human-like motion as a connected system of different functional modules (Burdet et al., 2013) where optimization, learning, and control are integrated. However, the included studies do not refer to a common framework of motor generation for optimizing human motor models, learning internal models and control on external disturbances. On a local level, human motor minimization principles are applied, while learning techniques are used for clustering motion on a global level, for extracting regularities of captured human movements and for implementing decision-making settings. Studies have shown that the synthesis of motor solutions happens in the Central Nervous System before execution (Schwartz, 2016). However, the reviewed controllers act in the absence of prior optimized trajectories that may guide the process of on-line regulation. Moreover, these controllers are applied to a

few degrees of freedom and remain to be tested on more complex redundant manipulators and interacting tasks.

The contribution of this PhD project enriches the current state of art by facing the open challenges and addressing inconsistencies in the domain of robotic human-like arm motion generation. Particularly, a three-modules framework is introduced to provide a common ground where optimization, learning and controlling can be integrated and resemble human functioning. An optimization-based motion planner is presented and showed to be capable of generating human-like upper-limb movements for a wide range of uni-manual and bi-manual tasks. Importantly, a human-like obstacles avoidance that can be consistent and reliable on complex tasks has revealed to be missing. Therefore, the generation of human-like collisions-free arm trajectories is addressed and tested in interacting scenarios with human partners. A learning module is studied and developed to support the optimization stage in solving predictable variations of known planning problems. Internal models are fed and evolve for predicting motion solutions and ease optimization before the onset of an action. Unlike many learning techniques applied to generate human-like arm movements, this module does not depend on observations of human movements to be trained, but it can acquire experiences of action trial after trial. A feed-back loop controller is also developed to track the planned human-like arm trajectories and use the redundancy of manipulation to satisfy priorities of secondary importance, such as on-line obstacles avoidance, singularities avoidance and respect of joints physical limits. This controller maintains the human-like characteristics of the reference trajectory and adds the capability of reacting to unpredictable external perturbation that may influence the pose of targets and of objects in the workspace. Unlike the reviewed techniques of human-like regulation, this controller disposes of a reference of human-likeness and can be applicable for the execution of complex tasks during the interaction with humans subjects.

Chapter 3

Human upper-limb motor behaviour

In this chapter, it is given a detailed description of the human upper-limb motor behaviour that has been inspiring for the human-like motion planning system, which is the main contribution of this thesis. section 3.1 introduces the posture-based motion planning model, which provides important insights concerning motion planning of reach-and-grasp movements. In section 3.2, it is shown how human upper-limb movements are featured by different phases during pick-and-place tasks and the kinematics of the hand changes with accuracy between these phases. In section 3.3 some of the kinematical measures that have been used in psychological studies are presented because they are also considered, in the next chapters, for the evaluation of the degree of human-likeness that features the planned movements. section 3.4 provides a brief overview concerning the functional principles of bi-manual coordination in humans, which inspired the integration of human-like synchronous bi-manual motion planning for humanoid robots.

3.1 The posture-based motion planning model

In the last decades cognitive psychologists have studied human upper-limb motor behaviours in different manners. The posture-based motion planning model, which was firstly described by Rosenbaum et al., 1999, provides a computational approach to motion generation that is easily accessible to roboticists and programmers. Reaching, grasping and aperture of the fingers during prehension have been mainly simulated by this model, which can reproduce the basic kinematics features of a human upper-limb during motion. The substantial assumption concerns the fact that any task is defined as an organized hierarchy of constraints (Rosenbaum et al., 1999; Rosenbaum et al., 2001). According to Rosenbaum et al., 2009, the ranking of constraints from the most to the least important is based on an implicit process of winnowing, which is often approximated with an optimization process that brings to the minimization of a travel

cost during motion. As the authors of Rosenbaum et al., 2001 showed, Flash and Hogan, 1985 firstly hypothesized that upper-limb movements may obey a "minimum jerk" principle, where jerk is the third time derivative of position. Particularly, the authors discovered that reaching movements are featured by a hand tangential velocity profile that is bell-shaped and that this characteristic minimizes the mean squared of the jerk over the duration of a movement. While the "minimum jerk" principle only refers to kinematics, Uno et al., 1989 hypothesized that upper-limb movements might obey a "minimum torque" principle, which also considered the dynamics of the motion. The posture-based motion planning model is entirely referred to kinematic variables because they are sufficient to study human upper-limb motion at a computational level. However, cognitive psychologists are willing to integrate dynamics into the model in the future work (Vaughan et al., 2006).

The authors of (Rosenbaum et al., 2009) affirmed that the posture-based motion planning model was developed and tested in the domain of object manipulation also because, in such a task, there is the reflection of the intentionality of the motion. The way an object is grasped and the position of the grip implicitly declares the future use of the object in the current task. This phenomena is particularly evident in the end-state comfort effect (Rosenbaum, 1990). The end-state comfort effect is the attitude of humans to assume initial awkward arm posture to satisfy the need of having a future posture that is more comfortable according to the final intention in a task. A typical example can be seen when a waiter grasps an inverted glass on a table with an awkward thumb-down grip type because the goal is to put the glass upright, which means ending the movement in a more comfortable arm posture while the glass is being filled (for more details see Rosenbaum, 1990). These findings support the hypothesis that upper-limb motion is mentally planned starting from a goal posture, which regards the purpose of a particular task, and then a trajectory is generated towards it (Rosenbaum et al., 2001; Rosenbaum et al., 2009). The selection of this goal posture and an overview of a grasping model are illustrated in Figure 3.1.

The model has been embodied in a stick figure and firstly applied in bi-dimensional tasks (Meulenbroek et al., 2001; Rosenbaum et al., 2001). However, in Vaughan et al., 2006 it has been successfully extended for three-dimensional applications. In reach-and-grasp movements a target on the operational space is defined and the upper-limb is in its start posture (Figure 3.1a). Then, a set of stored postures are identified according to that task that is going to be performed (Figure 3.1b). In (Figure 3.1c), a best retrieval is selected for the selection of the correct goal posture that brings the hand onto the target (Figure 3.1d). With a similar process a bounce posture is selected to serve obstacle avoidance during the entire motion (Figure 3.1e). Finally, the resulting trajectory can be executed (Figure 3.1f).

The following sections describe in details each step of the planning procedure. Specifically, in section 3.1.1 the process of selection of the goal posture is described, while in section 3.1.2 that one related

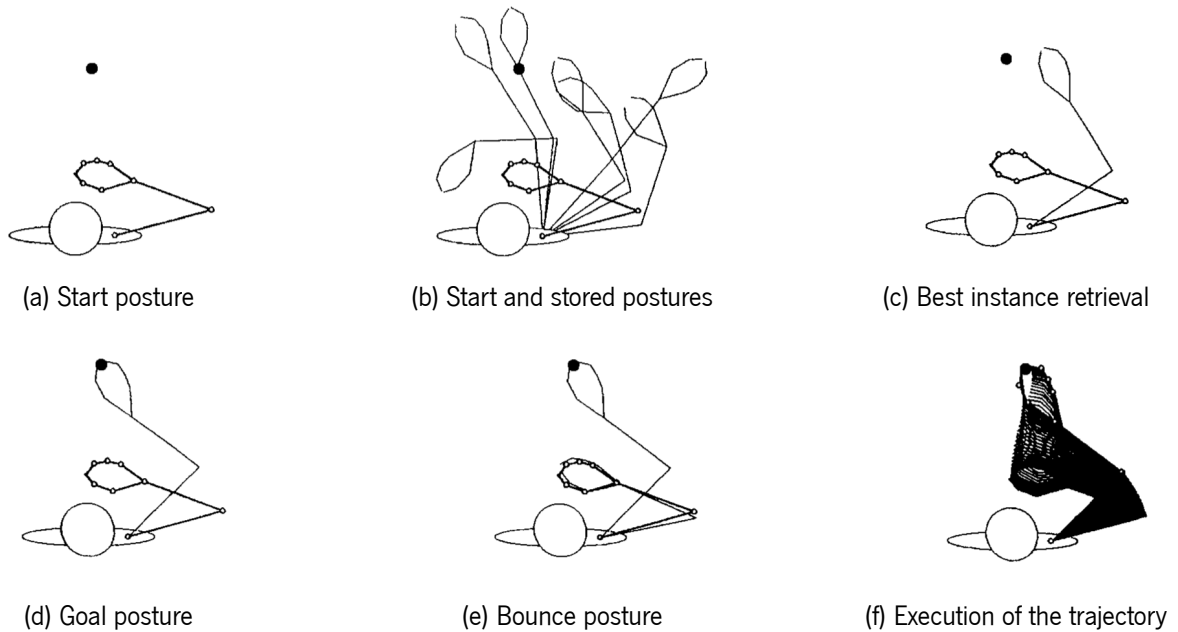


Figure 3.1: Overview of the grasping model (Rosenbaum et al., 2001).

to the bounce posture is presented and the composition of the complete movement is outlined. Section 3.1.3 refers to the formulation of the duration of the planned movement and section 3.1.4 concerns the kinematic analysis of some movements simulated by the posture-based motion planning model.

3.1.1 Selection of a goal posture

According to the theory of the posture-based motion planning model, human beings generate of goal postures starting from the evaluation of previously learned stored postures with respect to a hierarchy of constraints corresponding to a current target (Rosenbaum et al., 2001; Rosenbaum et al., 2009). This process is also called *diffusion* until a time runs out according a certain deadline. The term *diffusion* is adopted because the generation of postures involves the random dispersion in the joint space that starts from the most promising posture (Figure 3.1c). This starting point is essentially task dependent (Rosenbaum et al., 1999; Rosenbaum et al., 2001).

In reach-and-grasp movements, the selection of a goal hand posture is a task with higher priority than others. The main reason lies on the fact that the grip type during object manipulation encodes the intention of a movement (Rosenbaum, 1990; Rosenbaum et al., 1999). Therefore, as described in Rosenbaum et al., 1999, the distance between the thumb and the index finger must approximate the diameter of a cylindrical object that is being hold. Consequently, a goal arm posture is found to permit the hand to assume its goal posture. Additionally, the selection of a goal arm posture must avoid collisions with any object in the workspace, including the object that is being grasped during prehension. With less importance, the

selected goal arm posture must minimize the travel cost of the motion. For this reason, the movement that goes from the start to the goal posture (*direct* movement) has a velocity profile in the joint space equals to $v = \sin^2(t)$, $0 \leq t \leq \pi$, which closely approximates a trajectory with minimum jerk (Rosenbaum et al., 2001; Vaughan et al., 2006). The computed *direct* movement is then mentally checked for collisions and, if no collision is anticipated, the movement is executed. On the contrary, obstacle-avoidance is dealt with the selection of a bounce posture, which is also called via posture.

3.1.2 Selection of a bounce posture

The evaluation of possible collisions is performed mentally by checking for spatial overlap of the body, the start posture, the goal posture and few postures adopted during the movement with the objects in the scenario to be avoided. When collisions are anticipated a bounce or via posture is selected. It defines a virtual via point in the joint space and serves as a sub-goal body position for a *back-and-forth* movement, also called bounce movement (Rosenbaum et al., 1999; Rosenbaum et al., 2001; Vaughan et al., 2006). The *back-and-forth* movement has a velocity profile in the joint space equals to $v = \sin(t) \cdot \sin(2t)$, $0 \leq t \leq \pi$ and transports the upper-limb from its start posture to the selected bounce posture and back. The complete movement is the result of the superimposition of the *back-and-forth* movement on the *direct* movement (Figure 3.2).

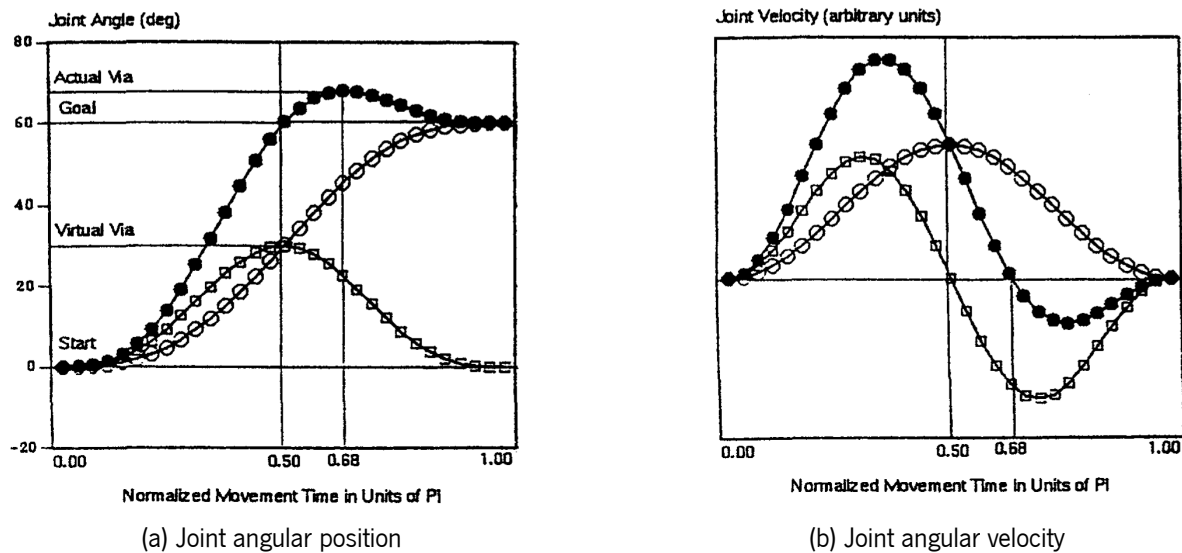


Figure 3.2: Superimposition of angular position and velocity profiles (Rosenbaum et al., 2001). ($\circ$) is the direct movement, ($\square$) is the back-and-forth movement and ($\bullet$) is the composite movement.

All the joints participate synchronously to the movement. They start and stop their motion together

with a bell-shaped angular velocity profile (Rosenbaum et al., 1999). The main reason behind the superimposition of these two movements relies on the fact that moving the joints to a bounce posture and back, while the *direct* movement is on the way, allows obstacle-avoidance without changing the selected goal posture. Only the path of the movement differs and the actual via point, which marks when the bounce posture is reached, occurs generally later than its virtual counterpart (Figure 3.2).

The search procedure of the bounce posture is essentially equal to that one for the selection of the goal posture. The reduction of the travel-cost is considered, but with less importance. The process starts with the selection of the most promising bounce stored postures, which are evaluated with respect to constraints applied to the complete composite movement. Once a bounce posture is computed, the superimposition of the *direct* movement (from the start posture to the goal posture) with the *back-and-forth* movement (from the start posture to the bounce posture and back) provides a collision-free trajectory towards the target of the hand.

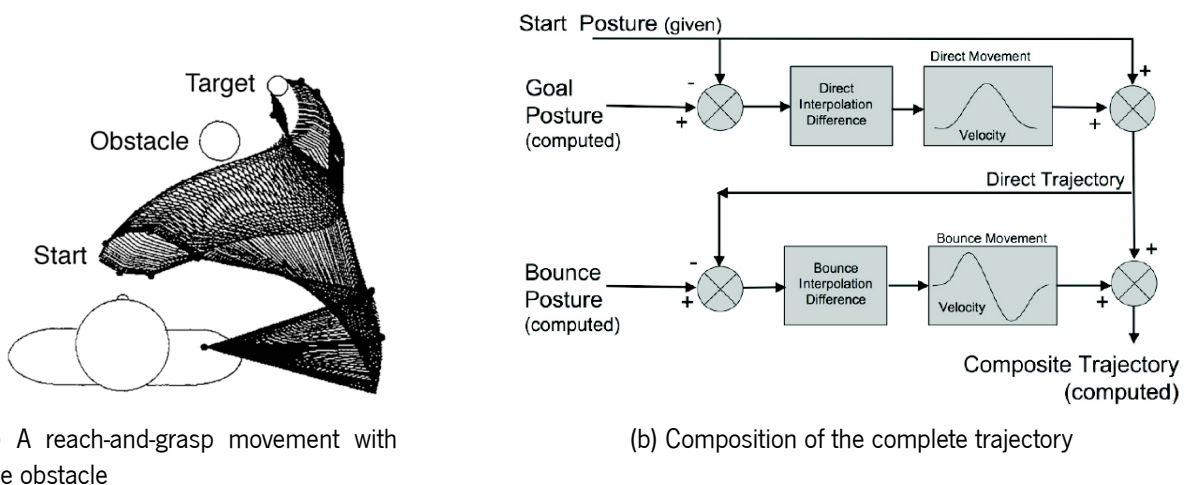


Figure 3.3: The composition of the direct and the bounce movements allows to avoid obstacles and the object to be grasped during prehension (Vaughan et al., 2006).

An illustration of the described trajectory generation and of the obstacle-avoidance mechanism is given in Figure 3.3. The circumvention of an obstacle between the body and the target is realistically supported by the fact that the joints go through the via posture with a zero-crossing velocity profile (Rosenbaum et al., 1999). As it is mentioned in Rosenbaum et al., 2001, the feeling of how this method works is given considering the act of bringing the arm from under a table to the top of the same table. This motion implies a detour that can be obtained by superimposing a straight-line motion from the start to the goal hand position, which would inevitably collide, with a movement from the start to a bounce position at the level of the table and back.

3.1.3 The duration of a movement

One of the claims of the posture-based motion planning model is that it can specify the duration of a planned movement endogenously (Rosenbaum et al., 2001). In particular, the optimal time for the movement of the j^{th} joint is given by Equation 3.1.1 (Rosenbaum et al., 1995)

$$T_j^*(\alpha_j) = k_j \cdot \ln(\alpha_j + 1) \quad (3.1.1)$$

, where k_j is a non-negative real number, which serves as an expense factor, and α_j is the absolute angular displacement of the j^{th} joint.

The travel cost of the j^{th} joint to cover the absolute displacement α_j in a time T_j (not necessarily equals to T_j^*) is given by Equation 3.1.2

$$V_j(\alpha_j, T_j) = \left(\frac{k_j \alpha_j}{r} \right) \cdot \left(1 + \frac{[T_j - T_j^*(\alpha_j)]^2}{s^2} \right) \quad (3.1.2)$$

, where r is the unit of absolute angular displacement and s is the unit of time.

Consequently, the travel cost associated to a posture can be computed by summing the costs of the joints as it is shown by Equation 3.1.3.

$$V_p = \sum_{j=1}^n [V_j(\alpha_j, T_j)] \quad (3.1.3)$$

The optimal time for a posture T_p is that time that minimizes V_p . Therefore, setting $dV_p/dT_p = 0$, the optimal time can be calculated and it is described by Equation 3.1.4. This optimal time replaces T_j for every joint during its motion.

$$T_p = \frac{\sum_{j=1}^n [k_j \alpha_j T_j^*(\alpha_j)]}{\sum_{j=1}^n [k_j \alpha_j]} \quad (3.1.4)$$

The duration of a movement may also depends on the accuracy required by the task that is being performed. This feature of human motion was firstly studied by Fitts, 1954, which brought to the definition of the Fitt's Law. Interestingly, the posture-based motion planning model finds support in the extension of the Fitt's law to three-dimensional obstacle-avoidance (Vaughan et al., 2010). The authors of Vaughan et al., 2010 synthesized a formula of the movement time that takes the amplitude of the *direct* and of the *back-and-forth* movements as parameters. Results demonstrated that the proposed formula sufficiently approximates the evaluation of the movement time also when obstacles have to be avoided in a three-dimensional workspace.

3.1.4 Kinematical analysis of the motion

Rosenbaum et al., 2001 simulated reach-and-grasp movements with the proposed model and the results were analysed kinematically and are illustrated in Figures 3.4 and 3.5.

Results in Figure 3.4b show that the joints have a bell-shaped velocity profile during the motion illustrated in Figure 3.4a. This motion generates a tangential wrist velocity that is also bell-shaped roughly symmetric (Figure 3.4c). A function of this form minimizes the mean square of the jerk over the duration of the movement (Rosenbaum et al., 2001).

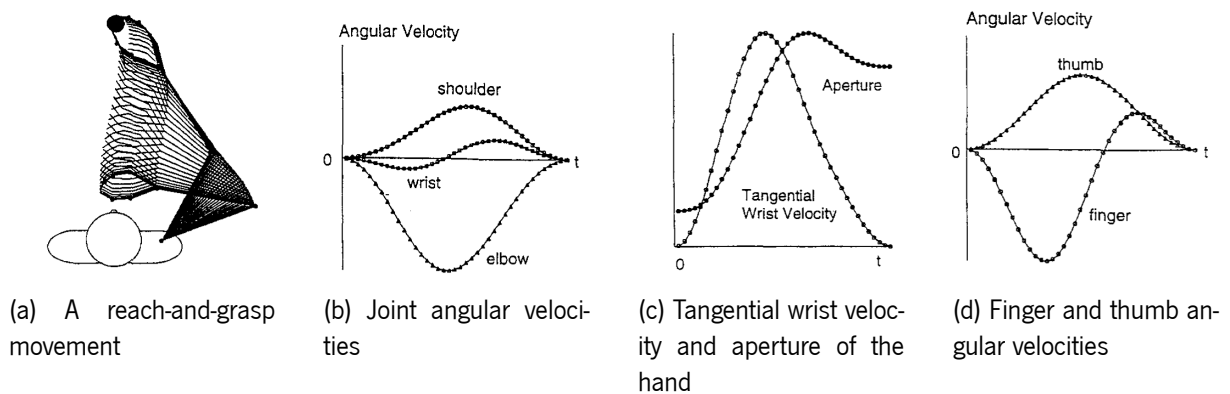


Figure 3.4: A simulated reach-and-grasp movement (Rosenbaum et al., 2001).

Figure 3.4c also shows that the aperture of the fingers reaches its peak in the second half of the movement, when the object is almost grasped and approximately the velocity of the finger changes sign (Figure 3.4d). Rosenbaum et al., 1999 attribute this kind of pre-shaping of the hand both to the obstacle-avoidance and to the travel-costs constraints. Additionally, Meulenbroek et al., 2001 deeply studied the relationship between the aperture of the fingers and the object location, object size and initial aperture. Particularly, the authors found that the maximum aperture varies approximately linearly with the object size and that it tends to occur at the start of a low-phase velocity if it is present. This specific case is illustrated in Figure 3.5c. The reach-and-grasp movement, shown in Figure 3.5a, requires a stronger detour of the wrist to avoid collisions with the object that is going to be grasped. As a consequence, the shoulder and the elbow are featured by a zero-crossing on their velocity profiles and this produces a bi-phasic tangential velocity profile of the wrist (see Figure 3.5b and compare with Figure 3.4b). This means that the requirements of collision-avoidance during prehension may give rise to double-peaked and bell-shaped tangential wrist velocity profiles (Rosenbaum et al., 2001).

More recently, a bi-phasic hand velocity profile during 3D obstacle-avoidance has been also studied by Grimme et al., 2012. In this paper, the authors carried out different experiments with participants who were asked to transport an object towards a target while avoiding one obstacle with unconstrained arm

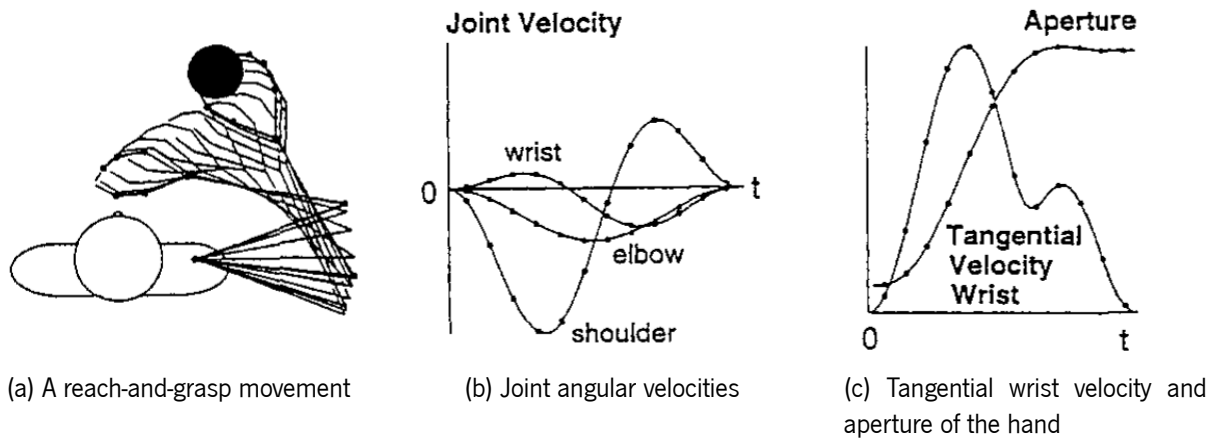


Figure 3.5: A simulated reach-and-grasp movement with a low-velocity phase (Rosenbaum et al., 2001).

movements. Results demonstrated that the planarity and isochrony of the motion are regularities. The former was measured with the torsion of the hand, while the latter refers to the fact that the movement time does not significantly change with the path length. In addition, Grimme et al., 2012 identified two movement primitives, which varied independently with the obstacle conditions. Both these primitives are accounted for the formation of the double-peaked feature that may rise on the tangential wrist velocity profile during obstacle avoidance.

3.2 Phases in a movement

Human arm movements can be studied and examined in relation with a task that is being performed. For instance, reach-and-grasp movements and the act of grasping in humans and macaque monkeys were extensively reviewed by Castiello, 2005. As fundamental feature of grasping, power and precision grips (Figure 3.6) were firstly defined by Napier, 1956 and discussed by Castiello, 2005. Specifically, the

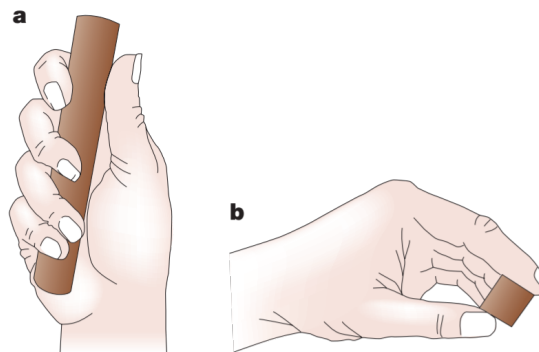


Figure 3.6: (a) Power grip between the thumb and the other fingers. (b) Precision grip between the thumb and the index finger (Castiello, 2005).

grip type goes beyond the physical rationality of a movement, because it is an expression of the underlying control principle that guides a movement. The proposed hypothesis is that a grip type shows the intention of goal-directed movements (Castiello, 2005; Napier, 1956). Additionally, studies on gaze and eye movements show that an actor and an observer of a movement predict and monitor sensory events in the same way (Flanagan et al., 2006). Studies on the neuro-imaging of grasping in humans show a similar regional cerebellum blood flow (rCBF) when grasping is performed and when it is observed. (Castiello, 2005). These results suggest that the intention of an actor's motion is understood by a process of mapping and prediction into the observer's brain.

Even though comparing studies between humans and monkeys is still unclear and controversial, the neuroscience of grasping implies the activation of different areas of the brain in both subjects (Castiello, 2005). For instance, in humans and in monkeys, lesions of the primary motor cortex (F1) produce a deficit in controlling fingers individually, while power grips seem to remain functioning. In humans, damages of the superior parietal lobe (SPL) bring to an apparent normal hand opening, but the control during grasping quickly degenerates in numerous peaks of the hand velocity profile. In monkeys, inactivation of the anterior intraparietal sulcus (AIP) and of the premotor cortex (F5) impair hand shaping during prehension and the hand posture is not appropriate for the size of the object to be grasped.

The fact that different features of reach-and-grasp movements, such as hand reaching, hand shaping and individual control of the fingers, are activated by different neural circuits in the brain, suggests the idea that these movements might result from a composition of single motor commands. The review conducted by Flanagan et al., 2006 on control strategies of the hand and eye movements during object manipulation, is in favour of this hypothesis. Flanagan et al., 2006 put emphasis on prediction as the key of the control mechanism: humans would predict hand motor commands and the consequent sensory events. In specific, contact events produce tactile, visual and auditory sensory signals that divides the control in different phases of the object manipulation task. For instance, contacts between the fingers and the object that is being manipulated, mark the end of an hand transportation phase and the beginning of an object lifting phase. Similarly, contacts between the object that is being manipulated and a surface of another object mark the end of the lifting phase and the beginning of a hand releasing phase. This temporal segmentation of a task in clear phases has been already used in robotics by Stulp et al., 2012 to propose a novel reinforcement learning algorithm. Particularly, Stulp et al., 2012 used the motion primitives paradigm in object manipulation: each action phase of the task corresponded to a motion primitive and sub-goals of the task were goal parameters of the primitives to be optimized. The algorithm proposed by Stulp et al., 2012 improved the robustness of everyday pick-and-place tasks.

In Sing Bing Kang and Ikeuchi, 1994, the temporal separation of pick-and-place tasks was identified for task programming of robotic manipulators. Sing Bing Kang and Ikeuchi, 1994 presented an algorithm

for the determination of breakpoints in human hand motion and the proposed segmentation is illustrated by the state transition diagrams in Figure 3.7. The idea proposed by Sing Bing Kang and Ikeuchi, 1994 is

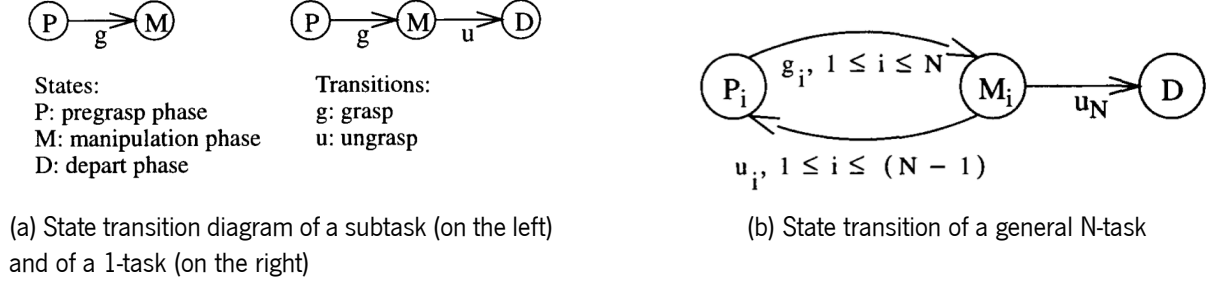


Figure 3.7: State transition diagrams of pick-and-place tasks (Sing Bing Kang and Ikeuchi, 1994).

based on the definition of a task unit, also called sub-task. A sub-task is formed by a phase of hand transportation, which is simultaneous with hand preshaping during prehension (pregrasp phase). This phase ends with a grasp transition, which is the moment when the fingers touch the object to be grasped with the intended hand posture. Then, the manipulation phase starts and it is constituted by the relocation of the object in the workspace. At the end of the manipulation phase, a sub-task is complete. However, if there is an ungrasp transition, the object is released and a depart phase starts. In this particular case, at the end of the depart phase, 1-task is defined and complete (see Figure 3.7a). Similarly, a N-task is defined as a task with N sub-tasks, where the pregrasp and the manipulation phases alternate each other N times (see Figure 3.7b).

For identification purposes, Sing Bing Kang and Ikeuchi, 1994 introduced the *volume sweep rate* as a kinematic measure for temporal segmentation of the task. The *volume sweep rate* was defined as the product between the hand tangential velocity and a polygon area formed by the fingertips during prehension. Results demonstrated the successful application of the proposed algorithm to determine sequences in pick-and-place tasks according to the state transition diagrams in Figure 3.7.

The studies of Sing Bing Kang and Ikeuchi, 1994 also showed the importance of kinematic measures in the diagnosis of action phases during human arm motion. Kinematics was critical in human motor behaviour research. In particular, the tangential velocity profile of the wrist was not only the major focus of the kinematical analysis by Rosenbaum et al., 2001 (see section 3.1.4), but it also was essential for studying accurate movements (Burdet and Milner, 1998; Milner, 1992). Milner, 1992 presented a model for the generation of movements with endpoint precision. Specifically, the participants of this study were instructed to move their hand on a plane and to place a cylindrical object from a start position to target holes with different diameters. Participants were asked to move as fast and as precise as possible to examine the effect of the trade-off between speed and accuracy on the wrist tangential velocity profile. The latter was recorded and decomposed into sub-movements with bell-shaped velocity profiles. This

process of decomposition was linear and was carried on by trial-and-error loops until matching the recorded movement. Each sub-movement was generated either when there was a zero-crossing in one of the velocity components of the wrist or when there was a significant change in the hand path. The results are illustrated in Figure 3.8.

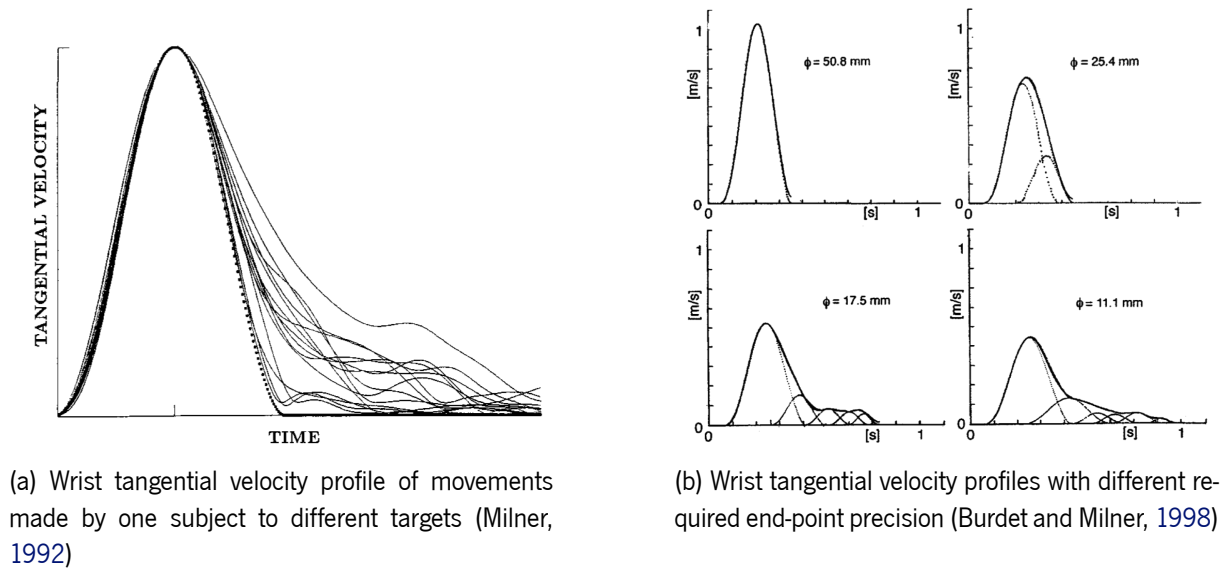


Figure 3.8: Kinematics of accurate human hand movements.

The tangential velocity of the wrist is bell-shaped and asymmetric with an acceleration phase that is similar in any trial. Its deceleration phase is significantly dependent on the level of accuracy required by the size of the hole. In particular, the duration of the deceleration phase increases as the diameter of the hole decreases. Sometimes, this effect produces a tale on the wrist tangential velocity profile in the proximity of the target position. The model introduced by Milner, 1992 can replicate the recorded movement by generating sub-movements according to the required level of precision. Milner, 1992 also stated that their proposed model can integrate the minimum jerk model (Flash and Hogan, 1985) and the minimum torque model (Uno et al., 1989), because these models are not able to predict the rise of sub-movements during the deceleration phase of the wrist.

The hypothesis that arm movements, represented as wrist tangential velocity profiles, are the result of a linear superimposition of bell-shaped units, is also supported by the learning model presented by Burdet and Milner, 1998. The purpose of Burdet and Milner, 1998 was to replicate the learning process in humans to acquire accuracy in arm motion. The algorithm is based on the quantization of the movement firstly described by Milner, 1992 and on the minimization of time and uncertainty. Comparisons with movements in adults showed that the model can successfully replicate the kinematics of accurate movements.

3.3 Human-likeness evaluation of arm movements

Although it is not always possible to objectively measure the amount of human-likeness of upper-limb movements, quantitative and qualitative metrics that have been proposed in psychology and in neuroscience to evaluate patterns in human arm movements are here shortly described.

In brief, human-like arm movements are featured by a bell-shaped hand velocity profile when there is no risk of collision, a bi-phasic hand velocity profile while avoiding obstacles or when there is a significant change in the wrist orientation, a slightly curved path of the hand and a segmentation of a movement in phases (Biegstraaten et al., 2003; Flash and Hogan, 1985; Lommertzen et al., 2009; Milner, 1992; Morasso, 1981; Rosenbaum et al., 1999; Rosenbaum et al., 2001; Saling et al., 1998; Uno et al., 1989). Similar kinematic metrics have been already considered in robotics as good indexes of human-likeness (see, for example, Täix et al., 2013).

Two important kinematic variables have been used in studies of human biomechanics, which namely are the **Normalized Jerk Score (NJS)** and the **Number of Movement Units (NMU)**. These measures have been used by Chang et al., 2005 to analyse the kinematics of reaching movements in children with cerebral palsy. In Chang et al., 2008, the authors also used the **NJS** and the **NMU** to measure the difference between the reaching performance in healthy adults and in post-stroke patients. Moreover, similar kinematic metrics have been recently used in robotics as well (see, for example, De Momi et al., 2016). Chang et al., 2008 formulated the **NJS** as:

$$NJS = \sqrt{\frac{1}{2} \cdot \left(\frac{T^5}{L^2}\right) \cdot \int \left(\left(\frac{d^3x_H}{dt^3}\right)^2 + \left(\frac{d^3y_H}{dt^3}\right)^2 + \left(\frac{d^3z_H}{dt^3}\right)^2 \right) dt} \quad (3.3.1)$$

where (x_H, y_H, z_H) are the Cartesian coordinates of the hand, T is the duration of the movement and L is the distance of the hand.

The **NMU** is computed according to the following procedure (Chang et al., 2005; Kamper et al., 2002):

1. the velocity profile of the hand is searched for local minima and maxima;
2. an increase in velocity between the adjacent minimum and maximum that exceeds a threshold value signifies an occurrence of a movement unit;
3. a threshold value of 10% of the maximal velocity has been chosen.

Studies have shown that healthy subjects control very well reaching arm movements, which result smoother than those ones in non-healthy subjects. This represents a fewer number of movements units (typically one in normal reaching movements) and a smaller normalized jerk score, which is generally less than 100

during reaching movements (Chang et al., 2005; Chang et al., 2008). To what extent my knowledge, there are not studies that quantify the **NJS** of humans in pick-and-place tasks. However, it is intuitive that there might be a significant increase that depends on the complexity of the task and of the scenario.

Studies on human arm movements have also demonstrated that there is a precise relationship between the velocity and the geometry of the hand path. Pollick et al., 2009 have generalized the *two-thirds power law* (2/3-PL), which supports the fact that the angular velocity of planar hand writing movements is inversely proportional to the curvature to the power of two-thirds (Lacquaniti et al., 1983). This generalization resulted in what the authors called *one-sixth power law* (1/6-PL), which introduced the torsion in the law (Equation 3.3.2).

$$\ln(v) = \ln(\alpha) - \frac{1}{6} \cdot \ln(k^2|\tau|) \quad (3.3.2)$$

$$k = \frac{\left\| \frac{dr}{dt} \times \frac{d^2r}{dt^2} \right\|}{\left\| \frac{d^3r}{dt^3} \right\|^3} \quad (3.3.3)$$

$$\tau = \frac{\frac{dr}{dt} \cdot \left(\frac{d^2r}{dt^2} \times \frac{d^3r}{dt^3} \right)}{\left\| \frac{dr}{dt} \times \frac{d^2r}{dt^2} \right\|^2} \quad (3.3.4)$$

According to the 1/6-PL, the velocity of the hand is inversely related to the one-sixth power of the square curvature, k , multiplied by the torsion, τ (α is a constant). Equation 3.3.3 and Equation 3.3.4 describe the curvature and the torsion, respectively, where r denotes the three-dimensional position of the hand along its path. As Pollick et al., 2009 conclude, the 1/6-PL overcomes the systematic errors that the 2/3-PL commits in the description of planar movements correlated with the global shape of the path (Wann et al., 1988) and of spatial movements in correlation with the global non-planarity of the path (Schaal and Sternad, 2001). For these reasons, the 1/6-PL is adopted instead of the 2/3-PL as a good metric of human-likeness concerning unconstrained planar and spatial scribbling movements.

A qualitative measure of human-likeness can be also extracted by the perception of a human partner, which observes the robot moving its arm and hand. The perception of the robotic arm movements as natural, smooth and human-like enhances the possibilities of a successful human-robot interaction and collaboration because the intention of the robot becomes predictable to humans (Bicho et al., 2011b; Gielniak et al., 2013; Ray et al., 2008; Schaal et al., 2007). For these reasons, a user study is useful to collect and classify the robotic arm movements and, therefore, extract conclusion concerning the experience of the participants.

3.4 Functional models of bimanual coordination

In response of established behavioural phenomena, different functional models of bi-manual coordination have been proposed and confirmed by neurophysiological evidences (Cardoso de Oliveira, 2002). For instance, the spatio-temporal similarity of bi-manual movements, which often emerges during the execution of two-handed tasks, supports the hypothesis of having a communal motor program that governs both limbs. This hypothesis was formulated in the framework of Generalized Motor Programs (GMP), which assumes that movements are represented with programs and parameters in the Central Nervous System and are entirely specified before the execution (Schmidt, 1975). The GMP concepts have been suggested to be also valid for the generation of bi-manual movements (Schmidt et al., 1979) with a proper definition of a hierarchy of parameters that either refer to both arms or specifically to one of them (Figure 3.9).

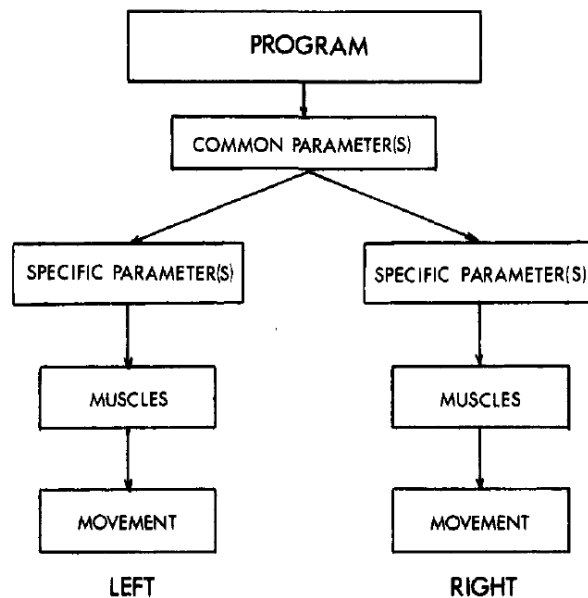


Figure 3.9: Motor program and parametric hierarchy of a bi-manual action (Schmidt et al., 1979).

The GMP applied for two-handed motor programs can explain the remarkable kinematic similarity between the limbs during the execution of bi-manual tasks (Kelso et al., 1979; Schmidt et al., 1979). For example, studies showed that different distances can simultaneously be reached by the two hands, suggesting that positioning and orienting specifications might be encoded as singular-limb parameters. On the contrary, human subjects often find difficulties in moving the hands at different times, which suggests a time-based common programming of the motion (Kelso et al., 1979; Schmidt et al., 1979).

In support of the existence of two independent motor programs, inter-manual crosstalk models have been proposed to describe the interaction of controlling signals for the two arms at two levels. A high-level

crosstalk for planning with a common parametrization and a low-level interaction that occurs during execution (Cardoso de Oliveira, [2002](#)). In favour of a common control strategy the framework of coordinative structures has also been introduced to regulate sets of muscles jointly as a single functional unit. These coordinative structures find implementation in the dynamic systems theory that are often applied, but not limited, to the generation of cyclic movements (Cardoso de Oliveira, [2002](#)).

Chapter 4

The hardware setup

An overview concerning the hardware setup for design, experimentations and tests is described in this chapter. In particular, the anthropomorphic robot ARoS and the collaborative robot Sawyer are presented in section 4.1. A three-fingered artificial hand is described in section 4.2, while section 4.3 formulates the robotic upper-limb representation that is considered in the following chapters to describe the process of human-like motion generation.

4.1 The robots

The solutions of human-like upper-limb motion generation that are described in the following chapters have firstly involved experiments with the humanoid robot ARoS, which is presented in section 4.1.1. Then, these techniques were extended to cope with the needs of collaborative robotic manipulators, such as Sawyer (section 4.1.2).

4.1.1 ARoS

The **A**nthropomorphic **R**obotic **S**ystem) (ARoS) is a humanoid robot designed and built at University of Minho that is composed with a static body torso equipped with two anthropomorphic manipulators, two Barrett Hands as end-effectors (section 4.2) and a stereo video system (Figure 4.1). The dimensions of the body torso were determined taking into account anthropometry studies (Drillis et al., 1966), while its structural stability and resistance were accurately analysed and simulated by Silva et al. (2008). ARoS has two Light Weight Arms by *AMTEC Robotics/Schunk*TM with seven degrees of freedom each ¹. The physical limits of the joints are reported in Table 4.1, while Table 4.2 lists the dimensions of a *Light Weight Arm* in

¹For detailed technical specifications see <http://www.schunk-modular-robotics.com/>

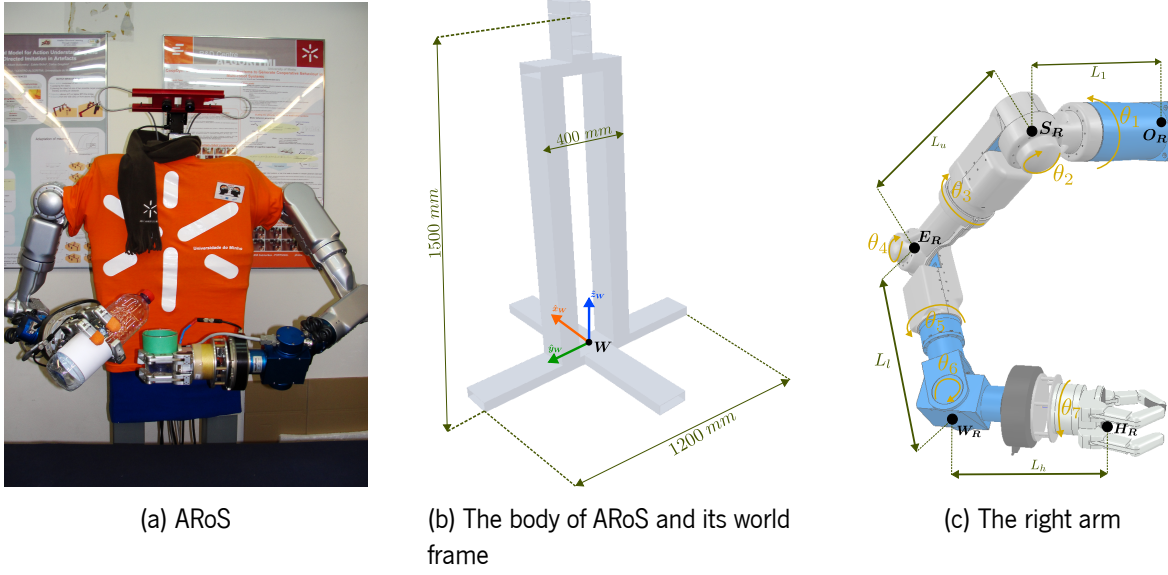


Figure 4.1: The robot ARoS (Figure 4.1a), its body (Figure 4.1b) and its 7-DOFs anthropomorphic right arm (Figure 4.1c), which is equal to its left arm. The joint angles, the physical parameters and most significant human-like points along an arm are displayed.

relation with corresponding key points of a human arm. In particular reference to Figure 4.1c, the point O_R is the origin for the coordinates on the right arm, S_R , E_R , W_R and H_R represent the shoulder, the elbow, the wrist and the hand of the right arm, respectively. Similarly, the points O_L , S_L , E_L , W_L and H_L assume the same meaning for the left arm. The detailed study of the direct kinematics of the arms is given in Appendix A.1. Figure 4.2 illustrates the hardware of the vision system and its functional schema of

Joint	Physical limits [deg]
θ_1	$[-165, 165]$
θ_2	$[-105, 91]$
θ_3	$[-165, 165]$
θ_4	$[-115, 106]$
θ_5	$[-165, 165]$
θ_6	$[-120, 120]$
θ_7	$[-165, 165]$

Table 4.1: Arm physical limits of the robot ARoS

Arm segment	Description	Dimension [mm]
L_1	distance from $O_{\{R,L\}}$ to $S_{\{R,L\}}$	340
L_u	distance from $S_{\{R,L\}}$ to $E_{\{R,L\}}$	395
L_l	distance from $E_{\{R,L\}}$ to $W_{\{R,L\}}$	370
L_h	distance from $W_{\{R,L\}}$ to $H_{\{R,L\}}$	95

Table 4.2: Arm dimensions of the robot ARoS

the robot ARoS. This system is necessary to obtain visual information data from the workspace². For the purposes of this project, the type and the pose (position and orientation) of the objects that are seen by the robot are needed inputs for a collisions-free motion generation. ARoS is also endowed with a cognitive architecture that implements task-related decision-making and a high-level goal setting (Bicho et al., 2011a;

²The complete URDF model of the robot ARoS is maintained at the address https://github.com/zohannn/aros_description

Bicho et al., 2009; Bicho et al., 2011b). In relation with human-like motion planning, these reasoning capabilities can establish appropriate grip types that, in pick-and-place tasks, show intentionality without the need of verbal communication. This behaviour is, indeed, commonly observed in action understanding during human cooperation (Sebanz et al., 2006).

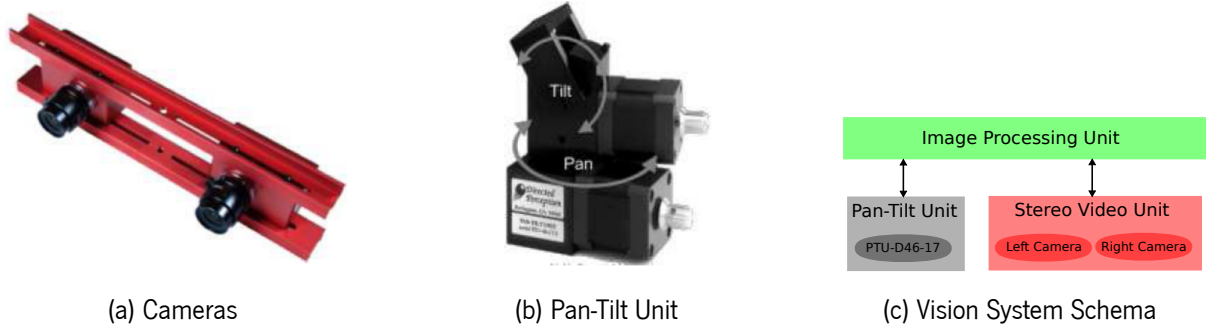


Figure 4.2: The video system of the robot ARoS. A stereo camera system (Figure 4.2a), model STH-DCSG-VAR-C by Videre Design™, which is mounted on a Pan-Tilt unit (Figure 4.2b), model PTU-D46-17 by FLIR Motion Control Systems™. The image processing unit uses this hardware to retrieve type, position and orientation of the objects in the scene (Figure 4.2c).

4.1.2 Sawyer

The collaborative robot Sawyer by *Rethink Robotics*™ (Figure 4.3) is one of the most recent and advanced robotic platforms on the market. It was designed to be flexible and safe for the execution with different tasks in workspaces that are shared with human partners³. Figure 4.3a illustrates the joint angle variables and dimensions along the Sawyer manipulator equipped with a Barrett Hand (section 4.2). The physical limits of the joints are reported in Table 4.3, while Table 4.4 lists the dimensions of the arm in relation with corresponding key points in Figure 4.3c. For adaptation purposes with the human arm, the points in

Joint	Physical limits [deg]
θ_1	$[-175, 175]$
θ_2	$[-219, 131]$
θ_3	$[-175, 175]$
θ_4	$[-175, 175]$
θ_5	$[-171, 171]$
θ_6	$[-171, 171]$
θ_7	$[-270, 270]$

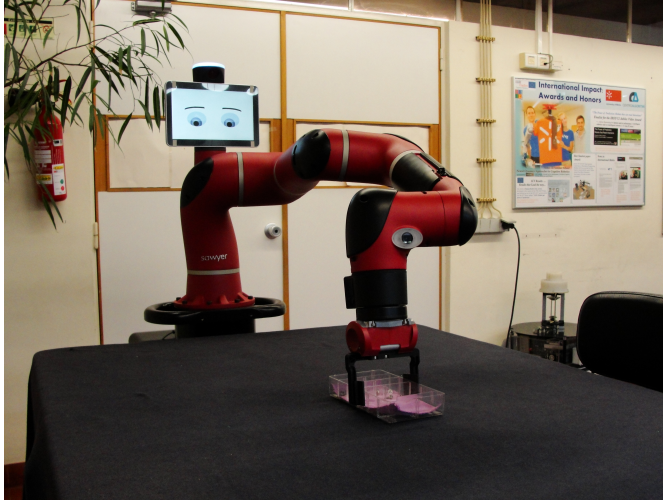
Table 4.3: Arm physical limits of the robot Sawyer

Arm segment	Description	Dimension [mm]
L_1	distance from the base to S_{off}	317
A_1	distance from S_{off} to θ_2 revolution axis	81
L_2	distance from θ_2 revolution axis to S	192.5
L_3	distance from S to E_{off}	400
L_4	distance from E_{off} to E	168.5
L_5	distance from E to W_{off}	400
L_6	distance from W_{off} to W	136.3
L_7	distance from W to H	265.2

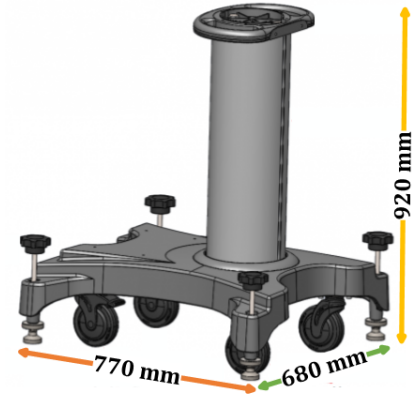
Table 4.4: Arm dimensions of the robot Sawyer

Figure 4.3c are named the shoulder offset (S_{off}), shoulder (S), the elbow offset (E_{off}), the elbow (E),

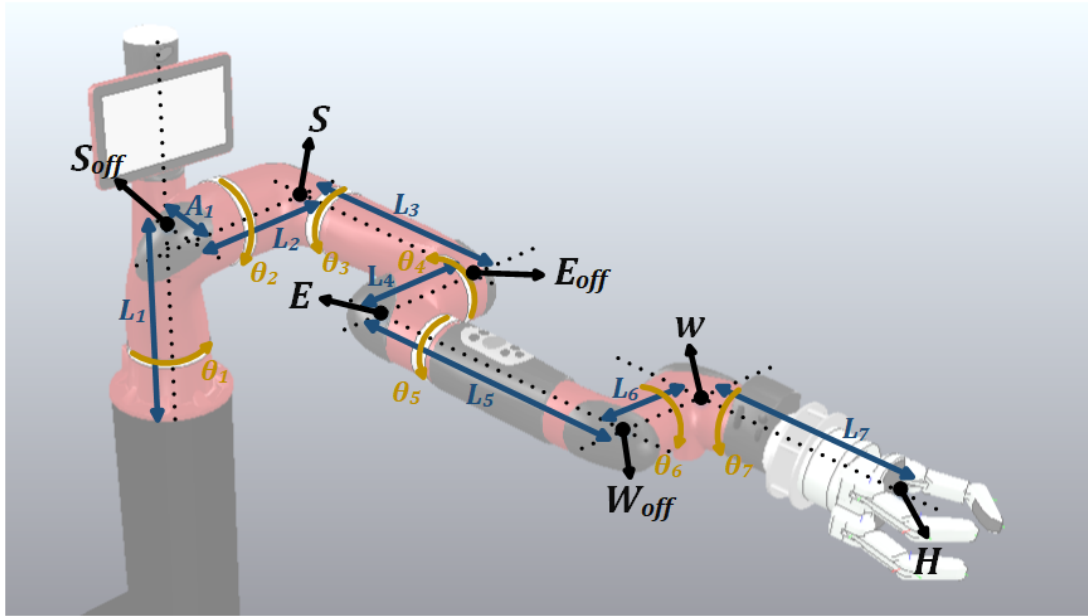
³For detailed technical specifications see <http://mfg.rethinkrobotics.com/>



(a) Sawyer



(b) The pedestal of the robot Sawyer



(c) The parameters of Sawyer manipulator

Figure 4.3: The robot Sawyer (Figure 4.3a), its pedestal (Figure 4.3b) and its 7-DOFs anthropomorphic manipulator (Figure 4.3c). The joint angles, the physical parameters and most significant human-like points along an arm are displayed.

the wrist offset (W_{off}), the wrist (W) and the hand (H). The detailed study of the arm direct kinematics is given in Appendix A.2.

4.2 The Barrett Hand

The manipulators of the robots ARoS and Sawyer are equipped with robotic hands BH8-262 by *Barrett Hand™* that serve as end-effectors. In Figure 4.4, the structure and the most significant parameters of

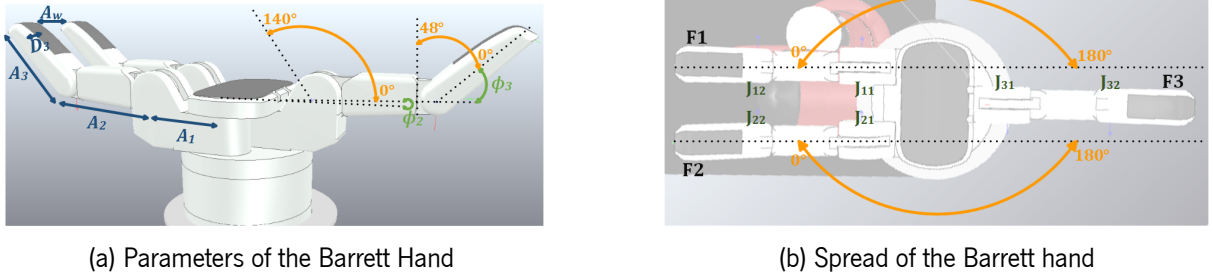


Figure 4.4: Parameters, nomenclature and angular displacement ranges of the Barrett Hand

the Barrett Hand are illustrated ⁴. The hand has three fingers (F1, F2 and F3), which are composed by two phalanxes that are controlled by the groups of joints J1 and J2. For the purposes of this project, only the motion of the first phalanx is considered variable (J1), while the displacement of the second phalanx (J2) is fixed to be 1/3 of the first one. Therefore, the joint angles of the fingers plus the spread between the fingers F1 and F2 (Figure 4.4b) compose a vector of four controlled variables in total. The physical limits of these joints are reported in Table 4.5, while Table 4.6 lists the values of the parameters of the Barrett Hand. In pick-and-place tasks the opening and closing transitions of the robotic hand are crucial

Joint	Physical limits [deg]
θ_8	[0, 180]
θ_9	[0, 140]
θ_{10}	[0, 140]
θ_{11}	[0, 140]

Table 4.5: Physical limits of the Barrett Hand

Parameter	Description	Dimension
A_w	minimal distance between the fingers F1 and F2	25 mm
A_1	distance between the base of the hand and the J1	50 mm
A_2	distance between the J1 and the J2	70 mm
A_3	distance between the J2 and the finger tips	50 mm
D_3	depth of the finger tips	9.5 mm
ϕ_2	angle between the base of the hand and the J1	2.46°
ϕ_3	angle between the J1 and J2	50°

Table 4.6: Dimensions of the Barrett Hand

for a successful accomplishment and interaction with partners. Starting from the sizes of the objects to manipulate and a type of grip, it is possible to obtain the correct values of the fingers' joints. The study of the direct and inverse kinematics of the Barrett Hand is accurately described in Appendix A.3.

⁴For detailed technical specifications see <https://advanced.barrett.com/barretthand>

4.3 Upper-limb formulation

The arm and the hand configurations compose robotic upper-limb representations of equivalent human upper-limb postures. To describe the posture of a robotic upper-limb, the following notation is used in this project:

$$\boldsymbol{\theta} = (\boldsymbol{\theta}_{arm}, \boldsymbol{\theta}_{hand}) \quad (4.3.1)$$

where, $\boldsymbol{\theta}_{arm} \in \mathbb{R}^n$ represents the arm posture with n joints:

$$\boldsymbol{\theta}_{arm} = (\theta_1, \dots, \theta_n)^\top \quad (4.3.2)$$

and $\boldsymbol{\theta}_{hand} \in \mathbb{R}^m$ represents the hand posture with m joints:

$$\boldsymbol{\theta}_{hand} = (\theta_{n+1}, \dots, \theta_{n+m})^\top \quad (4.3.3)$$

A single manipulator of ARoS (either the right or left one) and the robot Sawyer are featured with $n = 7$ and $m = 4$. In this cases, we obtain the following equations:

$$\boldsymbol{\theta}_{arm} = (\theta_1, \theta_2, \theta_3, \theta_4, \theta_5, \theta_6, \theta_7)^\top \quad (4.3.4)$$

$$\boldsymbol{\theta}_{hand} = (\theta_8, \theta_9, \theta_{10}, \theta_{11})^\top \quad (4.3.5)$$

Part II

Planning

Chapter 5

Human-like upper-limb motion planning

This chapter provides the conceptual bases and detailed descriptions of the planning of human-like upper-limb movements in humanoid robots. A more compact version of the planning framework is also described in Gulletta et al., [2015](#); Gulletta et al., [2021](#). Without loss of generality, the introduced planning algorithm refers to configurations of a robotic manipulator with an end-effector (upper-limb), which can be formulated by the vector in Equation [4.3.1](#). In section [5.1](#) the principal concepts of the proposed human-like upper-limb motion planner are outlined. section [5.2](#) introduces the generation of a movement in sequential phases, while, in section [5.3](#), the time parametrization of the planned trajectories is described. The main planning algorithms are then reported in section [5.4](#), while sections [5.5](#), [5.6](#), [5.7](#) present important upgrades concerning the generation of rectilinear motion, the planning of human-like synchronous bi-manual movements and the representation of generic robotic manipulators, respectively.

5.1 The Human-like Upper-limb Motion Planner: concepts

The Human-like Upper-limb Motion Planner (HUMP) results from the continuation of the work presented by Costa e Silva ([2011](#)) and its main theoretical concepts are described in this section. The HUMP takes inspiration from the posture-based motion planning model by Rosenbaum and colleagues (section [3.1](#)), which has been applied to explain how humans generate upper-limb movements in a variety of reaching and manipulation tasks. It solves a motor redundant problem by first selecting a target posture and subsequently generating a collisions-free trajectory. The planning process takes into account a task-dependent hierarchy of spatial and postural constraints modelled as cost functions. In a typical human-robot interacting task, the selection of a grasp posture should, for instance, depend on whether the robot intends to hand over an object to the human partner or to place it elsewhere (Bicho et al., [2011a](#); Bicho et al., [2011b](#)). Another important inspiration from the posture-based motion planning model is the integrated

obstacles avoidance mechanism, which is based on the concept of bounce postures. Human-like obstacles avoidance is achieved by the superimposition of two movements: a *direct movement* from the start to a target posture and a *back-and-forth movement* from the start posture to a bounce posture and back (Figure 5.1).

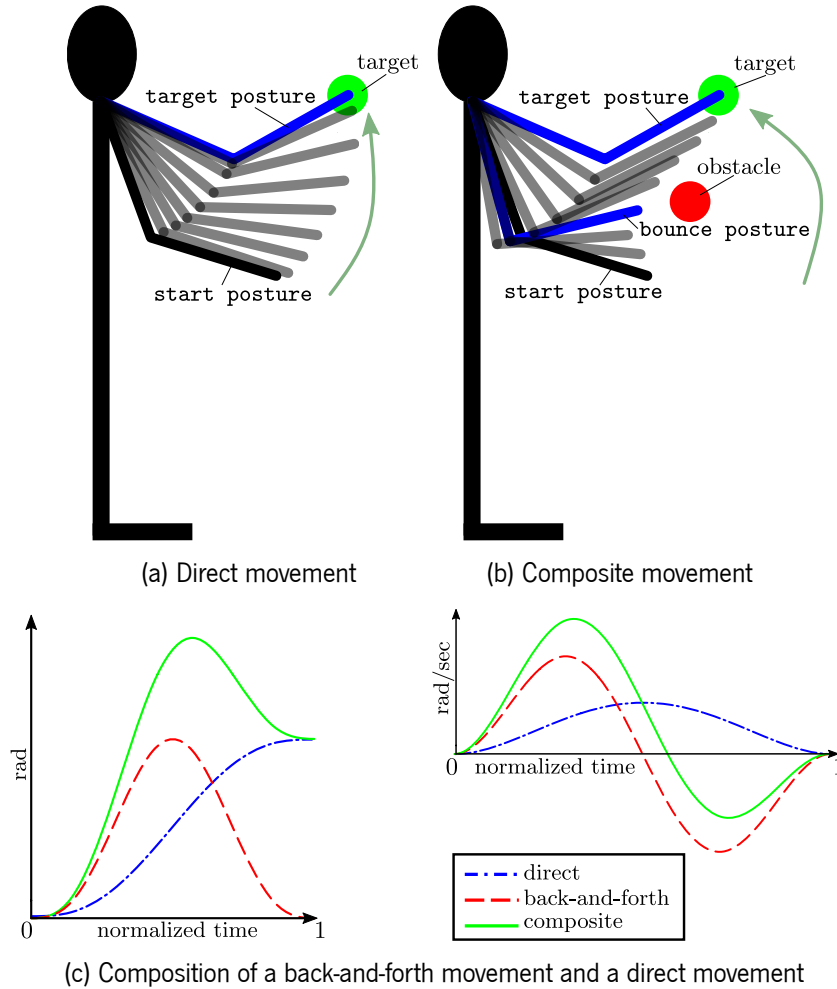


Figure 5.1: A direct movement from the start to the target posture (5.1a) and a composite movement, which is the superimposition of a direct and a back-and-forth movement (5.1b). A back-and-forth movement is a movement from the start to a bounce posture and back (5.1b), which is selected to obtain a collision-free composite movement. Kinematics of a joint during a movement (5.1c).

While the main concepts of the HUMP are inspired from the posture-based motion planning model, there are aspects that have been adapted for the specific needs of a robotic implementation. With the goal to support fluent and efficient human-robot interactions, there is no any explicit inverse kinematics that may result computationally costly and lack of generalization. A second modification concerns the integration of sequential movement phases (section 3.2) which may require high endpoint precision (Flanagan et al., 2006). The implementation of this concept was inspired from the kinematical analysis of human

endpoint precision motion, which reveals a superimposition of single sub-movements to describe the trade-off between accuracy and speed of three-dimensional hand movements (Burdet and Milner, 1998; Milner, 1992). Moreover, a human-like time parametrization is then adopted on the resulting path, which assumes a duration in respect with the obtained absolute angular displacements (section 3.1.3).

The generation of human-like upper-limb movements is resulting from the composition of different phases (section 5.2), while the inner-phase planning algorithm is featured with routines that find motivations in the psychological studies of the posture-based motion planning model (Rosenbaum et al., 2001). At the beginning there is a selection of a task-constrained collision-free target posture. This routine takes into account the start posture of a manipulator (the initial arm-hand configuration), the information concerning the type and the pose (position and orientation) of the objects in the workspace the robot, the pose of a specified target and an eventual grasping behaviour (in case of reaching to grasp). Subsequently, a joints-space trajectory (time history of position, velocity and acceleration) from the start to the target posture is computed and refers to the *direct movement* (Figure 5.1a). Then, a feasible collisions-free movement is searched on the basis of a selected bounce posture. This bounce posture serves as a sub-goal for a *back-and-forth movement*, which is superimposed on the previously obtained *direct movement* (Figure 5.1b). The target posture that is finally reached is the same as for the *direct movement*, only the path differs to guarantee collision avoidance (Figure 5.1). Importantly, in the selection procedure of the target and bounce postures, the HUMP also takes into account anatomical constraints and obstacle avoidance for the entire composite movement. Thus, when these postures are found, the generated upper-limb movements are always anatomically feasible. The details of how the HUMP solves the redundancy problem with precise task requirements are given in sections 5.1.1 and 5.1.2.

5.1.1 Target posture selection

A target posture, θ_{tar} , at $t = T$, where T is the movement duration¹, is a posture that minimizes the displacements of the joints from the initial to the target posture, taking into account obstacle avoidance, joint limits and a desired grip type at the end of the movement (e.g., the grip at the moment of grasping an object). The target posture selection process for a robotic arm and hand can be divided into the selection of a set of finger joints and a set of arm joints. Hence, a target posture is denoted by Equation 5.1.1

$$\theta_{tar} = (\theta_{tar,arm}, \theta_{tar,hand}) \quad (5.1.1)$$

where $\theta_{tar,arm}$ and $\theta_{tar,hand}$ are vectors with the values to be specified for the angular joint positions of the arm and of the fingers of the hand, respectively. The hand target posture, $\theta_{tar,hand}$, depends on

¹The computation of movement duration or movement time is discussed in section 5.3.

the specific hardware of the robotic hand and on the type of movement to be generated. For our case studies with a three-fingered Barrett Hand, the selection of a hand target posture is described by the inverse kinematics solutions in Appendix A.3.2. The selection procedure of the arm target posture, $\theta_{tar,arm}$, is provided in section 5.1.1.1.

5.1.1.1 Arm target posture selection

An arm target posture, $\theta_{tar,arm}$ at $t = T$, is a posture that satisfies the following constraints:

- allows an object to be successfully grasped or manipulated,
- collision avoidance of the arm-hand with the obstacles in the workspace and with the body of the robot,
- respects the mechanical joint limits and
- minimizes the cost associated with moving the arm from the start, $\theta_{0,arm}$ ($t = 0$), to a target posture, $\theta_{tar,arm}$, following a minimum angular jerk trajectory.

The motivation for this latter constraint has origins in the minimum angular jerk criterion as a model to explain simultaneously the slightly curved paths and the bell-shaped hand velocity profiles that are consistently observed in human arm movements (Rosenbaum et al., 1995; Rosenbaum et al., 2001). The cost associated to moving the arm-hand from the start posture, $\theta_{0,arm}$, to a target posture $\theta_{tar,arm}$, with movement duration T , following a minimum angular jerk trajectory is given by Equation 5.1.2.

$$C(\theta_{tar,arm}) = \frac{360}{T^5} \sum_{k=1}^n (\theta_{k,0} - \theta_{tar,k})^2 \quad (5.1.2)$$

where n is the number of DOFs of the robotic arm. Equation 5.1.2 directly derives from Equation C.1.1 that exhibits the angular jerk cost of a generic n -dimensional trajectory.

Different joints may have different expense or cost factors, therefore the relative importance of the various joints for the selection of the target posture may vary, depending, for instance, on personal preferences (Rosenbaum et al., 2001). Thus, by taking inspiration from these findings, we replace $\frac{360}{T^5}$ with a scalar factor for each joint, λ_k . Hence, the new objective cost function is given by Equation 5.1.3.

$$C(\theta_{tar,arm}) = \sum_{k=1}^n \lambda_k (\theta_{0,k} - \theta_{tar,k})^2, \lambda_k \geq 0 \quad (5.1.3)$$

where the non-negative parameters λ_k are the joint expense factors, which allow the joints to contribute differently to the selection process and, to some extent, provide a degree of generalization of the planned movements.

The posture $\theta_{tar,arm}$ is selected as the optimal solution of the following non-linear constrained optimization problem, taking Equation 5.1.3 as cost function:

$$\min_{\theta_{tar,arm}} C(\theta_{tar,arm}) \quad (5.1.4)$$

$$\text{s.t.} \quad \mathbf{h}_{tar}(\theta_{tar}; \mathbf{P}_{tar}) \leq \zeta_{tar} \quad (5.1.5)$$

$$\mathbf{h}_{obs}(\theta_{tar}; \zeta_{obs}) < \mathbf{0} \quad (5.1.6)$$

$$\theta_{m,arm} \leq \theta_{tar,arm} \leq \theta_{M,arm} \quad (5.1.7)$$

Here:

- constraints (5.1.5) impose a determined relationship between the hand pose, which is obtained through direct kinematics of the arm target posture, $\theta_{tar,arm}$, and the hand target pose, $\mathbf{P}_{tar}$, which is a seven-dimensional vector representing the target position $\mathbf{O}_{tar} = (x_{tar}, y_{tar}, z_{tar})$, and orientation (i.e., the parameters of the quaternion) of the local frame $(\hat{\mathbf{x}}_{tar}, \hat{\mathbf{y}}_{tar}, \hat{\mathbf{z}}_{tar})$, and ζ_{tar} is a vector of allowed tolerances in positioning and orienting the end-effector at the given target;
- constraints (5.1.6) guarantee obstacle avoidance at the target posture θ_{tar} , where $\zeta_{obs}(> 0)$ is a constant clearance distance from the obstacles; and
- constraints (5.1.7) ensure that the selected arm target posture, $\theta_{tar,arm}$, satisfies the joints limits, where $\theta_{m,arm}, \theta_{M,arm} \in \mathbb{R}^n$ represent the lower and the upper mechanical joint limits of the arm, respectively.

The vector functions $\mathbf{h}_{tar}$ and $\mathbf{h}_{obs}$, in (5.1.5) and (5.1.6) respectively, are non-linear functions of the target posture θ_{tar} . They are defined using the direct kinematics which map the space of the joints to the Cartesian space. This mapping is well defined for any robotic arm and uses only the Denavit-Hartenberg (D-H) parameters of the robot through direct kinematics (see Appendix A.1 concerning the direct kinematics of the robot ARoS).

The vector function $\mathbf{h}_{tar}$ in Equation 5.1.5 constraints the hand to assume a determined position and orientation when the target is reached, and $\mathbf{h}_{tar} = (h_{pos}, h_{or})^T$, where h_{pos} concerns the constraints in position, while h_{or} is related to those ones in orientation. The function h_{pos} is formulated so that the distance between the target and the palm of the hand is exactly d_{FH} (Equation 5.1.8), which depends on manipulation with a specific end-effector (more details are given in Appendix A.3.2).

$$h_{pos}(\theta_{tar}) = \|\mathbf{H}(\theta_{tar,arm}) + d_{FH}(\theta_{tar,hand}) \cdot \hat{\mathbf{z}}_7(\theta_{tar,arm}) - \mathbf{O}_{tar}\|^2 \quad (5.1.8)$$

,while h_{or} constrains the orientation of the hand (Equation 5.1.9):

$$h_{or}(\theta_{tar,arm}) = \|\hat{\mathbf{x}}_7(\theta_{tar,arm}) - \hat{\mathbf{x}}_{tar}\|^2 + \|\hat{\mathbf{z}}_7(\theta_{tar,arm}) - \hat{\mathbf{z}}_{tar}\|^2 \quad (5.1.9)$$

We only define constraints for the $\hat{\mathbf{x}}_7$ and $\hat{\mathbf{z}}_7$ axes because the $\hat{\mathbf{y}}_7$ is then automatically determined by this definition. For the left arm of the robot ARoS, the axes $\hat{\mathbf{x}}_7$ and $\hat{\mathbf{z}}_7$ in Equations 5.1.8 and 5.1.9 are replaced with $\hat{\mathbf{x}}_{18}$ and $\hat{\mathbf{z}}_{18}$, respectively.

The vector function $\mathbf{h}_{obs}$ in Equation 5.1.6 ensures that there is no collision of the arm and of the hand with the body of the robot and with any obstacle in the workspace. Therefore, $\mathbf{h}_{obs}$ can be written as $\mathbf{h}_{obs} = (\mathbf{h}_{obs_1}, \mathbf{h}_{obs_2})^\top$, where $\mathbf{h}_{obs_1}$ is related to the constraints against the body of the robot:

$$h_{obs_1}^k(\boldsymbol{\theta}_{tar}) = 1 - \left(\frac{P_{k,x}(\boldsymbol{\theta}_{tar}) - x_c}{\sigma_x + r_k} \right)^2 - \left(\frac{P_{k,y}(\boldsymbol{\theta}_{tar}) - y_c}{\sigma_y + r_k} \right)^2 \quad (5.1.10)$$

while $\mathbf{h}_{obs_2}$ concerns those ones against the obstacles in the scenario:

$$h_{obs_2}^{k,l}(\boldsymbol{\theta}_{tar}) = 1 - (\mathbf{P}_k(\boldsymbol{\theta}_{tar}) - \mathbf{C}_l)^\top \mathbf{R}_l^\top \mathbf{A}_{l,k} \mathbf{R}_l (\mathbf{P}_k(\boldsymbol{\theta}_{tar}) - \mathbf{C}_l) \quad (5.1.11)$$

Equation 5.1.10 describes the k^{th} component of the vector function $\mathbf{h}_{obs_1}$, where k indexes the spheres that model the arm, the hand and eventually the object that is being manipulated (see Appendix B for details). Equation 5.1.11 describes the $k \times l$ component of the vector function $\mathbf{h}_{obs_2}$, with $\forall l \in \{1, 2, \dots, n_{obs}\}$, where n_{obs} is the number of the obstacles in a scenario. In Equation 5.1.10 x_c and y_c are the Cartesian coordinates of the center of the cylinder that models the body of the robot ARoS, while σ_x and σ_y are its sizes (Appendix B). In Equations 5.1.10 and 5.1.11, $\mathbf{P}_k(\boldsymbol{\theta}_{tar}) = (P_{k,x}(\boldsymbol{\theta}_{tar}), P_{k,y}(\boldsymbol{\theta}_{tar}), P_{k,z}(\boldsymbol{\theta}_{tar}))^\top$ is the center of the k^{th} sphere with radius r_k along the robotic upper-limb. In Equation 5.1.11, $\mathbf{C}_l = (C_{l,x}, C_{l,y}, C_{l,z})^\top$ is the center of the l^{th} obstacle in the scenario, $\mathbf{R}_l$ is its rotation matrix and $\mathbf{A}_{l,k} = (\text{diag}(r_{l,x}, r_{l,y}, r_{l,z}) + (r_k + \zeta_{obs})\mathbf{I})^{-2}$, where $r_{l,x}, r_{l,y}, r_{l,z}$ are the sizes of the l^{th} obstacle along the three main axes of its wrapping ellipsoid (Appendix B). $\mathbf{I}$ is a three-dimensional identity matrix.

With reference to the Appendix B, Equation 5.1.10 guarantees that the spheres in the Figures B.1a, B.1b and B.2b stay outside of the cylinder illustrated in Figure B.1c. Similarly, Equation 5.1.11 ensures that the points of these spheres are kept at a safe distance (tuned by the parameter ζ_{obs}) from the ellipsoids illustrated in Figure B.2a.

In conclusion, the solution of the optimization problem 5.1.4-5.1.7 is a human-like collision-free target arm posture $\boldsymbol{\theta}_{tar,arm}$, which, composed with the target hand posture $\boldsymbol{\theta}_{tar,hand}$, provides a target posture $\boldsymbol{\theta}_{tar}$ that successfully poses the end-effector onto the given target.

5.1.2 Trajectory selection

Given a human-like target posture, a trajectory from the start posture to the target posture is obtained by the composition of a direct movement (section 5.1.2.1) and of a back-and-forth movement (section 5.1.2.2).

5.1.2.1 Direct movement

The trajectory that brings the arm and the hand from their start posture θ_0 to their target posture θ_{tar} is defined by

$$\mathcal{T}(t; \theta_0, \theta_{tar}) = \theta_0 + \mathcal{T}_{direct}(t; \theta_0, \theta_{tar}) \quad (5.1.12)$$

where $\mathcal{T}_{direct}$ is the *direct movement*. Its specific formulation depends on the type of movement phase - *transport*, *approach*, *retreat* - to be generated (c.f. section 5.2), but it is always defined so that: (i) it generates a hand velocity profile that is consistent with observations in human arm movements (Milner, 1992; Rosenbaum et al., 2001); (ii) guarantees boundary conditions on position, velocity and acceleration; and (iii) the joint angles are within their physical limits during the entire movement duration.

5.1.2.2 Collisions avoidance: the back-and-forth movement

Using Equation 5.1.12 and the direct kinematics model of the arm-hand system (Appendix A), it is possible to predict collisions with the obstacles in the environment, including the body of the robot and the object to be grasped (in case of manipulation). If a collision is anticipatively detected, then a feasible alternative movement, that still brings the arm and the hand from the start posture, θ_0 , to the target posture, θ_{tar} , needs to be computed. For this purpose, we use the notion of bounce posture, which was firstly introduced concerning planar movements (Meulenbroek et al., 2001; Rosenbaum et al., 1999) and later in relation with three-dimensional movements (Vaughan et al., 2006). The alternative movement is found by computing a suitable bounce posture, i.e. a posture that serves as subgoal, for a *back-and-forth movement*. This movement consists of first moving the arm and the hand from their start posture, θ_0 , to a bounce posture, θ_b , (*forth*) and next moving back from this posture to the start one (*back*). The *back-and-forth movement* is superimposed on the *direct movement*:

$$\mathcal{T}(t; \theta_0, \theta_{tar}, \theta_b) = \theta_0 + \mathcal{T}_{direct}(t; \theta_0, \theta_{tar}) + \mathcal{T}_{bkfor}(t; \theta_0, \theta_b) \quad (5.1.13)$$

and it is intended to avoid collisions with obstacles during the entire execution of the (composite) movement. The *back-and-forth movement*, $\mathcal{T}_{bkfor}$, is modeled by Equation 5.1.14:

$$\mathcal{T}_{bkfor}(t; \theta_0, \theta_b) = \frac{\tau(1-\tau)}{t_b(1-t_b)} (\theta_b - \theta_0) \sin^2 \left(\pi \tau^g \right) \quad (5.1.14)$$

where $\tau = t/T$ is the normalized time, being T the movement duration (see section 5.3). As a result of the superimposition in Equation 5.1.13, trajectories change but the target posture remains the same (Figure 5.1).

The motivation for using a squared sinus function for the *back-and-forth movement* is primarily based on the fact that discrete movements that start and end in a standstill, can, in principle, be considered half a

period of a cyclical movement (Guiard, 1993). The sinusoidal element remains a key in the moment that the directional reversal occurs, while the zero velocity crossing remains a key at the start and at the end of the movement. The kinematic profile adopted here thus capitalizes on core kinematic features of both discrete and cyclical motion. Furthermore, the squared sine function is also continuously differentiable, which is an important feature when we use deterministic optimization methods that require such smooth functions.

In (5.1.14), $\vartheta = -\frac{\ln 2}{\ln t_b}$, with $t_b \in]0, 1[$ is the parameter used to control when the bounce posture, θ_b , is reached. We set $t_b = 0.5$, so that the “forth” and the “back” phases take 50% of the total movement time, respectively. The underlying assumption is that the impact of the obstacles on the movement trajectory will be evenly distributed across the entire movement. Empirical evidence of this behaviour has been reported by Vaughan et al. (2001). It is worthwhile noting that the model does not presume that the location of the (anticipated) collision is at 50% of the straight-line distance between start and end of the movement spatially, nor does it presume that it is halfway in terms of movement time. In contrast, the location can be anywhere in the workspace, because the bounce posture, θ_b , is selected taking into account that no collision occurs for the entire duration of the composite movement given by Equation 5.1.13 (section 5.1.2.3).

5.1.2.3 Bounce Posture Selection

Similarly to the target posture selection procedure, the aim is to select a suitable bounce posture, $\theta_b = (\theta_{b,1}, \dots, \theta_{b,n+m})^\top$, where n and m are the number of DOFs of the arm and of the hand, respectively. In fact, the bounce posture to be selected, θ_b , is a composition of the arm bounce posture, $\theta_{b,arm}$, and of the hand bounce posture, $\theta_{b,hand}$:

$$\theta_b = (\theta_{b,arm}, \theta_{b,hand}) \quad (5.1.15)$$

In our case studies $n = 7$ and $m = 4$ with $\theta_{b,hand} = (\theta_{b,8}, \theta_{b,9}, \theta_{b,10}, \theta_{b,11})^\top$ for the right hand and $\theta_{b,hand} = (\theta_{b,19}, \theta_{b,20}, \theta_{b,21}, \theta_{b,22})^\top$ for the left hand of the robot ARoS.

The bounce posture minimizes the displacement of the joints of the arm and of the hand from the start posture, θ_0 , to the bounce posture, θ_b , subject to following constraints:

- collision avoidance, over the all duration of the composite movement, with any obstacle in the workspace, with the body of the robot and, eventually, with the object that has to be manipulated during the entire trajectory;
- the mechanical joint limits are respected over the all duration of the composite movement;
- minimization of the cost function

$$C(\theta_b) = \sum_{k=1}^{n+m} \lambda_k (\theta_{0,k} - \theta_{b,k})^2, \lambda_k \geq 0 \quad (5.1.16)$$

that is associated with moving the robotic upper-limb from the start posture, θ_0 , to the bounce posture, θ_b .

To formulate this selection problem, the composite movement, $\mathcal{T}(t)$ in Equation 5.1.13, is taken into consideration and the time $t \in [0, T]$ is discretized into equally spaced N_{steps} steps denoted as $t_i = i\Delta t_{step}$, where Δt_{step} is the step size and $i = 0, 1, \dots, N_{steps}$ (see section 5.3 for details). The convention adopted is that $\mathcal{T}^i = \mathcal{T}(t_i)$ represents the configuration of the joints at time step t_i during the trajectory. We formalize these concepts as a non-linear constrained optimization problem as follows:

$$\min_{\theta_b} C(\theta_b) \quad (5.1.17)$$

$$\text{s.t. } \theta_m \leq \mathcal{T}^i \leq \theta_M \quad (5.1.18)$$

$$\mathbf{h}_b(\mathcal{T}^i; \boldsymbol{\varepsilon}_b^i) \leq \mathbf{0} \quad (5.1.19)$$

$$\theta_m \leq \theta_b \leq \theta_M \quad (5.1.20)$$

where $\theta_m, \theta_M \in \mathbb{R}^{n+m}$ are the vectors with the lower and the upper mechanical joint limits, respectively, $\mathbf{h}_b$ is a vector of non-linear functions defined to guarantee the avoidance of collisions, being $\boldsymbol{\varepsilon}_b^i = \boldsymbol{\varepsilon}_b(t_i)$ a vector of positive clearance distances at time t_i . For every instant t_i , constraints (5.1.18) guarantee that the trajectory of the joints do not overcome their physical limits; inequalities (5.1.19) ensure that there is no collision with: (i) the object to grasp (only in picking), (ii) the obstacles in the scenario, and (iii) the body of the robot; constraints (5.1.20) ensure that the selected optimal bounce posture is within the limits of the joints.

When manipulation is not required, the hand bounce posture, $\theta_{b,hand}$, is completely selected by the solution of the optimization problem given by (5.1.17)-(5.1.20). During transportation, an object is held by the hand for the entire duration of a trajectory. Therefore, in this case, $\theta_{b,hand} = \theta_{0,hand}$. This implies a null contribution of the back-and-forth movement concerning the total motion of the fingers in Equation 5.1.13. When grasping is required, due to the simplification described in Appendix A.3.2, the middle finger is always opposite to the other two fingers of a Barrett hand, thus $\theta_{b,8} = 0$ for the right hand and $\theta_{b,19} = 0$ for the left hand of the robot ARoS. Moreover, $\theta_{b,9} = \theta_{b,10} = \theta_{b,11} = \theta_{b,finr}$ for the right hand and $\theta_{b,20} = \theta_{b,21} = \theta_{b,22} = \theta_{b,finl}$ for the left hand of the robot ARoS, since the fingers have the same lengths (see Appendix A.3.2 for details).

The vector of constraints $\mathbf{h}_b$ (Equation (5.1.19)) changes its composition according to the type of movement that is being planned. When grasping is not required, $\mathbf{h}_b$ is written as $(\mathbf{h}_{b_1}, \mathbf{h}_{b_2})^\top$, where $\mathbf{h}_{b_1}$ and $\mathbf{h}_{b_2}$ guarantee the avoidance of collisions with the body of the robot and with any obstacle in the scenario, respectively. The tuning of $\boldsymbol{\varepsilon}_b^i = \boldsymbol{\varepsilon}_{obs}^i = \boldsymbol{\varepsilon}_{obs}(t_i)$ varies the minimum acceptable distance at time t_i with

the obstacles.

$$h_{b_1}^k(t_i) = 1 - \left(\frac{P_{k,x}(t_i) - x_c}{\sigma_x + r_k} \right)^2 - \left(\frac{P_{k,y}(t_i) - y_c}{\sigma_y + r_k} \right)^2 \quad (5.1.21)$$

Equation 5.1.21 describes the k^{th} component of the vector function $\mathbf{h}_{b_1}(t_i)$, where k indexes the spheres that model the arm, the hand and eventually the object that is being manipulated (see Appendix B for details).

$$h_{b_2}^{k,l}(t_i) = 1 - (\mathbf{P}_k^i - \mathbf{C}_l)^\top \mathbf{R}_l^\top \mathbf{A}_{l,k}^i \mathbf{R}_l (\mathbf{P}_k^i - \mathbf{C}_l) \quad (5.1.22)$$

Equation 5.1.22 describes the $k \times l$ component of the vector function $\mathbf{h}_{b_2}(t_i)$, with $l = 1, \dots, n_{obs}$, where n_{obs} is the total number of the obstacles. In Equations 5.1.21 and 5.1.22, $\mathbf{P}_k^i = \mathbf{P}_k(t_i) = (P_{k,x}(t_i), P_{k,y}(t_i), P_{k,z}(t_i))^\top$ is the center of the k^{th} sphere along the robotic upper-limb with radius r_k at time t_i . In Equation 5.1.22, $\mathbf{C}_l = (C_{l,x}, C_{l,y}, C_{l,z})^\top$ is the geometrical center of the l^{th} obstacle in the scenario, $\mathbf{R}_l$ is its rotation matrix and $\mathbf{A}_{l,k}^i = \mathbf{A}_{l,k}(t_i) = (diag(r_{l,x}, r_{l,y}, r_{l,z}) + (r_k + \varepsilon_{obs}^i)I)^{-2}$, with $r_{l,x}$, $r_{l,y}$, $r_{l,z}$ the sizes of the l^{th} obstacle along the three axes of its wrapping ellipsoid (Appendix B). I is a three-dimensional identity matrix.

When grasping is required, more attention is necessary on avoiding collision with the object that has to be picked, because it is in a region where the end-effector is approaching to, but it is also an obstacle until the completion of the grasp transition. For this reason, this issue is addressed with specific constraints in the bounce posture selection problem. Particularly, another vector of constraints, $\mathbf{h}_{b_3}$, is added to the vector $\mathbf{h}_b$, which is then written as $(\mathbf{h}_{b_1}, \mathbf{h}_{b_2}, \mathbf{h}_{b_3})^\top$. The total number of obstacles, n_{obs} , is decremented by one, because the object that is going to be picked is treated separately by the constraints $\mathbf{h}_{b_3}$. Additionally, $\varepsilon_b^i = \varepsilon_b(t_i) = (\varepsilon_{obs}(t_i), \varepsilon_{obj}(t_i))^\top$, where $\varepsilon_{obs}(t_i)$ and $\varepsilon_{obj}(t_i)$ are the minimum acceptable distance at time t_i with the obstacles and with the object to pick, respectively.

$$h_{b_3}^k(t_i) = 1 - (\mathbf{P}_k^i - \mathbf{O}_{obj})^\top \mathbf{R}_{obj}^\top \mathbf{A}_{obj,k}^i \mathbf{R}_{obj} (\mathbf{P}_k^i - \mathbf{O}_{obj}) \quad (5.1.23)$$

Equation 5.1.23 introduces the k^{th} component of the vector $\mathbf{h}_{b_3}(t_i)$, where k indexes the spheres that model the arm and the hand (Appendix B) and $\mathbf{O}_{obj}$ is the geometrical center of the object to be picked. $\mathbf{A}_{obj,k}^i = (diag(r_{obj,x}, r_{obj,y}, r_{obj,z}) + (r_k + \varepsilon_{obj}^i)I)^2$, wherein $r_{obj,x}$, $r_{obj,y}$, $r_{obj,z}$ are the dimensions on the axes of the ellipsoid wrapping the object to be picked (Appendix B). I is a three-dimensional identity matrix and $\mathbf{R}_{obj}$ is the rotational matrix the object to be picked.

To conclude, the solution of the optimization problem 5.1.17-5.1.20, is a human-like collision-free bounce posture, $\boldsymbol{\theta}_b$, to successfully reach the target posture, $\boldsymbol{\theta}_{tar}$, while avoiding collisions during the execution of the composite trajectory $\mathcal{T}(t)$ in Equation 5.1.13.

5.2 The Human-like Upper-limb Motion Planner: trajectory generation in sequential phases

5.2.1 Classification and segmentation of a movement

Typical object manipulation tasks can be segmented into meaningful action phases in which an object is approached, grasped, lifted off a surface, moved, and placed on a surface again. The transition between the different phases is defined by distinct sensory events like making or breaking contact between the manipulator and an object or between an object and a surface (see section 3.2). These events define subgoals of the task (Flanagan et al., 2006). The division in functional units or primitives of goal-directed motor acts supports a flexible task planning since the primitives represent elementary building blocks which can be combined and adapted in many different ways to generate complex goal-directed behaviours (for robotics examples see Stulp et al., 2012). In the posture-based motion planning model (Rosenbaum et al., 2001), a temporal task segmentation implies that a selected target posture of a certain motor act (e.g., picking) becomes the start posture of the subsequent motor primitive (e.g., lifting).

A segmentation in different motor primitives has been also proposed for the kinematics level to explain observed regularities in human motion. During reach-to-grasp movements, the wrist velocity typically shows a single-peaked, bell-shaped profile, with the acceleration phase lasting shorter than the deceleration phase. This asymmetry tends to become more prominent for movements requiring higher precision, leading to a nearly constant velocity when the arm approaches the target position (Figure 3.8).

To account for the impact of precision constraints on the limb velocity profile and integrate it in the posture-based motion planning model, picking and placing are further segmented into an initial *transporting phase* followed by an *approaching phase*. This segmentation well captures the bi-phasic nature of the grasping component of typical human prehension, which appears to be temporally coupled with the reaching component (Castiello, 2005). During hand movements, there is first a progressive opening of the grip up to a maximum grip aperture which is proportional to the size of the object. This is followed by a gradual closure of the grip until it matches the size of the object. The maximum grip aperture occurs at about two-thirds of the way through the duration of the reaching, which roughly coincides with the start of the *approaching phase*.

Figure 5.2 illustrates the temporal segmentation of a pick-and-place task in the different phases of an upper-limb movement with the labels that are used in this project. Grasp (**g**) and ungrasp (**u**) are treated as transitions between the approaching (to pick) and lifting motions, and the approaching (to place) and depart motions, respectively. We use the common label R (for *Retreat*) to describe both the object lifting from the surface and the final depart of the hand from the placing position, since the planning process

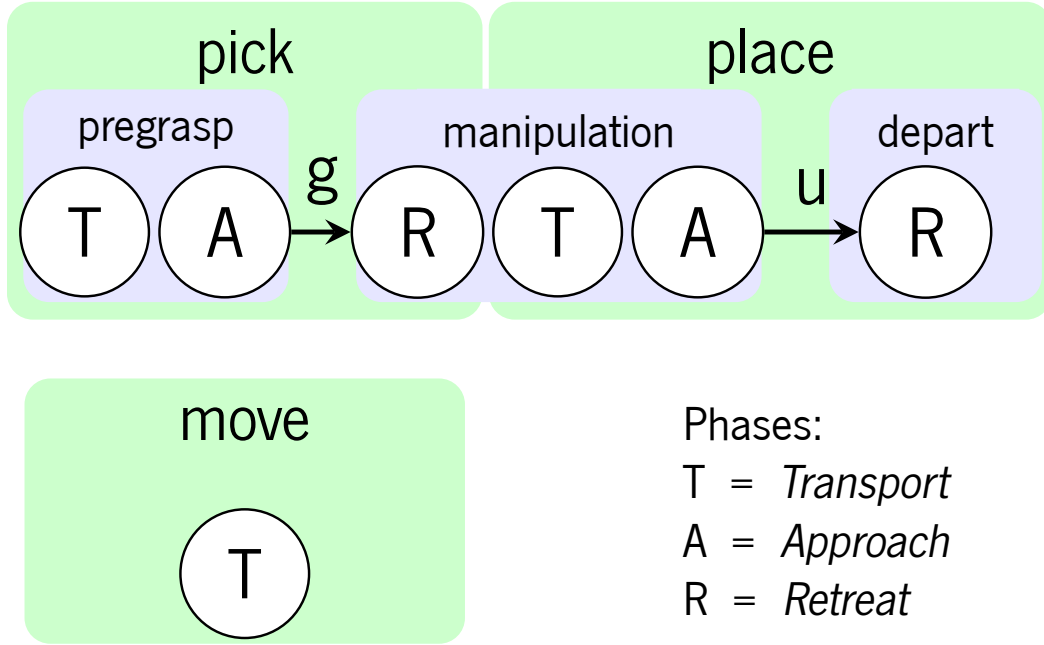


Figure 5.2: Temporal segmentation of the movements.

treats both phases equally. Pick and place movements consist of a combination of three object-directed primitives *Transport* (T), *Approach* (A), and *Retreat* (R).

A pick movement starts to be formed by a *Transport* phase followed by an *Approach* phase: pregrasp. The *Transport* phase is responsible for transporting the hand into the proximity of the object to grasp and for partially pre-shaping the hand. The *Approach* phase brings the hand into the pose of grasping and concludes object grasping. Next, manipulation starts with a *Retreat* phase that lifts the object. Similarly to a pick movement, a place movement is partially formed by a *Transport* followed by an *Approach* phase. During the *Transport* phase, the grasped object is transported into the proximity of its final location. Then, the *Approach* phase concludes the relocation of the object in its goal pose. After the ungrasp transition, depart starts with a *Retreat* phase that takes the hand away from the object. In addition to the object-directed movements pick and place, the HUMP also generates move movements that do not manipulate any object in the workspace and, therefore, are only composed by a *Transport* phase, which transports the hand towards its target pose and brings the fingers towards their desired posture.

The *Transport*, *Approach* and *Retreat* phases are also distinguished by a selection of specific target postures as it is described in section 5.1.1. Next, we present how these phases are generated, taking into account the type of movement (pick, place or move) to which they belong.

5.2.2 Transport phase

The transport target posture is a posture $\theta_{tran} = (\theta_{tran,arm}, \theta_{tran,hand})$, where $\theta_{tran,arm}$ is the arm target posture and $\theta_{tran,hand}$ is the hand target posture. $\theta_{tran,arm}$ is selected as explained in section 5.1.1.1 for a given transport target hand pose. Next, the selection of $\theta_{tran,hand}$, and the composition of the trajectory in the *Transport* phase are described.

5.2.2.1 Hand Transport target posture selection

In pick movements, the hand transport target posture for grasping is calculated as described in Appendix A.3.2. However, if an *Approach* phase follows, the values of $\theta_{tran,hand}$ are modified to obtain a slight greater aperture of the fingers than it is necessary. This pre-shaping of the fingers is done to mimic prehension and guarantee a successful grasping transition.

In place movements, the hand transport target posture is constrained by the fact that the object that is being manipulated has to remain held by the end-effector. Therefore, the hand transport target posture is set to be equal to the hand start posture, i.e. $\theta_{tran,hand} = \theta_{0,hand}$.

Move movements are only featured by a hand *Transport* phase, which is not directed towards objects. Hence, in this particular situation, the joint angle values in $\theta_{tran,hand}$ can be set arbitrarily.

5.2.2.2 Transport direct movement

For the *Transport* phase, the *direct movement* consists of a trajectory that is generated based on the minimum angular jerk principle. With this approach, the end-effector trajectories (in extrinsic spatial coordinates) with a bell-shaped velocity profile does not need to be modelled explicitly, but rather emerges from the joint displacements (Rosenbaum et al., 1995). The *direct movement* is computed by minimizing the angular jerk over the duration of a movement (Equation 5.2.1)

$$C_{dm}(\theta) = \frac{1}{2} \int_0^T \sum_{k=1}^n \left(\frac{d^3 \theta_k(t)}{dt^3} \right)^2 dt \quad (5.2.1)$$

where n is the number of joints, $\theta_k(t)$ is the k^{th} joint angle position at instant t and T is the movement duration. This is a typical variational problem and the solution is a 5th order polynomial whose coefficients may be determined applying boundary conditions on position, velocity and acceleration (see Appendix C.1 for details). The solution is the *transport direct movement* (Equation 5.2.2)

$$\begin{aligned} \mathcal{T}_{tran}^{direct}(t; \theta_{tran}) = & (\theta_{tran} - \theta_0) (10\tau^3 - 15\tau^4 + 6\tau^5) \\ & + \omega_0 T (\tau - 6\tau^3 + 8\tau^4 - 3\tau^5) + \omega_{tar} T (-4\tau^3 + 7\tau^4 - 3\tau^5) \\ & + \frac{1}{2} \alpha_0 T^2 (\tau^2 - 3\tau^3 + 3\tau^4 - \tau^5) + \frac{1}{2} \alpha_{tar} T^2 (\tau^3 - 2\tau^4 + \tau^5) \end{aligned} \quad (5.2.2)$$

where $\theta_0 = (\theta_{0,1}, \dots, \theta_{0,n})^\top$ is the vector with the start joint angles ($t = 0$), $\theta_{tran} = (\theta_{tran,1}, \dots, \theta_{tran,n})^\top$ is the vector with the target joint angles ($t = T$) and $\tau = t/T$ is the normalized time. ω_0 and ω_{tar} are the start and the target joint velocities, respectively, while α_0 and α_{tar} are the start the target joint accelerations, respectively. Equation 5.2.2 guarantees that the joint angles are within their mechanical limits during the entire movement duration because (i) the values of the transport target posture θ_{tran} are into their admissible range and (ii) Equation 5.2.2 is a monotonic increasing or decreasing function that generates values from $\theta_{0,k}$ to $\theta_{tran,k}$ for each k^{th} joint. Therefore, since both the start and the target posture obey the constraints limits of the joints, the same happens in any step of the trajectory. Angular velocities and accelerations of the *Transport phase* can be obtained by deriving Equation 5.2.2 over time and are reported in Appendix C.1. The movement of the joints follows a straight-line path in the joint space with a bell-shaped uni-modal velocity profile. In the task space this may result in a straight path or in a slightly curved path, both with bell-shaped velocity profile, which is consistent with the observations on human upper-limb movements (Rosenbaum et al., 2001). By this method, there is no need to model explicitly the bell-shaped velocity profile of the end-effector, which rather emerges from the displacements of the joints (c.f. chapter 6).

5.2.2.3 Transport back-and-forth movement

The second component of the *Transport phase* consists in the *back-and-forth movement* that is given by Equation 5.1.14, and it is intended to avoid collisions (during the entire execution of the *Transport phase*) with the obstacles in the workspace, the body of the robot, and, in case of pick movements, also with the object to grasp. The bounce posture selection process (section 5.1.2.3) is here described to specify the constraints of the optimization problem (5.1.17)-(5.1.20) according to the type of movement. In move movements (composed only by a *Transport phase*), the bounce hand posture, $\theta_{b,hand}$, is completely selected by the solution of the optimization problem (5.1.17)–(5.1.20). During the *Transport phase*, in place movements, the object that is being manipulated is held by the hand for the entire duration of the transportation. Therefore, in this case, the bounce hand posture is equal to the start hand posture, i.e. $\theta_{b,hand} = \theta_{0,hand}$. This implies a null contribution of the *back-and-forth movement* concerning the total motion of the fingers in Equation 5.1.13. In pick movements, likewise for the hand target posture selection (section 5.1.1), the selection of $\theta_{b,hand}$ depends on the specific hardware of the robotic hand. In the three-fingered Barrett Hand, the middle finger is always opposite to the other two fingers, thus $\theta_{b,8} = 0$, and we can also control the position of the fingers with one variable, i.e. $\theta_{b,fing}$. This variable is then selected by the solution of the problem (5.1.17)–(5.1.20) (see section 5.1.2.3 for details). The solution of the optimization problem (5.1.17)–(5.1.20), is a bounce posture, θ_b , which placed in

Equation 5.1.14, gives the *back-and-forth* component of the *Transport* phase trajectory.

5.2.2.4 Composite movement of the Transport phase

In the *Transport* phase, the trajectory of each joint is obtained by the superimposition of the *back-and-forth* movement onto the *transport direct movement* Equation 5.2.3

$$\mathcal{T}_{tran}(t, \theta_{tran}, \theta_b) = \theta_0 + \mathcal{T}_{tran}^{direct}(t, \theta_{tran}) + \mathcal{T}_{tran}^{bkfor}(t, \theta_b) \quad (5.2.3)$$

where $\mathcal{T}_{tran}^{direct}$ and $\mathcal{T}_{tran}^{bkfor}$ are given by Equation 5.2.2 and Equation 5.1.14, respectively. Equation 5.2.3 guarantees a feasible and human-like collision-free trajectory with continuous angular velocity and acceleration. Joint limits are not violated and, by setting the movement time T appropriately, the maximum allowed angular velocity and the maximum joint acceleration of each joint are never reached (c.f. section 5.3).

Velocities and accelerations in the *Transport* phase are obtained by deriving Equation 5.2.3 with respect to time are reported in Equations 5.2.4 and 5.2.5.

$$\begin{aligned} \dot{\mathcal{T}}_{tran}(t; \theta_{tran}, \theta_b) = & \frac{30}{T} (\theta_{tran} - \theta_0) (\tau^2 - 2\tau^3 + \tau^4) \\ & + \omega_0 (1 - 18\tau^2 + 32\tau^3 - 15\tau^4) + \omega_{tar} (-12\tau^2 + 28\tau^3 - 15\tau^4) \\ & + \frac{1}{2} \alpha_0 T (2\tau - 9\tau^2 + 12\tau^3 - 5\tau^4) + \frac{1}{2} \alpha_{tar} T (3\tau^2 - 8\tau^3 + 5\tau^4) \\ & + \frac{1}{T t_b (1 - t_b)} (\theta_b - \theta_0) \cdot \left[(1 - 2\tau) \sin^2(\pi \tau^\vartheta) \right. \\ & \left. + (1 - \tau) \pi \vartheta \tau^{\vartheta-1} \sin(2\pi \tau^\vartheta) \right] \end{aligned} \quad (5.2.4)$$

$$\begin{aligned} \ddot{\mathcal{T}}_{tran}(t; \theta_{tran}, \theta_b) = & \frac{60}{T^2} (\theta_{tran} - \theta_0) (\tau - 3\tau^2 + 2\tau^3) \\ & + \frac{12}{T} \omega_0 (-3\tau + 8\tau^2 - 5\tau^3) + \frac{12}{T} \omega_{tar} (-2\tau + 7\tau^2 - 5\tau^3) \\ & + \alpha_0 (1 - 9\tau + 18\tau^2 - 10\tau^3) + \alpha_{tar} (-3\tau - 12\tau^2 + 10\tau^3) \\ & + \frac{2}{T^2 t_b (1 - t_b)} (\theta_b - \theta_0) \left[-\sin^2(\pi \tau^\vartheta) + \pi^2 \vartheta^2 (1 - \tau) \tau^{2\vartheta-1} \cos(2\pi \tau^\vartheta) \right. \\ & \left. + (1 - 2\tau) \pi \vartheta \tau^{\vartheta-1} \sin(2\pi \tau^\vartheta) \right] \end{aligned} \quad (5.2.5)$$

5.2.3 Approach phase

Following the *Transport* phase comes the *Approach* phase. The approach target posture is a posture $\theta_{app} = (\theta_{app,arm}, \theta_{app,hand})$, where $\theta_{app,arm}$ is the arm target posture and $\theta_{app,hand}$ is the hand

target posture. The selection of $\theta_{app,arm}$ is performed as explained in section 5.1.1.1 for a given approach target hand pose.

5.2.3.1 Hand Approach target posture selection

The hand target posture, $\theta_{app,hand}$, is differently computed for pick and for place movements. In pick movements, the *Transport* phase transports the hand onto the close proximity of the object to grasp with the adequate preshaping of the fingers. Next, the *Approach* phase brings the hand into the pose of grasping and concludes it. For a successful grasp, the value of joints in the fingers are computed according to the size of the object to grasp, as explained for the Barrett hand in Appendix A.3.2.

In place movements, the *Transport* phase transports the grasped object into the close vicinity of its final location. Then, the *Approach* phase concludes positioning of the object in its goal pose. Thus, the approach hand target posture remains constant to hold safely the object being placed, i.e. $\theta_{app,hand} = \theta_{0,hand}$.

5.2.3.2 Approach direct movement

The *Transport* phase always brings the hand into close proximity of its goal pose using a feasible and collision-free trajectory for the arm-hand, because it is reasonable to assume that there are no obstacles in the close vicinity of the target. On the contrary, there might not exist a solution of the planning problem, because the paths towards the goal are critically obstructed. Thus, a *direct movement per se* is sufficient to generate a collision free trajectory for the *Approach* phase.

This phase, brings the arm and the hand from the target posture of the *Transport* phase, θ_{tran} , (which is now the start posture for the *Approach* phase, θ_0) to the approach target posture, θ_{app} . The formulation of the *approach direct movement*, $\mathcal{T}_{app}(t; \theta_{app})$, has been inspired from the work in human motor control presented in Burdet and Milner, 1998; Milner, 1992 (Figure 3.8) and it is given by Equations 5.2.6, 5.2.7 and 5.2.8:

$$\mathcal{T}_{app}(t; \theta_{app}) = \theta_0 + \frac{1}{4}(\theta_{app} - \theta_0)(5\tau - \tau^5) \quad (5.2.6)$$

$$\dot{\mathcal{T}}_{app}(t; \theta_{app}) = \omega_0(1 - \tau^4) \quad (5.2.7)$$

$$\ddot{\mathcal{T}}_{app}(t; \theta_{app}) = -\alpha_0\tau^3 \quad (5.2.8)$$

where $\omega_0 = \frac{5}{4T}(\theta_{app} - \theta_0)$, $\alpha_0 = \frac{4\omega_0}{T}$, T is the total duration of the *Approach* phase and $\tau = t/T$ is the normalized time. The trajectory of the joints during the *Approach* phase (Equation 5.2.6) is a monotonic increasing, if $(\theta_{app,k} - \theta_{0,k})$ is positive, or decreasing function, if $(\theta_{app,k} - \theta_{0,k})$ is negative, that generates values from $\theta_{k,0}$ to $\theta_{k,app}$ for each k^{th} joint. Therefore, since both the start and the target postures are

within the admissible range of the joints, the entire *approach direct movement* respects their physical limits. Equation 5.2.7 and Equation 5.2.8 model the velocity and the acceleration of the *Approach* phase, respectively. In Equation 5.2.7, it is possible to note that the arm and the hand stop at the end the *Approach* phase in order to start the grasp/ungrasp transition. At the beginning of the *Approach* phase, the joints have velocity ω_0 and acceleration α_0 , which must be equal to the velocity ω_{tar} and acceleration α_{tar} of the *Transport* phase movement in Equation 5.2.3, respectively. This implementation ensures a correct composition of the *Transport* phase and of the *Approach* phase.

5.2.4 Retreat phase

After the grasp/ungrasp transition, the *Retreat* phase comes following the *Approach* phase. The retreat target posture is a posture $\theta_{ret} = (\theta_{ret,arm}, \theta_{ret,hand})$, where $\theta_{ret,arm}$ is the arm target posture and $\theta_{ret,hand}$ is the hand target posture. The retreat arm target posture, $\theta_{ret,arm}$, selection is performed as explained in section 5.1.1.1 for a given retreat target hand pose.

5.2.4.1 Hand Retreat target posture selection

The hand target posture, $\theta_{ret,hand}$, is differently computed for pick and for place movements. In pick movements, the hand target posture for the *Retreat* phase is constrained by the fact that the object is held by the hand. Therefore, the retreat hand target posture is equal to the retreat hand start posture, i.e. $\theta_{ret,hand} = \theta_{0,hand}$.

For a successful ungrasp in place movements, the values of the retreat hand target posture are computed as described in Appendix A.3.2 considering a slightly increased aperture of the fingers with respect to the size of the object that is to be released.

5.2.4.2 Retreat direct movement

Similarly to the *Approach* phase, the *Retreat* phase is also formed by a *direct movement* that brings the joints from their target of the *Approach* phase (start posture, θ_0) to the current target posture, i.e. the retreat target posture, θ_{ret} . In this phase, it is reasonable to assume an obstacles-free region in the close proximity of the current goal, for the same reasons of problem feasibility mentioned in section 5.2.3.2.

Since the *Retreat* phase is the initial part of a *Transport* phase in the next movement, the formulation replicates the increasing velocity of the joints until the achievement of the retreat target posture

and it is given by Equations 5.2.9, 5.2.10 and 5.2.11:

$$\mathcal{T}_{ret}(t; \theta_{ret}) = \theta_0 + \frac{1}{3}(\theta_{ret} - \theta_0)(5\tau^4 - 2\tau^5) \quad (5.2.9)$$

$$\dot{\mathcal{T}}_{ret}(t; \theta_{ret}) = \omega_{tar}(2\tau^3 - \tau^4) \quad (5.2.10)$$

$$\ddot{\mathcal{T}}_{ret}(t; \theta_{ret}) = \alpha_{tar}(3\tau^2 - 2\tau^3) \quad (5.2.11)$$

where $\omega_{tar} = \frac{10}{3T}(\theta_{ret} - \theta_0)$, $\alpha_{tar} = \frac{2\omega_{tar}}{T}$, T is the total duration of the *Retreat* phase and $\tau = t/T$ is the normalized time. The trajectory of the joints in Equation 5.2.9 is monotonically increasing or decreasing according to the sign of $(\theta_{ret,k} - \theta_{0,k})$ for each k^{th} joint. Since θ_0 and θ_{ret} are within the physical range of the joints, Equation 5.2.9 guarantees that the values of the trajectory remain within their physical limits. Equation 5.2.10 and Equation 5.2.11 model the velocity and the acceleration of the *Retreat* phase, respectively. It is worth noting that the arm and the hand start with zero velocity and acceleration immediately after the grasp/ungrasp transition. The *Retreat* phase ends with velocity ω_{tar} and acceleration α_{tar} , which must be set as start boundary conditions of the *Transport* phase in a subsequent movement. This setting guarantees the correct composition of the *Retreat* phase with the following *Transport* phase.

5.3 Time parametrization

All the movements that are generated by the HUMP are time normalized and time is also discretized in a certain number of steps. Given the trajectories of each phase in a movement, their duration, T , the number of steps, N_{steps} , and size of each time step, Δt , need to be computed.

5.3.1 Phase period

In Rosenbaum et al., 2001 it is assumed that for each k^{th} joint there is an optimal movement duration, T_k , that is based on Fitts' Law applied to joint angular displacement, $\Delta\theta_k$ (Galen and Jong, 1995). Equations 5.3.1, 5.3.2 and 5.3.3 reports the formulation for a human-like time parametrization.

$$T_k = N_{steps} \frac{\Delta\theta_{max,k}}{\omega_{max,k}} + \lambda_k \ln(1 + \Delta\theta_k) \quad (5.3.1)$$

$$\Delta\theta_k = \sum_{i=1}^{N_{steps}} |\theta_{i,k} - \theta_{i-1,k}| \quad (5.3.2)$$

$$\Delta\theta_{max,k} = \max_{i=1, \dots, N_{steps}} |\theta_{i,k} - \theta_{i-1,k}| \quad (5.3.3)$$

where λ_k is the expense factor of the k^{th} joint, N_{steps} is the number of time steps in the movement,

$\omega_{\max,k}$ is the maximum allowed angular velocity of the k^{th} joint. The term $N_{steps} \frac{\Delta\theta_{\max,k}}{\omega_{\max,k}}$ guarantees that the maximum allowed angular velocity of the k^{th} joint is never reached. Then, since that all the joints start and end their movement synchronously (Breteler and Meulenbroek, 2006), an optimal total duration, T , is computed as a weighted average of the optimal duration for each joint (Equation 5.3.4).

$$T = \frac{\sum_{k=1}^n \lambda_k \Delta\theta_k T_k}{\sum_{k=1}^n \lambda_k \Delta\theta_k}. \quad (5.3.4)$$

5.3.2 Number of steps

The number of steps, N_{steps} , is computed according to Equation 5.3.5 and it is dependant on the distance between a start posture, θ_0 , and a target posture, θ_{tar} :

$$N_{steps} = \left\lceil N_m + (N_M - N_m) \frac{\|\theta_{tar} - \theta_0\|}{\|\theta_M - \theta_m\|} \right\rceil \quad (5.3.5)$$

where N_M and N_m are the acceptable maximum and the minimum number of steps, respectively.

5.3.3 Time step

A first approximation of the correct time step size is given by Equation 5.3.6:

$$\Delta t = \frac{T}{N_{steps}} \quad (5.3.6)$$

which is based on the trajectory with null boundary conditions, i.e. the movement starts and ends with zero velocity and acceleration. The obtained time step size, Δt , should provide a trajectory that does not reach the maximum allowed angular velocity, $\omega_{\max}$, and the maximum allowed angular acceleration, $\alpha_{\max}$, but with the settings of the boundary conditions ($\omega_0, \omega_{tar}, \alpha_0, \alpha_{tar}$) in Equation 5.2.2, this condition might not be respected. Therefore, the time step size, Δt , is increased cyclically of an arbitrarily small value, $\epsilon_t \geq 0$, if at least one joint acquires a maximum angular velocity greater than $\omega_{\max}$ or a maximum angular acceleration greater than $\alpha_{\max}$ during its trajectory. However, this is simply a safety check because, for a correct composition of the phases in the planned movements, boundary conditions should be set properly (see sections 5.2.3.2 and 5.2.4.2) and, more importantly, should not be set close to their physical limits. The obtained time step size can also be strategically multiplied by an arbitrarily small factor, $\mu \geq 1$, when a task requires significantly limited velocities and accelerations during the *Approach* and *Retreat* phases.

5.4 The HUMP main algorithms

The general implementation of the HUMP is given by the pseudo-code in Algorithms 5.1 and 5.2 that show how a movement (pick, place or move) is generated.

Algorithm 5.1 The HUMP main algorithm – Part 1**Inputs**

$\theta_0 = [\theta_{0,arm}, \theta_{0,hand}]$ – start posture
 ω_0, α_0 – boundary conditions
 P_{tran} – position and orientation of the transport phase target
 D_{obj} – size of the object that is being picked

Outputs

$\mathcal{T}, \dot{\mathcal{T}}, \ddot{\mathcal{T}}$ – angular positions, velocities and accelerations
 Δt – time steps of the planned trajectory

```

1: switch movement do
2:   case pick
3:      $\theta_{tran,hand} \leftarrow \text{handInverseKinematics}(D_{obj})$  // see section 5.2.2.1
4:      $\theta_{tran,arm}(P_{tran}) \leftarrow (5.1.4)-(5.1.7)$ 
5:      $\theta_{tran} = [\theta_{tran,arm}, \theta_{tran,hand}]$ 
6:      $\theta_b \leftarrow (5.1.17)-(5.1.20), \mathcal{T}_{tran}^{bkfor}(\theta_b) \leftarrow (5.1.14)$  // back-and-forth movement
7:   case place
8:      $\theta_{tran,hand} = \theta_{b,hand} = \theta_{0,hand}$  // an object is being transported
9:      $\theta_{tran,arm}(P_{tran}) \leftarrow (5.1.4)-(5.1.7)$ 
10:     $\theta_{tran} = [\theta_{tran,arm}, \theta_{tran,hand}]$ 
11:     $\theta_{b,arm} \leftarrow (5.1.17)-(5.1.20), \theta_b = [\theta_{b,arm}, \theta_{b,hand}]$ 
12:     $\mathcal{T}_{tran}^{bkfor}(\theta_b) \leftarrow (5.1.14)$  // back-and-forth movement
13:   case move
14:      $\theta_{tran,hand}$  is arbitrarily set // there is no manipulation
15:      $\theta_{tran,arm}(P_{tran}) \leftarrow (5.1.4)-(5.1.7)$ 
16:      $\theta_{tran} = [\theta_{tran,arm}, \theta_{tran,hand}]$ 
17:      $\theta_b \leftarrow (5.1.17)-(5.1.20), \mathcal{T}_{tran}^{bkfor}(\theta_b) \leftarrow (5.1.14)$  // back-and-forth movement

```

As illustrated in Figure 5.2, a move movement consists of a single movement phase *Transport*, T , while pick and place movements can also be composed by a sequence of an *Approach* phase, A , and of an *Retreat* phase, R . Thus, the *Transport* phase, is always planned for any of the three types of movements, while *Approach* (Algorithm 5.3) and *Retreat* (Algorithm 5.4) are only planned for pick and place movements.

The hardware of the robot provides the initial kinematical conditions of the robotic upper-limb system and the geometrical information concerning the body of the robot. The pose and the size of the objects in the workspace are provided by the vision system of the robot and considered either as obstacles or as items to be manipulated by a planned movement. Additionally, each phase of a movement is given with the pose of a target for the end-effector for the optimization of a corresponding human-like target posture that permits the computation of a *direct movement*. Only the *Transport phase* needs the selection of a

Algorithm 5.2 The HUMP main algorithm – Part 2

```

18:  $\mathcal{T}_{tran}^{direct}(\theta_0, \omega_0, \alpha_0, \theta_{tran}) \leftarrow (5.2.2)$  //transport direct movement
19:  $\mathcal{T}_{tran} \leftarrow (5.2.3), \dot{\mathcal{T}}_{tran} \leftarrow (5.2.4), \ddot{\mathcal{T}}_{tran} \leftarrow (5.2.5)$  //transport trajectory
20:  $\mathcal{T} = \mathcal{T}_{tran}, \dot{\mathcal{T}} = \dot{\mathcal{T}}_{tran}, \ddot{\mathcal{T}} = \ddot{\mathcal{T}}_{tran}$  //update the output
21:  $\Delta t = \Delta t_{tran}$  //see section 5.3
22: if approach then
23:   if straight-line then
24:     go to Algorithm 5.5 //add the approach trajectory with straight-line motion
25:   else
26:     go to Algorithm 5.3 //add the human-like approach trajectory
27:   end if
28: end if
29: if retreat then
30:   if straight-line then
31:     go to Algorithm 5.6 //add the retreat trajectory with straight-line motion
32:   else
33:     go to Algorithm 5.4 //add the human-like retreat trajectory
34:   end if
35: end if

```

bounce posture for the generation of a *back-and-forth movement* that, together with the previous *direct movement*, completes the resulting transport phase trajectory. The planning process starts with the selection of a transport target posture for the hand and, then, for the arm. The transport target posture permits the definition of a human-like *transport direct movement* from the initial conditions of the robot (line 18 of Algorithm 5.2). Then, to avoid obstacles in a scenario, a suitable bounce posture is selected (lines 6,11 and 17 of Algorithm 5.1). This bounce posture serves as a sub-goal for a *back-and-forth movement*, which is superimposed on the *transport direct movement* that has been previously computed (line 19 of Algorithm 5.2). The transport target posture that is finally achieved is the same as for the *transport direct movement* and only the selected path differs to guarantee collision avoidance. When the algorithm runs to generate move movements, the planning is complete after a proper time parametrization (lines 20 and 21 of Algorithm 5.2). On the contrary, pick and place movements may also have *Approach* and *Retreat* phases. When straight-line motion constraints are not demanded (see section 5.5), the *Approach* and *Retreat* phases are computed according to the human-like inspiration principles described in the sections 5.2.3 and 5.2.4. The *Approach* phase starts with the selection of an approach target posture (line 9 of Algorithm 5.3) and then an *approach direct movement*, which is from a transport to an approach target posture, is computed (line 10 of Algorithm 5.3). After the grasp/ungrasp transition, the *Retreat* phase begins with a retreat target selection (line 9 of

Algorithm 5.3 The HUMP approach phase algorithm**Inputs**

$\mathcal{T}, \dot{\mathcal{T}}, \ddot{\mathcal{T}}$ – angular positions, velocities and accelerations of the previous phase
 Δt – time steps of the previous trajectory
 P_{app} – position and orientation of the approach phase target
 D_{obj} – size of the object that is being picked

Outputs

$\mathcal{T}, \dot{\mathcal{T}}, \ddot{\mathcal{T}}$ – angular positions, velocities and accelerations
 Δt – time steps of the planned trajectory

```

1:  $\mathcal{T}(end) = \theta_0 = [\theta_{0,arm}, \theta_{0,hand}]$  // start posture of this phase
2:  $\dot{\mathcal{T}}(end) = \omega_0$  // start velocity of this phase
3:  $\ddot{\mathcal{T}}(end) = \alpha_0$  // start acceleration of this phase
4: switch movement do
5:   case pick
6:      $\theta_{app,hand} \leftarrow \text{handInverseKinematics}(D_{obj})$  // see section 5.2.3.1
7:   case place
8:      $\theta_{app,hand} = \theta_{0,hand}$  // an object is being transported
9:    $\theta_{app,arm}(P_{app}) \leftarrow (5.1.4)-(5.1.7), \theta_{app} = [\theta_{app,arm}, \theta_{app,hand}]$ 
10:   $\mathcal{T}_{app} \leftarrow (5.2.6), \dot{\mathcal{T}}_{app} \leftarrow (5.2.7), \ddot{\mathcal{T}}_{app} \leftarrow (5.2.8)$  // approach trajectory
11:   $\mathcal{T} \leftarrow [\mathcal{T}, \mathcal{T}_{app}], \dot{\mathcal{T}} \leftarrow [\dot{\mathcal{T}}, \dot{\mathcal{T}}_{app}], \ddot{\mathcal{T}} \leftarrow [\ddot{\mathcal{T}}, \ddot{\mathcal{T}}_{app}]$  // update the output
12:   $\Delta t \leftarrow [\Delta t, \Delta t_{app}]$  // see section 5.3

```

Algorithm 5.4) and with the computation of a *retreat direct movement*, which goes from the approach to a retreat target posture (line 9 of Algorithm 5.4). In conclusion, the phases of the movements are composed and a time parametrization of each phase is provided.

It is important to note that avoidance of the obstacles in the workspace is only addressed and solved during the *Transport* phase. This choice is made because we assume that both *Approach* and *Retreat* phases are executed in a collision-free region in proximity of the targets. This assumption is not too restrictive because the approaching and retreating directions are considered to lay on an obstacles-free path. Indeed, the movement can only be successfully planned if there is at least one obstacles-free path in the vicinity of the targets. In the selection procedure of the target and bounce postures, the planner also takes into account anatomical constraints. Thus, when these postures are computed, the generated upper-limb (arm-hand) movements are always anatomically feasible.

Algorithm 5.4 The HUMP retreat phase algorithm**Inputs**

$\mathcal{T}, \dot{\mathcal{T}}, \ddot{\mathcal{T}}$ – angular positions, velocities and accelerations of the previous phase
 Δt – time steps of the previous trajectory
 P_{ret} – position and orientation of the retreat phase target
 D_{obj} – size of the object that is being released

Outputs

$\mathcal{T}, \dot{\mathcal{T}}, \ddot{\mathcal{T}}$ – angular positions, velocities and accelerations
 Δt – time steps of the planned trajectory

```

1:  $\mathcal{T}(end) = \theta_0 = [\theta_{0,arm}, \theta_{0,hand}]$  // start posture of this phase
2:  $\dot{\mathcal{T}}(end) = \omega_0$  // start velocity of this phase
3:  $\ddot{\mathcal{T}}(end) = \alpha_0$  // start acceleration of this phase
4: switch movement do
5:   case pick
6:      $\theta_{ret,hand} = \theta_{0,hand}$  // an object is being transported
7:   case place
8:      $\theta_{ret,hand} \leftarrow \text{handInverseKinematics}(D_{obj})$  // see section 5.2.4.1
9:    $\theta_{ret,arm}(P_{ret}) \leftarrow (5.1.4)-(5.1.7), \theta_{ret} = [\theta_{ret,arm}, \theta_{ret,hand}]$ 
10:   $\mathcal{T}_{ret} \leftarrow (5.2.9), \dot{\mathcal{T}}_{ret} \leftarrow (5.2.10), \ddot{\mathcal{T}}_{ret} \leftarrow (5.2.11)$  // retreat trajectory
11:   $\mathcal{T} \leftarrow [\mathcal{T}, \mathcal{T}_{ret}], \dot{\mathcal{T}} \leftarrow [\dot{\mathcal{T}}, \dot{\mathcal{T}}_{ret}], \ddot{\mathcal{T}} \leftarrow [\ddot{\mathcal{T}}, \ddot{\mathcal{T}}_{ret}]$  // update the output
12:   $\Delta t \leftarrow [\Delta t, \Delta t_{ret}]$  // see section 5.3

```

5.5 Adding hand straight-line motion capabilities

The effectiveness of robotic manipulators is sometimes found in applications that demand straight-line motion in the operational space (Taylor, 1979; Wang and Lien, 1989). To this purpose, the implementation of the minimum jerk model has demonstrated to be capable of generating straight paths in presence of null boundary velocities and accelerations (Engelbrecht, 2001). This particular characteristic has been considered a limitation for human-like motion planning, which motivated researchers to adopt other models (e.g., the minimum torque change model by Uno et al. (1989)) and to use via-points constrained optimization to mimic curved human hand paths (Flash and Hogan, 1985).

The capability of planning straight-line operational-space motion is added to the HUMP to increase accuracy during the execution of pick-and-place tasks. In specific, it is required a straight-line path of the end-effector during the phases of *Approach* and *Retreat* that are critically important for their effects on

manipulation.

$$\min_{\theta_{tar,arm}} C(\theta_{tar,arm}) \quad (5.5.1)$$

$$\text{s.t.} \quad \mathbf{h}_{tar}(\theta_{tar}; \mathbf{P}_{tar}) \leq \zeta_{tar} \quad (5.5.2)$$

$$\theta_{m,arm} \leq \theta_{tar,arm} \leq \theta_{M,arm} \quad (5.5.3)$$

When straight-line paths in the operational space are demanded, it is assumed that the positions of the end-effector have the highest priority and obstacles avoidance can be ignored. In fact, it is assumed that the choice of a rectilinear motion solves the problem of searching a collisions-free path. This situation may occur, for instance, when a robotic arm operates in scenarios that are densely cluttered with obstacles and passage through narrow spaces is expected. For these reasons, the approach target posture and the retreat target posture selection problems (sections 5.2.3 and 5.2.4) are solved by the formulation of Equations (5.5.1)–(5.5.3). The pseudo-code that empowers the HUMP with straight-line motion generation capabilities in the *Approach* and *Retreat* phases of pick and place movements is reported by the Algorithms 5.5 and 5.6, respectively. Both algorithms are analogous in planning straight-line motion, except for the selection of the hand target posture that serves different purposes during the actions of picking and placing (lines 5–9 in Algorithm 5.5 and lines 5–9 in Algorithm 5.6). Then, the goal-directed arm target posture is selected, the number of steps from the start posture and the human-like time parametrization of the path are computed (lines 10,11 and 12 in Algorithm 5.5 and lines 10,11 and 12 in Algorithm 5.6). Lines 13 in Algorithm 5.5 and 13 in Algorithm 5.6 report how the size of positional steps along a rectilinear hand path and the size of linear increments of the end-effector joints are computed. For each point on the identified operational-space straight line path, positions, velocities and accelerations in the joints space are calculated and added to the phase trajectory (lines 15–21 in Algorithm 5.5 and lines 15–21 in Algorithm 5.6). Finally, the kinematics of the last step is computed to complete the phase trajectory, which is then appended to the previous phases of the planned movement (lines 23–26 in Algorithm 5.5 and lines 23–26 in Algorithm 5.6). In conclusion, the resulting movement is featured with a human-like transport phase that is combined with approaching and retreating rectilinear paths with mostly constant velocities. Moreover, the selection of each trajectory step by the optimization of (5.5.1)–(5.5.3) ensures the respect of the physical limits of the joints along the path, while angular velocities and accelerations are checked to guarantee safety and feasibility (see also section 5.3).

Algorithm 5.5 The HUMP approach phase algorithm with straight-line motion**Inputs** $\mathcal{T}, \dot{\mathcal{T}}, \ddot{\mathcal{T}}$ – angular positions, velocities and accelerations of the previous phase Δt – time steps of the previous trajectory P_{app} – position and orientation of the approach phase target D_{obj} – size of the object that is being picked **Outputs** $\mathcal{T}, \dot{\mathcal{T}}, \ddot{\mathcal{T}}$ – angular positions, velocities and accelerations Δt – time steps of the planned trajectory

```

1:  $\mathcal{T}(end) = \theta_0 = [\theta_{0,arm}, \theta_{0,hand}]$  // start posture of this phase
2:  $\dot{\mathcal{T}}(end) = \omega_0$  // start velocity of this phase
3:  $\ddot{\mathcal{T}}(end) = \alpha_0$  // start acceleration of this phase
4:  $P_0 \leftarrow \text{armDirectKinematics}(\theta_{0,arm})$  // start position and orientation of the hand
5: switch movement do
6:   case pick
7:      $\theta_{app,hand} \leftarrow \text{handInverseKinematics}(D_{obj})$  // see section 5.2.3.1
8:   case place
9:      $\theta_{app,hand} = \theta_{0,hand}$  // an object is being transported
10:   $\theta_{app,arm}(P_{app}) \leftarrow (5.5.1)-(5.5.3), \theta_{app} = [\theta_{app,arm}, \theta_{app,hand}]$ 
11:   $N_{steps}(\theta_0, \theta_{app}) \leftarrow (5.3.5)$  // number of steps in the approach phase path
12:   $\Delta t \leftarrow [\Delta t, \Delta t_{app}]$  // from the human-like approach trajectory (see section 5.3)
13:   $\Delta_H = (P_{app} - P_0) / N_{steps}, \Delta\theta = (\theta_{app,hand} - \theta_{0,hand}) / N_{steps}$  // positional steps
14:   $\mathcal{T}_{app} = \theta_0, \dot{\mathcal{T}}_{app} = \omega_0, \ddot{\mathcal{T}}_{app} = \alpha_0, P_n = P_0, \theta_{n,hand} = \theta_{0,hand}$  // initialization
15:  for  $n \in \{1, \dots, N_{steps} - 1\}$  do
16:     $P_n \leftarrow P_n + \Delta_H$  // one linearly incremented step
17:     $\theta_{n,hand} \leftarrow \theta_{n,hand} + \Delta\theta, \theta_{n,arm}(P_n) \leftarrow (5.5.1)-(5.5.3)$ 
18:     $\theta_n = [\theta_{n,arm}, \theta_{n,hand}]$  // position at the  $n^{th}$  step
19:     $\omega_n = (\theta_n - \theta_{n-1}) / \Delta t_{app}$  // velocity at the  $n^{th}$  step
20:     $\alpha_n = (\omega_n - \omega_{n-1}) / \Delta t_{app}$  // acceleration at the  $n^{th}$  step
21:     $\mathcal{T}_{app} \leftarrow [\mathcal{T}_{app}, \theta_n], \dot{\mathcal{T}}_{app} \leftarrow [\dot{\mathcal{T}}_{app}, \omega_n], \ddot{\mathcal{T}}_{app} \leftarrow [\ddot{\mathcal{T}}_{app}, \alpha_n]$ 
22:  end for
23:   $\omega_{app} = (\theta_{app} - \theta_{N_{steps}-1}) / \Delta t_{app}$  // velocity on the approach target posture
24:   $\alpha_{app} = (\omega_{app} - \omega_{N_{steps}-1}) / \Delta t_{app}$  // acceleration on the approach target posture
25:   $\mathcal{T}_{app} \leftarrow [\mathcal{T}_{app}, \theta_{app}], \dot{\mathcal{T}}_{app} \leftarrow [\dot{\mathcal{T}}_{app}, \omega_{app}], \ddot{\mathcal{T}}_{app} \leftarrow [\ddot{\mathcal{T}}_{app}, \alpha_{app}]$ 
26:   $\mathcal{T} \leftarrow [\mathcal{T}, \mathcal{T}_{app}], \dot{\mathcal{T}} \leftarrow [\dot{\mathcal{T}}, \dot{\mathcal{T}}_{app}], \ddot{\mathcal{T}} \leftarrow [\ddot{\mathcal{T}}, \ddot{\mathcal{T}}_{app}]$  // update the output

```

Algorithm 5.6 The HUMP retreat phase algorithm with straight-line motion**Inputs** $\mathcal{T}, \dot{\mathcal{T}}, \ddot{\mathcal{T}}$ – angular positions, velocities and accelerations of the previous phase Δt – time steps of the previous trajectory P_{ret} – position and orientation of the retreat phase target D_{obj} – size of the object that is being picked **Outputs** $\mathcal{T}, \dot{\mathcal{T}}, \ddot{\mathcal{T}}$ – angular positions, velocities and accelerations Δt – time steps of the planned trajectory

```

1:  $\mathcal{T}(end) = \theta_0 = [\theta_{0,arm}, \theta_{0,hand}]$  // start posture of this phase
2:  $\dot{\mathcal{T}}(end) = \omega_0$  // start velocity of this phase
3:  $\ddot{\mathcal{T}}(end) = \alpha_0$  // start acceleration of this phase
4:  $P_0 \leftarrow \text{armDirectKinematics}(\theta_{0,arm})$  // start position and orientation of the hand
5: switch movement do
6:   case pick
7:      $\theta_{ret,hand} = \theta_{0,hand}$  // an object is being transported
8:   case place
9:      $\theta_{ret,hand} \leftarrow \text{handInverseKinematics}(D_{obj})$  // see section 5.2.4.1
10:  $\theta_{ret,arm}(P_{ret}) \leftarrow (5.5.1)-(5.5.3), \theta_{ret} = [\theta_{ret,arm}, \theta_{ret,hand}]$ 
11:  $N_{steps}(\theta_0, \theta_{ret}) \leftarrow (5.3.5)$  // number of steps in the retreat phase path
12:  $\Delta t \leftarrow [\Delta t, \Delta t_{ret}]$  // from the human-like retreat trajectory (see section 5.3)
13:  $\Delta_H = (P_{ret} - P_0) / N_{steps}, \Delta_\theta = (\theta_{ret,hand} - \theta_{0,hand}) / N_{steps}$  // positional steps
14:  $\mathcal{T}_{ret} = \theta_0, \dot{\mathcal{T}}_{ret} = \omega_0, \ddot{\mathcal{T}}_{ret} = \alpha_0, P_n = P_0, \theta_{n,hand} = \theta_{0,hand}$  // initialization
15: for  $n \in \{1, \dots, N_{steps} - 1\}$  do
16:    $P_n \leftarrow P_n + \Delta_H$  // one linearly incremented step
17:    $\theta_{n,hand} \leftarrow \theta_{n,hand} + \Delta_\theta, \theta_{n,arm}(P_n) \leftarrow (5.5.1)-(5.5.3)$ 
18:    $\theta_n = [\theta_{n,arm}, \theta_{n,hand}]$  // position at the  $n^{th}$  step
19:    $\omega_n = (\theta_n - \theta_{n-1}) / \Delta t_{ret}$  // velocity at the  $n^{th}$  step
20:    $\alpha_n = (\omega_n - \omega_{n-1}) / \Delta t_{ret}$  // acceleration at the  $n^{th}$  step
21:    $\mathcal{T}_{ret} \leftarrow [\mathcal{T}_{ret}, \theta_n], \dot{\mathcal{T}}_{ret} \leftarrow [\dot{\mathcal{T}}_{ret}, \omega_n], \ddot{\mathcal{T}}_{ret} \leftarrow [\ddot{\mathcal{T}}_{ret}, \alpha_n]$ 
22: end for
23:  $\omega_{ret} = (\theta_{ret} - \theta_{N_{steps}-1}) / \Delta t_{ret}$  // velocity on the retreat target posture
24:  $\alpha_{ret} = (\omega_{ret} - \omega_{N_{steps}-1}) / \Delta t_{ret}$  // acceleration on the retreat target posture
25:  $\mathcal{T}_{ret} \leftarrow [\mathcal{T}_{ret}, \theta_{ret}], \dot{\mathcal{T}}_{ret} \leftarrow [\dot{\mathcal{T}}_{ret}, \omega_{ret}], \ddot{\mathcal{T}}_{ret} \leftarrow [\ddot{\mathcal{T}}_{ret}, \alpha_{ret}]$ 
26:  $\mathcal{T} \leftarrow [\mathcal{T}, \mathcal{T}_{ret}], \dot{\mathcal{T}} \leftarrow [\dot{\mathcal{T}}, \dot{\mathcal{T}}_{ret}], \ddot{\mathcal{T}} \leftarrow [\ddot{\mathcal{T}}, \ddot{\mathcal{T}}_{ret}]$  // update the output

```

5.6 Planning human-like synchronous bimanual movements

Inspired from the hypothesis of a common human motor control mechanism for bi-manual coordination (section 3.4), the capabilities of the presented human-like upper-limb motion planner have been extended for the generation of bi-manual synchronous movements. The evidences that result from the adoption of a Generalized Motor Program (GMP), which was described by Kelso et al. (1979), Schmidt (1975), and Schmidt et al. (1979), at the genesis of human two-handed movements have been of inspiration for the present formulation of human-like synchronous bi-manual motion planning. Specifically, when the GMP works for the generation of two-handed movements, then *time* is a *common parameter* that is shared with both arms. On the contrary, *movement distance* is a *specific parameter* that concerns the motion of a single upper-limb (Schmidt et al., 1979). Therefore, the planning of uni-manual human-like trajectories that has been presented so far can be logically upgraded for human-like synchronous bi-manual movements by extending the joints space to include the left upper-limb of a humanoid robot. Although, the left upper-limb starts and stops moving with the right one (synchrony in *time*), it is featured by limb-related constraints that are dependent on the scope of a given task and on the type of movement that is being planned (specific *movement distances*). This upgrading was then tested with the robot ARoS (Figure 4.1a), it was developed in collaboration with Eira Araújo (2014) and proposes to be the logical continuation of the human-like asynchronous bi-manual planning mechanism presented by Silva et al. (2015).

To proceed for the description of a dual upper-limb posture the following formulation is used:

$$\theta = (\theta_{arm_R}, \theta_{hand_R}, \theta_{arm_L}, \theta_{hand_L}) \quad (5.6.1)$$

where, $\theta_{arm_R} \in \mathbb{R}^{n_R}$, $\theta_{arm_L} \in \mathbb{R}^{n_L}$ represent the right arm posture with n_R joints (Equation 5.6.2) and of the left arm posture with n_L joints (Equation 5.6.4), respectively. $\theta_{hand_R} \in \mathbb{R}^{m_R}$, $\theta_{hand_L} \in \mathbb{R}^{m_L}$ represent the right hand posture with m_R joints (Equation 5.6.3) and the left hand posture with m_L joints (Equation 5.6.5).

$$\theta_{arm_R} = (\theta_1, \dots, \theta_{n_R})^T \quad (5.6.2)$$

$$\theta_{hand_R} = (\theta_{n_R+1}, \dots, \theta_{n_R+m_R})^T \quad (5.6.3)$$

$$\theta_{arm_L} = (\theta_{n_R+m_R+1}, \dots, \theta_{n_R+m_R+n_L})^T \quad (5.6.4)$$

$$\theta_{hand_L} = (\theta_{n_R+m_R+n_L+1}, \dots, \theta_{n_R+m_R+n_L+m_L})^T \quad (5.6.5)$$

For the dual arm-hand system of ARoS, the manipulators are featured with $n_R = n_L = n = 7$ and $m_R = m_L = m = 4$. In this case, the following equations are obtained:

$$\boldsymbol{\theta}_{arm_R} = (\theta_1, \theta_2, \dots, \theta_7)^\top \quad (5.6.6)$$

$$\boldsymbol{\theta}_{hand_R} = (\theta_8, \theta_9, \theta_{10}, \theta_{11})^\top \quad (5.6.7)$$

$$\boldsymbol{\theta}_{arm_L} = (\theta_{12}, \theta_{13}, \dots, \theta_{18})^\top \quad (5.6.8)$$

$$\boldsymbol{\theta}_{hand_L} = (\theta_{19}, \theta_{20}, \theta_{21}, \theta_{22})^\top \quad (5.6.9)$$

The general concepts of human-like upper-limb motion planning that have been presented in section 5.1, the phase construction of movements in section 5.2 and the main algorithms in section 5.4 remain valid when dual arm-hand postures $\boldsymbol{\theta}$ of Equation 5.6.1 are considered in the formation of dual joints-space trajectories $\mathcal{T}$. The fundamental modifications for the planning of synchronous bi-manual movements concern the formulation of the non-linear optimization problems for the selection of dual target postures (section 5.6.1) and dual bounce postures (section 5.6.2). These formulations allow the integration for the generation of two-handed movements that logically augment the target posture selection and the bounce posture selection presented in section 5.1.1 and in section 5.1.2, respectively.

5.6.1 Dual target posture selection

A dual target posture, $\boldsymbol{\theta}_{tar}$, at $t = T$, where T is the movement duration (see section 5.3), is a posture that minimizes the displacements of the joints from the initial to the dual target posture, taking into account obstacle avoidance, inter-limb collision avoidance, joint limits and desired grip types at the end of a bi-manual movement (e.g., the grips at the moment of grasping). The dual target posture selection process is divided into the selection of a set of finger joints and a set of arm joints. Hence, a dual target posture is denoted by Equation 5.6.10:

$$\boldsymbol{\theta}_{tar} = (\boldsymbol{\theta}_{tar,arm_R}, \boldsymbol{\theta}_{tar,hand_R}, \boldsymbol{\theta}_{tar,arm_L}, \boldsymbol{\theta}_{tar,hand_L}) \quad (5.6.10)$$

where $\boldsymbol{\theta}_{tar,arm_R}$ and $\boldsymbol{\theta}_{tar,arm_L}$ are vectors with the values to be specified for the angular joint positions of the right and left arm, respectively. Analogously, $\boldsymbol{\theta}_{tar,hand_R}$ and $\boldsymbol{\theta}_{tar,hand_L}$ are vectors with the values to be specified for the angular joint positions for the fingers of the right and left hand, respectively. The hand target postures, $\boldsymbol{\theta}_{tar,hand_R}$ and $\boldsymbol{\theta}_{tar,hand_L}$, depend on the specific hardware of the end-effectors and on the type of movement that is generated. For our case studies with two three-fingered Barrett hands, the selection of the hand target postures is described by the inverse kinematics solutions in Appendix A.3.2. The selection of the arm target postures, $\boldsymbol{\theta}_{tar,arm_R}$ and $\boldsymbol{\theta}_{tar,arm_L}$, is processed by

searching a solution of the non-linear optimization problem (5.6.11)–(5.6.18).

$$\min_{\substack{\theta_{tar,arm_R} \\ \theta_{tar,arm_L}}} C(\theta_{tar,arm_R}) + C(\theta_{tar,arm_L}) \quad (5.6.11)$$

$$\text{s.t.} \quad \mathbf{h}_{tar,R}(\theta_{tar,R}; \mathbf{P}_{tar,R}) \leq \zeta_{tar,R} \quad (5.6.12)$$

$$\mathbf{h}_{tar,L}(\theta_{tar,L}; \mathbf{P}_{tar,L}) \leq \zeta_{tar,L} \quad (5.6.13)$$

$$\mathbf{h}_{obs,R}(\theta_{tar,R}; \zeta_{obs,R}) < \mathbf{0} \quad (5.6.14)$$

$$\mathbf{h}_{obs,L}(\theta_{tar,L}; \zeta_{obs,L}) < \mathbf{0} \quad (5.6.15)$$

$$\mathbf{q}_{tar}(\theta_{tar,arm_R}, \theta_{tar,arm_L}; \zeta_{arms}) \leq \mathbf{0} \quad (5.6.16)$$

$$\theta_{m,arm_R} \leq \theta_{tar,arm_R} \leq \theta_{M,arm_R} \quad (5.6.17)$$

$$\theta_{m,arm_L} \leq \theta_{tar,arm_L} \leq \theta_{M,arm_L} \quad (5.6.18)$$

The objective function (5.6.11) that is being minimized sums the cost functions (Equation 5.1.3) associated with the right and the left arm postures. The constraints of the dual arm target posture selection are described as follows:

- constraints (5.6.12) and (5.6.13) impose a determined relationship between the right and the left hand poses, which are obtained through the direct kinematics of the right and the left arm target postures, θ_{tar,arm_R} and θ_{tar,arm_L} , and the hand target poses, $\mathbf{P}_{tar,R}$ and $\mathbf{P}_{tar,L}$, which represent the target positions for the right hand, $\mathbf{O}_{tar,R} = (x_{tar,R}, y_{tar,R}, z_{tar,R})$, and left hand, $\mathbf{O}_{tar,L} = (x_{tar,L}, y_{tar,L}, z_{tar,L})$, and the target orientations of the right target frame $(\hat{\mathbf{x}}_{tar,R}, \hat{\mathbf{y}}_{tar,R}, \hat{\mathbf{z}}_{tar,R})$, and of the left target frame, $(\hat{\mathbf{x}}_{tar,L}, \hat{\mathbf{y}}_{tar,L}, \hat{\mathbf{z}}_{tar,L})$. $\zeta_{tar,R}$ and $\zeta_{tar,L}$ are vectors of allowed tolerances in positioning and orienting the end-effectors at their given targets;
- constraints (5.6.14) and (5.6.15) guarantee obstacle avoidance at the target postures $\theta_{tar,R}$ and $\theta_{tar,L}$, where $\zeta_{obs,R}(> 0)$ and $\zeta_{obs,L}(> 0)$ are a constant clearance distances from the obstacles in the workspace;
- constraints (5.6.16) ensure that there is no collision between the right and the left arms at their target postures, θ_{tar,arm_R} and θ_{tar,arm_L} , where $\zeta_{arms}(> 0)$ is a constant clearance inter-limb distance;
- constraints (5.6.17) and (5.6.18) ensure that the selected arm target postures, θ_{tar,arm_R} and θ_{tar,arm_L} , satisfy the joints limits, where $\theta_{m,arm,R}, \theta_{M,arm,R} \in \mathbb{R}^{n_R}$ and $\theta_{m,arm,L}, \theta_{M,arm,L} \in \mathbb{R}^{n_L}$ represent the lower and the upper mechanical joint limits of both arms, respectively.

The vectors $\mathbf{h}_{tar,R}$ and $\mathbf{h}_{tar,L}$ in Equations 5.6.12 and 5.6.13 constrain both hands to assume a determined position and orientation when the targets are reached. These vectors are composed as $\mathbf{h}_{tar,R} = (h_{pos,R}, h_{or,R})^\top$ and $\mathbf{h}_{tar,L} = (h_{pos,L}, h_{or,L})^\top$, where $h_{pos,R}$ and $h_{pos,L}$ concern the constraints in position, while $h_{or,R}$ and $h_{or,L}$ are related to those ones in orientation. The functions $h_{pos,R}$ and $h_{pos,L}$ are formulated so that the distance between the targets and the palm of the hands is exactly d_{FH_R} for the right one and d_{FH_L} for the left one (Equations 5.6.19 and 5.6.20), which depend on manipulation with a specific end-effector (more details are given in Appendix A.3.2).

$$h_{pos,R}(\boldsymbol{\theta}_{tar,R}) = \|\mathbf{H}_R(\boldsymbol{\theta}_{tar,arm_R}) + d_{FH_R}(\boldsymbol{\theta}_{tar,hand_R}) \cdot \hat{\mathbf{z}}_{n_R}(\boldsymbol{\theta}_{tar,arm_R}) - \mathbf{O}_{tar,R}\|^2 \quad (5.6.19)$$

$$h_{pos,L}(\boldsymbol{\theta}_{tar,L}) = \|\mathbf{H}_L(\boldsymbol{\theta}_{tar,arm_L}) + d_{FH_L}(\boldsymbol{\theta}_{tar,hand_L}) \cdot \hat{\mathbf{z}}_{n_R+m_R+n_L}(\boldsymbol{\theta}_{tar,arm_L}) - \mathbf{O}_{tar,L}\|^2 \quad (5.6.20)$$

,while $h_{or,R}$ and $h_{or,L}$ constrain the orientation of both hands (Equations 5.6.21 and 5.6.22):

$$h_{or,R}(\boldsymbol{\theta}_{tar,arm_R}) = \|\hat{\mathbf{x}}_{n_R}(\boldsymbol{\theta}_{tar,arm_R}) - \hat{\mathbf{x}}_{tar,R}\|^2 + \|\hat{\mathbf{z}}_{n_R}(\boldsymbol{\theta}_{tar,arm_R}) - \hat{\mathbf{z}}_{tar,R}\|^2 \quad (5.6.21)$$

$$h_{or,L}(\boldsymbol{\theta}_{tar,arm_L}) = \|\hat{\mathbf{x}}_{n_R+m_R+n_L}(\boldsymbol{\theta}_{tar,arm_L}) - \hat{\mathbf{x}}_{tar,L}\|^2 + \|\hat{\mathbf{z}}_{n_R+m_R+n_L}(\boldsymbol{\theta}_{tar,arm_L}) - \hat{\mathbf{z}}_{tar,L}\|^2 \quad (5.6.22)$$

The constraints for the $\hat{\mathbf{x}}$ and $\hat{\mathbf{z}}$ are defined axes because the $\hat{\mathbf{y}}$ axes are then automatically determined by definition.

The vectors $\mathbf{h}_{obs,R}$ and $\mathbf{h}_{obs,L}$ in Equations 5.6.14 and 5.6.15 ensure that there is no collision of the arms and of the hands with the body of the robot and with any obstacle in the workspace. Therefore, $\mathbf{h}_{obs,R}$ and $\mathbf{h}_{obs,L}$ can be written as $(\mathbf{h}_{obs_1}(\boldsymbol{\theta}_{tar,R}), \mathbf{h}_{obs_2}(\boldsymbol{\theta}_{tar,R}))^\top$ and $(\mathbf{h}_{obs_1}(\boldsymbol{\theta}_{tar,L}), \mathbf{h}_{obs_2}(\boldsymbol{\theta}_{tar,L}))^\top$, respectively. The vector $\mathbf{h}_{obs_1}$ is related to the constraints against the body of the robot (Equation 5.1.10), while the vector $\mathbf{h}_{obs_2}$ concerns the constraints to avoid the obstacles in the scenario (Equation 5.1.11) with clearance tolerances $\zeta_{obs,R}$ and $\zeta_{obs,L}$.

The vector $\mathbf{q}_{tar}$ (Equation 5.6.16) ensures that there is no spatial overlap between the right and the left upper-limbs at their target postures. Each component of this vector is formulated by Equation 5.6.23:

$$q_{tar}^{k,h}(\boldsymbol{\theta}_{tar,R}, \boldsymbol{\theta}_{tar,L}) = r_k + r_h + \zeta_{arms} - \|\mathbf{P}_k(\boldsymbol{\theta}_{tar,R}) - \mathbf{P}_h(\boldsymbol{\theta}_{tar,L})\| \quad (5.6.23)$$

In Equation 5.6.23, $\mathbf{P}_k(\boldsymbol{\theta}_{tar,R})$ and r_k represent to the center and the radius of the k^{th} sphere that models the right upper-limb, while $\mathbf{P}_h(\boldsymbol{\theta}_{tar,L})$ and r_h are the center and the radius of the h^{th} sphere

that models the left upper-limb. The parameter ζ_{arms} defines the clearance distance between the k^{th} and the h^{th} spheres for avoiding their overlapping (more details on the sphere-based modelling of the robotic upper-limbs of the robot ARoS are given in Appendix B).

5.6.2 Dual bounce posture selection

Similarly to the bounce posture selection procedure presented in section 5.1.2.3, the selection of a proper dual bounce posture, $\theta_b = (\theta_{b,1}, \dots, \theta_{b,n_R+m_R+n_L+m_L})^\top$ is here described. In particular, a dual bounce posture, θ_b , is composed of the bounce postures of both arms and of the bounce postures of both hands (Equation 5.6.24):

$$\theta_b = (\theta_{b,arm_R}, \theta_{b,hand_R}, \theta_{b,arm_L}, \theta_{b,hand_L}) \quad (5.6.24)$$

With the robot ARoS, $n_R = n_L = n = 7$ and $m_R = m_L = m = 4$, which means that $\theta_{b,arm_R} = (\theta_{b,1}, \dots, \theta_{b,7})^\top$, $\theta_{b,hand_R} = (\theta_{b,8}, \theta_{b,9}, \theta_{b,10}, \theta_{b,11})^\top$ for the right arm-hand and that $\theta_{b,arm_L} = (\theta_{b,12}, \dots, \theta_{b,18})^\top$, $\theta_{b,hand_L} = (\theta_{b,19}, \theta_{b,20}, \theta_{b,21}, \theta_{b,22})^\top$ for the left hand.

The dual bounce posture minimizes the displacement in the joints space from the dual start posture, θ_0 , to the dual bounce posture, θ_b , subject to following constraints:

- collision avoidance, over the all duration of the composite movement, with any obstacle in the workspace, with the body of the robot and, eventually, with the objects that are being manipulated during the entire trajectory;
- the mechanical joint limits are respected over the all duration of the composite movement;
- inter-limb collision avoidance, over the all duration of the composite movement;
- minimization of the cost function in Equation 5.6.25 that is associated with moving the robotic upper-limb from the dual start posture, θ_0 , to the dual bounce posture, θ_b .

$$C_{dual}(\theta_b) = \sum_{k=1}^{n_R+m_R+n_L+m_L} \lambda_k (\theta_{0,k} - \theta_{b,k})^2, \lambda_k \geq 0 \quad (5.6.25)$$

To formulate this selection problem, the composite movement, $\mathcal{T}(t)$ in Equation 5.1.13, is taken into consideration and the time $t \in [0, T]$ is discretized into equally spaced N_{steps} steps denoted as $t_i = i\Delta t_{step}$, where Δt_{step} is the step size and $i = 0, 1, \dots, N_{steps}$ (see section 5.3 for details). The convention adopted is that $\mathcal{T}^i = \mathcal{T}(t_i)$ represents the configuration of the joints at time step t_i during the trajectory.

These concepts are formalized as a non-linear constrained optimization problem (5.6.26)–(5.6.30):

$$\min_{\theta_b} C_{dual}(\theta_b) \quad (5.6.26)$$

$$\text{s.t. } \theta_m \leq \mathcal{T}^i \leq \theta_M \quad (5.6.27)$$

$$\mathbf{h}_b(\mathcal{T}^i; \boldsymbol{\varepsilon}_b^i) \leq \mathbf{0} \quad (5.6.28)$$

$$\mathbf{q}_b(\mathcal{T}_i; \varepsilon_{arms,i}) \leq \mathbf{0} \quad (5.6.29)$$

$$\theta_m \leq \theta_b \leq \theta_M \quad (5.6.30)$$

where $\theta_m, \theta_M \in \mathbb{R}^{n_R+m_R+n_L+m_L}$ are the vectors with the lower and the upper mechanical joint limits, respectively, $\mathbf{h}_b$ is a vector of non-linear functions defined to guarantee the avoidance of collisions, being $\boldsymbol{\varepsilon}_b^i = \boldsymbol{\varepsilon}_b(t_i)$ a vector of positive clearance distances at time t_i . For every instant t_i , constraints (5.6.27) guarantee that the trajectory of the joints do not overcome their physical limits; inequalities (5.6.28) ensure that there is no collision with: (i) the object to grasp (only in picking), (ii) the obstacles in the scenario, and (iii) the body of the robot. The constraints (5.6.30) ensure that the selected dual bounce posture is within the limits of the joints.

When manipulation is not required, the hand bounce postures, $\theta_{b,hand_R}$ and $\theta_{b,hand_L}$, are completely selected by the solution of the optimization problem given by (5.6.26)–(5.6.30). During transportation, some object is held by each hand for the entire duration of a trajectory. Therefore, in this case, $\theta_{b,hand_R} = \theta_{0,hand_R}$ and $\theta_{b,hand_L} = \theta_{0,hand_L}$. This implies a null contribution of the back-and-forth movement concerning the total motion of the fingers in Equation 5.1.13. When grasping is required, due to the simplification described in Appendix A.3.2, the middle finger is always opposite to the other two fingers of a Barrett hand, thus $\theta_{b,8} = 0$ for the right hand and $\theta_{b,19} = 0$ for the left hand of the robot ARoS. Additionally, $\theta_{b,9} = \theta_{b,10} = \theta_{b,11}$ for the right hand and $\theta_{b,20} = \theta_{b,21} = \theta_{b,22}$ for the left hand of the robot ARoS, since the fingers have the same lengths (see Appendix A.3.2 for details).

The vector of constraints $\mathbf{h}_b$ (Equation (5.6.28)) changes its composition according to the type of movement that is being planned. When grasping is not required, $\mathbf{h}_b$ is composed with $(\mathbf{h}_{b_1}, \mathbf{h}_{b_2})^\top$, where $\mathbf{h}_{b_1}$ and $\mathbf{h}_{b_2}$ guarantee the avoidance of collisions with the body of the robot and with any obstacle in the scenario, respectively. The tuning of $\boldsymbol{\varepsilon}_b^i = \boldsymbol{\varepsilon}_{obs}^i = \boldsymbol{\varepsilon}_{obs}(t_i)$ varies the minimum acceptable distance at time t_i with the obstacles. Similarly to the single-arm counterpart, Equations 5.1.21 and 5.1.22 are used to implement the elements that compose the vectors $\mathbf{h}_{b_1}(t_i)$ and $\mathbf{h}_{b_2}(t_i)$, respectively. In specific, each sphere that models both upper-limbs is checked for spatial overlap with the body of the humanoid robot and the obstacles in the workspace

When grasping is required, more attention is necessary on avoiding collision with the objects that are being picked, because they are in regions where the end-effectors are approaching to, but they are also obstacles until the completion of the grasp transitions. For this reason, this issue is addressed with specific

constraints in the dual bounce posture selection problem. Particularly, another vector of constraints, $\mathbf{h}_{b_3}$, is added to the vector $\mathbf{h}_b$, which is then written as $(\mathbf{h}_{b_1}, \mathbf{h}_{b_2}, \mathbf{h}_{b_3})^\top$. The total number of obstacles, n_{obs} , is decremented by two, because the objects that are going to be picked are treated separately by the constraints represented with $\mathbf{h}_{b_3}$. Additionally, $\boldsymbol{\varepsilon}_b^i = \boldsymbol{\varepsilon}_b(t_i) = (\varepsilon_{obs}(t_i), \varepsilon_{obj_R}(t_i), \varepsilon_{obj_L}(t_i))^\top$, where $\varepsilon_{obs}(t_i)$ is the minimum acceptable distance at time t_i with the obstacles, while $\varepsilon_{obj_R}(t_i)$ and $\varepsilon_{obj_L}(t_i)$ are the minimum acceptable distance at time t_i with the objects to pick by the right and the left hand, respectively.

$$\mathbf{h}_{b_3}^k(t_i) = 1 - (\mathbf{P}_k^i - \mathbf{O}_{obj_R})^\top \mathbf{R}_{obj_R}^\top \mathbf{A}_{obj_R,k}^i \mathbf{R}_{obj_R} (\mathbf{P}_k^i - \mathbf{O}_{obj_R}) \quad (5.6.31)$$

$$\mathbf{h}_{b_3}^h(t_i) = 1 - (\mathbf{P}_h^i - \mathbf{O}_{obj_L})^\top \mathbf{R}_{obj_L}^\top \mathbf{A}_{obj_L,h}^i \mathbf{R}_{obj_L} (\mathbf{P}_h^i - \mathbf{O}_{obj_L}) \quad (5.6.32)$$

Equations 5.6.31 and 5.6.32 introduce the k^{th} and the h^{th} components of the vector $\mathbf{h}_{b_3}(t_i)$, where k indexes the spheres that model the right arm-hand and h indexes the spheres that model the left arm-hand (Appendix B). $\mathbf{O}_{obj_R}$ and $\mathbf{O}_{obj_L}$ are the geometrical centres of the objects that are being picked by the right and the left hands, respectively. $\mathbf{A}_{obj_R,k}^i = (\text{diag}(r_{obj_R,x}, r_{obj_R,y}, r_{obj_R,z}) + (r_k + \varepsilon_{obj_R}^i)I)^{-2}$, wherein $r_{obj_R,x}$, $r_{obj_R,y}$, $r_{obj_R,z}$ are the dimensions on the axes of the ellipsoid wrapping the object to be picked by the right hand (Appendix B). Similarly, $\mathbf{A}_{obj_L,h}^i = (\text{diag}(r_{obj_L,x}, r_{obj_L,y}, r_{obj_L,z}) + (r_h + \varepsilon_{obj_L}^i)I)^{-2}$, wherein $r_{obj_L,x}$, $r_{obj_L,y}$, $r_{obj_L,z}$ are the dimensions on the axes of the ellipsoid wrapping the object to be picked by the left hand (Appendix B). I is a three-dimensional identity matrix, while r_k and r_h are the radii of the k^{th} and of the h^{th} sphere modelling the right and the left upper-limb, respectively. $\mathbf{R}_{obj_R}$ is the rotational matrix the object to be picked by the right hand, while $\mathbf{R}_{obj_L}$ is the rotational matrix the object to be picked by the left hand.

When grasping the same object by both hands is required, then the number of obstacles n_{obs} is decremented by one and Equations 5.6.31 and 5.6.32 are modified by taking into consideration that $\mathbf{O}_{obj_R} = \mathbf{O}_{obj_L} = \mathbf{O}_{obj}$, $\mathbf{R}_{obj_R} = \mathbf{R}_{obj_L} = \mathbf{R}_{obj}$ and $\varepsilon_{obj_R}^i = \varepsilon_{obj_L}^i = \varepsilon_{obj}^i$. Additionally, in this case, we have that $\mathbf{A}_{obj_R,k}^i = \mathbf{A}_{obj,k}^i$ and $\mathbf{A}_{obj_L,h}^i = \mathbf{A}_{obj,h}^i$.

The vector $\mathbf{q}_b$ (Equation 5.6.29) ensures that there is no spatial overlap between the right and the left upper-limbs at every the i^{th} time step. Each element of this vector is formulated by Equation 5.6.33:

$$q_b^{k,h}(t_i) = r_k + r_h + \varepsilon_{arms,i} - \|\mathbf{P}_k^i - \mathbf{P}_h^i\| \quad (5.6.33)$$

In Equation 5.6.33, $\mathbf{P}_k^i$ and r_k represent to the center and the radius of the k^{th} sphere at time $t = t_i$ that models the right upper-limb, while $\mathbf{P}_h^i$ and r_h are the center and the radius of the h^{th} sphere at time $t = t_i$ that models the left upper-limb. The parameter $\varepsilon_{arms,i}$ defines the clearance distance between the k^{th} and the h^{th} spheres at time $t = t_i$ for avoiding their overlapping (more details on the sphere-based

modelling of the robotic upper-limbs of the robot ARoS are given in Appendix B).

When the two hands are demanded to manipulate and transport one single objects, then additional constraints are necessary to avoid any possible drops and breakages. For this reason, in this case, the optimization problem (5.6.26)–(5.6.30) is also constrained by the following expression (5.6.34):

$$\mathbf{d}_b(\mathcal{T}_i, \mathbf{d}_{HH}^0) \leq \zeta_{HH} \quad (5.6.34)$$

where the components of the vector $\mathbf{d}_b$ constrain both hands in position and orientation during the manipulation of the object. At every i^{th} time step, the vector $\mathbf{d}_b^i$ can be decomposed into position and orientation constraint functions, $\mathbf{d}_b^i = (d_{b,pos}^i, d_{b,or}^i)^\top$, that obey to respect constant distances $\mathbf{d}_{HH}^0 = (d_{HH,pos}^0, d_{HH,\hat{y}}^0, d_{HH,\hat{z}}^0)^\top$ with given clearance tolerances $\zeta_{HH} = (\zeta_{HH,pos}, \zeta_{HH,or})^\top$. In position (Equation 5.6.35), $d_{b,pos}^i$ guarantees that the distance between the palm of the right hand $\mathbf{H}_R^i$ and the palm of the left hand $\mathbf{H}_L^i$ remains constantly equal to the initial one at the beginning of manipulation, i.e., $\mathbf{d}_{HH,pos}^0 = (\mathbf{H}_R^0 - \mathbf{H}_L^0)^\top$. In orientation (Equation 5.6.36), $d_{b,or}^i$ ensures that the difference between the $\hat{\mathbf{y}}_{n_R}^i$ axis of the right hand frame and the $\hat{\mathbf{y}}_{n_R+m_R+n_L}^i$ axis of the left hand frame remains constantly equals to the initial one at the beginning of manipulation, i.e., $\mathbf{d}_{HH,\hat{y}}^0 = (\hat{\mathbf{y}}_{n_R}^0 - \hat{\mathbf{y}}_{n_R+m_R+n_L}^0)^\top$. Analogously, $d_{b,or}^i$ ensures that the difference between the $\hat{\mathbf{z}}_{n_R}^i$ axis of the right hand frame and the $\hat{\mathbf{z}}_{n_R+m_R+n_L}^i$ axis of the left hand frame remains constantly equals to the initial one at the beginning of manipulation, i.e., $\mathbf{d}_{HH,\hat{z}}^0 = (\hat{\mathbf{z}}_{n_R}^0 - \hat{\mathbf{z}}_{n_R+m_R+n_L}^0)^\top$. Therefore, the orientation on the $\hat{\mathbf{x}}$ axis is also maintained by definition.

$$d_{b,pos}^i = \left\| \mathbf{H}_R^i - \mathbf{d}_{HH,pos}^0 - \mathbf{H}_L^i \right\|^2 \quad (5.6.35)$$

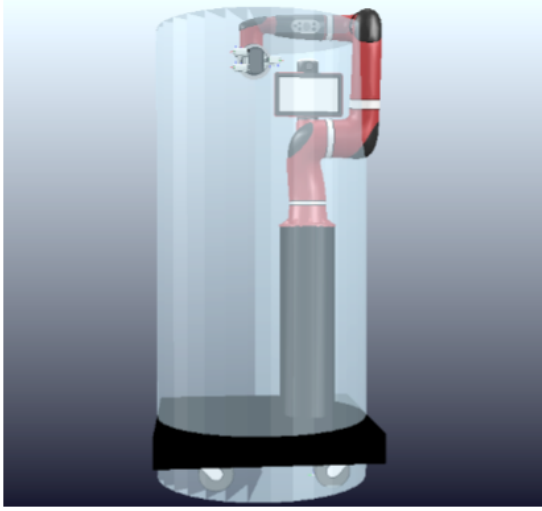
$$d_{b,or}^i = \left\| \hat{\mathbf{y}}_{n_R}^i - \mathbf{d}_{HH,\hat{y}}^0 - \hat{\mathbf{y}}_{n_R+m_R+n_L}^i \right\|^2 + \left\| \hat{\mathbf{z}}_{n_R}^i - \mathbf{d}_{HH,\hat{z}}^0 - \hat{\mathbf{z}}_{n_R+m_R+n_L}^i \right\|^2 \quad (5.6.36)$$

5.7 Upgrading for collaborative robots: modelling the robot Sawyer

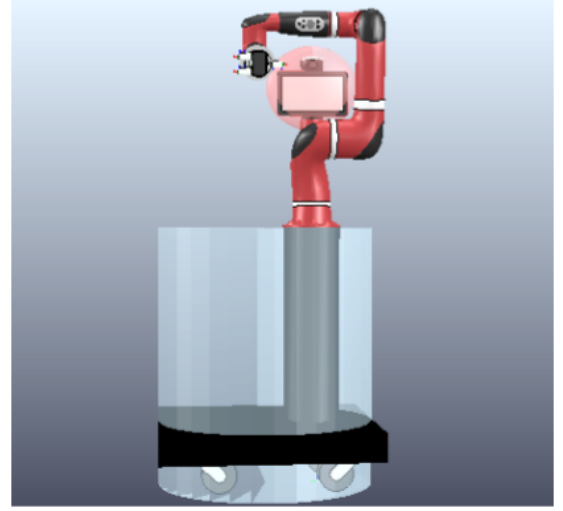
The described human-like motion planning technique is tailored for humanoid robotic platforms with particular geometrical specifications of the given manipulators. In this section, the planner is upgraded to be generalized for any system equipped with a 7-DOFs robotic arm that is typically used during collaborative tasks with human partners. The use case for this upgrading is the robot Sawyer (Figure 4.3) and the development for generalization was achieved in collaboration with Martins de Sá (2018). section 5.7.1 reports the modifications on modelling the body of the robot, while in section 5.7.2, an algorithm for modelling redundant 7-DOFs manipulators is presented.

5.7.1 Modelling the body

Modelling the body of the robot Sawyer with an elliptic cylinder, as it was designed for the robot ARoS (Appendix B), would negatively interfere with the planning process because there is a critical overlap with the workspace of the manipulator (Figure 5.3a).



(a) Model of the body



(b) Model of the body and of the head

Figure 5.3: The body of the robot Sawyer is basically constituted by its pedestal and the dimensions of the head are given by its screen and adjacent components. Figure 5.3a shows the representation of the body with an elliptic cylinder, while Figure 5.3b illustrates the ellipsoid wrapping the head and the pedestal modelled as a super-ellipsoid with shaping parameters $\epsilon_1 = 0.1$ and $\epsilon_2 = 1$.

For this reason, the body and the screen-related components (the head) are modelled separately (Figure 5.3b). The head is considered an obstacle in the workspace of the robotic upper-limb and, therefore, is represented as an ellipsoid for collision avoidance, like any other object in the scenario (see Appendix B). The body is modelled as a super-ellipsoid that is described by the surface in Equation 5.7.1, where x_s, y_s and z_s are the Cartesian coordinates of its center, ϵ_1 and ϵ_2 are shaping parameters and σ_x, σ_y and σ_z are its dimensions on the x, y and z axes, respectively.

$$F(x, y, z) = \left(\left(\frac{x - x_s}{\sigma_x} \right)^{\frac{2}{\epsilon_2}} + \left(\frac{y - y_s}{\sigma_y} \right)^{\frac{2}{\epsilon_2}} \right)^{\frac{\epsilon_2}{\epsilon_1}} + \left(\frac{z - z_s}{\sigma_z} \right)^{\frac{2}{\epsilon_1}} \leq 1 \quad (5.7.1)$$

The function $F(x, y, z)$ is also called *inside-outside* function because it can determine where a point x, y, z takes place with respect to the surface of the ellipsoid. Specifically, if a point $E = (x_e, y_e, z_e)$ belongs to the surface, then $F(x_e, y_e, z_e) = 1$. If the point E is outside the surface, then $F(x_e, y_e, z_e) > 1$ else $F(x_e, y_e, z_e) < 1$, if it is inside. This function is, therefore, used for constraining to obstacle avoidance in

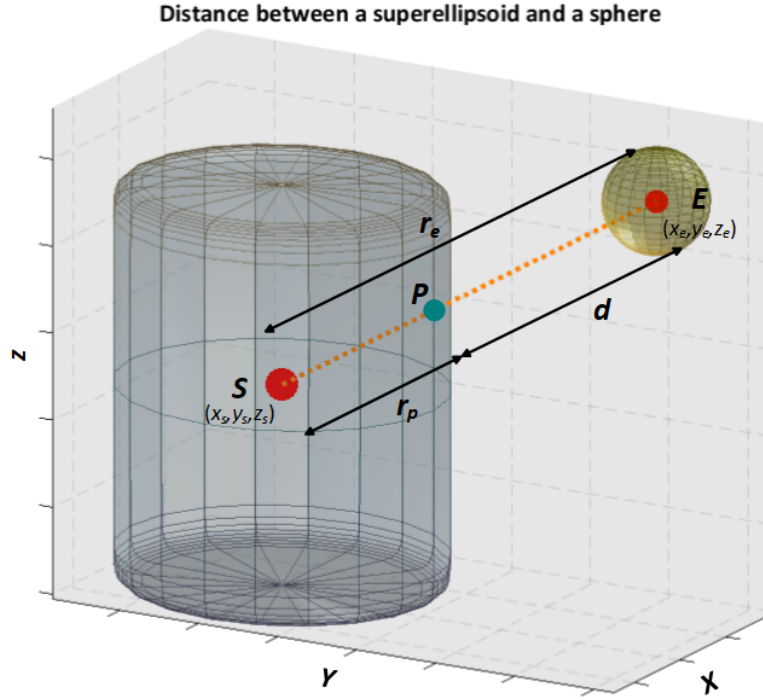


Figure 5.4: The distance between the center of a sphere and the surface of a super-ellipsoid is illustrated. To prevent collisions, the point E must be outside the super-ellipsoid and its distance from the point P must be greater than the radius of the sphere.

the formulation of the non-linear optimization problems (5.1.4)–(5.1.7) and (5.1.17)–(5.1.20).

With reference to Figure 5.4, the distance d from the center of a sphere to the point P on the surface of the super-ellipsoid is given by Equation 5.7.2, where P is expressed as a portion of the vector $\mathbf{r}_e$, i.e. $P = (\beta x_e, \beta y_e, \beta z_e)$. Therefore, from the equivalence $F(\beta x_e, \beta y_e, \beta z_e) = 1$, the expression $\beta = F(x_e, y_e, z_e)^{-\frac{\epsilon_1}{2}}$ can be obtained.

$$d = |\mathbf{r}_e - \mathbf{r}_p| = |\mathbf{r}_e - \beta \mathbf{r}_e| = |\mathbf{r}_e| |1 - \beta| = |\mathbf{r}_e| \left| 1 - F(x_e, y_e, z_e)^{-\frac{\epsilon_1}{2}} \right| \quad (5.7.2)$$

Starting from the formulation of the distance d in Equation 5.7.2 and considering the radius, r_k , of the k^{th} sphere that is modelling a portion of the manipulator (see section 5.7.2), the target and the bounce posture selection problems (section 5.1) can be upgraded. Particularly, for the robot Sawyer, the functions in Equations 5.1.10 and 5.1.21 are replaced by Equations 5.7.3 and 5.7.4, respectively.

$$h_{obs_1}^k(\boldsymbol{\theta}_{tar}) = 1 - \left(\left(\frac{P_{k,x}(\boldsymbol{\theta}_{tar}) - x_s}{\sigma_x + r_k} \right)^{\frac{2}{\epsilon_2}} + \left(\frac{P_{k,y}(\boldsymbol{\theta}_{tar}) - y_s}{\sigma_y + r_k} \right)^{\frac{2}{\epsilon_2}} \right)^{\frac{\epsilon_2}{\epsilon_1}} - \left(\frac{P_{k,z}(\boldsymbol{\theta}_{tar}) - z_s}{\sigma_z + r_k} \right)^{\frac{2}{\epsilon_1}} \quad (5.7.3)$$

$$h_{b_1}^k(t_i) = 1 - \left(\left(\frac{P_{k,x}(t_i) - x_s}{\sigma_x + r_k} \right)^{\frac{2}{\epsilon_2}} + \left(\frac{P_{k,y}(t_i) - y_s}{\sigma_y + r_k} \right)^{\frac{2}{\epsilon_2}} \right)^{\frac{\epsilon_2}{\epsilon_1}} - \left(\frac{P_{k,z}(t_i) - z_s}{\sigma_z + r_k} \right)^{\frac{2}{\epsilon_1}} \quad (5.7.4)$$

Note that Equations 5.7.3 and 5.7.4 are valid when the super-ellipsoid is designed to be oriented as the world frame of the scenario. Moreover, for the robot Sawyer, setting $\epsilon_1 = 0.1$ and $\epsilon_2 = 1$, there is a sufficiently accurate representation of its pedestal.

5.7.2 Modelling the arm

As it is reported in Appendix B, the manipulators of the robot ARoS are represented by a set of spheres to formulate the constraints for collisions avoidance in the target and bounce posture selection problems described in section 5.1. This representation was specifically tailored for the robot ARoS, but it has been extended to redundant manipulators with a certain degree of generalization. For this purpose, the arm of the robot Sawyer presents a more complex structure with spatial offsets on the D-H frames of the joints θ_2, θ_4 and θ_6 . The size of these offsets are given by the D-H d_i parameters on even frames along the arm (see Appendix A.2). Algorithm 5.7 describes how a set of a maximum of $2n$ spheres with arbitrary radii are generated for a generic manipulator with n degrees of freedom.

A sphere is generated on each even-positioned D-H frame along the arm (line 4) and the D-H d parameters are checked to detect eventual offsets on the $\hat{z}$ axis (line 5). If there is no offset and the length of the analysed link is greater than a given threshold, d_{th} , then a sphere is created on the middle point between the position of the analysed D-H frame and the previous one (line 15). If there is an offset, a new sphere on the offset point is created (line 6) and two spheres on the middle points of the previous and the next links, respectively, are created when the threshold d_{th} is overcome (lines 8 and 11). If n is even, the last sphere entirely covers the tip of the manipulator, otherwise the position of the end-effector is computed (line 20) and two spheres are created at the 45% and 75% of the distance between the two last D-H frames. The Algorithm 5.7 assumes that the complete direct kinematics of the manipulator is known and the radii, S_{rad} , of the modelling spheres are correctly provided for a sufficient coverage of the physical parts of interest.

Figure 5.5 illustrates examples of two 7-DOFs manipulators with (Figure 5.5a) and without offsets (Figure 5.5b) on the even-positioned D-H frames. The maximum number of spheres for modelling a 7-DOFs manipulator with offsets is 14, while at the most 8 spheres are sufficient to model an offsets-free arm with the same degrees of freedom. For example, by selecting a threshold of $d_{th} = 350mm$, the arm of the robot Sawyer is represented by a set of 10 spheres that are positioned as illustrated in Figure 5.6. With the same settings, the Algorithm 5.7 can provide the modelling spheres of the arms for the robot ARoS, which are showed in Figure B.1a and positioned as reported in the Tables B.1 and B.2. However, in this particular case, the spheres that should be positioned on the right and left shoulders of ARoS are ignored by the planner (and, therefore, not reported) because their position does not change with the configuration of the robot. Consequently, the consideration of these spheres would not increase the range

Algorithm 5.7 Modelling a manipulator with a set of spheres**Inputs**

n – number of degrees of freedom
 $S_{rad} \in \mathcal{R}^{2n}$ – radii of the spheres
 DH_{arm} – D-H parameters of the arm
 ${}^W_O T$ – transformation matrix from the world frame to the origin frame of the arm
 d_{th} – threshold value of the parameter d for creating link spheres

```

1:  $P_0 \leftarrow \text{getPos}({}^W_O T, DH_{arm}, 0)$ ,  $P_A = P_B = P_0$  // position of the frame 0
2: for  $i \in \{1, \dots, n\}$  do
3:   if  $i \% 2 = 0$  then
4:     // i is even
5:      $P_B \leftarrow \text{getPos}({}^W_O T, DH_{arm}, i)$ ,  $\text{createSphere}(P_B, S_{rad}[2i])$ 
6:     if  $\text{getDHd}(i) \neq 0$  then
7:       // point with an offset
8:        $P_{off} \leftarrow \text{getPos}({}^W_O T, DH_{arm}, i - 1)$ ,  $\text{createSphere}(P_{off}, S_{rad}[2i - 2])$ 
9:       if  $\text{getDHd}(i - 1) \geq d_{th}$  then
10:         $\text{createSphere}((P_A + P_{off}) / 2, S_{rad}[2(i - 1) - 1])$ 
11:      end if
12:      if  $\text{getDHd}(i) \geq d_{th}$  then
13:         $\text{createSphere}((P_{off} + P_B) / 2, S_{rad}[2i - 1])$ 
14:      end if
15:    else
16:      // point without an offset
17:      if  $\text{getDHd}(i - 1) \geq d_{th}$  then
18:         $\text{createSphere}((P_A + P_B) / 2, S_{rad}[2(i - 1) - 1])$ 
19:      end if
20:    end if
21:     $P_A = P_B$ 
22:  else if  $i = n$  then
23:    // i is odd and equals to n
24:     $H \leftarrow \text{getPos}({}^W_O T, DH_{arm}, n)$  // position of the end-effector
25:     $\text{createSphere}(P_B + 0.45 \cdot (H - P_B), S_{rad}[2n - 1])$ 
26:     $\text{createSphere}(P_B + 0.75 \cdot (H - P_B), S_{rad}[2n])$ 
27:  end if
28: end for
  
```

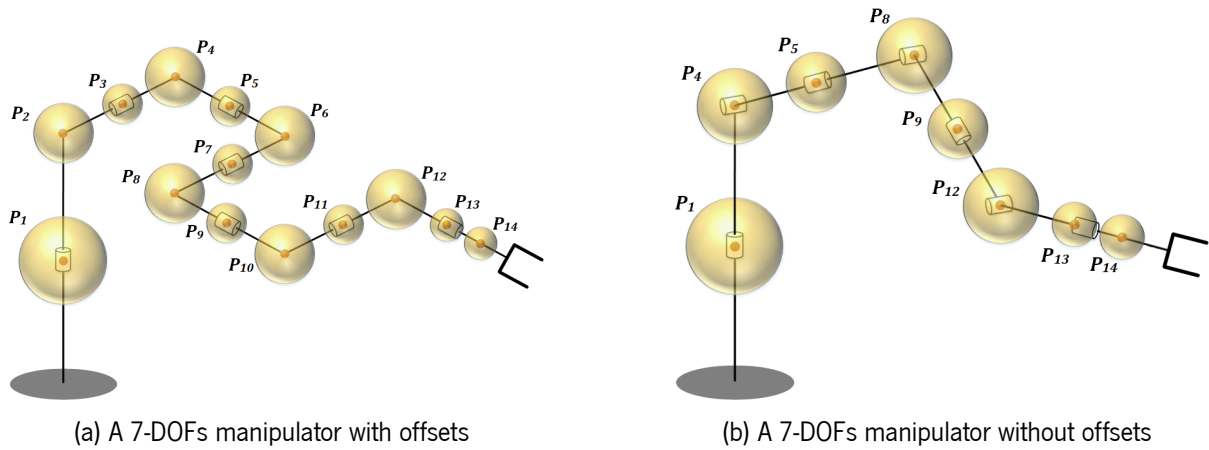



Figure 5.5: A 7-DOFs manipulator with offsets on the D-H frames 2,4 and 6 can be represented by a maximum of 14 spheres (Figure 5.5a). A 7-DOFs manipulator without these offsets can be represented with a maximum of 8 spheres (Figure 5.5b).

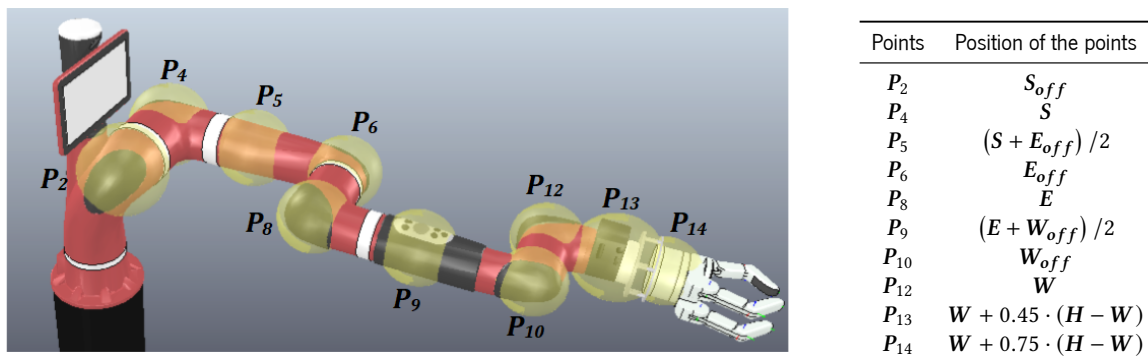


Figure 5.6: Model of the Sawyer manipulator with the position of the spheres

of possible collisions-free solutions during the search of a human-like trajectory. On the contrary, these spheres should be also included in the modelling of the arms if, for instance, the robot ARoS had another degree of freedom on its static body like in other humanoid systems (Figure 2.8).

Chapter 6

Experiments and results

This chapter provides an overview on the implementation of the algorithms described in chapter 5 and the results of different experiments with the robots ARoS and Sawyer are reported and discussed. Part of these results obtained experimenting with the robot ARoS are published in Gulletta et al., 2015; Gulletta et al., 2021. The proposed Human-like Upper-limb Motion Planner (HUMP) generates joints-space trajectories that here are proven to be feasible for applications on simulated and real environments. The direct kinematics of the robots is used to analyse the planned trajectories in the operational space and evaluate human-likeness according to the metrics introduced in section 3.3. The numerical results were obtained using an Intel(R) Core(TM) i7-4770 CPU 3.40GHz running Linux Mint 17 64-bits with a AMD Radeon HD 6570 video card and 8GB of RAM Memory. In section 6.1, an overview of the software settings is described. section 6.2 provides first results on planning human-like trajectories in absence of obstacles, while section 6.3 presents a series of tests with the robot ARoS acting on different scenarios cluttered with obstacles. Finally, in section 6.4, experimentations with the robot Sawyer are given.

6.1 Experimental setup

In this section, details concerning the implementation of the HUMP algorithms (sections 5.1 and 5.4) are given and the experimental setup for analyses and testing is described. The target posture and the bounce posture selection problems introduced in chapter 5 are non-linear constrained optimization problems, which are coded with **AMPL** (**A** Modeling **P**rogramming **L**anguage) (Fourer et al., 1989) and solved by using the solver **IPOPT** (**I**nterior **P**oint **OPT**imizer) (Wachter and Biegler, 2005). **AMPL** is a modeling language for linear and non-linear optimizations and provides an interface allowing the user to choose between solvers and have some options that may improve the solver performance. It is a high-level language that is based on modeling principles and allows for great flexibility. One of the advantages of

this language is the adoption of automatic differentiation, which permits to define the objective and the constraints functions without the formulation of derivatives. The **AMPL** is able to abstract mathematical models in a compact and perceptible algebraic notation (Fourer et al., 1989). **IPOPT** is an open-source software package for large-scale non-linear optimization, which implements an Interior-Point Filter Line search method (Wachter and Biegler, 2005). It was designed to find optimal solutions for non-linear constrained optimization problems. The optimal solution is obtained by solving a sequence of barrier problems for a decreasing sequence of barrier parameters converging to zero. It is important to note that there are other solvers for non-linear constrained optimization like KNITRO, SNOPT and LOQO, however none of these were tested in this work. The main reason is due to the fact that in our previous work (e.g., see Silva et al., 2011) the solvers KNITRO, SNOPT and LOQO presented high computational costs when compared to IPOPT.

The HUMP algorithms were developed as a C++ library¹, which exploits the functionalities offered by AMPL and IPOPT in the resolution of the target and bounce posture selection problems. The complete experimental setup is illustrated in Figure 6.1 and presents different software modules that communicate to one another by **ROS**, the **Robot Operating System** (Quigley et al., 2009). In specific, there is a vision

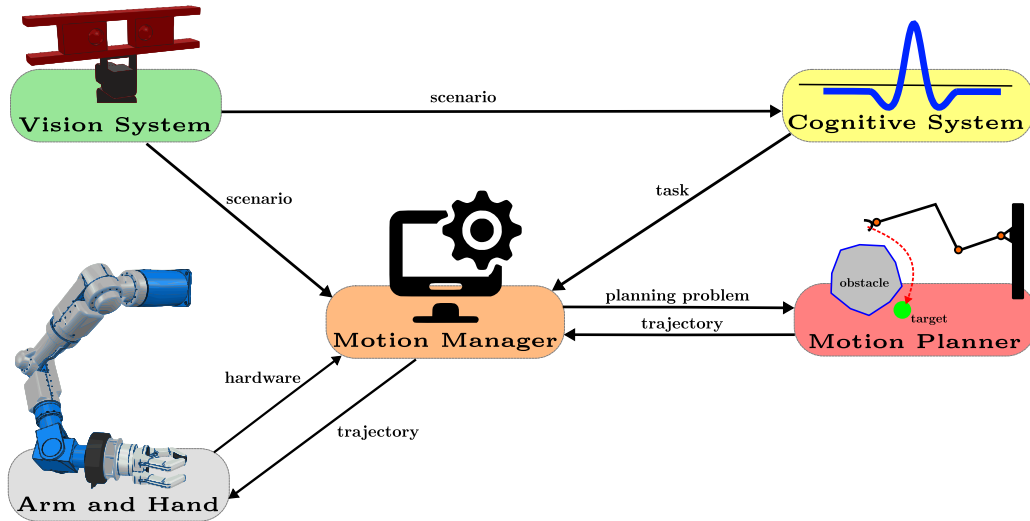


Figure 6.1: Sketch of the software architecture. The movement is planned using the target and grip type defined by the Cognitive System (which takes into account the intention of the action), the pose of the objects in the workspace given by the Vision System, and the initial conditions of the robotic arm and hand. Then, the Motion Planner provides a safe trajectory to the hardware for execution.

system can detect and provide positions and orientations of the objects in a given scenario. Inspired by the neuro-cognitive mechanisms behind human interaction (Bekkering et al., 2009; Sebanz et al., 2006), a cognitive system is implemented by a distributed multi-layer network in which each layer is formalized by

¹<https://github.com/zohannn/HUMP>

a Dynamic Neural Field (for details see Bicho et al., 2009; Erlhagen and Bicho, 2006). It endows the robot with the capacity to read motor intentions of the user and to select an adequate complementary action based on the inferred motor intention and other contextual information. Most of these complementary actions involve movements of the arm and hand, such as reaching and grasping an object as part of a handing over procedure or for placing it at a particular location in the workspace. Successfully handing over an object means, for instance, that the receiver usually has to select a complementary or opposite grasping behaviour to the one observed. Depending on the task, the cognitive model provides the robot with the high-level information about the grip type (e.g. precision or full as defined by Napier, 1956) to be chosen by the robot to achieve a desired target posture. Together with the information coming from the vision system concerning the type of the object and its pose (orientation and position), the decision about a complementary behaviour has to be translated by a Motion Planner into low-level motor commands (Figure 6.1). The Motion Manager (Figure 6.1) is a software module that translate the task-related directives of the cognitive system into a constrained planning problem. In particular, the Motion Manager is a ROS package² that collects the data supplied by the hardware and by the vision system, such as the start state of the robot, the pose of the objects in the workspace and the pose of the targets concerning a particular movement. Then, the specifications of a movement and the planning algorithm are defined. Afterwards, the planning problem is sent to a Motion Planner module, which replies with a safe trajectory in the joints space of manipulation.

The experimental setup in Figure 6.1 is then simulated as it is illustrated in Figure 6.2. For testing purposes, the information provided by the vision system and the cognitive model are replaced with a simulator environment built on **V-REP** (a **V**irtual **R**obot **E**xperimentation **P**latform). **V-REP** is a simulation environment that is widely used for fast algorithm prototyping and allows the users to easy control each object or model via ROS nodes (Rohmer et al., 2013). By the mean of this development framework, the robot ARoS and Sawyer acting in different scenarios are simulated during human-robot interaction and collaboration tasks.

The objective of the experiments is to demonstrate that the proposed planner HUMP is reliably capable of generating human-like collision-free arm movements for robotic manipulators. Additionally, this planning process must have a reasonable computational cost that allows fluent human-robot interactions and collaboration in joint tasks. For this reason, the time required by the HUMP to solve a specific problem is compared with that one needed by five selected sampling-based motion planners. This comparison provides a general idea about the complexity of a determined planning problem and approximately rate the performance of the proposed method. Therefore, the Motion Planner works as a container for different planning algorithms: the HUMP (chapter 5) and five widely used sampling-based motion planners, which

²https://github.com/zohannn/motion_manager

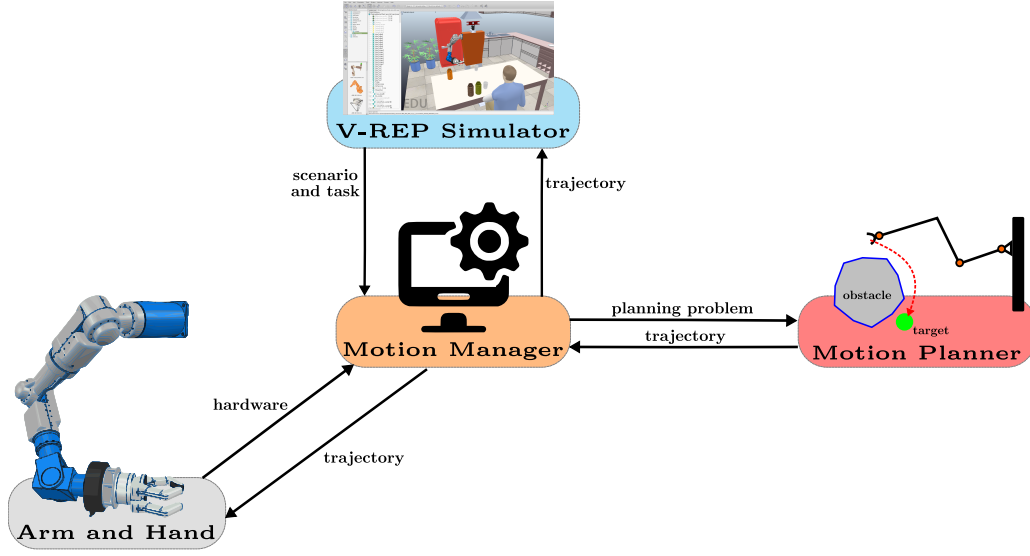


Figure 6.2: Sketch of the software modules in the simulated experimental setup. The V-REP environment simulates and provides all the necessary information concerning the humanoid robot, the objects in its workspace and the targets of the movements that are planned. The Motion Manager module elaborates this data and formulates a planning problem according to the user settings. The Motion Planner module runs a planning algorithm selected in a list of different available planners. Given the generated trajectory, the Motion Manager module can execute it with the simulated and the real robot.

are RRT (LaValle, 1998), RRTConnect (Kuffner and LaValle, 2000), PRM (Kavraki et al., 1996), RRT* and PRM* (Karaman and Frazzoli, 2011). Our implementation of the latter planners is based on the MoveIt! libraries³, which use the OMPL (the Open Motion Planning Library Sucan et al., 2012).

6.2 Move movements in an obstacles-free scenario

In this section, a move movement is planned for the robot ARoS (Figure 4.1a) from a start hand pose to a target hand pose with the same orientation. This simple experiment allows to understand how the proposed method performs in solving uncomplicated planning problems. Since no collision avoidance is involved, there is no need to integrate a *back-and-forth* movement (section 5.2.2.3), and therefore, the move movement is only composed by a *transport direct movement* (Equation 5.2.2). The hand path and the velocity profile of the move movement generated by the HUMP are illustrated in Figure 6.3. Given a Normalized Jerk Score (NJS) of 24.8 and a Number of Movement Units (NMU) of 1, the HUMP generated a trajectory that is clearly smooth with a human-like bell-shaped hand velocity profile. Additionally, the proposed method only required 567 ms to generate this trajectory.

Since human unconstrained reaching movements show to respect the *one-sixth* power law (Pollick et

³<http://moveit.ros.org>

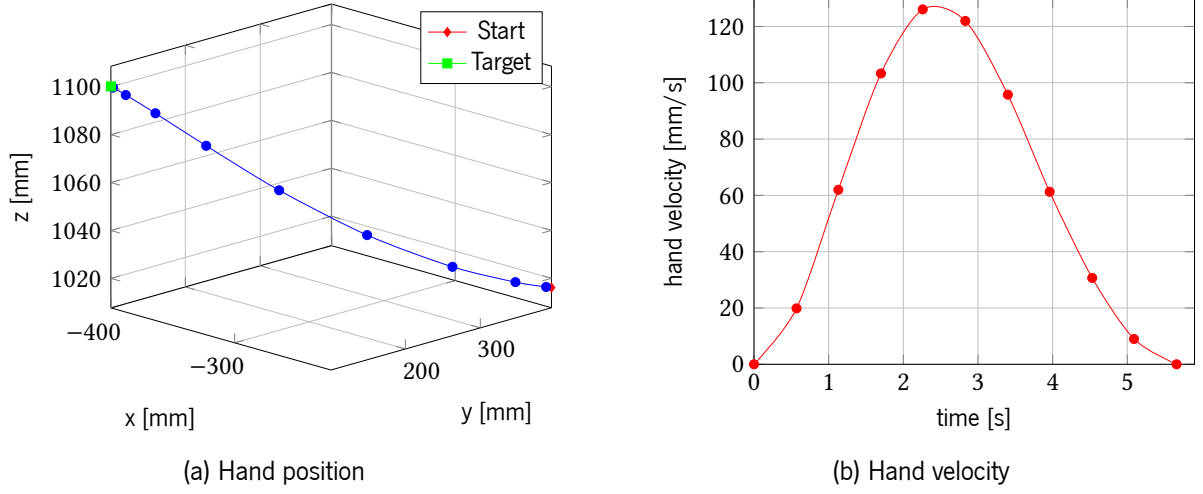


Figure 6.3: 3D-movement planned by the HUMP in absence of obstacles (NJS=24.8, NMU=1, solving time=567 ms).

al., 2009), the next set of experiments (section 6.2.1) is directed to fit this law of motion from random repetitions of planned move movements.

6.2.1 The one-sixth power law fitting

Experimenting with the robot ARoS, in this section the results of planning obstacles-free move movements with randomly selected target positions in the workspace are presented. The position of the target hand pose, (x_H, y_H, z_H) , is randomly selected such that $x_H \in [-600, -100]$, $y_H \in [200, 800]$ and $z_H \in [900, 1500]$, where coordinates are in millimetres and are related to the world frame (Figure 4.1). Specifically, in the reachable three-dimensional workspace, 600 move movements (divided into six sessions of 100 movements each) were generated by the HUMP algorithms (section 5.4).

Equation 3.3.2 represents a proportional relationship between the logarithm of the tangential hand velocity and the logarithm of the square curvature multiplied by the torsion (section 3.3). This lawful relationship can thus be verified by fitting a line to the values obtained by the planned movements. For a correct 1/6-PL fitting procedure, we considered the points of the trajectories with curvature greater than zero. Moreover, we excluded those points with absolute torsion less than 2 m^{-1} , in order to avoid physically unrealizable cups at locations where the torsion changes sign (Pollick et al., 2009). This choice implied the rejection of some trajectories, resulting in a total of 10190 movements used for the analysis (approximately 71% of all planned movements).

To simulate the experiment executed by Pollick et al. (2009), it is necessary to mimic aimless movements of the arm (scribbling-like paths). To this purpose, the orientation of the target hand pose cannot be randomly selected, because otherwise the risk is to select an unnatural hand orientation for a given hand

position. This may lead to unnatural target arm postures and, consequently, unnatural arm trajectories towards them. Since the goal is to generate human-like movements, the results of the study by Roby-Brami et al., 2003 to constrain the hand orientation are taken into consideration. The authors showed that the hand orientation in unconstrained reaching movements depends on the movement direction and on the movement distance. However, the variability of the hand roll is strongly dependent on the individual, but for each subject it is very small and mainly constant. Without loss of generality, we therefore considered a constant orientation of the target hand pose, which, for the sake of simplicity, does not privilege any plane of motion and allows the robot to reach the corners of the workspace with at least one posture that respects the joint limits with a sufficient span: $[\frac{\pi}{4}, -\frac{\pi}{2}, 0]^T$ in *Roll-Pitch-Yaw* angles (Craig, 2005).

Figure 6.4 illustrates the 1/6-PL fitting of the trajectories generated by the HUMP in one session. The

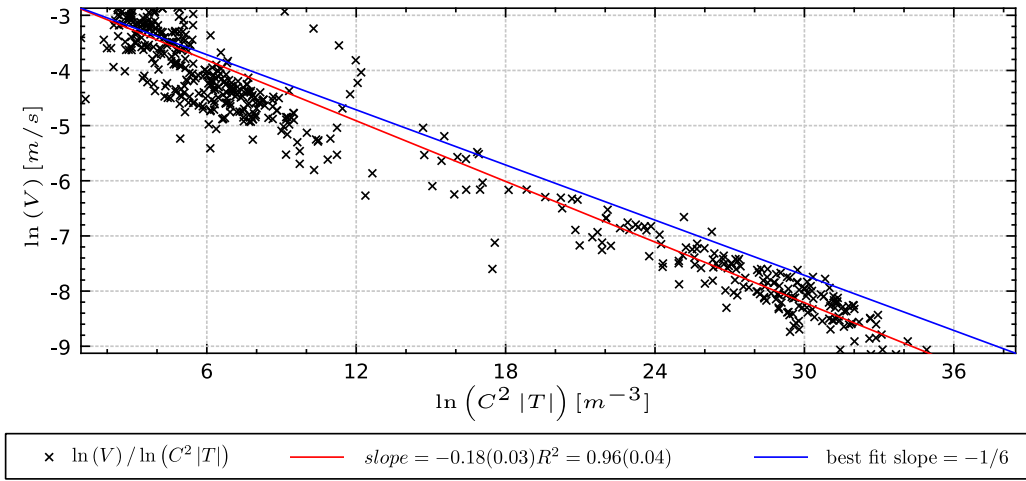


Figure 6.4: One-sixth power law fitting: one session of obstacles-free move movements planned by the HUMP in the three-dimensional workspace. V denotes the tangential velocity of the hand, C and T the curvature and the torsion of the hand path, respectively. The average NJS of the planned movements is 27.04 with a standard deviation of 7.88, while the NMU is always 1.

Planner	mean	sd	min	median	max	95% CI
HUMP	-0.174	0.019	-0.264	-0.17	0.032	[-0.175 -0.172]

Table 6.1: Descriptive statistics of the six sessions, 95% confidence interval (CI) of the slope in 1/6-PL fitting.

proposed human-like planner fits quite well the power law by a line with a slope that is close to $-1/6$ and with a good coefficient of determination $R^2 = 0.96$. It is worth noting that HUMP confirms a small and limited NJS and the NMU is always one.

The statistical analysis of all six sessions essentially confirmed the results illustrated in Figure 6.4 for one

session of 100 movements. As can be seen in Table 6.1, 95% of the movements generated by the HUMP exhibit slopes ranging between -0.175 and -0.172 that does not include the exact value $-1/6$, even though is very close to it. However, the small standard deviation and the restricted amplitude of the confidence interval show that the proposed method can generate movements that resemble the *one-sixth power law* with reliability.

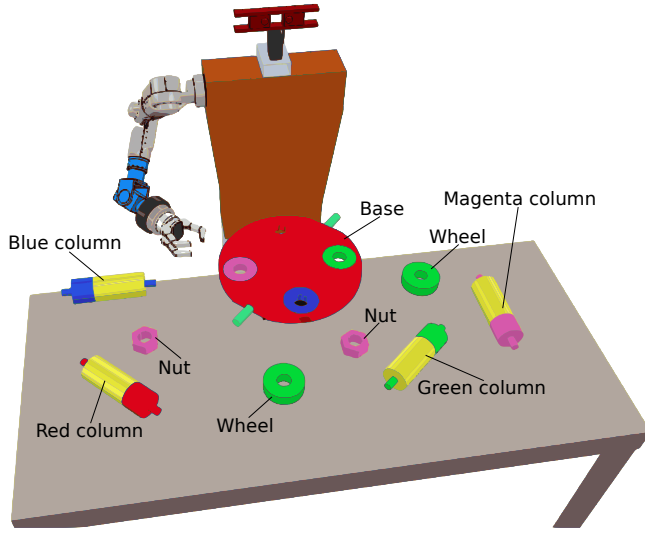
6.3 Tests with the robot ARoS in scenarios in presence of obstacles

In this section, tests with the robot ARoS acting in different scenarios cluttered with obstacles are showed and discussed. The scenarios are designed to analyse the performances of the proposed planner in the generation of single-arm and dual-arm pick, place and move movements. This design takes also into consideration the possibility of human-robot interaction and/or collaboration. Therefore, the objects in the scenarios can be manipulated by ARoS and by a human partner. For safety reasons, the workspace of the robot and that one of the human being are not shared, but the partners can communicate to one another and pass any object from one workspace to the other one. section 6.3.1 describes the results obtained during the joint construction of a toy vehicle, while in section 6.3.2 a sequence of movements to serve a drink is reported and analysed. In section 6.3.3, the results of a small user study on the perceived human-likeness of motion are discussed. A deeper analysis on the proposed human-like obstacle avoidance mechanism is provided in section 6.3.4, while section 6.3.5 reports experimentation on the generation of hand rectilinear paths in a densely cluttered environment. Finally, in section 6.3.6, planning synchronous bi-manual motion for humanoid robots is tested and results on different types of manipulation are discussed.

6.3.1 Toy vehicle assembly

This scenario is prepared by taking inspiration from our previous work (Bicho et al., 2011a; Bicho et al., 2011b; Sousa et al., 2015). Here a construction paradigm is presented: a robot and a human partner has to jointly assemble a toy vehicle (Figure 6.5b) from component parts that are initially distributed on a table in front of the robot ARoS (Figure 6.5a). Since for safety reasons the robot and the human partner operate in separate workspaces, the joint action paradigms include the transfer of objects between the co-workers as a subtask. This scenario allows to analyse the proposed planner in goal-directed action sequences composed of pick, place and move movements. Simple sequences of the action are addressed to maintain the focus on the evaluation of the human-likeness of the motion rather than on the complexity of the object manipulation. In particular, the analysis is focused on the concatenation of different movement phases

that permits to reach a continuous stream of motion that humans exhibit in object manipulation tasks. Additionally, as it was mentioned in section 6.1, this example is used for a comparison with five sampling-based planners to understand how the proposed HUMP algorithms generally perform.



(a) pieces of the toy vehicle



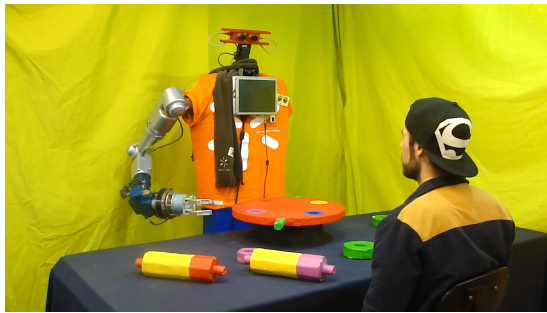
(b) the toy vehicle fully assembled

Figure 6.5: Toy vehicle assembly.

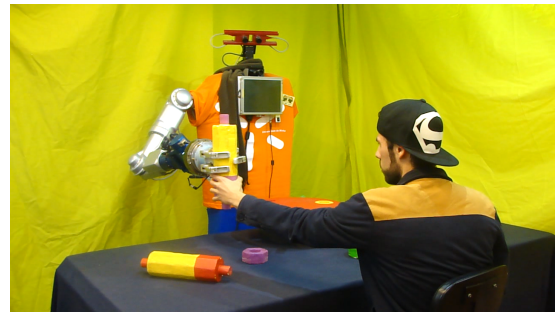
As an example of manipulation, a subtask is considered for assembling the magenta column as shown in Figure 6.6. This task starts with the human partner holding out the column for the robot. ARoS executes a reach-to-grasp movement (Figure 6.6a and Figure 6.6b) to pick up the column at the exchange position (Figure 6.6b), and then places it into the corresponding (magenta) hole in the base platform (Figure 6.6c). Finally, the robot returns to its start posture (Figure 6.6d). The pick and the place movements are composed by three phases: the *Transport*, the *Approach* and the *Retreat* phases. As described in section 5.2, each phase has a given hand target pose, which is the same for every planner used in this test.

The kinematic features of the pick movement planned by the HUMP are illustrated in Figure 6.7. Qualitatively, the pick movement is featured by a uni-modal bell-shaped hand velocity profile and a quasi-rectilinear hand path in each phase of motion. Additionally, the hand only stops at the moment of grasping and it is featured by an approach phase in the vicinity of the object to grasp. Quantitatively, the pick movement has a single-peaked hand velocity profile (NMU=1), its NJS is 1798,28 and its solving time is 2823 ms.

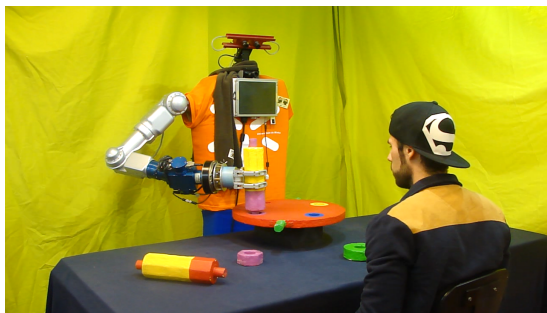
Table 6.2 shows the required time and the rate of success of the sampling-based planners in solving



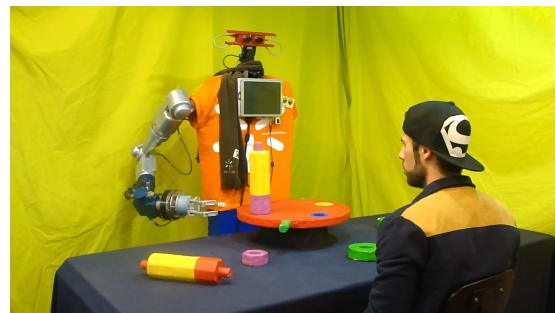
(a) start posture



(b) pick movement

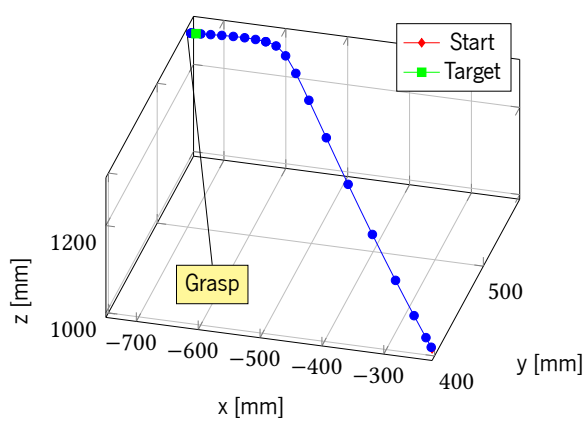


(c) place movement

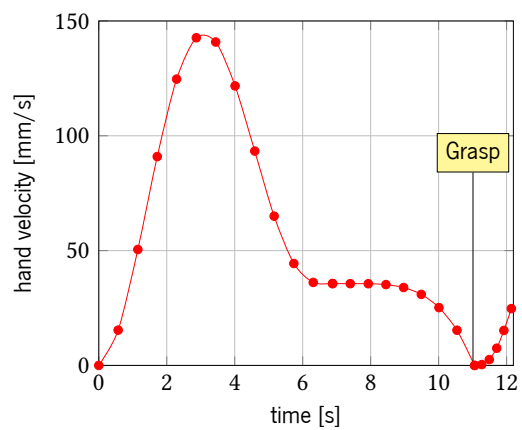


(d) move movement

Figure 6.6: Four snapshots that show the sequence of the movements planned by the HUMP in the toy vehicle assembly scenario. The complete video can be seen at the link <https://youtu.be/GgQeSUV-6UA>.



(a) Hand position



(b) Hand velocity

Figure 6.7: Pick movement planned by the HUMP (NJS=1798.28, NMU=1, solving time = 2823 ms).

the same planning problem. Table 6.2 reveals that RRTConnect solved the pick movement planning

Planner	Solving time: mean(SD) [ms]	Rate of success [%]
RRT	1676.25(1331.24)	32
RRTConnect	1203.28(707.49)	90
RRT*	4367.15(1607.88)	20
PRM	1602.49(1628.84)	75
PRM*	4889.51(1424.65)	78

Table 6.2: Results over 100 trials of the pick movement planning problem in the toy vehicle scenario (Figure 6.6).

problem 90 times over 100 trials. The other sampling-based planners showed a much lower success rate and RRTConnect is also the fastest planner with the smallest standard deviation. On the contrary, PRM showed the biggest standard deviation revealing an evident right-skewed distribution of solving time values.

The hand position and velocity of the place movement planned by the HUMP are illustrated in Figure 6.8. Due to the *Retreat* phase of the previous pick movement, the trajectory starts with a non-zero velocity and acceleration. The *Transport* phase is featured by a uni-modal hand velocity profile that ends with an accentuated *Approach* phase, which brings the column into the hole of the base. Moreover, the hand only stops during the release transition and then retreats away from the placing location. Again, the

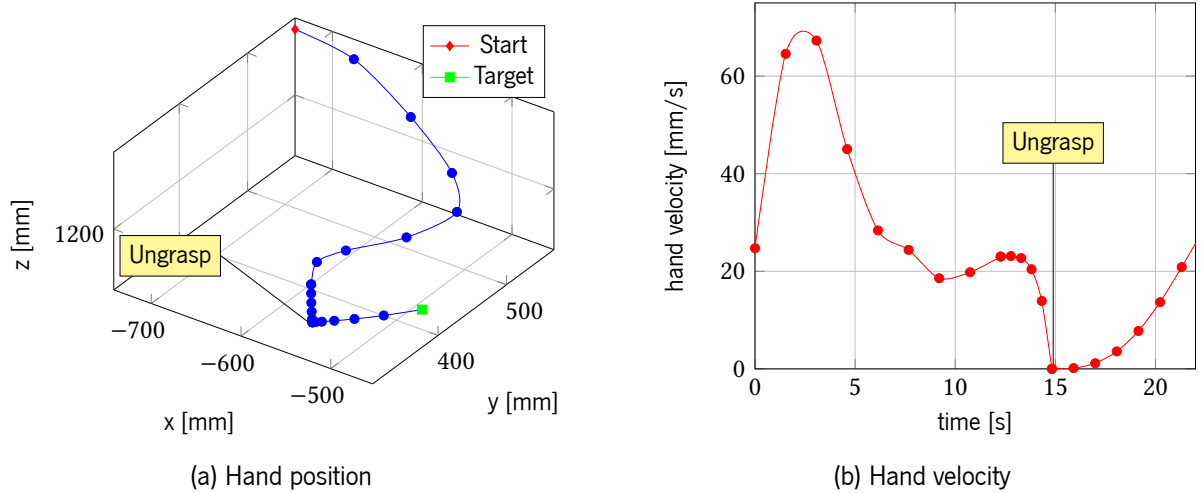


Figure 6.8: Place movement planned by the HUMP (NJS=15702.5, NMU=1, solving time = 2261 ms).

HUMP generates a trajectory with one NMU, but its NJS is significantly greater (15702.5) than that one of the previous pick movement. This might be due to the endpoint precision required in placing the column, which is featured by a prominent hand velocity profile during its *Approach* phase.

As summarized in Table 6.3, the rate of success over 100 trials shows that this planning problem was

complex for all the tested sampling-based planners. The HUMP required a solving time (2261 ms) that is generally greater than that one needed by RRT, PRM and RRTConnect, but significantly smaller than that one required by RRT* and PRM*.

Planner	Solving time: mean(SD) [ms]	Rate of success [%]
RRT	1345.61(892.02)	23
RRTConnect	1303.45(830.31)	33
RRT*	5148.77(76.05)	13
PRM	1007.24(816.18)	34
PRM*	4227.13(1647.97)	39

Table 6.3: Results over 100 trials of the place movement planning problem in the toy vehicle scenario (Figure 6.6).

The final movement of this task is a move movement that brings the robotic upper-limb back to its start posture. The kinematic characteristics of the movement planned by the HUMP are illustrated in Figure 6.9. The bell-shaped single-peaked hand velocity profile starts from a non-zero value, due to the *Retreat* phase of the previous place movement. Quantitatively speaking, the NMU remains 1 and the

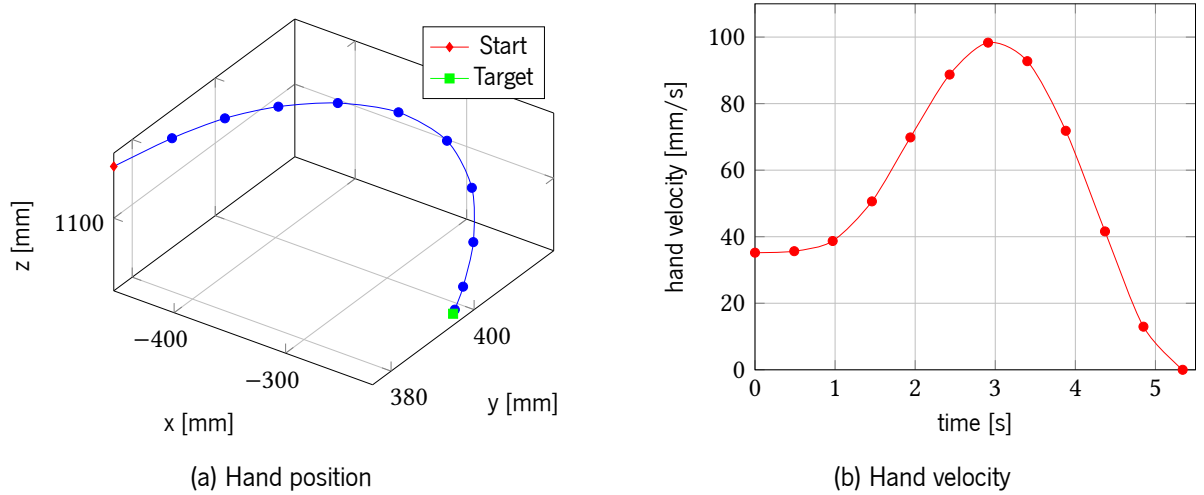


Figure 6.9: Move movement planned by the HUMP (NJS=18.06, NMU=1, solving time = 847 ms).

NJS is very small (18.06), which is consistent with a very smooth bell-shaped hand velocity profile with a single peak. Compared with the other planners, the HUMP produces the move movement in a minimum amount of time of 847 ms, which is also an index of the low complexity of the planning problem that has been solved successfully in most of the trials (Table 6.4). The concatenation of the three movements of this task is illustrated in Figure 6.10. Since the robot starts and ends its motion with the same posture, the start and the target hand positions correspond to the same point in the space (Figure 6.10a). The hand

Planner	Solving time: mean(SD) [ms]	Rate of success [%]
RRT	5264.53(143.64)	45
RRTConnect	1527.33(415.87)	99
RRT*	5169.89(52.52)	100
PRM	5507.87(414.20)	97
PRM*	5953.54(277.79)	96

Table 6.4: Results over 100 trials of the move movement planning problem in the toy vehicle scenario (Figure 6.6).

velocity profile, shown in Figure 6.10b, reveals that the grasp and the ungrasp transitions clearly define the first two phases of the place movement, while its *Retreat* phase connects to the move movement. These transitions are the only points in time during the whole task execution when arm and hand come to rest.

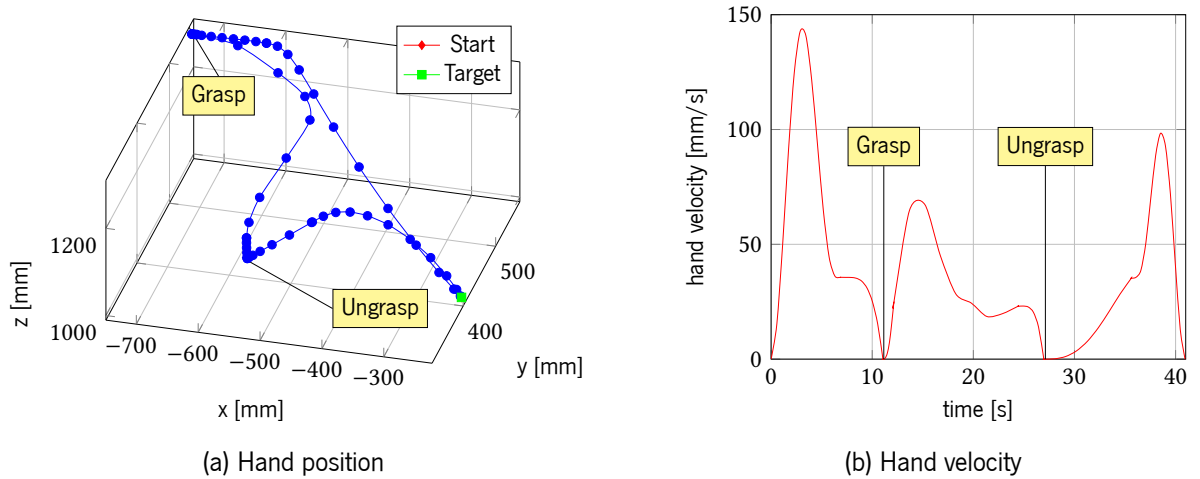


Figure 6.10: The complete task planned by the HUMF.

6.3.2 Serving a drink

To further test the capacity of generating human-like upper-limb movements with the robot ARoS in complex tasks, a second example in a household environment has been selected. In this scenario, the robot prepares a drink and a human partner observes a sequence of robotic actions intended to pour some orange juice into a glass. The workspace of the robot consists of various bottles and glasses that are on a table (Figure 6.11). The robot has to pick up the (open) bottle with orange juice, transport it to a pouring position close to the glass while avoiding the collision of its body with a brown bottle located at a position in between the orange bottle and the target location. The robot pours the liquid into the glass, moves the bottle again into an upright position to prevent spilling during transportation, and finally repositions the

bottle at its initial location (Figure 6.12). For the pouring action, it is assumed that the bottle moves mostly in a 2-dimensional plane so that the orientation can be modelled by a one-dimensional tilt angle. This is a realistic assumption for a pouring action in which the source container is taller than the receiving container (Yamaguchi et al., 2014). Since an autonomous flow control goes beyond the scope of this paper, it is further assumed that the desired tilt angle and the time interval for pouring are predefined. The pouring stops when the bottle is moved in an upright position again, which is called the post-pouring phase. As illustrated

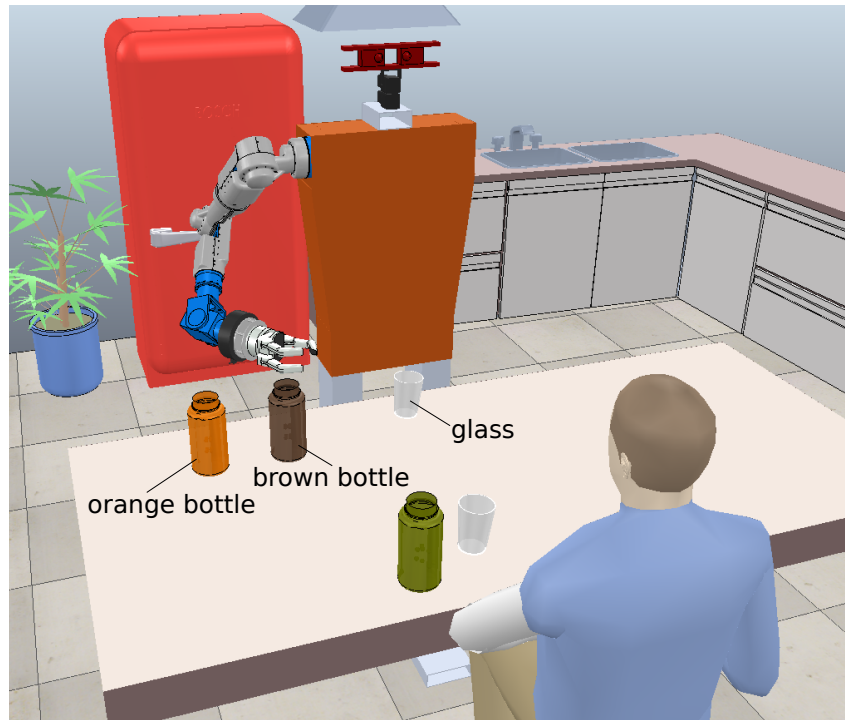


Figure 6.11: Serving a drink scenario.

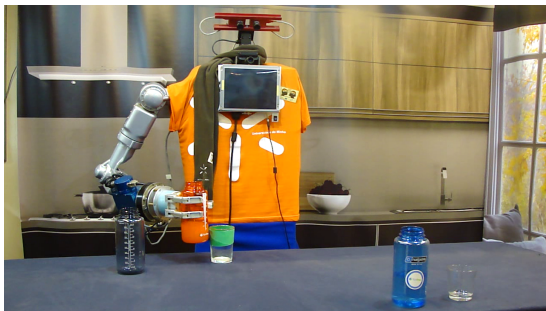
in Figure 6.12, the planning and the execution of the task can be segmented in 6 movement phases defined by specific sub-goals. The sequence starts with a *pick* movement directed towards the orange bottle (Figure 6.12b), followed by moving the bottle to the pouring position (*place* movement 1, Figure 6.12c), pouring the juice (*pouring* movement, Figure 6.12d), re-orientating the bottle (*post-pouring* movement, Figure 6.12e), placing the bottle at the initial position (*place* movement 2, Figure 6.12f), releasing the bottle and moving the arm back to its home position (*move* movement, Figure 6.12g). Figure 6.13 illustrates the kinematic characteristics of the *pick* movement planned by the HUMP. The movement is composed of a *Transport* phase with a bell-shaped hand velocity profile, a relatively long *Approach* phase towards the bottle to be grasped and a *Retreat* phase that detaches the bottle from the table. The movement has a single NMU and a NJS equals to 6377.39, which suggests a relative low level of complexity that seems confirmed by the high rates of success of the considered sampling-based planners in Table 6.5, even though RRT



(a) start posture



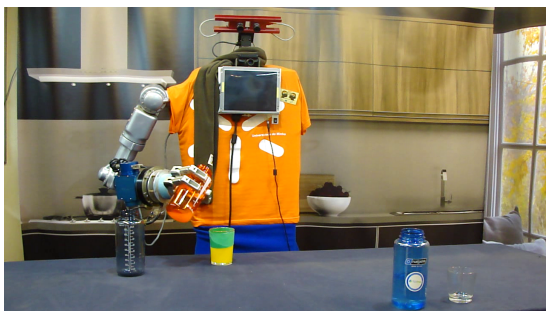
(b) pick movement



(c) place movement 1



(d) pouring movement



(e) post-pouring movement



(f) place movement 2



(g) move movement

Figure 6.12: Seven snapshots showing the sequence of the movements planned by the HUMP in the task of serving a drink. The complete video can be seen at the link https://youtu.be/_Zj1hZQrj5c.

and RRT* failed more often than others. Additionally, the HUMP solved the pick problem in significantly less time than the other optimization-based planners RRT* and PRM*, but took generally longer than the fastest planners RRT and RRTConnect (Table 6.5).

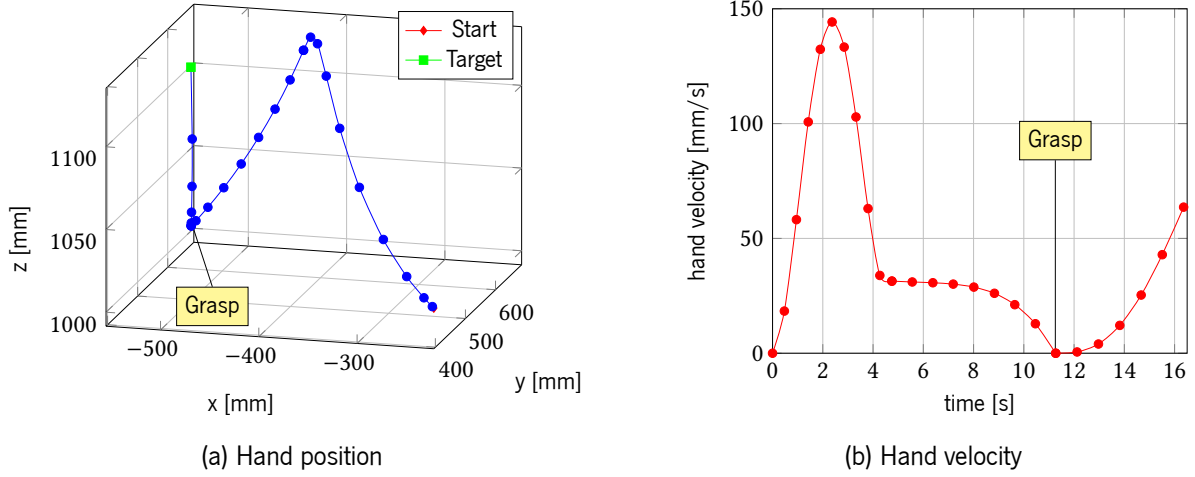


Figure 6.13: Pick movement planned by the HUMP in the drink serving task (NJS=6377.39, NMU=1, solving time=1225 ms).

Planner	Solving time: mean(SD) [ms]	Rate of success [%]
RRT	1030.07(673.16)	57
RRTConnect	390.36(87.65)	74
RRT*	4380.29(1550.46)	48
PRM	670.54(163.93)	69
PRM*	4232.59(1782.33)	75

Table 6.5: Results over 100 trials of the pick movement in the drink serving scenario (Figure 6.12).

The hand position and velocity profile of the first place movement are shown in Figures 6.14a and in Figure 6.14b, respectively. This movement is formed by a *Transport* phase and an *Approach* phase. A *Retreat* phase is not needed because the robot continues holding the bottle. The *Transport* phase is featured by a single-peaked, bell-shaped velocity profile and the *Approach* phase is shorter compared to the pick movement. The HUMP generated the movement with a small NJS of 933.52 and a single NMU in 848 ms, which is on average the minimum amount of time when compared to all probabilistic planners (Table 6.6).

The pouring movement is also composed of two phases, the *Transport* and the *Approach* phase, at the end of which, the juice is poured into the glass. As can be seen in Figure 6.15b, the *Transport* phase is featured by a low-velocity profile of the tangential hand velocity in its early seconds of motion and a second bigger

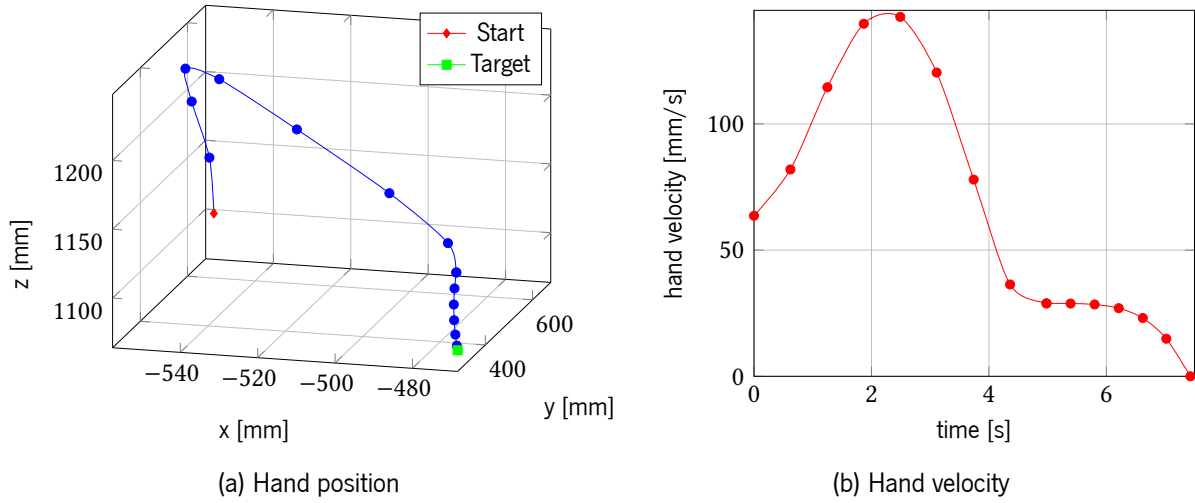


Figure 6.14: Place movement 1 planned by the HUMP in the drink serving task (NJS=933.52, NMU=1, solving time=848 ms).

Planner	Solving time: mean(SD) [ms]	Rate of success [%]
RRT	2040.46(1314.72)	35
RRTConnect	1069.60(439.03)	85
RRT*	4604.88(1384.01)	24
PRM	1199.69(1172.38)	83
PRM*	4722.37(1504.79)	79

Table 6.6: Results over 100 trials of the place movement 1 in the drink serving scenario (Figure 6.12).

velocity peak during the rest of the motion. This movement is thus characterized by a NMU of two, which is often observed in human hand motion as well, in particular, in the proximity of an obstacle (here the glass) that has to be avoided (Rosenbaum et al., 2001). In this particular case, the hand detour in position is mainly caused by its high rate of change in orientation, which contributes in giving a rise to a double peak in the tangential velocity of the hand (Grimme et al., 2012; Lommertzen et al., 2009; Rosenbaum et al., 2001). Further details about human-like kinematics are described in section 3.1.4. Table 6.7 shows that the HUMP solved this planning problem in 1603 ms, generally much faster than RRT* and PRM*, but on average much longer than RRTConnect, which was rather the fastest planner. The rate of success of the probabilistic planners over 100 trials revealed that this planning problem was quite complex to solve with the PRM* giving the highest rate.

The post-pouring movement is meant to rotate the bottle towards a given orientation to avoid that the remaining juice drops. Since in this case an accurate approaching phase is not required and the bottle cannot

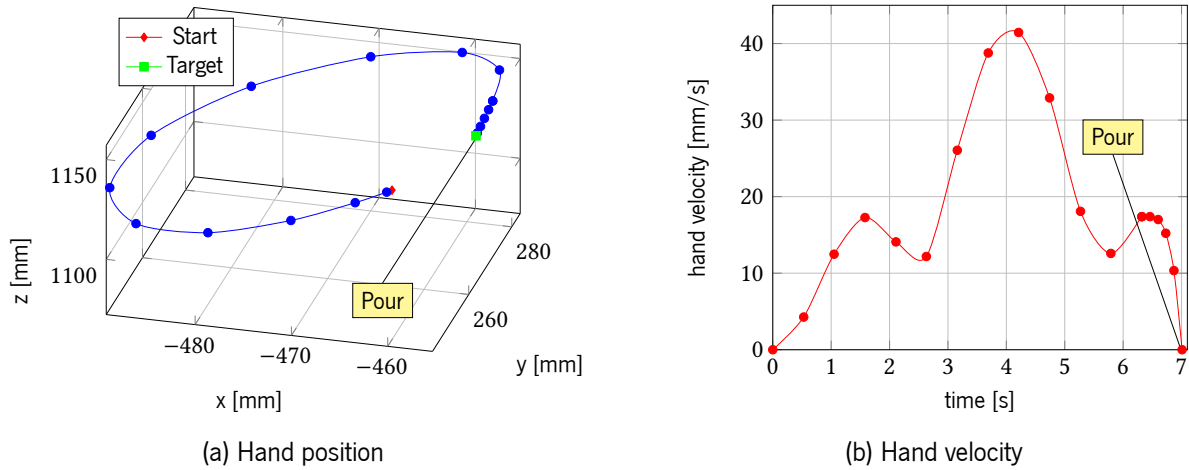


Figure 6.15: Pouring movement planned by the the HUMP in the drink serving task (NJS=5504.86, NMU=2, solving time=1603 ms).

Planner	Solving time: mean(SD) [ms]	Rate of success [%]
RRT	1546.19(1262.89)	47
RRTConnect	800.24(357.29)	62
RRT*	4484.59(1455.44)	29
PRM	2120.62(1021.04)	53
PRM*	4283.70(1862.24)	64

Table 6.7: Results over 100 trials of the pouring movement in the drink serving scenario (Figure 6.12).

be released, this movement consists of the *Transport* phase only. Similarly to the pouring movement, Figure 6.16 illustrates that the *Transport* phase is featured by an accentuated bi-phasic hand velocity profile (the NMU is 2). Similarly here, the rise of two peaks is due to a significant detour of the hand that changes its orientation in a relatively short period of time (Rosenbaum et al., 2001). The NJS score (199.65) indicates that the HUMP generated a very smooth movement and its planning time of 620 ms is on average slower than the fastest planner (RRTConnect), but fairly faster than the other probabilistic planners that generally scored a good rate of success (Table 6.8).

The second place movement transports the orange bottle back to its initial position and orientation. This movement is composed by a *Transport*, an *Approach* phase and a *Retreat* phase. As illustrated in Figure 6.17, the hand path during the *Transport* phase is highly curved to avoid collisions with the brown bottle. The hand velocity profile is uni-modal and bell-shaped with a smooth transition to the low-velocity *Approach* phase. The hand stops during the ungrasp transition, which starts the *Retreat* phase. This movement clearly has a NMU equals to 1 and a restrained NJS of 5566.22. As it is shown in Table 6.9, the

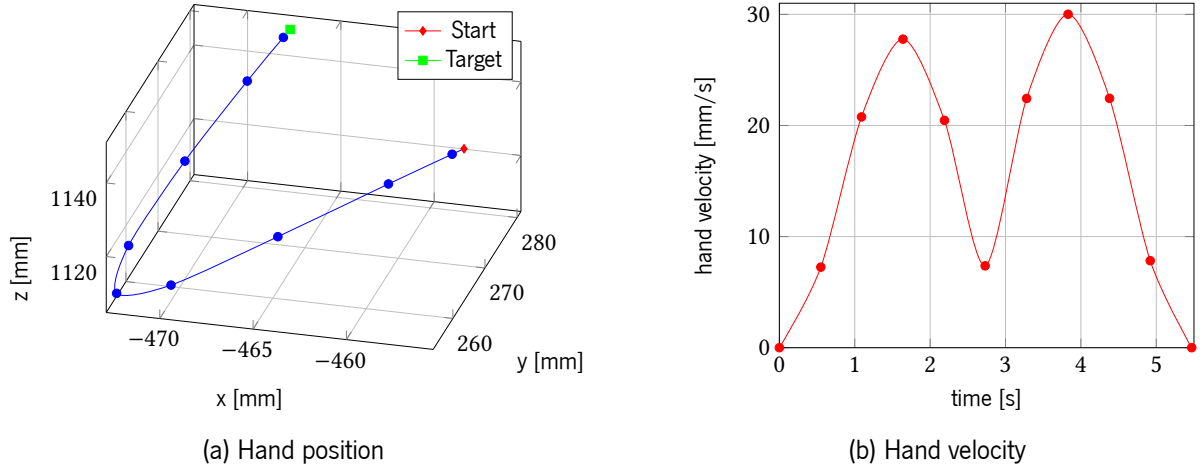


Figure 6.16: Post-pouring movement planned by the HUMP in the drink serving task (NJS=199.65, NMU=2, solving time=620 ms).

Planner	Solving time: mean(SD) [ms]	Rate of success [%]
RRT	746.45(587.38)	62
RRTConnect	383.40(185.97)	84
RRT*	4813.81(1072.94)	47
PRM	1057.89(1102.26)	81
PRM*	4226.54(1751.17)	83

Table 6.8: Results over 100 trials of the post-pouring movement in the drink serving scenario (Figure 6.12).

planning problem was quite complex except for the RRTConnect that had a rate of success of 73 %. Moreover, the RRTConnect also was the fastest planner and the HUMP was on average lightly slower than it demanding 1375 ms of solving time.

Planner	Solving time: mean(SD) [ms]	Rate of success [%]
RRT	1858.56(1080.45)	27
RRTConnect	1038.63(510.94)	73
RRT*	5110.14(564.79)	36
PRM	1866.93(966.32)	69
PRM*	4301.50(1744.09)	60

Table 6.9: Results over 100 trials of the place movement 2 in the drink serving scenario (Figure 6.12).

After having released the bottle, the robot ARoS moves the hand back to the start pose by the mean of a move movement that is shown in Figure 6.18.

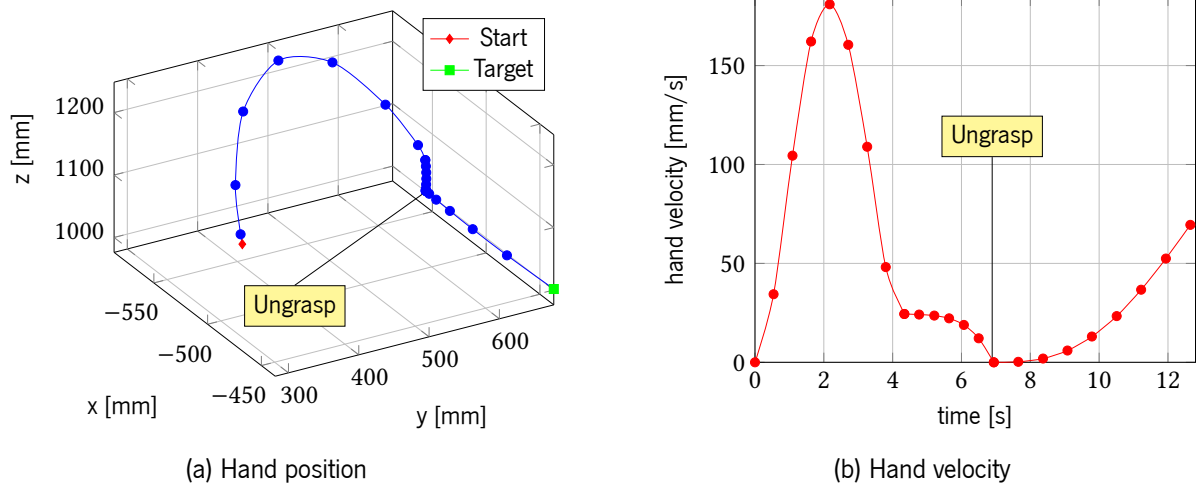


Figure 6.17: Place movement 2 planned by the HUMP in the drink serving task (NJS=5566.22, NMU=1, solving time=1375 ms).

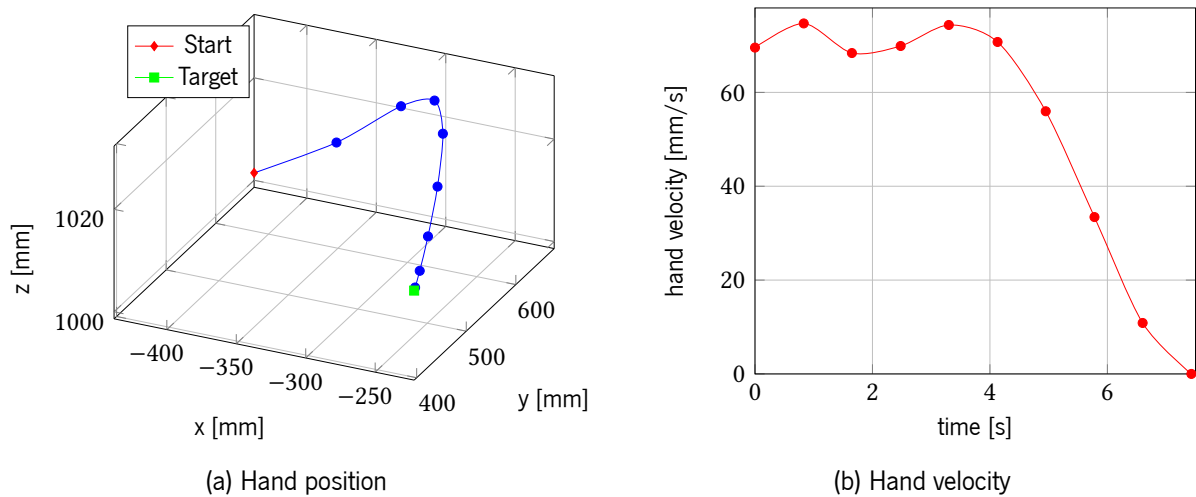


Figure 6.18: Move movement planned by the HUMP in the drink serving task (NJS=24.57, NMU=0, solving time=297 ms).

Planner	Solving time: mean(SD) [ms]	Rate of success [%]
RRT	5207.98(107.87)	93
RRTConnect	1855.15(431.90)	100
RRT*	5224.96(76.60)	53
PRM	5416.57(542.50)	99
PRM*	5888.02(295.42)	98

Table 6.10: Results over 100 trials of the move movement planning in the drink serving scenario (Figure 6.12).

Interestingly, this movement has zero NMU despite the occurrence of two velocity peaks which are, however, not detected since the increase/decrease of hand velocity between the adjacent maximum and minimum does not exceed the threshold value for defining a movement unit. Importantly, the evaluation of the *Retreat* phase of the previous movement and the move movement together (Figure 6.19) reveals that the HUMP generated a human-like reaching movement characterized by a NMU score of 1 and a bell-shaped velocity profile. The proposed method planned a very smooth move movement with a NJS of

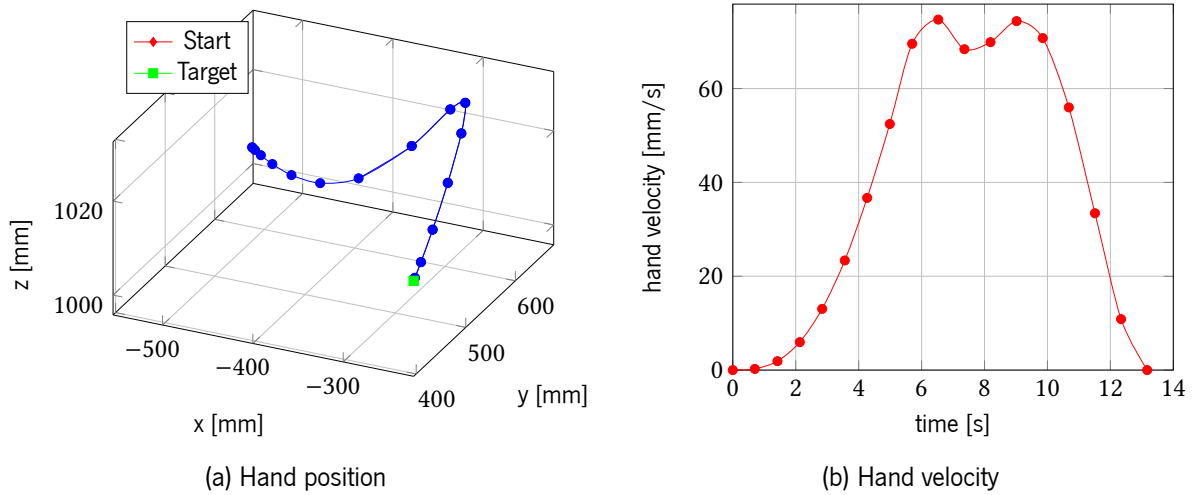


Figure 6.19: Retreat stage of the place movement 2 in Figure 6.15 plus the move movement in Figure 6.18 planned by the HUMP in the drink serving task (NMU=1).

24.57. Moreover, Table 6.10 shows that all the probabilistic planners, except the RRT*, scored a high rate of success suggesting a low complexity of the planning problem. This is also confirmed by the fact that the HUMP only required 297 ms to generate a trajectory, which marks it as the fastest planner in this particular case.

Figure 6.20 shows the hand path and hand velocity profile of the entire movement sequence defining the drink serving task. The start and the target pose of the task are the same because the robot starts and ends with the same posture (Figure 6.20a). The *Approach* phase is most prominent in the pick movement at the beginning of the task to guarantee a precise grasping behaviour. Furthermore, we note that the hand stops five times between the start and the end of the task, namely for the grasp transition, when the bottle has reached the pouring position, while pouring the juice into the glass, during the re-orientation of the bottle to avoid spilling of liquid and for the ungrasp transition.

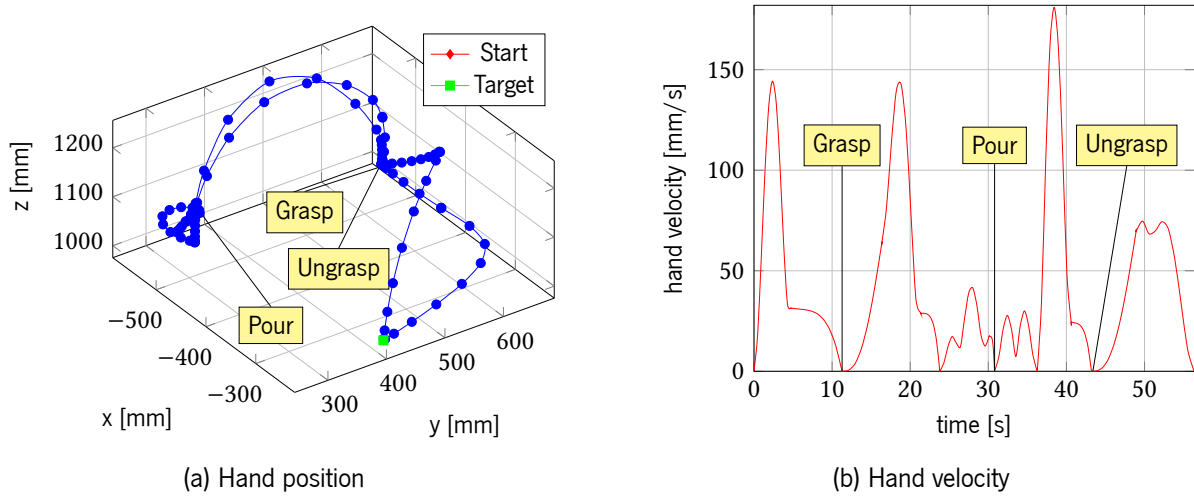


Figure 6.20: Complete movement sequence of the drink serving task planned by the HUMP.

6.3.3 The user study

The results of a short survey for a qualitative evaluation of human-likeness is reported in this section. The objective is to assess the perception of naïve users when watching trajectories with human-like characteristics for the scenarios of the toy vehicle assembly (section 6.3.1) and of serving a drink (section 6.3.2). In this user study, 50 participants (21 females and 29 males aged between 21 and 54 years) were asked to label the videos in Figures 6.6 and 6.12 as “human-like” or “not human-like” according to their perception. The participants watched the videos through web-based survey application to ensure well-controlled and repeatable conditions for all subjects. The videos were silenced to avoid any influence of robot motor noise on the personal judgement. Since the tasks were firstly explained, one can assume that each participant had expectations on how a human-like execution should look like. After the videos stopped playing, the participant had to choose between the labels “human-like” and “not human-like”. The user ratings showed that the HUMP generated human-like movements for the 78% and the 92% of the observers during the execution of the toy vehicle assembly task (Figure 6.6) and of serving a drink task (Figure 6.12), respectively.

6.3.4 Naturalistic obstacle avoidance

The toy vehicle assembly (section 6.3.1) and the serving a drink (section 6.3.2) tasks show that the HUMP generates smooth trajectories that avoid unwanted collisions with obstacles and the objects that are being manipulated. Since several studies of human obstacle avoidance have reported velocity profiles with a characteristic double-peak structure (Grimme et al., 2012; Rosenbaum et al., 2001), the claim that the HUMP generates kinematically human-like movement profiles requires further investigation.

To better understand how obstacle position and height might affect movement kinematics during obstacle avoidance, additional robotics experiments are conducted in a setup inspired by studies of human obstacle avoidance strategies in a three dimensional workspace (Grimme et al., 2012). The top view of the experimental setup shown in Figure 6.21 illustrates the spatial layout. The robot AROS reaches for a cylindrical object (a bottle) on a table in front of it and transports the object from the starting position to one of two possible target positions: Target 1 (T1) at a distance 30 cm away from the starting position in straight ahead direction, and Target 2 (T2) shifted 15 cm to the right of T1, thus located in a direction diagonal to the starting position. For each target, an obstacle was located at mid-distance between start and target either on the left or on the right of the straight line connecting start and target. The obstacle was a cylinder with a diameter of 8 cm, which could be either tall (28 cm in height) or small (14 cm in height). Table 6.11 summarizes the labels we use in the text to characterize the different conditions. The last condition in this table indicates if the obstacle avoidance mechanism is used. In other words, whether the *back-and-forth movement* (Equation 5.1.14) is superimposed on the *direct movement* (Equation 5.2.2) to obtain the trajectory in Equation 5.2.3 or only the *direct movement* is planned. Figure 6.22 illustrates two possible combinations in which only the height of the obstacle is changed.

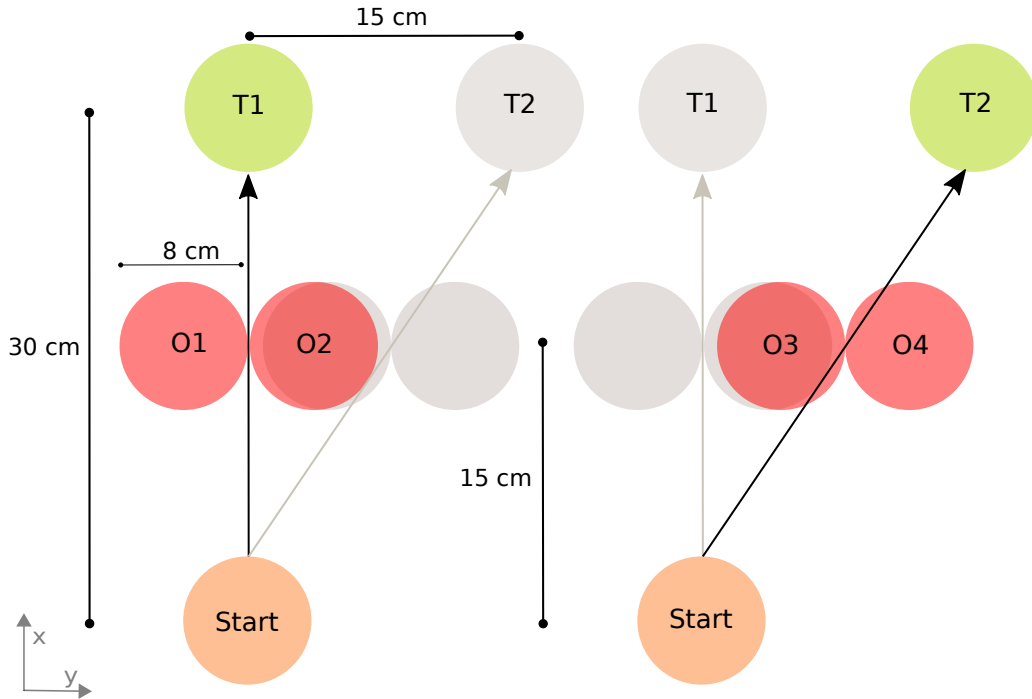


Figure 6.21: Spatial layout of the experimental setup. For details see the text. Obstacle locations O1 and O2 were only applied in combination with target T1, whereas obstacles locations O3 and O4 were combined with target T2.

For the sake of brevity, only the results of four experimental conditions and three videos are reported in

Target Position		Obstacle Position		Obstacle height		Obstacle Avoidance	
Straight	Diagonal	Left	Right	Small:14 cm	Tall:28 cm	Yes	No
T1	T2	O1 O3	O2 O4	S	T	Y	N

Table 6.11: Description of the symbols used in the experiments.

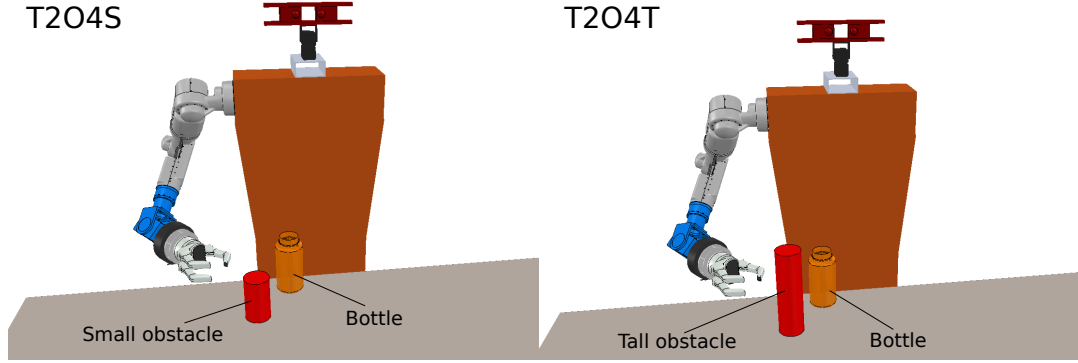


Figure 6.22: Configuration T204 with a small (T204S) and a tall obstacle (T204T).

this section, which nevertheless allow to qualitatively compare the results with fundamental findings of the study conducted in Grimme et al., 2012. Since the primary interest is in the avoidance of the obstacle, the analysis is focused on the *Retreat* phase of the pick movement and on the *Transport* and *Approach* phases of the subsequent place movement (section 5.2.1). Following the paradigm in Grimme et al., 2012, the torsion of the hand is measured as reported in Equation 3.3.4 to verify their finding that movement of obstacle avoidance in this paradigm are approximately planar. In Equation 3.3.4, the three-dimensional position of the hand along its path brings numerical problems when the velocity and the acceleration of the hand are close to zero. This is visible in the results at the beginning and at the end of a movement, but it does not relate to the conclusions. In addition, the tangential velocity is decomposed into a transport and a lift/descend component because this decomposition allows to understand the occurrence of a double peak structure in the tangential speed profile (Grimme et al., 2012). The transport component is obtained by projecting the path on the straight line connecting the start position to the target position, which corresponds to the Euclidean x-component (for configurations T2, the frame of Cartesian coordinates is rotated accordingly), and the lift/decent component is the orthogonal complement defined by Equation 6.3.1.

$$l(t) = \sqrt{y(t)^2 + z(t)^2} \quad (6.3.1)$$

The hand tangential velocity is then given by Equation 6.3.2.

$$\dot{h}(t) = \sqrt{\dot{l}(t)^2 + \dot{x}(t)^2} \quad (6.3.2)$$

Figures 6.23 and 6.24 show the results of the planning process in configuration T204TN, which is without obstacle avoidance and with a tall obstacle present at position O4. As expected, the hand collides with the mid-way obstacle. The hand path is planar during most of the time and the resulting velocity profile is defined by the dominant, bell-shaped transport component and a negligible lift component.

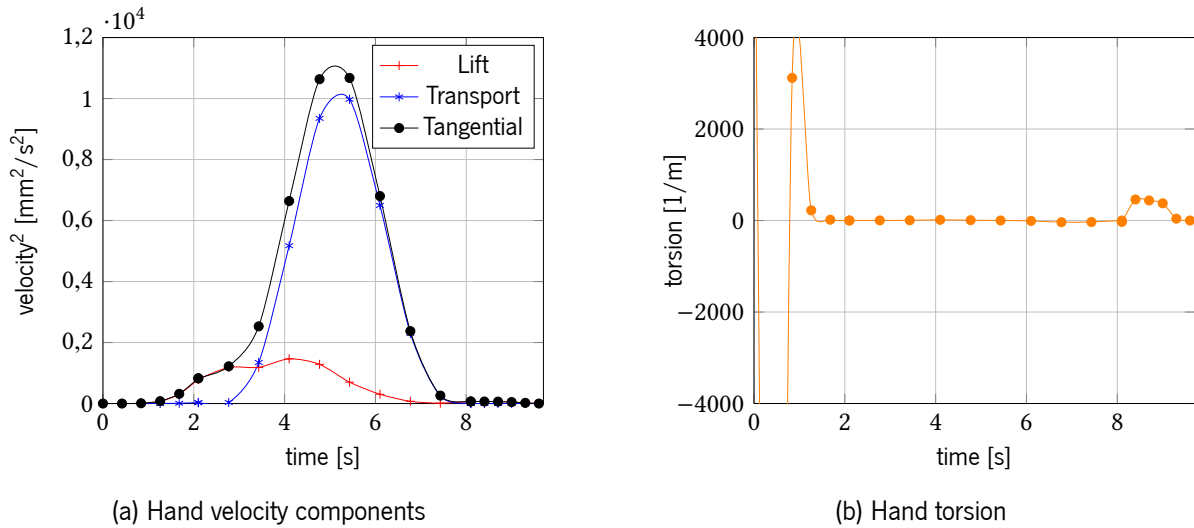


Figure 6.23: Hand velocity components and hand torsion of the manipulation in configuration T204TN.

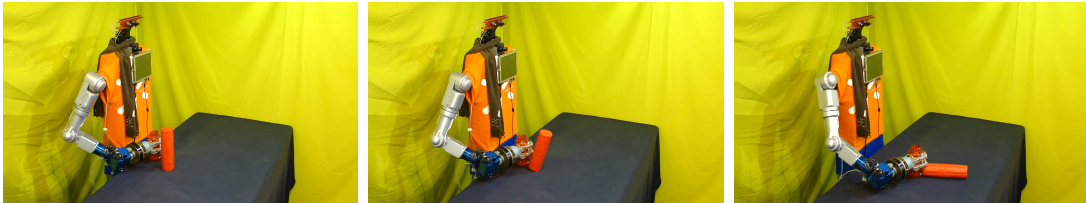
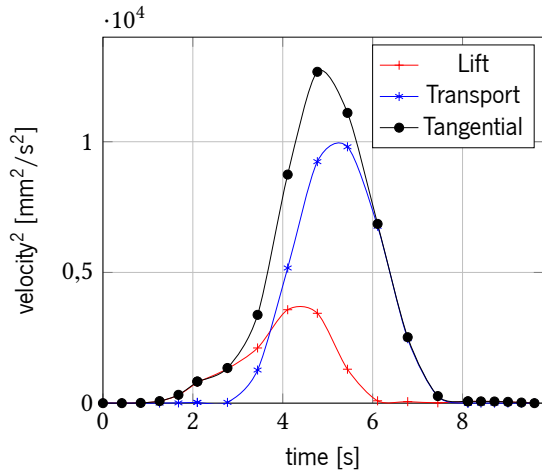


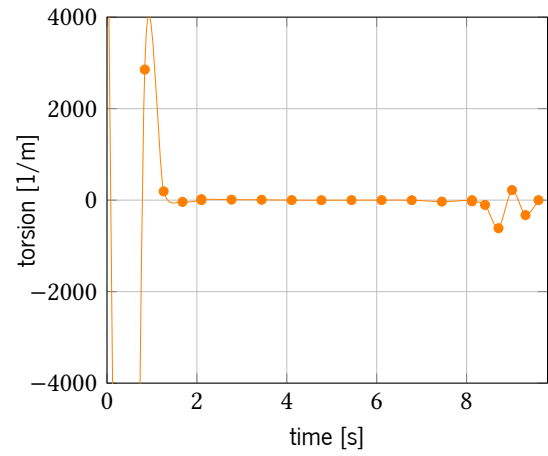
Figure 6.24: Three snapshots that show the movement planned by the HUMP in the configuration T204TN. The complete video can be seen at the link https://youtu.be/Wlh_QIO-0Xg.

An example of the same configuration with obstacle avoidance (T204TY) is shown in Figures 6.25 and 6.26. A significant lift/descend component is needed to avoid the tall cylinder that, however, has a single velocity peak. Since a movement path with relative large safety margin to the obstacle has been planned, this suggests that the tangential velocity profile is linked to the path geometry and not directly coupled to the obstacle passage.

Configuration T102TY with a tall obstacle, in contrast, produced a tangential velocity profile with a clear double peak (Figures 6.27 and 6.28). This comes from the much larger peaks in the lift/descend component compared to configuration T204TY (see Figures 6.25 and 6.26 for comparison). In contrast, in configuration T101SY, the small size of the cylinder did not require a significant lift/descend component

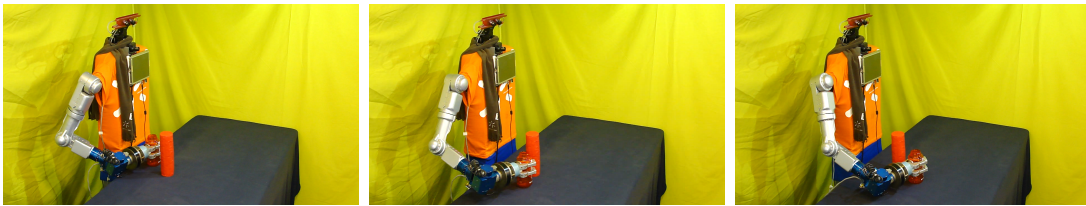
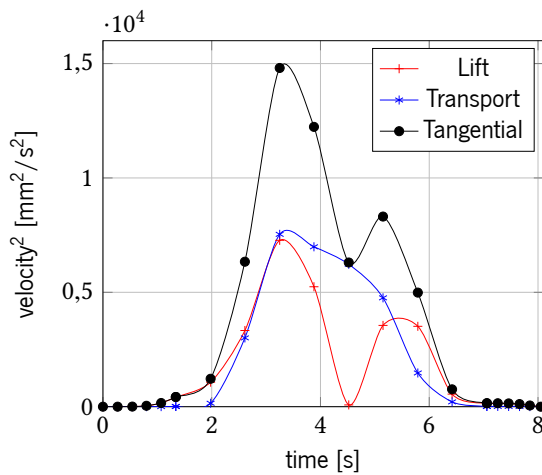


(a) Hand velocity components

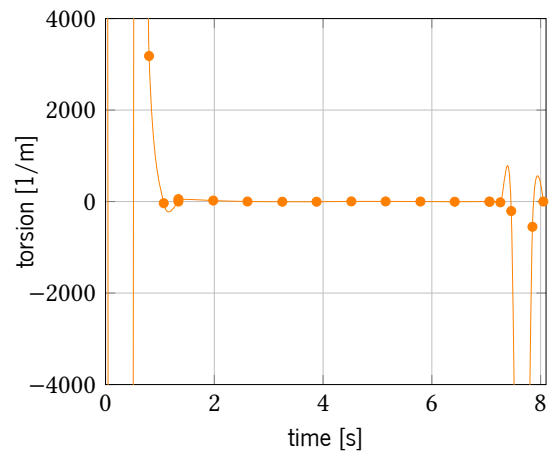


(b) Hand torsion

Figure 6.25: Hand velocity components and hand torsion of the manipulation in configuration T204TY.

Figure 6.26: Three snapshots that show the movement planned by the HUMP in the configuration T204TY. The complete video can be seen at the link https://youtu.be/Wlh_QI0-0Xg.

(a) Hand velocity components



(b) Hand torsion

Figure 6.27: Hand velocity components and hand torsion of the manipulation in configuration T102TY.

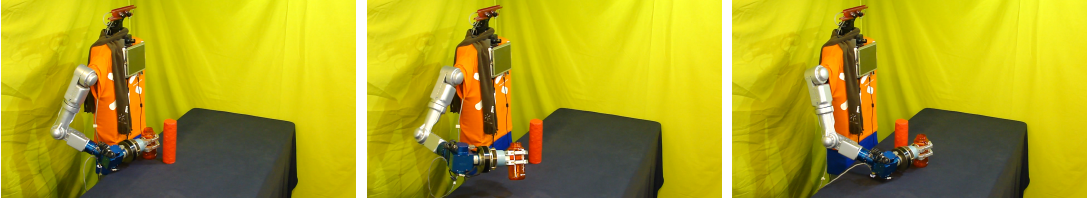


Figure 6.28: Three snapshots that show the movement planned by the HUMP in the configuration T102TY. The complete video can be seen at the link https://youtu.be/Wlh_QIO-0Xg.

(which, however, shows a double peak). The results of this configuration are interesting because even though there is a double-peak in the lift component, the latter has not a sufficient amplitude to raise a second peak in the total component (compare Figure 6.29a with Figure 6.27a). The resulting tangential velocity profile almost coincides with its transport component and the motion is essentially planar (Figure 6.29).

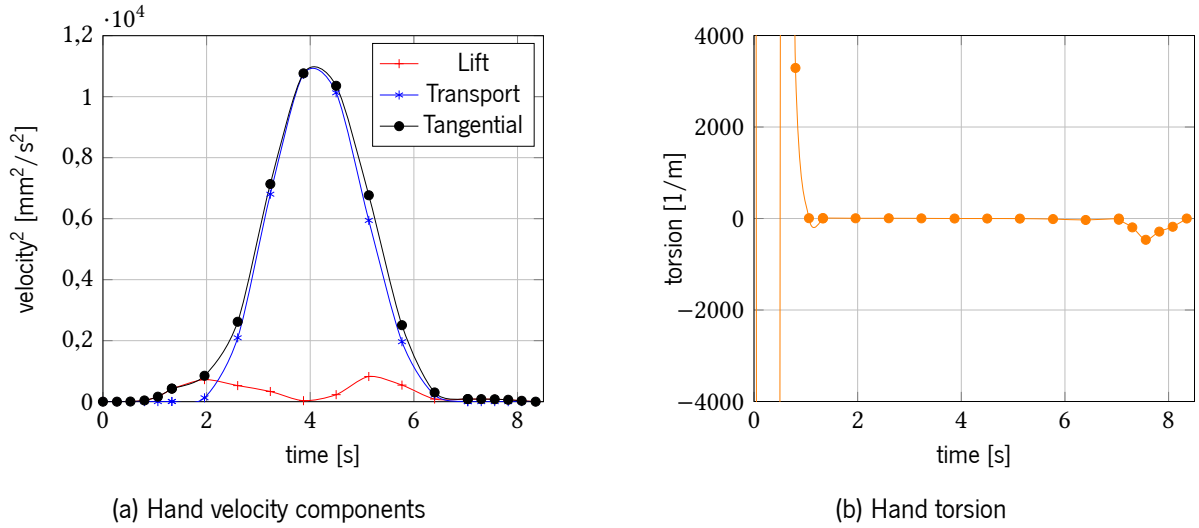


Figure 6.29: Hand velocity components and hand torsion of the manipulation in configuration T101SY.

The results of the obstacle avoidance experiments not only reproduce the observed planarity of the movements but are qualitatively in good agreement with other findings of Grimme et al. (2012) as well. The authors consistently observed a zero crossing of hand velocity during the transition from lift to descend. Moreover, the magnitude of the lift/descend component, which was often correlated with the obstacle height, defines whether the speed profile is uni- or bi-modal. However, these experiments with the robot ARoS also indicate that the size of the obstacle might not be the only factor affecting the hand velocity profile. Additional task demands represented on a higher cognitive planning level like accuracy and safety that influence the choice of, for instance, the clearance distances (see the constraints (5.1.6) and (5.1.19)) or the number of steps used for time discretization (Equation 5.3.5), might affect the hand kinematics as

well.

6.3.5 Hand straight-line motion planning with ARoS

The algorithms for generating straight-line hand paths during approaching on targets and retreating from them were introduced in section 5.5. Tests on the capability to plan rectilinear hand paths by applying the HUMP in complex scenarios are here presented and discussed. In specific, a pick-and-place task from a narrow shelf is the scenario illustrated in Figure 6.30, which has been selected to demonstrate the ability to accomplish goal-directed actions in highly cluttered environments. For demonstration purposes,

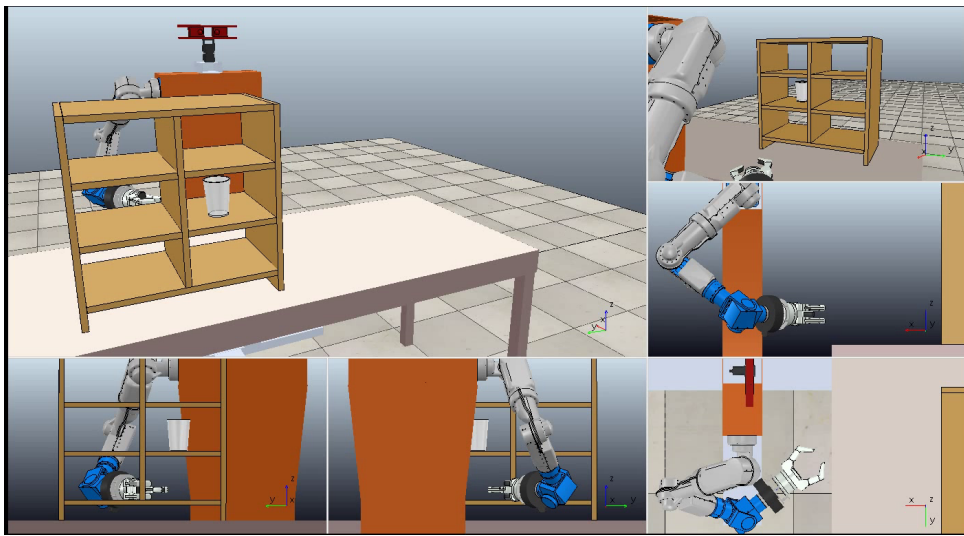


Figure 6.30: Pick-and-place from a narrow shelf scenario.

a glass is placed on the middle part of the shelf and has to be grasped and moved to the lower part while collisions must be avoided with the table, the shelf, the body of the robot and the glass itself for a successful pick and place (Figure 6.31). The shelf is placed on the table in front of the robot ARoS with a glass at clear sight but closely surrounded by rectangularly shaped obstacles (Figure 6.31a). In such a situation, where manipulation must occur in a densely cluttered environment, forcing the end-effector to move on a straight-line path is a simple workaround to prevent the explicit definition of collisions-avoidance constraints, which can inefficiently overload the computational burden of a planning algorithm. Then, the glass is picked by tracking hand straight lines during the *Approach* and *Retreat* phases of the generated pick movement (Figure 6.31b). Similarly, the glass is transported and placed on the lower level of the shelf following hand straight lines during the *Approach* and *Retreat* phases of the generated place movement (Figure 6.31c). At the end, the manipulator departs from the shelf and returns to its start configuration by a move movement (Figure 6.31d). The kinematic characteristics of the planned pick movement are showed in Figure 6.32. This movement is featured by a single-peaked bell-shaped hand velocity profile followed

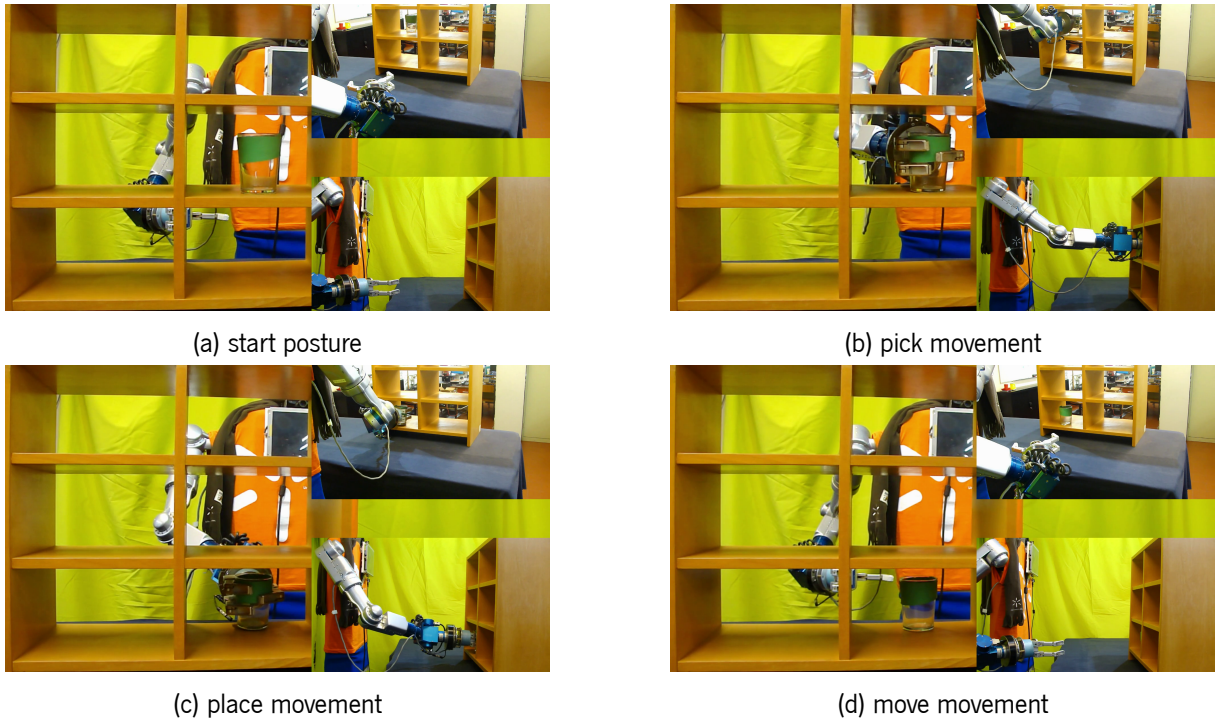


Figure 6.31: Four snapshots showing, with three different angles, the sequence of the movements planned by the HUMP in the task of picking and placing from a narrow shelf. The complete video can be seen at the link <https://youtu.be/1dwj-HP0USI>.

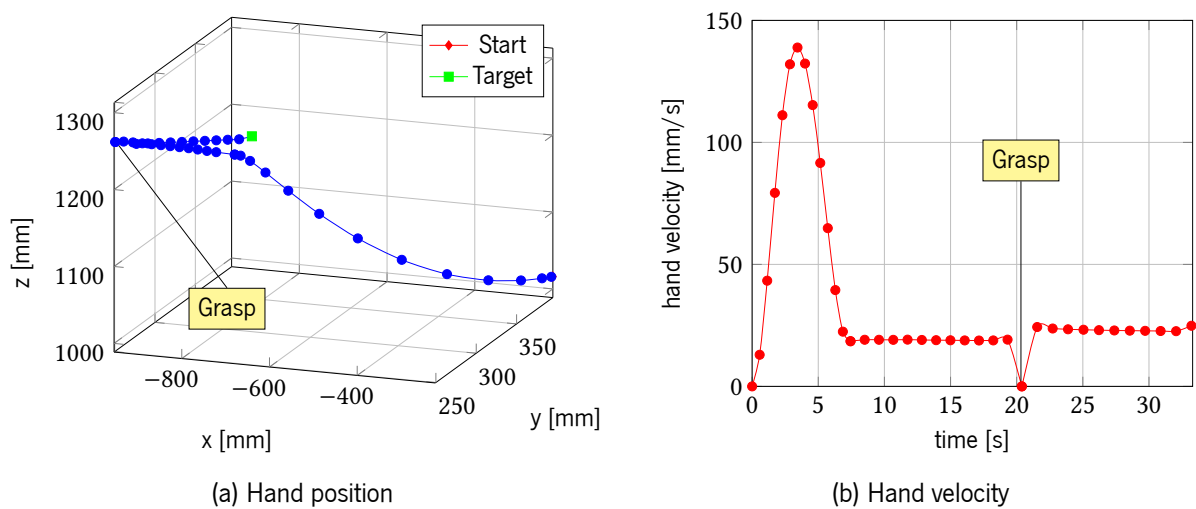


Figure 6.32: Pick movement planned by the HUMP (NJS=25732.52, NMU=1, solving time = 8726 ms).

by *Approach* and *Retreat* phases with quasi-constant speeds (Figure 6.32b). Additionally, the phases in the vicinity of the glass present rectilinear hand paths from the start to the target hand poses with main variations on the $\hat{x}$ and on the $\hat{y}$ axes and equal hand orientations to guarantee a collisions-free passage and a successful picking (Figure 6.32a). Importantly, the end-effector only stops at the moment of grasping because the trajectory is composed with respect to the boundary conditions of its elementary motion phases. The planned pick movement results with a single NMU, a NJS of 25732.52 and a solving time of 8726 ms, which signals the difficulties of generating a joints-space trajectory in the vicinity of a densely cluttered area of the workspace. This also emerges from the low rate of success that is encountered with the application of different sampling-based method in solving the same planning problem (Table 6.12). In

Planner	Solving time: mean(SD) [ms]	Rate of success [%]
RRT	2665.00(1532.92)	7
RRTConnect	1704.25(1196.01)	56
RRT*	4174.25(1699.30)	4
PRM	734.28(425.68)	58
PRM*	4195.65(1865.95)	63

Table 6.12: Results over 100 trials of the pick movement planning problem in the pick-and-place from a narrow shelf scenario (Figure 6.31).

specific, the selected sampling-based planners were also set to approach and retreat without the consideration of obstacles and to move on a series of way-points along a rectilinear hand path. Table 6.12 shows that the PRM and PRM* algorithms provided relatively higher rates of success and that the probabilistic planners performed averagely better than the HUMP.

The hand position and velocity of the planned place movement are illustrated in Figure 6.33. Similarly to the antecedent pick movement, there is single-peaked bell-shaped velocity profile that is joint with quasi-constant approaching and retreating speeds (Figure 6.33b). Figure 6.33a shows again hand rectilinear paths during the *Approach* and the *Retreat* phases that guarantees no collisions between the manipulator and the shelf. The movement start with non-null boundary conditions taken from the preceding pick movement and the hand only stops when the glass is un-grasped at its placing location (Figure 6.33b). The place movement presents a NMU of one, a NJS of 14337.64 and the problem was solved in 3478 ms. Table 6.13 shows again a small rate of success for the sampling-based planners applied to solve the placing problem, except for the RRTConnect algorithm that found a solution 90 times over 100 trials. With exception for the PRM*, the selected probabilistic planners averagely performed better than the HUMP. Generally speaking, Table 6.13 confirms again a certain complexity of the planning problem in generating collisions-free trajectories in the vicinity of the shelf.

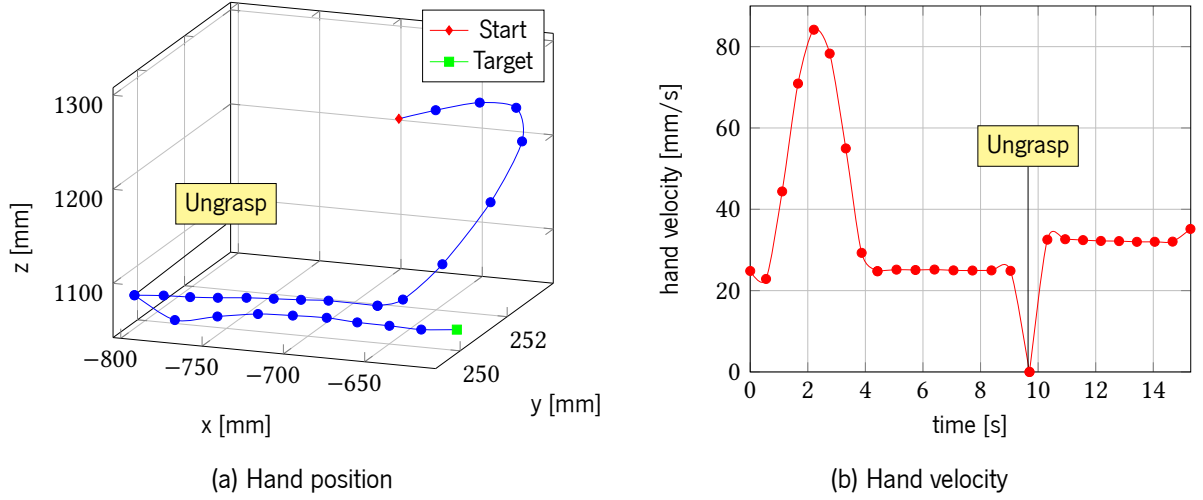


Figure 6.33: Place movement planned by the HUMP (NJS=14337.64, NMU=1, solving time = 3478 ms).

Planner	Solving time: mean(SD) [ms]	Rate of success [%]
RRT	1636.21(1258.81)	28
RRTConnect	2008.77(1090.42)	90
RRT*	5153.45(56.52)	22
PRM	2046.23(2082.94)	44
PRM*	4695.61(1604.37)	41

Table 6.13: Results over 100 trials of the place movement planning problem in the pick-and-place from a narrow shelf scenario (Figure 6.31).

The last movement of the task is a move movement that brings the robotic upper-limb back to its start configuration with the kinematics illustrated in Figure 6.34. The movement starts with non-null boundary conditions that coincide the final kinematic values of the antecedent place movement. Figure 6.34b also illustrates a human-like bell-shaped hand velocity profile, which presents a very modest second peak that emerges from a significant change in the hand orientation at the beginning of the motion. Indeed, the NMU remains 1, the NJS of 35.31 indicates a smooth human-like trajectory and a solving time of 4813 ms shows once again the significant influence of an elevated number of constraints on the optimization-based planning procedures of the HUMP. The high rates of success reported in Table 6.14 show that this is a relatively simple planning problem for the selected sampling-based algorithms. In the generation of this move movement, the HUMP was averagely faster than RRT, RRT* and PRM*, while it was averagely slower than RRTConnect and PRM.

The concatenation of the movement that compose the pick-and-place task from the narrow shelf is illustrated in Figure 6.35. The hand starts and finish its motion at the same point of the workspace and it is featured by straight-lines when approaching and retreating picking and placing targets (Figure 6.35a).

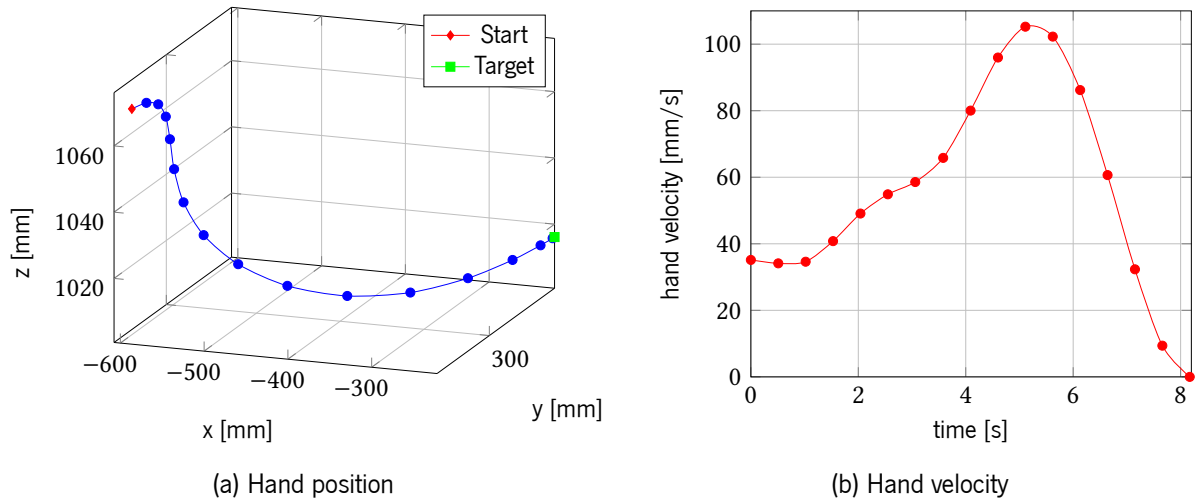


Figure 6.34: Move movement planned by the HUMP (NJS=35.31, NMU=1, solving time = 4813 ms).

Planner	Solving time: mean(SD) [ms]	Rate of success [%]
RRT	5179.86(268.11)	100
RRTConnect	3156.00(924.10)	100
RRT*	5331.74(104.23)	88
PRM	1197.50(323.94)	100
PRM*	5855.53(283.39)	91

Table 6.14: Results over 100 trials of the move movement planning problem in the pick-and-place from a narrow shelf scenario (Figure 6.31).

Figure 6.35b clearly shows three bell-shaped peaks and that the hand only stops for grasping on one level of the shelf and for un-grasping on its lower level. Interestingly, the *Approach* and the *Retreat* phases of the pick and of the place movements present quasi-constant velocities along the rectilinear paths of the end-effector with small increments on the boundaries. This is due to the fixed time step that there is among the way-points of a selected straight-line hand path, which cause little bumps on joints-space velocities when a target posture has to be reached at the end of approaching and retreating (see the algorithms section 5.5 for details). Given a sufficient number of steps along the rectilinear path, these effects do not negatively impact on the composition of the motion in the vicinity of a target and on the overall resulting trajectory.

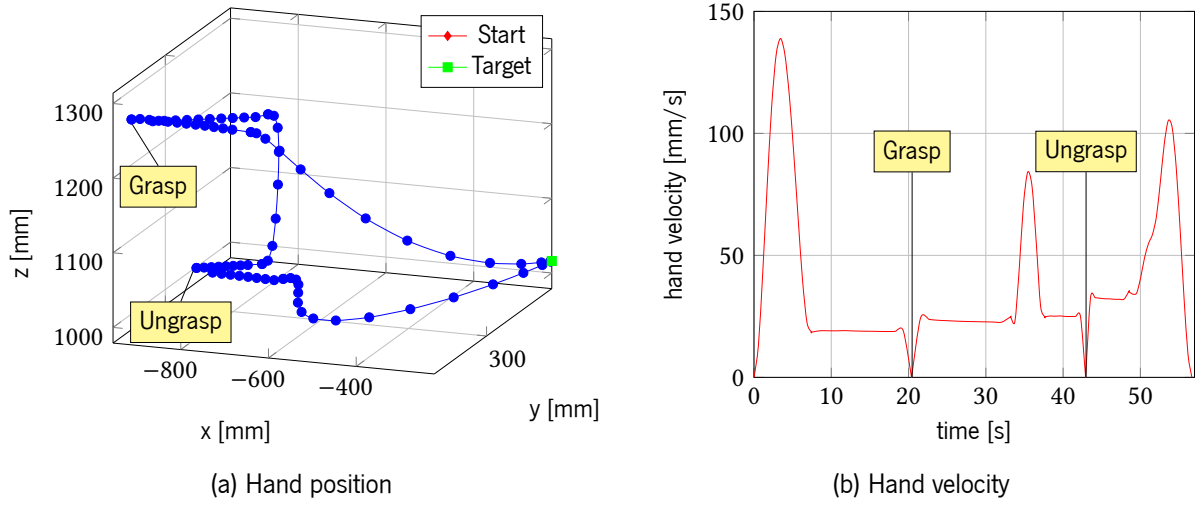


Figure 6.35: The complete task planned by the HUMF.

6.3.6 Human-like synchronous bimanual motion planning with ARoS

In this section, the mathematical formulation for the generation of synchronous bi-manual movements (section 5.6) is tested with the robot ARoS in two different scenarios. For the sake of brevity, the comparisons in solving times with sets of sampling-based planners are here skipped. Reasonable conclusions can be drawn from the results reported in sections 6.3.1, 6.3.2 and 6.3.5 concerning the generation of single-arm movements. section 6.3.6.1 reports the tests on inter-limbs collisions avoidance in a construction paradigm inspired from the toy vehicle assembly scenario (section 6.3.1), while section 6.3.6.2 shows an application for the simultaneous transportation of one object with two hands in a serving paradigm inspired from the scenario introduced in section 6.3.2. Both of these sections present results on simulated environments, while preliminary experimentations with the real robotic platform are discussed in section 6.3.6.3.

6.3.6.1 Toy vehicle assembly: swapping two columns simultaneously

The collaboration with a human partner during the assemblage of a toy vehicle (section 6.3.1) may result in undesired situations like that one illustrated in Figure 6.36. For some reasons, the magenta and the green columns are inserted into the base at wrong places. Specifically, the green column is plugged into the magenta location, while the magenta column is in the green spot. A humanoid robot with two arms can simultaneously swap the columns and insert them into the right holes of the base. The sequence of the synchronous bi-manual movements planned for the robot ARoS are showed in Figure 6.37. The scenario presents the green and the magenta columns inserted into the base of the toy vehicle in front of the robot ARoS (Figure 6.37a). The task starts with picking the columns by crossing the right and the left robotic

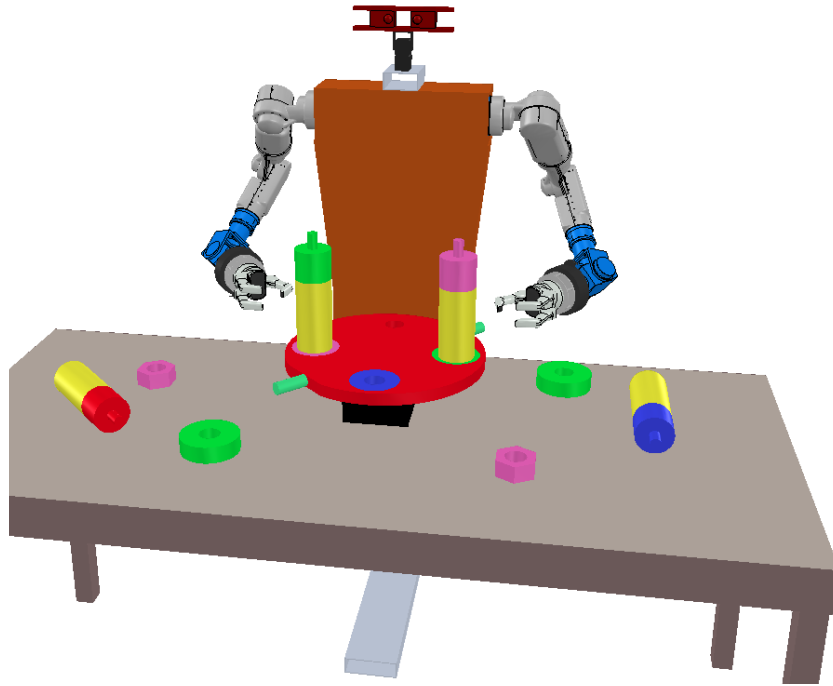
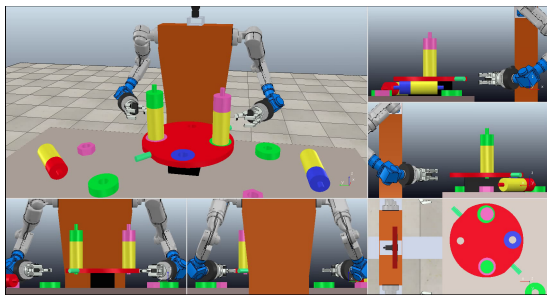
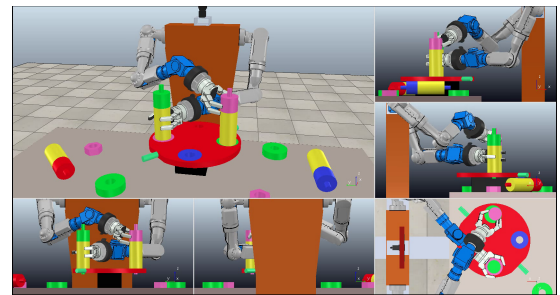


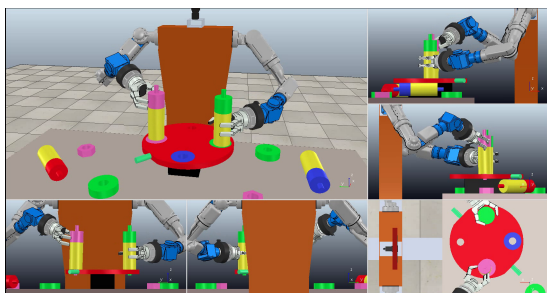
Figure 6.36: Toy vehicle assembly: scenario for swapping two columns.



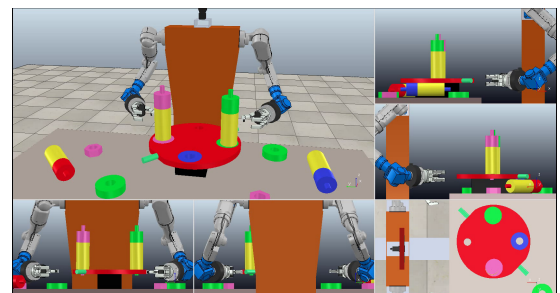
(a) start posture



(b) pick movement



(c) place movement



(d) move movement

Figure 6.37: Four snapshots that show the sequence of the synchronous bi-manual movements planned by the HUMP in the toy vehicle assembly scenario. The complete video can be seen at the link <https://youtu.be/Sj-nXmMg6U4>.

manipulators without collisions (Figure 6.37b). Then, the columns are swapped and placed in the correct locations (Figure 6.37c). In conclusion, the robot can return to its initial configuration (Figure 6.37d). The synchronous bi-manual movements are planned as described in section 5.6 and the pick and place motion is featured with *Approach* and *Retreat* phases.

The kinematic characteristics of the pick movement planned by the HUMP are illustrated in Figure 6.38. Qualitatively, both hands show collisions-free paths and uni-modal bell-shaped velocity profiles with signif-

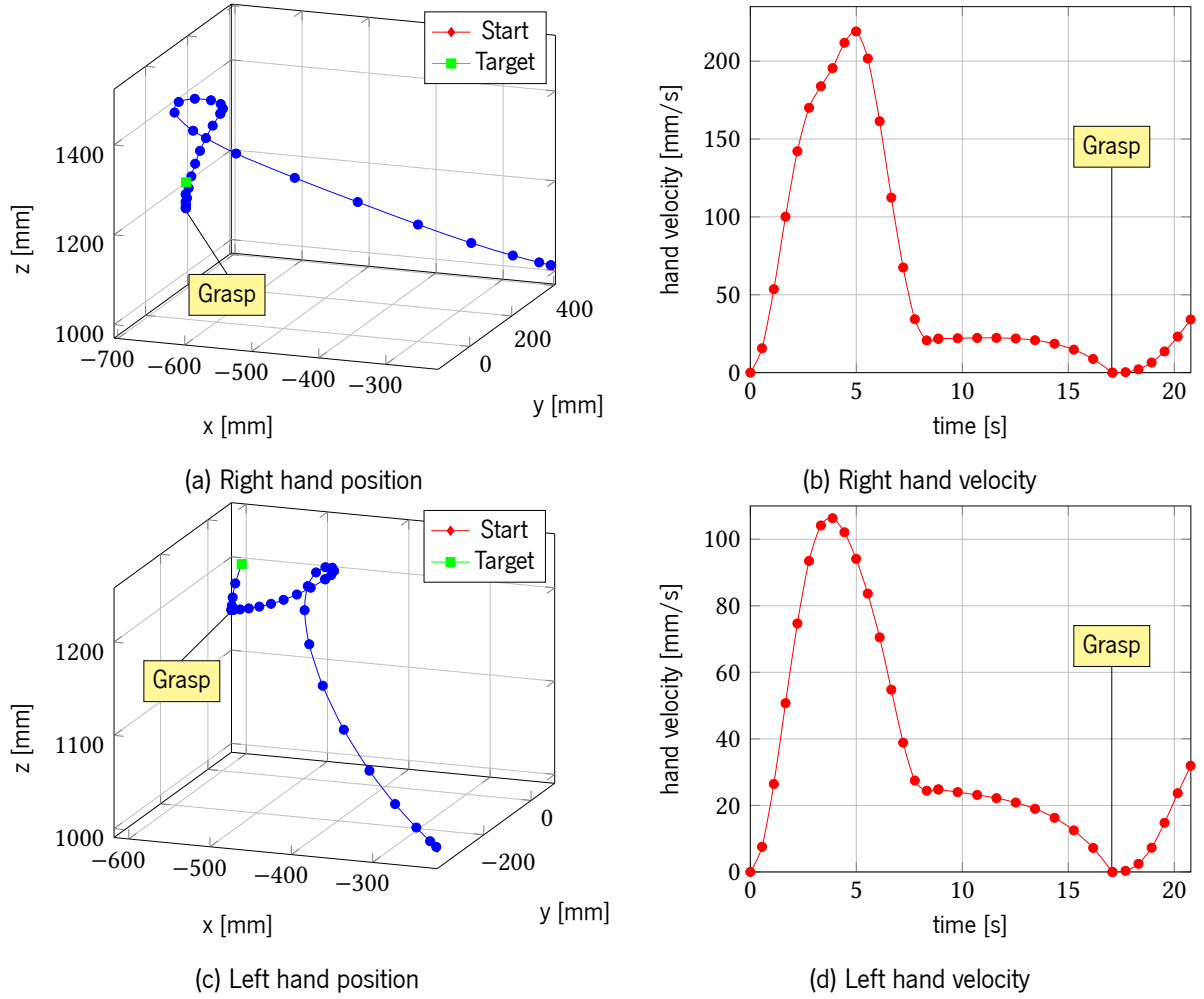


Figure 6.38: Synchronous bi-manual pick movement planned by the HUMP (right NJS=3798.96, left NJS=3952.60, Right NMU=1, Left NMU=1, solving time = 18269 ms).

icant *Approach* phases to ensure a certain degree of accuracy (Figures 6.38b and 6.38d). In particular, both trajectories are featured with a single NMU and similar normalized jerk scores (right NJS=3798.96, left NJS=3952.60). It can be noted that the insertion of left-arm related constraints provoked a considerable increase of the solving time (18269 ms).

The planned synchronous bi-manual place movement is also characterized by single-peaked bell-shaped

hand velocity profiles with accurate Approach and Retreat phases. It starts with non-null boundary conditions that terminate the antecedent pick movement for a correct kinematic composition (Figures 6.39b and 6.39d). Both hand trajectories have single normalized movement units, while the left hand path

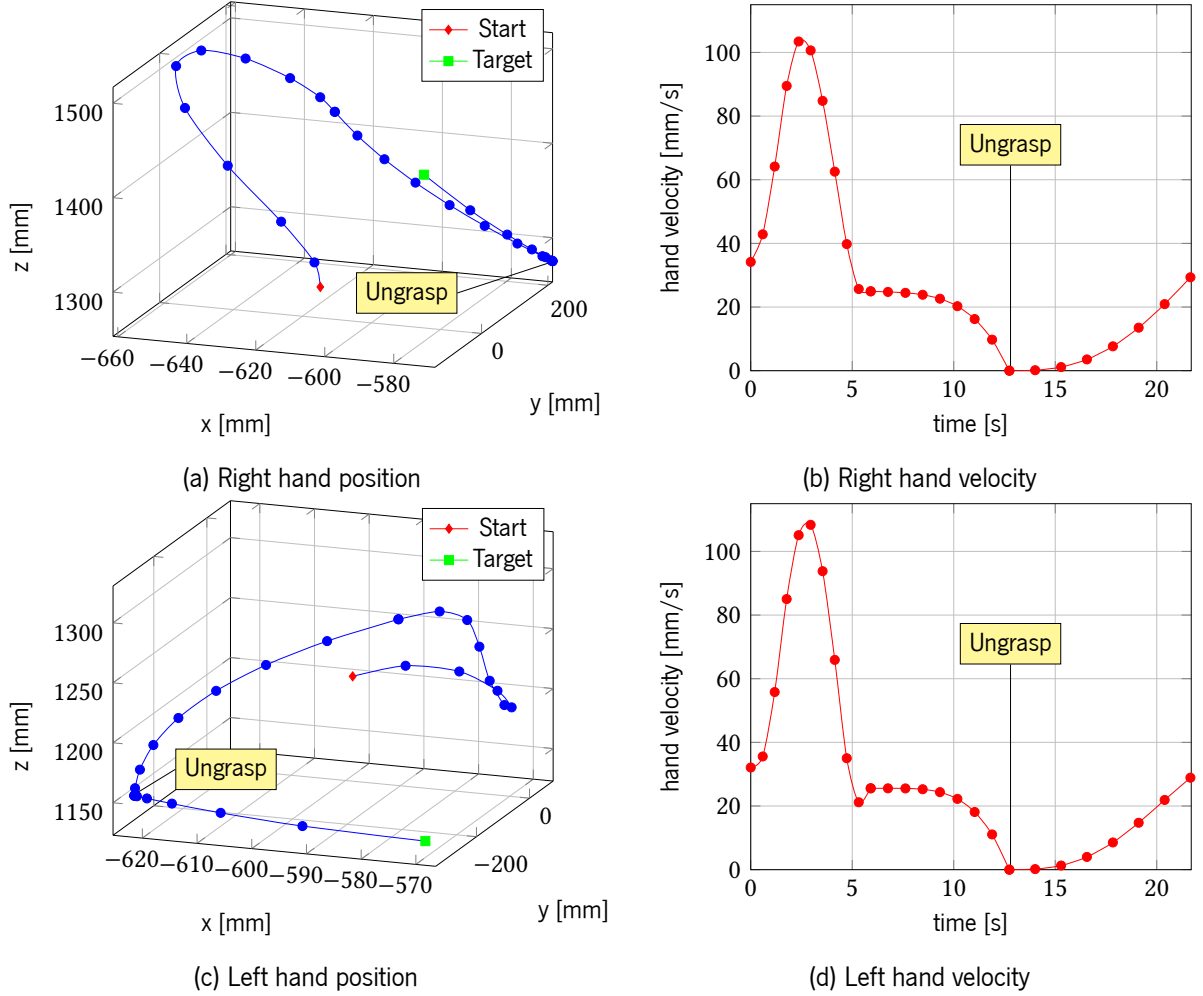


Figure 6.39: Synchronous bi-manual place movement planned by the HUMP (right NJS=9696.65, left NJS=7217.45, Right NMU=1, Left NMU=1, solving time = 16310 ms).

presents a smaller NJS when compared to the right hand path of the place movement. A more pronounced regularity of the left hand path can also be qualitatively detected by observing the positions of the hands in Figures 6.39a and 6.39c. Similarly to the antecedent planning problem, the HUMP provided a collisions-free placing solution in 16310 ms.

Finally, the task is concluded with a synchronous bi-manual move movement that brings the robot ARoS to its initial configuration (Figure 6.40). In specific, the manipulators move with very smooth hand paths (Figures 6.40a and 6.40c) and bell-shaped hand velocity profiles (Figures 6.40b and 6.40d). Due to the significant change in the orientation of the right hand, the left hand trajectory results smoother (left NJS=20.30)

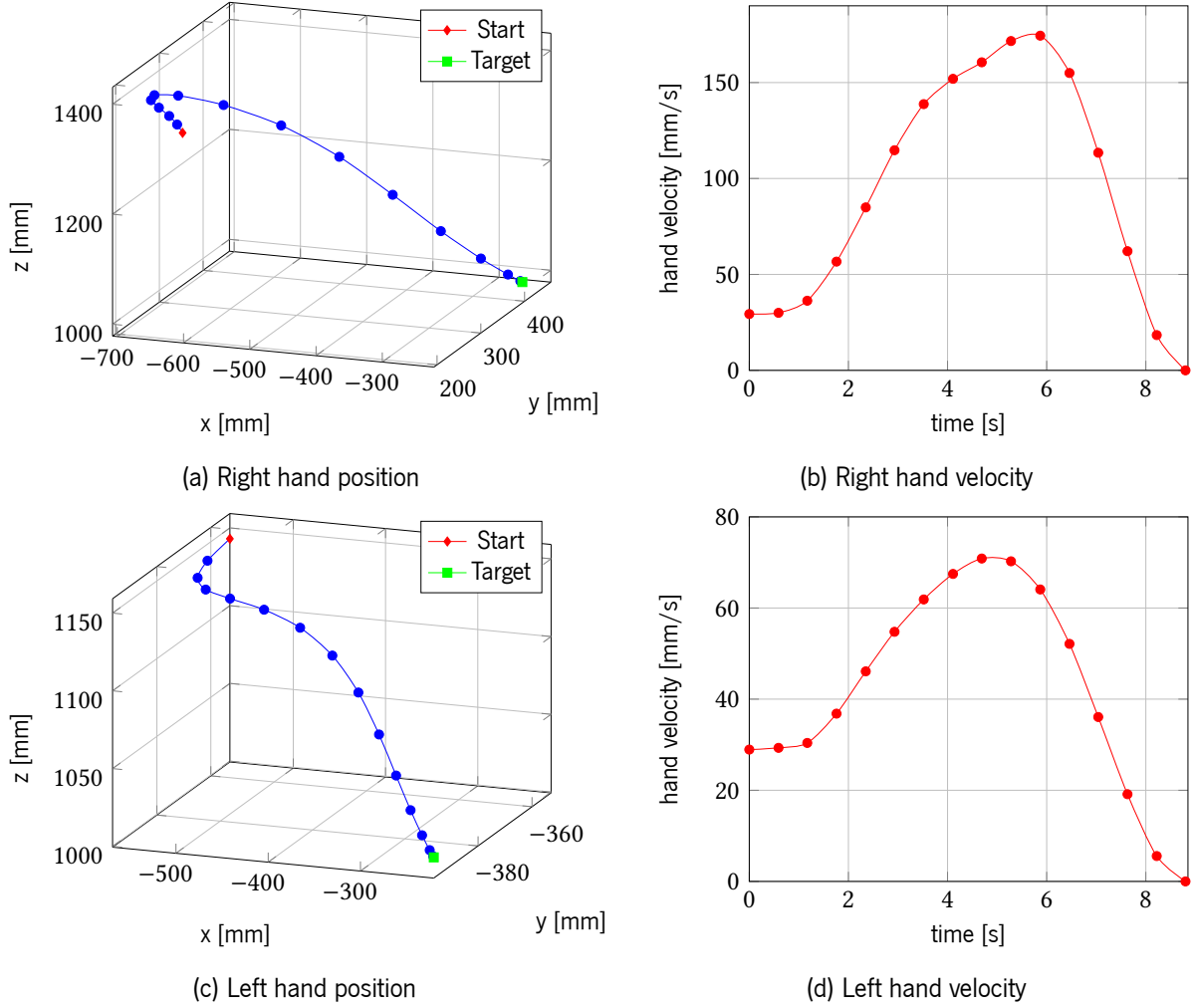


Figure 6.40: Synchronous bi-manual move movement planned by the HUMP (right NJS=65.05, left NJS=20.30, Right NMU=1, Left NMU=1, solving time = 9899 ms).

than the right one (right NJS=65.05). Both hand paths are featured by a single NMU, although a second velocity peak slightly rises on the profile of the right hand trajectory (Figure 6.40b). The solution of this move movement was provided by the HUMP in 9899 ms.

The composition of the three movements is illustrated in Figure 6.41. It can be noticed that both hands move synchronously and the robot only stops to grasp the columns and to un-grasp them for insertion in their correct places. Running this experiment heavily tested the contribution of the constraints for collisions-avoidance between the robotic upper-limbs and showed how two objects can be synchronously manipulated by both hands in a safe and human-like manner. Differently, in section 6.3.6.2, synchronous manipulation of one object with both hands is tested.

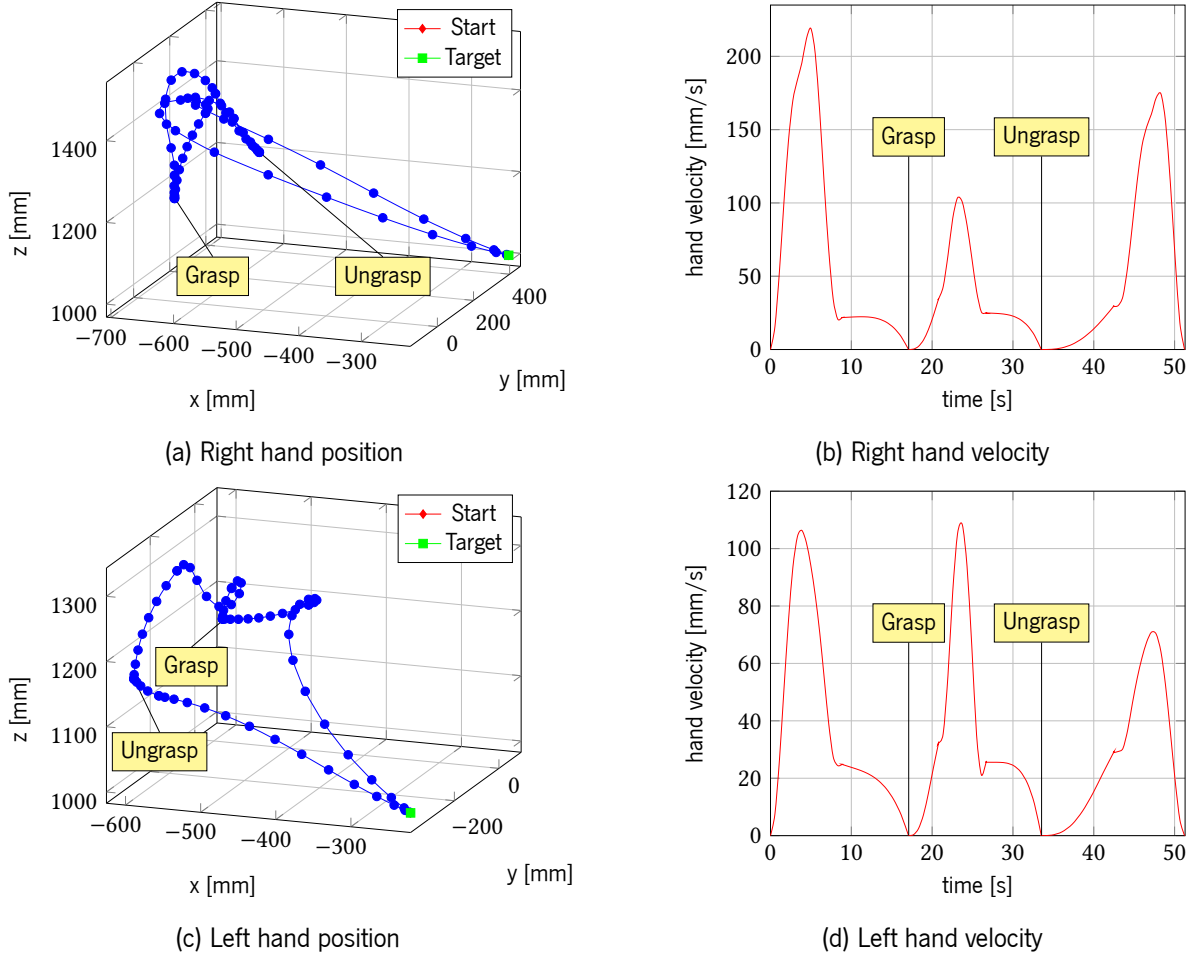


Figure 6.41: The complete synchronous bi-manual task planned by the HUMP.

6.3.6.2 Serving a drink: manipulating a tray

Serving a drink to human interacting partners (section 6.3.2) might also include typical serving subtasks that require a tight inter-limb coordination, such as transporting a tray with drinks. This set of experiments with the robot ARoS takes place in the scenario illustrated in Figure 6.42 and tests how the HUMP responds to a further augmentation of constraints (see details in section 5.6). A tray with two black handles is in front of the robot that has to pick it up and place it on top of the black box on its left side. During the motion, collisions have to be particularly avoided with the black box and the near brown bottle. Additionally, the planned movements have to keep the tray on its initial upright orientation under certain tolerances in order to prevent undesired drops of juice. The sequence of the movements for the synchronous bi-manual manipulation of the tray are showed in Figure 6.43. At the beginning, a tray with a glass of orange juice is in front of ARoS, which is in its initial configuration (Figure 6.43a). The tray is picked on its handles by both hands and lifted while its orientation is maintained (Figure 6.43b). The original upright orientation of the

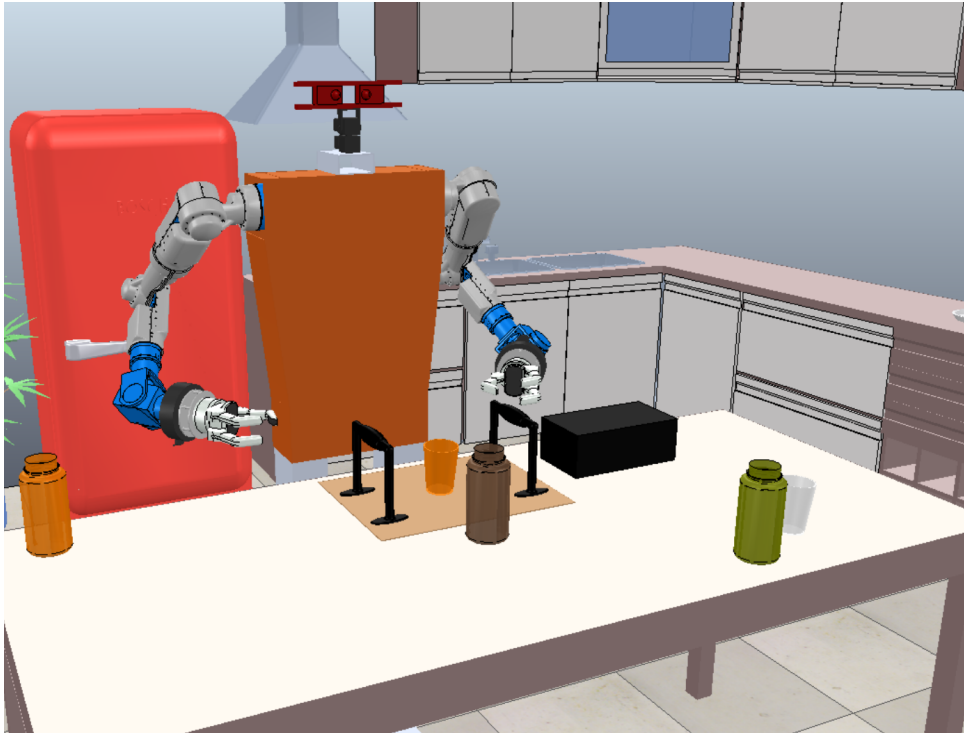
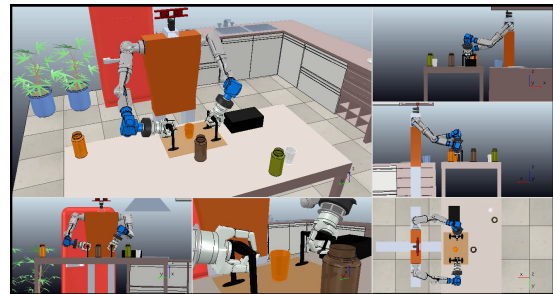


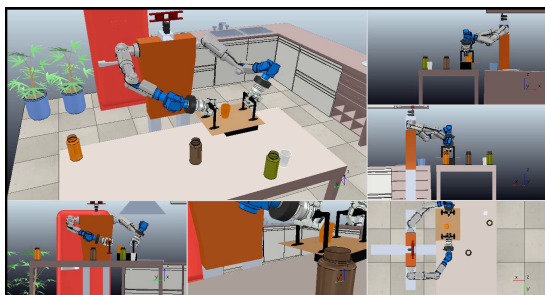
Figure 6.42: Serving a drink: moving a tray with two hands.



(a) start posture



(b) pick movement



(c) place movement



(d) move movement

Figure 6.43: Four snapshots that show the sequence of the synchronous bi-manual movements planned by the HUMP in the serving a drink scenario. The complete video can be seen at the link <https://youtu.be/Wf19IqvYsCA>.

tray needs also to be kept when the tray is transported and placed onto a black box on the left side of the robot (Figure 6.43c). Finally, the robot ARoS returns to the start posture of its manipulators (Figure 6.43d). The kinematic features of the synchronous bi-manual pick movement are illustrated in Figure 6.44. The right hand trajectory presents a smoother path than the left one with a single-peaked bell-shaped profile and a prolonged *Retreat* phase, which causes the rise of peak in velocity to stabilize the position of the tray in a less cluttered area of the workspace (Figures 6.44a and 6.44b). Due to significant changes in

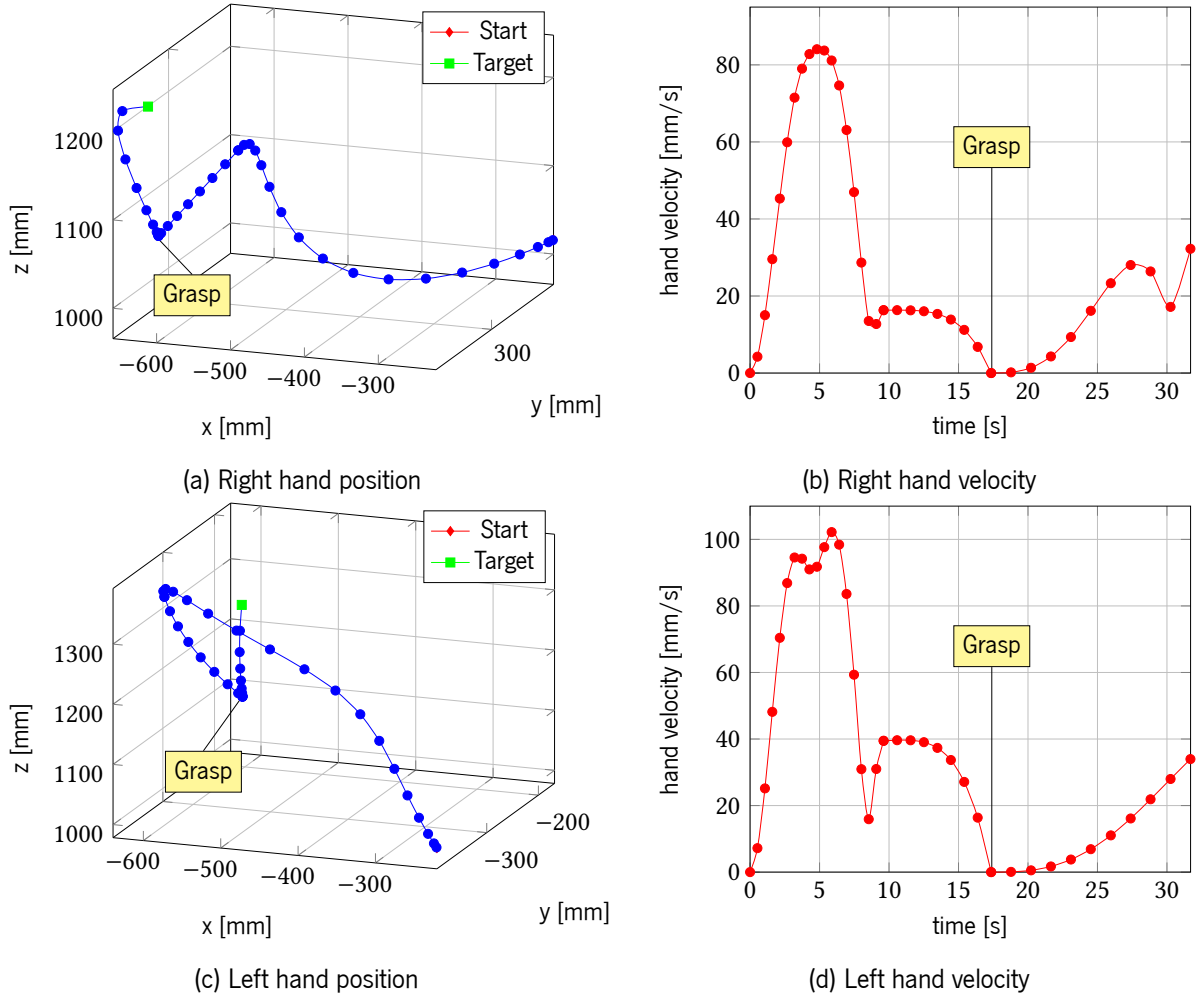


Figure 6.44: Synchronous bi-manual pick movement planned by the HUMP (Right NJS=18974.93, Left NJS=22075.82, Right NMU=1, Left NMU=1, solving time = 14561 ms).

the orientation of the end effector, the left hand trajectory shows two close peaks in its velocity profile, but the second peak is not sufficiently strong to raise the number of movement units to 2 (Figures 6.44c and 6.44d). The action only stops at the moment of grasping at the end of the *Approach* phases and the planning problem was solved in 14561 ms.

During placing, the path of the left hand results slightly smoother than the right one, but both are qualitatively similar with a single-peaked bell-shaped hand velocity profiles (Figures 6.45b and 6.45d). This

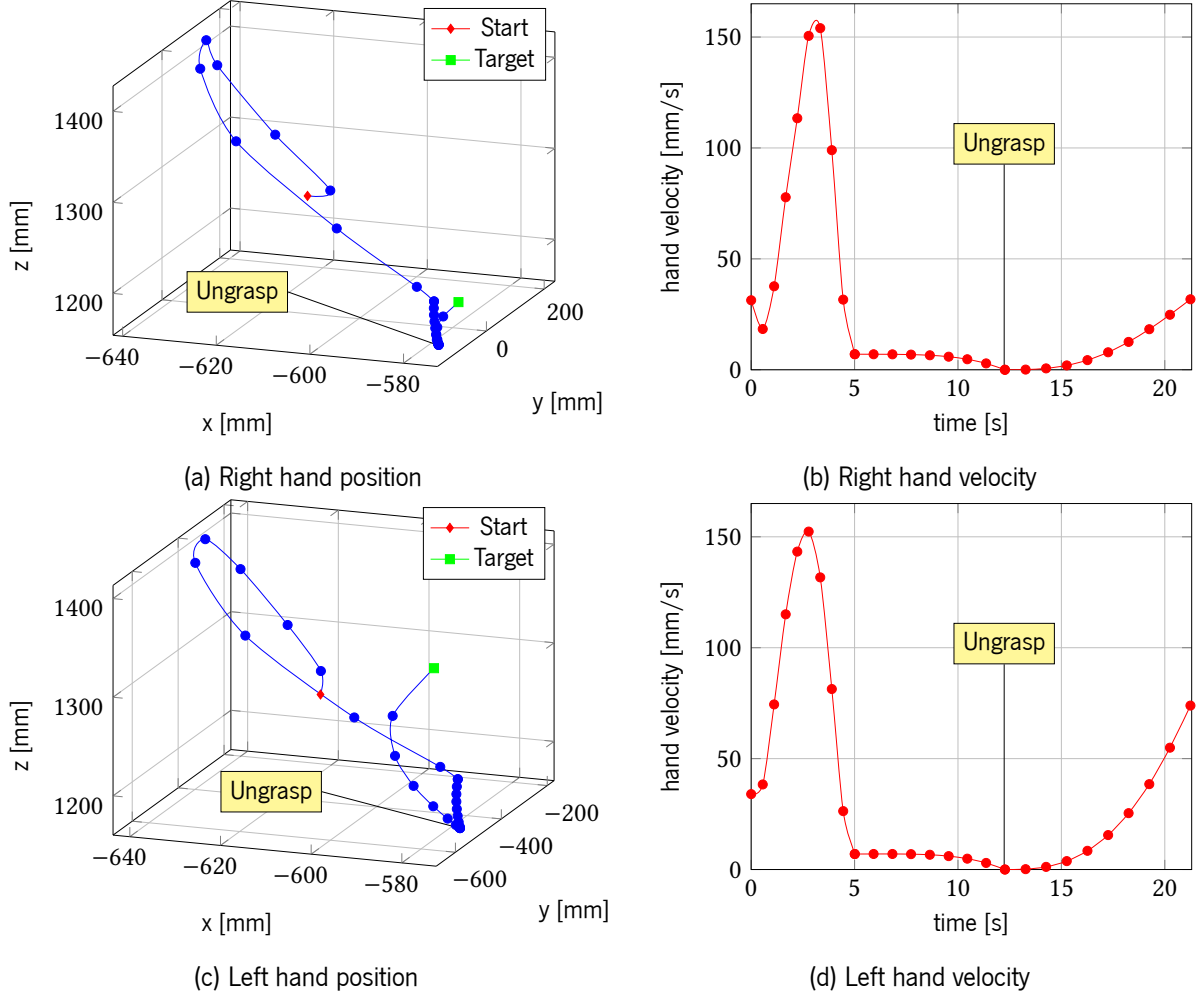


Figure 6.45: Synchronous bi-manual place movement planned by the HUMP (Right NJS=18933.08, Left NJS=14695.44, Right NMU=1, Left NMU=1, solving time = 6312 ms).

bi-manual place movement starts with non-null boundary conditions taken from the antecedent pick movement and finishes with significantly extended *Retreat* phases that safely transport the robotic hands away from the handles of the tray. Relaxing the tolerances on the orientation of the tray being transported and on the inter-hands distance, the placing problem could be also solved successfully and relatively faster than the picking one with a solving time of 6312 ms.

Figure 6.46 illustrates the kinematics of the synchronous bi-manual move movement. With a normalized jerk score of 157.53 and 2 number of movement units, the left hand trajectory presents a double-peaked velocity profile that is provoked by the significant change in the orientation of the end-effector (Figure 6.46d). Two peaks also feature the velocity profile of the right hand path, but the amplitude of the first peak is not

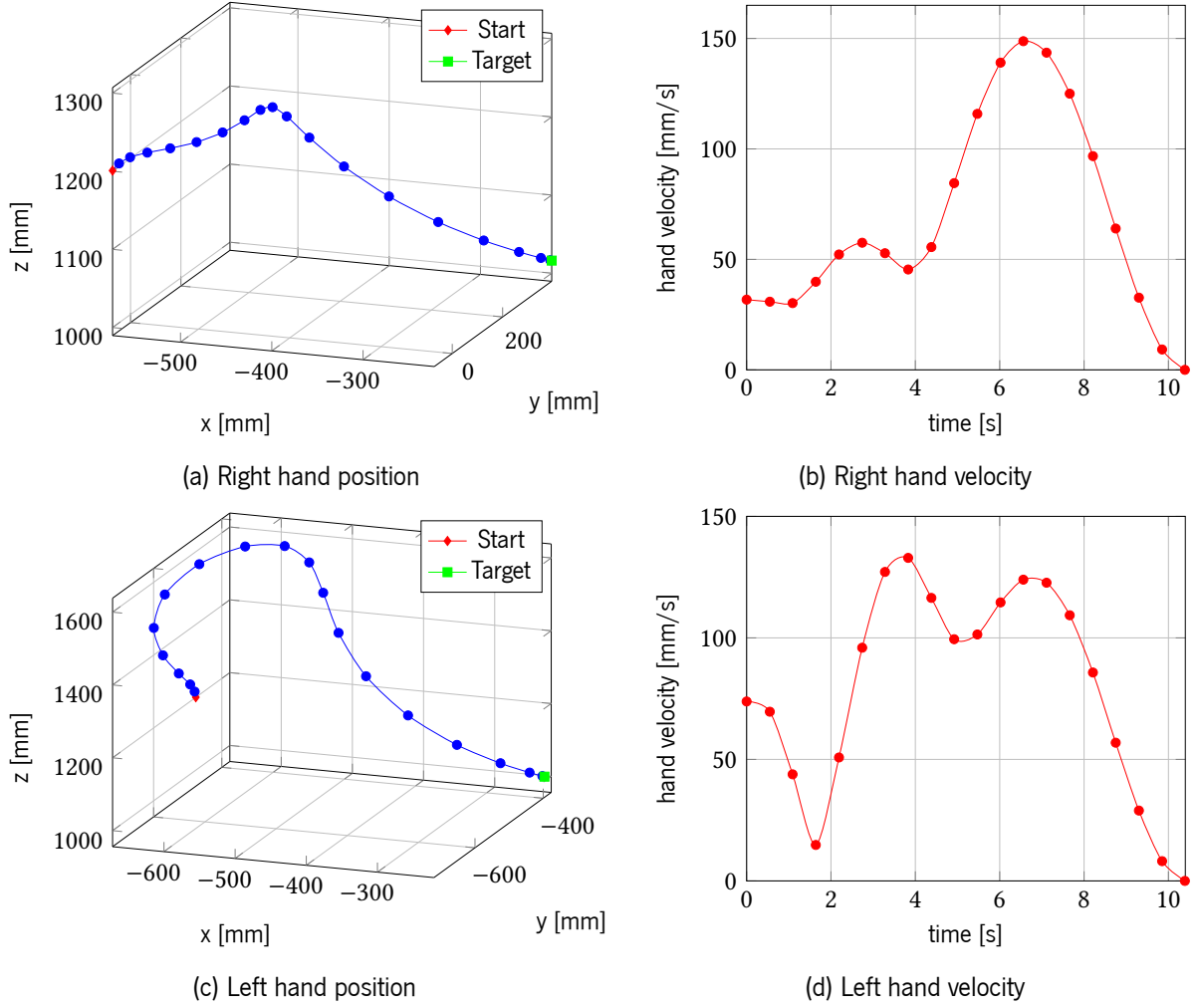


Figure 6.46: Synchronous bi-manual move movement planned by the HUMP (Right NJS=65.66, Left NJS=157.53, Right NMU=1, Left NMU=2, solving time = 6742 ms).

sufficient to raise the NMU to 2 (Figure 6.46b). For the correct composition of the task, the move movement starts with non-null boundary conditions taken from the previous place movement, which causes important decelerations of the left hand trajectory.

The complete concatenation of the planned synchronous bi-manual pick, place and move movements is illustrated in Figure 6.47. It can be noticed that the robot only stops for grasping and un-grasping the tray during picking and placing, respectively. The right hand trajectory is mainly characterized by the three uni-modal velocity profiles slightly shaped by second peaks during placing and returning to the initial configuration (Figure 6.47b). The left hand trajectory is significantly influenced by important changes in the orientation of the end-effector, which add second peaks to the velocity profiles of the first pick and of the last move movements (Figure 6.47d). The composition of the latter with the second place movement provokes the rise of smaller peak in the left hand velocity profile after the un-grasping transition. This is

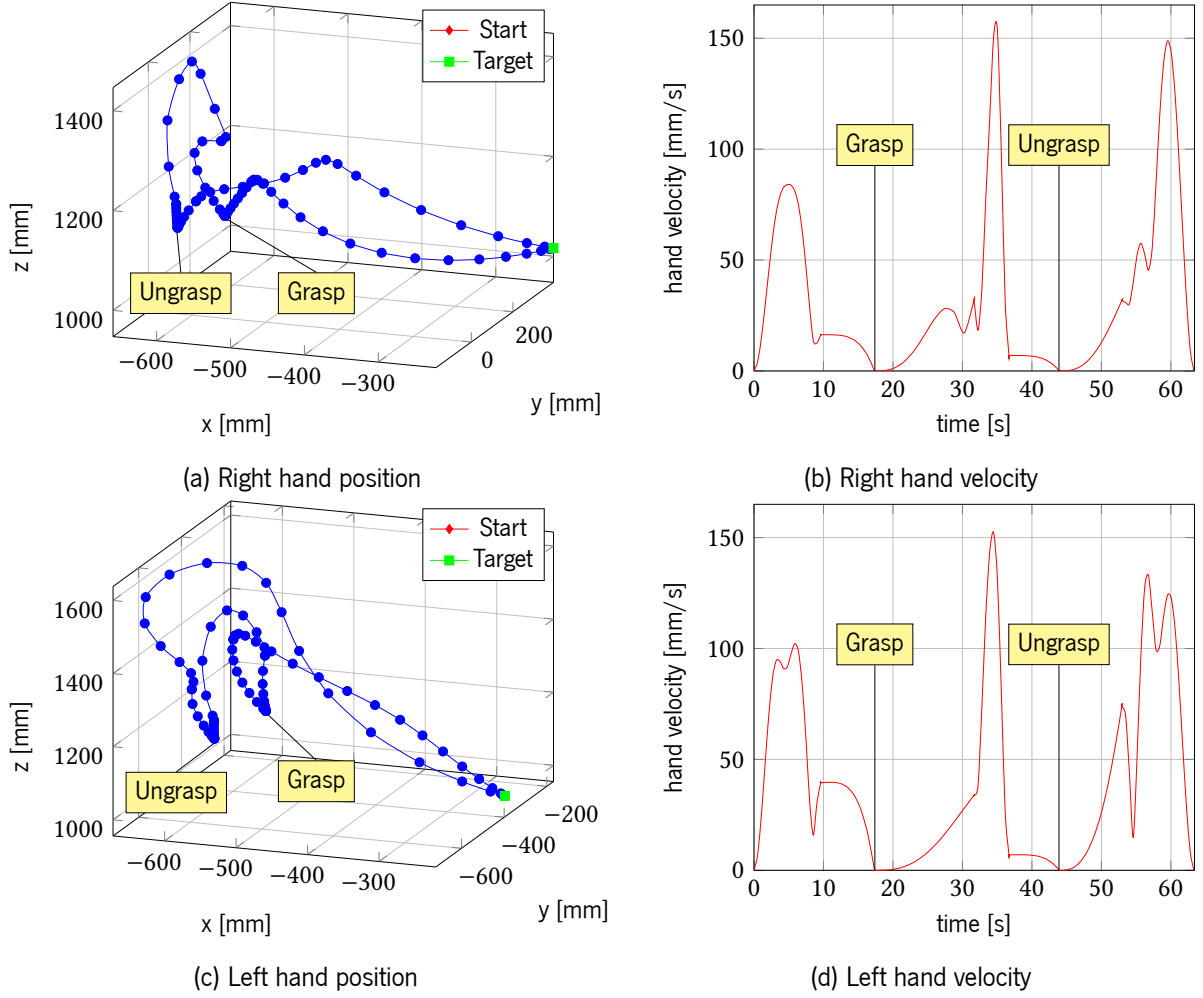


Figure 6.47: The complete synchronous bi-manual task planned by the HUMP.

due to an emphasized *Retreat* phase of the antecedent place movement, which necessarily transports the left hand away from the tray, and a sudden deceleration in position that serves for a rapid change in orientation of the palm.

6.3.6.3 Preliminary results with the real robot

This section reports preliminary results obtained with the execution of planned synchronous bi-manual movements on the real robot ARoS. Part of the content of this section has been published in Gulletta et al., 2015. The feasibility of the mathematical formulation introduced in section 5.6 is demonstrated and its limitations on real applications are discussed. Specifically, Figure 6.48 illustrates snapshots during tests on the tasks of swapping two columns in the toy vehicle scenario (section 6.3.6.1). During the execution of the *Retreat* phase of the first pick movement, there is a collision between the blue column and the base of the toy vehicle that does not negatively interfere with the correct accomplishment of the task. During

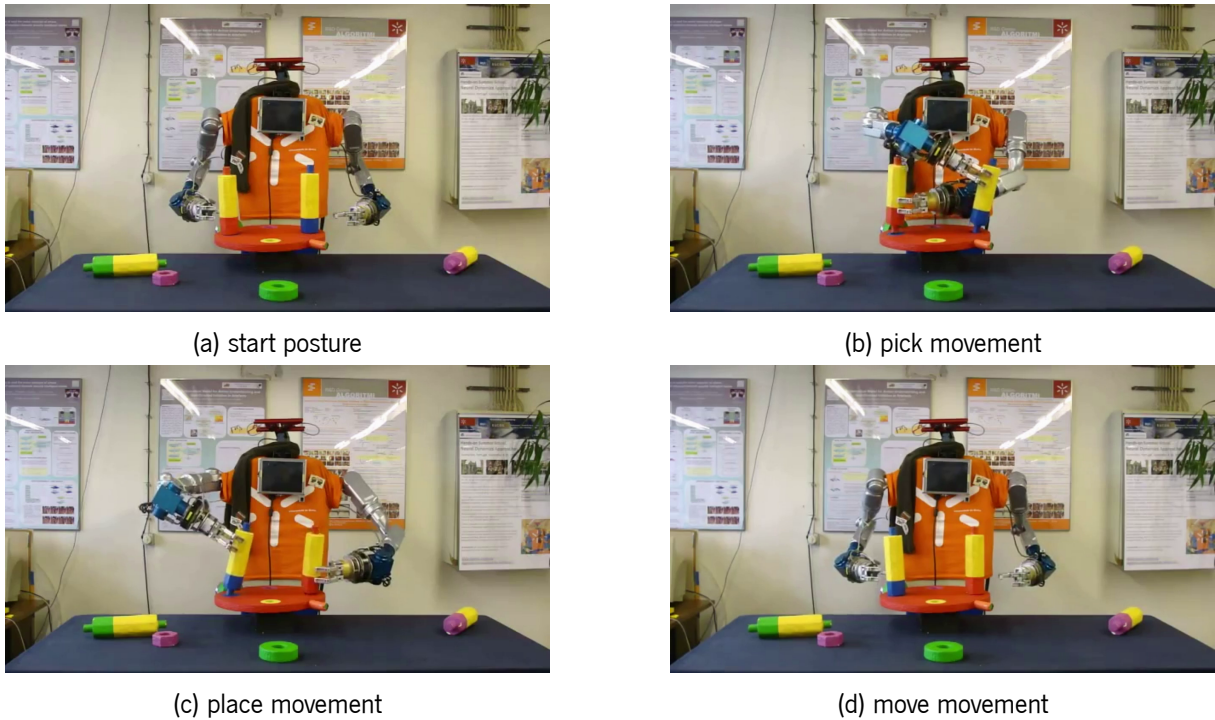


Figure 6.48: Four snapshots that show the sequence of the synchronous bi-manual movements planned by the HUMPT in the toy vehicle assembly scenario. When there is no adaptation to undesired contacts, the successful accomplishment of the task may be undermined. The complete video can be seen at the link <https://youtu.be/ZgnVPnBzKoA>.

the *Retreat* trajectories there is no collision avoidance and imperfections of the objects in the workspace can induce small contacts. This limitation might also occur when uni-manual movements are planned, but it is more likely to happen during the execution of synchronous bi-manual movements for a more limited free space of manipulation. For this reason, it can be affirmed that inter-limbs collision avoidance is tested successfully, but it is difficult to ensure a certain degree of reliability on real applications without systems of on-line adaptation to unexpected events.

Similar conclusions can be drawn when one single object is being manipulated by synchronous two-handed movements. In Figure 6.49 four snapshots during the transportation of the base of the toy vehicle are illustrated. The base is successfully transported from the white box onto the black one with the help of a human partner. Tolerances on inter-hands positioning and orienting may provoke significant misalignments of the object, which might negatively affect the execution of the given bi-manual task (Figure 6.49c). Moreover, a safe bi-manual manipulation of the object would not only require visual feedbacks for an on-line adaptation to unpredicted events, but sensing the external generalized forces acting on the manipulators would also be necessary to prevent possible damages and accidental drops. Therefore, the transportation of one object with synchronous two-handed movements has been successfully tested, but further improvements

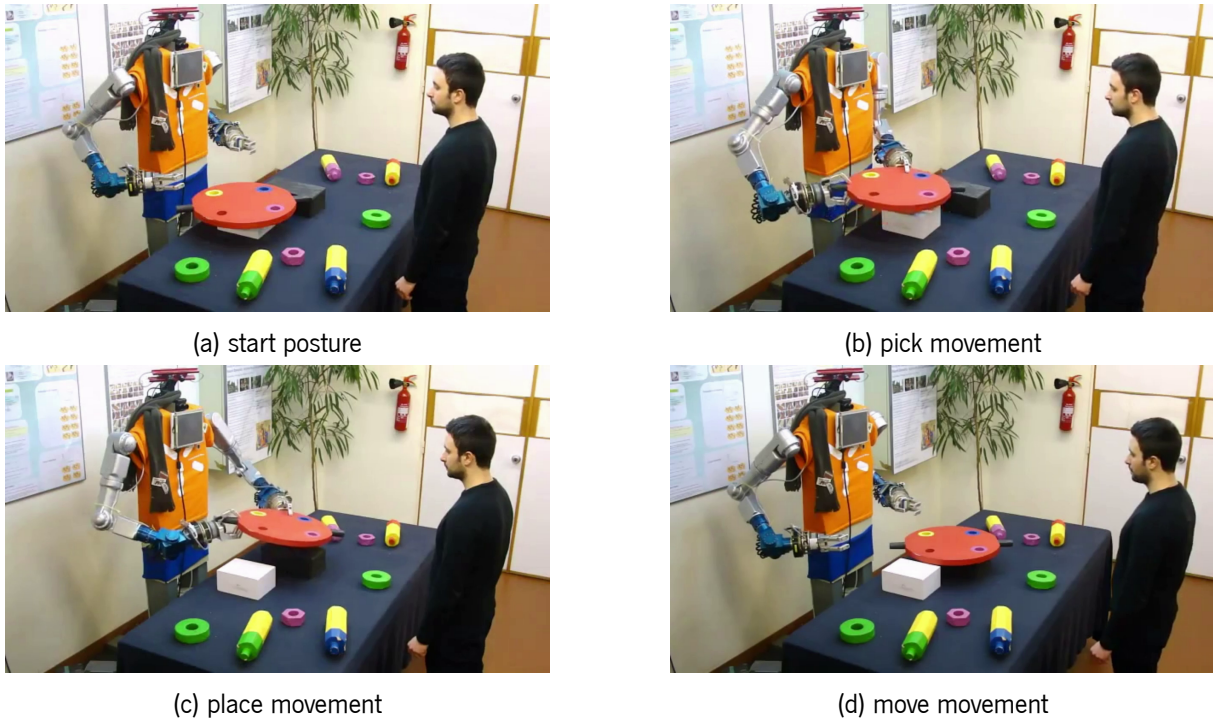


Figure 6.49: Four snapshots that show the sequence of the synchronous bi-manual movements planned by the HUMP in the toy vehicle assembly scenario. Misalignments of the base show the necessity of having an on-line adaptation mechanism. The complete video can be seen at the link https://youtu.be/w3J_F0J2LBg.

are needed to provide acceptable levels of reliability of real robotic applications.

6.4 Human-like upper-limb motion planning with Sawyer

The upgrades for the HUMP for the collaborative robot Sawyer, which are described in section 5.7, are here tested with the construction paradigm used in section 6.3.1 for the robot ARoS. Specifically, Figure 6.50 illustrates the pieces of the toy vehicle on a table in front of the robot Sawyer. A human partner is handing over a magenta column and the robot is instructed to reach and grasp it, insert it into the magenta hole of the base and go back to its start configuration. The complete sequence of the movements that compose the task is illustrated in Figure 6.51. The robot Sawyer is equipped with a Barrett Hand and starts the task in the configuration showed in Figure 6.51a. Then, the magenta column is picked from a given hand over position (Figure 6.51b) and assembled with the base of the toy vehicle (Figure 6.51c). In the last movement, the robot returns to its initial posture (Figure 6.51d).

The hand position and velocity profiles of the first pick movement can be observed in Figure 6.7. The end-effector moves with a single-peaked bell-shaped velocity with a significant *Approach* phase, which provide

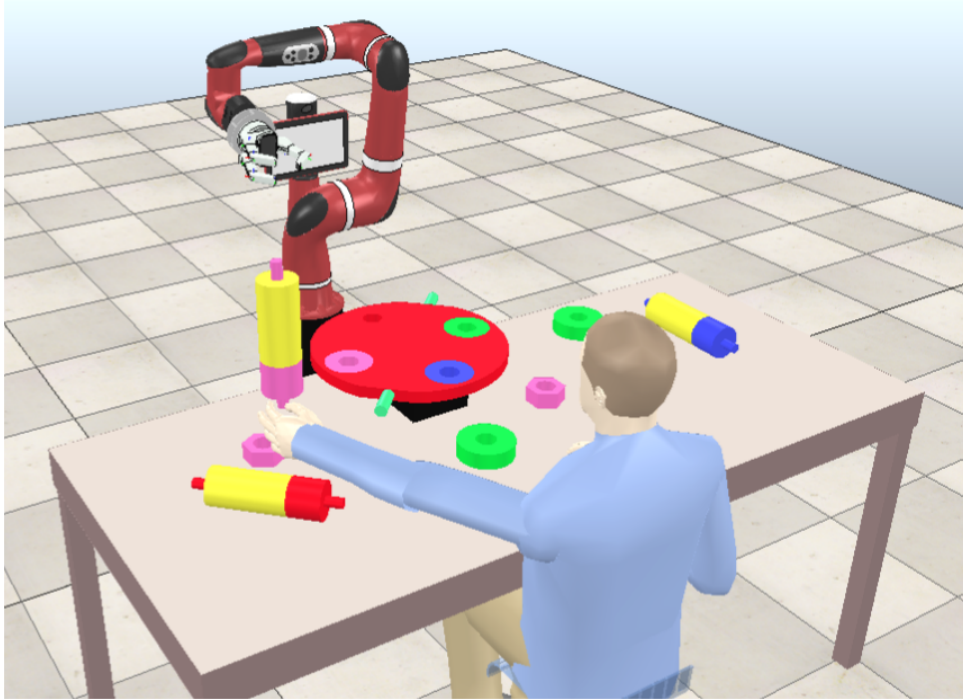


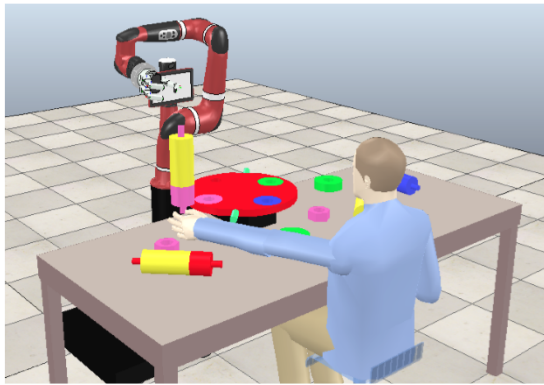
Figure 6.50: The toy vehicle assembly scenario with the robot Sawyer.

accuracy and show predictability to the human observer. The robot only stops for grasping the magenta column and a short *Retreat* phase is sufficient to depart from the given hand over position (Figures 6.52a and 6.52b). The picking problem was solved in 7229 ms and the solution resulted in very smooth movement with a NJS of 1448.36 and a single NMU.

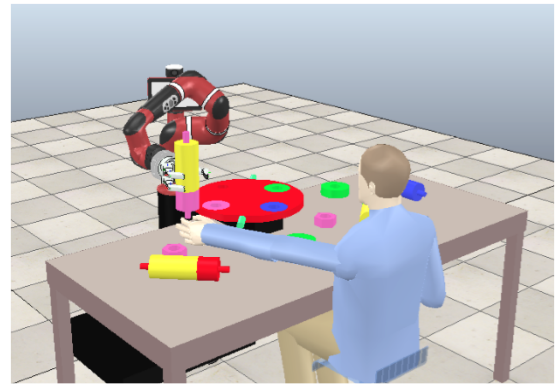
The kinematic features of the second *place* movement are illustrated in Figure 6.53. The act of placing the column starts with non-null boundary conditions taken from the antecedent *pick* movement and presents a single-peak on the bell-shaped hand velocity profile (Figure 6.53b). The *Approach* and the *Retreat* phases are of fundamental importance to guarantee a successful assemblage with the base and a collisions-free departing from the inserting area (Figures 6.53a and 6.53b). The placing problem was solved in 3272 ms and the solution provided a less smooth movement than the previous one with a single NMU and a NJS of 2502.

The kinematical characteristics of the last move movement are showed in Figure 6.54. The movement continues departing the end-effector from the area of insertion with a bell-shaped hand velocity profile that starts with non-null boundary conditions from the antecedent *place* movement (Figures 6.9a and 6.9b). This planning problem was solved in 4120 ms and the resulting trajectory is very smooth with a single NMU and a NJS of 22.89.

The correct concatenation of the *pick*, *place* and *move* movements form the complete task of the considered construction paradigm. The kinematical features of this composition are illustrated in Figure 6.55. The



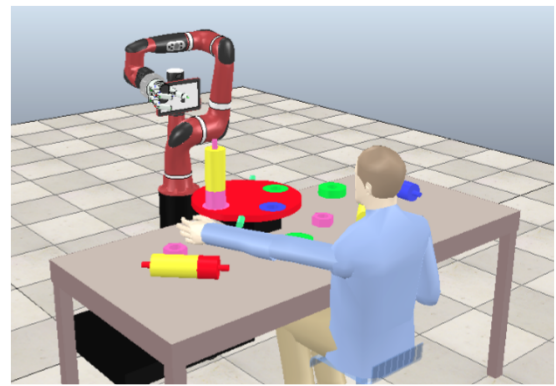
(a) start posture



(b) pick movement

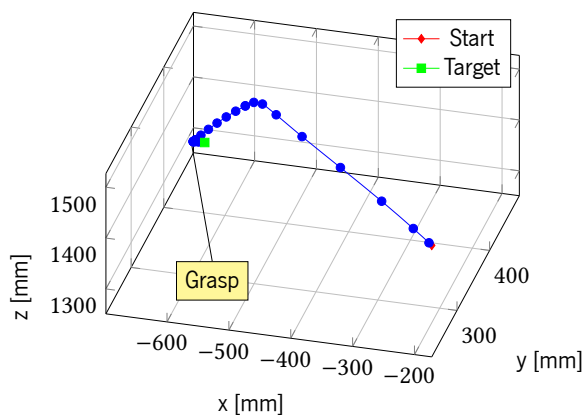


(c) place movement

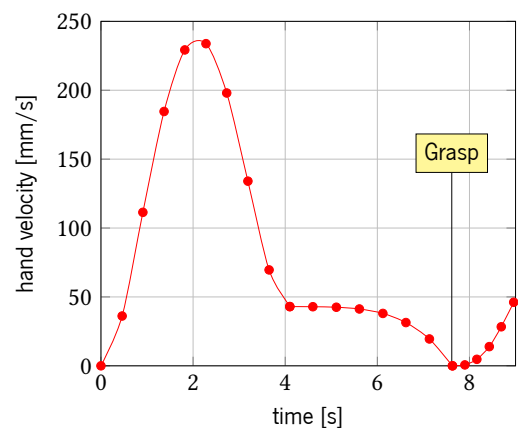


(d) move movement

Figure 6.51: Four snapshots that show the sequence of the movements planned by the HUMP for the simulated robot Sawyer in the toy vehicle assembly scenario. The complete video can be seen at the link <https://youtu.be/QpEUIGuBpXU>.



(a) Hand position



(b) Hand velocity

Figure 6.52: Pick movement planned by the HUMP for Sawyer (NJS=1448.36, NMU=1, solving time = 7229 ms).

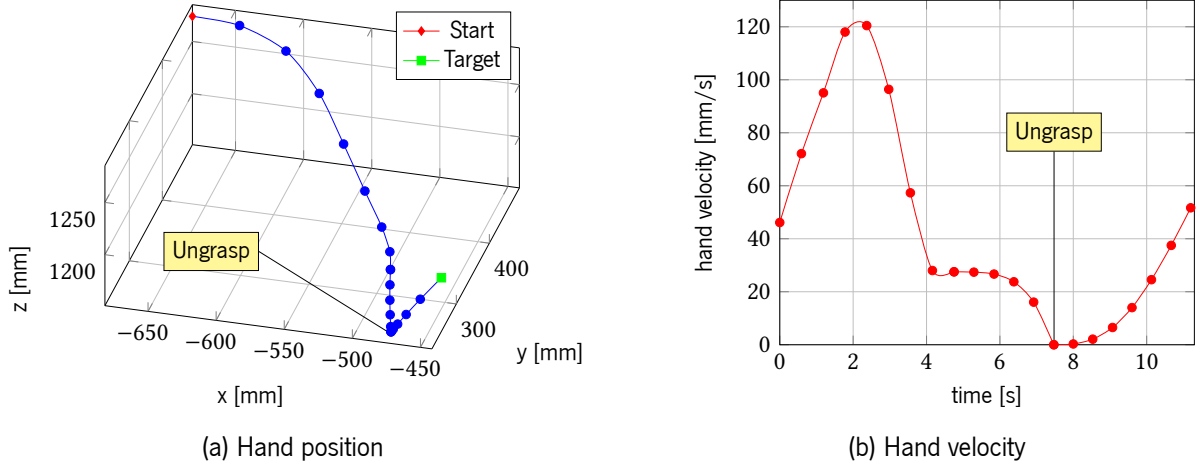


Figure 6.53: Place movement planned by the HUMP for Sawyer (NJS=2502.39, NMU=1, solving time = 3272 ms).

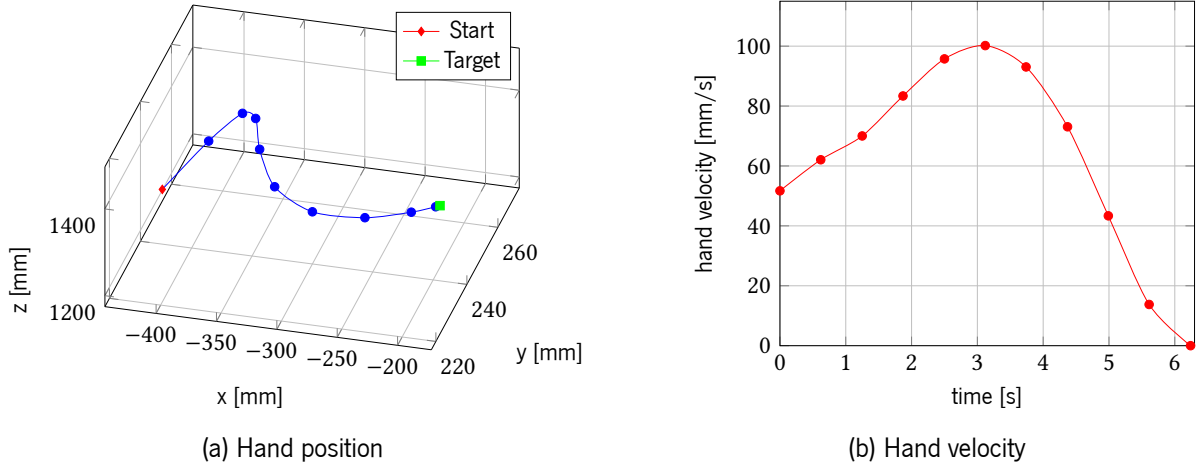


Figure 6.54: Move movement planned by the HUMP for Sawyer (NJS=22.89, NMU=1, solving time = 4120 ms).

robot Sawyer only stops for the grasping and un-grasping transitions of the pick-and-place sequence and that three bell-shaped peaks are a clear sign of human-likeness in the generated motion (Figure 6.55b). As it was expected the modifications on the geometrical representation of a robot increased the complexity of the constrained optimization problems for the selection of target and bounce postures, which inevitably raised the time required for planning (details are given in section 5.7).

The first experiments with the real robot Sawyer are here illustrated and collected in a video (Figure 6.56). For simplicity of implementation on real applications, the robot Sawyer is equipped with the original gripper from Rethink Robotics™ instead of a three-fingered Barrett Hand. The adoption of a gripper in the upper-limb formulation (section 4.3) is made by adding one prismatic joint to the arm posture and set it in

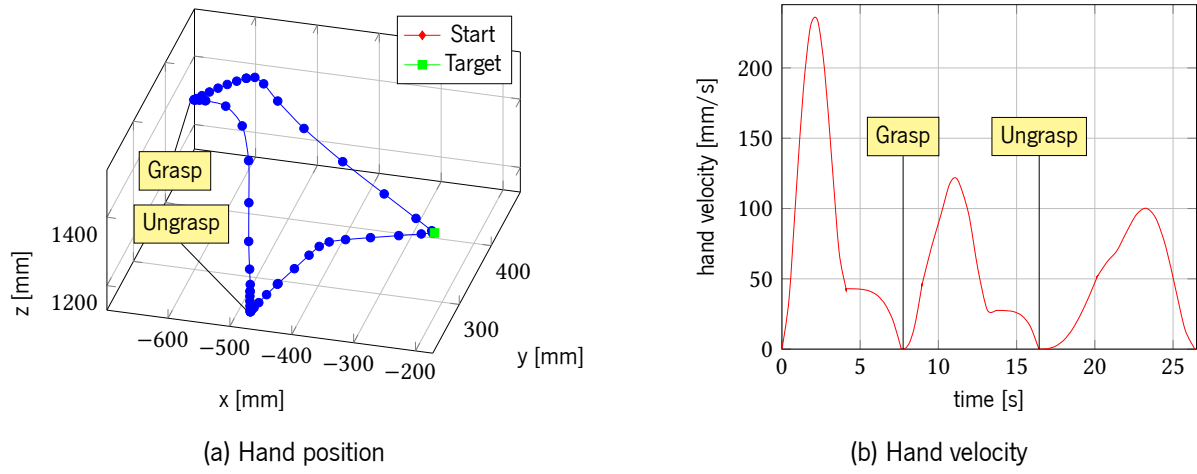
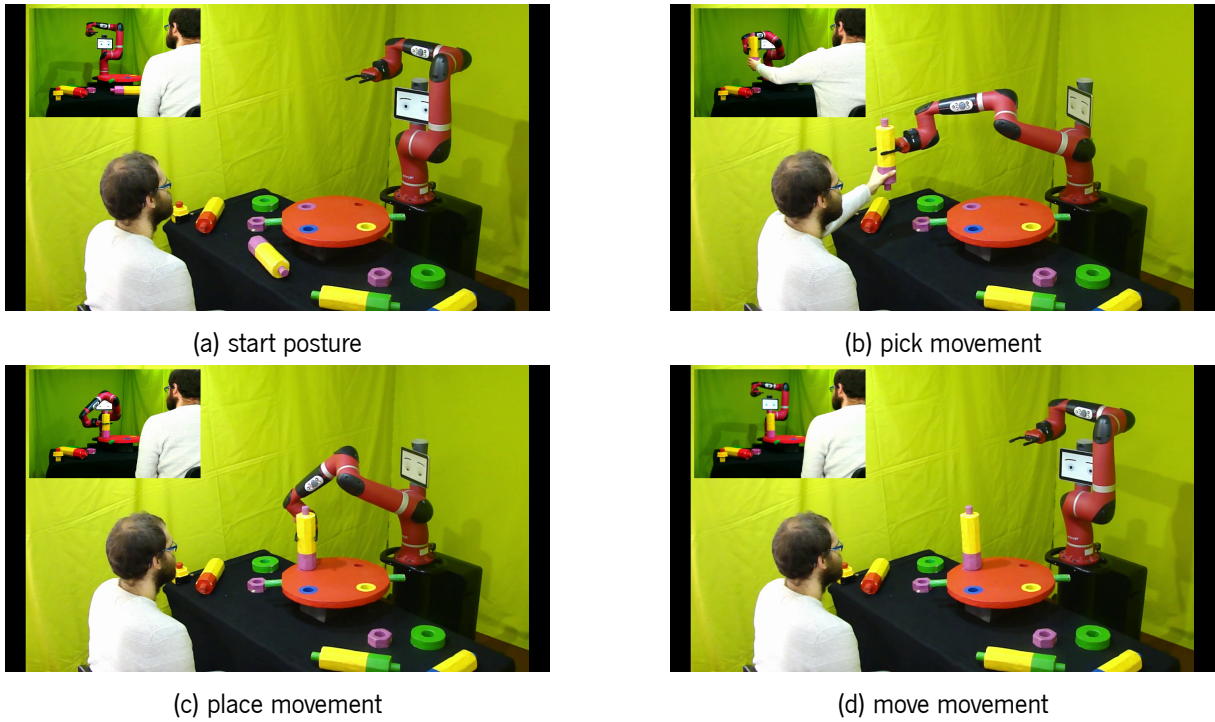


Figure 6.55: The complete task planned by the HUMP for Sawyer.

Figure 6.56: Four snapshots of the sequence of the movements planned by the HUMP for the real robot Sawyer equipped with a gripper in the toy vehicle assembly scenario. The complete video can be seen at the link <https://youtu.be/wJWujc-0I-Y>.

accordance with the diameter of the object that is being manipulated. Similarly to the geometrical representation of the Barrett Hand (section B), the gripper was modelled as a set of spheres placed arbitrarily on the most exposed parts and added to the kinematic chain of the manipulator. These modifications on the structure of the robotic upper-limb do not change the conclusions generally drawn from experimentation with the Barrett Hand in simulated environments. This consideration can also be intuitively determined through a qualitative analysis of the video reported in Figure 6.56. It can be, therefore, affirmed that the feasibility of the upgrades introduced in section 5.7 was also successfully tested on real environments.

Part III

Learning

Chapter 7

Adaptive learning of human-like upper-limb postures

In this chapter, a system capable of adaptively learning human-like upper-limb postures is proposed. The key characteristics of the method are mainly related to the possibility of online incremental learning concepts, which are implicit in the experience gained by the robot in planning sets of similar movements. Additionally, adaptation to new concepts is enabled by the application of a time-relevant forgetting policy on the training datasets. section 7.1 introduces the principal notions concerning experiential learning in human beings and a short state of art on learning from experience in robotics is discussed. In section 7.2, general insights on the optimizer IPOPT (Wachter and Biegler, 2005) and different techniques for warm-starting interior-point methods are described. Details of the proposed posture learner are explained in section 7.3, while section 7.4 reports experiments and results deriving from its testing on different reaching tasks.

7.1 Introduction

In human beings, ideas, which can be conceptualized as the mental atoms of thoughts, are formed and re-formed through experience. This process is continuous and results in learning creative ways for adaptation. In specific, the so-called “experiential learning” (or “learning by doing”) paradigm was firstly introduced by Kolb (1984) as the process of creating knowledge from the transformation of previous experience. The classic model is illustrated in Figure 7.1. It starts with concrete experience, follows with the reflective observation of the consequent events, then abstract conceptualization transforms experience into new hypothesis and active experimentation tests them. Therefore, the cycle can be initiated again by going through unprecedented experience. This continuous learning procedure finds feedbacks on subjective experiences related to feelings and on objective experiences that respond on environmental events. In Rizk, 2011, for

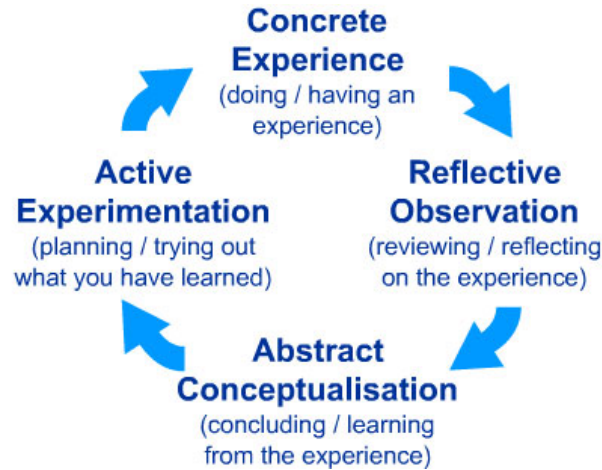


Figure 7.1: The experiential learning cycle (Kolb, 1984).

instance, this learning paradigm was successfully tested on a professional environment. A three-stage model of an experiential approach was applied to train the staff of the National Library of Egypt. Results were specifically linked to the implementation of the reflective practice and showed satisfying outcomes. Jensen et al. (2014) demonstrated that the experience of determined actions is significantly dependent on the intention and not on body movements. Specifically, it was showed that the source of an action is reliably recalled after the execution of voluntary movements. On the contrary, the participants to the study could not successfully recall the action of imposed involuntary movements when these ones were not congruent with the voluntary ones. Therefore, the experience of the actions are strongly linked to the intentions and not to the low-level motor execution signals. Intuitively, the underlying intention of a movement during pick-and-place tasks can be expressed by a specific type of grip, which uniquely specifies the hand position and orientation (pose) with respect to the object to manipulate (e.g., see Bicho et al., 2011b). A hand target pose can then be translated to a target arm posture by a selection among a set of stored postures, which are given from past experience on similar tasks (Rosenbaum et al., 2001). When predictable perturbations on the target pose occur, it has been demonstrated that the Central Nervous System (CNS) optimizes feed-forward planning rather than feed-back planning, which is instead prioritized during unpredictable disturbance (Fan et al., 2006). Therefore, the feed-forward model is dominant in producing accurate movements in static scenarios or in environments subjected to predictable changes. In the feed-forward paradigm, the prediction of a motion comes in support to the planning process that can best adapt to disturbed situations (Burdet et al., 2013; Schwartz, 2016).

Roboticians have been working in searching methods to improve the performances of motion planners through learning and prediction. For instance, Lampariello et al. (2011) introduced a trajectory planning

method that can be applied on applications with real-time demands, such as ball-catching by a robotic manipulator. In specific, the problem of catching a small flying object is formulated as a non-linear constrained optimization problem that allows to select a collisions-free joints-space trajectory with the minimum mechanical energy over the duration of the motion. Constraints on the end-effector ensure either a static or a dynamic grasp at the moment of catching through the selection of different boundary conditions. The search of the global optimum, starting from random initial guesses of the optimization parameters, demonstrates the presence of local minima the optimizer may get stuck in when it is not guided on a cost-efficient basin of attraction. To this purpose, different machine learning techniques are compared on a training dataset that relates the object velocity with good guessing points to efficiently starts the local planner and obtain a quick convergence on a feasible trajectory. On-line implementations of the catching task exhibited a clear trade-off among the amount of the training data, the time required for planning and the average cost increase.

In Dragan et al., 2011, the authors proposed a semantic learning framework for optimization-based motion planners that, like CHOMP (Zucker et al., 2013), might get stuck in high-cost solutions. The optimization of a trajectory is significantly improved with good initial guesses that lead towards low-cost feasible solutions. The prediction of these initial guesses is linked to qualitative attributes that represent semantic characteristics of motion, like “go up the microwave” and “go left the table”. A minimal set of situational features, which describe the environment surrounding a robot, are mapped to these attributes that identify a subset of trajectories capable of placing the local optimizer into the most efficient basin of attraction. The selection of one of these trajectories can, therefore, smartly initialize and guide optimization towards low-cost solutions. The mapping between situations and qualitative attributes of a movement forms meaningful experience one can learn from. Experimentation with different learning techniques showed better performance with non-linear regressors and with the application of inverse optimal control. With sufficient amount of training data, good generalization capabilities were also exhibited.

Fast motion planning that benefits from past experience was also investigated by Jetchev and Toussaint (2013). In specific, the prediction of a trajectory from past optimal motion can improve the performance of movements generation in novel situations. Data concerning situational features and feasible low-cost trajectories are firstly collected by running a planner multiple times with default initialization. Then, the online prediction of good initializations is learnt and provided to speed up the performance of motion planning (Figure 7.2). Situations are represented with a high-dimensional descriptor that evaluates the task space with geometrical relationships between the robot and the objects surrounding its neighbourhood. A voxel descriptor is also utilized to identify probabilities of having obstacles in different regions of the workspace. Given a novel situation, these descriptors are used for predicting the optimal planning solution to the most similar situation in the database. Then, a task-space operator transfers this optimal solution to the novel

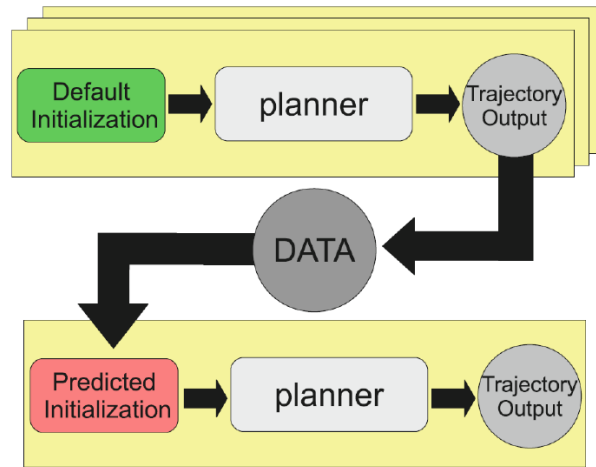


Figure 7.2: Illustration of trajectory prediction for planning in perturbed situations (Jetchev and Toussaint, 2013).

situation in order to feed the motion planner with good initialization parameters. Classical regression techniques were compared with a nearest neighbour predictor in different scenarios cluttered with obstacles. Results showed that the proposed trajectory prediction method could guide the planner towards a quick convergence with a smaller feature vector than classical regressors, which may lack of generalization from a given training dataset.

Optimization-based motion planners like CHOMP (Zucker et al., 2013), STOMP (Kalakrishnan et al., 2011) and TrajOpt (Schulman et al., 2014) may get stuck in local optima solutions due to the non-convex formulations of constraints for obstacle avoidance. For this reason, their performance strongly depends on the provided trajectory as initial guess for the optimization. In this context, Pan et al. (2014) proposed a binary classification that is trained on a database of 10 thousands trajectories per scenarios, which are labelled as “good”, when they lead to feasible solutions and as “bad” otherwise. Different combinations of situational features based on Spatial Signed Distances (Schulman et al., 2014) are experimented to predict good initial guesses for the TrajOpt planner. Results showed that the accuracy of classification increases with the number of features in a given scenario. However, the performance of the classifier decreases when transferring on different situations. One of the major reasons may depend on the design of the features that locally represent the space around a trajectory and ignore the global description of the environment.

The Human-like Upper-limb Motion Planner, which is described in chapter 5 and tested in chapter 6, depends on the solution of two different non-linear constrained optimization problems for the selection of target and bounce postures (see details section 5.1). Since these problems are characterized with non-convex constraints for collisions-avoidance, the interior-point filter line-search algorithm implemented by IPOPT (Wachter and Biegler, 2005) may encounter local optima of infeasible solutions and, more generally,

may have a decline on its performance, when it is inadequately initialized. Inspired from the mentioned studies on trajectory prediction in robotics, an adaptive learning technique is here described to map seen past situations with optimal target postures and predict good initial guesses on-demand for unseen situations. The main goal of learning into environments with predictively situational changes is to place the optimizer into low-cost basins of attraction and increase its planning performance. In specific, the capabilities of warm-starting IPOPT were used to feed initializations that can be predicted by learning on past motion planning experiences.

7.2 Warm starting interior-point methods

Primal-dual interior-point methods are algorithms derived from the applications of log-barrier functions to obtain approximate solutions (a primal-dual pair values) of the local optimum (Nocedal and Wright, 1999). Wachter and Biegler (2005) introduced IPOPT as the implementation of primal-dual interior-point methods for large-scale non-linear optimization. Generally, a non-linear optimization problem can be formulated as showed by Equations 7.2.1–7.2.3.

$$\min_{x \in \mathbb{R}^n} f(x) \quad (7.2.1)$$

$$\text{s.t.} \quad c(x) = 0 \quad (7.2.2)$$

$$x \geq 0 \quad (7.2.3)$$

where, $x \in \mathbb{R}^n$ is the vector of primal variables, $f(x)$ is the objective function, $c(x)$ represents equality and inequality constraints and the condition $x \geq 0$ ensures the respect of boundary conditions. Inequalities in Equation 7.2.2 are reformulated as equalities by introducing slack variables, while the condition in Equation 7.2.3 is obtained by normalizing on lower and upper bounds (Wachter and Biegler, 2005). In order to solve the problem (7.2.1)–(7.2.3), IPOPT computes approximate solutions of a sequence of barrier problems (7.2.4)–(7.2.5) for decreasing values of the barrier parameter $\mu \geq 0$.

$$\min_{x \in \mathbb{R}^n} \varphi_\mu(x) = f(x) - \sum_{i=1}^n \ln(x_i) \quad (7.2.4)$$

$$\text{s.t.} \quad c(x) = 0 \quad (7.2.5)$$

where $\varphi_\mu(x)$ is the objective barrier function and x_i is the i^{th} component of the vector $x \in \mathbb{R}^n$. Equivalently, these solutions are characterized by the Karush-Kuhn-Tucker (KKT) conditions in Equation 7.2.6–7.2.9

$$\nabla f(x) + \nabla c(x) \lambda - z = 0 \quad (7.2.6)$$

$$c(x) = 0 \quad (7.2.7)$$

$$XZe - \mu e = 0 \quad (7.2.8)$$

$$x, z \geq 0 \quad (7.2.9)$$

where λ and z are the Lagrangian multipliers of the constraints in Equations 7.2.2 and 7.2.3, respectively, $X = \text{diag}(x)$, $Z = \text{diag}(z)$ and $e = (1, 1, \dots, 1)^T \in \mathbb{R}^n$. The unscaled optimality error of each barrier problem is given by Equation 7.2.10.

$$E_\mu(x, \lambda, z) = \max \{ \|\nabla f(x) + \nabla c(x) \lambda - z\|_\infty, \|c(x)\|_\infty, \|XZe - \mu e\|_\infty \} \quad (7.2.10)$$

The optimality error of the original problem (7.2.1)–(7.2.3) $E_0(x, \lambda, z)$ (with $\mu = 0$) is related to the approximate solution $(\tilde{x}_*, \tilde{\lambda}_*, \tilde{z}_*)$ and causes the termination of the algorithm when the condition (7.2.11) is respected.

$$E_0(\tilde{x}_*, \tilde{\lambda}_*, \tilde{z}_*) \leq \epsilon_{tol} \quad (7.2.11)$$

where ϵ_{tol} is a given error tolerance. Primal-dual methods generate iterates of the variables (x, λ, z) that satisfy the condition (7.2.9) strictly. This property is behind the origin of the name *interior-point* and the performance of the algorithm mainly relies on its ability to remain on the central path of the feasible region (Nocedal and Wright, 1999; Wachter and Biegler, 2005). For this reason, the provided initialization can be modified by IPOPT to ensure that the initial point is sufficiently away from its boundaries, which can also be relaxed to avoid numerical issues in the line-search procedure (Wachter and Biegler, 2005). These features of the algorithm make difficult its warm start by considering the optimal solution of a “similar” problem because it would lead to the blocking of the search direction, if it was close to the boundaries of the feasible region (Gondzio and Grothey, 2002; Shahzad, 2010).

In literature, different techniques for efficiently warm-starting interior-point method have been presented and tested. The majority of these strategies is referred to linear programming (LP), while only few assumptions have been inferred on warm-starting interior-point solvers for the resolution of non-linear programming problems. For example, Yildirim and Wright (2002) proposed two warm-strategies that take into consideration the solution of an “original” LP instance to generate a good starting point for a “perturbed” LP instance. Adjustments on the given solution are computed in relation with the size of the perturbations. The value of the initial barrier parameter is selected to be not too large and derived from the analysis

of the iterates generated while solving the original LP problem. Worst-case estimates on the number of iterations required to obtain convergence after the warm start-up are derived and mainly depend on the size of the perturbations. Similarly, Gondzio and Grothey (2002) proposed warm-starting strategies for re-optimization in a sequence of linear programming problems. Specific conditions of the perturbations that form a “nearby” LP problem are discussed to guarantee the assimilation of the perturbations in one Newton step. A good starting point candidate ensures a small initial barrier parameter, small primal and dual infeasibilities and a specifically bounded complementarity measure. Numerical tests on the developed routine reported a reduction of up to 40% on the number of iterations in comparison with the standard starting point strategy. Warm-starting interior-point methods for non-linear programming were also discussed and the outcomes were specialized for linear programming (Forsgren, 2006). In particular, warm-starting interior-point methods from the solution of a “nearby” problem turned out to be closely related to solving an equality-constrained problem by sequential quadratic programming (SQP). In other words, for small perturbations, a primal-dual step behaves like a SQP step when warm-starting from a “nearby” problem. For large perturbations, the step is likely to violate the non-negativity of the multipliers and to ignore the inactive constraints, i.e. those constraints that do not affect the value of the objective function. John and Yildırım (2008) reviewed and tested four different warm-start strategies for interior-point methods applied in linear programming. All the iterates of the “original” LP instance need to be saved to compute adjustments on its resulting solution that starts the “perturbed” LP instance. Three variations of least-squares-based adjustments and a Newton-step-based adjustment are compared on the resolution of LP instances under different types of perturbations. The performance of the described techniques is measured with the number of iterations and the CPU time needed to reach convergence. Results showed that a default parametrization of a least-squares-based adjustment outperformed the other techniques. A study on predictive control was presented by Shahzad (2010), where a sequence of quadratic programming (QP) problems need to be solved at each sampling instant time. The QP instances are “similar” and their optimization can be warm-started by taking into account the solution of an antecedent instance. Shahzad (2010) proposed a strategy to warm start an interior-point method applied to solve this sequence of QP problems. The initial guess of the primal variables, which form the vector of commands in predictive control, is set to be equal to the solution of the prior problem. The multipliers are set such that the set of active constraints remain equal for the first iteration of the subsequent problem and the starting point is also centred away from the boundaries of the feasible region. Tests on two problems showed that there is a significant reduction of the computational cost when compared to the standard “cold-start” initialization. Due to the non-linearities of the target and the bounce selection problems (chapter 5), none of the cited studies can be applied for an efficient warm-start of the solver IPOPT. Therefore, other techniques need to be investigated. The proposed warm-starting strategy takes advantage of the experience gained

during the generation of robotic upper-limb trajectories in order to offer efficient initializations to the motion planner in “unseen” similar tasks and situations. Specifically, the presented learning technique concerns the prediction of efficient initial guesses to warm-start the optimization of the problem (5.1.4)–(5.1.7) for the selection of human-like arm target postures.

7.3 An adaptive Human-like Arm Posture Learner

The proposed learning method aims at simulating the human experiential learning cycle (Figure 7.4) that has found evidences in different applications and environments (Kolb, 1984; Rizk, 2011). The presence of a cycle infers an *online* and *incremental* manner of learning that is *adaptive* to the incoming data of the experience gained during active experimentation. In the learning cycle paradigm, a training dataset is updated regularly according to one or more *data streams*, i.e. ordered sequences of data items with inputs arriving continuously as the time progresses (Beringer and Hüllermeier, 2007). The order of the inputs is not under the control of the system and different policies can be applied to either discard or store in memory a new incoming element. Strict time requirements generally encourage incremental learning instead of learning from scratch every time new data are arriving. Limitations on the memory size demand the application of *forgetting* strategies to maintain a meaningful level of adaptation to any detected changes of the inputs (Beringer and Hüllermeier, 2007; Salganicoff, 1993b). Interestingly, the capability of incrementally update a learning model with each new training sample is a typical characteristic of the cerebral cortex of the brain (Lowe, 1995). Lowe (1995) also indicated the fact that biological learning mechanisms do not suffer of strong interference, i.e. learning in a subset of the input space does not cause any degradation of the learning performance in other parts of the input space. *Lazy* or *memory-based* learning methods can guarantee to avoid negative interference between old and new data and are suitable for the implementation of online robot learning strategies (Atkeson et al., 1997). Being opposed to model-based machine learning techniques that generate a model on the given data, these techniques can also be called *instance-based* learning algorithms because a query is answered by selecting relevant instances on a stored memory (Beringer and Hüllermeier, 2007). This relevance is often measured by *distance* functions, which can be defined according to the application for an efficient detection of “nearby” points at query time. Typical lazy learning algorithms are the *nearest neighbours search learning*, which makes predictions in correspondence of each neighbour of the query point and the *locally weighted learning*, which combines nearby training points to extrapolate new predictions. The approach described in this section applies *locally weighted regression* that uses a weighted distance to fit a function on previous motion experiences and, then, predicts optimal initializations during the selection of human-like arm target

postures. Specifically, the proposed adaptive learning technique takes inspiration from the *trajectory prediction* method introduced by Jetchev and Toussaint (2009) and Jetchev and Toussaint (2013), implements regularizations for incremental learning described by Lowe (1995) and applies a *time weighted forgetting* strategy to update the collected data (Salganicoff, 1993b).

The definition of the predicting problem is outlined in section 7.3.1, while the formulation of a geometrical situational descriptor is given in section 7.3.2. The collecting and updating data policies are described in section 7.3.3 and section 7.3.4 introduces the strategy of predicting low-cost initializations. Finally, in section 7.3.5 a loss function is formulated to train a vector of learning parameters and in section 7.3.6 the proposed adaptive learning method is summarized in four cyclic stages.

7.3.1 Problem definition

A sequence of primal and dual variables, which is searched as the solution of the optimization problem (7.2.6)–(7.2.9), is denoted with the vector $\mathbf{q}$ as shown in Equation 7.3.1.

$$\mathbf{q} = (x, \lambda, z) \quad (7.3.1)$$

$$C(\mathbf{f}, \mathbf{q}_*) = E_0(\mathbf{q}_* | \mathbf{f}) \quad (7.3.2)$$

$$\mathbf{q}_* = \arg \min_{\mathbf{q}} C(\mathbf{f}, \mathbf{q}) = \mathcal{O}_f \mathbf{q}_{init} \quad (7.3.3)$$

$$\mathbf{f} \rightarrow \mathbf{q}_{init} \rightarrow \mathcal{O}_f \mathbf{q}_{init} = \mathbf{q}_* \quad (7.3.4)$$

The unscaled optimality error (Equation 7.2.11), which is computed at the solution $\mathbf{q}_* = (\tilde{x}_*, \tilde{\lambda}_*, \tilde{z}_*)$, can also be interpreted as a “cost” given by a solution $\mathbf{q}_*$ and a situational descriptor $\mathbf{f}$ that represents the environment around a robot and task-related specifications (Equation 7.3.2). Given a certain situation $\mathbf{f}$ and an initialization $\mathbf{q}_{init}$, a human-like arm target posture is selected by searching a solution $\mathbf{q}_*$ that minimizes the cost $C(\mathbf{f}, \mathbf{q}_*)$ (Equation 7.3.3). As shown in Equation 7.3.3, the act of minimizing can also be denoted with the planning operator $\mathcal{O}_f$ to highlight the situation $\mathbf{f}$ and the starting point $\mathbf{q}_{init}$. Similarly to the *trajectory prediction* method by Jetchev and Toussaint (2009) and Jetchev and Toussaint (2013), the aim of the proposed learning algorithm is to construct an approximate model of the mapping $\mathbf{f} \rightarrow \mathbf{q}_{init}$ (Equation 7.3.4) given datasets of previously optimized target postures. The resulting model permits to efficiently warm-start the motion planner for a significant reduction of computational costs and of the convergence speed.

7.3.2 A geometrical situation descriptor

A problem instance or situation is fully described by the initial configuration of a robotic manipulator, the objects in its workspace and the targets specified by a task. The proposed approach considers a high-dimensional and redundant geometrical descriptor reported in Equation 7.3.5.

$$\mathbf{f} = (\psi_1, \psi_2, \dots, \psi_a, \phi_1, \phi_2, \dots, \phi_b) \in \mathbb{R}^{s=a+b} \quad (7.3.5)$$

where $a = 3n_{arm} + 6(n_{obst} + 1)$, $b = 7(n_{arm} + n_{obst} + n_{arm}n_{obst})$ with n_{arm} being the number of considered points on a robotic arm and n_{obst} being the number of obstacles in the scene. The first ψ elements represent the Cartesian coordinates of the points on the manipulator and the position and the orientation (in *Roll-Pitch-Yaw* angles (Craig, 2005)) of the obstacles and of the target in the operational space. The ϕ elements of the situation descriptor $\mathbf{f}$ represent specific geometrical relationships between pairs of elements, such as robot point–obstacle and robot point–target. Similarly to the situational descriptor introduced by Jetchev and Toussaint, 2013, these relationships are given by the positional difference of the i^{th} and the j^{th} points in the world frame of the robot, i.e. $\mathbf{p}_{ij} = (p_{ij}^x, p_{ij}^y, p_{ij}^z)$, by their distance $d_{ij} = \|\mathbf{p}_{ij}\|$ and by the azimuth of the three axes $\{\arccos(p_{ij}^x/d_{ij}), \arccos(p_{ij}^y/d_{ij}), \arccos(p_{ij}^z/d_{ij})\}$ for a total of 7 elements for each pair. This geometrical description of a problem instance is appropriate for the application of feature selection techniques to infer which features are critical for the selection of a target posture and, therefore, need to be of major consideration in the prediction of warm-start initializations.

7.3.3 Data collecting and updating

Like for the *trajectory prediction* method (Jetchev and Toussaint, 2009; Jetchev and Toussaint, 2013), the learning of an approximate model for the mapping 7.3.3 works through the construction of two datasets, which collect the experience gained by the robot in previously optimized trajectories. In specific, pairs of situations $\mathbf{f}$ and solutions $\mathbf{q}$ are collected in a dataset D_{cold} that is formulated in Equation 7.3.6.

$$D_{cold} = \{(\mathbf{f}_i, \mathbf{q}_i), \forall i = \{1, \dots, d\}\}, \mathbf{q}_i = \arg \min_{\mathbf{q}} C(\mathbf{f}_i, \mathbf{q}) \quad (7.3.6)$$

The “cold dataset” D_{cold} is composed by solutions $\mathbf{q}$ that are initially obtained by an offline process with a standard initialization of the optimizer IPOPT. The standard initialization is called “cold initialization” in opposition to the strategies of warm-start in interior-point methods. In an online paradigm, the D_{cold} dataset is updated with new pairs of unseen situations and corresponding solutions, which are optimized in warm-start mode from predicted optimal initializations. The quality of the solutions and of predicted initializations depends significantly on the performance of the applied learning technique. For a better

understanding of the good grade of an initialization, a refinement cost is defined (Equation 7.3.7) and a new dataset is necessary to be considered by the proposed learning approach.

$$F(\mathbf{f}, \mathbf{q}) = C(\mathbf{f}, \mathcal{O}_f^j \mathbf{q}) \quad (7.3.7)$$

The refinement cost indicates how much refinement a given initialization would need to obtain an optimal target posture. In particular, the cost $C(\mathbf{f}, \mathcal{O}_f^j \mathbf{q})$ represents the cost associated to the solution $\mathcal{O}_f^j \mathbf{q}$ found by IPOPT after j iterations, starting from the initialization $\mathbf{q}$. By this definition, the cost F quantifies the effect a certain initialization has on the convergence speed of the optimizer (Jetchev and Toussaint, 2013). Another dataset D_{warm} is, then, defined in Equation 7.3.8 to highlight the influence of an initialization on different training samples.

$$D_{warm} = \{(\mathbf{f}_j, \mathbf{f}_i, F(\mathbf{f}_j, \mathbf{q}_i))\}, (\mathbf{f}_i, \mathbf{q}_i) \in D_{cold}, \mathbf{f}_j \in D_j \quad (7.3.8)$$

For each novel inexperienced situation $\mathbf{f}_j \in D_j$, IPOPT is warm started with initializations $\mathbf{q}_i$ associated with previous experienced situations $\mathbf{f}_i$ collected in the cold dataset D_{cold} . The effect on convergence speed that returns from warm starting with different solutions of “similar” experienced situations is, therefore, collected and at disposal for any technique of supervised learning. Specifically, as described in section 7.3.4, the presented learning approach is a *memory-based learning* method that applies locally weighted regression with an optimized situational similarity function.

Memory-based learning approaches simply rely on the storage of observations and do not have intrinsic mechanisms of adaptation to changing environments (Salganicoff, 1993b). Therefore, online memory-based learning systems, which are also meant to track the environment, need explicit algorithms to exclude samples that are considered no longer representative. Such algorithms are called *forgetting* algorithms and permit to track the changing concepts of the stored observations by applying policies of sample deletion (Salganicoff, 1993b). *Forgetting* methods mitigate the counterproductive resilient characteristics, which are typical of biological learning systems (Salganicoff, 1993a). In human learning, indeed, *resiliency* is that property that makes difficult to unlearn some behaviour when it has been trained for a long period of time (Salganicoff, 1993a). Therefore, on-line experiential learning approaches that operate on many sessions of planning instances have the necessity of forgetting obsolete concepts that do not represent the most relevant observations. The proposed learning method becomes adaptive to the environment by the application of a time-weighted forgetting (TWF) algorithm that privileges the most recent incoming samples, while the older ones are gradually forgotten and eventually eliminated by the memory. The TWF method limits the maximum number of samples that can be stored and makes efficient use of the memory size by fixing a constant size window on the data stream (Salganicoff, 1993b). In specific, each sample of the datasets D_{cold} and D_{warm} are associated with weights w , which are initialized to 1 for the most recent

samples and decremented over time according to the rule given by Equation 7.3.9.

$$w_{k+1} = \tau w_k, 0 < \tau \leq 1 \quad (7.3.9)$$

Then, the weight value at the time step $k + 1$ is always smaller than the value at time step k of a rate controlled by the parameter τ . When the value of a weight w becomes smaller than a given threshold θ , the associated sample is deleted from the memory.

7.3.4 Locally weighted regression

Assuming that similar situations f would be paired with similar solutions q , a situational similarity function is defined by Equation 7.3.10. It is a Gaussian kernel centred in a given situation f with d_i being the distance of the i^{th} neighbour on a set of N nearest neighbours.

$$k(f, f_i) = \exp(-d_i^2/2\sigma^2) \quad (7.3.10)$$

$$d_i^2 = (f - f_i)^\top W (f - f_i) \quad (7.3.11)$$

$$W = \text{diag}(w_1^2, w_2^2, \dots, w_s^2) \quad (7.3.12)$$

$$\sigma = \frac{r}{M} \sum_{m=1}^M d_m \quad (7.3.13)$$

Given the high-dimensional situational descriptor in Equation 7.3.5, a simple negative Euclidean distance would not probably be the best tool to define a notion of similarity between situation vectors. For this reason, a weighted distance is introduced in Equation 7.3.11, which adds the similarity metrics $\mathbf{w} = (w_1, \dots, w_s)$ that are organized in the diagonal matrix shown in Equation 7.3.12 and are being learned to extract the most relevant situational features. The width of the Gaussian kernel σ is given by Equation 7.3.13 and is a multiple of the average distance $\bar{d}$ on the first M neighbours to a given situation f (Lowe, 1995). Usually some of the neighbours are used to compute the average distance (typically $M = N/2$) and there is a multiplicative parameter r that is learned together with the similarity metrics $\mathbf{w}$. Its initial value is typically set to 0.6 to place the average distance near the steepest slope of the kernel (see Figure 7.3). A fixed-width Gaussian kernel at each neighbour could be either too small to achieve averaging data points when they are sparse or too big to achieve locality when data points are dense (Lowe, 1995). Averaging the distance on the first M neighbours provides more stability to the measure of distance on changing similarity metrics. Learning the parameter r allows to find an optimal trade-off between averaging and locality in accordance to the given datasets (Lowe, 1995).

Similarly to the *trajectory prediction* method presented by Jetchev and Toussaint (2009), a locally weighted

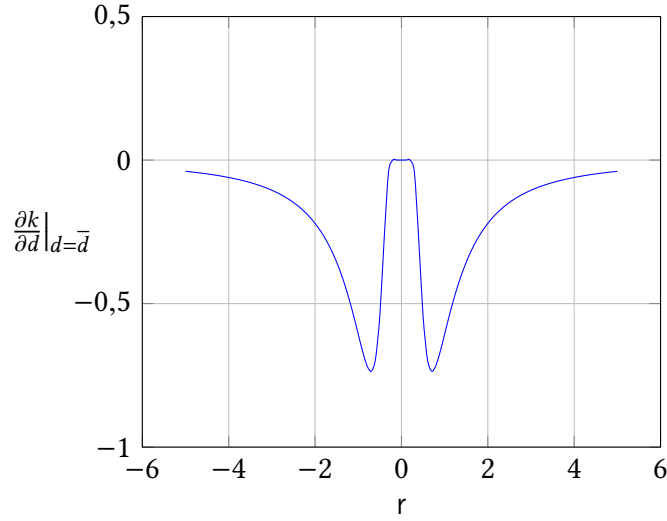


Figure 7.3: First derivative of the kernel computed at the mean distance $\bar{d}$ versus r . With values of r close to ± 0.6 , the slope of the Gaussian kernel is near its maximum negative value.

regressor is defined in Equation 7.3.14.

$$\mathbf{q}_{init}(\mathbf{f}) = \frac{\sum_{i=1}^N k(\mathbf{f}, \mathbf{f}_i) \mathbf{q}_i}{\sum_{i=1}^N k(\mathbf{f}, \mathbf{f}_i)}, (\mathbf{f}_i, \mathbf{q}_i) \in D_{cold} \quad (7.3.14)$$

Given a situation $\mathbf{f}$, the LWR predictor in Equation 7.3.14 averages the N nearest neighbours in the joint space of the manipulator to provide an efficient warm-start initialization for the selection of a human-like arm target posture.

7.3.5 Training

The parameters r and $\mathbf{w} = (w_1, \dots, w_s)$ are trained by taking into consideration the refinement cost in Equation 7.3.7 as an index of initialization quality. The probability of having an initialization $\mathbf{q}_i \in D_{cold}$ given a situation $\mathbf{f}_j \in D_j$ is formulated in Equation 7.3.15. Consequently, the expected value of the refinement cost F on the situation $\mathbf{f}_j$ starting from the initialization $\mathbf{q}_i$ can be computed (Equation 7.3.16).

$$P(\mathbf{q}_i | \mathbf{f}_j) = \frac{1}{Z} k(\mathbf{f}_j, \mathbf{f}_i), Z = \sum_{i=1}^N k(\mathbf{f}_j, \mathbf{f}_i) \quad (7.3.15)$$

$$E\{F(\mathbf{f}_j, \mathbf{q}_i)\} = \sum_{i=1}^N P(\mathbf{q}_i | \mathbf{f}_j) F(\mathbf{f}_j, \mathbf{q}_i) \quad (7.3.16)$$

In Equations 7.3.15 and 7.3.16 N is the number of nearest neighbours and Z is a normalizing constant. The goal of the learning technique is to train the parameters r and $\mathbf{w} = (w_1, \dots, w_s)$ by minimizing

the expected value in Equation 7.3.16 for each situation $f_j \in D_j$. For this reason, the function in Equation 7.3.17 is proposed as the loss on the data collected in the datasets D_{cold} and D_{warm} (Equations 7.3.6 and 7.3.8). The first derivative of the loss with respect to the similarity metrics and the parameter r are given by Equations 7.3.18 and 7.3.19, respectively.

$$L(\mathbf{W}, r) = \frac{1}{|D_j|} \sum_{f_j \in D_j} E\{F(f_j, \mathbf{q}_i)\} + \lambda_w^2 \sum_{k=1}^s \ln^2\left(\frac{w_k}{w_{0k}}\right) + \lambda_r^2 \ln^2\left(\frac{r^2}{r_0^2}\right) \quad (7.3.17)$$

$$\frac{\partial L}{\partial w_k} = \frac{1}{|D_j|} \sum_{f_j \in D_j} \frac{\partial E\{F(f_j, \mathbf{q}_i)\}}{\partial w_k} + 2 \frac{\lambda_w^2}{w_k} \ln\left(\frac{w_k}{w_{0k}}\right) \quad (7.3.18)$$

$$\frac{\partial L}{\partial r} = \frac{1}{|D_j|} \sum_{f_j \in D_j} \frac{\partial E\{F(f_j, \mathbf{q}_i)\}}{\partial r} + 4 \frac{\lambda_r^2}{r} \ln\left(\frac{r^2}{r_0^2}\right) \quad (7.3.19)$$

The first regularizing term of the loss function L corresponds to the regularization introduced by Lowe (1995) and affects the search of the similarity metrics $\mathbf{w}$, which are guided to stay as close as possible to their initial values $\mathbf{w}_0$. The first session of the online learning paradigm sets the initial parameters as $\mathbf{w}_0 = (1, 1, \dots, 1)$ as each feature is assumed to be of equal relevance. However, with further sessions (cycles) of online experiential learning, the selections of the similarity metrics starts with an initial $\mathbf{w}_0$ that corresponds to the solution of the antecedent optimized session. In this manner, a task-related feature extraction is obtained incrementally on cyclically updated datasets. The second regularizing term concerns the parameter r that is constrained to maintain a squared value close to the squared value of $r_0 = 0.6$. This restriction avoids degenerative kernels in which r is either too close to 0 or too big to appreciate any difference in distancing from the selected neighbours (see Figure 7.3).

The first derivatives of the expected value of the refinement cost are taken from Lowe (1995) and reported in Equations 7.3.20–7.3.22 with respect to the parameters $\mathbf{w}$. Similarly, Equations 7.3.23–7.3.25 introduce the first derivatives of the expected refinement cost with respect to the parameter r .

$$\frac{\partial E\{F(f_j, \mathbf{q}_i)\}}{\partial w_k} = \sum_{i=1}^N \frac{\partial P(\mathbf{q}_i | f_j)}{\partial w_k} F(f_j, \mathbf{q}_i) \quad (7.3.20)$$

$$\frac{\partial P(\mathbf{q}_i | f_j)}{\partial w_k} = \frac{1}{Z^2} \left(Z \frac{\partial k(f_j, f_i)}{\partial w_k} - k(f_j, f_i) \sum_{i=1}^N \frac{\partial k(f_j, f_i)}{\partial w_k} \right) \quad (7.3.21)$$

$$\frac{\partial k(f_j, f_i)}{\partial w_k} = -\frac{k(f_j, f_i) w_k}{\sigma^2} \left((f_{jk} - f_{ik})^2 - \frac{r d_{ji}^2}{M \sigma} \sum_{m=1}^M \frac{(f_{jk} - f_{mk})^2}{d_{jm}} \right) \quad (7.3.22)$$

$$\frac{\partial E \{F(f_j, q_i)\}}{\partial r} = \sum_{i=1}^N \frac{\partial P(q_i | f_j)}{\partial r} F(f_j, q_i) \quad (7.3.23)$$

$$\frac{\partial P(q_i | f_j)}{\partial r} = \frac{1}{Z^2} \left(Z \frac{\partial k(f_j, f_i)}{\partial r} - k(f_j, f_i) \sum_{i=1}^N \frac{\partial k(f_j, f_i)}{\partial r} \right) \quad (7.3.24)$$

$$\frac{\partial k(f_j, f_i)}{\partial r} = \frac{k(f_j, f_i) d_{ji}^2}{r \sigma^2} \quad (7.3.25)$$

7.3.6 Online experiential learning

The various steps of the proposed adaptive human-like arm posture learner can be integrated onto the human experiential learning cycle illustrated in Figure 7.1. This integration is summarized and illustrated in Figure 7.4. In specific, the phase of *concrete experience* is deployed by a continuous updating of the

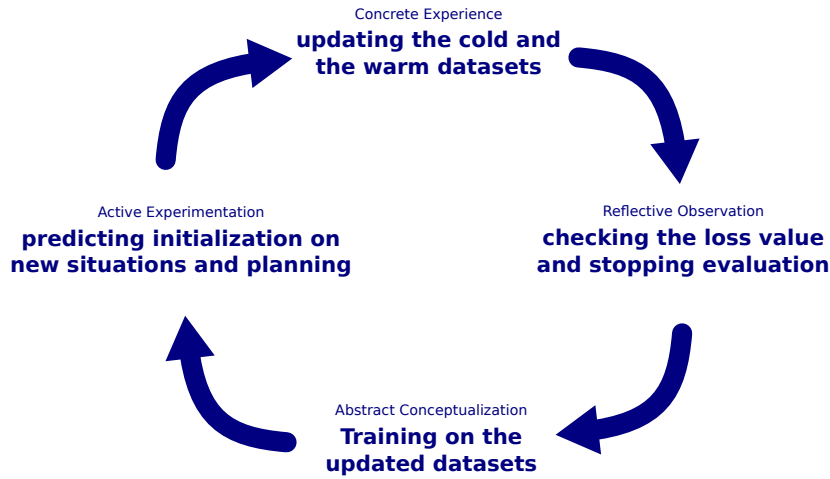


Figure 7.4: A cyclic session of online experiential learning.

cold and the warm datasets D_{cold} and D_{warm} (Equations 7.3.6 and 7.3.8) by the explicit application of a time weighted forgetting policy. The experience of the planning problem instances is stored in two type of constantly-sized memory relating cold and warm initializations, which encode how the convergence speed of the optimizer varies with situations. The experience is then ready for reviewing in the phase of *reflective observation*. A reflective observation on the collected experience can be done by a simple check on the un-regularized loss value of the function L (Equation 7.3.17) with the learning parameters r and $\mathbf{w}$ at disposal on that specific session. At this point of the cycle, a stopping evaluation criterion can also be taken into consideration to check, for example, whether some requirements on the loss function

value are respected. The learning from the collected experience is performed on the stage of *abstract conceptualization* where the concepts characterizing the datasets are learned. In particular, the parameter r is learned starting from a guess of $r_0 = 0.6$, while the similarity metrics $\mathbf{w}$ are incrementally trained starting from the previously learned metrics. After the training phase, the performance of the learner can be tested by an *active experimentation* on new and unseen planning problem situations. Specifically, warm initializations are predicted and provided to the optimizer IPOPT by the locally weighted regressor reported in Equation 7.3.14. The results of this experimentation are collected for future *reflective observation* and to integrate the *concrete experience* for a subsequent learning session.

7.4 Experiments and results

In this section, details concerning the implementation of the adaptive learning approach described in section 7.3 are given and results of different tests are reported. The reaching scenario illustrated in Figure 7.5 is taken into consideration as an exemplar planning problem instance with past motion experience, which is used for learning good initializations for the selection of human-like arm target postures. In this scenario

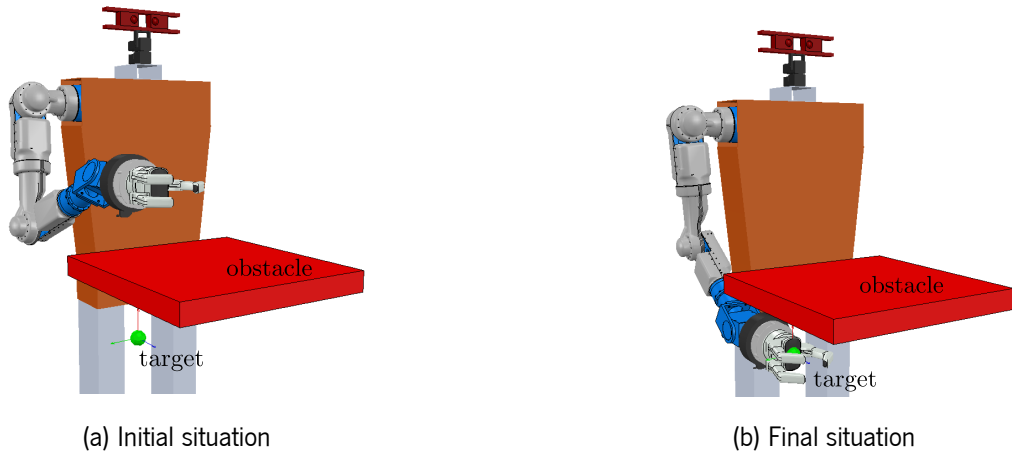


Figure 7.5: Illustration of one example of the considered reaching task. The manipulator has to reach the green target while avoiding collisions with the red obstacle. The proposed adaptive learning technique predicts efficient initial guesses for the selection of a human-like target configuration of the robotic arm.

there is one big red obstacle ($n_{obst} = 1$) between the right upper-limb of the robot ARoS (Figure 4.1a) and a green target for the end-effector. The total number of points considered for the formulation of a situation descriptor $\mathbf{f}$ (Equation 7.3.5) is $n_{arm} = 12$, which corresponds to 3 points on the arm (right elbow, wrist and hand) and to 9 points on the three-fingered Barrett Hand (3 points on each finger). Further details on the positions of these points are given in sections A.1 and A.3.1. Then, the scenario under examination counts for $s = 223$ as the total number of features. In order to simulate the manifestation of

several “similar” situations, the initial configuration of the robot, the pose of the obstacle and of the target are changed by randomly selecting different situations on a uniform distribution of values centred on the initial situation illustrated in Figure 7.5a. The refinement cost introduced by Equation 7.3.7 is obtained on 20 IPOPT iterations ($j = 20$) for the selection of an optimal target posture like the one illustrated in Figure 7.5b. The numerical results were obtained using an Intel(R) Core(TM) i7-4770 CPU 3.40GHz running Linux Mint 17 64-bits with a AMD Radeon HD 6570 video card and 8GB of RAM Memory. In order to improve the performance of the proposed learning approach, data are preprocessed by scaling them robustly around their median and clipping the maximum refinement cost to the value of 1. On the same datasets the performances of two methods are compared:

- LWR_{Opt} , which implements the locally weighted regressor given by Equation 7.3.14 with trained similarity metrics $\mathbf{W}$ and parameter r .
- LWR_{Eucl} , which implements the locally weighted regressor given by Equation 7.3.14 with Euclidean similarity metrics, i.e. $\mathbf{W} = \text{diag}(1, 1, \dots, 1)$ and Gaussian kernel width $\sigma = 1$.

Additionally, initializations $\mathbf{q}_i \in D_{cold}$ are randomly picked for a warm start-up in order to understand to what extent the introduction of similarity functions improves the performance of planning human-like arm target postures. The setup of the modules of software that are employed during the experiments is described in section 7.4.1. Then, results are reported in section 7.4.2.

7.4.1 Software setup

The setup of the software for experimentation is illustrated in Figure 7.6. The modular setup showed in Figure 6.2 is augmented with the functionalities of a motion learner, which implements the proposed on-line adaptive learning system. The Human-like Upper-limb Posture Learner (HUPL) is a Python library¹ that can utilize the experience collected and updated by the motion manager to learn a task-related abstract conceptualization from it. In specific, the proposed method LWR_{Opt} minimizes the loss function in Equation 7.3.17 by applying the Quasi-Newton limited-memory BGFS optimization method (Nocedal and Wright, 1999), which is implemented and provided by the SciPy library (Virtanen et al., 2020). The regularization parameters in Equation 7.3.17 are selected to be $\lambda_w = 0.04$ and $\lambda_r = 0.1$. The chosen number of nearest neighbours for prediction is $N = 10$ for both the predictors LWR_{Opt} and LWR_{Eucl} , which means setting $M = 5$ on averaging distances in the computation of the Gaussian kernel width (Equation 7.3.13). Concerning the online updating of the datasets D_{cold} (Equation 7.3.6) and D_{warm} (Equation 7.3.8), the forgetting rate is $\tau = 0.95$ and the threshold is $\theta = 0.01$, which produce a constant temporal window on

¹<https://github.com/zohannn/HUPL>

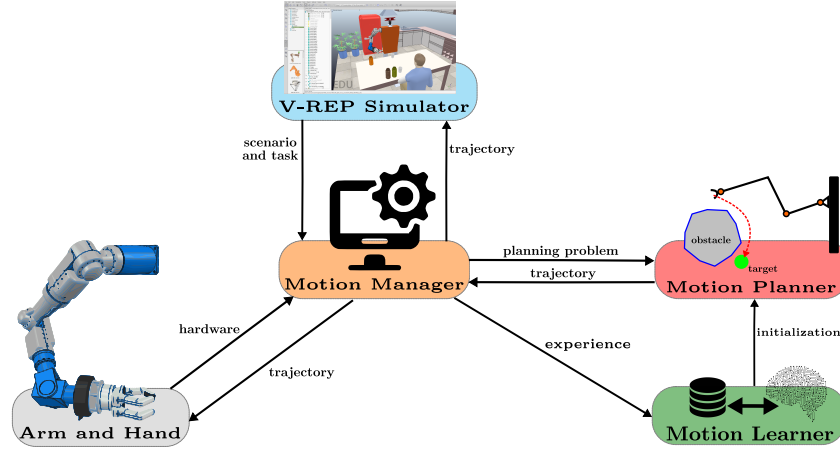


Figure 7.6: Sketch of the software setup that includes online cyclic experiential learning. The Motion Manager module collects and updates the datasets necessary to build a concrete experience. The Motion Learner module applies memory-based learning on these experience to predict efficient initializations of the Motion Planner module. When trajectories are successfully planned, pairs of situations and solutions are added to the present experience and a new learning cycle can start.

the data stream of 90 samples for each dataset. After preliminary tests, this size of the training datasets revealed to be sufficient to provide acceptable minimum values of the loss function in Equation 7.3.17, which was validated on situations excluded from the training.

7.4.2 Results

The first test on the scenario illustrated in Figure 7.5 concerns the utilization of the memory size as a limited resource of the proposed memory-based learning method. The application of a time weighted forgetting (TWF) algorithm (Salganicoff, 1993b) with a forgetting rate $\tau = 0.95$ and a threshold $\theta = 0.01$ restricts the size of the datasets D_{cold} and D_{warm} to a maximum of 90 exemplars (Figure 7.7). Figure 7.7 shows that the number of data samples increases linearly until the first 100 novel situations and, then, old samples are being deleted as their weights assume values smaller than the given threshold. This collecting and updating strategy guarantees an efficient utilization of the memory size and privileges the most recent situations that are being experienced by the robot.

Once the datasets D_{cold} and D_{warm} are constructed with initial data representative of the given situations, online cyclic sessions of learning can be applied according to the steps illustrated in Figure 7.4. A test of 10 training sessions with 10 new situations each is performed to learn the similarity metrics of the LWR_{Opt} predictor in “similar” environments, which are uniformly sampled around the one in Figure 7.5. The obtained un-regularized loss function values over the sessions are compared with the loss values computed with a LWR_{Eucl} predictor and are illustrated in Figure 7.8. Over the session, the values of the

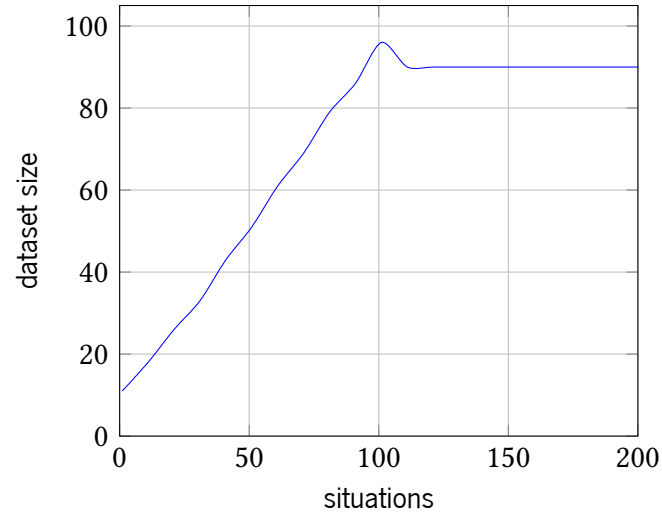


Figure 7.7: Variation of the size of the datasets D_{cold} and D_{warm} with new incoming situations of the environment.

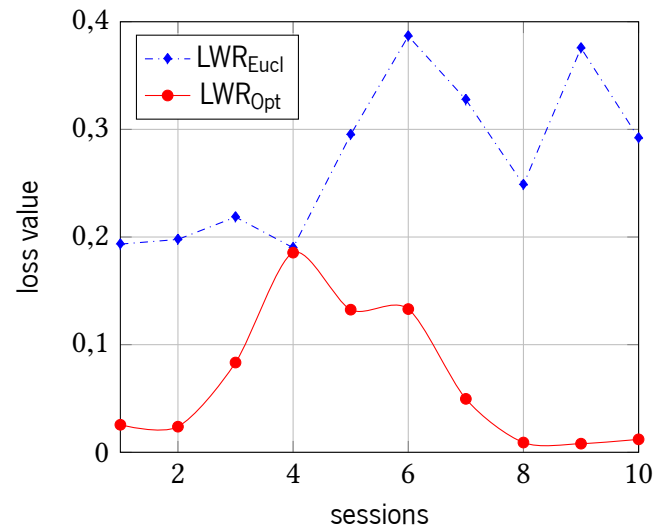


Figure 7.8: Un-regularized loss values (Equation 7.3.17) of the LWR_{Eucl} and of the LWR_{Opt} predictors on 10 sessions of on-line learning. Each session is composed by 10 new situations that update the datasets before the training starts.

trained loss of the LWR_{Opt} predictor remain substantially smaller than the losses computed on the datasets by using an Euclidean-based similarity metric (Figure 7.8). In the first sessions, with few novel situations in the datasets to select as nearest neighbours, the LWR_{Eucl} loss remains basically constant. Due to the poor feature extraction on the geometrical situation descriptor, the LWR_{Opt} loss increases, but its value is limited by a small r parameter that reduces the width of the Gaussian kernel in Equation 7.3.10. The training usually ends with values of ± 0.3 for the parameter r indicating the presence of dense and noisy data in memory (Lowe, 1995). Therefore, costly situations may result very dissimilar even with a poor feature extraction of the parameters $\mathbf{W}$ (Equation 7.3.12). However, as more novel situations occupy the memory, the loss increases without a proper feature selection and, therefore, the introduction of a variable kernel would not significantly improve the performance of prediction. With more sessions of online training the LWR_{Opt} loss decreases towards values very close to 0, while the LWR_{Eucl} loss assumes oscillatory values that significantly depend on the data in memory (Figure 7.8). The evolution of the weights $\mathbf{W}$ (Equation 7.3.12), which are incrementally learnt over the sessions, is shown in Figure 7.9. After one session, some of the most significant features of the descriptor in Equation 7.3.5 show immediately their relevance for a low-cost selection of a human-like arm target posture (Figure 7.9a). Session by session, some metrics of similarity increase of magnitude and may change their ranking positions in favour of features that would not have emerged with sufficient learning cycles (Figures 7.9b and 7.9c). After 10 sessions (see Figure 7.9d), it is clear that the positions of the target and of the obstacle (especially along the $\hat{z}$ axis) are principally crucial to obtain high-quality solutions of the optimization problem (5.1.4)–(5.1.7) in solving the task in Figure 7.5. When the value of the loss function does not encounter significant reductions and reach a stable value, the online cyclic training can be stopped and the learned similarity metrics can be used to predict high-performance initializations at query time in presence of novel situations. The trained metrics are tested and their performance is compared with the results obtained by the application of the LWR_{Eucl} predictor and by randomly uniformly sampling the solutions stored in the dataset D_{cold} to initialize the target posture selection (5.1.4)–(5.1.7). The statistics on 500 situations of this comparison are reported in Table 7.1. The

	LWR_{Opt}	LWR_{Eucl}	Random
Mean of iterations	255.92	320.6	300.405
Mean of CPU time [s]	0.765537	1.05404	0.960806
Mean of error	8696.79	30619.7	20323
Median of $\ \mathbf{q}_* - \mathbf{q}_{init}\ $	17.0768	37.8946	24.6169

Table 7.1: Performance of predicting initializations to the optimization problem (5.1.4)–(5.1.7) on 500 situations uniformly samples from the task in Figure 7.5. The optimizer IPOPT is set to work in warm-start mode with a maximum number of iterations of 5000.

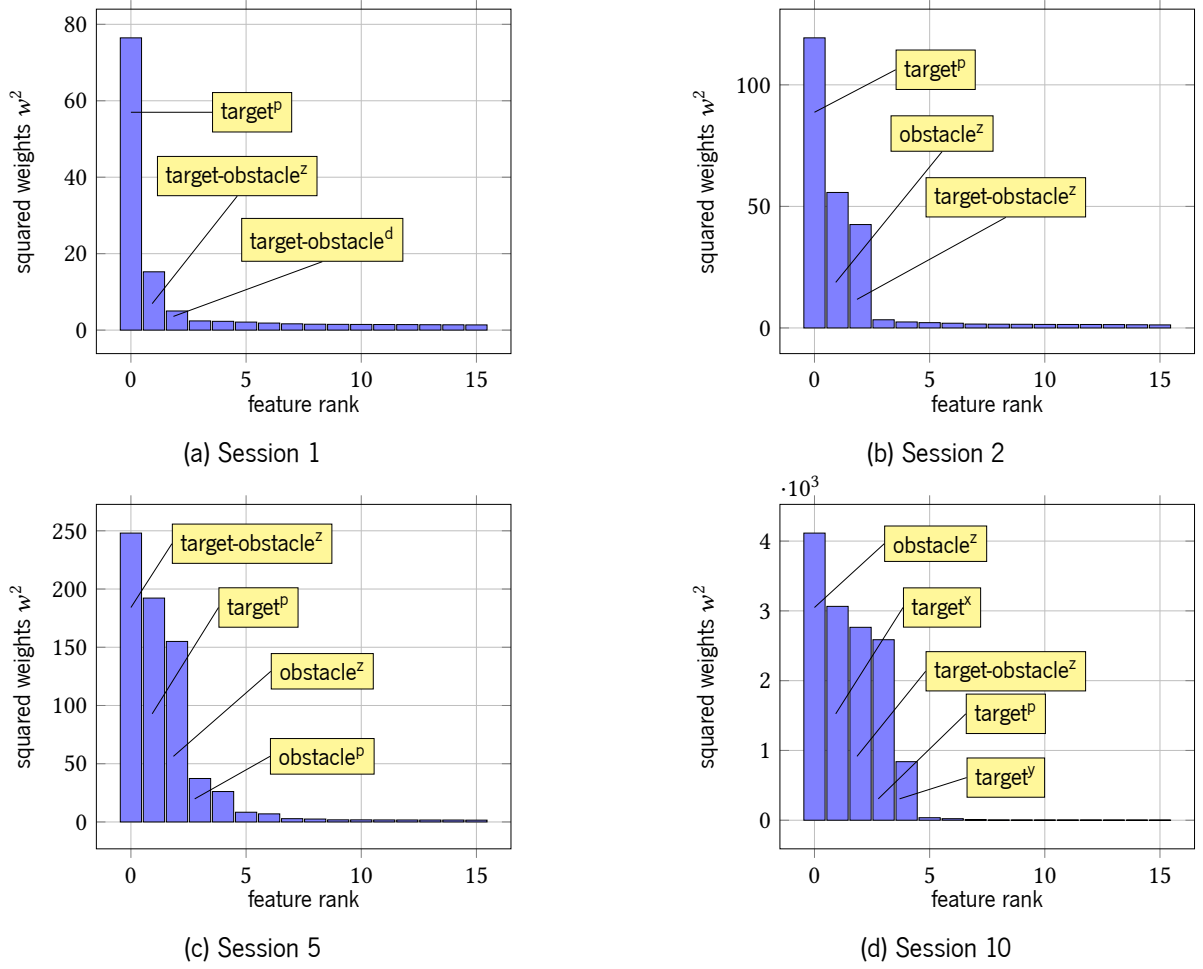


Figure 7.9: The most 15 significant learned features for the task in Figure 7.5. Incremental feature extraction over 10 sessions of online experiential learning with 10 new situations per session. In superscript the component of the element is written: $x = \hat{x}$ axis, $y = \hat{y}$ axis, $z = \hat{z}$ axis, d is the L_2 norm, $r = \text{roll}$ angle, $p = \text{pitch}$ angle and $y = \text{yaw}$ angle.

analysis of the data in Table 7.1 shows that the application of the proposed learning approach on average saves iterations and CPU time compared to the other more classical methods of predictions. Additionally, the total error returned by IPOPT indicates the supply of solutions of better quality. Interestingly, the LWR_{Opt} predictions tend to stay on the median of the distribution of the solutions. This latter result demonstrates that, even though feeding close-solutions initializations to interior-point methods is not certainly the best warm-start strategy (Shahzad, 2010), predicting low-cost initializations also means predicting initializations that lay on a small-distance basin from the corresponding solutions. It is also noteworthy the fact that the LWR_{Eucl} predictor is generally performing worse than the random predictor (Table 7.1). With a greater number of samples not uniformly distributed, this result might be opposite.

For a better understanding of the forgetting-based datasets update, a test on more 10 sessions with 10

new situations each is experimented. Similarly, situations of the type illustrated in Figure 7.5 are randomly uniformly selected. However, in this test the initial configuration of the robot remains constant over the situations, which means that only the target and the obstacle poses (position and orientation) change. Session by session, the loss function values of the LWR_{Eucl} and the LWR_{Opt} predictors are monitored and no training is executed. The results are collected and illustrated in Figure 7.10. In the first sessions, there is a

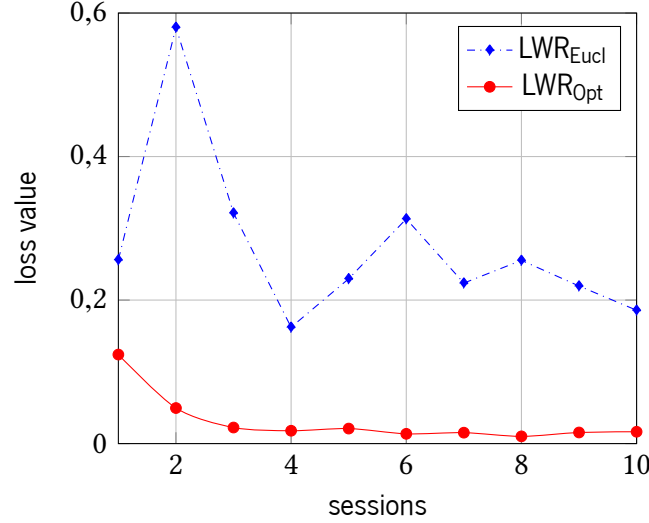


Figure 7.10: Unregularized loss values (Equation 7.3.17) of the LWR_{Eucl} and of the LWR_{Opt} predictors on 10 sessions of on-line learning. Each session is composed by 10 new situations that update the datasets before the training starts. The new situations differ from the others only for the target and the obstacle poses. The initial configuration of the robot remains constant.

small number of the novel situation with constant arm configurations compared to the size of the datasets. This fact implies an inevitable increase of the LWR_{Opt} loss value, which is comparable and slightly smaller than the LWR_{Eucl} loss value. Over the sessions, old situations are gradually forgotten and deleted from the datasets D_{cold} and D_{warm} to be replaced with the new situations that the robot is experiencing. This consequence of the forgetting algorithm is observable on the gradual decrease of the LWR_{Opt} loss and its stabilization to a value close to 0. On the contrary, the LWR_{Eucl} loss seems to reach a stable oscillation around the value 0.2. Therefore, it can be deduced that, after an initial degradation of the performance, the LWR_{Opt} predictor well adapts to the new datasets by utilizing the previously trained similarity metrics to compute the measure of distance. Although a different forgetting algorithm was studied, this outcome is basically coherent with the experimental results obtained by Salganicoff (1993a). After 10 sessions, the datasets D_{cold} and D_{warm} are totally populated by the new situations with a constant configuration of the robot. With this new data stored in memory, tests on the considered predictors with 500 similar situations were also carried on and the statistics of the results are collected in Table 7.2. As was expected, due to the geometrical similarity of the data, the results in Table 7.2 are more proximate than those ones

	LWR_{Opt}	LWR_{Eucl}	Random
Mean of iterations	161.925	190.65	165.78
Mean of CPU time [s]	0.326972	0.362131	0.325854
Mean of error	2748.58	6273.91	10315
Median of $\ q_* - q_{init}\ $	15.4763	15.7992	23.4537

Table 7.2: Performance of predicting initializations to the optimization problem (5.1.4)–(5.1.7) on 500 situations uniformly samples from the task in Figure 7.5. The initial configuration of the robot remains constant on the different situations. The optimizer IPOPT is set to work in warm-start mode with a maximum number of iterations of 5000.

in Table 7.1. The proposed LWR_{Opt} predictor generally performs better than the others, except for the average CPU time that is very close to that resulting from random predictions on previous stored solutions. However, the average error returned by IPOPT on the optimization of the problem (5.1.4)–(5.1.7) is much smaller when the proposed predictor is applied, which means that the quality of the returned solutions is higher when low-cost initializations are learned from experience. Interestingly, both the LWR_{Opt} and LWR_{Eucl} predictors provide initializations similarly close to the median of the distributions of solutions. However, the significant difference between the corresponding mean errors confirms that predicting a solution (not a low-cost initialization) to warm start an interior-point method is not a sufficient strategy when “nearby” optimization problems are addressed.

In order to assess the fact that a successful abstraction of the data concepts was accomplished by the proposed learning method, the trained parameters W and r are tested on a new dataset of situations randomly uniformly selected from that one showed in Figure 7.11. In specific, the initial configuration of the robot (Figure 7.11a) is different than that one in Figure 7.5a. Moreover, the final situation (Figure 7.11b) is featured by a position and orientation of the target significantly different than that on illustrated in Figure 7.5b. The pose information of the red obstacle remain equal to the previous ones. Considering this latter scenario, the performance of the predictors is compared on 500 new situations and the statistics of the results are given by Table 7.3. It can be observed that the proposed LWR_{Opt} predictor outperforms the others in terms of average number of iterations, average CPU time and average error returned by the optimizer IPOPT. Similarly to the statistics concerning the previous reaching task (Table 7.1), LWR_{Eucl} predictor performed worse than the random predictor and the low-cost initializations are also the statistically closer ones to their corresponding solutions. This latter set of experiments demonstrated that the feature extraction illustrated in Figure 7.9 might remain valid and efficient as long as the datasets are kept updated with the most “relevant” (recent) situations that occur in task-changing environments. In these tests, re-training sessions were not necessary, but the proposed learning paradigm illustrated in Figure 7.4 is

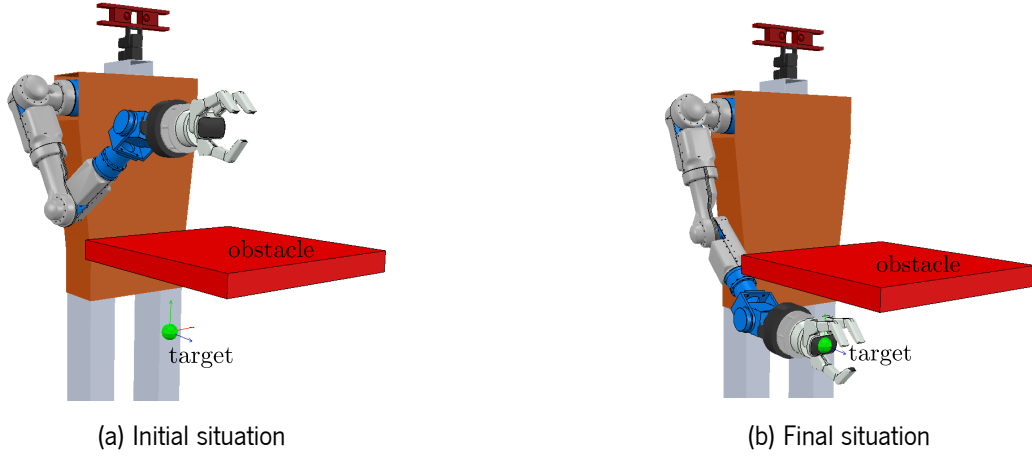


Figure 7.11: Illustration of one example of the considered reaching task. The manipulator has to reach the green target while avoiding collisions with the red obstacle. The main differences with the reaching task in Figure 7.5 refer to the initial configuration of the manipulator and to position and the orientation of the target.

	LWR_{Opt}	LWR_{Euc}	Random
Mean of iterations	243.488	500.58	287.668
Mean of CPU time [s]	0.435206	0.923416	0.536539
Mean of error	34745.3	58302.5	38690.6
Median of $\ q_* - q_{init}\ $	73.5271	14638.4	109.124

Table 7.3: Performance of predicting initializations to the optimization problem (5.1.4)–(5.1.7) on 500 situations uniformly samples from the task in Figure 7.11. The optimizer IPOPT is set to work in warm-start mode with a maximum number of iterations of 5000.

designed for an incremental training on *abstract conceptualization* when the stage of *reflective observation* detects a degradation of the performance. Generally speaking, as it was also pointed out by Salganicoff (1993a), the application of a forgetting policy enables the adaptive capabilities of a learning system, but, on the contrary, may also risk to decrease the predicting performances when situations jump rapidly from a task to another in the same environment. This observation clearly arises the need of reaching a trade-off between fast adaptation and the desired learning performance when tuning the forgetting parameters τ and θ (Salganicoff, 1993a).

Part IV

Controlling

Chapter 8

Tracking human-like upper-limb motion in the operational space

In this chapter, a system for tracking antecedently planned human-like upper-limb trajectory is described. These trajectories are generated as described in chapter 5 in accordance with the information extracted at query time from the workspace of a robot. However, unpredicted modifications of the scenario might occur during the execution of a movement, which is then required to promptly adapt and maintain the provided human-like features as possible. While joints-space controllers are proposed to track the motion of the fingers, transferring onto the operational-space of manipulation is needed to preserve the kinematic human-like features of the end-effector. Additionally, tracking human-like swivel angle motion is proposed as a second-priority task in combination with online obstacle avoidance, maximization of manipulability and respect of the physical joint limits. A temporal coupling of the controllers is also designed to synchronize the different desired movements and offer the possibility of a temporal scaling. section 8.1 introduces the principal notions of feedback loop control and how it is addressed to generate human-like motion in robotics. In section 8.2, the transfer from a given human-like joints-space trajectory to a reference in the operational space of manipulation is described. section 8.3 introduces the proposed second-order tracking algorithms, while section 8.4 shows the results of different tests with the right arm-hand system of humanoid robot ARoS (Figure 4.1a).

8.1 Introduction

When a new action takes place, the human brain possesses complete specifications of a trajectory before the execution of the motion. However, there is also a constant redefinition of these specifications while different body parts are voluntarily moving (Schwartz, 2016). These evidences suggest a tight collaboration

between independent and interconnected processes of planning and controlling (Burdet et al., 2013). For this reason, in accordance with the model illustrated in Figure 8.1, this chapter introduces the design of feedback loop controllers for tracking human-like upper-limb trajectories, which are provided as references by the motion planner described in chapter 5. The redundancy of a 7-DOFs robotic manipulator is used

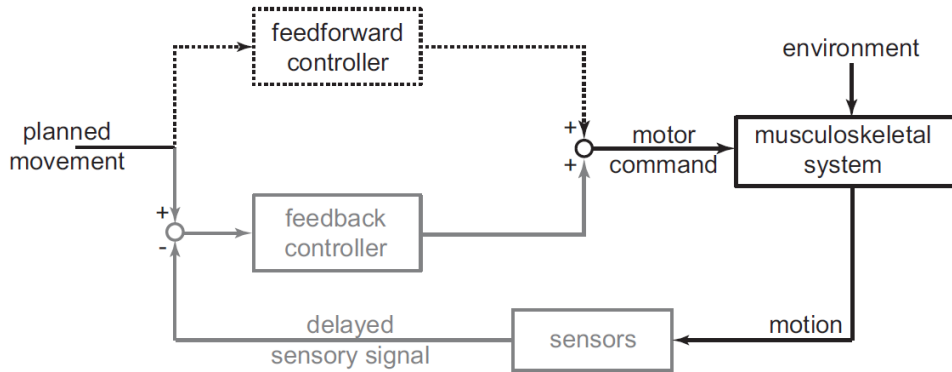


Figure 8.1: Picture from Burdet et al., 2013 illustrating the schematic of human motor control.

to satisfy specific constraints (e.g., online obstacle avoidance) and is also solved by maintaining typical human-like features of a given planned trajectory. The proposed laws of control act to reduce errors in the operational space of manipulation because tracking in the joints-space would not allow to cope with unpredicted changes of the environment and, more importantly, would not permit to utilize the redundancy for customized purposes (Chiaverini et al., 1991; Liu et al., 2010). In the recent literature, the *swivel angle* of human-like robotic manipulators has been widely adopted to encode the biological characteristics of human reaching movements and uniquely solve the redundancy problem (see e.g. Hyunchul Kim et al., 2011; Su et al., 2018; Zanchettin et al., 2013). The *swivel angle* (also called *elbow angle*) can be defined as the angle between the sagittal body plane of motion and the plane projected by the triangle wrist-elbow-shoulder and it is a characteristic factor of human arm postures (Kreutz-Delgado et al., 1990; Zanchettin et al., 2011). For instance, Hyunchul Kim et al. (2011) presented a technique for resolving the redundancy of 7-DOFs wearable robotic manipulators that is based on the estimation of the human swivel angle in daily reaching tasks. The relative position of the elbow is expressed as a function of the swivel angle and allows to compute a straightforward and unique solution to the inverse kinematic problem. The estimation of the swivel angle is applied by considering the hypothesis that biological movement are planned for an efficient retraction of the palm to the head region in the vicinity of the mouth. More specifically, human arms move according an actual target point and a virtual target point placed on the head region for potential retraction. This hypothesis is directly linked with the measure of manipulability and its ellipsoid, which describes how joints-space velocities affect the directions of motion in the operational space (Siciliano et al., 2009). The proposed method assumes that the movement towards the virtual target follows the shortest path

connecting it to the wrist. Then, the swivel angle is selected such that the projection of the major axis of the manipulability ellipsoid onto this path is maximized. The experimental procedure involved several subjects equipped with markers along the right arm and the chest while performing repetitions of different reaching tasks. The results showed an estimation error of less than 5 degrees, which is acceptable in many applications that require solving the inverse kinematics problem in a closed form.

Zanchettin et al. (2013) described a method for extracting a function that relates the swivel angle and the hand pose in human assembly-like arm movements. In particular, the motion of a group of healthy adults was captured in an experimental setup that simulates the movements of the hand during the assemblage of small parts on a work bench. More specifically, a ball was put on a table in front of the subject and it was moved with the right hand palm down. Markers on the shoulder, the elbow and the wrist were placed to record specific positions during the motion. An Extended Kalman Smoother (EKS) was utilized to encode the seven degrees of freedom of the human arm in seven joint angle values from the data collected on the markers. Then, the data were clustered according to the interested portion of the operational space and the standard deviation of the computed swivel angles was considered as an intra-cluster similarity measure. The limited values of this similarity clearly indicated the possibility to express the swivel angle as a function of the hand pose. Since a non-linear relationship was expected, the identification in least-squares minimization was performed with a mixture of non-linear regressors. Considering all the clusters, the root mean squared error of the resulting model was of 9 degrees, which is acceptable for a closed-form solution of the redundancy. Experiments with the dual-arm robot FRIDA (Figure 2.8e) showed the feasibility of the proposed technique and how the inverse kinematics can be extended to two-armed assembly tasks assuming that the left end-effector is always symmetric to the right one.

A human-like redundancy resolution based on the swivel angle estimation was also addressed by Su et al. (2018) for anthropomorphic manipulators applied in teleoperated tasks. Since fast real-time control is required in teleoperation, a given value of the swivel angle permits to solve the inverse kinematic problem of 7-DOFs robotic arms in a closed form. Moreover, a human-like swivel angle allows to reach high levels of motion predictability and of acceptance when interactions with human partners are necessary. The estimation of a human-like swivel angle was based on the acquisition of reaching motion from one single subject to avoid variabilities of the arm lengths. Then, a neural network with twelve neurons in one hidden layer and one neuron in the output layer was trained to fit the function that relates the swivel angle value to given hand poses. After training and validating, the predicted human-like swivel angle is projected onto the feasible space of swivel angles, which respects the physical limits of the joints. Inside the feasible region, small errors are detected when human-like swivel angles are predicted. Experiments on teleoperated tasks demonstrated that the proposed technique could resolve the redundancy of the anthropomorphic manipulator in a way that is more acceptable to human co-workers and that achieves high levels of manipulability

in the operational space.

Even though knowing how human-like swivel angles are generated is essential to solve the redundancy of anthropomorphic manipulators involved in online real-time control tasks, most of the proposed controllers do not consider human-like features that the end-effector should manifest in the execution of more complex movements. As described in chapter 3, specific kinematical behaviours arise in performing pick-and-place human upper-limb movements, which can only be qualitatively imitated by tracking explicit preconfigured operational-space trajectories. It is worth noting that there is no evidence that human-like hand kinematics is correlated with particular swivel angle values, even though the replication of the latter can be obtained as a second-priority task in controlling redundant manipulators. Indeed, specific operational-space trajectories can also be successfully tracked also with non-redundant robotic arms (see, e.g., Caccavale et al., 1997). The techniques of feed-back control introduced in this chapter aim at tracking the human-like trajectories generated by the planner HUMP described in chapter 5. Since applications with anthropomorphic manipulators are addressed, operational-space tracking is accomplished as a first-priority task, while swivel angle tracking and the optimization of custom functions is achieved as a second-priority task by using the redundancy of manipulation. For this purpose, simplified acceleration-based second-order algorithms are used because have demonstrated good tracking capabilities (Nakanishi et al., 2008).

8.2 The desired reference trajectories in the operational space

As described in chapter 5, a human-like upper-limb trajectory is generated in different phases to compose goal-directed movements (Figure 5.2). Each phase is featured by a trajectory in the joints space of the manipulator with proper boundary conditions that guarantee continuity in position and velocity. Tracking in the joints space would ensure the most desirable level of human-likeness, but would not allow the robot to compensate noisy measurements and to promptly react to sudden changes of the workspace, such as variations of the end-effector target pose and the approaching of obstacles during the execution of motion. For this reason, a human-like trajectory is firstly projected onto the operational space so that controlling the end-effector path also permits to use the self-motion of a redundant robotic arm and, more generally, to cope with various online requirements. For robotic arms, the projection of a joints-space state $(\theta, \dot{\theta}, \ddot{\theta})$ onto the operational-space state $(x, \dot{x}, \ddot{x})$ can be simply achieved by applying the rules of direct kinematics reported by the Equations 8.2.1, 8.2.2 and 8.2.3 (Siciliano et al., 2009). For instance, the direct kinematics (the function *kin*() in Equation 8.2.1) of the arms of the humanoid robot ARoS (Figure 4.1a) is described

in details in section A.1.

$$\mathbf{x} = \begin{bmatrix} \mathbf{p} \\ \mathbf{Q} \end{bmatrix} = \mathbf{kin}(\boldsymbol{\theta}) \quad (8.2.1)$$

$$\dot{\mathbf{x}} = \begin{bmatrix} \dot{\mathbf{p}} \\ \dot{\boldsymbol{\omega}} \end{bmatrix} = \mathbf{J}\dot{\boldsymbol{\theta}} \quad (8.2.2)$$

$$\ddot{\mathbf{x}} = \begin{bmatrix} \ddot{\mathbf{p}} \\ \ddot{\boldsymbol{\omega}} \end{bmatrix} = \mathbf{J}\ddot{\boldsymbol{\theta}} + \dot{\mathbf{J}}\dot{\boldsymbol{\theta}} \quad (8.2.3)$$

The position of the end-effector $\mathbf{p} = \{p_x, p_y, p_z\} \in \mathbb{R}^3$ is given by its Cartesian coordinates $\{p_x, p_y, p_z\}$ and its orientation is represented by its unit quaternion $\mathbf{Q} = \{\eta, \boldsymbol{\epsilon}\} \in \mathbb{R}^4$ to avoid singularities of representation (Siciliano et al., 2009). As shown in Equation 8.2.2, linear velocities $\dot{\mathbf{p}} \in \mathbb{R}^3$ and angular velocities $\dot{\boldsymbol{\omega}} \in \mathbb{R}^3$ of the end-effector measured with respect to the world reference frame are computed by considering the geometrical Jacobian $\mathbf{J} \in \mathbb{R}^{6 \times 7}$ of the manipulator. Analogously, linear accelerations $\ddot{\mathbf{p}} \in \mathbb{R}^3$ and angular accelerations $\ddot{\boldsymbol{\omega}} \in \mathbb{R}^3$ are obtained as described by Equation 8.2.3. In order to avoid the inconvenient calculus of the term $\dot{\mathbf{J}}$, numerical differentiation can be applied directly on the velocities $\dot{\mathbf{x}}$ to obtain the accelerations $\ddot{\mathbf{x}}$ ¹. It is worth noting that the derivatives of the end-effector state $\dot{\mathbf{x}}$ and $\ddot{\mathbf{x}}$ are featured by the angular velocity $\boldsymbol{\omega}$ and the angular acceleration $\dot{\boldsymbol{\omega}}$, respectively, instead of the derivatives of the quaternion $\mathbf{Q} = \{\eta, \boldsymbol{\epsilon}\}$. For this reason, it is important considering the relations given by Equations 8.2.4–8.2.9, which refer to a *quaternion propagation* (Siciliano et al., 2009).

$$\dot{\eta} = -\frac{1}{2}(\boldsymbol{\epsilon}^\top \boldsymbol{\omega}) \quad (8.2.4)$$

$$\ddot{\eta} = -\frac{1}{2}(\dot{\boldsymbol{\epsilon}}^\top \boldsymbol{\omega} + \boldsymbol{\epsilon}^\top \dot{\boldsymbol{\omega}}) \quad (8.2.5)$$

$$\dot{\boldsymbol{\epsilon}} = \frac{1}{2}(\eta \boldsymbol{\omega} - \boldsymbol{\epsilon} \times \boldsymbol{\omega}) \quad (8.2.6)$$

$$\ddot{\boldsymbol{\epsilon}} = \frac{1}{2}(\dot{\eta} \boldsymbol{\omega} + \eta \dot{\boldsymbol{\omega}} - \dot{\boldsymbol{\epsilon}} \times \boldsymbol{\omega} - \boldsymbol{\epsilon} \times \dot{\boldsymbol{\omega}}) \quad (8.2.7)$$

$$\boldsymbol{\omega} = 2(\boldsymbol{\epsilon} \times \dot{\boldsymbol{\epsilon}} + \eta \dot{\boldsymbol{\epsilon}} - \dot{\eta} \boldsymbol{\epsilon}) \quad (8.2.8)$$

$$\dot{\boldsymbol{\omega}} = 2(\boldsymbol{\epsilon} \times \ddot{\boldsymbol{\epsilon}} + \eta \ddot{\boldsymbol{\epsilon}} - \dot{\eta} \dot{\boldsymbol{\epsilon}}) \quad (8.2.9)$$

Being essential for the composition of a complete human-like upper-limb path, the optimal target and bounce postures (see sections 5.1.1 and 5.1.2.3) are translated to their counterparts in the operational

¹The numerical differentiations implemented in the control system are robust against noise and are given by the smooth differentiators introduced at <http://www.holoborodko.com/pavel/numerical-methods/numerical-derivative/smooth-low-noise-differentiators/>

space of an anthropomorphic manipulator. While a bounce hand state pose $\mathbf{x}_b = \mathbf{kin}(\boldsymbol{\theta}_{b,arm})$ is computed through direct kinematics from the optimal bounce arm posture $\boldsymbol{\theta}_{b,arm}$ (section 5.1.2.3), a target hand state pose $\mathbf{x}_{tar}$ cannot, in general, be computed from a selected target arm posture $\boldsymbol{\theta}_{tar,arm}$ (section 5.1.1). Otherwise, a manipulator could not cope with sudden changes of its end-effector target pose during the execution of a movement. For example, if an object that has to be picked changes position as a consequence of the interaction with a human partner, the robot needs to compensate this perturbation by tracking the object on its new location. Therefore, a target hand pose $\mathbf{x}_{tar}$ is selected to be the desired hand pose in each phase of a movement ($\mathbf{P}_{tar}$ in Equation 5.1.5). In absence of unpredicted perturbations of the workspace, this strategy would ensure avoidance of collisions without the application of online obstacle avoidance mechanisms and would promote target hand poses $\mathbf{x}_{tar}$ in correspondence with planned human-like target arm postures $\boldsymbol{\theta}_{tar,arm}$. In order to achieve this result, the desired reference operational-space trajectories $(\mathbf{X}, \dot{\mathbf{X}}, \ddot{\mathbf{X}})$ resemble their joints-space counterparts $(\mathcal{T}, \dot{\mathcal{T}}, \ddot{\mathcal{T}})$ by taking into consideration a bounce hand pose $\mathbf{x}_b$ instead of a bounce arm posture $\boldsymbol{\theta}_{b,arm}$ and target hand poses $\mathbf{x}_{tar}$ instead of target arm postures $\boldsymbol{\theta}_{tar,arm}$. In particular, a desired *Transport* phase reference trajectory (Equation 8.2.10) is featured by a target hand pose $\mathbf{x}_{tran}$ and a bounce hand pose $\mathbf{x}_b$. Similarly, desired *Approach* and *Retreat* phase reference trajectories are featured by target hand poses $\mathbf{x}_{app}$ and $\mathbf{x}_{ret}$, respectively (see Equations 8.2.11 and 8.2.12).

$$\mathbf{x}_d(\mathbf{x}_{tran}, \mathbf{x}_b) = \mathcal{X}_{tran}^{direct}(\mathbf{x}_{tran}) + \mathcal{X}_{tran}^{bkfor}(\mathbf{x}_b) \quad (8.2.10)$$

$$\mathbf{x}_d(\mathbf{x}_{app}) = \mathcal{X}_{app}(\mathbf{x}_{app}) \quad (8.2.11)$$

$$\mathbf{x}_d(\mathbf{x}_{ret}) = \mathcal{X}_{ret}(\mathbf{x}_{ret}) \quad (8.2.12)$$

Analogously to a planned human-like joints-space trajectory, a desired *Transport* phase path in the operational space of a manipulator is resulting from the composition of a *direct* and a *back-and-forth* human-like movements. The former transports the end-effector from its initial state $\mathbf{x}_0$ towards its target state $\mathbf{x}_{tran}$, while the latter ensures collisions avoidance of the end-effector in a static workspace. The positions, velocities and accelerations of the desired operational-space *direct* motion are respectively reported in Equations 8.2.13, 8.2.14 and 8.2.15. Equations 8.2.16, 8.2.17 and 8.2.18 provide, respectively, positions, velocities and accelerations of the desired operational-space *back-and-forth* motion in the planned *Transport* phase. The desired kinematics of the operational-space *Approach* and *Retreat* phases are described

by Equations 8.2.19–8.2.21 and by Equations 8.2.22–8.2.24, respectively.

$$\begin{aligned}
 \mathbf{x}_{tran}^{direct}(g; \mathbf{x}_{tran}) &= \mathbf{x}_0 + (\mathbf{x}_{tran} - \mathbf{x}_0) (10g^3 - 15g^4 + 6g^5) \\
 &\quad + \dot{\mathbf{x}}_0 T (g - 6g^3 + 8g^4 - 3g^5) \\
 &\quad + \dot{\mathbf{x}}_{tran} T (-4g^3 + 7g^4 - 3g^5) \\
 &\quad + \frac{1}{2} \ddot{\mathbf{x}}_0 T^2 (g^2 - 3g^3 + 3g^4 - g^5) \\
 &\quad + \frac{1}{2} \ddot{\mathbf{x}}_{tran} T^2 (g^3 - 2g^4 + g^5)
 \end{aligned} \tag{8.2.13}$$

$$\begin{aligned}
 \dot{\mathbf{x}}_{tran}^{direct}(g; \mathbf{x}_{tran}) &= \frac{30}{T} (\mathbf{x}_{tran} - \mathbf{x}_0) (g^2 - 2g^3 + g^4) \\
 &\quad + \dot{\mathbf{x}}_0 (1 - 18g^2 + 32g^3 - 15g^4) \\
 &\quad + \dot{\mathbf{x}}_{tran} (-12g^2 + 28g^3 - 15g^4) \\
 &\quad + \frac{1}{2} \ddot{\mathbf{x}}_0 T (2g - 9g^2 + 12g^3 - 5g^4) \\
 &\quad + \frac{1}{2} \ddot{\mathbf{x}}_{tran} T (3g^2 - 8g^3 + 5g^4)
 \end{aligned} \tag{8.2.14}$$

$$\begin{aligned}
 \ddot{\mathbf{x}}_{tran}^{direct}(g; \mathbf{x}_{tran}) &= \frac{60}{T^2} (\mathbf{x}_{tran} - \mathbf{x}_0) (g - 3g^2 + 2g^3) \\
 &\quad + \frac{12}{T} \dot{\mathbf{x}}_0 (-3g + 8g^2 - 5g^3) \\
 &\quad + \frac{12}{T} \dot{\mathbf{x}}_{tran} (-2g + 7g^2 - 5g^3) \\
 &\quad + \ddot{\mathbf{x}}_0 (1 - 9g + 18g^2 - 10g^3) \\
 &\quad + \ddot{\mathbf{x}}_{tran} (-3g - 12g^2 + 10g^3)
 \end{aligned} \tag{8.2.15}$$

$$\begin{aligned}
 \mathbf{x}_{tran}^{bkfor}(g; \mathbf{x}_b) &= \frac{g(1-g)}{t_b(1-t_b)} (\mathbf{x}_b - \mathbf{x}_0) \sin^2(\pi g^\vartheta), \\
 \vartheta &= -\frac{\ln 2}{\ln t_b}, t_b \in]0,1[
 \end{aligned} \tag{8.2.16}$$

$$\dot{\mathbf{x}}_{tran}^{bkfor}(g; \mathbf{x}_b) = \frac{1}{T t_b(1-t_b)} (\mathbf{x}_b - \mathbf{x}_0) \cdot \left[(1-2g) \sin^2(\pi g^\vartheta) + (1-g) \pi \vartheta g^\vartheta \sin(2\pi g^\vartheta) \right] \tag{8.2.17}$$

$$\begin{aligned}
 \ddot{\mathbf{x}}_{tran}^{bkfor}(g; \mathbf{x}_b) &= \frac{2}{T^2 t_b(1-t_b)} (\mathbf{x}_b - \mathbf{x}_0) \\
 &\quad \cdot \left[-\sin^2(\pi g^\vartheta) + \pi^2 \vartheta^2 (1-g) g^{2\vartheta-1} \cos(2\pi g^\vartheta) \right. \\
 &\quad \left. + (1-2g) \pi \vartheta g^{\vartheta-1} \sin(2\pi g^\vartheta) \right]
 \end{aligned} \tag{8.2.18}$$

$$\mathbf{x}_{app}(g; \mathbf{x}_{app}) = \mathbf{x}_0 + \frac{1}{4}(\mathbf{x}_{app} - \mathbf{x}_0)(5g - g^5) \quad (8.2.19)$$

$$\dot{\mathbf{x}}_{app}(g; \mathbf{x}_{app}) = \frac{5}{4T}(\mathbf{x}_{app} - \mathbf{x}_0)(1 - g^4) \quad (8.2.20)$$

$$\ddot{\mathbf{x}}_{app}(g; \mathbf{x}_{app}) = -\frac{5}{T^2}(\mathbf{x}_{app} - \mathbf{x}_0)g^3 \quad (8.2.21)$$

$$\mathbf{x}_{ret}(g; \mathbf{x}_{ret}) = \mathbf{x}_0 + \frac{1}{3}(\mathbf{x}_{ret} - \mathbf{x}_0)(5g^4 - 2g^5) \quad (8.2.22)$$

$$\dot{\mathbf{x}}_{ret}(g; \mathbf{x}_{ret}) = \frac{10}{3T}(\mathbf{x}_{ret} - \mathbf{x}_0)(2g^3 - g^4) \quad (8.2.23)$$

$$\ddot{\mathbf{x}}_{ret}(g; \mathbf{x}_{ret}) = \frac{20}{3T^2}(\mathbf{x}_{ret} - \mathbf{x}_0)(3g^2 - 2g^3) \quad (8.2.24)$$

The duration of motion T is specific for each phase and is calculated as described in section 5.3. Similarly to the planning of the joints-space path, the kinematical boundary conditions $\dot{\mathbf{x}}_{tran}$ and $\ddot{\mathbf{x}}_{tran}$ of the *Transport* phase (in Equations 8.2.13–8.2.15) are set to guarantee continuity with the subsequent phase and avoid undesired jerky intra-movement kinematics. The normalized time parameter g is designed to avoid the explicit dependence of time as it was proposed by Schaal et al. (2007). More specifically, the behaviour of the parameter g follows the dynamics of the system in Equation 8.2.25, which can also be interpreted as shown in Equation 8.2.26.

$$\tau \dot{g} + \frac{a}{1 + w \|\mathbf{x}_{tar} - \mathbf{x}\|^2} g = \frac{a}{1 + w \|\mathbf{x}_{tar} - \mathbf{x}\|^2} \quad (8.2.25)$$

$$g(t) = 1 - \exp\left(-\frac{a}{\tau (1 + w \|\mathbf{x}_{tar} - \mathbf{x}\|^2)} t\right) \quad (8.2.26)$$

The variables $a, \tau, w \in \mathbb{R}^+$ are tuned accordingly a given reference trajectory so that $g(0) = 0$ and $g(T) = 1$. Indeed, the parameter $g \in [0, 1]$ starts from 0 and smoothly increases to 1 when the actual hand pose $\mathbf{x}$ reaches the desired target hand pose $\mathbf{x}_{tar}$. When the error $\|\mathbf{x}_{tar} - \mathbf{x}\|^2$ is large, the exponential term in Equation 8.2.26 vanishes at a slow pace and eventually stops the raising of g towards its final value 1. This behaviour is particularly desired in trajectory tracking when sudden perturbations to the target hand pose $\mathbf{x}_{tar}$ occur (Schaal et al., 2007). Mapping the evolution of time onto the interval $[0, 1]$ also permits an agile management of the switch between two adjacent phases of a reference movement by simply setting thresholds $0 < g_{th} \leq 1$. For example, if during the tracking of the *Transport* phase of a pick movement the condition $g \geq g_{th}$ is verified, then the reference set-point abandons the *Transport* phase and follows the trajectory given by the subsequent *Approach* phase.

8.3 The controllers

Given a trajectory $(\mathbf{x}, \dot{\mathbf{x}}, \ddot{\mathbf{x}})$ in a m -dimensional operational space of a n -DOFs manipulator, a simple inversion of the kinematics can be derived from Equation 8.2.3 and it is given in Equation 8.3.1 when $n = m$. When the manipulator is redundant ($n > m$), the pseudo-inverse of the Jacobian matrix $\mathbf{J}^+$ is used instead of its classical inverse $\mathbf{J}^{-1}$ (Equations 8.3.2 and 8.3.3). The degrees of redundancy allow a customize usage of the null space of the Jacobian $\mathcal{N}(\mathbf{J}) \in \mathbb{R}^n$, which is that subspace of accelerations $\ddot{\boldsymbol{\theta}}_{null}$ that do not affect the motion of the end-effector (Siciliano et al., 2009). This approach to the control of redundant manipulators allows to apply a priority strategy on motion tasks, which downgrades the null space motion to a second-priority level (Chiaverini et al., 1991).

$$\ddot{\boldsymbol{\theta}} = \mathbf{J}^{-1} (\ddot{\mathbf{x}} - \dot{\mathbf{J}}\dot{\boldsymbol{\theta}}) \quad (8.3.1)$$

$$\ddot{\boldsymbol{\theta}} = \mathbf{J}^+ (\ddot{\mathbf{x}} - \dot{\mathbf{J}}\dot{\boldsymbol{\theta}}) + \ddot{\boldsymbol{\theta}}_{null} \quad (8.3.2)$$

$$\mathbf{J}^+ = \mathbf{J}^T (\mathbf{J}\mathbf{J}^T)^{-1} \quad (8.3.3)$$

$$\ddot{\boldsymbol{\theta}}_{null} = (\mathbf{I} - \mathbf{J}^+ \mathbf{J}) \ddot{\boldsymbol{\theta}}_0 \quad (8.3.4)$$

As shown in Equation 8.3.4, the accelerations $\ddot{\boldsymbol{\theta}}_{null}$ result from the projection onto the null space of the Jacobian $\mathcal{N}(\mathbf{J})$ of accelerations $\ddot{\boldsymbol{\theta}}_0$, which are properly designed to satisfy desired constraints of the motion (Nakanishi et al., 2008; Siciliano et al., 2009). In this study, anthropomorphic robotic manipulators with 7-DOFs are addressed and, therefore, there is one degree of redundancy when the end-effector is fully constrained in a 6-dimensional operational space. For this reason, Equation 8.3.2 is taken into consideration to design the proposed laws of control. In specific, the accelerations of the end-effector $\ddot{\mathbf{x}}$ are chosen to respect the law of control in Equation 8.3.5, which can be re-written as in Equation 8.3.6 in the space of the corresponding errors (Equations 8.3.7).

$$\ddot{\mathbf{x}} = \ddot{\mathbf{x}}_d + \mathbf{K}_v (\dot{\mathbf{x}}_d - \dot{\mathbf{x}}) + \mathbf{K}_c (\mathbf{x}_d - \mathbf{x}) \quad (8.3.5)$$

$$\mathbf{0} = \ddot{\tilde{\mathbf{x}}} + \mathbf{K}_v \dot{\tilde{\mathbf{x}}} + \mathbf{K}_c \tilde{\mathbf{x}} \quad (8.3.6)$$

$$\ddot{\tilde{\mathbf{x}}} = \begin{bmatrix} \ddot{\mathbf{p}}_d - \ddot{\mathbf{p}} \\ \ddot{\boldsymbol{\omega}}_d - \ddot{\boldsymbol{\omega}} \end{bmatrix} \quad \dot{\tilde{\mathbf{x}}} = \begin{bmatrix} \dot{\mathbf{p}}_d - \dot{\mathbf{p}} \\ \dot{\boldsymbol{\omega}}_d - \dot{\boldsymbol{\omega}} \end{bmatrix} \quad \tilde{\mathbf{x}} = \begin{bmatrix} \mathbf{p}_d - \mathbf{p} \\ \mathbf{e}_o \end{bmatrix} \quad (8.3.7)$$

The matrices of control gains $\mathbf{K}_v$ $\mathbf{K}_c$ are chosen to be diagonal and positive definite to ensure the stability of the dynamical system in Equation 8.3.6 (see section 8.3.3). The errors defined by Equations 8.3.7 measure the differences with the desired translational state $(\ddot{\mathbf{p}}_d, \dot{\mathbf{p}}_d, \mathbf{p}_d)$ and with the desired rotational state $(\dot{\boldsymbol{\omega}}_d, \boldsymbol{\omega}_d, \mathbf{Q}_d)$ of the end-effector. The orienting error $\mathbf{e}_o$ (Equation 8.3.8) is computed in reference with the desired orientation of the end-effector, which is expressed by the quaternion $\mathbf{Q}_d = \{\eta_d, \boldsymbol{\epsilon}_d\}$.

$$\mathbf{e}_o = \int_0^T \boldsymbol{\omega}_d dt - \int_0^T \boldsymbol{\omega} dt \quad (8.3.8)$$

Therefore, the orienting error in the operational space can be expressed by the quaternion ΔQ that is formulated in Equations 8.3.9, 8.3.10 and 8.3.11 (Siciliano et al., 2009).

$$\Delta Q = \{\Delta\eta, \Delta\epsilon\} = Q_d * Q^{-1} \quad (8.3.9)$$

$$\Delta\epsilon = \eta(\theta) \epsilon_d - \eta_d \epsilon(\theta) - \epsilon_d \times \epsilon(\theta) \quad (8.3.10)$$

$$\Delta\eta = \eta(\theta) \eta_d + \epsilon(\theta)^\top \epsilon_d \quad (8.3.11)$$

It is worth noting that the quaternion $\Delta Q = \{1, \mathbf{0}\}$ when the actual rotational matrix R and the desired rotational matrix R_d of the end-effector are aligned. Therefore, defining $e_o = \Delta\epsilon$ provides sufficient information to the controller in Equation 8.3.6 (Siciliano et al., 2009).

The complete feedback control scheme for tracking a trajectory in the operational space of an anthropomorphic manipulator is illustrated in Figure 8.2. In order to avoid the inconvenient calculus of the first

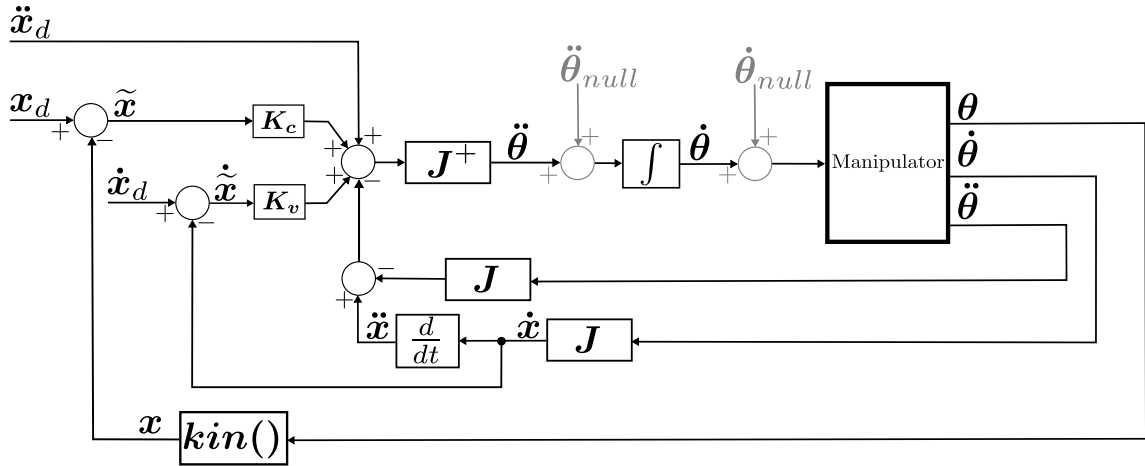


Figure 8.2: Diagram of the feedback loop control in the operational space of the manipulator.

derivative of the Jacobian $\dot{J}$, the term $J\dot{\theta}$ in Equation 8.3.2, is obtained from Equation 8.2.3 as the subtraction $\dot{x} - J\ddot{\theta}$. The motion of second-priority tasks is projected onto the null space of the Jacobian $\mathcal{N}(J)$ and can be added to either the accelerations or the velocities in the joints space according to the physical nature of the imposed constraints.

As suggested by Yuan (1988), the control of the errors in the operational space of manipulation (Equation 8.3.6) is split to separately manage positional and rotational tracking (Equations 8.3.12 and 8.3.13).

$$(\ddot{p}_d - \ddot{p}) + K_d(\dot{p}_d - \dot{p}) + K_p(p_d - p) = \mathbf{0} \quad (8.3.12)$$

$$(\dot{\omega}_d - \dot{\omega}) + K_\omega(\omega_d - \omega) + K_o e_o = \mathbf{0} \quad (8.3.13)$$

The systems in Equations 8.3.12 and 8.3.13 are stable because the matrices K_d , K_p , K_ω and K_o are chosen to be diagonal and positive definite (see section 8.3.3). Given the system of errors formulated in

Equation 8.3.6, implementing the control in Equations 8.3.12 and 8.3.13 implies the relations formulated in Equations 8.3.14.

$$K_v = \begin{bmatrix} K_d & 0 \\ 0 & K_\omega \end{bmatrix} \quad K_c = \begin{bmatrix} K_p & 0 \\ 0 & K_o \end{bmatrix} \quad (8.3.14)$$

A desired human-like joints-space trajectory not only guides the hand of the manipulator but also carries information of the motion of the fingers and of a desired swivel angle, which are tracked by the controllers described in sections 8.3.1 and 8.3.2, respectively.

8.3.1 Finger motion tracking

The anthropomorphic manipulators of the robot ARoS (Figure 4.1a) and Sawyer (Figure 4.3) are equipped with a 3-fingered Barrett Hand (Figure 4.4), which adds 4 variables to the joints space of the considered robotic manipulator. Therefore, the vector of joints values $\theta \in \mathbb{R}^4$ is four-dimensional because it considers the flexion of each finger and the spread between the first two fingers (see Figure A.4). The motion of a desired trajectory of the fingers $(\ddot{\theta}_d, \dot{\theta}_d, \theta_d)$ is directly provided by the HUMP module as it is described in chapter 5. In correspondence with the phase of a movement, a desired path for the fingers is provided after a normalization according to the mapping parameter $g(t)$ described by the Equations 8.2.25 and 8.2.26. This normalization ensures the correct temporal coupling between a desired trajectory of the fingers and a desired end-effector-trajectory. For tracking purposes, the actual accelerations of the fingers $\ddot{\theta}$ is chosen to respect the second-order control law of Equation 8.3.15, which can be re-written as in Equation 8.3.16 by enhancing the differences between the desired and the actual trajectories as shown in Equations 8.3.17.

$$\ddot{\theta} = \ddot{\theta}_d + H_v (\dot{\theta}_d - \dot{\theta}) + H_c (\theta_d - \theta) \quad (8.3.15)$$

$$0 = \ddot{\tilde{\theta}} + H_v \dot{\tilde{\theta}} + H_c \tilde{\theta} \quad (8.3.16)$$

$$\ddot{\tilde{\theta}} = \ddot{\theta}_d - \ddot{\theta} \quad \dot{\tilde{\theta}} = \dot{\theta}_d - \dot{\theta} \quad \tilde{\theta} = \theta_d - \theta \quad (8.3.17)$$

The gains matrices H_v and H_c are set to be diagonal and positive definite to ensure the stability of the system in Equation 8.3.16. The complete diagram of the described control strategy is illustrated in Figure 8.3.

8.3.2 Null space motion

The motion that is projected onto the null space of the Jacobian $\mathcal{N}(J)$ is computed to satisfy necessary constraints for robots that operate in a dynamic workspace. Specifically, the proposed control scheme in Figure 8.2 is designed to add velocities and accelerations in the joints space of an anthropomorphic

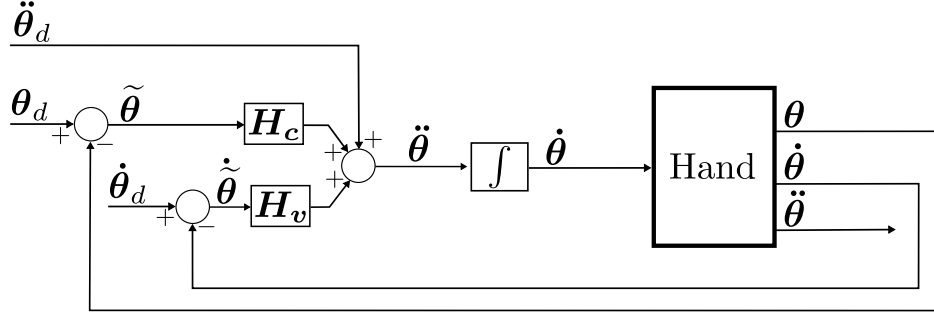


Figure 8.3: Diagram of the feedback loop control in the joints space of the robotic hand.

manipulator to ensure the respect of physical joint limits, the avoidance of singularities in the calculus of the pseudo-inverse J^+ , the avoidance of dynamic obstacles and the tracking of a desired *swivel angle* trajectory. While the latter generates angular accelerations (see section 8.3.2.1), the former requirements are satisfied in the joints space of velocities by adopting the adaptive robust gradient projection method (ARGPM) proposed by Liu et al. (2010). The angular velocities projected onto the null space $\mathcal{N}(J)$ result from the composition of three essential requisites as it is shown in Equation 8.3.18.

$$\dot{\theta}_{null} = \dot{\theta}_{\xi_{jlim}} + \dot{\theta}_{\xi_{sing}} + \dot{\theta}_{\xi_{obst}} \quad (8.3.18)$$

The vector $\dot{\theta}_{null} \in \mathbb{R}^n$ is the sum of the angular velocities $\dot{\theta}_{\xi_{jlim}}$, $\dot{\theta}_{\xi_{sing}}$ and $\dot{\theta}_{\xi_{obst}}$, which guarantee the respect of joint physical limits, the avoidance of singularities and of obstacles, respectively. These second-priority tasks of the motion are achieved by the minimization of scalar objective functions ξ_{jlim} , ξ_{sing} and ξ_{obst} through the projection of their gradient onto the null space of the Jacobian $\mathcal{N}(J)$. Concerning a generic objective function ξ , its gradient $\nabla \xi$ produces angular velocities as formulated in Equation 8.3.19.

$$\dot{\theta}_{\xi} = k_{\xi} \cdot f_{\xi} \cdot (I - J^+ J) \nabla \xi \quad (8.3.19)$$

$$k_{\xi} = -\frac{V_{max} - \|J^+ \dot{x}\|}{\|I - J^+ J\| \|(I - J^+ J) \nabla \xi\|} \quad (8.3.20)$$

$$f_{\xi} = \beta (1 - \exp(-\lambda \|(I - J^+ J) \nabla \xi\|)) \quad (8.3.21)$$

The term k_{ξ} in Equation 8.3.20 brings robustness for a fast convergence of the gradient projection method while ensuring that a given maximum angular velocity V_{max} is never exceeded. The multiplication with the term f_{ξ} (Equation 8.3.21) avoids numerical oscillations because the velocities are adapted to the value of the gradient $\nabla \xi$. The constants β and λ in Equation 8.3.21 are arbitrarily tuned according to the objective

function ξ . More details on the robust adaptive gradient projection method are given in Liu et al., 2010. Each component of the null space motion in Equation 8.3.18 has been described in details by Vu (2012) with particular reference to the right arm of the robot ARoS (Figure 4.1a). Taking inspiration from the latter study, three objective functions are here introduced for the avoidance of physical joints limits, of singularities in the calculus of the pseudo-inverse J^+ and of obstacles, respectively. For instance, the function that measures the distance between an actual joint value and the middle of the physical angular range is given by Equation 8.3.22.

$$\xi_j = \frac{1}{2} \sum_{i=1}^n \left(\frac{\theta_i - \theta_{i,mid}}{\theta_{i,max} - \theta_{i,min}} \right)^2 \quad (8.3.22)$$

$$\frac{\partial \xi_j}{\partial \theta_i} = \frac{\theta_i - \theta_{i,mid}}{\theta_{i,max} - \theta_{i,min}} \quad (8.3.23)$$

$$\frac{\partial \xi_{jlim}}{\partial \theta_i} = \frac{\partial \xi_j}{\partial \theta_i} \left(1 + \exp \left(\frac{\frac{\partial \xi_j}{\partial \theta_i} - \mu_j}{\rho_j} \right) \right) \quad (8.3.24)$$

$$\nabla \xi_{jlim} = \left[\frac{\partial \xi_{jlim}}{\partial \theta_1} \quad \dots \quad \frac{\partial \xi_{jlim}}{\partial \theta_n} \right]^\top \quad (8.3.25)$$

The first partial derivative of the function in Equation 8.3.22 with respect to the i^{th} joint value is reported by Equation 8.3.23. For better performance, this derivative is modified as shown in Equation 8.3.24 to form the gradient in Equation 8.3.25. The coefficients μ_j and ρ_j serve as a threshold to the increase of the unmodified gradient and as an increasing rate, respectively (Vu, 2012).

The avoidance of the singularities in the calculus of the pseudo-inverse of the Jacobian J^{th} is achieved by minimizing the function defined in Equation 8.3.26, which maximizes the manipulability measure of the manipulator (Siciliano et al., 2009).

$$\xi_{sing} = -\sqrt{|JJ^\top|} \quad (8.3.26)$$

$$\frac{\partial \xi_{sing}}{\partial \theta_i} = \frac{\xi_{sing}(\theta_i + \Delta\theta_i) - \xi_{sing}(\theta_i)}{\Delta\theta_i} \quad (8.3.27)$$

$$\nabla \xi_{sing} = \left[\frac{\partial \xi_{sing}}{\partial \theta_1} \quad \dots \quad \frac{\partial \xi_{sing}}{\partial \theta_n} \right]^\top \quad (8.3.28)$$

The first partial derivatives are computed numerically by finite differences as shown in Equation 8.3.27 and form the gradient in Equation 8.3.28 (Vu, 2012).

The minimization of the objective function in Equation 8.3.29 permits the online avoidance of obstacles

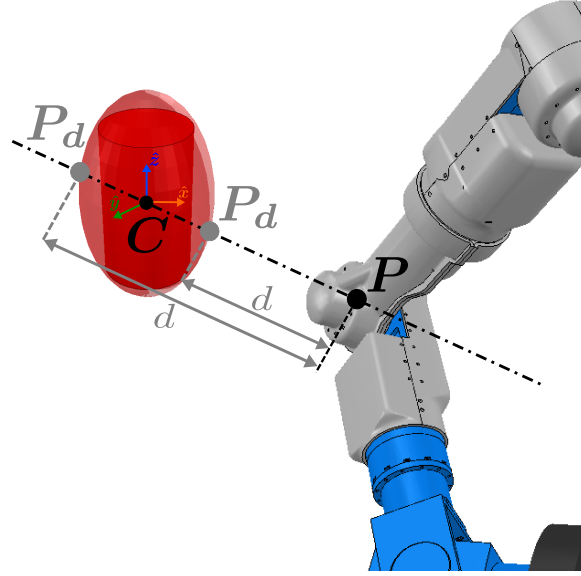


Figure 8.4: Distance between an obstacle and a point on the robotic arm. The point $P_d = P + D$ defines the distances $d = \|D\|$. Only the minimum distance is considered in the online obstacle avoidance mechanism.

that may change position and orientation in the workspace.

$$\xi_{obst} = \sum_{k=1}^{n_p} \sum_{l=1}^{n_{obst}} \frac{1}{k_{off} + d_{k,l}} \quad (8.3.29)$$

$$d_{k,l} = \|D_{k,l}\| \quad (8.3.30)$$

$$1 = (P_k + D_{k,l} - C_l)^T R_l^T A_l R_l (P_k + D_{k,l} - C_l) \quad (8.3.31)$$

Given a number of obstacles in the workspace n_{obst} and a number of points on the manipulator n_p , the distance between the k^{th} point P_k and the l^{th} obstacle $d_{k,l}$ is computed as the minimum distance between the point and the ellipsoid modelling the obstacle (Figure 8.4). The coefficient k_{off} stabilizes the performance of the criterion in Equation 8.3.29 and avoids numerical issues when $d_{k,l}$ is very small (Vu, 2012). More specifically, the distance $d_{k,l}$ is the norm of the vector $D_{k,l} = \begin{bmatrix} x_{d_{k,l}} & y_{d_{k,l}} & z_{d_{k,l}} \end{bmatrix}^T$ (Equation 8.3.30), which is the minimum distance between the point $P_k = \begin{bmatrix} x_{p_k} & y_{p_k} & z_{p_k} \end{bmatrix}^T$ and the point P_d . The latter point is the result of the intersection between the ellipsoid surface enveloping the l^{th} obstacle (Figure 8.4) and the line connecting P_k and the ellipsoid center $C_l = \begin{bmatrix} x_{c_l} & y_{c_l} & z_{c_l} \end{bmatrix}^T$. The surface of the ellipsoid is given by Equation 8.3.31 where C_l is the center of the l^{th} obstacle, R_l is its rotational matrix and $A_l = \text{diag}(r_{l,x}, r_{l,y}, r_{l,z})$ with $r_{l,x}$, $r_{l,y}$ and $r_{l,z}$ being the dimensions of the ellipsoid along its x , y and z axis, respectively. The point on the surface of the ellipsoid is explicated as the sum of the points P_k and $D_{k,l}$ ($P_d = P_k + D_{k,l}$) to enhance the dependence on distance. In order to compute

the three variables $x_{d_{k,l}}$, $y_{d_{k,l}}$ and $z_{d_{k,l}}$, other two equations are necessary in addition to Equation 8.3.31. These two equations are given by the system 8.3.32 that describes the line in the space connecting a point on the robotic arm with the geometrical center of an obstacle as illustrated in Figure 8.4.

$$\begin{cases} \frac{x_{p_k} + x_{d_{k,l}} - x_{c_l}}{x_{p_k} - x_{c_l}} = \frac{y_{p_k} + y_{d_{k,l}} - y_{c_l}}{y_{p_k} - y_{c_l}} \\ \frac{y_{p_k} + y_{d_{k,l}} - y_{c_l}}{y_{p_k} - y_{c_l}} = \frac{z_{p_k} + z_{d_{k,l}} - z_{c_l}}{z_{p_k} - z_{c_l}} \end{cases} \quad (8.3.32)$$

Two solutions are possible but only the intersection that provides the minimum distance with the manipulator is considered in the objective function in Equation 8.3.29. The latter solution is, then, analytically derived and applied for an online calculus of the distance in Equation 8.3.30.

For the sake of clarity, the swivel angle tracking problem is described separately in section 8.3.2.1 and resolved with the introduction of a second-order law of control that operates in the null space of the Jacobian $\mathcal{N}(J)$.

8.3.2.1 Swivel angle tracking

The *swivel angle* of an anthropomorphic arm can be defined in different ways to indicate the same motion feature (e.g., see Su et al., 2018; Zanchettin et al., 2011). For the design of a controller that tracks a given swivel angle trajectory, the definition illustrated in Figure 8.5 is considered. In specific, the definition provided by Kreutz-Delgado et al. (1990) is used with the sagittal plane of motion as the reference plane. Consequently, the forward kinematic that maps the vector of joints angles θ to the swivel angle α can be computed and it is given by Equation 8.3.33 (Kreutz-Delgado et al., 1990; Zanchettin et al., 2011).

$$\alpha = \text{atan2}(\hat{\mathbf{v}}_{ws}^\top (\hat{\mathbf{z}} \times \mathbf{p}), \hat{\mathbf{z}}^\top \mathbf{p}) = \text{kin}_\alpha(\theta) \quad (8.3.33)$$

$$\dot{\alpha} = J_\alpha \dot{\theta} \quad (8.3.34)$$

$$\ddot{\alpha} = J_\alpha \ddot{\theta} + \dot{J}_\alpha \dot{\theta} \quad (8.3.35)$$

In Equation 8.3.33, the vector $\hat{\mathbf{z}}$ represents the z axis of the origin frame of the manipulator (Figure 8.5), $\hat{\mathbf{v}}_{ws} = \mathbf{v}_{ws} / \|\mathbf{v}_{ws}\|$ with $\mathbf{v}_{ws} = \mathbf{W} - \mathbf{S}$ and $\mathbf{p} = \mathbf{v}_{es} - \mathbf{v}_{ws}^\top \mathbf{v}_{ws} \mathbf{v}_{es}$ with $\mathbf{v}_{es} = \mathbf{E} - \mathbf{S}$. The points O , S , E and W in Figure 8.5 represents the positions of the right arm origin, shoulder, elbow and wrist, respectively, which can be calculated through direct kinematics (see section A.1). As shown in Equations 8.3.34 and 8.3.35, the first and the second derivatives of the swivel angle are mapped onto the joints-space velocities

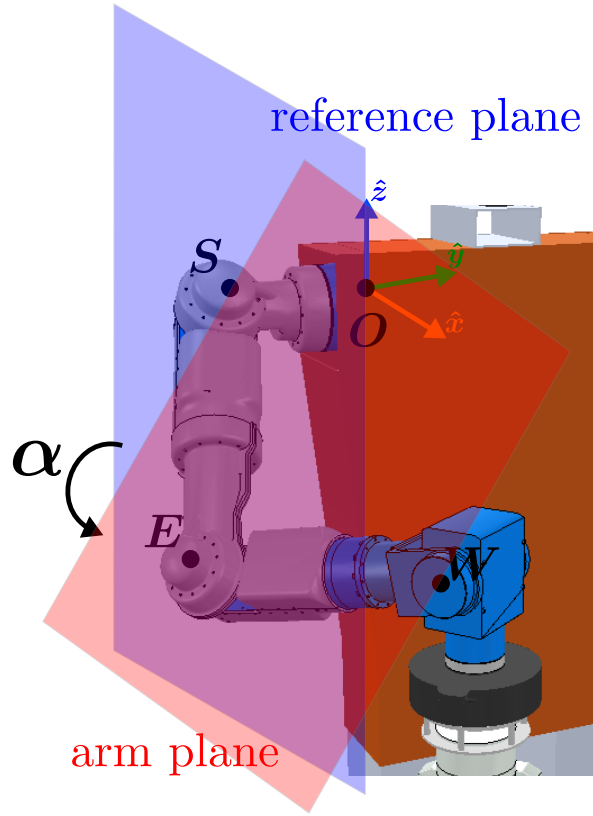


Figure 8.5: Definition of the swivel angle α : the swivel angle is that angle between the reference plane (the sagittal plane) and the arm plane of motion, which contains the shoulder, the elbow and the wrist points of the manipulator.

and accelerations by the Jacobian matrix $J_\alpha \in \mathbb{R}^{1,7}$. This Jacobian matrix was introduced by Kreutz-Delgado et al. (1990) and it is formulated in Equation 8.3.36.

$$J_\alpha = \frac{(\hat{\mathbf{v}}_{ws} \times \hat{\mathbf{p}})^\top}{\|\mathbf{p}\|} J_E + \left[\frac{\hat{\mathbf{z}}^\top \mathbf{v}_{ws}}{\|\mathbf{l}\|} (\hat{\mathbf{v}}_{ws} \times \hat{\mathbf{l}})^\top - \frac{\mathbf{v}_{ws}^\top \mathbf{v}_{es}}{\|\mathbf{v}_{ws}\| \|\mathbf{p}\|} (\hat{\mathbf{v}}_{ws} \times \hat{\mathbf{p}})^\top \right] J_W \quad (8.3.36)$$

$$J_E = [\hat{\mathbf{z}}_1 \times (\mathbf{E} - \mathbf{O}_1) \quad \hat{\mathbf{z}}_2 \times (\mathbf{E} - \mathbf{O}_2) \quad 0 \quad 0 \quad 0 \quad 0 \quad 0] \quad (8.3.37)$$

$$J_W = [\hat{\mathbf{z}}_1 \times (\mathbf{W} - \mathbf{O}_1) \quad \dots \quad \hat{\mathbf{z}}_5 \times (\mathbf{W} - \mathbf{O}_4) \quad 0 \quad 0] \quad (8.3.38)$$

With reference to Equation 8.3.36, $\hat{\mathbf{p}} = \mathbf{p}/\|\mathbf{p}\|$, $\mathbf{l} = (\mathbf{v}_{ws} \times \hat{\mathbf{z}}) \times \mathbf{v}_{ws}$, $\hat{\mathbf{l}} = \mathbf{l}/\|\mathbf{l}\|$, while $J_E \in \mathbb{R}^{3,7}$ and $J_W \in \mathbb{R}^{3,7}$ are Jacobian matrices defined in Equations 8.3.37 and 8.3.38, respectively. These matrices represent the positional elements of the Jacobian matrices calculated with respect to the elbow $\mathbf{E}$ and the wrist $\mathbf{W}$ points, respectively. With reference, for example, to the right arm of the robot ARoS (Figure A.1), the axis $\hat{\mathbf{z}}_i$ and the origin $\mathbf{O}_i$ belong to the i^{th} frame of the Denavit-Hartenberg open chain. In particular, the points $\mathbf{E}$ and $\mathbf{W}$ correspond to the points $\mathbf{O}_3$ and $\mathbf{O}_5$ that are the origins of the 3^{rd} and of the 5^{th} frames, respectively.

As it was shown by Su et al. (2019), the null space of the Jacobian $\mathcal{N}(J)$ can be used to position the elbow as desired and resolve the redundancy of a task. The proposed method introduces angular accelerations, which are related to given swivel angle accelerations and project them onto the null space of the Jacobian according to the diagram shown in Figure 8.6 (Equation 8.3.39).

$$\ddot{\theta}_{null} = \ddot{\theta}_\alpha = (I - J^+ J) \ddot{\theta} \quad (8.3.39)$$

Given the acceleration value of the swivel angle $\ddot{\alpha}$, the inversion of kinematics from Equation 8.3.35 can

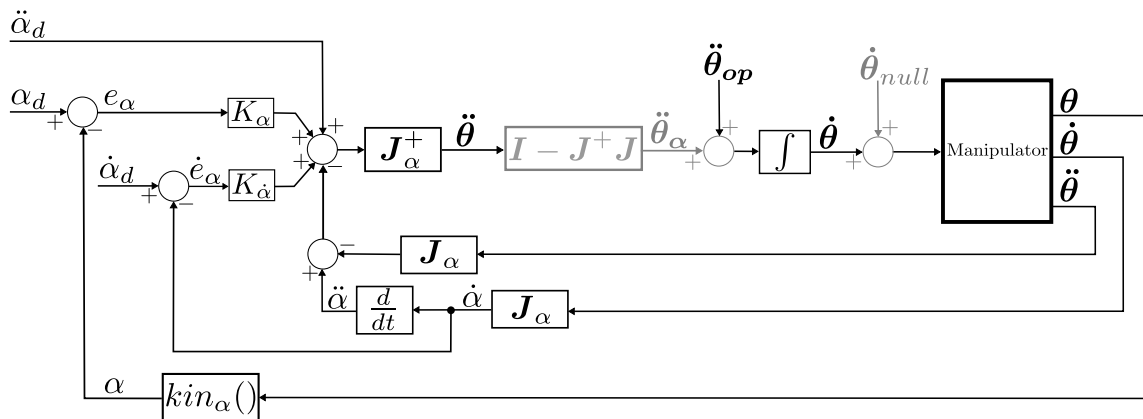


Figure 8.6: Diagram of the feedback loop control for tracking the swivel angle in the null space of manipulation $\mathcal{N}(J)$. The angular accelerations $\ddot{\theta}_{op}$ are computed in the range space of the Jacobian $\mathcal{R}(J)$ by the controller illustrated in Figure 8.2. The angular velocities $\dot{\theta}_{null}$ are computed in the null space $\mathcal{N}(J)$ as formulated in Equation 8.3.19.

be formulated by Equation 8.3.40 with the introduction of the pseudo-inverse J_{α}^{+} (Equation 8.3.41).

$$\ddot{\theta} = J_{\alpha}^{+} \left(\ddot{\alpha} - \dot{J}_{\alpha} \dot{\theta} \right) \quad (8.3.40)$$

$$J_{\alpha}^{+} = J_{\alpha}^{\top} (J_{\alpha} J_{\alpha}^{\top})^{-1} \quad (8.3.41)$$

The acceleration $\ddot{\alpha}$ is selected to respect the law of control in Equation 8.3.42, which can be re-written as in Equation 8.3.43 in the error space (Equations 8.3.44).

$$\ddot{\alpha} = \ddot{\alpha}_d + K_{\dot{\alpha}}(\dot{\alpha}_d - \dot{\alpha}) + K_{\alpha}(\alpha_d - \alpha) \quad (8.3.42)$$

$$0 = \ddot{e}_\alpha + K_{\dot{\alpha}} \dot{e}_\alpha + K_\alpha e_\alpha \quad (8.3.43)$$

$$\ddot{e}_\alpha = \ddot{\alpha}_d - \ddot{\alpha} \quad \dot{e}_\alpha = \dot{\alpha}_d - \dot{\alpha} \quad e_\alpha = \alpha_d - \alpha \quad (8.3.44)$$

The control gains K_α and $K_{\dot{\alpha}}$ are positive to guarantee the stability of the dynamical system 8.3.43 (see section 8.3.3). The errors in Equations 8.3.44 measure the difference between a desired swivel angle state

$(\ddot{\alpha}_d, \dot{\alpha}_d, \alpha_d)$ and the actual swivel angle state $(\ddot{\alpha}, \dot{\alpha}, \alpha)$. As formulated by the Equations 8.3.45, 8.3.46 and 8.3.47, a desired state of the swivel angle can be directly derived from a desired human-like joints-space state $(\ddot{\theta}_d, \dot{\theta}_d, \theta_d)$ through direct kinematics.

$$\alpha_d = \text{kin}_\alpha(\theta_d) \quad (8.3.45)$$

$$\dot{\alpha}_d = J_\alpha \dot{\theta}_d \quad (8.3.46)$$

$$\ddot{\alpha}_d = J_\alpha \ddot{\theta}_d + \dot{J}_\alpha \dot{\theta}_d \quad (8.3.47)$$

A desired human-like joints-space state is part of a human-like angular trajectory, which is planned as it is described in chapter 5 and is also used to generate the desired set-points of the control diagrams in Figures 8.2 and 8.3. In order to couple with a desired operational-space trajectory, a desired swivel angle trajectory is also time normalized according to the mapping parameter $g(t)$ introduced in Equations 8.2.25 and 8.2.26.

8.3.3 Analysis of the stability

The second-order kinematic controllers that act at the levels of the hand, the fingers and the swivel angle need to be analysed to ensure the stability of the overall motion system. The second-order dynamical systems of the error space result from the proposed control laws and are re-written in Equations 8.3.48, 8.3.49 and 8.3.50 for the sake of clarity.

$$\ddot{\tilde{x}} + K_v \dot{\tilde{x}} + K_c \tilde{x} = 0 \quad (8.3.48)$$

$$\ddot{\tilde{\theta}} + H_v \dot{\tilde{\theta}} + H_c \tilde{\theta} = 0 \quad (8.3.49)$$

$$\ddot{e}_\alpha + K_{\dot{\alpha}} \dot{e}_\alpha + K_\alpha e_\alpha = 0 \quad (8.3.50)$$

Specifically, the system in Equation 8.3.48 results from the composition of the dynamical systems for positioning and orienting the end-effector and are re-written in Equations 8.3.51 and 8.3.52.

$$(\ddot{p}_d - \ddot{p}) + K_d(\dot{p}_d - \dot{p}) + K_p(p_d - p) = 0 \quad (8.3.51)$$

$$(\dot{\omega}_d - \dot{\omega}) + K_\omega(\omega_d - \omega) + K_o e_o = 0 \quad (8.3.52)$$

While Equation 8.3.52 is a first-order homogeneous dynamical system of the orientation error in the operational space, Equations 8.3.51, 8.3.49 and 8.3.50 can be similarly analysed as a second-order homogeneous dynamical system formulated by Equation 8.3.53 in the generic variable $x \in \mathbb{R}^n$ (Zhang and Yu, 2018).

$$A_2 \ddot{x} + A_1 \dot{x} + A_0 x = 0 \quad (8.3.53)$$

In this particular case, A_2 is an identity matrix in Equations 8.3.49 and 8.3.51, while it is equal to 1 in Equation 8.3.50. A_1 and A_0 refer to the control gains for the errors in the velocities and in the positions, respectively. For instance, $A_0 = H_c$, $A_1 = H_v$ for the system 8.3.49, $A_0 = K_\alpha$, $A_1 = K_{\dot{\alpha}}$ for the system 8.3.50 and $A_0 = K_p$, $A_1 = K_d$ for the system 8.3.51. The stability of the generic second-order system 8.3.53 can be studied according to the Lyapunov criterion proposed by Zhang and Yu (2018) and used for a proper design of the control gains, while the convergence of the orientation error in Equation 8.3.52 was studied by Yuan (1988) and it is here briefly reported for the sake of completion.

Given the Lyapunov function shown in Equation 8.3.54, its first time derivative can be formulated as given by Equation 8.3.55 using the formulas of *quaternion propagation* (Equations 8.2.4 and 8.2.6) and the control law in Equation 8.3.52.

$$V = (\eta - \eta_d)^2 + (\epsilon - \epsilon_d)^\top (\epsilon - \epsilon_d) + \frac{1}{2} (\omega - \omega_d)^\top (\omega - \omega_d) \quad (8.3.54)$$

$$\dot{V} = -(\omega - \omega_d)^\top K_\omega (\omega - \omega_d) + (\omega - \omega_d)^\top (\Delta\epsilon - K_o e_o) \quad (8.3.55)$$

By selecting the gain $K_o = I$ and the orientation error $e_o = \Delta\epsilon$, the time derivative of the Lyapunov function takes the form shown in Equation 8.3.56.

$$\dot{V} = -(\omega - \omega_d)^\top K_\omega (\omega - \omega_d) \quad (8.3.56)$$

If the matrix K_ω is selected to be positive definite, then $\dot{V}(t) \leq 0 \forall t$ and there is the convergence of the orientation error e_o for any value $\omega \neq \omega_d$. When $\omega = \omega_d$, from Equation 8.3.52, the Equation 8.3.57 is obtained.

$$\frac{d}{dt} (\omega_d - \omega) = -K_o e_o = -K_o \Delta\epsilon \quad (8.3.57)$$

Therefore, at the equilibrium $\Delta\epsilon = 0$ we have $\omega(t) = \omega_d(t) \forall t$. When $\Delta\epsilon \neq 0$, then the condition $\omega = \omega_d$ holds instantaneously because the difference $(\omega_d - \omega)$ changes according to Equation 8.3.57. Therefore, $\dot{V}$ becomes negative, V decreases and $\Delta\epsilon \rightarrow 0$. The demonstrated asymptotic stability of the system 8.3.52 is also global because it holds for any initial value of the error $\Delta\epsilon$ (Yuan, 1988).

Considering the criterion introduced by Zhang and Yu (2018) to reach the stability condition of the homogeneous system 8.3.53, the Lyapunov Equations 8.3.58, 8.3.59 and 8.3.60 must be satisfied.

$$A_0^\top P_1 A_1 + A_1^\top P_1 A_0 = C_0^\top C_0 \quad (8.3.58)$$

$$A_1^\top P_2 A_2 + A_2^\top P_2 A_1 = C_1^\top C_1 \quad (8.3.59)$$

$$A_0^\top P_1 A_2 + A_0^\top P_2 A_2 = C_0^\top C_1 \quad (8.3.60)$$

With n being the number of variables and m arbitrary, the matrices $P_1, P_2 \in \mathbb{R}^{n \times n}$ are positive definite and $C_0, C_1 \in \mathbb{R}^{m \times n}$ are non-zeros matrices (Zhang and Yu, 2018). In the analysis of the systems 8.3.49,

8.3.50 and 8.3.51, $A_2 = I$, therefore the Equations 8.3.58, 8.3.59 and 8.3.60 can be simplified to obtain the Equations 8.3.61, 8.3.62 and 8.3.63.

$$A_0^T P_1 A_1 + A_1^T P_1 A_0 = C_0^T C_0 \quad (8.3.61)$$

$$A_1^T P_2 + P_2 A_1 = C_1^T C_1 \quad (8.3.62)$$

$$A_0^T P_1 + A_0^T P_2 = C_0^T C_1 \quad (8.3.63)$$

For the sake of simplicity, the matrices A_0 , A_1 , P_0 and P_1 are chosen to be diagonal as it is shown in Equations 8.3.64 and 8.3.65.

$$A_0 = \begin{bmatrix} a_{00} & 0 & 0 & \dots & 0 \\ 0 & a_{01} & 0 & \dots & 0 \\ \vdots & & \ddots & & \\ 0 & 0 & & & a_{0n} \end{bmatrix} \quad A_1 = \begin{bmatrix} a_{10} & 0 & 0 & \dots & 0 \\ 0 & a_{11} & 0 & \dots & 0 \\ \vdots & & \ddots & & \\ 0 & 0 & & & a_{1n} \end{bmatrix} \quad (8.3.64)$$

$$P_1 = \begin{bmatrix} p_{10} & 0 & 0 & \dots & 0 \\ 0 & p_{11} & 0 & \dots & 0 \\ \vdots & & \ddots & & \\ 0 & 0 & & & p_{1n} \end{bmatrix} \quad P_2 = \begin{bmatrix} p_{20} & 0 & 0 & \dots & 0 \\ 0 & p_{21} & 0 & \dots & 0 \\ \vdots & & \ddots & & \\ 0 & 0 & & & p_{2n} \end{bmatrix} \quad (8.3.65)$$

With $m = 1$, the matrices C_0 and C_1 can be defined by two arbitrary non-zero values c_0 and c_1 , respectively, as shown in Equations 8.3.66.

$$C_0 = [c_0 \dots c_0] \quad C_1 = [c_1 \dots c_1] \quad c_0, c_1 \in \mathbb{R} - 0 \quad (8.3.66)$$

Substituting Equations 8.3.64, 8.3.65 and 8.3.66 into Equation 8.3.62, the i^{th} element of the matrix P_2 is obtained as shown in Equation 8.3.67, with $i = 0, 1, \dots, n$. Similarly, considering the formula in Equation 8.3.67, from Equation 8.3.61, the i^{th} element of the matrix P_1 is obtained as shown in Equation 8.3.68.

$$p_{2i} = \frac{c_1^2}{2a_{1i}} \quad (8.3.67) \quad p_{1i} = \frac{c_0^2}{2a_{0i}a_{1i}} \quad (8.3.68)$$

In order to have the matrix P_2 positive definite, the terms a_{0i} and a_{1i} must be positive and need to be chosen such that c_1^2 is not negligible when compared to $2a_{1i}$. Similarly, the matrix P_1 is positive definite if c_0^2 is not negligible when compared to $2a_{0i}a_{1i}$. By setting p_{1i} and p_{2i} as in Equations 8.3.68 and 8.3.67, from Equation 8.3.63, Equation 8.3.69 is obtained.

$$a_{1i} = \frac{c_0}{2c_1} + \frac{c_1}{2c_0} a_{0i} \quad (8.3.69)$$

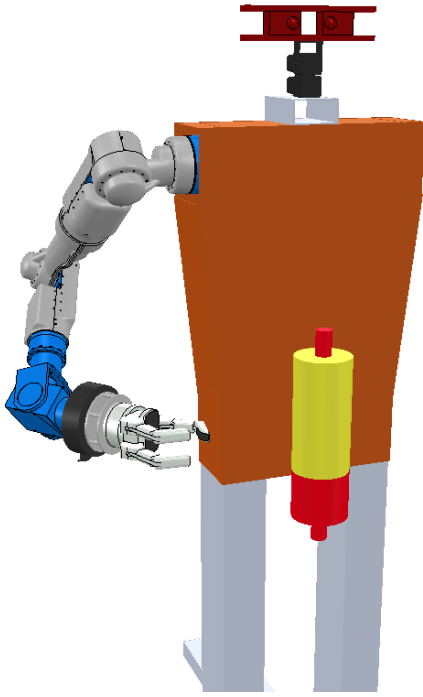
By setting a_{0i} , a positive solution of a_{1i} can be calculated from Equation 8.3.69 for a stable control of the system 8.3.53. It can be noted that, given positive values of a_{0i} , the value of a_{1i} cannot result to be too big, otherwise the positive definiteness of the matrices P_1 and P_2 degenerates and oscillations may occur leading to possible unstable states of the error. Equation 8.3.69 shows that it is always possible to find non-zero values of c_0 and c_1 such that $a_{0i}, a_{1i} > 0 \forall i$ and, therefore, the matrices P_1 and P_2 are positive definite. Additionally, there are no constraints against the selection of non-zero values of c_0 and c_1 such that $C_1\dot{x} + C_0x \neq 0 \Leftrightarrow x \neq 0$ (Zhang and Yu, 2018). For these reasons, the system 8.3.53 is globally asymptotically stable for positive values of the control gains a_{0i} and $a_{1i} \forall i$, which can maintain the matrices P_1 and P_2 positive definite with a sufficient safe margin.

8.4 Experiments and results

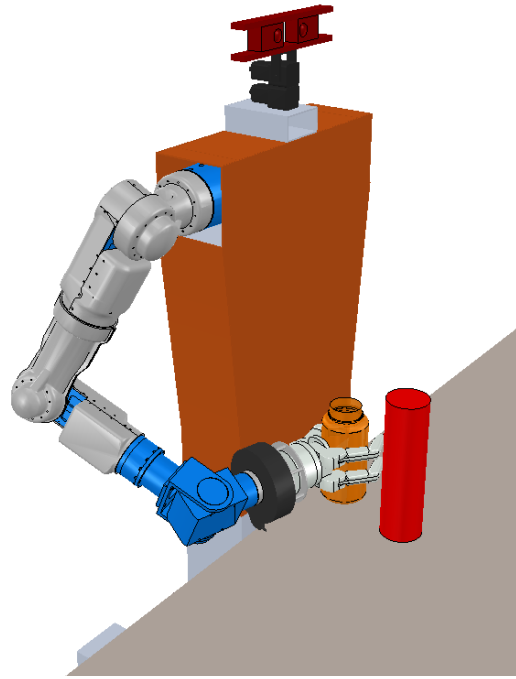
This section describes the implementing details of the controllers introduced in section 8.3 and reports the results of different trajectory-tracking experiments with the humanoid robot ARoS (Figure 4.1a). Specifically, the scenarios illustrated in Figure 8.7 are taken into consideration as principal examples for showing the capabilities of the controllers in picking and placing tasks. In Figures 8.7a and 8.7c, the red column of a toy vehicle (Figure 6.5) is placed in front of the robot and may be exposed to perturbations that change its position and orientation with respect to the world frame. Such a scenario allows to test the tracking performances of the systems described in Equations 8.3.48, 8.3.49 and 8.3.50 along the different phases of motion (Figure 5.2) under unperturbed and perturbed circumstances. Figures 8.7b and 8.7d illustrate the setup described in section 6.3.4, which shows the robot ARoS executing a p_lace movement of an orange bottle avoiding a tall red obstacle that obstructs the path. Given the narrow space between the obstacle and the manipulated object, this scenario requires a high tracking performance from the controllers to respect positions in a collision-free path of the robotic hand. In general, the resulting motion is expected to loyally track the desired operational-space and swivel angle trajectories to qualitatively replicate the kinematic human-like features described in chapter 5. section 8.4.1 briefly describes the setup of the software, while the results of the performed experiments are collected and discussed in section 8.4.2.

8.4.1 Software setup

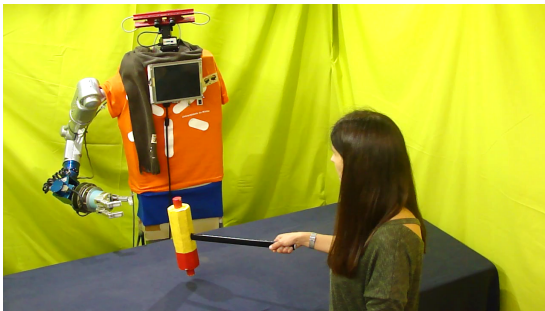
As it is illustrated in Figure 8.8, the framework in Figure 7.6 is augmented with a Motion Controller module that is integrated into the Motion Manager module as a separated thread. The integration with the Motion Manager permits a practical and direct usage of the environmental data, which are either extracted from a simulated scenario or constantly received from the visual sensors of the real robot ARoS (see section 4.1.1). In simulation, sensory noise is artificially added in the detection of objects in the workspace



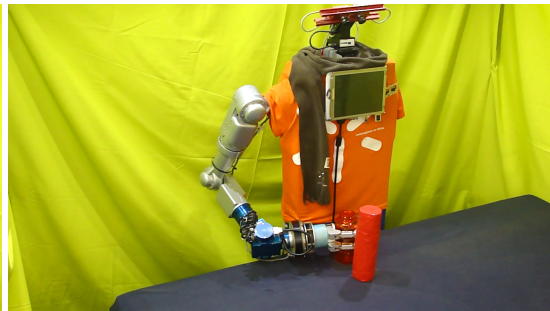
(a) Simulated picking the red column of a toy vehicle



(b) Simulated naturalistic obstacle avoidance



(c) Picking the red column of a toy vehicle



(d) Naturalistic obstacle avoidance

Figure 8.7: Scenarios with the robot ARoS performing two tasks for testing the proposed trajectory-tracking method. Tests with the real robotic system in these scenarios can be seen at link <https://youtu.be/8BJK-qCiews>.

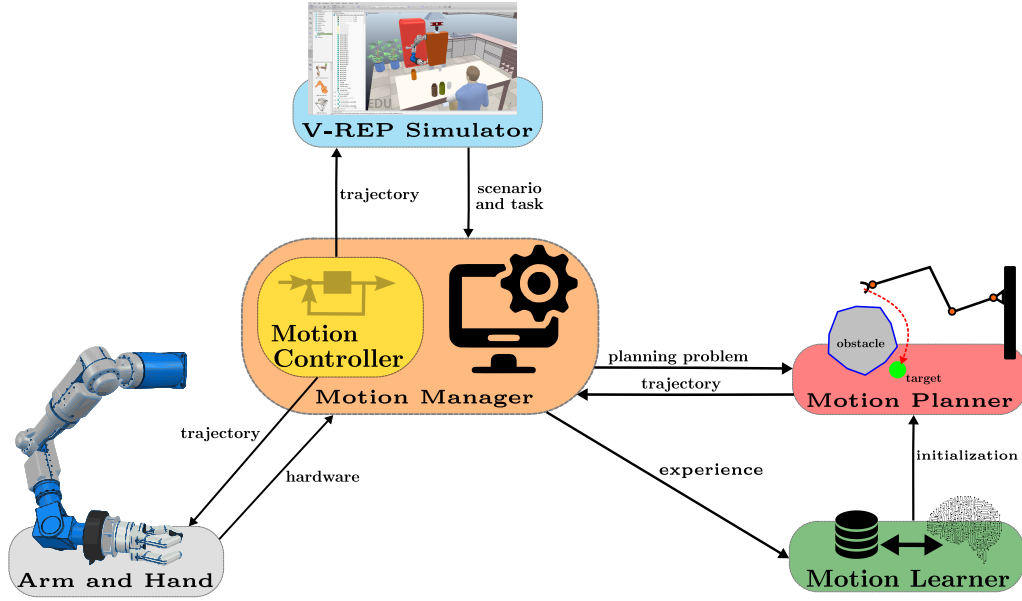


Figure 8.8: Sketch of the software setup that also includes feedback loop controlling. The Motion Controller module implements the proposed controllers and it is integrated into the Motion Manager as a separated thread, which is constantly updated with the latest environmental changes.

of the robot. In such cases and in connection with the real robot, low-pass filtering is applied to mitigate the effect of noisy incoming data. Additionally, when it is necessary, the positional data coming from the manipulator and the end-effector are numerically differentiated to obtain velocities and accelerations in the joints space and in the operational space of manipulation. This differentiation is implemented to improve noise suppression by the mean of filters with variable lengths¹. The process of tuning the time mapping parameter (Equation 8.2.26) and the control gains of the second-order systems 8.3.48, 8.3.49 and 8.3.50 is performed to guarantee stability without oscillations during the convergence of the errors and tailored on the desired human-like upper-limb trajectories.

8.4.2 Results

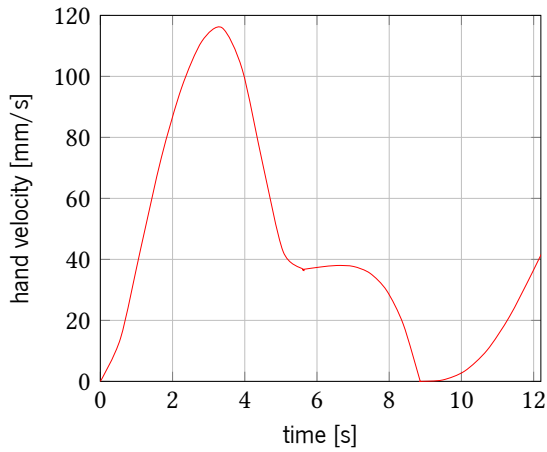
In this section, testing the controllers introduced in section 8.3 for tracking desired human-like motion references is reported. As illustrated in Figure 8.7, a simulated environment with the robot ARoS is firstly utilized to understand the capabilities of the proposed trajectory-tracking system, while experiments on the real robotic platform prove its feasibility. The minimization of the functions formulated in Equations 8.3.22, 8.3.26 and 8.3.29 through the null space of the Jacobian $\mathcal{N}(J)$ was already successfully tested with the robot ARoS by Vu (2012) and, for this reason, the corresponding results are here omitted. The following experiments, therefore, only aim at showing the tracking capabilities of the controllers

introduced in section 8.3 with and without perturbations of the workspace during the execution of a movement. After the tuning of the time mapping parameters (Equations 8.2.25 and 8.2.26) on the planned human-like trajectories, the controlling gains can be selected to obtain stability and a desired convergence of the errors. For more accurate results, each phase of a planned movement (Figure 5.2) is controlled by different gain values for the convergence of the operational-space error in the system 8.3.48. Without regard to the phase of motion, the systems 8.3.49 and 8.3.50 have shown good convergence capabilities by setting constant gain values. For instance, the control of picking the red column target (Figures 8.7a and 8.7c) is tuned with $K_p = 50I$, $K_d = 7I$, $K_o = 30I$ and $K_\omega = 0.5I$ during the *Transport* phase and $K_p = 20I$, $K_d = 4I$, $K_o = 5I$ and $K_\omega = 0.2I$ during the *Approach* and *Retreat* phases. The control of the fingers (Equation 8.3.49) is tuned with $H_c = 200I$ and $H_v = 15I$, while the swivel angle control (Equation 8.3.50) is tuned with $K_\alpha = 100$ and $K_{\dot{\alpha}} = 10$. The human-like place movement of the task in Figures 8.7b and 8.7d is composed by a *Transport* and an *Approach* phases, which are both controlled by setting $K_p = 100I$, $K_d = 2I$, $K_o = 80I$, $K_\omega = 2I$, $H_c = 200I$, $H_v = 15I$, $K_\alpha = 100$ and $K_{\dot{\alpha}} = 10$. The time mapping threshold that signals the passage from a *Transport* to an *Approach* phase is set early as $g_{th} = 0.88$ because a small margin of the error is acceptable to successfully place the end-effector in the vicinity of its target. On the contrary, the time mapping threshold to pass from an *Approach* to a *Retreat* phase is set as $g_{th} = 0.96$ because the end-effector needs to stop moving before either grasp or un-grasp a manipulated object.

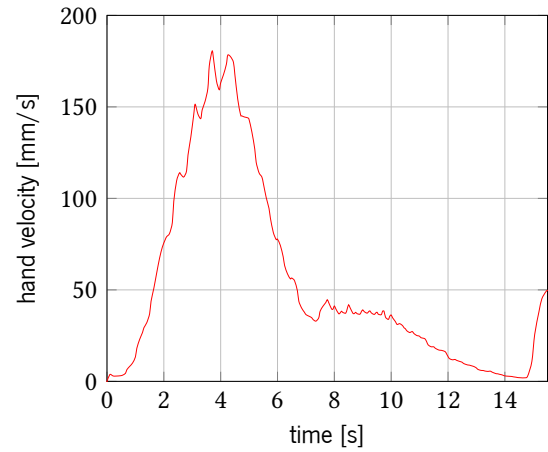
In order to study the tracking performance, first experiments are executed in static scenarios where modifications of the environment do not occur (section 8.4.2.1). In section 8.4.2.2, the proposed controllers are tested in presence of perturbations of the hand pose target that take place after the beginning of the motion.

8.4.2.1 Tests without perturbations of the target

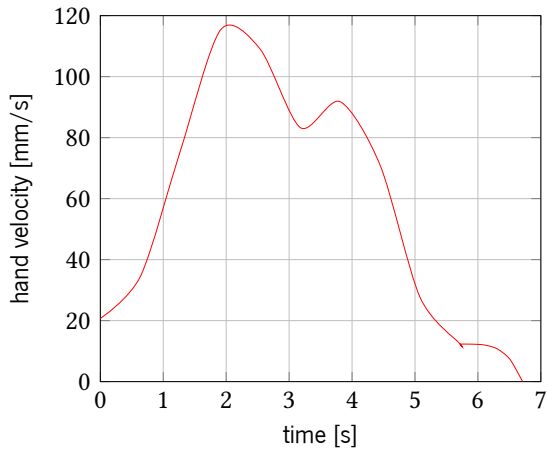
Both scenarios in Figure 8.7 are left unperturbed during the execution of a movement to study the tracking capabilities of the proposed feedback loop controllers. In Figure 8.9, a qualitative analysis of the resulting hand velocity profiles can be done. Although there might be differences in amplitude and noise needs to be filtered, the human-like shape of the profile is preserved. For example, the hand velocity profile illustrated in Figure 8.9b clearly exhibits the three phases of the pick movement. The robotic hand is firstly transported into the vicinity of the red column with a dominant uni-modal bell-shaped velocity profile. Then, the target is approached with a quasi-constant velocity until stopping guarantees a safe grip. Finally, the hand retreats from the target location with a velocity profile that may start a new bell-shaped peak if joined with another planned human-like movement. Good tracking results can be also observed in Figure 8.9d where the second peak rising from the avoidance of the major obstacle in the path towards the target (see



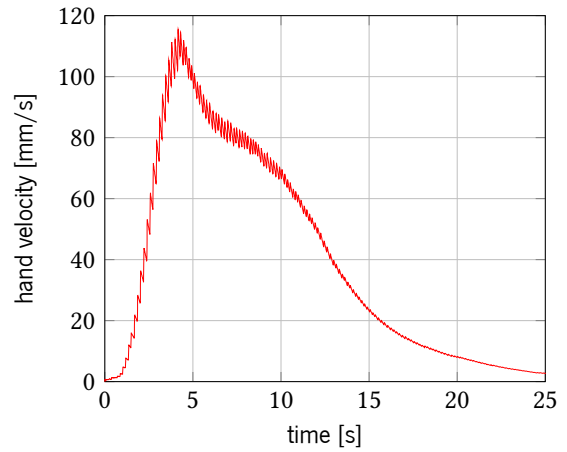
(a) Planned hand velocity in picking (Figures 8.7a and 8.7c)



(b) Actual hand velocity in picking (Figures 8.7a and 8.7c)



(c) Planned hand velocity in naturalistic obstacle avoidance (Figures 8.7b and 8.7d)



(d) Actual hand velocity in naturalistic obstacle avoidance (Figures 8.7b and 8.7d)

Figure 8.9: Qualitative comparison between the planned and the actual hand velocities in absence of perturbations in the workspace of the robot.

section 6.3.4) is reproduced during the transportation of the bottle. Then, for a careful deposit of the bottle, approaching the place location is performed at a slow pace. In comparison with the planned references, the time delay experienced on the actual trajectories is due to different factors. For instance, depending on the task, extended *Approach* phases are often necessary to ensure accuracy in positioning with limited velocities of the manipulator. A more rigorous tuning of the time mapping parameters (Equation 8.2.26) on the values of a desired operational-space trajectory would decrease the delay and shift the plots in Figures 8.9b and 8.9d left on the time axis. Interestingly, different settings of the time mapping system in Equation 8.2.26 would lead to time scales that might suit different accuracy/speed requirements without degrading the desired human-likeness of the trajectory. These properties of the proposed system and

the analysis of the planned time duration (section 5.3.1) during control will be addressed in the future work. However, the observed time delays are not related with the capability of reproducing human-like hand velocity profiles in the different phases of a movement. The designed feedback loop controllers can keep track of the desired human-like features along the planned trajectories, while the errors tend towards zero in each phase as illustrated in Figure 8.10. While the robot is reaching to grasp the red

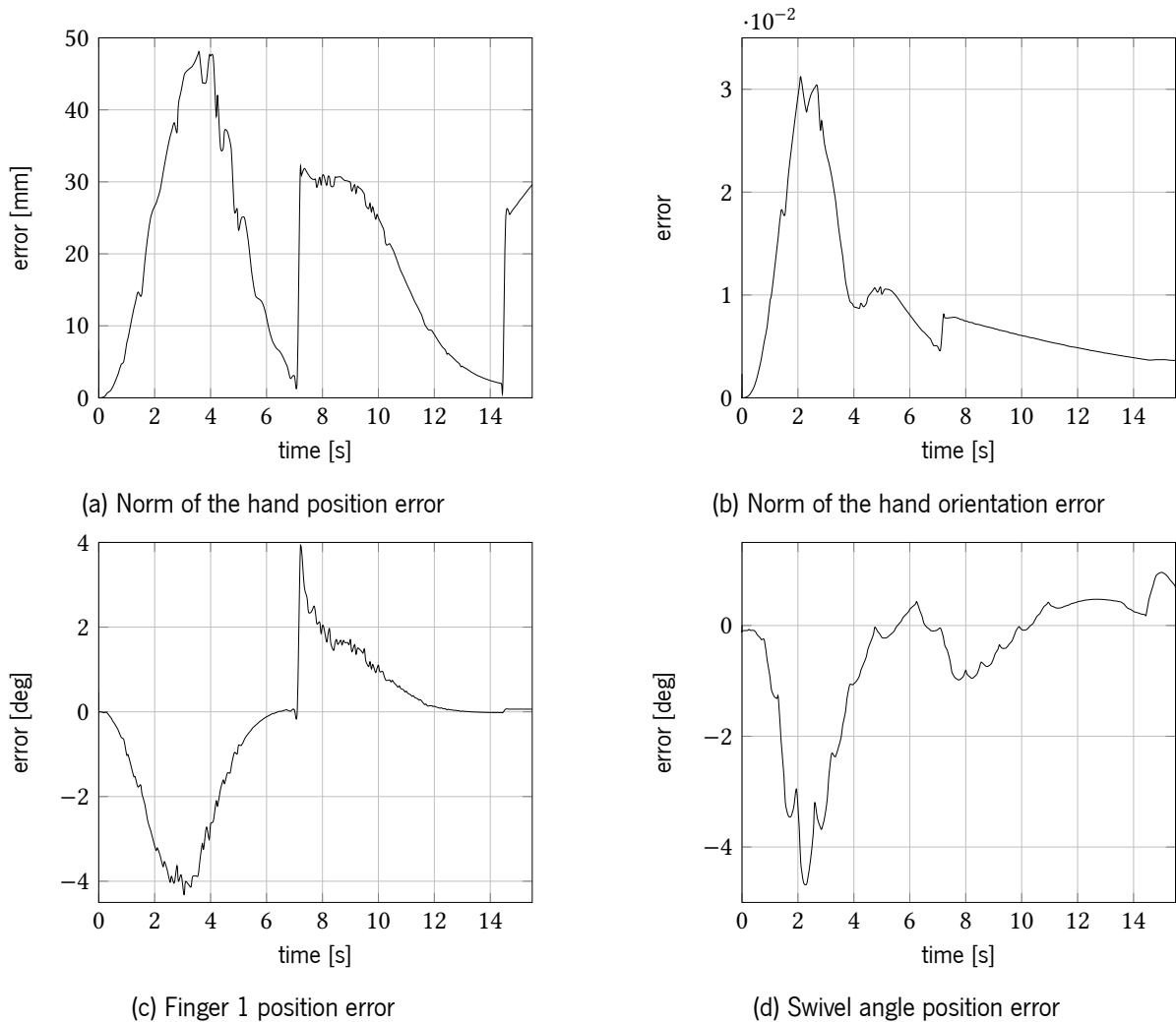


Figure 8.10: Errors during the unperturbed tracking of a human-like pick movement in the scenario in Figure 8.7a.

column (Figure 8.7a), the operational-space error in position and in orientation is sufficiently limited and reaches its peak during the *Transport* phase of the movement (Figures 8.10a and 8.10b). The error on positioning the fingers fully converges after the *Transport* and the *Approach* phases before that the grasp transition occurs (Figure 8.10c). Figure 8.10d shows the error on tracking the desired human-like swivel angle, which reaches its maximum during the *Transport* phase and remains sufficiently limited. More

insights on the tracking performances can be obtained by plotting the desired and the actual trajectories as exhibited in Figure 8.11. It can be observed that a good matching between a desired and the actual

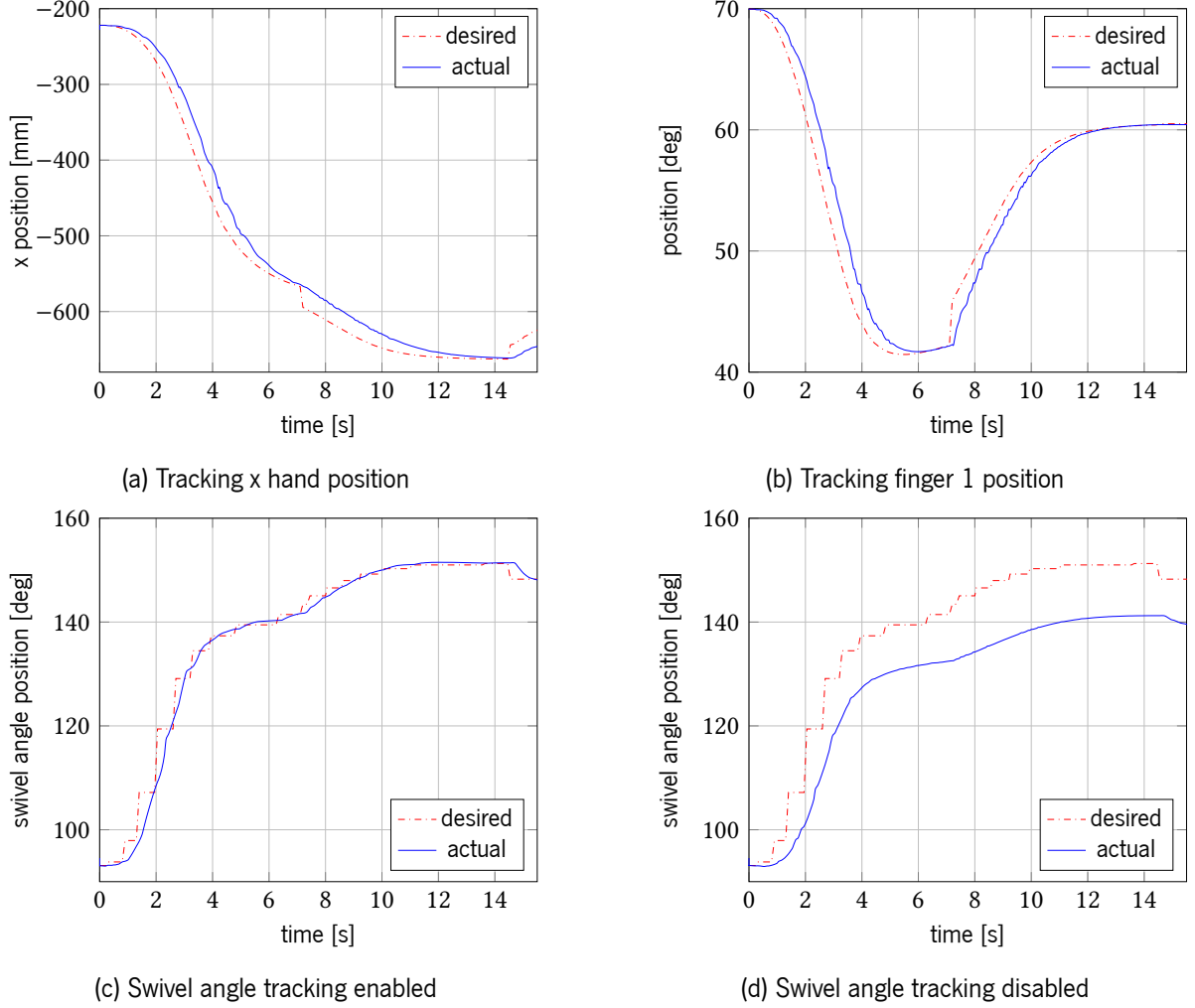


Figure 8.11: Desired and actual trajectories in the scenario in Figure 8.7a.

trajectory is showed, except for the swivel angle tracking when it is disabled (Figure 8.11d). In Figures 8.11a and 8.11b the separation between the phases of the pick movement can be noted when the desired trajectories have discontinuities due to the setting of early time mapping thresholds g_{th} . The desired swivel angle trajectory $\alpha_d(\theta_d(g))$ (Figures 8.11c and 8.11d) results discretized in the parameter g to avoid the online computation of the function kin_α (Equation 8.3.33) and can be successfully tracked when the controller in Equation 8.3.50 is enabled. This circumstance becomes even more evident when the pick movement is planned with the target rotated of 90 degrees around its y axis as illustrated in Figure 8.12a. From the analysis of the Figures 8.12b and 8.12c, it can be concluded that there is an evident good matching between the desired and the actual swivel angle trajectories when the controller formulated in

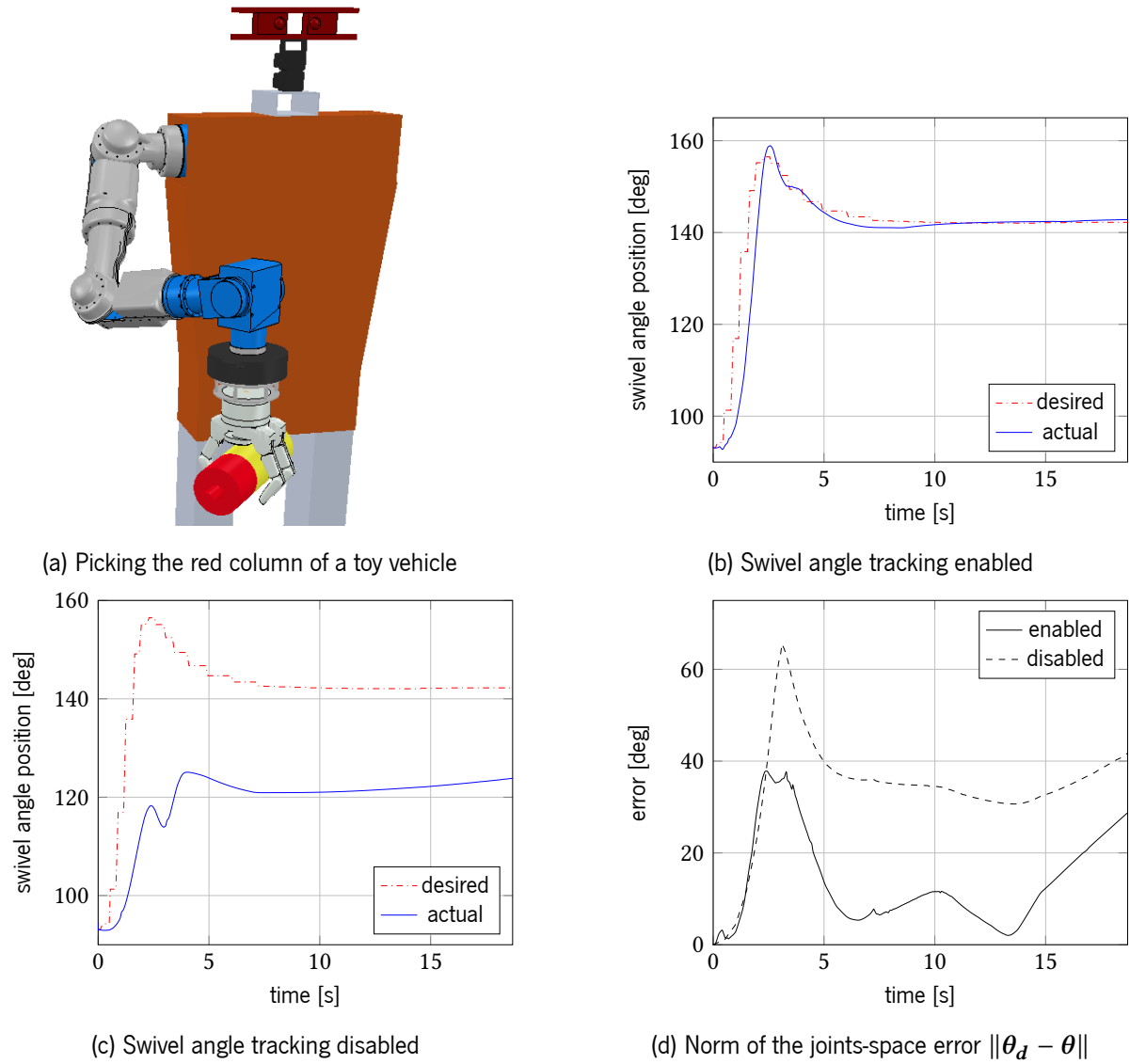
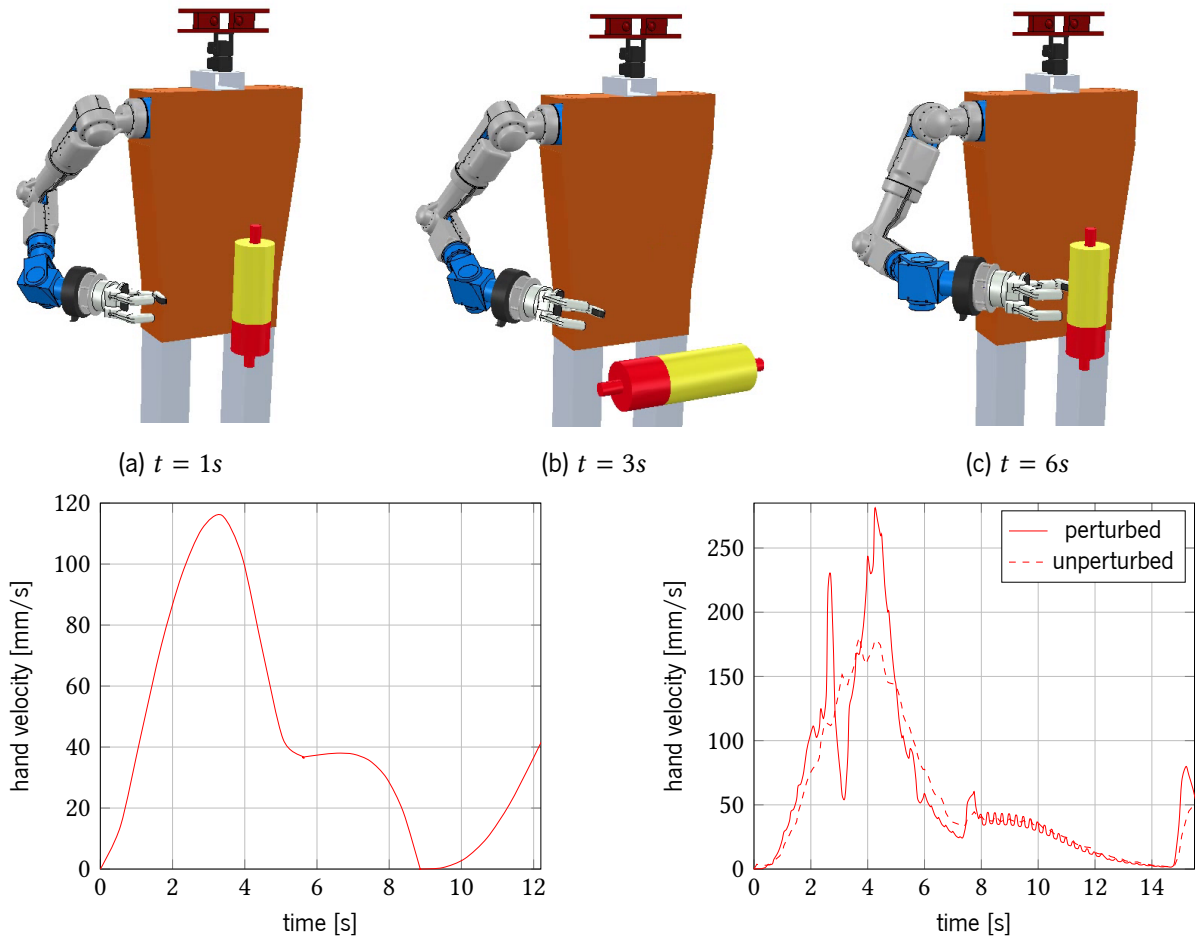


Figure 8.12: Effects of swivel angle tracking on the joints-space trajectory.

Equation 8.3.50 is enabled and otherwise when it is disabled. Moreover, as illustrated in Figure 8.12d, there is a significant smaller difference between the desired planned joints-space trajectory and the actual controlled joints-space trajectory when swivel angle tracking is enabled. In particular, at the moment of grasping this difference is about to vanish indicating a very good match between the planned and the actual target posture of the robotic upper-limb. Therefore, using the redundancy to track a planned swivel angle trajectory increases the actual human-likeness at the level of the joints.

8.4.2.2 Tests under perturbations of the target

Introducing small perturbations on the position and on the orientation of the red column during the execution of the picking task in Figure 8.7a permits to analyse how the designed controllers react. With this purpose, two perturbations occur on the red column as it is shown in Figures 8.13a, 8.13b and 8.13c. At time $t = 2.5s$ the red column is rotated of 90 degrees around its y axis and of 45 degrees around its x axis. Simultaneously, the target also changes position on the xy plane of 44 cm with the respect to its initial configuration. Then, at time $t = 4s$, the red column returns to its start pose. The actual hand velocity profile in Figure 8.13e is significantly modified when these perturbations occur. Specifically, two peaks are formed in the Transport phase of the movement, which results to be spiky and less smooth than desired. On the contrary, the following Approach and Retreat phases substantially track the referenced human-like characteristics because no perturbations of the target are here experienced. As illustrated in Figure 8.14, the positional errors of the proposed controllers are clearly affected at the moment of the disturbances. In particular, at $t = 2.5s$, there is a decrease of the hand position error (Figure 8.14a) because the target suddenly is placed closer the end-effector, while the hand orientation error increases (Figure 8.14b) because the robotic hand should quickly turn of 90 degrees as shown in Figure 8.13b. The finger position error and the swivel angle position error change sign and maintain about the same amplitude as illustrated in Figures 8.14c and 8.14d, respectively. When the second perturbation occur, at $t = 4s$, the amplitude of the errors significantly increases and then returns to shape an unperturbed error profile. These results clearly show a time synchronization of the controllers formulated in Equations 8.3.48, 8.3.49 and 8.3.50. This coupling is due to the time mapping introduced by Equation 8.2.25 and becomes more evident when desired and actual trajectories are plotted (Figure 8.15). At the instants $t = 2.5s$ and $t = 4s$, the desired trajectories are significantly perturbed by the changes on the pose of the target by the mean of the mapping parameter g . For 1.5s, the target position is suddenly closer to the end-effector and, for this reason, the fingers stop opening and start closing to anticipate the grasp transition (see Figures 8.15a and 8.14c). The changes in the target orientation influence more significantly the tracking performance of the controllers in Equations 8.3.52 and 8.3.50. For example, the perturbed final orientation encourages a sudden turn of the robotic hand as the peak in Figure 8.15c shows. The small oscillations in Figure 8.15d



(d) Planned hand velocity in picking (Figures 8.7a and 8.7c)

(e) Actual hand velocity in perturbed picking (Figures 8.13a, 8.13b, 8.13c)

Figure 8.13: Three snapshots of picking the red column under perturbations. The red column changes position and orientation at $t = 2.5s$ and returns to its initial pose at $t = 4s$. Compared to the planned trajectory, the actual hand velocity shows more peaks in the *Transport* phase of the pick movement, which are due to the perturbations on the hand target.

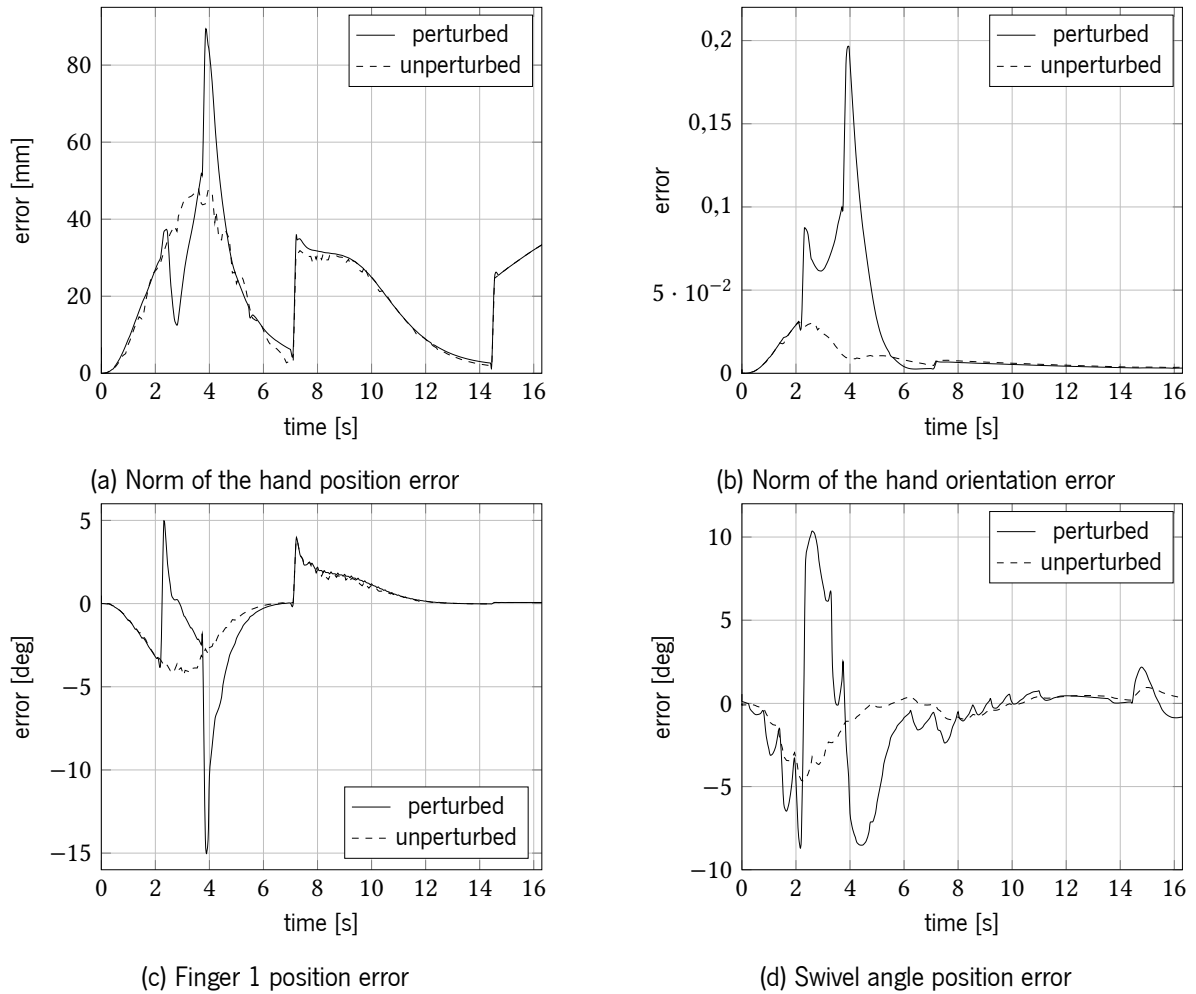
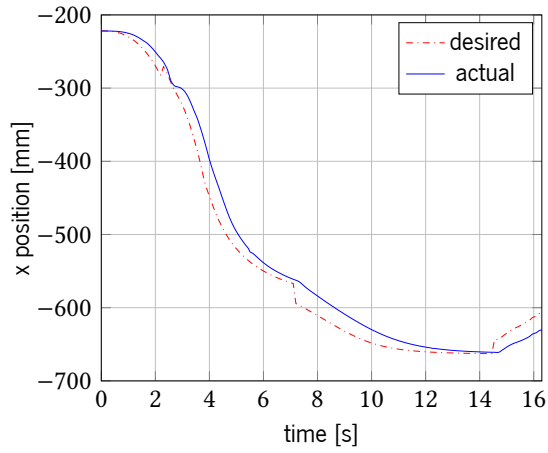
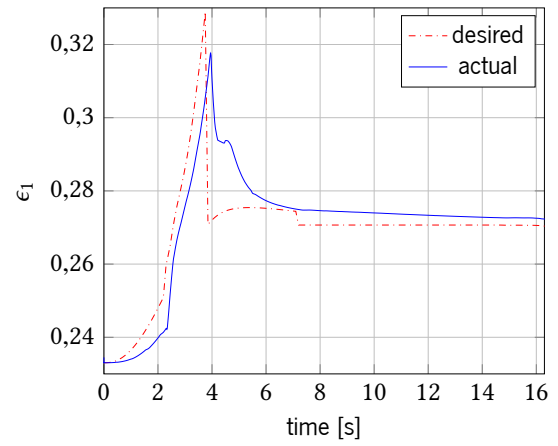
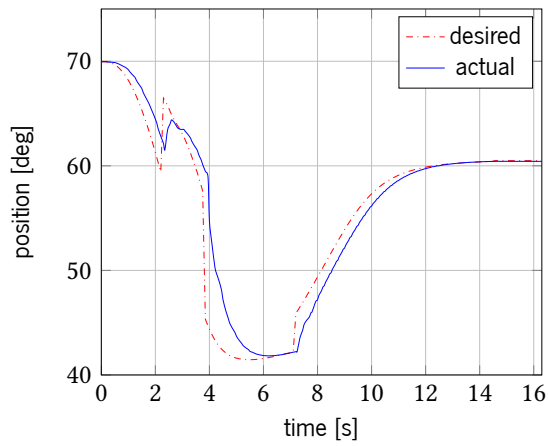


Figure 8.14: Errors during the perturbed tracking of a human-like pick movement (Figure 8.13).

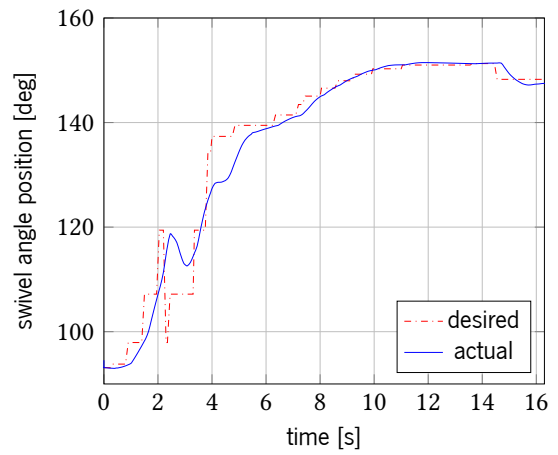
exhibit the attempt of tracking smaller values of the swivel angle, which would raise the elbow and permit a different hand orientation for the new final grip. To sum up, the perturbations generally decrease the qualitative level of human-likeness that is required at the planning stage, but the designed controllers have good capabilities of synchronous adaptation to sudden disturbances that may occur on the desired hand target.



(a) Tracking x hand position


 (b) Tracking hand quaternion ϵ_1


(c) Tracking finger 1 position



(d) Swivel angle tracking enabled

Figure 8.15: Desired and actual trajectories during the perturbations shown in Figures 8.13a, 8.13b and 8.13c.

Part V

Conclusions

Chapter 9

Conclusions

The goal of this PhD project was to design and implement a comprehensive modular framework for the generation of human-like upper-limb movements. Human motion principles of optimization, learning and control inspired the design and the development of separated and interconnected modules with the purpose of mimicking human motor control and respond to the demands of the forthcoming human-centred robotics (Burdet et al., 2013; Schaal et al., 2007; Schwartz, 2016). Due to the lack of complete and consistent human-like movement planning techniques, most of this project research has focused on the investigation and the improvement of human-like mechanisms of manipulation and of naturalistic obstacle-avoidance (see chapters 5 and 6). Additionally, chapters 7 and 8 have introduced methods to cope with predictable and unpredictable variations of a scenario, respectively, by adapting the motor behaviour of a robot. Concerning the proposed planning module, conclusions are discussed in section 9.1, while the developed techniques of learning and controlling are summarized in sections 9.2 and 9.3, respectively. Few insights on possible future directions of investigations are also described in relation with each module.

9.1 Planning

A Human-like Upper-limb Motion Planner (HUMP) was proposed in chapter 5 and tested as it is described in chapter 6. The algorithm is formulated for moving anthropomorphic robotic arm-hand setups and it is deeply inspired from human motor control strategies (chapter 3) to generate human-like motion that enhances human-robot interaction. Following the conceptual ideas of the posture-based motion planning model (Rosenbaum et al., 2001), the proposed planner is principally based on the following assumptions:

1. The planning process, which is featured by a sequence of non-linear constrained optimization problems, is backward starting with the selection of a target posture followed by the selection of a joints-space trajectory towards it.

2. A task is defined as a hierarchy of conditions constrained in the operational space, which take into account the minimization of the angular jerk of motion and the respect of the physical joint limits.
3. Collisions with objects in the workspace are avoided in a naturalistic manner by the superimposition of a direct movement towards a target posture with a back-and-forth movement towards a bounce posture.
4. The application of the Fitts' law (Galen and Jong, 1995) on the overall angular displacement on the trajectory ensures a human-like time parametrization of the planned path. Neglecting the time dimension of human motion may lead to unrealistic and non-human velocities.
5. Evidences on the temporal separation of goal-directed upper-limb movements (Sing Bing Kang and Ikeuchi, 1994) are integrated by mimicking the typical hand kinematical features that are shown during the execution of accurate motion. (Burdet and Milner, 1998; Milner, 1992).

The proposed planning procedures make only use of the direct kinematics of a manipulator (Appendix A) and geometrically model the robot and the objects in its workspace with enclosed surfaces, which wrap different shapes based on their major dimensions (Appendix B). Firstly designed for humanoid platforms (e.g., the robot ARoS in Figure 4.1a), this modellization has also been upgraded to be suitable for a generic 7-DOFs manipulator such as that one of the collaborative robot Sawyer (Figure 4.3). Although this simplification of the world may lack of precision in presence of narrow passages, it increases the speed of convergence during the selection of optimal arm configurations. The HUMP algorithm was also endowed with the capability of generating rectilinear motion in the operational space of the manipulator, which might reveal to be necessary in environments densely cluttered with obstacles (section 5.5). Moreover, the resultant framework for motion planning was also extended to include the possibility of generating synchronous bi-manual human-like movements for anthropomorphic robotic devices (section 5.6).

The selection of human-like upper-limb target postures overcomes typical issues of the Newton-Raphson method applied for iterative inverse kinematics algorithms, such as the convergence failures due to joints limits and the inactivity in presence of local solutions (Beeson and Ames, 2015). Unlike the HUMP, other approaches to human-like arm-hand kinematics are based on examples of human-recorded trajectories (Gielniak et al., 2013; Liarokapis et al., 2017; Zanchettin et al., 2011). These approaches need to solve the correspondence problem for mapping agents with different embodiments. Moreover, motion capture data may not be available or difficult to record for a specific task. This does not exclude, however, that existing information about human poses may be integrated into the proposed HUMP algorithm to bias, for instance, the selection of target and bounce postures.

The HUMP algorithm and its extensions were successfully tested as described in chapter 6 considering a

fully-constrained operational space in different tasks, which range from simple reaching (or move) movements to more complex pick-and-place sequences. Importantly, a preliminary comparison with probabilistic algorithms, which have been proven to be very effective for motion planning in high-dimensional spaces (see the MoveIt! library¹), showed limited solving times that are favourable for fluent interactions with human partners. An obvious difference is that probabilistic planners may fail to return a solution even when it exists, whereas the deterministic HUMP will always find a feasible human-like trajectory when target and bounce postures are defined.

Since in the robotics community there is no generally accepted metric to quantify the similarity between robot and human arm movements, our work can be seen as a contribution towards a possible standard. We adopted smoothness measures – the normalized jerk score, NJS, and the number of movement units, NMU (see section 3.3) – that have been used in the past to discriminate between well-coordinated and impaired human motion (Chang et al., 2005; Chang et al., 2008). Additionally, the ability of the proposed planner to reproduce the motion invariant known as the one-sixth power law (Equation 3.3.2) was also verified. This law of biological upper-limb motion relates the hand velocity with the curvature and the torsion of a given human hand path (Pollick et al., 2009). The statistical results of the experiments with obstacle-free move movements revealed that the slope of the fitted line (95% CI: $[-0.175, -0.172]$) matches quite well the power-law exponent $-1/6$ with a fit score of $R^2 = 0.96$ (Figure 6.4). The relatively small deviation from the mathematically derived constant equi-affine speed law may be explained by the non-oscillatory nature of the generated trajectories. Since spatial movements that are consistent with this law are easier to predict for a human observer than non-biological motion (Maoz and Flash, 2014), it can be expected that the movements generated by the HUMP will support more natural and fluent human-robot interaction.

The capacity of the HUMP to generate interconnected sequences of object-directed movements was also tested with the robots ARoS and Sawyer in a paradigm adopted from our previous work on action coordination during collaborative tasks (Bicho et al., 2011a; Bicho et al., 2011b; Sousa et al., 2015). The assembly paradigm is segmented in a pick, a place and a move movement. The movement phases generated by the HUMP appeared in general not jerky with smooth transitions and few stops of the end-effector. The quantitative smoothness measures applied for each individual movement confirmed this observation. The HUMP consistently achieved a NMU score of one and performed very well in the jerk-based measure with a NJS in general small according to the motion being planned.

The household task of serving a drink (section 6.3.2) introduced new challenges for the proposed planner, in particular for the human-like obstacle avoidance mechanism. The task was segmented into six movements, which included the avoidance of an external object during the transportation of a bottle towards its pouring position and the avoidance of the receiving glass during the pouring and post-pouring actions. The concatenation of the movement phases revealed the same picture concerning the qualitative smoothness

evaluation as found in the assembly task. The HUMP generated trajectories with human-like values of NMU and NJS for each individual movement phase and consequently also for the sequence as a whole.

Computational cost is a major concern of all optimization approaches to trajectory planning. Therefore, a first comparison with some probabilistic planners of the MoveIt! library is experimented. Considering all individual movements, the HUMP ranked always better than RRT* and PRM* and often comparably with RRT, RRTConnect and PRM. In conclusion, the generation of motion with human-like features does not come at the expense of a significant slower planning process.

The results of the subjective ratings of naïve users watching ARoS in action are consistent with the idea that the smoothness measures applied for the quantitative evaluation are good predictors for perceiving a robot action as human-like or not (section 6.3.3). The HUMP got a human-like score of 92% for the drink serving scenario and 72% for the assembly task. Since no further information was requested from the participants, one can only speculate why the simpler assembly sequence got a lower score than the longer and more complex drink serving sequence. Future user studies will address the impact of biological versus non-biological robot motion on human behaviours applying time-based evaluation metrics for joint action tasks (e.g., reaction/waiting time).

The fact that testing the HUMP with the assembly and household tasks was not sufficiently exhaustive in describing the mechanism of a bi-phasic hand velocity profile in presence of obstacles, motivated to additionally investigate the naturalistic obstacle avoidance behaviour. The experimental design of a recent study with human subjects that addressed this topic (Grimme et al., 2012) was firstly replicated. The results obtained with the HUMP qualitatively reproduced a main finding of the study: a decomposition of the tangential hand velocity in a transport and a lift/decent component revealed that a zero crossing of the latter component is responsible for the nature of the speed profile. A bi-modal profile occurs when the lift/decent component is prominent which may depend on the height of the obstacle but also on other more cognitive factors, such as accuracy and safety margins.

The generation of rectilinear hand motion was also successfully tested in a scenario where a pick-and-place task needs to be performed in the narrow space of a shelf ledge. In such a task, the hand is recommended to move very precisely on a straight line with a small constant velocity towards its target. This mode permits to avoid obstacles and bias the normal detection of collisions. Results showed the feasibility of the proposed technique and a constant hand velocity profile of the *Approach* and *Retreat* phases during the manipulation. Therefore, the human-like kinematic features of a movement can only be maintained during a *Transport* phase, but such a task is too constrained in the operational space to draw positive conclusions concerning the naturalistic appearance of the action. Compared to the other probabilistic planners under examination, the HUMP generally demanded more time to generate a trajectory when the straight-line hand mode is enabled.

There are several issues of the presented HUMP algorithm that need further attention. The geometrical models for the objects to be manipulated, the obstacles and the robot body are not precise enough to cover all planning problems that are physically feasible. In particular, when obstacles are in close proximity to a target object, the present version of the HUMP may not find a grasping pose even if the robotic system is in principle able to reach the target. Another issue concerns the extension of the HUMP algorithms to cover robotic platforms that are not fully humanoids. Even though some degree of generalization was formulated and tested on the robot Sawyer, modelling generic n -DOFs manipulators with $n > 7$ and variable dimensions of the links remains to be addressed. For instance, it can be expected that a detached industrial robotic arm with human-like joint configuration may facilitate to some extent human-robot interactions if it moves in a way that resembles a human arm movement (De Momi et al., 2016; Liarokapis et al., 2017). It should be noticed, however, that the benefit of human-like motion seems to be significantly larger for an artificial agent which also mimics the human form compared to an industrial robot (Chaminade et al., 2005; Glasauer et al., 2010).

While the experiments with the robots ARoS and Sawyer have shown that the computational cost of the proposed HUMP algorithms is sufficiently small to permit fluent and efficient interactions with human partners, possibilities to further speed up the planning processes need to be explored. For instance, an open-source alternative to **AMPL** (e.g., JuMP by Dunning et al. (2017)) could be integrated into the HUMP C++ code to avoid external system calls to modelling programming software modules. The selection of the target and the bounce postures, which are formulated as non-linear constrained optimization problems, are certainly crucial planning steps that may constitute a bottle neck for the overall performance of the HUMP. For this reason, the conversion of the hard constraints into soft constraints should be experimented to achieve the same quality of human-likeness in less time. Due to the non-convex nature of the constraints, the risk of being trapped in local colliding solutions is always present. Therefore, a fast collisions detection should be investigated to endow the proposed planner with the capability of eventually restarting the optimization with a different initialization. As described by Schulman et al. (2014), the measure of continuous collision cost based on signed distances between shapes showed very good performances and, for this reason, its application on the generation of human-like paths should be investigated. Similarly to the solution presented by Kalakrishnan et al. (2011), a gradient-free and stochastic version of the HUMP should also be investigated for a better search of the configuration space, which improves the general success rate and reduces the probability of getting stuck in local minima. This direction of research would also allow to explore and take inspiration from the stochastic variations of the human motor behaviour that are witnessed in the execution of similar tasks (Burdet et al., 2013; Scholz and Schöner, 1999).

Generally speaking, the proposed HUMP algorithms can be upgraded in different manners. For example,

being a common requirement in the robotics community, future work should also investigate ways to integrate the possibility of setting way-points along a human-like trajectory. The resulting motion is required to maintain the expected biological kinematic features that are observed in reaching tasks (Flash and Hogan, 1985; Viviani and Flash, 1995). The application of the minimum torque principle (Uno et al., 1989) on different manipulative tasks should also be investigated to identify and use a complete dynamic model of a manipulator and explore the demands of physical human-robot interaction (Noohi et al., 2016). The capability of generating human-like bi-manual synchronous movements needs to be upgraded to safely apply a humanoid robot in collaborative transporting tasks that involve human co-workers. In such tasks, a flexible and reactive coupling between the generation of synchronous and asynchronous bi-manual motion would be auspicious to increase the possibilities of joint manipulation and, therefore, should also be investigated.

9.2 Learning

In the attempt of mimicking the experiential learning cycle that characterizes human training through active experimentation (Kolb, 1984), an adaptive Human-like Upper-limb Posture Learner (HUPL) was proposed and tested in chapter 7. Inspired from the trajectory prediction technique introduced by Jetchev and Toussaint (2013), the HUPL relies on the collections of situation–solution pairs, which are related to a target posture selection problem (5.1.4)–(5.1.7) with the scope of providing an efficient initialization. The presented learner incrementally (cycle after cycle) trains the optimal parameters of a similarity metric that indicates the distance between task-related redundant situations of the robot. The values of these parameters gradually extract the important situational features that can be exclusively considered when an initialization for a novel situation is queried. The incremental training is designed to minimize the expected error returned by the solver IPOPT, which is launched for a limited number of iterations with different initializations. Locally weighted regression is adopted to predict a low-cost initialization for “unseen” situations because, like the human brain, it has the capability of learning robustly against the interferences of new incoming data (Lowe, 1995). Time weighted forgetting of the data was also integrated for an efficient management of the memory size and to allow adaptation to consistent changes in new experienced situations without the need of re-training (Salganicoff, 1993a). Preliminary experiments of the proposed HUPL were executed with the robot ARoS in a simple reaching task (Figure 7.5) for the selection of a human-like target posture. Results showed that datasets with only 90 exemplars were sufficient for the proposed optimal light weighted regressor to minimize the loss value in less than 10 training cycles. When compared with an Euclidean light weighted regressor and a random initializer, the proposed HUPL demonstrated to be capable of offering initializations that lead the optimizer towards high-quality solutions at the minimum computational cost. The application of a time forgetting policy showed a fast adaptation

to new perturbed situations with a constant initial configuration of the robot and, as expected, the trained metric could maintain similar performances in a slightly modified reaching task (Figure 7.11). Although there was no need of re-training, the proposed posture learner could start new training cycles from the latest metrics if, for instance, an increase of the loss value was detected. In other words, the advantage of the proposed learning system is its on-line adaptability to new incoming situations, which does not always require new training sessions. However, the implementation of a cyclic experiential learning paradigm allows to start further sessions of training based on the new experience acquired from the selection of the latest human-like target postures.

Despite the obtained promising results, the HUPL is in its early stage of development and future work should proceed to different directions. For instance, different forgetting policies should be investigated to better study the degree of adaptation to perturbations of different nature and intensity (Salganicoff, 1993b). The proposed learning technique should also be upgraded to predict low-cost initializations for the selection of bounce postures formulated in the problem (5.1.17)–(5.1.20). In this case, particular attention is necessary in designing a situational descriptor that considers the variations on the number of steps in relation with possible perturbations of the environment. A more compact descriptor of task-related and environmental situations should also be investigated to gain independence on the number of the present obstacles and extract information from the obstacles-free workspace (see, e.g. Jetchev and Toussaint, 2013). Importantly, clustering the operational space of the manipulator according to possible perturbations of the hand target pose should be experimented. This preliminary step pre-processes the input data to improve the training performance of multiple internal learning models (see, e.g., Park and Kim, 2010). The presence of a plurality of models also finds support on the most recent findings in human adaptation to generic temporal evolutions. Indeed, the human motor control system also entitles different internal models to implement adaptation mechanisms and submit them to processes of constant refinements in order to cope with external perturbations (Ivaldi et al., 2012).

9.3 Controlling

Unexpected perturbations of the workspace of a robot during the execution of a movement demand the capability of interpreting feedback signals and react on-line to adjust a planned trajectory. A constant redefinition of a planned human-like movement is designed and tested with the robot ARoS in chapter 8 with the introduction of three controllers on three key points of action: the end-effector, the fingers and the swivel angle. The feed-back control loop of the human motor control schematic in Figure 8.1 is replicated in the attempt to maintain the desired kinematic human-like features that are defined at the planning stage. A joints-space trajectory generated by the HUMP (chapter 5) is transferred onto the operational space of a

7-DOFs arm to obtain a human-like reference of the end-effector motion, while the redundancy is used to satisfy important constraints with second priority. Being an important factor for a human-like redundancy resolution (Su et al., 2018), the planned trajectory of the swivel angle is also controlled without interferences on the hand motion tracking, which has the priority for a successful accomplishment of a task. The fingers are controlled in their joints space by following a human-like reference trajectory provided from the HUMP. Overall, the proposed tracking system is composed by three second-order feed-back loop controllers that are timely coupled by the first-order dynamical system defined in Equation 8.2.25 (Schaal et al., 2007). This time mapping system allows to accommodate time in connection with the Euclidean distance with a known target in the operational-space. When sudden perturbations on the target pose occur, the desired trajectories synchronously adapt the reference values provided to the controllers and ensure a meaningful temporal linkage among the movements of the end-effector, the fingers and the swivel angle. Moreover, such management of the time allows a customized time scaling and a flexible transition between two adjacent stages of a movement (Figure 5.2) that can be tuned according to the particular requirements of a pick-and-place task. The global asymptotic stability of the proposed controllers is also briefly analysed and experiments with the humanoid robot ARoS are discussed. In absence of perturbations of the target pose, the presented controllers showed good tracking performances and the ability to replicate the hand human-like kinematic features, which are observed in picking and are typical of a naturalistic obstacle avoidance (Figure 8.9). Additionally, it is demonstrated that, when the swivel angle tracking system is enabled, the actual joints-space trajectory is more homogeneous with the desired joints-space trajectory and almost identical joint values are reached when the hand stops for grasping (Figure 8.12). Tests under perturbations of the target pose reported the synchronous adaptation of the desired trajectories and the capability of the controllers to promptly adjust the actual trajectories accordingly. In this case, conclusions on the resulting degree of human-likeness can be hardly drawn in absence of similar studies in human motor control. However, the main advantages of the proposed tracking system rely on maintaining the kinematic human-like characteristics in static workspaces and on the possibility to some extent replicate human-like arm configurations by controlling in the operational space. This was possible by a time coupling of a desired arm movement with the aperture of the fingers and with a human-like swivel angle motion. In case of perturbations, the controllers proved to be able to redefine a trajectory and accomplish the task successfully.

The presented motion tracker suffers of some limitations that should certainly be addressed in the future investigations. For example, like in the human arm, the contribution of the elbow joint value (i.e., the forth joint of a 7-DOFs arm) to the null space of the Jacobian $\mathcal{N}(J)$ is always 0 (Zanchettin et al., 2011). In other words, when the end-effector is fully constrained in the operational space, then the elbow joint is constrained too. Therefore, when perturbations on the target pose occur, the respect of the physical

limits of the elbow joint cannot be satisfied by the designed controllers and potentially dangerous configurations cannot be avoided. Due to the calculus of the atan2 function in Equation 8.3.33, discontinuities on the desired swivel angle trajectory may take place. Therefore, different formulations of the swivel angle should be investigated to avoid the generation of jerky null-space motion. Techniques that can suppress the null-space motion as the end-effector stops should also be addressed in the future work to maintain a certain degree of human-like synchrony in the joints space (Breteler and Meulenbroek, 2006) and improve the stability of the self motion (see, e.g. Jiang et al., 2017). A low-cost re-planning strategy should be eventually considered when significant perturbations of the workspace may either seriously degrade the expected level of human-likeness or undermine the feasibility of a given task. Finally, the dynamic model of a robotic manipulator should be considered to improve the tracking performance of the feed-back loop (An et al., 1988) and to add capabilities of direct dynamics feed-forward control that are certainly part of the human motor control schematic (Burdet et al., 2013). In addition, the inclusion of the dynamic parameters becomes essential during synchronous bi-manual manipulation of one rigid object in order to keep constant forces and torques in the operational space of the manipulators (Shin and Kim, 2015).

Bibliography

- Albrecht, S., K. Ramirez-Amaro, F. Ruiz-Ugalde, D. Weikersdorfer, M. Leibold, M. Ulbrich, and M. Beetz (2011). "Imitating human reaching motions using physically inspired optimization principles." In: *2011 11th IEEE-RAS International Conference on Humanoid Robots*. Bled, Slovenia: IEEE, pp. 602–607.
- An, C. H., C. G. Atkeson, and J. M. Hollerbach (1988). *Model-Based Control of a Robot Manipulator*. Cambridge, MA, USA: MIT Press.
- Arimoto, S. and M. Sekimoto (2006). "Human-like movements of robotic arms with redundant DOFs: virtual spring-damper hypothesis to tackle the Bernstein problem." In: *Proceedings 2006 IEEE International Conference on Robotics and Automation*. IEEE, pp. 1860–1866.
- Artemiadis, P. K., P. T. Katsiaris, and K. J. Kyriakopoulos (2010). "A biomimetic approach to inverse kinematics for a redundant robot arm." In: *Autonomous Robots* 29, pp. 293–308.
- Atawnih, A., D. Papageorgiou, and Z. Doulgeri (2014). "Reaching for redundant arms with human-like motion and compliance properties." In: *Robotics and Autonomous Systems* 62, pp. 1731–1741.
- Atkeson, C. G., A. W. Moore, and S. Schaal (1997). "Locally Weighted Learning." In: *Lazy Learning*. Vol. 11. Dordrecht: Springer Netherlands, pp. 11–73.
- Bae, J.-H., W. Yang, D. Kim, Y. Oh, B.-J. You, and S.-R. Oh (2011). "Robotic arm control inspired by human muscle tension effect under the gravity." In: *2011 IEEE International Conference on Robotics and Automation*. IEEE, pp. 1404–1411.
- Beeson, P. and B. Ames (2015). "TRAC-IK: An open-source library for improved solving of generic inverse kinematics." In: *2015 IEEE-RAS 15th International Conference on Humanoid Robots (Humanoids)*. IEEE, pp. 928–935.
- Bekkering, H., E. R. A. de Bruijn, R. H. Cuijpers, R. Newman-Norlund, H. T. van Schie, and R. Meulenbroek (2009). "Joint Action: Neurocognitive Mechanisms Supporting Human Interaction." In: *Topics in Cognitive Science* 1, pp. 340–352.
- Beringer, J. and E. Hüllermeier (2007). "An Efficient Algorithm for Instance-Based Learning on Data Streams." In: *Advances in Data Mining. Theoretical Aspects and Applications*. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 34–48.

- Bhattacharjee, T., Y. Oh, J.-H. Bae, and S.-R. Oh (2011). "Control design for human-like reaching movements using redundancy in robot arm-trunk systems." In: *International Journal of Control, Automation and Systems* 9, pp. 1173–1186.
- Bicho, E., W. Erlhagen, N. Hipolito, E. Costa e Silva, R. Silva, and L. Louro (2011a). "A dynamic field approach to goal inference, error detection and anticipatory action selection in human-robot collaboration." In: *New Frontiers in Human Robots Interaction*, pp. 135–164.
- Bicho, E., L. Louro, N. Hipolito, and W. Erlhagen (2009). "A dynamic field approach to goal inference and error monitoring for human-robot interaction." In: *International Symposium on New Frontiers in Human-Robot Interaction*, pp. 31–37.
- Bicho, E., W. Erlhagen, L. Louro, and E. Costa e Silva (2011b). "Neuro-cognitive mechanisms of decision making in joint action: A human-robot interaction study." In: *Human Movement Science* 30, pp. 846–868.
- Biegstraaten, M., J. B. J. Smeets, and E. Brenner (2003). "The influence of obstacles on the speed of grasping." In: *Experimental Brain Research* 149, pp. 530–534.
- Billard, A. and D. Kragic (2019). "Trends and challenges in robot manipulation." In: *Science* 364.
- Billard, A., S. Calinon, R. Dillmann, and S. Schaal (2008). "Robot Programming by Demonstration." In: *Springer Handbook of Robotics*. Vol. 48. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 1371–1394.
- Bingham, G. P., K. Hughes, and M. Mon-Williams (2008). "The coordination patterns observed when two hands reach-to-grasp separate objects." In: *Experimental brain research* 184, pp. 283–93.
- Bishop, C. M. (2006). *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Berlin, Heidelberg: Springer-Verlag.
- Bisio, A., A. Sciutti, F. Nori, G. Metta, L. Fadiga, G. Sandini, and T. Pozzo (2014). "Motor Contagion during Human-Human and Human-Robot Interaction." In: *PLoS ONE* 9. Ed. by A. Avenanti, e106172.
- Breteler, M. D. K. and R. G. J. Meulenbroek (2006). "Modeling 3D object manipulation: synchronous single-axis joint rotations?" In: *Experimental Brain Research* 168, pp. 395–409.
- Brooks, S., A. Gelman, G. Jones, and X.-L. Meng (2010). *Handbook of Markov Chain Monte Carlo*.
- Burdet, E. and T. E. Milner (1998). "Quantization of human motions and learning of accurate movements." In: *Biological Cybernetics* 78, pp. 307–318.
- Burdet, E., D. W. Franklin, and T. E. Milner (2013). *Human Robotics*. The MIT Press.
- Caccavale, F., S. Chiaverini, and B. Siciliano (1997). "Second-order kinematic control of robot manipulators with Jacobian damped least-squares inverse: theory and experiments." In: *IEEE/ASME Transactions on Mechatronics* 2, pp. 188–194.

- Caggiano, V., A. De Santis, B. Siciliano, and A. Chianese (2006). "A biomimetic approach to mobility distribution for a human-like redundant arm." In: *The First IEEE/RAS-EMBS International Conference on Biomedical Robotics and Biomechatronics, 2006. BioRob 2006*. Vol. 2006. IEEE, pp. 393–398.
- Cardoso de Oliveira, S. (2002). "The neuronal basis of bimanual coordination: recent neurophysiological evidence and functional models." In: *Acta psychologica* 110, pp. 139–59.
- Castiello, U. (2005). "The neuroscience of grasping." In: *Nature Reviews Neuroscience* 6, pp. 726–736.
- Chaminade, T., D. W. Franklin, E. Oztop, and G. Cheng (2005). "Motor interference between Humans and Humanoid Robots: Effect of Biological and Artificial Motion." In: *Proceedings. The 4th International Conference on Development and Learning, 2005*. Vol. 2005. IEEE, pp. 96–101.
- Chang, J.-J., T.-I. Wu, W.-L. Wu, and F.-C. Su (2005). "Kinematical measure for spastic reaching in children with cerebral palsy." In: *Clinical Biomechanics* 20, pp. 381–388.
- Chang, J. J., Y. S. Yang, L. Y. Guo, W. L. Wu, and F. C. Su (2008). "Differences in reaching performance between normal adults and patients post stroke a kinematic analysis." In: *Journal of Medical and Biological Engineering* 28, pp. 53–58.
- Chen, P., H. Zhao, X. Zhao, D. Ge, and H. Ding (2018). "Dimensionality Reduction for Motion Planning of Dual-arm Robots." In: *2018 IEEE International Conference on Mechatronics and Automation (ICMA)*. IEEE, pp. 718–723.
- Chen, W., C. Xiong, and S. Yue (2016). "On Configuration Trajectory Formation in Spatiotemporal Profile for Reproducing Human Hand Reaching Movement." In: *IEEE Transactions on Cybernetics* 46, pp. 804–816.
- Chiaverini, S., B. Siciliano, and O. Egeland (1991). "Redundancy resolution for the human-arm-like manipulator." In: *Robotics and Autonomous Systems* 8, pp. 239–250.
- Costa e Silva, E., J. P. Araújo, D. Machado, M. F. Costa, W. Erlhagen, and E. Bicho (2013). "Generating human-like movements on an anthropomorphic robot using an interior point method." In: *International Conference of Numerical Analysis and Applied Mathematics*, pp. 602–605.
- Costa e Silva, E. (2011). "Reaching, Grasping and Manipulation in Anthropomorphic Robot Systems." Doctoral dissertation. University of Minho.
- Craig, J. J. (2005). *Introduction to Robotics*. Ed. by A. Dworkin. 3rd. Pearson Education Inc.
- De Momi, E., L. Kranendonk, M. Valenti, N. Enayati, and G. Ferrigno (2016). "A Neural Network-Based Approach for Trajectory Planning in Robot-Human Handover Tasks." In: *Frontiers in Robotics and AI* 3, pp. 1–10.
- Dragan, A. and S. Srinivasa (2014). "Integrating human observer inferences into robot motion planning." In: *Autonomous Robots* 37, pp. 351–368.

- Dragan, A. D., G. J. Gordon, and S. S. Srinivasa (2011). "Learning from Experience in Manipulation Planning: Setting the Right Goals." In: *Isrr*, pp. 1–16.
- Dragan, A. D., K. C. T. Lee, and S. S. Srinivasa (2013). "Legibility and predictability of robot motion." In: *2013 8th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*. IEEE, pp. 301–308.
- Drillis, R., R. Contini, and M. Bluestein (1966). *Body Segment Parameters*. Tech. rep. New York.
- Duffy, B. R. (2003). "Anthropomorphism and the social robot." In: *Robotics and Autonomous Systems* 42, pp. 177–190.
- Dunning, I., J. Huchette, and M. Lubin (2017). "JuMP: A Modeling Language for Mathematical Optimization." In: *SIAM Review* 59, pp. 295–320.
- Eira Araújo, S. M. (2014). "Manipulação Bimanual em Robôs Antropomórficos : Movimentos Sincronos." Doctoral dissertation. University of Minho.
- Engelbrecht, S. E. (2001). "Minimum Principles in Motor Control." In: *Journal of Mathematical Psychology* 45, pp. 497–542.
- Erel, H., T. Shem Tov, Y. Kessler, and O. Zuckerman (2019). "Robots are Always Social." In: *Extended Abstracts of the 2019 CHI Conference on Human Factors in Computing Systems - CHI EA '19*. New York, New York, USA: ACM Press, pp. 1–6.
- Ericson, C. (2004). *Real-time collision detection*. Ed. by M. Kaufman. Morgan Kaufman.
- Erlhagen, W. and E. Bicho (2006). "The dynamic neural field approach to cognitive robotics." In: *Journal of Neural Engineering* 3, R36–R54.
- Erlhagen, W. and G. Schöner (2002). "Dynamic field theory of movement preparation." In: *Psychological Review* 109, pp. 545–572.
- Fan, J., J. He, and S. I. H. Tillery (2006). "Control of hand orientation and arm movement during reach and grasp." In: *Experimental Brain Research* 171, pp. 283–296.
- Fitts, P. M. (1954). "The information capacity of the human motor system in controlling the amplitude of movement." In: *Journal of Experimental Psychology* 47, pp. 381–391.
- Flanagan, J. R., M. C. Bowman, and R. S. Johansson (2006). "Control strategies in object manipulation tasks." In: *Current Opinion in Neurobiology* 16, pp. 650–659.
- Flash, T. and N. Hogan (1985). "The Coordination of Arm Movements: An Experimentally Confirmed Mathematical Model." In: *The journal of Neuroscience* 5, pp. 1688–1703.
- Floreano, D., A. J. Ijspeert, and S. Schaal (2014). "Robotics and Neuroscience." In: *Current Biology* 24, R910–R920.
- Fong, T., I. Nourbakhsh, and K. Dautenhahn (2003). "A survey of socially interactive robots." In: *Robotics and Autonomous Systems* 42, pp. 143–166.

- Forsgren, A. (2006). "On Warm Starts for Interior Methods." In: *System Modeling and Optimization*. Vol. 199. Boston: Kluwer Academic Publishers, pp. 51–66.
- Fourer, R. D. M. Gay, and B. W. Kernighan (1989). *AMPL: A mathematical programming language*. Tech. rep.
- Fu, K. C. D., Y. Nakamura, T. Yamamoto, and H. Ishiguro (2012). "Studies of motor synergies in generating optimal goal-directed movements in human-like robotic arm." In: *2012 IEEE International Conference on Robotics and Biomimetics (ROBIO)*. IEEE, pp. 808–813.
- Fuchs, M., C. Borst, P. R. Giordano, A. Baumann, E. Kraemer, J. Langwald, R. Gruber, N. Seitz, G. Plank, K. Kunze, R. Burger, F. Schmidt, T. Wimboeck, and G. Hirzinger (2009). "Rollin' Justin - Design considerations and realization of a mobile platform for a humanoid upper body." In: *2009 IEEE International Conference on Robotics and Automation*. IEEE, pp. 4131–4137.
- Fukuda, T., R. Michelini, V. Potkonjak, S. Tzafestas, K. Valavanis, and M. Vukobratovic (2001). "How far away is "artificial man" ." In: *IEEE Robotics and Automation Magazine* 8, pp. 66–73.
- Galen, G. P. van and W. P. de Jong (1995). "Fitts' law as the outcome of a dynamic noise filtering model of motor control." In: *Human Movement Science* 14, pp. 539–571.
- Garcia, E., M. A. Jimenez, P. G. De Santos, and M. Armada (2007). "The evolution of robotics research." In: *IEEE Robotics and Automation Magazine* 14, pp. 90–103.
- Garcia, N., J. Roseli, and R. Suarez (2018). "Modeling human-likeness in approaching motions of dual-arm autonomous robots." In: *2018 IEEE International Conference on Simulation, Modeling, and Programming for Autonomous Robots (SIMPAP)*. IEEE, pp. 43–48.
- Gelfand, I. M. and S. V. Fomin (1963). *Calculus of Variations*. Ed. by R. A. Silverman. Englewood Cliffs, N.J.: Prentice-Hall, Inc.
- Gielniak, M., K. Liu, and A. Thomaz (2013). "Generating human-like motion for robots." In: *The International Journal of Robotics Research* 32, pp. 1275–1301.
- Gladden, M. E. (2019). "Who Will Be the Members of Society 5.0? Towards an Anthropology of Technologically Posthumanized Future Societies." In: *Social Sciences* 8, p. 148.
- Glasauer, S., M. Huber, P. Basili, A. Knoll, and T. Brandt (2010). "Interacting in time and space: Investigating human-human and human-robot joint action." In: *19th International Symposium in Robot and Human Interactive Communication*. IEEE, pp. 252–257.
- Gondzio, J. and A. Grothey (2002). "Reoptimization With the Primal-Dual Interior Point Method." In: *SIAM Journal on Optimization* 13, pp. 842–864.
- Gouaillier, D., V. Hugel, P. Blazevic, C. Kilner, J. Monceaux, P. Lafourcade, B. Marnier, J. Serre, and B. Maisonnier (2008). "The NAO humanoid: a combination of performance and affordability." In: *CoRR*, pp. 1–10.

- Grimme, B., J. Lipinski, and G. Schöner (2012). "Naturalistic arm movements during obstacle avoidance in 3D and the identification of movement primitives." In: *Experimental Brain Research* 222, pp. 185–200.
- Guiard, Y. (1993). "On Fitts's and Hooke's laws: Simple harmonic movement in upper-limb cyclical aiming." In: *Acta Psychologica* 82, pp. 139–159.
- Gulletta, G., S. M. Araújo, E. Costa e Silva, M. F. Costa, W. Erlhagen, and E. Bicho (2015). "Nonlinear optimization for human-like synchronous movements of a dual arm-hand robotic system." In: *International Conference of Numerical Analysis and Applied Mathematics*, p. 140007.
- Gulletta, G., W. Erlhagen, and E. Bicho (2020). "Human-Like Arm Motion Generation: A Review." In: *Robotics* 9, p. 102.
- Gulletta, G., E. C. e Silva, W. Erlhagen, R. Meulenbroek, M. F. P. Costa, and E. Bicho (2021). "A Human-like Upper-limb Motion Planner: Generating naturalistic movements for humanoid robots." In: *International Journal of Advanced Robotic Systems* 18.
- Hassani, S. (1999). *Mathematical physics: a modern introduction to its foundations*. New York.
- Hughes, C. M. L., P. Reissig, and C. Seegelke (2011). "Motor planning and execution in left- and right-handed individuals during a bimanual grasping and placing task." In: *Acta psychologica* 138, pp. 111–8.
- Hugues, O., V. Weistroffer, A. Paljic, P. Fuchs, A. A. Karim, T. Gaudin, and A. Buendia (2016). "Determining the Important Subjective Criteria in the Perception of Human-Like Robot Movements Using Virtual Reality." In: *International Journal of Humanoid Robotics* 13, p. 1550033.
- Hyunchul Kim, L. M. Miller, A. Al-Refai, M. Brand, and J. Rosen (2011). "Redundancy resolution of a human arm for controlling a seven DOF wearable robotic system." In: *2011 Annual International Conference of the IEEE Engineering in Medicine and Biology Society*. IEEE, pp. 3471–3474.
- Ijspeert, A. J., J. Nakanishi, H. Hoffmann, P. Pastor, and S. Schaal (2013). "Dynamical movement primitives: learning attractor models for motor behaviors." In: *Neural computation* 25, pp. 328–73.
- Ivaldi, S., O. Sigaud, B. Berret, and F. Nori (2012). "From Humans to Humanoids: the Optimal Control Framework." In: *Paladyn Journal of Behavioral Robotics* 3, pp. 75–91.
- Jeannerod, M. (1984). "The Timing of Natural Prehension Movements." In: *Journal of Motor Behavior* 16, pp. 235–254.
- Jensen, M., E. Vagnoni, M. Overgaard, and P. Haggard (2014). "Experience of action depends on intention, not body movement: An experiment on memory for mens rea." In: *Neuropsychologia* 55, pp. 122–127.
- Jetchev, N. and M. Toussaint (2009). "Trajectory prediction." In: *Proceedings of the 26th Annual International Conference on Machine Learning - ICML '09*, pp. 1–8.

- Jetchev, N. and M. Toussaint (2013). “Fast motion planning from experience: Trajectory prediction for speeding up movement generation.” In: *Autonomous Robots* 34, pp. 111–127.
- Jiang, P., S. Huang, J. Xiang, and M. Z. Q. Chen (2017). “A Unified Approach for Second-Order Control of the Manipulator With Joint Physical Constraints.” In: *Journal of Mechanisms and Robotics* 9, p. 041009.
- John, E. and E. A. Yildirim (2008). “Implementation of warm-start strategies in interior-point methods for linear programming in fixed dimension.” In: *Computational Optimization and Applications* 41, pp. 151–183.
- Kalakrishnan, M., S. Chitta, E. Theodorou, P. Pastor, and S. Schaal (2011). “STOMP: Stochastic trajectory optimization for motion planning.” In: *2011 IEEE International Conference on Robotics and Automation*. IEEE, pp. 4569–4574.
- Kamper, D. G., A. N. McKenna-Cole, L. E. Kahn, and D. J. Reinkensmeyer (2002). “Alterations in reaching after stroke and their relation to movement direction and impairment severity.” In: *Archives of Physical Medicine and Rehabilitation* 83, pp. 702–707.
- Kanazawa, A., J. Kinugawa, and K. Kosuge (2019). “Adaptive Motion Planning for a Collaborative Robot Based on Prediction Uncertainty to Enhance Human Safety and Work Efficiency.” In: *IEEE Transactions on Robotics* 35, pp. 817–832.
- Kaneko, K., F. Kanehiro, S. Kajita, H. Hirukawa, T. Kawasaki, M. Hirata, K. Akachi, and T. Isozumi (2004). “Humanoid robot HRP-2.” In: *IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA '04. 2004*. IEEE, pp. 1083–1090.
- Karaman, S. and E. Frazzoli (2011). “Sampling-based algorithms for optimal motion planning.” In: *The International Journal of Robotics Research* 30, pp. 846–894.
- Kashima, T. and K. Hori (2016). “Control of biomimetic robots based on analysis of human arm trajectories in 3D movements.” In: *Artificial Life and Robotics* 21, pp. 24–30.
- Kavraki, L. E., P. Švestka, J. C. Latombe, and M. H. Overmars (1996). “Probabilistic roadmaps for path planning in high-dimensional configuration spaces.” In: *IEEE Transactions on Robotics and Automation* 12, pp. 566–580.
- Kawato, M., K. Furukawa, and R. Suzuki (1987). “A hierarchical neural-network model for control and learning of voluntary movement.” In: *Biological Cybernetics* 57, pp. 169–185.
- Kawato, M. (1999). “Internal models for motor control and trajectory planning.” In: *Current Opinion in Neurobiology* 9, pp. 718–727.
- Kelso, J., D. Southard, and D. Goodman (1979). “On the nature of human interlimb coordination.” In: *Science* 203, pp. 1029–1031.

- Kenwright, B. (2012). "A beginners guide to dual-quaternions: What they are, how they work, and how to use them for 3D character hierarchies." In: *20th International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision, WSCG 2012 - Conference Proceedings*, pp. 1–10.
- Kim, C., S. Kim, S. Ra, and B. J. You (2007). "Regenerating human-like arm motions of humanoid robots for a movable object." In: *SICE Annual Conference 2007*. IEEE, pp. 1081–1086.
- Kim, S., C. Kim, and J. Park (2006). "Human-like Arm Motion Generation for Humanoid Robots Using Motion Capture Database." In: *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, pp. 3486–3491.
- Kolb, D. A. (1984). *Experiential learning: experience as the source of learning and development*. Englewood Cliffs, N.J.: Prentice-Hall.
- Koppenborg, M., P. Nickel, B. Naber, A. Lungfiel, and M. Huelke (2017). "Effects of movement speed and predictability in human-robot collaboration." In: *Human Factors and Ergonomics in Manufacturing and Service Industries 27*, pp. 197–209.
- Koskinopoulou, M. and P. Trahanias (2016). "A methodological framework for robotic reproduction of observed human actions: Formulation using latent space representation." In: *2016 IEEE-RAS 16th International Conference on Humanoid Robots (Humanoids)*. IEEE, pp. 565–572.
- Kreutz-Delgado, K., M. Long, and H. Seraji (1990). "Kinematic analysis of 7 DOF anthropomorphic arms." In: *Proceedings., IEEE International Conference on Robotics and Automation*. IEEE Comput. Soc. Press, pp. 824–830.
- Kuffner, J. J. and S. M. LaValle (2000). "RRT-connect: An efficient approach to single-query path planning." In: *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No.00CH37065)*. Vol. 2. IEEE, pp. 995–1001.
- Kupferberg, A., S. Glasauer, M. Huber, M. Rickert, A. Knoll, and T. Brandt (2011). "Biological movement increases acceptance of humanoid robots as human partners in motor interaction." In: *AI and SOCIETY 26*, pp. 339–345.
- Kupferberg, A., M. Huber, B. Helfer, C. Lenz, A. Knoll, and S. Glasauer (2012). "Moving Just Like You: Motor Interference Depends on Similar Motility of Agent and Observer." In: *PLoS ONE 7*. Ed. by N. Wenderoth, pp. 1–8.
- Lacquaniti, F., C. Terzuolo, and P. Viviani (1983). "The law relating the kinematic and figural aspects of drawing movements." In: *Acta Psychologica 54*, pp. 115–130.
- Lampariello, R., D. Nguyen-Tuong, C. Castellini, G. Hirzinger, and J. Peters (2011). "Trajectory planning for optimal robot catching in real-time." In: *2011 IEEE International Conference on Robotics and Automation*. IEEE, pp. 3719–3726.

- Lamperti, C., A. M. Zanchettin, and P. Rocco (2015). "A redundancy resolution method for an anthropomorphic dual-arm manipulator based on a musculoskeletal criterion." In: *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, pp. 1846–1851.
- Lauretti, C., F. Cordella, and L. Zollo (2019). "A Hybrid Joint/Cartesian DMP-Based Approach for Obstacle Avoidance of Anthropomorphic Assistive Robots." In: *International Journal of Social Robotics* 11, pp. 783–796.
- LaValle, S. M. (1998). *Rapidly Exploring Random Trees: A New Tool For Path Planning*. Tech. rep., p. 302.
- Lewis, F. L., D. L. Vrabie, and V. L. Syrmos (2012). *Optimal Control*. Hoboken, NJ, USA: John Wiley and Sons, Inc.
- Li, B., Y. F. Zheng, H. Hemami, and Da Che (2013). "Human-like robotic handwriting and drawing." In: *2013 IEEE International Conference on Robotics and Automation*. IEEE, pp. 4942–4947.
- Liarokapis, M., C. P. Bechlioulis, P. K. Artemiadis, and K. J. Kyriakopoulos (2017). "Deriving Humanlike Arm Hand System Poses." In: *Journal of Mechanisms and Robotics* 9, pp. 1–11.
- Liu, W., D. Chen, and J. Steil (2017). "Analytical Inverse Kinematics Solver for Anthropomorphic 7-DOF Redundant Manipulators with Human-Like Configuration Constraints." In: *Journal of Intelligent and Robotic Systems* 86, pp. 63–79.
- Liu, Y., J. Zhao, and B. Xie (2010). "Obstacle avoidance for redundant manipulators based on a Novel Gradient Projection Method with a functional scalar." In: *2010 IEEE International Conference on Robotics and Biomimetics*. IEEE, pp. 1704–1709.
- Lommertzen, J., E. Costa e Silva, R. H. Cuijpers, and R. G. J. Meulenbroek (2009). "Collision-Avoidance Characteristics of Grasping." In: *Anticipatory Behavior in Adaptive Learning Systems*. Springer, pp. 188–208.
- Lowe, D. G. (1995). "Similarity Metric Learning for a Variable-Kernel Classifier." In: *Neural Computation* 7, pp. 72–85.
- Lynch, K. M. and F. C. Park (2017). *Modern Robotics: Mechanics, Planning, and Control*. Cambridge University Press.
- Maoz, U. and T. Flash (2014). "Spatial constant equi-affine speed and motion perception." In: *Journal of Neurophysiology* 111, pp. 336–349.
- Martins de Sá, S. F. (2018). "Planeamento de Movimentos Compreensíveis pelo Humano para o Robô Sawyer." Doctoral dissertation. University of Minho.
- Maurice, P., M. E. Huber, N. Hogan, and D. Sternad (2018). "Velocity-Curvature Patterns Limit Human–Robot Physical Interaction." In: *IEEE Robotics and Automation Letters* 3, pp. 249–256.

- McColl, D., A. Hong, N. Hatakeyama, G. Nejat, and B. Benhabib (2016). "A Survey of Autonomous Human Affect Detection Methods for Social Robots Engaged in Natural HRI." In: *Journal of Intelligent and Robotic Systems* 82, pp. 101–133.
- Meulenbroek, R., D. Rosenbaum, C. Jansen, J. Vaughan, and S. Vogt (2001). "Multijoint grasping movements." In: *Experimental Brain Research* 138, pp. 219–234.
- Milner, T. E. (1992). "A model for the generation of movements requiring endpoint precision." In: *Neuroscience* 49, pp. 487–496.
- Morasso, P. (1981). "Spatial control of arm movements." In: *Experimental Brain Research* 42, pp. 223–227.
- Morsella, E., J. A. Bargh, and P. M. Gollwitzer (2009). *Oxford handbook of human action*. Oxford University Press.
- Nakanishi, J., R. Cory, M. Mistry, J. Peters, and S. Schaal (2008). "Operational Space Control: A Theoretical and Empirical Comparison." In: *The International Journal of Robotics Research* 27, pp. 737–757.
- Napier, J. R. (1956). "THE PREHENSILE MOVEMENTS OF THE HUMAN HAND." In: *The Journal of Bone and Joint Surgery. British volume* 38B, pp. 902–913.
- Nocedal, J. and S. J. Wright (1999). *Numerical Optimization*. New York.
- Noohi, E., M. Zefran, and J. L. Patton (2016). "A Model for Human-Human Collaborative Object Manipulation and Its Application to Human-Robot Interaction." In: *IEEE Transactions on Robotics* 32, pp. 880–896.
- Pan, J., Z. Chen, and P. Abbeel (2014). "Predicting initialization effectiveness for trajectory optimization." In: *2014 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, pp. 5183–5190.
- Park, C., J. Pan, and D. Manocha (2012). "ITOMP: Incremental trajectory optimization for real-time replanning in dynamic environments." In: *Proceedings of the International Conference on Automated Planning and Scheduling*, pp. 207–215.
- Park, G.-R. and C. Kim (2010). "Constructing of optimal database structure by imitation learning based on evolutionary algorithm." In: *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, pp. 2698–2703.
- Pattacini, U., F. Nori, L. Natale, G. Metta, and G. Sandini (2010). "An experimental evaluation of a novel minimum-jerk cartesian controller for humanoid robots." In: *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, pp. 1668–1674.
- Pollick, F. E., U. Maoz, A. A. Handzel, P. J. Gliblin, G. Sapiro, and T. Flash (2009). "Three-dimensional arm movements at constant equi-affine speed." In: *Cortex* 45, pp. 325–339.

- Quigley, M., K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, E. Berger, R. Wheeler, and A. Ng (2009). "ROS: an open-source Robot Operating System." In: *ICRA Workshop on Open Source Software*. Vol. 3, p. 5.
- Rano, I. and I. Iossifidis (2013). "Modelling human arm motion through the attractor dynamics approach." In: *2013 IEEE International Conference on Robotics and Biomimetics (ROBIO)*. IEEE, pp. 2088–2093.
- Ray, C., F. Mondada, and R. Siegwart (2008). "What do people expect from robots?" In: *Proceedings 2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*. Nice: IEEE, pp. 3816–3821.
- Rizk, L. (2011). "Learning by Doing: Toward an Experiential Approach to Professional Development." In: *IFLA*, pp. 1–6.
- Roby-Brami, A., S. Jacobs, N. Bennis, and M. F. Levin (2003). "Hand orientation for grasping and arm joint rotation patterns in healthy subjects and hemiparetic stroke patients." In: *Brain Research* 969, pp. 217–229.
- Rohmer, E., S. P. N. Singh, and M. Freese (2013). "V-REP: A versatile and scalable robot simulation framework." In: *Proceedings 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, pp. 1321–1326.
- Rosado, J., F. Silva, V. Santos, and Z. Lu (2013). "Reproduction of human arm movements using Kinect-based motion capture data." In: *2013 IEEE International Conference on Robotics and Biomimetics (ROBIO)*. IEEE, pp. 885–890.
- Rosell, J., R. Suarez, A. Perez, and C. Rosales (2011). "Including virtual constraints in motion planning for anthropomorphic hands." In: *2011 IEEE International Symposium on Assembly and Manufacturing (ISAM)*. IEEE, pp. 1–6.
- Rosenbaum, D. A. (1990). *Attention and Performance XIII*. Ed. by M. Jeannerod. Psychology Press.
- Rosenbaum, D. A., L. D. Loukopoulos, R. G. J. Meulenbroek, J. Vaughan, and S. E. Engelbrecht (1995). "Planning reaches by evaluating stored postures." In: *Psychological Review* 102, pp. 28–63.
- Rosenbaum, D. A., R. G. J. Meulenbroek, J. Vaughan, and C. Jansen (1999). "Coordination of reaching and grasping by capitalizing on obstacle avoidance and other constraints." In: *Experimental Brain Research* 128, pp. 92–100.
- Rosenbaum, D. A., R. J. Meulenbroek, J. Vaughan, and C. Jansen (2001). "Posture-based motion planning: Applications to grasping." In: *Psychological Review* 108, pp. 709–734.
- Rosenbaum, D. A., R. G. Cohen, R. G. J. Meulenbroek, and J. Vaughan (2006). "Plans for Grasping Objects." In: *Motor Control and Learning*. Boston: Springer US, pp. 9–25.
- Rosenbaum, D. A., R. G. Cohen, A. M. Dawson, S. A. Jax, R. G. Meulenbroek, R. van der Wel, and J. Vaughan (2009). "The Posture-Based Motion Planning Framework: New Findings Related to Object

- Manipulation, Moving Around Obstacles, Moving in Three Spatial Dimensions, and Haptic Tracking.” In: *Progress in Motor Control*. Ed. by D. Sternad. Vol. 957, pp. 485–497.
- Rossi, S., F. Ferland, and A. Tapus (2017). “User profiling and behavioral adaptation for HRI: A survey.” In: *Pattern Recognition Letters* 99, pp. 3–12.
- Salganicoff, M. (1993a). “Density-Adaptive Learning and Forgetting.” In: *Tenth International Conference on Machine Learning*. Kaufmann, Morgan, pp. 276–283.
- (1993b). *Explicit Forgetting Algorithms for Memory Based Learning*. Tech. rep.
- Saling, M., J. Alberts, G. E. Stelmach, and J. R. Bloedel (1998). “Reach-to-grasp movements during obstacle avoidance.” In: *Experimental Brain Research* 118, pp. 251–258.
- Sandini, G., G. Metta, and D. Vernon (2008). “The iCub Cognitive Humanoid Robot: An Open-System Research Platform for Enactive Cognition.” In: *50 Years of Artificial Intelligence*. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 358–369.
- Schaal, S., P. Mohajerian, and A. Ijspeert (2007). “Dynamics systems vs. optimal control - a unifying view.” In: *Progress in brain research*. Ed. by P. Cisek, T. Drew, and J. F. Kalaska. Vol. 165. Elsevier B.V., pp. 425–45.
- Schaal, S. (2002). “Learning Robot Control.” In: *The handbook of brain theory and neural networks*. 2nd ed. Cambridge: MIT Press, pp. 983–987.
- (2007). “The new robotics: towards human-centered machines.” In: *HFSP Journal* 1, pp. 115–126.
- Schaal, S. and D. Sternad (2001). “Origins and violations of the 2/3 power law in rhythmic three-dimensional arm movements.” In: *Experimental Brain Research* 136, pp. 60–72.
- Schmidt, R. A. (1975). “A Schema Theory of Discrete Motor Skill Learning.” In: *Psychological Review*. American Psychological Association Inc.
- Schmidt, R. A., H. Zelaznik, B. Hawkins, J. S. Frank, and J. T. J. Quinn (1979). “Motor-output variability: A theory for the accuracy of rapid motor acts.” In: *Psychological Review* 86, pp. 415–451.
- Scholz, J. P. and G. Schöner (1999). “The uncontrolled manifold concept: identifying control variables for a functional task.” In: *Experimental Brain Research* 126, pp. 289–306.
- Schulman, J., Y. Duan, J. Ho, A. Lee, I. Awwal, H. Bradlow, J. Pan, S. Patil, K. Goldberg, and P. Abbeel (2014). “Motion planning with sequential convex optimization and convex collision checking.” In: *The International Journal of Robotics Research* 33, pp. 1251–1270.
- Schwartz, A. B. (2016). “Movement: How the Brain Communicates with the World.” In: *Cell* 164, pp. 1122–1135.
- Sebanz, N, H Bekkering, and G Knoblich (2006). “Joint action: bodies and minds moving together.” In: *Trends in Cognitive Sciences* 10, pp. 70–76.
- Shahzad, A. (2010). “A warm-start interior-point method for predictive control.” In: *Proceedings of UKACC*.

- Sheridan, T. B. (2016). "Human-Robot Interaction: Status and Challenges." In: *Human Factors: The Journal of the Human Factors and Ergonomics Society* 58, pp. 525–532.
- Shin, S. Y. and C. Kim (2015). "Human-Like Motion Generation and Control for Humanoid's Dual Arm Object Manipulation." In: *IEEE Transactions on Industrial Electronics* 62, pp. 2265–2276.
- Shin, S. Y., Jun won Lee, and C. Kim (2012). "Humanoid's dual arm object manipulation based on virtual dynamics model." In: *2012 IEEE International Conference on Robotics and Automation*. IEEE, pp. 2599–2604.
- Shukla, A. and A. Billard (2012). "Coupled dynamical system based arm-hand grasping model for learning fast adaptation strategies." In: *Robotics and Autonomous Systems* 60, pp. 424–440.
- Siciliano, B., L. Sciacivico, L. Villani, and G. Oriolo (2009). *Robotics*. 1st. Advanced Textbooks in Control and Signal Processing. London: Springer London.
- Silva, E. C., M. F. Costa, E Bicho, and W Erlhagen (2011). "Human-Like Movement of an Anthropomorphic Robot: Problem Revisited." In: *AIP Conference Proceedings*. Vol. 1389, pp. 779–782.
- Silva, E. C., M. F. Costa, J. P. Araújo, D Machado, L Louro, W Erlhagen, and E Bicho (2015). "Towards human-like bimanual movements in anthropomorphic robots : a nonlinear optimization." In: *Applied Mathematics and Information Sciences* 9, pp. 619–629.
- Silva, R., E. Bicho, and W. Erlhagen (2008). "AROS: an anthropomorphic robot for human-robot interaction and coordination studies." In: *CONTROLO. Conference - 8th Portuguese Conference on Automatic Control, UTAD*, pp. 819–826.
- Sing Bing Kang and K. Ikeuchi (1994). "Determination of motion breakpoints in a task sequence from human hand motion." In: *Proceedings of the 1994 IEEE International Conference on Robotics and Automation*. IEEE Comput. Soc. Press, pp. 551–556.
- Sousa, E., W. Erlhagen, F. Ferreira, and E. Bicho (2015). "Off-line simulation inspires insight: A neurodynamics approach to efficient robot task learning." In: *Neural Networks* 72, pp. 123–139.
- Srinivasa, S. S., D. Berenson, M. Cakmak, A. Collet, M. R. Dogar, A. D. Dragan, R. A. Knepper, T. Niemueller, K. Strabala, M. Vande Weghe, and J. Ziegler (2012). "Herb 2.0: Lessons Learned From Developing a Mobile Manipulator for the Home." In: *Proceedings of the IEEE* 100, pp. 2410–2428.
- Stefanovic, F. and H. L. Galiana (2014). "An adaptive spinal-like controller: tunable biomimetic behavior for a robotic limb." In: *BioMedical Engineering OnLine* 13, p. 151.
- Strait, M. K., V. A. Floerke, W. Ju, K. Maddox, J. D. Remedios, M. F. Jung, and H. L. Urry (2017). "Understanding the Uncanny: Both Atypical Features and Category Ambiguity Provoke Aversion toward Humanlike Robots." In: *Frontiers in Psychology* 8, pp. 1–17.
- Strauss, S. and D. Heinke (2012). "A Robotics-Based Approach to Modeling of Choice Reaching Experiments on Visual Attention." In: *Frontiers in Psychology* 3, pp. 1–11.

- Stulp, F., E. A. Theodorou, and S. Schaal (2012). "Reinforcement Learning With Sequences of Motion Primitives for Robust Manipulation." In: *IEEE Transactions on Robotics* 28, pp. 1360–1370.
- Su, H., N. Enayati, L. Vantadori, A. Spinoglio, G. Ferrigno, and E. De Momi (2018). "Online human-like redundancy optimization for tele-operated anthropomorphic manipulators." In: *International Journal of Advanced Robotic Systems* 15, p. 172988141881469.
- Su, H., W. Qi, C. Yang, A. Aliverti, G. Ferrigno, and E. De Momi (2019). "Deep Neural Network Approach in Human-Like Redundancy Optimization for Anthropomorphic Manipulators." In: *IEEE Access* 7, pp. 124207–124216.
- Suarez, R., J. Rosell, and N. Garcia (2015). "Using synergies in dual-arm manipulation tasks." In: *2015 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, pp. 5655–5661.
- Sucan, I. A., M. Moll, and L. E. Kavraki (2012). "The Open Motion Planning Library." In: *IEEE Robotics and Automation Magazine* 19, pp. 72–82.
- Tanizaki, Y., F. Jimenez, T. Yoshikawa, and T. Furuhashi (2018). "Impression Investigation of Educational Support Robots using Sympathy Expression Method by Body Movement and Facial Expression." In: *2018 Joint 10th International Conference on Soft Computing and Intelligent Systems (SCIS) and 19th International Symposium on Advanced Intelligent Systems (ISIS)*. IEEE, pp. 1254–1258.
- Taylor, R. H. (1979). "Planning and Execution of Straight Line Manipulator Trajectories." In: *IBM Journal of Research and Development* 23, pp. 424–436.
- Täix, M., M. T. Tran, P. Souères, and E. Guigon (2013). "Generating human-like reaching movements with a humanoid robot: A computational approach." In: *Journal of Computational Science* 4, pp. 269–284.
- Todorov, E. (2004). "Optimality principles in sensorimotor control." In: *Nature neuroscience* 7, pp. 907–15.
- Todorov, E. and M. I. Jordan (2002). "Optimal feedback control as a theory of motor coordination." In: *Nature neuroscience* 5, pp. 1226–35.
- Togo, S., T. Kagawa, and Y. Uno (2016). "Uncontrolled Manifold Reference Feedback Control of Multi-Joint Robot Arms." In: *Frontiers in Computational Neuroscience* 10, pp. 1–18.
- Tomić, M., C. Chevallereau, K. Jovanović, V. Potkonjak, and A. Rodić (2018). "Human to humanoid motion conversion for dual-arm manipulation tasks." In: *Robotica* 36, pp. 1167–1187.
- Uno, Y., M. Kawato, and R. Suzuki (1989). "Formation and Control of Optimal Trajectory in Human Multijoint Arm Movement - Minimum Torque-Change Model." In: *Cybernetics, Biological* 61, pp. 89–101.
- Vaughan, J., D. A. Rosenbaum, and R. G. J. Meulenbroek (2006). "Modeling Reaching and Manipulating in 2- and 3-D Workspaces: The Posture-Based Model." In: *International Conference on Development and Learning (ICDL)*. Vol. 1. Bloomington, pp. 3–8.

- Vaughan, J., D. A. Rosenbaum, and R. G. J. Meulenbroek (2001). "Planning Reaching and Grasping Movements: The Problem of Obstacle Avoidance." In: *Motor Control*. Human Kinetics Publishers, pp. 116–135.
- Vaughan, J., D. A. Barany, A. W. Sali, S. A. Jax, and D. A. Rosenbaum (2010). "Extending Fitts' Law to three-dimensional obstacle-avoidance movements: support for the posture-based motion planning model." In: *Experimental Brain Research* 207, pp. 133–138.
- Virtanen, P., R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright, S. J. van der Walt, M. Brett, J. Wilson, K. J. Millman, N. Mayorov, A. R. J. Nelson, E. Jones, R. Kern, E. Larson, C. J. Carey, İlhan Polat, Y. Feng, E. W. Moore, J. VanderPlas, D. Laxalde, J. Perktold, R. Cimrman, I. Henriksen, E. A. Quintero, C. R. Harris, A. M. Archibald, A. H. Ribeiro, F. Pedregosa, and P. van Mulbregt (2020). "SciPy 1.0: fundamental algorithms for scientific computing in Python." In: *Nature Methods* 17, pp. 261–272.
- Viviani, P. and T. Flash (1995). "Minimum-jerk, two-thirds power law, and isochrony: converging approaches to movement planning." In: *Journal of Experimental Psychology: Human Perception and Performance* 21, pp. 32–53.
- Vu, H. M. (2012). "Control of an anthropomorphic manipulator involved in physical Human-Robot Interaction." Doctoral dissertation.
- Wachter, A. and L. T. Biegler (2005). "On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming." In: *Mathematical Programming* 106, pp. 25–57.
- Wada, Y. and M. Kawato (2004). "A via-point time optimization algorithm for complex sequential trajectory formation." In: *Neural Networks* 17, pp. 353–364.
- Wada, Y., Y. Kaneko, E. Nakano, R. Osu, and M. Kawato (2001). "Quantitative examinations for multi joint arm trajectory planning—using a robust calculation algorithm of the minimum commanded torque change trajectory." In: *Neural Networks* 14, pp. 381–393.
- Wang, K. and T. K. Lien (1989). "The planning of a straight line trajectory via interactive computer graphics." In: *Robotics and Computer Integrated Manufacturing* 5, pp. 215–221.
- Wang, S., S. O. Lilienfeld, and P. Rochat (2015). "The uncanny valley: Existence and explanations." In: *Review of General Psychology* 19, pp. 393–407.
- Wann, J., I. Nimmo-Smith, and A. M. Wing (1988). "Relation between velocity and curvature in movement: Equivalence and divergence between a power law and a minimum-jerk model." In: *Journal of Experimental Psychology: Human Perception and Performance* 14, pp. 622–637.
- Welfare, K. S., M. R. Hallowell, J. A. Shah, and L. D. Riek (2019). "Consider the Human Work Experience When Integrating Robotics in the Workplace." In: *2019 14th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*. IEEE, pp. 75–84.

- Wiese, E., G. Metta, and A. Wykowska (2017). "Robots as intentional agents: Using neuroscientific methods to make robots appear more social." In: *Frontiers in Psychology* 8, pp. 1–19.
- Xia, J., Z. Jiang, H. Liu, and H. Cai (2014). "Analytical inverse kinematic computation for anthropomorphic manipulator based on human-like motion optimization and maximum reachable region optimization." In: *2014 IEEE International Conference on Robotics and Biomimetics (ROBIO 2014)*. IEEE, pp. 2292–2297.
- Xie, B., J. Zhao, and Y. Liu (2011). "Human-like motion planning for robotic arm system." In: *2011 15th International Conference on Advanced Robotics (ICAR)*. IEEE, pp. 88–93.
- Xu, H. and X. Ding (2017). "A synergic method for anthropomorphic dual-arm robots to plan bimanual transport tasks." In: *2017 IEEE International Conference on Robotics and Biomimetics (ROBIO)*. IEEE, pp. 1020–1025.
- Yamaguchi, A., C. G. Atkeson, S. Niekum, and T. Ogasawara (2014). "Learning pouring skills from demonstration and practice." In: *2014 IEEE-RAS International Conference on Humanoid Robots*. IEEE, pp. 908–915.
- Yang, W., J.-H. Bae, Y. Oh, N. Y. Chong, and B. J. You (2010). "CPG based self-adapting multi-DOF robotic arm control." In: *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, pp. 4236–4243.
- Yildirim, E. A. and S. J. Wright (2002). "Warm-Start Strategies in Interior-Point Methods for Linear Programming." In: *SIAM Journal on Optimization* 12, pp. 782–810.
- You, B.-J., D. Kim, C. Kim, Y.-H. Oh, M.-H. Jeong, and S.-R. Oh (2008). "Network-based Humanoid 'MAHRU' as Ubiquitous Robotic Companion." In: *IFAC Proceedings Volumes* 41, pp. 724–729.
- Yuan, J. S. (1988). "Closed-loop manipulator control using quaternion feedback." In: *IEEE Journal on Robotics and Automation* 4, pp. 434–440.
- Zacharias, F., C. Schlette, F. Schmidt, C. Borst, J. Rossmann, and G. Hirzinger (2011). "Making planned paths look more human-like in humanoid robot manipulation planning." In: *2011 IEEE International Conference on Robotics and Automation*. IEEE, pp. 1192–1198.
- Zanchettin, A. M., P. Rocco, L. Bascetta, I. Symeonidis, and S. Peldschus (2011). "Kinematic analysis and synthesis of the human arm motion during a manipulation task." In: *2011 IEEE International Conference on Robotics and Automation*. IEEE, pp. 2692–2697.
- Zanchettin, A. M., L. Bascetta, and P. Rocco (2013). "Achieving Humanlike Motion: Resolving Redundancy for Anthropomorphic Industrial Manipulators." In: *IEEE Robotics and Automation Magazine* 20, pp. 131–138.
- Zhang, G. and P. Yu (2018). "Lyapunov method for stability of descriptor second-order and high-order systems." In: *Journal of Industrial and Management Optimization* 14, pp. 673–686.

- Zhao, J. and Y. Wei (2017). "A Novel Algorithm of Human-Like Motion Planning for Robotic Arms." In: *International Journal of Humanoid Robotics* 14, p. 1650023.
- Zhao, J., B. Xie, and C. Song (2014). "Generating human-like movements for robotic arms." In: *Mechanism and Machine Theory* 81, pp. 107–128.
- Zucker, M., N. Ratliff, A. D. Dragan, M. Pivtoraiko, M. Klingensmith, C. M. Dellin, J. A. Bagnell, and S. S. Srinivasa (2013). "CHOMP: Covariant Hamiltonian optimization for motion planning." In: *The International Journal of Robotics Research* 32, pp. 1164–1193.

Part VI

Appendices

Appendix A

Kinematics

In this chapter, the kinematics of the robots ARoS and Sawyer are described in details. In particular, section A.1 presents the direct kinematics for the right and the left arms of the robot ARoS, while in section A.2 the direct kinematics of the robot Sawyer is explained. In relation with the purposes of this project, section A.3 reports the direct and the inverse kinematics of the robotic hand BH8-262 that is mounted on both robots.

A.1 Arms direct kinematics of the robot ARoS

The **Anthropomorphic Robotic System** (ARoS) has two 7-DOFs arms that are respectively equipped with a three-fingered Barrett Hand (Figure 4.1a). Like the human arms, the manipulators have one degree of redundancy and can be represented as a series of links that are connected by joints. As Figures A.1 and A.2 illustrate, the direct kinematics of the right and the left arms are described with the popular Denavit-Hartenberg (D-H) convention (Craig, 2005). The corresponding parameters are collected in the Tables A.1 and A.2, where L_1, L_u, L_l, L_h are the physical dimensions of both arms.

i	$\alpha_{i-1}[deg]$	$a_{i-1}[mm]$	$d_i[mm]$	$\theta_i[deg]$
1	90	0	L_1	θ_1
2	90	0	0	θ_2
3	-90	0	L_u	θ_3
4	90	0	0	θ_4
5	-90	0	L_l	θ_5
6	90	0	0	θ_6
7	-90	0	L_h	θ_7

Table A.1: D-H parameters of the right arm

i	$\alpha_{i-1}[deg]$	$a_{i-1}[mm]$	$d_i[mm]$	$\theta_i[deg]$
12	90	0	L_1	θ_{12}
13	90	0	0	θ_{13}
14	-90	0	L_u	θ_{14}
15	90	0	0	θ_{15}
16	-90	0	L_l	θ_{16}
17	90	0	0	θ_{17}
18	-90	0	L_h	$\theta_{18} - 180$

Table A.2: D-H parameters of the left arm

The transformation matrix from the world frame at the base of the robot (Figure 4.1b) to the origin frame

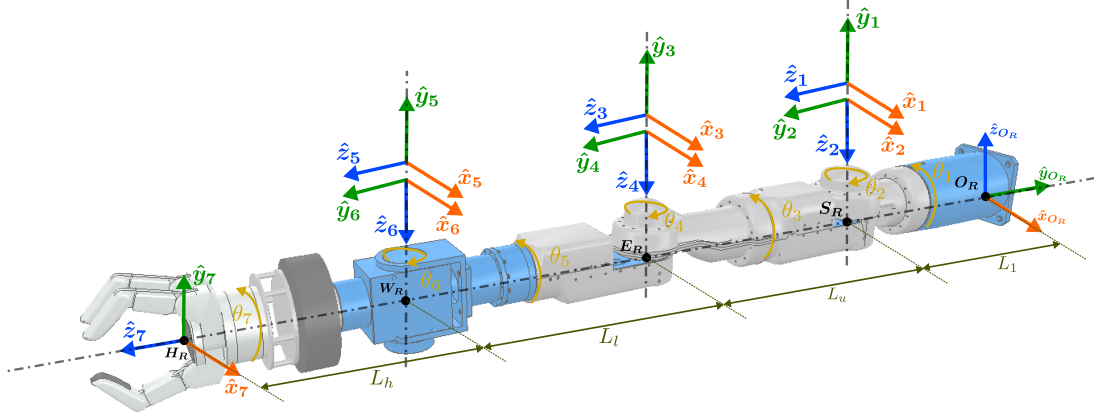


Figure A.1: Right arm in home posture and reference frames assigned according to the Denavit-Hartenberg convention.

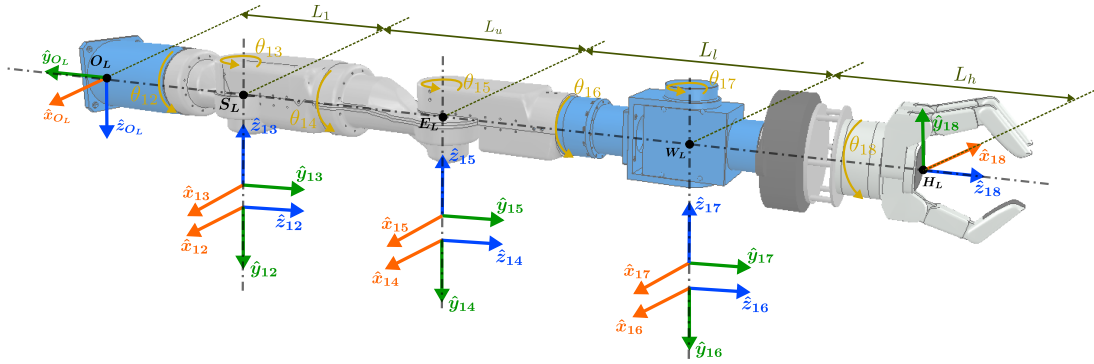


Figure A.2: Left arm in home posture and reference frames assigned according to the Denavit-Hartenberg convention.

of the right arm is given by Equation A.1.1, while Equation A.1.2 described the transformation matrix from the world frame to the origin of the left arm. Similarly, Equations A.1.3 and A.1.4 show the transformations from the origins of the arms to their first frame.

$${}^W_{O_R}T = \begin{bmatrix} -1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 50 \\ 0 & 0 & 1 & 1500 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (\text{A.1.1})$$

$${}^W_{O_L}T = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & -50 \\ 0 & 0 & -1 & 1500 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (\text{A.1.2})$$

$${}^{O_R}_1T = \begin{bmatrix} c_1 & -s_1 & 0 & 0 \\ 0 & 0 & -1 & -L_1 \\ s_1 & c_1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (\text{A.1.3})$$

$${}^{O_L}_{12}T = \begin{bmatrix} c_{12} & -s_{12} & 0 & 0 \\ 0 & 0 & -1 & -L_1 \\ s_{12} & c_{12} & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (\text{A.1.4})$$

A local frame $\{\hat{\mathbf{x}}_i, \hat{\mathbf{y}}_i, \hat{\mathbf{z}}_i\}$, or simply $\{i\}$, ($\forall i \in \{2, \dots, 7\} \cup \{13, \dots, 18\}$) is assigned to each joint of the arms. A generic transformation from frame $\{i-1\}$ to frame $\{i\}$ is described by the D-H parameters $a_{i-1}, \alpha_{i-1}, d_i, \theta_i$ (in Tables A.1 and A.2) and it is given by the transformation matrix showed in Equation A.1.5.

$${}^{i-1}_i T(\theta_i) = \begin{bmatrix} c_i & -s_i & 0 & a_{i-1} \\ s_i c^{i-1} & c_i c^{i-1} & -s^{i-1} & -s^{i-1} d_i \\ s_i s^{i-1} & c_i s^{i-1} & c^{i-1} & c^{i-1} d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad \forall i \in \{2, \dots, 7\} \cup \{13, \dots, 18\} \quad (\text{A.1.5})$$

where $c_i = \cos(\theta_i)$, $s_i = \sin(\theta_i)$, $c^{i-1} = \cos(\alpha_{i-1})$, $s^{i-1} = \sin(\alpha_{i-1})$. We note that the parameters $a_{i-1}, \alpha_{i-1}, d_i$ are geometrical constants, while θ_i are the variable angular positions of the joints.

As reported by Equations A.1.6 and A.1.7, the direct kinematics for the right and the left arms is computed as linked chains of transformation matrices.

$${}^W_7 T(\theta_1, \theta_2, \dots, \theta_7) = {}^W_{O_R} T \cdot {}^{O_R}_1 T \cdot {}^1_2 T \cdot \dots \cdot {}^6_7 T \quad (\text{A.1.6})$$

$${}^W_{18} T(\theta_{12}, \theta_{13}, \dots, \theta_{18}) = {}^W_{O_L} T \cdot {}^{O_L}_{12} T \cdot {}^{12}_{13} T \cdot \dots \cdot {}^{17}_{18} T \quad (\text{A.1.7})$$

The first three elements of the first three columns of ${}^W_7 T$ provide the axes of the frame $\{7\}$ ($\hat{\mathbf{x}}_7, \hat{\mathbf{y}}_7, \hat{\mathbf{z}}_7$) as functions of $(\theta_1, \theta_2, \dots, \theta_7)$. The first three elements of the fourth column of ${}^W_7 T$ gives the Cartesian coordinates of the origin of the local frame $\{7\}$ as a function of $(\theta_1, \theta_2, \dots, \theta_7)$. A similar representation of the frame $\{18\}$ and its origin can also be described for the left arm.

The direct kinematics chains described in Equations A.1.6 and A.1.7 allows to compute the positions of the human-like key points along the arms, as it is illustrated in Figures A.1 and A.2. In the following Equations we use the notation $[T]_{*,j}$ to denote the first three elements of the j -column of the matrix T . The Equations A.1.8, A.1.9, A.1.10 and A.1.11 describe how to compute the position of the right shoulder ($\mathbf{S}_R$), of the right elbow ($\mathbf{E}_R$), of the right wrist ($\mathbf{W}_R$) and of the right hand ($\mathbf{H}_R$), respectively.

$$\mathbf{S}_R = \left[{}^W_{O_R} T \cdot {}^{O_R}_1 T \right]_{*,4} \quad (\text{A.1.8})$$

$$\mathbf{E}_R(\theta_1, \theta_2) = \left[{}^W_{O_R} T \cdot {}^{O_R}_1 T \cdot {}^1_2 T \cdot {}^2_3 T \right]_{*,4} \quad (\text{A.1.9})$$

$$\mathbf{W}_R(\theta_1, \theta_2, \theta_3, \theta_4) = \left[{}^W_{O_R} T \cdot {}^{O_R}_1 T \cdot \dots \cdot {}^4_5 T \right]_{*,4} \quad (\text{A.1.10})$$

$$\mathbf{H}_R(\theta_1, \dots, \theta_7) = \left[{}^W_{O_R} T \cdot {}^{O_R}_1 T \cdot \dots \cdot {}^6_7 T \right]_{*,4} \quad (\text{A.1.11})$$

Similarly for the left arm, the Equations A.1.12, A.1.13, A.1.14 and A.1.15 describe how to compute the position of the left shoulder ($\mathbf{S}_L$), of the left elbow ($\mathbf{E}_L$), of the left wrist ($\mathbf{W}_L$) and of the left hand ($\mathbf{H}_L$),

respectively.

$$\mathbf{S}_L = \left[{}^W_{O_L}T \cdot {}^{O_L}_{12}T \right]_{*,4} \quad (\text{A.1.12})$$

$$\mathbf{E}_L(\theta_1, \theta_2) = \left[{}^W_{O_L}T \cdot {}^{O_L}_{12}T \cdot {}^{12}_{13}T \cdot {}^{13}_{14}T \right]_{*,4} \quad (\text{A.1.13})$$

$$\mathbf{W}_L(\theta_1, \theta_2, \theta_3, \theta_4) = \left[{}^W_{O_L}T \cdot {}^{O_L}_{12}T \cdots {}^{15}_{16}T \right]_{*,4} \quad (\text{A.1.14})$$

$$\mathbf{H}_L(\theta_1, \dots, \theta_7) = \left[{}^W_{O_L}T \cdot {}^{O_L}_{12}T \cdots {}^{17}_{18}T \right]_{*,4} \quad (\text{A.1.15})$$

The axes of the reference frame $\{7\}$ on the palm of the right hand (Figure A.1) are given by Equations A.1.16, A.1.17 and A.1.18. The axes of the reference frame $\{18\}$ on the palm of the left hand (Figure A.2) are given by Equations A.1.19, A.1.20 and A.1.21.

$$\hat{\mathbf{x}}_7(\theta_1, \dots, \theta_7) = \left[{}^W_{O_R}T \cdots {}^6_7T \right]_{*,1} \quad (\text{A.1.16})$$

$$\hat{\mathbf{y}}_7(\theta_1, \dots, \theta_7) = \left[{}^W_{O_R}T \cdots {}^6_7T \right]_{*,2} \quad (\text{A.1.17})$$

$$\hat{\mathbf{z}}_7(\theta_1, \dots, \theta_7) = \left[{}^W_{O_R}T \cdots {}^6_7T \right]_{*,3} \quad (\text{A.1.18})$$

$$\hat{\mathbf{x}}_{18}(\theta_{12}, \dots, \theta_{18}) = \left[{}^W_{O_L}T \cdots {}^{17}_{18}T \right]_{*,1} \quad (\text{A.1.19})$$

$$\hat{\mathbf{y}}_{18}(\theta_{12}, \dots, \theta_{18}) = \left[{}^W_{O_L}T \cdots {}^{17}_{18}T \right]_{*,2} \quad (\text{A.1.20})$$

$$\hat{\mathbf{z}}_{18}(\theta_{12}, \dots, \theta_{18}) = \left[{}^W_{O_L}T \cdots {}^{17}_{18}T \right]_{*,3} \quad (\text{A.1.21})$$

A.2 Arm direct kinematics of the robot Sawyer

The arm of the robot Sawyer has 7 degrees of freedom and can be represented as a series of rigid links connected by revolute joints. Like in humans, the arm has one degree of redundancy and can be featured by key points for a better human-like geometrical shaping. Figure A.3 illustrates the positions of these key points along the arm and the reference frames that are assigned according to the Denavit-Hartenberg (D-H) convention (Craig, 2005). Table A.3 collects the corresponding D-H parameters, where $A_1, L_1, L_2, L_3, L_4, L_5, L_6, L_7$ are the physical dimensions of the manipulator. We note that the parameters $a_{i-1}, \alpha_{i-1}, d_i$ are geometrical constants, while θ_i are the variable angular positions of the joints.

The direct kinematics chain from the reference frame $\{0\}$ at the base of the manipulator to the reference frame $\{7\}$ on the palm of the end-effector is computed as a linked chain of transformation matrices (Equation A.2.1). Generally, a transformation from frame $\{i-1\}$ to frame $\{i\}$ is given by Equation A.2.2.

$${}^0_7T(\theta_1, \theta_2, \dots, \theta_7) = {}^0_1T \cdot {}^1_2T \cdots {}^6_7T \quad (\text{A.2.1})$$

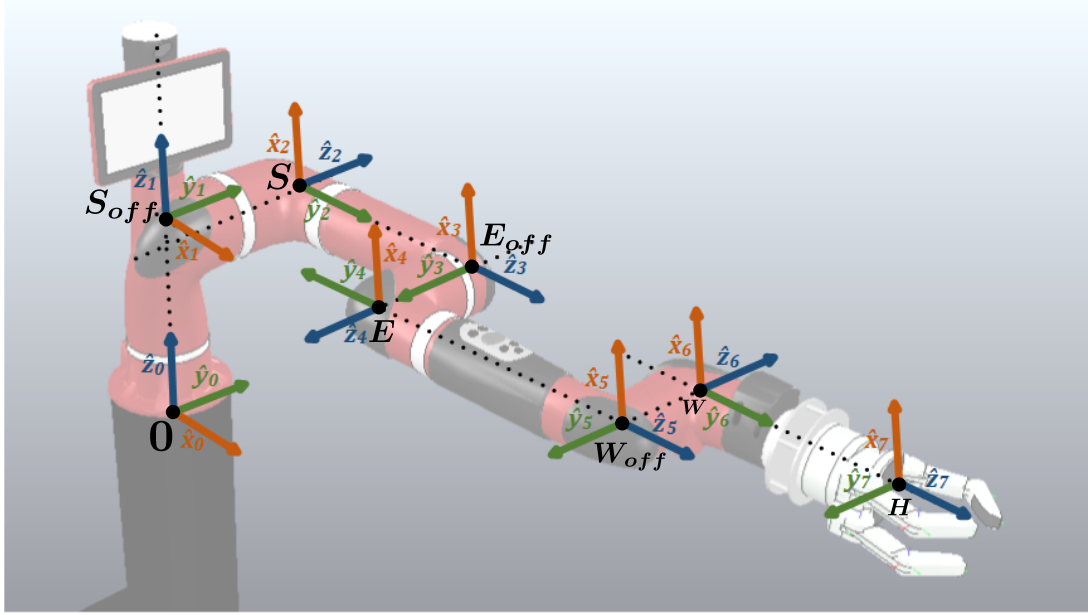


Figure A.3: The Sawyer arm in home posture and reference frames assigned according to the Denavit-Hartenberg convention.

i	$\alpha_{i-1}[deg]$	$a_{i-1}[mm]$	$d_i[mm]$	$\theta_i[deg]$
1	0	0	L_1	θ_1
2	-90	A_1	L_2	$\theta_2 - 90$
3	-90	0	L_3	θ_3
4	90	0	L_4	θ_4
5	-90	0	L_5	θ_5
6	90	0	L_6	θ_6
7	-90	0	L_7	θ_7

Table A.3: D-H parameters of the Sawyer arm

$${}^{i-1}T(\theta_i) = \begin{bmatrix} c_i & -s_i & 0 & a_{i-1} \\ s_i c^{i-1} & c_i c^{i-1} & -s^{i-1} & -s^{i-1} d_i \\ s_i s^{i-1} & c_i s^{i-1} & c^{i-1} & c^{i-1} d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad \forall i \in \{1, \dots, 7\} \quad (\text{A.2.2})$$

where $c_i = \cos(\theta_i)$, $s_i = \sin(\theta_i)$, $c^{i-1} = \cos(\alpha_{i-1})$, $s^{i-1} = \sin(\alpha_{i-1})$. In the following Equations we use the notation $[T]_{*,j}$ to denote the first three elements of the j -column of the matrix T . The Equations A.2.3 and A.2.4 describe how to compute the position of the shoulder offset (S_{off}) and of the shoulder (S), respectively. Similarly, Equations A.2.5 and A.2.6 describe how to compute the position of the elbow offset (E_{off}) and of the elbow (E), respectively. Equations A.2.7 and A.2.8 describe how to compute the

position of the wrist offset ($\mathbf{W}_{off}$) and of the wrist ($\mathbf{W}$), respectively.

$$\mathbf{S}_{off}(\theta_1) = \begin{bmatrix} 0 \\ 1 \end{bmatrix} T_{*,4} \quad (\text{A.2.3})$$

$$\mathbf{S}(\theta_1, \theta_2) = \begin{bmatrix} 0 \\ 1 \end{bmatrix} T \cdot \begin{bmatrix} 1 \\ 2 \end{bmatrix} T_{*,4} \quad (\text{A.2.4})$$

$$\mathbf{E}_{off}(\theta_1, \theta_2, \theta_3) = \begin{bmatrix} 0 \\ 1 \end{bmatrix} T \cdot \begin{bmatrix} 1 \\ 2 \end{bmatrix} T \cdot \begin{bmatrix} 2 \\ 3 \end{bmatrix} T_{*,4} \quad (\text{A.2.5})$$

$$\mathbf{E}(\theta_1, \theta_2, \theta_3, \theta_4) = \begin{bmatrix} 0 \\ 1 \end{bmatrix} T \cdot \begin{bmatrix} 1 \\ 2 \end{bmatrix} T \cdot \begin{bmatrix} 2 \\ 3 \end{bmatrix} T \cdot \begin{bmatrix} 3 \\ 4 \end{bmatrix} T_{*,4} \quad (\text{A.2.6})$$

$$\mathbf{W}_{off}(\theta_1, \dots, \theta_5) = \begin{bmatrix} 0 \\ 1 \end{bmatrix} T \cdot \begin{bmatrix} 1 \\ 2 \end{bmatrix} T \cdot \begin{bmatrix} 2 \\ 3 \end{bmatrix} T \cdot \begin{bmatrix} 3 \\ 4 \end{bmatrix} T \cdot \begin{bmatrix} 4 \\ 5 \end{bmatrix} T_{*,4} \quad (\text{A.2.7})$$

$$\mathbf{W}(\theta_1, \dots, \theta_6) = \begin{bmatrix} 0 \\ 1 \end{bmatrix} T \cdot \begin{bmatrix} 1 \\ 2 \end{bmatrix} T \cdot \begin{bmatrix} 2 \\ 3 \end{bmatrix} T \cdot \begin{bmatrix} 3 \\ 4 \end{bmatrix} T \cdot \begin{bmatrix} 4 \\ 5 \end{bmatrix} T \cdot \begin{bmatrix} 5 \\ 6 \end{bmatrix} T_{*,4} \quad (\text{A.2.8})$$

The position of the palm of the robotic hand is given by Equation A.2.9, while Equations A.2.10, A.2.11 and A.2.12 describe the computation of the end-effector reference frame $\{7\}$.

$$\mathbf{H}(\theta_1, \dots, \theta_7) = \begin{bmatrix} 0 \\ 1 \end{bmatrix} T \cdot \begin{bmatrix} 1 \\ 2 \end{bmatrix} T \cdots \begin{bmatrix} 6 \\ 7 \end{bmatrix} T_{*,4} \quad (\text{A.2.9})$$

$$\hat{\mathbf{x}}_7(\theta_1, \dots, \theta_7) = \begin{bmatrix} 1 \\ 0 \end{bmatrix} T \cdot \begin{bmatrix} 1 \\ 2 \end{bmatrix} T \cdots \begin{bmatrix} 6 \\ 7 \end{bmatrix} T_{*,1} \quad (\text{A.2.10})$$

$$\hat{\mathbf{y}}_7(\theta_1, \dots, \theta_7) = \begin{bmatrix} 1 \\ 0 \end{bmatrix} T \cdot \begin{bmatrix} 1 \\ 2 \end{bmatrix} T \cdots \begin{bmatrix} 6 \\ 7 \end{bmatrix} T_{*,2} \quad (\text{A.2.11})$$

$$\hat{\mathbf{z}}_7(\theta_1, \dots, \theta_7) = \begin{bmatrix} 1 \\ 0 \end{bmatrix} T \cdot \begin{bmatrix} 1 \\ 2 \end{bmatrix} T \cdots \begin{bmatrix} 6 \\ 7 \end{bmatrix} T_{*,3} \quad (\text{A.2.12})$$

A.3 Kinematics of the Barrett Hand

The kinematics of the Barrett Hand *BH8-262* is described in this section in accordance to the purposes of this project and to the robots that use it as end-effector. In section A.3.1 the direct kinematics of each finger is analysed, while section A.3.2 introduces the techniques of inverse kinematics that are used for grasping and releasing objects in pick-and-place tasks.

A.3.1 Direct kinematics of the fingers

By the Denavit-Hartenberg (D-H) convention (Craig, 2005), it is possible to compute the direct kinematics of the fingers for the Barrett hand. As it is illustrated in Figure A.4, the joints of the hand are referred as $\theta_8, \theta_9, \theta_{10}, \theta_{11}$ when the robotic hand is either mounted on the right arm of the robot ARoS or on the Sawyer manipulator. Concerning the Barrett hand of the left arm of ARoS, the joints variables are referred as $\theta_{19}, \theta_{20}, \theta_{21}$ and θ_{22} . However, only the name of the variables changes, while the kinematics procedures remain equal. For the sake of clarity, Figure A.4 shows an example for the finger F_3 that is featured by reference frames assigned along its phalanxes by the D-H convention. Each frame i is indexed with the number of the finger k and a label that corresponds to a determined position on the hand $\{0, 1, 2, tip\}$ (Figure A.4). The D-H parameters of the k^{th} finger ($k \in \{1, 2, 3\}$) in a Barrett Hand are collected in Ta-

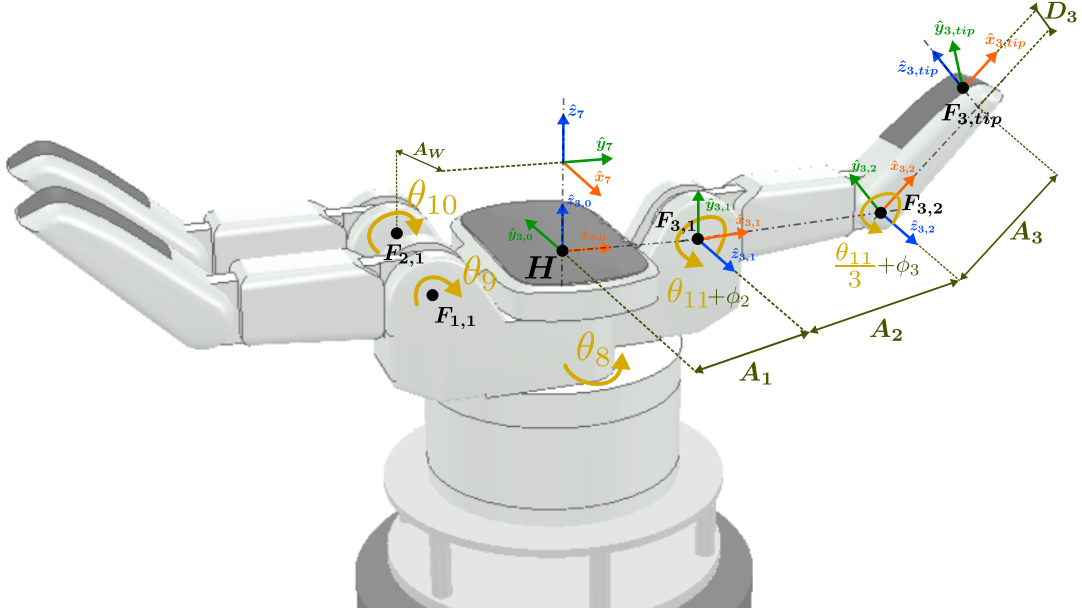


Figure A.4: Joint angles and reference frames on the finger F_3 are displayed as an example of direct kinematics on the Barrett Hand.

ble A.4. Note that in this table the variable θ_8 is replaced with θ_{19} when the Barrett Hand is the terminal of the left arm for the robot ARoS.

i	$\alpha_{i-1}[deg]$	$a_{i-1}[mm]$	$d_i[mm]$	$\theta_i[deg]$
$k, 0$	0	$r(k) \cdot A_W$	0	$a = r(k) \cdot \theta_8 - 90 \cdot j(k)$
$k, 1$	90	A_1	0	$b = \phi_2 + \theta_{8+k}$
$k, 2$	0	A_2	0	$c = \phi_3 + \frac{\theta_{8+k}}{3}$
k, tip	-90	A_3	D_3	0

Table A.4: D-H parameters of the k^{th} finger

where $r = \begin{bmatrix} -1 & 1 & 0 \end{bmatrix}^T$, $j = \begin{bmatrix} -1 & -1 & 1 \end{bmatrix}^T$ and $A_W, D_3, A_1, A_2, A_3, \phi_2, \phi_3$ are physical dimensions of the Barrett Hand. The transformation matrices between consecutive frames of a k^{th} finger are reported in Equations A.3.1, A.3.2, A.3.3 and A.3.4.

$${}^h_{k,0}T = \begin{bmatrix} c_a & -s_a & 0 & r(k) \cdot A_W \\ s_a & c_a & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (A.3.1)$$

$${}^{k,0}_{k,1}T = \begin{bmatrix} c_b & -s_b & 0 & A_1 \\ 0 & 0 & -1 & 0 \\ s_b & c_b & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (A.3.2)$$

$${}_{k,2}^{k,1}T = \begin{bmatrix} c_c & -s_c & 0 & A_2 \\ s_c & c_c & 1 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (\text{A.3.3})$$

$${}_{k,tip}^{k,2}T = \begin{bmatrix} 1 & 0 & 0 & A_3 \\ 0 & 0 & 1 & D_3 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (\text{A.3.4})$$

where, $h \in \{7, 18\}$, $c_a = \cos(a)$, $s_a = \sin(a)$, $c_b = \cos(b)$, $s_b = \sin(b)$, $c_c = \cos(c)$, $s_c = \sin(c)$. The auxiliary parameter h in Equation A.3.1 is equal to 7, when the Barrett Hand is the end-effector of the Sawyer manipulator and of the right arm of ARoS, otherwise $h = 18$ when the Barrett Hand is the terminal of the left arm of ARoS. With these matrices, the position of the finger tips and of the intermediate points along the finger (Figure A.4) can be calculated for the robot Sawyer (Equations A.3.5, A.3.6 and A.3.7), for the right arm (Equations A.3.8, A.3.9 and A.3.10) and the left arm of the robot ARoS (Equations A.3.11, A.3.12 and A.3.13).

$$F_{k,1} = \left[{}^0_7T \cdot {}^7_{k,0}T \cdot {}^{k,0}_{k,1}T \right]_{*,4} \quad (\text{A.3.5})$$

$$F_{k,2} = \left[{}^0_7T \cdot {}^7_{k,0}T \cdot {}^{k,0}_{k,1}T \cdot {}^{k,1}_{k,2}T \right]_{*,4} \quad (\text{A.3.6})$$

$$F_{k,tip} = \left[{}^0_7T \cdot {}^7_{k,0}T \cdot {}^{k,0}_{k,1}T \cdot {}^{k,1}_{k,2}T \cdot {}^{k,2}_{k,tip}T \right]_{*,4} \quad (\text{A.3.7})$$

$$F_{k,1}^R = \left[{}^W_{O_R}T \cdot {}^{O_R}_7T \cdot {}^7_{k,0}T \cdot {}^{k,0}_{k,1}T \right]_{*,4} \quad (\text{A.3.8})$$

$$F_{k,2}^R = \left[{}^W_{O_R}T \cdot {}^{O_R}_7T \cdot {}^7_{k,0}T \cdot {}^{k,0}_{k,1}T \cdot {}^{k,1}_{k,2}T \right]_{*,4} \quad (\text{A.3.9})$$

$$F_{k,tip}^R = \left[{}^W_{O_R}T \cdot {}^{O_R}_7T \cdot {}^7_{k,0}T \cdot {}^{k,0}_{k,1}T \cdot {}^{k,1}_{k,2}T \cdot {}^{k,2}_{k,tip}T \right]_{*,4} \quad (\text{A.3.10})$$

$$F_{k,1}^L = \left[{}^W_{O_L}T \cdot {}^{O_L}_{18}T \cdot {}^{18}_{k,0}T \cdot {}^{k,0}_{k,1}T \right]_{*,4} \quad (\text{A.3.11})$$

$$F_{k,2}^L = \left[{}^W_{O_L}T \cdot {}^{O_L}_{18}T \cdot {}^{18}_{k,0}T \cdot {}^{k,0}_{k,1}T \cdot {}^{k,1}_{k,2}T \right]_{*,4} \quad (\text{A.3.12})$$

$$F_{k,tip}^L = \left[{}^W_{O_L}T \cdot {}^{O_L}_{18}T \cdot {}^{18}_{k,0}T \cdot {}^{k,0}_{k,1}T \cdot {}^{k,1}_{k,2}T \cdot {}^{k,2}_{k,tip}T \right]_{*,4} \quad (\text{A.3.13})$$

A.3.2 Inverse kinematics of the fingers

The execution of pick-and-place tasks require the Barrett hand to pass through grasping and un-grasping transitions of the rigid objects that are being manipulated by the robot. For these purposes, the joint angles of the fingers have to be properly selected to ensure the success of these transitions with an intended grasping behaviour. As described in section 3.2, reach-to-grasp movements in humans and primates are

featured by precision and power/full grip types (Castiello, 2005; Napier, 1956). For robotic manipulation, the inverse kinematics of the fingers accounts for two variables that need to be set with the purpose of emulating the mentioned biological grip types. As it is illustrated by Figure A.5, these variables are d_{FF} and d_{FH} : d_{FF} represents the distance between the finger F_3 and the middle point between the opposing fingers F_1 and F_2 , while d_{FH} is the distance between this middle point and the center of the palm of the hand. If a

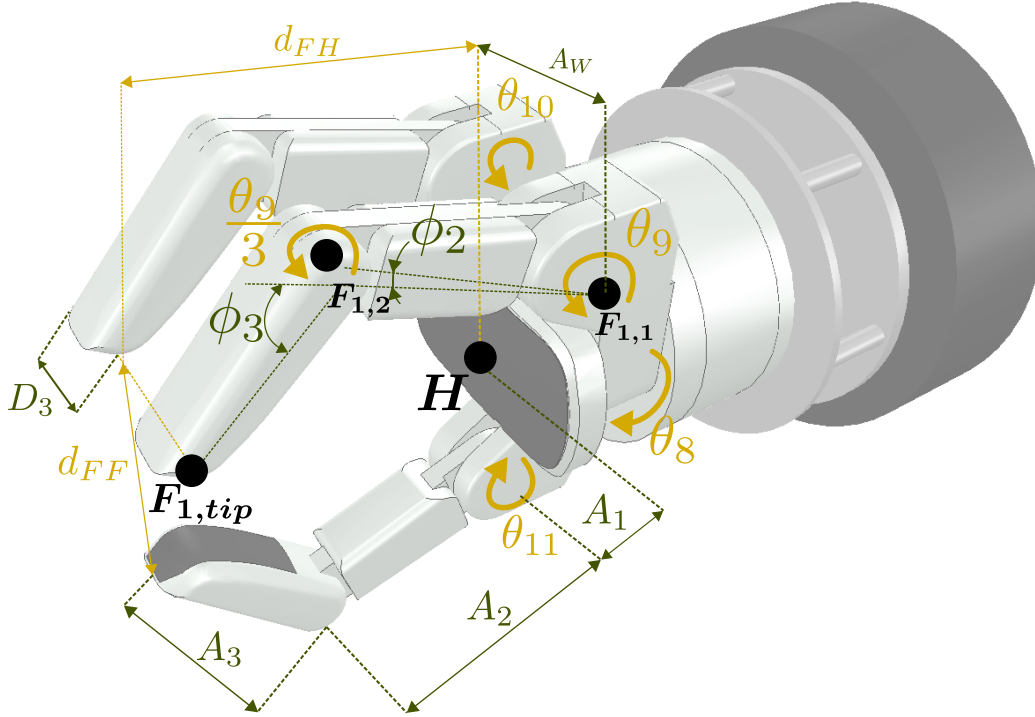


Figure A.5: Joint angles and distances d_{FF} and d_{FH} are displayed for the inverse kinematics of the Barrett Hand.

Barrett Hand is mounted as end-effector on the Sawyer manipulator and on the right arm of ARoS, the joints angles of the fingers are represented by the vector $\theta_{hand} = [\theta_8, \theta_9, \theta_{10}, \theta_{11}]^T$ (Figure A.5). When a Barrett Hand is mounted at the terminal of the left arm of ARoS, the hand joints variables are given by the vector $\theta_{hand} = [\theta_{19}, \theta_{20}, \theta_{21}, \theta_{22}]^T$. However, the considerations about the inverse kinematics of the fingers remain equally applicable. Therefore, for successful grasp and un-grasp transitions the vector θ_{hand} has to be carefully selected. In particular, the finger F_3 is assumed to be always opposite to the other two fingers F_1 and F_2 , therefore $\theta_8 = \theta_{19} = 0$. The position of the hand point target at the transitions is assumed to be so that the robotic hand can sufficiently grasp or un-grasp an object by an equal and simultaneous closure or aperture of the fingers. This hypothesis is not a significant limitation for the vast majority of pick-and-place tasks and allows us to simplify the inverse kinematic problem of the Barrett hand. In fact, since all fingers have equal lengths and can be equally closed or opened, we set $\theta_9 = \theta_{10} = \theta_{11} = \theta_{fing}$ and $\theta_{20} = \theta_{21} = \theta_{22} = \theta_{fing}$. Given the geometry of the hand, specifications of the object that has to be

manipulated, the positions of the joints of the fingers are determined by solving the Equation A.3.14 using the Newton-Raphson method (see Algorithm A.1).

$$\begin{aligned} d_{FF} = & 2 \cdot A_3 \cos\left(\frac{4}{3}\theta_{fing} + \phi_2 + \phi_3\right) \\ & - 2 \cdot D_3 \sin\left(\frac{4}{3}\theta_{fing} + \phi_2 + \phi_3\right) \\ & + 2 \cdot A_2 \cos(\theta_{fing} + \phi_2) + A_1 \end{aligned} \quad (\text{A.3.14})$$

The d_{FF} must be equal to a dimension of the manipulated object that is most significant for the act of grasping or un-grasping. A circular cylindrical object, for instance, can be grasped by setting its diameter as d_{FF} and can be un-grasped by increasing this value of a given percentage and let the fingers open. Algorithm A.1 describes how θ_{fing} is iteratively obtained from a selected size of an object. Having computed

Algorithm A.1 Inverse kinematics of the Barrett hand

Inputs

d_{obj} – size of the object that is manipulated

Outputs

θ_{fing} – angular position of the fingers

```

1:  $\theta_{new} = \theta_0$  // initial configuration of the fingers
2: while  $\|\theta_{new} - \theta_l\| \geq 10^{-4}$  do // check for convergence
3:    $\theta_l = \theta_{new}$  // save the new configuration to the internal loop variable
4:    $f = d_{FF}(\theta_l) - d_{obj}$  // compute the error with the given size of the object
5:    $df = \left. \frac{\partial d_{FF}}{\partial \theta} \right|_{\theta=\theta_l}$  // derivative of  $d_{FF}$  at  $\theta = \theta_l$ 
6:    $\theta_{new} \leftarrow \theta_l - \frac{f}{df}$  // save the new configuration of the fingers
7: end while
8:  $\theta_{fing} = \theta_l$  // save the output
```

the positions of the joints of the fingers the distance d_{FH} can be calculated according to Equation A.3.15.

$$d_{FH} = A_3 \sin\left(\frac{4}{3}\theta_{fing} + \phi_2 + \phi_3\right) + D_3 \cos\left(\frac{4}{3}\theta_{fing} + \phi_2 + \phi_3\right) + A_2 \sin(\theta_{fing} + \phi_2) \quad (\text{A.3.15})$$

When a precision grip is required, the value of d_{FH} is used to correctly place the palm of the Barrett Hand in respect with the target that is positioned on the line defined by the distance d_{FF} (see Figure A.5). When a power grip is required, the palm of the hand is in contact with the object during manipulation, therefore the value of d_{FH} cannot be used to identify the position of the palm in respect with a given target. For this purpose, a value is selected to reflect given rigid geometrical constraints: it can be, for instance, the radius of circular cylindrical objects. Power grips also demand the calculus of a θ_{fing} from a value of d_{FF} that slightly overcomes the size of an object and allows a greater aperture of the hand to successfully grasp and un-grasp.

Appendix B

Modelization of the robot ARoS and of the objects in the workspace

The purpose of modelling a robot for motion planning is not to replicate the geometry as precisely as possible, but to provide the simplest representation of those parts that are more exposed to collisions than others. This modelization is also fundamental to develop closed-form formulations of the constraints of the motion for collisions-avoidance. The robotic arms of ARoS and the fingers of the Barrett hand are modelled as a set of spheres (Equation B.0.1) centred at the points P_h , $\forall h \in \{1, \dots, 15\}$ for the right arm-hand system and $\forall h \in \{16, \dots, 30\}$ for the left arm-hand system. The positions of the shoulders, elbows, wrists and hands points are computed by direct kinematics (Appendix A.1) and are denoted by S_R , E_R , W_R and H_R , on the right arm, and by S_L , E_L , W_L and H_L , on the left arm. The points $F_{k,1}^R$, $F_{k,2}^R$ and $F_{k,tip}^R$, are points on the k^{th} finger ($\forall k \in 1, 2, 3$) of the right Barrett hand. Similarly, the points $F_{k,1}^L$, $F_{k,2}^L$ and $F_{k,tip}^L$, are points on the k^{th} finger ($\forall k \in 1, 2, 3$) of the left Barrett hand. In Appendix A.3.1 the direct kinematics to obtain the coordinates of these points is described.

$$\left(\frac{x - x_c}{r}\right)^2 + \left(\frac{y - y_c}{r}\right)^2 + \left(\frac{z - z_c}{r}\right)^2 \leq 1 \quad (\text{B.0.1})$$

The spheres have arbitrarily tunable radii and are centred on the points P_h as it is illustrated on the Figures B.1a and B.1b for the right robotic upper-limb of ARoS. The coordinates of the points P_h are set according to the human-like arm-hand positions that are collected in the Tables B.1 and B.2. The body of the robot ARoS is modelled as an elliptic cylinder (Equation B.0.2) that divides the arms and it is considered an obstacle that must be avoided during the generation and execution of a trajectory (Figure B.1c).

$$\left(\frac{x - x_c}{\sigma_x}\right)^2 + \left(\frac{y - y_c}{\sigma_y}\right)^2 \leq 1 \quad (\text{B.0.2})$$

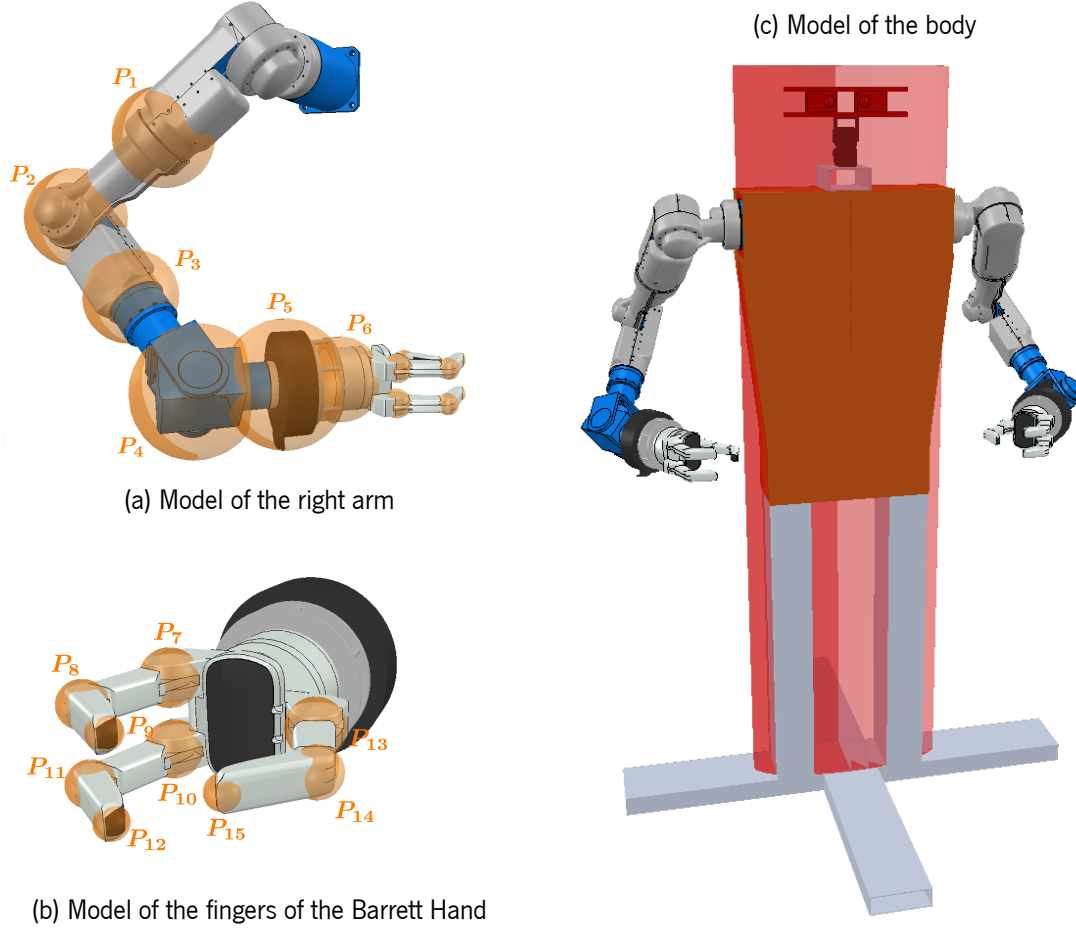


Figure B.1: The right robotic arm (Figure B.1a) and the three fingers of robotic hand (Figure B.1b) are modelled as a set of spheres. The body of the robot ARoS is modelled as an elliptic cylinder (Figure B.1c).

The objects in the workspace, which are not involved in the movement to execute, are obstacles for a robot and are modelled as ellipsoids (Equation B.0.3) as Figure B.2a illustrates. For a better coverage, the dimensions of the three axes, r_x , r_y and r_z , of the ellipsoid are 30% greater than the bounding box sizes of the enveloping object.

$$\left(\frac{x - x_c}{r_x}\right)^2 + \left(\frac{y - y_c}{r_y}\right)^2 + \left(\frac{z - z_c}{r_z}\right)^2 \leq 1 \quad (\text{B.0.3})$$

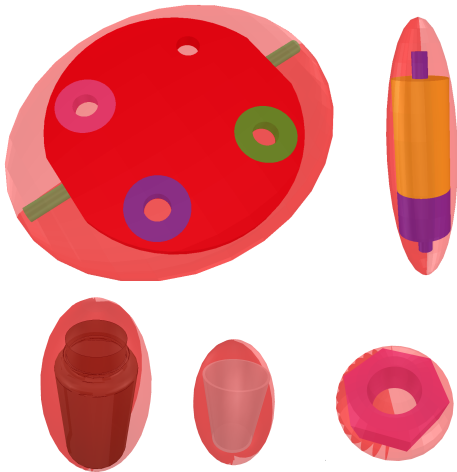
During the manipulation, the object that is held by the end-effector is considered as part of the kinematic chain of the robotic upper-limb system. Therefore, this object is also modelled with a set of spheres (Equation B.0.1) as the arms and the hands of ARoS (Figure B.2b). The aim of this representation is to model the bounding box of the object with a minimum number of spheres. In order to cover totally an object along the axis of its minimum size, the diameter of each sphere is given by the minimum-sized axis of its

Points	Position of the points
P_1	$(S_R + E_R) / 2$
P_2	E_R
P_3	$(E_R + W_R) / 2$
P_4	W_R
P_5	$W_R + 0.45 \cdot (H_R - W_R)$
P_6	$W_R + 0.75 \cdot (H_R - W_R)$
P_7	$F_{1,1}^R$
P_8	$F_{1,2}^R$
P_9	$F_{1,tip}^R$
P_{10}	$F_{2,1}^R$
P_{11}	$F_{2,2}^R$
P_{12}	$F_{2,tip}^R$
P_{13}	$F_{3,1}^R$
P_{14}	$F_{3,2}^R$
P_{15}	$F_{3,tip}^R$

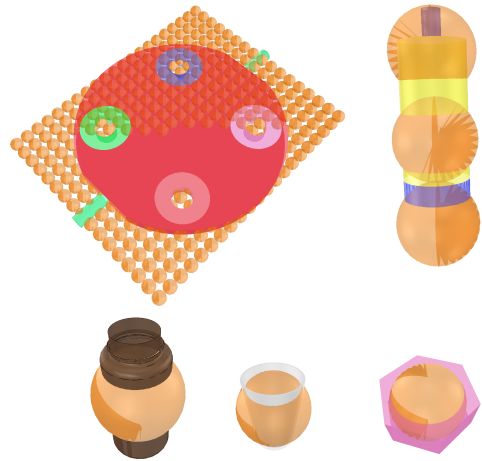
Table B.1: Positions of the points on the right arm and hand

Points	Position of the points
P_{16}	$(S_L + E_L) / 2$
P_{17}	E_L
P_{18}	$(E_L + W_L) / 2$
P_{19}	W_L
P_{20}	$W_L + 0.45 \cdot (H_L - W_L)$
P_{21}	$W_L + 0.75 \cdot (H_L - W_L)$
P_{22}	$F_{1,1}^L$
P_{23}	$F_{1,2}^L$
P_{24}	$F_{1,tip}^L$
P_{25}	$F_{2,1}^L$
P_{26}	$F_{2,2}^L$
P_{27}	$F_{2,tip}^L$
P_{28}	$F_{3,1}^L$
P_{29}	$F_{3,2}^L$
P_{30}	$F_{3,tip}^L$

Table B.2: Positions of the points on the left arm and hand



(a) Objects modelled as ellipsoids



(b) Objects modelled as a set of spheres

Figure B.2: The objects in the workspace are modelled as ellipsoids when they are considered as obstacles (Figure B.2a). When an object is being manipulated by the robot, it is modelled as a set of spheres (Figure B.2b).

wrapping ellipsoid. Along the other two axes, the spheres are replicated with equally spaced shifts. The pseudo-code of this process is outlined in Algorithm B.1 and the resulting sets of spheres on different objects are showed in Figure B.2b.

Algorithm B.1 Generation of a set of spheres in a given object

Inputs

$dims$ – dimensions of the ellipsoid wrapping the object

P_c – geometrical center of the object

Outputs

θ_{fing} – angular position of the fingers

```
1:  $(size_{min}, size_{middle}, size_{max}) \leftarrow \text{orderDims}(dims)$            // dimensions in ascending order
2:  $diam_{sphere} = size_{min}$                                            // the diameter of the spheres
3:  $n_1 = \text{round}(size_{middle}/diam_{sphere})$                            // number of spheres in the medium size axis
4:  $n_2 = \text{round}(size_{max}/diam_{sphere})$                              // number of spheres in the maximum size axis
5: for  $i \in \{0, \dots, n_1\}$  do
6:   for  $j \in \{0, \dots, n_2\}$  do
7:      $P_s \leftarrow \text{getSphereCenter}(P_c, diam_{sphere}, i, j)$        // center of the sphere  $\{i, j\}$ 
8:      $\text{createSphere}(P_s, diam_{sphere})$                              // create the sphere  $\{i, j\}$ 
9:   end for
10: end for
```

Appendix C

Mathematical formulation of the direct and the back-and-forth movements

C.1 Transport direct movements

A trajectory based on the minimum angular jerk principle is planned to minimize the change rate of angular acceleration. This means minimizing the integration of the angular jerk over the duration of a movement T (Wada et al., 2001):

$$C(\theta) = \frac{1}{2} \int_0^T \sum_{k=1}^n \left(\ddot{\theta}_k(t) \right)^2 dt \quad (\text{C.1.1})$$

where $\theta = (\theta_1, \theta_2, \dots, \theta_n)^T$, n is the number of joints and θ_k is the k^{th} joint. This minimization problem is subject to the boundary conditions

$$\theta(0) = \theta_0, \quad \theta(T) = \theta_{tar} \quad (\text{C.1.2})$$

$$\dot{\theta}(0) = \omega_0, \quad \dot{\theta}(T) = \omega_{tar} \quad (\text{C.1.3})$$

$$\ddot{\theta}(0) = \alpha_0, \quad \ddot{\theta}(T) = \alpha_{tar} \quad (\text{C.1.4})$$

where θ_0 , ω_0 and α_0 are the start postural position, velocity and acceleration, respectively. The θ_{tar} , ω_{tar} and α_{tar} are the target postural position, velocity and acceleration, respectively. While the start postures are given by the current situation of the robot, ω_{tar} and α_{tar} are either set arbitrarily or by the initial condition of the subsequent phase after *Transport*. The target posture, θ_{tar} , has to be selected so as to minimize the cost function in Equation C.1.1. This minimization refers to a typical problem of variational calculus where a stationary function needs to be found for the functional in Equation C.1.5:

$$L = \sum_{k=1}^n \left(\ddot{\theta}_k(t) \right)^2 = \sum_{k=1}^n (r_k(t))^2 = L(r_k(t)) \quad (\text{C.1.5})$$

where it is assumed that it is sufficiently differentiable in the interval $[0, T]$ and that

$$\theta_k(t) = u_k(t), \dot{\theta}_k(t) = p_k(t), \ddot{\theta}_k(t) = q_k(t), \dddot{\theta}_k(t) = r_k(t)$$

Any function $\theta_k(t) = u_k(t)$ of the cost in Equation C.1.1 must satisfy the associated Euler-Lagrange equation C.1.6 to be an extremum (Gelfand and Fomin, 1963).

$$\frac{\partial L}{\partial u_k} - \frac{d}{dt} \left(\frac{\partial L}{\partial p_k} \right) + \frac{d^2}{dt^2} \left(\frac{\partial L}{\partial q_k} \right) - \frac{d^3}{dt^3} \left(\frac{\partial L}{\partial r_k} \right) = 0 \quad (\text{C.1.6})$$

Applying the functional in Equation C.1.5 to Equation C.1.6, we obtain

$$\begin{aligned} \frac{d^3}{dt^3} \left(\sum_{k=1}^n \frac{\partial r_k^2}{\partial r_k} \right) &= 0 \Leftrightarrow \sum_{k=1}^n \frac{d^3}{dt^3} (2 \cdot r_k) = 0 \\ &\Leftrightarrow \sum_{k=1}^n 2 \cdot \frac{d^3}{dt^3} (\ddot{\theta}_k(t)) = 0 \\ &\Leftrightarrow 2 \cdot \sum_{k=1}^n \frac{d^6}{dt^6} (\theta_k(t)) = 0 \\ &\Leftrightarrow \frac{d^6}{dt^6} (\theta_k(t)) = 0 \quad \forall k \in \{1, \dots, n\} \end{aligned} \quad (\text{C.1.7})$$

The solution of Equation C.1.7 is a quintic function, i.e. a polynomial of degree five in Equation C.1.8

$$\theta_k(t) = \sum_{m=0}^5 a_{m,k} t^m = a_{0,k} + a_{1,k} t + a_{2,k} t^2 + a_{3,k} t^3 + a_{4,k} t^4 + a_{5,k} t^5 \quad (\text{C.1.8})$$

The coefficients of the fifth order polynomial in Equation C.1.8 are determined by applying the boundary conditions (Equations C.1.2, C.1.3 and C.1.4). Collecting the resulting trajectories for all the joints, the position is obtained (Equation C.1.9)

$$\begin{aligned} \theta(t) &= \theta_0 + (\theta_{tar} - \theta_0) (10\tau^3 - 15\tau^4 + 6\tau^5) \\ &\quad + \omega_0 T (\tau - 6\tau^3 + 8\tau^4 - 3\tau^5) \\ &\quad + \omega_{tar} T (-4\tau^3 + 7\tau^4 - 3\tau^5) \\ &\quad + \frac{1}{2} \alpha_0 T^2 (\tau^2 - 3\tau^3 + 3\tau^4 - \tau^5) \\ &\quad + \frac{1}{2} \alpha_{tar} T^2 (\tau^3 - 2\tau^4 + \tau^5) \end{aligned} \quad (\text{C.1.9})$$

where $\tau = t/T$ is the normalized time.

Deriving over time Equation C.1.9, the velocity, the acceleration and the jerk of the joints-space trajectory is obtained (Equations C.1.10, C.1.11 and C.1.12)

$$\begin{aligned}\dot{\theta}(t) = & \frac{30}{T} (\theta_{tar} - \theta_0) (\tau^2 - 2\tau^3 + \tau^4) \\ & + \omega_0 (1 - 18\tau^2 + 32\tau^3 - 15\tau^4) \\ & + \omega_{tar} (-12\tau^2 + 28\tau^3 - 15\tau^4) \\ & + \frac{1}{2} \alpha_0 T (2\tau - 9\tau^2 + 12\tau^3 - 5\tau^4) \\ & + \frac{1}{2} \alpha_{tar} T (3\tau^2 - 8\tau^3 + 5\tau^4)\end{aligned}\tag{C.1.10}$$

$$\begin{aligned}\ddot{\theta}(t) = & \frac{60}{T^2} (\theta_{tar} - \theta_0) (\tau - 3\tau^2 + 2\tau^3) \\ & + \frac{12}{T} \omega_0 (-3\tau + 8\tau^2 - 5\tau^3) \\ & + \frac{12}{T} \omega_{tar} (-2\tau + 7\tau^2 - 5\tau^3) \\ & + \alpha_0 (1 - 9\tau + 18\tau^2 - 10\tau^3) \\ & + \alpha_{tar} (-3\tau - 12\tau^2 + 10\tau^3)\end{aligned}\tag{C.1.11}$$

$$\begin{aligned}\ddot{\theta}(t) = & \frac{360}{T^3} (\theta_{tar} - \theta_0) \left(\frac{1}{6} - \tau + \tau^2 \right) \\ & + \frac{12}{T^2} \omega_0 (-3 + 16\tau - 15\tau^2) \\ & + \frac{12}{T^2} \omega_{tar} (-2 + 14\tau - 15\tau^2) \\ & + \frac{\alpha_0}{T} (-9 - 36\tau + 30\tau^2) \\ & + \frac{\alpha_{tar}}{T} (3 - 24\tau + 30\tau^2)\end{aligned}\tag{C.1.12}$$

The minimization of the distance between the vectors θ_{tar} and θ_0 implies the selection of a target posture that reduces the angular jerk in Equation C.1.12.

C.2 Back-and-forth movements

The back-and-forth movement is featured by a trajectory of the joints given in Equation C.2.1.

$$\begin{aligned}\theta(t) &= \frac{\tau(1-\tau)}{t_b(1-t_b)} (\theta_b - \theta_0) \sin^2(\pi \tau^\vartheta), \\ \vartheta &= -\frac{\ln 2}{\ln t_b}, t_b \in]0,1[\end{aligned}\tag{C.2.1}$$

where $\boldsymbol{\theta} = (\theta_1, \theta_2, \dots, \theta_n)^\top$, n is the number of joints, $\boldsymbol{\theta}_b$ is the bounce posture and $\boldsymbol{\theta}_0$ is the start posture.

$$\dot{\boldsymbol{\theta}}(t) = \frac{1}{T t_b (1 - t_b)} (\boldsymbol{\theta}_b - \boldsymbol{\theta}_0) \cdot \left[(1 - 2\tau) \sin^2(\pi \tau^\vartheta) + (1 - \tau) \pi \vartheta \tau^\vartheta \sin(2\pi \tau^\vartheta) \right] \quad (\text{C.2.2})$$

Deriving over time Equation C.2.1, the velocity (Equation C.2.2), the acceleration (Equation C.2.3) and the jerk (Equation C.2.4) of the back-and-forth movement are obtained.

$$\begin{aligned} \ddot{\boldsymbol{\theta}}(t) = & \frac{2}{T^2 t_b (1 - t_b)} (\boldsymbol{\theta}_b - \boldsymbol{\theta}_0) \\ & \cdot \left[-\sin^2(\pi \tau^\vartheta) + \pi^2 \vartheta^2 (1 - \tau) \tau^{2\vartheta-1} \cos(2\pi \tau^\vartheta) \right. \\ & \left. + (1 - 2\tau) \pi \vartheta \tau^{\vartheta-1} \sin(2\pi \tau^\vartheta) \right] \end{aligned} \quad (\text{C.2.3})$$

$$\begin{aligned} \dddot{\boldsymbol{\theta}}(t) = & \frac{2}{T^3 t_b (1 - t_b)} (\boldsymbol{\theta}_b - \boldsymbol{\theta}_0) \\ & \cdot \left[\left(\vartheta - 1 - 2\tau \vartheta + (\tau - 1) 2\vartheta^2 \tau^{2\vartheta} - \tau \right) \cdot \pi \vartheta \tau^{\vartheta-2} \sin(2\pi \tau^\vartheta) \right. \\ & \left. + (2\vartheta - 1 - 2\vartheta \tau + (1 - 2\tau) 2\pi) \cdot \pi \vartheta^2 \tau^{2\vartheta-2} \cos(2\pi \tau^\vartheta) \right] \end{aligned} \quad (\text{C.2.4})$$

The cost associated with the bounce posture selection process (Equation 5.1.16) is considered the weighted distance between the vectors $\boldsymbol{\theta}_b$ and $\boldsymbol{\theta}_0$ because impacts the amplitude of the angular jerk in Equation C.2.4.