

Universidade do Minho
Escola de Engenharia

Pedro João Silva Leite

Automotive Gestures Recognition Based on Capacitive Sensing

Dissertação de Mestrado
Mestrado Integrado em Engenharia Eletrónica
Industrial e Computadores

Trabalho efetuado sob a orientação do
Professor Doutor Paulo Cardoso

Janeiro de 2017

Declaração do Autor

Nome: Pedro João Silva Leite

Endereço eletrónico: a66161@alunos.uminho.pt

Telemóvel: 253615199

Cartão do Cidadão: 14154440

Título da dissertação: Automotive Gesture Recognition Based on Capacitive Sensing

Orientador: Professor Doutor Paulo Cardoso

Ano de conclusão: 2016

Mestrado Integrado em Engenharia Eletrónica Industrial e Computadores

DE ACORDO COM A LEGISLAÇÃO EM VIGOR, NÃO É PERMITIDA A RE-
PRODUÇÃO DE QUALQUER PARTE DESTA TESE/TRABALHO.

Universidade do Minho, _____ / _____ / _____

Assinatura:

Acknowledgements

Firstly, I would like to thank my supervisor Prof. Paulo Cardoso, who introduced me to Embedded Systems, for his advices and understanding. Secondly, I am also deeply in debt to Eng. Marco Martins primarily for his friendship, his guidance, support and industry experience were not only important for this research but also for life. Next, I must express my gratitude to all my friends, namely Vinícius Silva, Kevin Costa, Bruno Mendes, Artur Faria, António Ribeiro, Diogo Cruz, Rui Paixão, Áurea Salgado and Pedro Barbosa, who helped me through my academic studies and DEI workshop for their technical support and assistance.

Finally, I must express my gratitude to my parents for providing me with unconditional and continuous support through academic life and the conclusion of this life cycle with the process of researching and writing this thesis.

It is worth noting that this work has been financially supported by the Portugal Incentive System for Research and Technological Development in scope of the projects in co-promotion number 036265/2013 (HMIExcel 2013-2015), number 002814/2015 (iFACTORY 2015-2018) and number 002797/2015 (INNOV-CAR 2015-2018).

To all, my sincerely thank you!

Abstract

Driven by technological advancements, vehicles have steadily increased in sophistication, specially in the way drivers and passengers interact with their vehicles. For example, the BMW 7 series driver-controlled systems, contains over 700 functions. Whereas, it makes easier to navigate streets, talk on phone and more, this may lead to visual distraction, since when paying attention to a task not driving related, the brain focus on that activity. That distraction is, according to studies, the third cause of accidents, only surpassed by speeding and drunk driving.

Driver distraction is stressed as the main concern by regulators, in particular, National Highway Transportation Safety Agency (NHTSA), which is developing recommended limits for the amount of time a driver needs to spend glancing away from the road to operate in-car features. Diverting attention from driving can be fatal; therefore, automakers have been challenged to design safer and comfortable human-machine interfaces (HMIs) without missing the latest technological achievements.

This dissertation aims to mitigate driver distraction by developing a gestural recognition system that allows the user a more comfortable and intuitive experience while driving. The developed system outlines the algorithms to recognize gestures using the capacitive technology.

Keywords: Gestures Recognition, Capacitive Sensing, Automotive HMI

Resumo

Impulsionados pelos avanços tecnológicos, os automóveis tem de forma continua aumentado em complexidade, sobretudo na forma como os condutores e passageiros interagem com os seus veículos. Por exemplo, os sistemas controlados pelo condutor do BMW série 7 continham mais de 700 funções. Embora, isto facilite a navegação entre locais, falar ao telemóvel entre outros, isso pode levar a uma distração visual, já que ao prestar atenção a uma tarefa não relacionados com a condução, o cérebro se concentra nessa atividade. Essa distração é, de acordo com os estudos, a terceira causa de acidentes, apenas ultrapassada pelo excesso de velocidade e condução embriagada.

A distração do condutor é realçada como a principal preocupação dos reguladores, em particular, a National Highway Transportation Safety Agency (NHTSA), que está desenvolvendo os limites recomendados para a quantidade de tempo que um condutor precisa de desviar o olhar da estrada para controlar os sistemas do carro. Desviar a atenção da condução, pode ser fatal; portanto, os fabricante de automóveis têm sido desafiados a projetar interfaces homem-máquina (HMIs) mais seguras e confortáveis, sem perder as últimas conquistas tecnológicas.

Esta dissertação tem como objetivo minimizar a distração do condutor, desenvolvendo um sistema de reconhecimento gestual que permite ao utilizador uma experiência mais confortável e intuitiva ao conduzir. O sistema desenvolvido descreve os algoritmos de reconhecimento de gestos usando a tecnologia capacitiva.

Palavras-Chave: Reconhecimento de Gestos, Sensorização Capacitiva, Interfaces Homem-Máquina

Content

Declaração do Autor	iii
Acknowledgements	v
Resumo	vii
Abstract	ix
Content	xi
Figure List	xv
Table List	xix
Abbreviations	xxi
1 Introduction	1
1.1 Motivation	2
1.2 Research questions and Objectives	3
1.3 Contributions	3
1.4 Document Structure	4
2 State of the art	5
2.1 Gestures	5
2.1.1 Types of Gestures	6
2.2 Human Machine Interface	6
2.2.1 Automotive HMIs	7
2.2.2 Usability	8
2.2.3 Usability of In-Vehicle Information Systems (IVISs)	9
2.3 Gesture Recognition in Automotive	10
2.4 Capacitive Sensing	14
2.4.1 Capacitive Proximity Sensing	14
2.4.2 Proximity Sensing versus Touch Sensing	15
2.4.3 Sensing Configurations	15
2.4.4 Material and Geometry	16
2.4.5 Shielding	18
2.4.6 Applications	19
2.5 Alternative Sensing Technologies	24
2.5.1 Imaging Technologies	24
2.5.2 Non-Imaging Technologies	27

2.6	Mathematical Models for Gesture Recognition	30
2.6.1	Threshold Model	30
2.6.2	HMM	31
2.6.3	DTW	34
3	Analysis and System Specification	37
3.1	Methodology	37
3.2	Project Requirements	38
3.3	System Overview	39
3.3.1	Detailed Overview	39
3.4	Hardware Specification	40
3.4.1	Criteria to select the Sensing Devices	40
3.4.2	Selection of the Sensing Devices	40
3.4.3	Criteria to select the Development Board	41
3.4.4	Selection of the Development Board	42
3.5	Software Specification	43
3.5.1	Programming Language	43
3.5.2	Software Platforms	43
3.5.3	Algorithm Development	44
3.6	The system	44
3.6.1	Texas Instruments FDC1004 Interface	45
3.6.2	Hover Interface	49
3.6.3	Microchip MGC3130 Interface	49
3.6.4	Graphical user interface	51
3.6.5	GUI Application - CapGUI	52
3.6.6	GUI Application - CoorGUI	55
3.6.7	Cluster	58
3.7	HMI Concepts	60
3.8	Data Processing	62
3.9	System Validation	65
3.9.1	Final Tests	65
4	Implementation and Results	67
4.1	The system	67
4.1.1	Sensing Devices	68
4.1.2	Texas Instruments FDC1004	69
4.1.3	Hover Gesture Device	73
4.1.4	Microchip MGC3130	74
4.2	GUI Applications	75
4.2.1	CoorGUI	75
4.2.2	CapGUI	78
4.3	Server	79
4.3.1	Client	79
4.3.2	Server	79
4.4	Results	79
4.4.1	Interface with the Gesture Devices	80
4.4.2	Graphical User Interfaces	81

4.5	HMI System	83
4.6	Algorithm evaluation	83
4.6.1	Texas Instruments FDC1004	84
4.6.2	Hover Gesture	84
4.6.3	Microchip MGC3130	85
4.7	Results Analysis	86
5	Conclusions and Future Work	87
5.1	Conclusions	87
5.2	Future Work	88
	References	89

Figure List

2.1	Gesture-based automotive controls from the Google patent	7
2.2	Usability Diagram	8
2.3	BMW iDrive concept	10
2.4	BMW i8 iVisio concept	11
2.5	Hyundai Genesis concept	11
2.6	Gesture controlling sunroof of the Volkswagen Golf R Touch	12
2.7	Gesture controlling E Golf Touch	12
2.8	Volkswagen BUDD-e slidding door	13
2.9	Recognizable gestures by Visteon	13
2.10	Theremin - a contactless musical instrument	14
2.11	Measurement modes for capacitive proximity sensing	16
2.12	Spatial resolution of different materials at various distances	17
2.13	Prototype device supporting antenna wire	17
2.14	Threshold model with two different sensing modes	18
2.15	Microchip MG3031 electrodes layout	18
2.16	Differences between with shield and unshielded electrodes	19
2.17	Shielding to mitigate environmental interference	19
2.18	Active armrest sketch with six electrodes for finger gesture recognition	20
2.19	Thacker - four sensors arranged around the screen	21
2.20	Swiss-cheese prototype	21
2.21	Interaction between the data acquisition module, the capacitive sensor array, and the accelerometer wristband	22
2.22	Rainbowfish prototype	22
2.23	Gesture recognition on mobile device	22
2.24	Gestures Recognition Technology	24
2.25	Stereo Vision Concept	25
2.26	Structured Light Pattern	25
2.27	Time of Flight Concept	26
2.28	Depth image	26
2.29	Ultrasonic Concept	28
2.30	Infrared Concept	28
2.31	Controller based gestures principle	29
2.32	Radar principle	29
2.33	Threshold model	31
2.34	HMM Ergodic topology with 3 states	33
2.35	HMM Left-Right topology with 3 states	33
2.36	HMM Left-Right Banded topology with 3 states	33

2.37	Differences between the Euclidean Distance and the Dynamic Time Warping applied to the same signal	34
2.38	DTW path	35
3.1	Waterfall model	37
3.2	System Overview	39
3.3	Detailed overview of the overall system	39
3.4	Texas Instruments FDC1004 [55], [56]	41
3.5	Microchip MGC3130 device on the left and Hover device on the right [55], [56]	41
3.6	Embedded Artists - LPC4088 QuickStart Board [57]	42
3.7	NXP LPC1768 Board [58]	42
3.8	Detailed overview of the overall system	45
3.9	The interactive device - four electrodes distributed in a cross layout	46
3.10	TI FDC1004 - Swipe Left, Swipe Right, Swipe Up, Swipe Down recognition flowchart	48
3.11	Adaptive baseline	48
3.12	Hover extended gesture flowchart	49
3.13	The system's high-level architecture	49
3.14	Quadrants in the dx-dy plane	50
3.15	CapGUI layout	52
3.16	Graphical User Interface main routines: Data reception, Graphic window, Gesture window and Strength window	52
3.17	Data Reception Flowchart	53
3.18	Update Graph Flowchart	54
3.19	Gesture Window Flowchart	54
3.20	Strength Window Flowchart	55
3.21	CoorGUI Layout	55
3.22	Graphical User Interface main routines: Data reception, Track window, Gesture window and Arrow window	56
3.23	Data Reception Flowchart	57
3.24	Arrow Display Flowchart	57
3.25	2D Tracking Flowchart	58
3.26	Server Subsystem	58
3.27	Sequence diagram to send the gesture command to the HMI System	59
3.28	Sending a gesture via TCP/IP Flowchart	60
3.29	Supported Gestures	61
3.30	System Overview	61
3.31	Typical data capacitive sensing pipeline	62
3.32	Evaluate the gesture recognition probability of each model	63
3.33	Confusion matrix Concept[60]	64
4.1	Implementation View	68
4.2	Experimental setup	69
4.3	Raw capacitance value acquired using the FDC1004 sensor and the same signal passing through a IIR filter of 16 samples	71

4.4	Raw capacitance value acquired using the FDC1004 sensor and the integration algorithm for proximity detection	72
4.5	Result of reading the four channels of the Texas Instruments FDC1004	80
4.6	Result of reading the gestures from the Hover sensing device	81
4.7	Result of reading the gestures hand's trajectory with the MGC3130 sensing device	81
4.8	Graphical user interface for TI FDC1004	82
4.9	Graphical user interface for Microchip MGC3130 device	82
4.10	Result of receiving the TCP/IP commands from the microcontroller unit	83

Table List

2.1	Overview of capacitive proximity sensing for gestural interactions	23
2.2	3D Image Technologies Comparison	26
2.3	Comparison between different gestures technologies	30
4.1	Correlation between the size of the electrodes and the maximum distance detection	69
4.2	Texas Instruments FDC1004 - Evaluation of linear gesture recognition performance	84
4.3	Hover - Evaluation of linear gesture recognition performance . .	84
4.4	Hover - Evaluation of linear gesture recognition performance . .	85
4.5	Microchip MGC3130 - Evaluation of linear gesture recognition performance	85
4.6	Microchip MGC3130 - Evaluation of sequential gestures recognition performance	85
4.7	Microchip MGC3130 - Evaluation of rotational gestures recognition performance	86

Abbreviations

CES	Consumer Electronics Show
COTS	Commercial off-the-shelf
DSM	Driver Simulator Mockup
GAHMI	Global Automotive Machine
GUI	Graphical User Interface
HMM	Hidden Markov Models
HMI	Human-Machine Interface
HVAC	Heating Ventilation and Air Conditioning
ID	identification
IDE	Integrated Development Environment
IR	Infrared
ITO	Indium Tin Oxide
IVIS	In-Vehicle Information Systems
MCU	Microcontroller Unit
NHTSA	National Highway Transportation Safety Agency
TCP	Transmission Control Protocol
TI	Texas Instruments
TOF	Time of Flight
TUI	Touch User Interfaces
UART	Universal Asynchronous Receiver/Transmitter

Chapter 1

Introduction

Technological revolution is making a huge impact on automotive industry and people's lives. In spite of the incremental changes since Henry Ford introduced the moving assembly line, the last ten years were marked by changes in the way drivers and passengers interact with their vehicles, the human machine interfaces (HMI). This revolution is still in early stages, being the next steps toward connectivity and automation. Consequently, the analysts forecasts [1] expect the global automotive human machine interface market to grow by 11.25 percent each year over the period 2014-2019.

Therefore, driven by technological advancements, vehicles have steadily increased in sophistication. For example, the BMW 7 series driver-controlled systems, contains over 700 functions [2]. While these systems facilitate to navigate streets, talk on phone and more, they may lead to visual distraction, since when paying attention to a task not driving related, one part of the brain focus on that activity and, consequently, just the remaining part is focused on driving. That distraction is, according to studies, the third cause of accidents, only surpassed by speeding and drunk driving [3].

Driver distraction is stressed as the main concern by regulators, in particular, National Highway Transportation Safety Agency (NHTSA), which is developing recommended limits for the amount of time a driver needs to spend glancing away from the road to operate in-car features. Diverting attention from driving, at a critical moment, can be fatal. With this in mind, automakers have been challenged to design safer and comfortable Human Machine Interfaces (HMIs), without missing the latest technological achievements.

This thesis aims to develop an gesture-based interactive system gesture using capacitive sensing to implement contactless automotive HMI concepts that, possibly, will allow the user to have a more comfortable and intuitive interface experience while driving.

Capacitive technology is a good solution for automotive HMI concepts since it allows the detection of dynamic gestures. The set of gestures will allow to create a gesture recognition library that will be used to implement many in-vehicles applications such as navigation control, multimedia control, roof control, hood control, among others.

The integration in a simulated environment was not part of the objectives, thus the evaluation of the overall system was performed in laboratory environment.

1.1 Motivation

One-fifth of the traffic accidents are caused by driver distraction [4]. Therefore, the driver cognitive workload has been a subject of study in recent years [5]. To mitigate this problem, it is necessary to develop new interfaces that enable more natural, intuitive and safer interactions. Consequently, this dissertation aims to develop these interfaces, a gesture recognition interface that will possibly reduce the number of accidents and allows a more pleasant experience while driving.

Following this trend, gesture recognition offers a good possibility for designing better interfaces because it demands less cognitive effort and attention. With this solution, it is possible to create safer, simpler and more flexible HMI. Furthermore, it has been proved that drivers would accept gesture control for in-vehicle application [6]. In addition, a driver distraction research concludes that at least one hand can be used to perform gestures without distracting the driver [5]. Subsequently, according to a study [7] conducted by IHS (Information Handling Services), the number of gesture recognition or proximity awareness systems is expected rise sharply in the next years.

1.2 Research questions and Objectives

This research aims to explore the capacitive sensing technology for contactless proximity detection or recognition of simple gestures that would allow testing many in-vehicle functionalities in a simulated environment. During the process, the present research attempts to answer the following questions:

- What devices, based on capacitive sensing, should be used for the development of automotive HMI concepts?
- What hand gestures can be accurately recognized for the development of automotive HMI concepts?

This research questions lead to the following objectives:

- Study the capacitive technology, and select the devices, based on that technology, to be used for gesture recognition;
- Implementation of a gesture recognition library to interface with each device;
- Implementation of a functional prototype in order to test several HMI concepts;
- Validate the system in terms of accuracy, efficiency and reliability.

1.3 Contributions

This dissertation is primarily concerned with gesture recognition in order to develop new concepts of interaction to be applied in automotive HMI. Thereby, the results of the developed scientific work are an overview of the market about the existent solution; the theoretical fundamentals of capacitive sensing and a literature review on this technology; and finally, the implementation of the gesture recognition algorithms.

1.4 Document Structure

The present dissertation is divided into five chapters and is organized as follows:

- Chapter 2 - State of Art and Literature Review. This chapter presents the relevance of usability in interaction design and the impact they have on the automotive industry. Then, focus on gesture recognition system and compare the different technologies. Lastly, it presents the theoretical fundamentals of capacitive technology and the mathematical models used for gesture recognition;
- Chapter 3 - Analysis and System Specification . It presents the system architecture and the algorithms to be implemented;
- Chapter 4 - Implementation and Results. It describes what was implemented according to the strategy planned and performs a critical analysis to the results achieved after the implementation and testing of the different algorithms.;
- Chapter 5 - Conclusions and Future work. This chapter answers the research questions and propose new approaches that were not explored.

Chapter 2

State of the art

This dissertation is within the area of gesture recognition and automotive, therefore it approaches gestures in general and then it focus in the automotive industry.

Relevant for this dissertation is also the study of capacitive sensing, where the theoretical fundamentals are presented as well as some examples found in literature that use this technology for gesture detection. Moreover, a review about the complementary alternatives technologies that could be used for the same purpose can also be seen. Then, the different concepts that automotive industry is developing will be presented. Lastly, some mathematical models, such as HMM (Hidden Markov Models), DTW (Dynamic Time Warping) and Threshold based models are explained.

2.1 Gestures

Since the origin of mankind up to the present, humans felt the necessity of developing several forms of communication to exchange information. Being gestures one natural way for the humans to exchange information. It is worth noting that gestures are normally used as a complement of speech, nevertheless they can be used as a substitute.

There are a variety of definitions for gestures, depending on the specific research field. According to Kendon [8], the gestures correspond to a movement of individuals limbs of the body that are used to communicate information to others. In a gesture recognition perspective, gestures can be identified by their

movement trajectory. Albeit the different definitions for gestures, they can be classified according to their features, that topic will be addressed in the following section.

2.1.1 Types of Gestures

The classification of gestures was extensively studied in the academic and expert literature among different areas, such as computer vision, human-machine interfaces and psychological areas. However, they can be divided in touch or free-form gestures and static or dynamic gestures.

Touch gestures and Free-form Gestures

Gestural interfaces can be categorized as contact-based/touch or free-form [9]. The touch user interfaces (TUIs) require the user to be touching the device directly, thereby creating a constrain on the types of gestures that can be performed. Free-form gestural interfaces does not require the user to be in contact, although some technologies require a hand held controller or glove to be used.

Static and Dynamic gestures

Another classification can be established by analyzing the time of the gesture: static gestures and dynamic gestures. Static gestures can be comprehended as the position and orientation of the fingers as well as the hand in a single moment. These type of gestures have no movement. On the contrary, the dynamic gestures are defined by the hand's motion, such as swiping, rotating and waving.

2.2 Human Machine Interface

Gestures are used to control many every life products, such as motion-activated sink and motion sensors used to turn on lights. With new emergent

technologies, such as Kinect and Leap Motion, the use of the body as the only input device has become more common.

Gestural interfaces, named Natural User Interfaces (NUIs), allow a person to interact with a machine using only gestures. There are several applications for different areas: gaming [6], industry and robotics, consumer electronics, smart homes, automotive and others.

2.2.1 Automotive HMIs

There has been a great deal of studies towards gesture interaction with infotainment systems while driving. Chongyoon Chung and Esa Rantanen [2] created a gestural interface, a steering wheel with two touch pads, to recognize gestures for audio and climate control. Through this process they discovered that drivers preferred gestural interaction to voice commands when the control was simple and repetitive. Research towards standardization of the in-vehicle gestural interaction space [3] was also an object of study, in order to design guidelines to uniform all gestures devices in terms of area of interaction and association of the same gesture to a similar functionality in different vehicles. Another study [10] claims that proximity sensing in a two-dimensional plane is a viable approach to directly control a mouse cursor on a screen integrated into the dashboard. Moreover, Google created a patent in which predetermined gestures, in certain areas inside the vehicle, correspond to certain in-vehicle functionality. Figure 2.1 depicts the driver interaction with the vehicle.

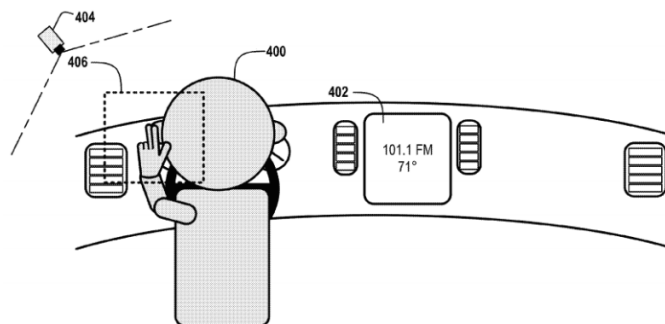


FIG. 4A

FIGURE 2.1: Gesture-based automotive controls from the Google patent [11]

2.2.2 Usability

The movements towards usability have emerge having to do with a few design considerations. Instead of defining usability Nielson [12] proposed five attributes to consider: learnability, efficiency, memorability, errors and satisfaction. The most widely accepted definition of usability, described in ISO 9241, is *"The extent to which a product can be used by specified users to achieve specified goals with effectiveness, efficiency and satisfaction in a specified context of use"*. Figure 2.2 depicts usability as the combination of the attributes: Efficiency, Effectiveness, and End-User Satisfaction.

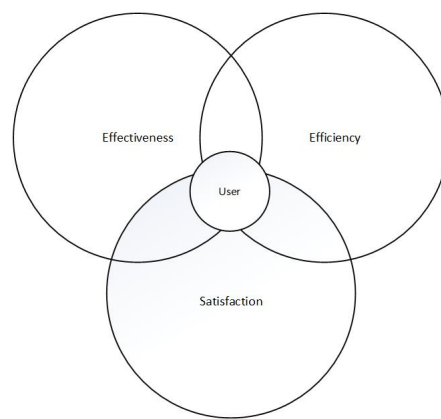


FIGURE 2.2: Usability Diagram

Effectiveness

A product is called effective if it does what it was designed for, in other words, if it is successful in producing the desired or intended result.

Efficiency

The efficiency is a metric that considers the users effort to perform one task. This metric is typically measured in time.

Satisfaction

The effectiveness and efficiency have a concrete definition, in contrast with satisfaction. Since the satisfaction concept is highly subjective, many significant

authors [13] in the usability field added the following components within the concept of usability:

- Guessability
- Learnability
- Memorability
- Flexibility
- Safety

2.2.3 Usability of In-Vehicle Information Systems (IVISs)

People interact with a great number of products, such as, television, smart-phones, and others. Frequently, it is difficult to use them, because the usability when designing was not prioritized. Confusing and non-intuitive interfaces can lead to frustration when operating these systems.

This is particularly important in the automotive industry since it is a crucial aspect to driver's acceptance of in-vehicle technologies. The HMI systems have that surprise factor that, usually, is a great marketing tool, however, the bad user experience (UX) may discourage to invest in these sophisticated systems, leading even the drivers to blame the automotive brands for buying that car. Therefore, the tendency is to follow a user-centered approach, considering the usability of the HMI systems in the initial stages of the design process in order to provide a great usability and good user experience (UX), but also to reduce the costs in the case of usability issue is detected.

The new HMI system requires people to adapt to the new methods of interaction, but if it requires too much time they tend to not use it again. Consequently, the learning time must be reduced by providing an intuitive interface.

Another aspect to be considered is the evaluation methods for in-vehicle interface. Ideally, these methods should be quantitative, based on measurable attributes of usability, such as, interaction times, performance error rates or user eye movements. A quantitative approach would allow for different interfaces to be compared directly. However, some usability factors, such as user satisfaction are impossible to assess objectively, therefore an overall estimation will be a combination of objective and subjective metrics.

2.3 Gesture Recognition in Automotive

The automotive industry is always attempting to improve their vehicles and gesture recognition will be the next trend. Therefore, they are already working in some gesture recognition technology. Several automakers including Audi, BMW, Cadillac, Ford, GM, Hyundai, Kia, Mercedes-Benz, Nissan, Toyota and Volkswagen demonstrated interest in gesture in-vehicles interaction and are in the process of implementing into their automobiles. Some of these systems will be addressed in the following sections.

BMW iDrive

Shown at Consumer Electronics Show (CES) 2015, the BMW 7 series model was the first model to include gesture recognition control as a possible source of interaction. In this vehicle, a time of flight (TOF) camera is positioned in the headliner and detects the position of a hand by monitoring the dashboard space between driver and passenger. It possesses four predefined gestures: twirling a finger clockwise and anti-clockwise; pointing and swiping a finger. Additionally, two gestures can also be programmed to perform a set of predefined functionalities. These gestures, performed in different contexts, can be used for controlling the audio, phone and navigation.



FIGURE 2.3: BMW iDrive concept [14]

BMW i8 iVision

At CES 2016, BMW presents their new model: BMW's iVision Future Interaction concept. The concept also uses TOF camera to recognize the driver's

hand when he moves it over the console center. The new feature, as shown in Figure 2.4, is the 21 inches' screen that is divided into four screens. The closer you move your hand to the screen, the more it highlights, allowing to select a particular function. Therefore, it will work as a touch screen but with no contact. However, when a menu or icon is activated, a switch, located on the steering wheel, briefly illuminates to confirm the action. Similarly, there is also a switch button for the passenger located on the side panel.



FIGURE 2.4: BMW i8 iVisio concept [15]

Hyundai Genesis

Hyundai showed a concept car with gesture recognition system to control radio volume and tracks, the HVAC (Heating, Ventilation and Air Conditioning) and the navigation systems and even smartphone connectivity functions. Hand gesture recognition is accomplished with advanced infrared and camera sensors. It provides an interesting approach since it allows, aside from the driver, let the passenger have gesture recognition system and the capability to interact with the main user. For example, the passenger can select a GPS route and with a swipe gesture pass it to the driver side.



FIGURE 2.5: Hyundai Genesis concep [16]

VW Golf R-Touch

In contrast to a more moderate approach by the other automakers, Volkswagen attempted to eliminate all physical dials and buttons, switching them by sensitive elements such as touch sensitive, proximity-aware and gesture control. For example, the roof and mirror are controlled by gestures (swiping gestures). The system uses a TOF camera implemented near the roof light capable of detecting one finger pointing and two finger pointing. To trigger the gesture recognition system is necessary to open the hand for a small period of time and only after will the gesture be recognized. This system allows the driver control of the mirrors, main menu, radio control, among others.



FIGURE 2.6: Gesture controlling sunroof of the Volkswagen Golf R Touch [17]

VW E-Golf Touch

In the VW E-Golf Touch the gesture controls system was more limited than those on the Golf R Touch concept due to a more cost-friendly infrared sensor instead of the more capable TOF camera. It is still possible to swipe left or right to select the next/previous song, but now there is no hand gesture for pausing or playing a track.



FIGURE 2.7: Gesture controlling E Golf Touch [18]

VW BUDD-e

Gesture recognition can begin from the exterior, with an intuitive swiping gesture the sliding door opens or closes, a simple foot movement opens the electrically operated tailgate. Gestures are recognized immediately, thus no needing to be explicitly activated for the gesture control system as it was the case in the Golf R Touch. Furthermore, the maximum operating distance have been considerably increased.

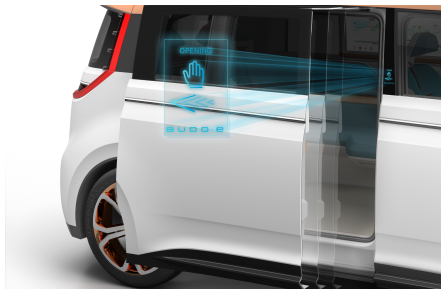


FIGURE 2.8: Volkswagen BUDD-e sliding door [19]

Visteon

Visteon begins to offer spatial gesture controls (swipe up, down, left, right, and rotary motions) with no cameras-solutions, being considerably less expensive than camera-based solutions. For more complex gestures, it uses TOF camera located almost at the middle of the console. The gestures allows the driver to touch-free control of automatic window control, audio and radio output, temperature, among others. It was possible to see this system in Ford C-Max at CES 2016.

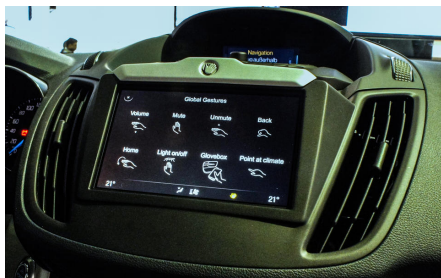


FIGURE 2.9: Recognizable gestures by Visteon system [20]

2.4 Capacitive Sensing

Capacitive sensing is a well understood technology with its first application dating to early twentieth century, namely the Theremin musical instrument, where the pitch and volume of the sound were both controlled by the distance of the musician hands to the two antennas. These instruments are still on the market, produced by Moog Music Inc.



(A) The first working model played by his inventor - Leon Theremin [21]



(B) Theremin by Moog Music Inc released in 2014 [22]

FIGURE 2.10: Theremin - a contactless musical instrument

2.4.1 Capacitive Proximity Sensing

Any living organism produces a small electric field generated by cell activity and ionic currents of the nervous system [23]. Subsequently, it is possible to measure the influence of the human body within external electric fields or couple a transmitter and measure the resulting electric field.

To better explain how it works lets remember that the capacitance of a capacitor can be expressed as:

$$C = \epsilon_r \frac{\epsilon_0 A}{d} \quad (2.1)$$

where,

- C is the capacitance in farads (F),
- ϵ_0 is the dielectric constant of the material between the plates
- ϵ_r is the permittivity of free space, which is $8.854 * 10^{-12}$ F/m,

A is the area (m^2),
d is the is the separation distance of the two plates (m).

The simplest capacitor consists of two metal plates very close together without touching each other. As the formula demonstrates, the capacitance is directly proportional to the area A and inversely proportional to the distance d and also being dependent on the static permittivity of the dielectric. Conductive objects such as the human hands act as the second plate and, thereby, increase the capacitance C by reducing the distance d .

The basic process of capacitive proximity consists in measure the stored energy. The capacitance value is obtained indirectly by measuring, periodically, the time it takes discharge a sensing electrode. When a conductive object enters the electric field the energy that can be stored is increased, causing a longer discharge time.

On the other hand, new varieties use a field between two electrodes. In this case, the energy is reduced by grounded objects.

2.4.2 Proximity Sensing versus Touch Sensing

It is possible to distinguish three different projected sensing methods:

- **Touch Sensing:** densely distributed sensor generates a weak electric field, allowing to detect one or more objects touching;
- **Floating Sensing:** densely distributed high-sensitivity sensor allows to detect both touch and very near objects (< 2cm) touch;
- **Proximity Sensing:** sparsely distributed sensor generates a strong electric field, allowing to detect objects over 20 centimetres.

2.4.3 Sensing Configurations

In his research "Electric Field Imaging" [24], Joshua Smith introduced different measuring modes that can be distinguished in capacitive sensing:

- **Loading-mode:** A periodic electric signal is applied to a single electrode. When a user approaches the electrode, the conductive properties of the human body form a weak capacitive link with the electrode, causing a signal change of total capacitance. By sensing the quantity of this change, touch or proximity is detected;
- **Shunt-mode:** An electric field is generated by applying a periodic signal to a transmit electrode. This field is then captured by one or more receive electrodes in proximity. The conducting properties of the human body absorb some of radiated field and shunts it to ground, causing variations in signals at the receive electrodes;
- **Transmit-mode:** The input signal is coupled with a person's body. This coupling turns the user into a transmitter whose signal can be picked up by one or more receivers.

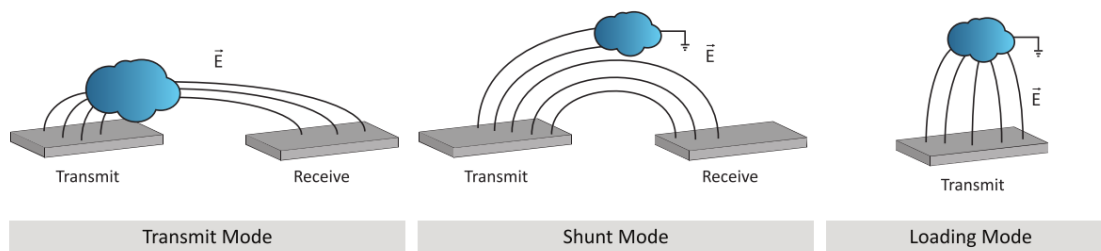


FIGURE 2.11: Measurement modes for capacitive proximity sensing [25]

2.4.4 Material and Geometry

One interesting feature of capacitive proximity sensors is the versatility they offer, in the sense that with different electrode materials, size and geometry it is possible to create highly personalized applications. For example, the electrodes materials include transparent metal oxide layers, woven conductive thread, copper wires, PCB boards, or simple aluminium foil. The material of the electrode should be selected according to the desired application. Some applications might require that the interaction device has a flexible surface, while on others applications the surface must be solid and opaque. In the first case, conductive thread could be used and solid metal electrodes seems to be a viable option for the second case. Additionally, there are other alternatives for

transparent materials. In his Master's thesis [26], Yannick Berghöfer evaluated the different types of electrodes materials including copper, ITO (Indium Tin Oxide) and PEDOT:PSS. ITO is a thin layer of indium titanium oxide which is highly conductive and PEDOT:PSS is conductive polymer that has a lower conductivity. Figure 2.12 shows the electrode spatial resolution at different distances between the object and the electrode. The spatial resolution is a measure used to evaluate the expected precision of the measurement, being based on collecting a time series of multiple samples and calculating the mean distance and standard reference.

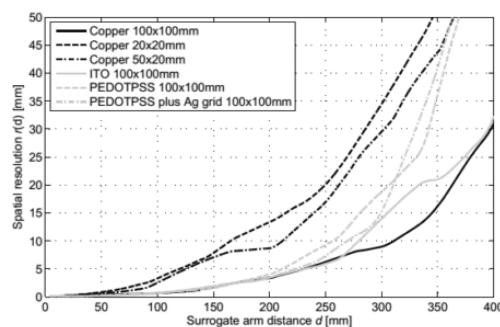


FIGURE 2.12: Spatial resolution of different materials at various distances [26]

His research concluded that copper has the best proprieties but ITO can still be used when transparency is required.

Another important aspect is the electrodes geometry since the layouts include simple straight wires [27] [28], plate electrodes, and more complex multi-dimensional structures [29]. In order to create a deliberate and natural interactions. Andreas Braun and Pascal Hamisu [27] [28] use capacitive proximity sensing to detect the presence of the human body. More specifically, as Figure 2.13 shows, they use wire antenna as a sensing electrode

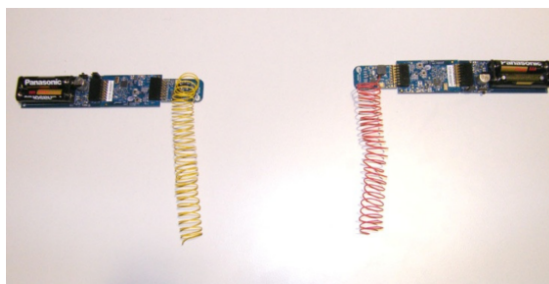


FIGURE 2.13: Prototype device supporting antenna wire [27]

A threshold model was implemented with two different sensing modes: Proximity Mode and State Mode (Figure 2.14). The former is the direct measure of the capacitance value, which is directly proportional to the proximity. The latter is a state model, basically, a quantization of the values in n states which can be used to model capacitive buttons.

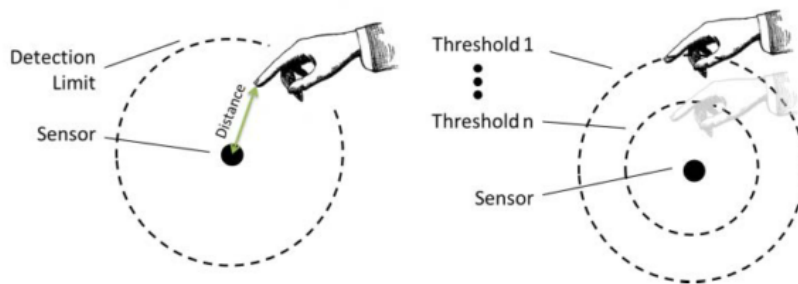


FIGURE 2.14: threshold model with two different sensing modes

A prime example of the more complex multi-dimensional structures is the GestIC technology by Microchip Technology. Based on shunt mode, the transmit electrode is at the bottom layer and four smaller receiver electrodes are placed on the edges of the top layer (Figure 2.15).

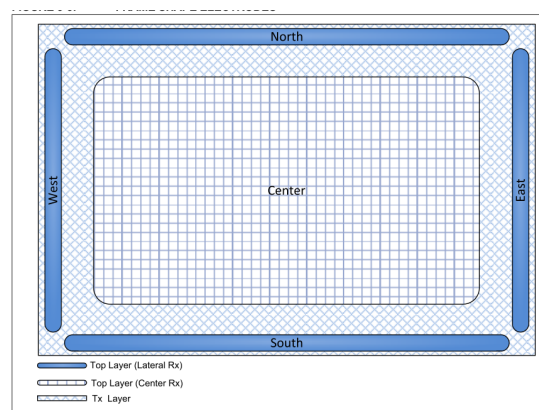


FIGURE 2.15: Microchip MG3031 electrodes layout [29]

2.4.5 Shielding

When it is anticipated that other objects might disturb the measurement, shielding should be used. The electric field will span in all directions unless it uses a shield to prevent detecting objects approaching from a certain direction.

Being more relevant when various electronic devices are integrated into the same prototype and it is necessary to minimize disturbance.

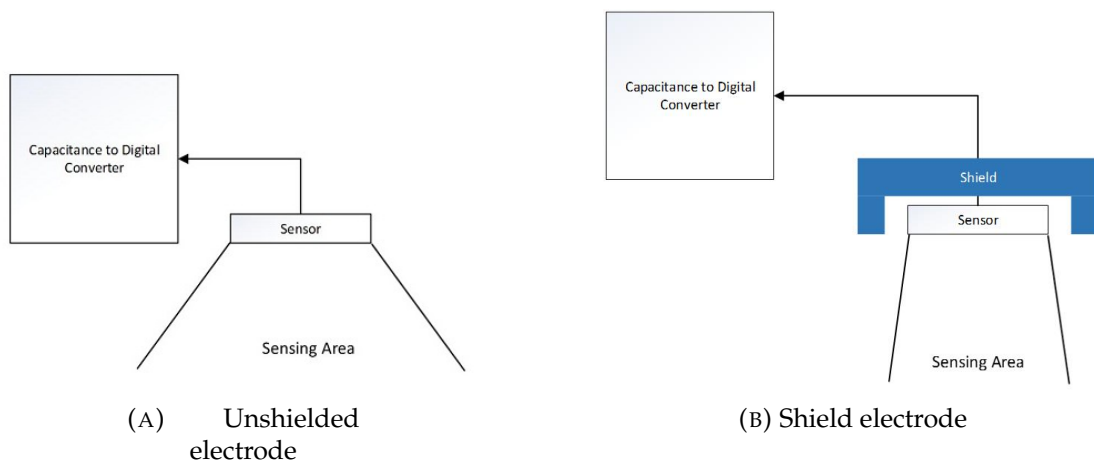


FIGURE 2.16: Differences between with shield and unshielded electrodes

Another important aspect to take into consideration is to use a shield cable when connecting the sensing electrode to the capacitive to digital converter. Thus, mitigating external interferences such as human hand, radiated electromagnetic signals, and noise from other electronic devices.

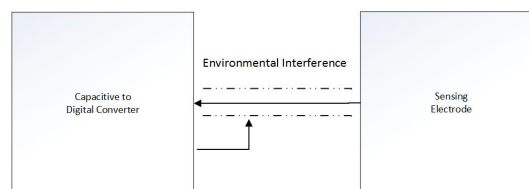


FIGURE 2.17: Shielding to mitigate environmental interference

2.4.6 Applications

This section provides a brief overview on researches using capacitive sensing for gestural interaction in terms of the prototype specification, the high level processing algorithms and their accuracy.

Active Armrest

Active Armrest [30] is a prototype that supports two different types of gestural interaction in the domain of automotive applications: touch and free-air

gestures. An array of capacitive proximity sensors detects the arm pose on the armrest as shown in Figure 2.18.

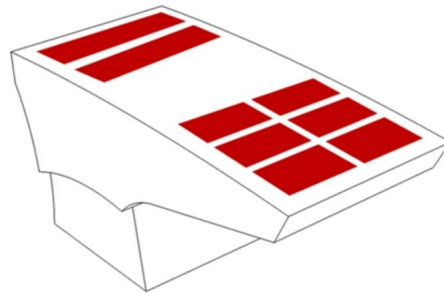


FIGURE 2.18: Active armrest sketch with six electrodes for finger gesture recognition [30]

Two electrodes are used to verify if the arm is resting, only triggering the finger gestures interactions when the arm is lifted. The arm presence detection is binary and the finger position in three dimension is calculated using a weighted average and interpolation, proceeding to use a SVM classifier to classify each gesture. They investigated the detection rate of the gesture recognition system with eleven participants, the touch set performed was considerably better than the free-air set. For the touch set, the results ranged from 77.3% and 90.9% for each gesture, for the free-air set the results, ranged from 45.5% for counter-clockwise circles to 81.8% for right swipes. The researchers stated that if more sophisticated methods, e.g. a random forest approach, for finger tracking were used, better results could be achieved. Lastly, they concluded that the system must be tested in a driving simulator environments for more realistic results.

Tracker

Tracker is a prototype that augments a regular monitor to detect hand gesture using capacitive proximity sensors allowing new interaction modes as picking and dropping an object on the screen [31]. As shown in Figure 2.19, it has four sensors arranged around the screen. They defined two modes: 3D interactions and "Pick and Drop" interactions. In the first mode, the movements will be interpreted as clicks within three centimeters of the screen and pointer movements for farther away. The second mode permits the interaction with objects on the screen by performing a picking gesture. Researchers concluded, by studying 10 participants, that absolute and dynamic positioning using gesture interfaces is reasonably intuitive and comfortable. They struggled to achieve

good results when performing gestures as zooming that with some optimization into the hardware design could have been overcome.



FIGURE 2.19: Thacker - four sensors arranged around the screen [31]

Swiss-Cheese Extended

Swiss-Cheese Extended [32] is a prototype for gesture recognition using capacitive proximity sensors to detect the three-dimensional position of multiple hands. The gestures supported are: swipe from left to right with a single hand and from bottom to top with two hands, combined zoom and rotation gesture and the corresponding zoom and rotation axis, grasp and release actions.



FIGURE 2.20: Swiss-cheese prototype [32]

The prototype device employs shunt mode measurements. Two receivers are placed in the center while eight transmitters are located at the device's edges. They achieved a resolution of approximately 3.5mm at object with distances around 50mm, and 35mm at object with distances of 200mm.

Textile Capacitive Sensor Arrays

Gesture Recognition using Textile Capacitive Sensor Arrays is a personalized gesture recognition system for people with upper extremity mobility impairment [33].

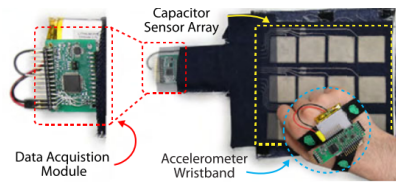


FIGURE 2.21: Interaction between the data acquisition module, the capacitive sensor array, and the accelerometer wristband [33]

Through fusing a wrist-worn accelerometer and the sensor array, the gesture recognition system can correct the rotation of the sensor array with respect to the hand. They evaluated the system with six participants and recognized gestures with an accuracy of 99% using DTW and HMM.

Rainbowfish

Rainbowfish combines a semi-transparent capacitive proximity-sensing surface for gesture recognition with an LED array for visual feedback [34].



FIGURE 2.22: Rainbowfish prototype [34]

The measurements were conducted in loading mode on the twelve transparent electrodes, which are made of Indium-Tin-Oxide (ITO).

Electric Field Sensor for 3D Interaction on Mobile Devices

With a purpose of detecting free-form gestures on mobile devices, the researchers [35] created a transparent electrode array to be measured by the Microchip sensor (MGC3130).



FIGURE 2.23: Gesture recognition on mobile device [35]

Then, by applying machine learning (RDF-based regression), the system allows simple visualization of the 3D input sensed from the device and the detect simple in-air swipe gesture on mobile devices.

Overview of capacitive sensing

The Table 2.1 summarizes the researches described before.

TABLE 2.1: Overview of capacitive proximity sensing for gestural interactions

Name	Description	Measuring layout	Data processing
Active Armrest [30]	System that allows finger gestures and armrest position to control in-vehicle infotainment system	Loading mode, six electrodes	SVM classifier
Thracker [31]	Track hand gestures in front of a screen	Loading mode, four electrodes	Gesture based on nearest object to electrode
Swiss Cheese Extended[32]	Track the three-dimensional coordinates of multiple hands	Shunt mode, ten electrodes - two receivers and eight transmitters	Threshold
Electric Field Sensing for mobile devices [35]	Recognize in-air gestures on mobile device	Shunt Mode five electrodes - four receivers and one transmitter	Random Decision Forest
Rainbowfish [34]	Visual feedback on gesture recognition	Loading Mode twelve electrode	Interpolation
Adaptive and Personalized Gesture Recognition Using Textile Capacitive Sensor Arrays[33]	System capable of recognizing gestures for users with limited mobility.	Loading mode, eight electrode	HMM and DTW

2.5 Alternative Sensing Technologies

Gesture recognition systems can be conducted by many technologies, such as infrared, capacitive, controller-based gestures, time of flight cameras, structured light cameras, stereo or single cameras and radar. Therefore, a study comparing most technologies is presented. They can be divided into the following categories: imaging technologies and non-image technologies (see Figure 2.24).

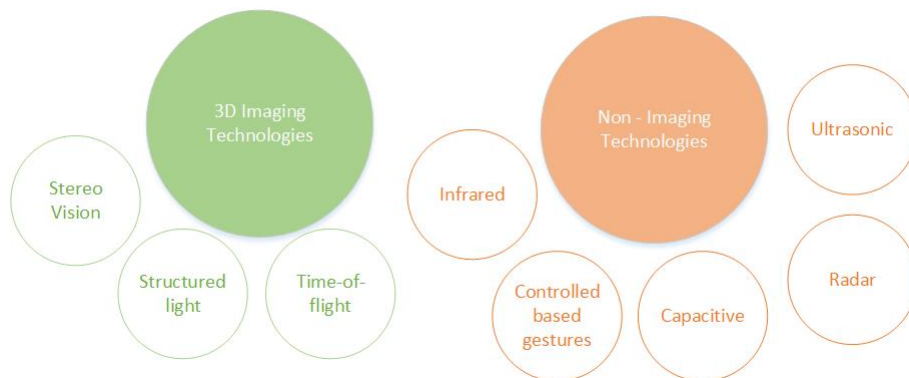


FIGURE 2.24: Gestures Recognition Technology

2.5.1 Imaging Technologies

Imaging system are widely used in different applications such as, surveillance systems, laptops, vehicles, smartphones. Being a well-known technology, the cameras can be successfully used as gesture recognition applications. It can perform complex gesture recognition as they can obtain the desire shape and form and after applying an algorithm it is possible to track the form in two/three dimensions. There are two types of systems: 2D and 3D cameras.

Although 2D cameras are widely used, this approach is only recommended if the lighting is controlled and the solution must be inexpensive. Moreover, because the information is limited (no depth sensing), an additional sensor 1D sensor or just a 3D camera should to be bought. The 3D cameras can overcome many of the 2D problems, since depth sensing can be used to distinguish foreground from background. In gesture recognition, the scene understanding is required to enable the detection of the face, hands and fingers. The next section will compare three 3D vision technologies: Stereo Vision, Time of Flight and Structured Light.

Stereo vision

As shown in Figure 2.25, stereo vision generally uses two cameras separated by a known distance (usually 6.35 cm) and by comparing images about a scene from two vantage points, 3D information can be extracted.

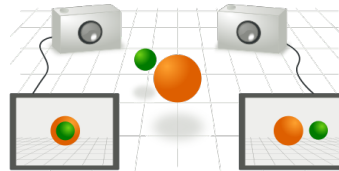


FIGURE 2.25: Stereo Vision Concept [36]

This method is considerably complex due to correspondence problem, which is, ascertaining which parts from scene A corresponds to which parts from Scene B. Solving this problem requires computationally intensive algorithms. Furthermore, for a robust correlation it is necessary sufficient intensity and colour variation. The major advantage of this technology is that the implementation cost is very low.

Structured light

Structured light can provide a depth map by projecting a fixed pattern of infrared spots onto the object (often grids or vertical/horizontal bars). Then, the receiver inspects the pattern distortion, the shifted grid of these spots, and the processor calculates the offset of each of the spots. Figure 2.26 shows the process of projecting vertical bars and with the distortion provoked by the hand, the surface shape can be reconstructed.

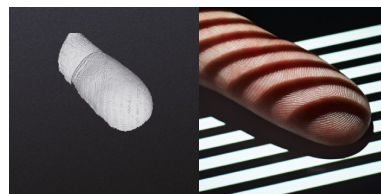


FIGURE 2.26: Structured Light Pattern [37]

This technology was implemented in the first version of Microsoft Kinect sensor and can achieve relatively high spatial resolution but is sensitive to optical interference. Therefore, structured light seems to be better applied to 3D scanning of objects.

Time of flight

Time of flight works by projecting a modulated light source, and observing the reflected light. The phase shift is then measured and translated to distance (Figure 2.27).

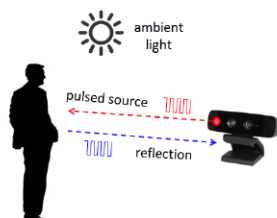


FIGURE 2.27: Time of Flight Concept [38]

As shown in Figure 2.28, the brighter the intensity the closer the object is. Consequently, it is easy to establish the difference between the foreground and background



FIGURE 2.28: Depth image [39]

Imaging technologies Comparison

Table 2.2 summarizes the differences between the imaging technologies.

TABLE 2.2: 3D Image Technologies Comparison

Considerations	Stereo Vision	Time of Flight	Structured Light
Material Cost	Low	Medium	High
Response Time	Medium	Fast	Slow
Depth Accuracy	Low	Medium	High
Low-light performance	Weak	Good	Good
Bright-light performance	Good	Good	Weak
Software complexity	High	Low	Medium

Stereo vision cameras are the least expensive, whereas it is weak in low light conditions and has high software complexity. Next, the structured light cameras is the most accurate even in low light conditions, although it has the highest response time and material cost. Finally, time of flight cameras may be the best option due to fast response time, as they deal well with light and they have the lowest software complexity.

2.5.2 Non-Imaging Technologies

Most technologies use camera-based technologies, nevertheless new solutions, more cost friendly, have emerged. This section describes some of them such as ultrasonic, wearable devices, capacitive, infrared and radar technologies.

Ultrasonic

The ultrasound technology works by calculating how long the signal takes to come back by microphones after the sensor sends it out from speakers to the air. If the object is very close to the sensor, the signal comes back quickly, and if the object is far away from the sensor, the signal takes longer to come back (see Figure 2.29). These sensors can be used for contactless proximity or gesture detection. The range of applications goes from a simple distance detection (1D) to gesture detection (3D). Gesture detection using multiple ultrasonic sensors can be seen in a Soundsense. SoundSense [40] uses ultrasonic sensing to detect 3D gestures (a set of 12 gestures), which are recognized by 4 ultrasonic rangefinders mounted on the Android tablet achieving 82.2 % accuracy. They had some issues with false triggering because the beam cone would diverge after a certain distance from the source and lead to some false positive recognitions and if the ultrasonic sensors were not synchronized they could interfere with each other.

Recently, Jaguar and Land Rover invested in a start-up company called Ultrahaptics due to their cool gesture control technology. This technology allows to make a touch operation in the air and feel it as a physical button. This tangible and invisible interface will create a touch area by combining several acoustic high pressure points.

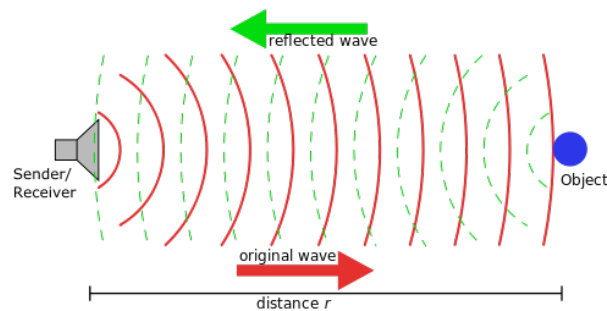


FIGURE 2.29: Ultrasonic Concept [41]

This technology should be considered essentially for distance detection as it is an inexpensive, mechanical reliable and low power consumption solution with a good detection range. Moreover, it is not dependent upon the surface colour or optical reflectivity of the object and has a good detection range. This technology requires the objects to have a minimum size and be the closer to flat. Also, ultrasonic sensors have a minimum sensing distance. Changes in the environment, such as temperature, pressure, humidity, air turbulence, and airborne particles affect ultrasonic response. Another of the problems is that surfaces with low density materials, that are likely to absorb the sound. Lastly, among the other the ultrasonic sensors tend to be slower.

Infrared

Infrared technology works by detecting the reflected infrared light by an object, i.e. hand, from the projected one. The intensity of light reflected and detected by the sensor is proportional with distance, meaning that the closer the object is, the higher the intensity (see Figure 2.30).

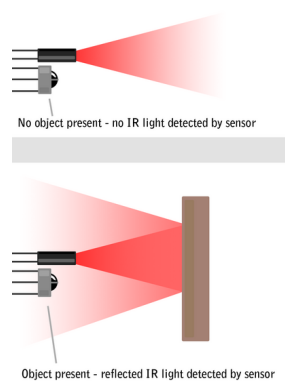


FIGURE 2.30: Infrared Concept [42]

Infrared is widely used because it supports 1 to 3 dimensions of sensing. It has good detection range, good cost, good reliability and good power consumption. On the other hand, in terms of accuracy, linearity, resolution, colour change sensitivity and light sources interferences.

Controlled based gestures

Wearables offer great opportunities to enhance your driving experience. These devices act as an extension of the human body, and making it possible to perform some intuitive gestures. The interesting feature is that, as they pack different sensors, they can be used for more than just gesture recognition. Figure 2.31 depicts the MYO armband, that monitors the muscle activity of the human arm for gesture recognition. The main limitation of these systems is battery life, size (in some cases), accuracy and cost.

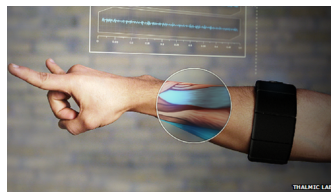


FIGURE 2.31: Controller based gestures principle [43]

Radar

Radar stands for radio detection and ranging. It emits radio waves in pre-determined directions, if these reflected waves are received again at the place of their origin, it means that an obstacle is in the propagation direction. The biggest challenge was to shrink this technology into something tiny enough to fit on a microchip. Google achieved it (Figure 2.32) and claims that it will be more accurate than tracking cameras.

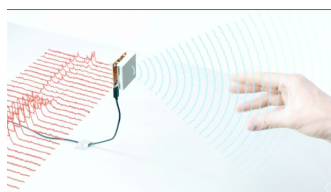


FIGURE 2.32: Radar principle [44]

Comparison

Table 2.3, summarize the technologies in terms of detection range, limitation, processing complexity and cost.

TABLE 2.3: Comparison between different gestures technologies

Technology	Environmental Influences	Detection Range (cm)	Material Cost	Processing Technology
Ultrasonic sensing	Acoustic occlusion, absorbing materials	Medium distance (10 - 600)	Low	Low
Capacitive proximity sensing	Electric fields, moisture, temperature, conductive objects	Near distance (<20)	Low	Low
Infrared proximity sensing	Occlusions, external infrared lights, color dependent	Near distance (<20)	Low	Low
Radar sensing	Electric fields, moisture, temperature, conductive objects	Near\Medium distance (10 - 1000)	High	High
Wearable technology	Electric fields, moisture, temperature, conductive objects	-	Medium	Medium

2.6 Mathematical Models for Gesture Recognition

2.6.1 Threshold Model

Threshold model is any model that relies on a threshold value or a set of threshold values. The threshold value is used to distinguish a range of value

where the normal behavior of the model varies.

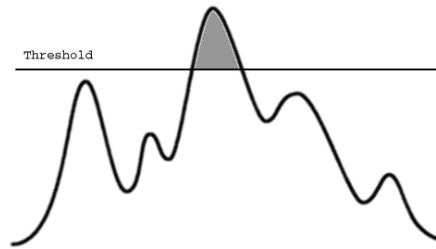


FIGURE 2.33: Threshold model [45]

Threshold model should be applied when the simplicity of the system is prioritized. This model can achieve good results with low level of complexity [46], [47]. In some cases the static threshold is not applicable due to environment changes. To overcome this problem an adaptive threshold is needed.

2.6.2 HMM

Markov Model is a mathematical model of stochastic process used to model a sequence with a finite number of states assuming that future states depend only on the current state and do not consider events that occurred before it [48]. When the states are hidden because they cannot be directly observed, the Markov Model is called Hidden Markov Model. At each state an output symbol is emitted with some probability and the state transition with another. If the observed variable is discrete, this model is called discrete HMM or DHMM. It can be expressed as:

$$\lambda = \{A, B, \pi\} \quad (2.2)$$

where,

- N is the number of states in the model $S = \{ S_1, S_2, \dots, S_1^N \}$,
- V is the number of distinct symbols $V = \{ V_1, V_2, \dots, V_1^N \}$,
- A is the state transition probabilities,
- B is the state observation probabilities,
- π is the initial state probabilities.

HMM's Problems

There are three problems that can be addressed:

- **Evaluation - the likelihood problem:** Given a observation sequence $O = X_1^T$ and a model $\lambda = \{A, B, \pi\}$, how to efficiently compute $P = (O \setminus \lambda)$, the probability of the observation sequence given the model
- **Recognition - the decoding problem:** Given a observation sequence $O = X_1^T$ and a model $\lambda = \{A, B, \pi\}$, how to select the corresponding state sequence $Q = q_1^T$ which is optimal.
- **Training - the learning problem:** Given a observation sequence $O = X_1^T$ how to adjust the model parameters $\lambda = \{A, B, \pi\}$ that maximize $P = (O \setminus \lambda)$.

The three problems for HMM: Evaluation, Decoding and Training that can be solved by using Forward or Backward algorithm, Viterbi algorithm and Baum-Welch algorithm respectively.

HMM's Topologies

There are some common topologies widely used in the HMM's applications that can be determined using the transition matrix A. The topology of the HMM can be: Fully Connected (Ergodic Model), Left-Right Model and Left-Right Banded.

$$A = \begin{bmatrix} a_{11} & x_{12} & a_{13} & \dots & a_{1n} \\ a_{21} & x_{22} & a_{23} & \dots & a_{2n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ a_{n1} & x_{n2} & a_{n3} & \dots & a_{nn} \end{bmatrix}$$

Full Connected- Ergodic

If the elements $a_{ij} > 0$ any state can be reached from other states.

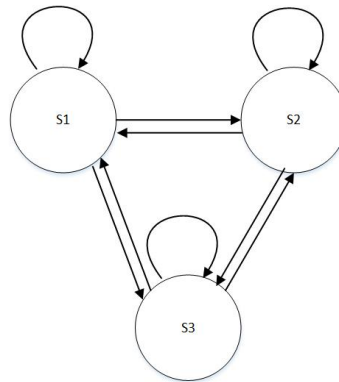


FIGURE 2.34: HMM Ergodic topology with 3 states

Left-Right model

If the elements $a_{ij} > 0$ only for $j > i$, the model is called Left-Right Model allowing each state to go back to itself or to the following states.

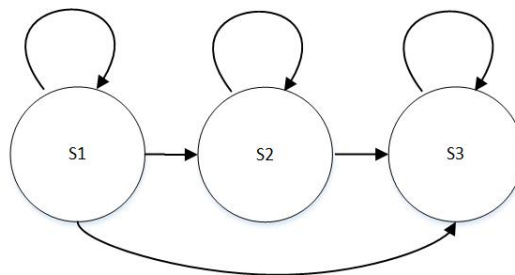


FIGURE 2.35: HMM Left-Right topology with 3 states

Left-Right Banded Model

If the elements $a_{ij} > 0$ only for $j = i$ or $j = i + 1$, the model is called Left-Right Banded. It is very similar to left-right model and allows it to go back to itself or the following state only.

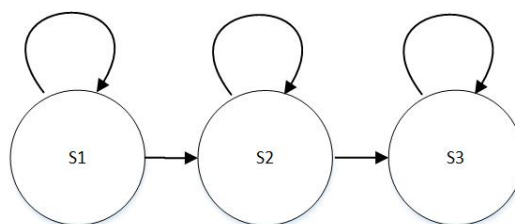


FIGURE 2.36: HMM Left-Right Banded topology with 3 states

2.6.3 DTW

Euclidean Distance (ED) between two time series x_1, x_2, \dots, x_n and y_1, y_2, \dots, y_n is

$$\sqrt{\left[\sum (x_i - y_i)^2\right]}$$

Euclidean Distance performs poorly in similarity, because it is very sensitive to distortions in the time domain, the Dynamic Time Warping was introduced. It is a distance measuring technique, that has been usually used in speech recognition. This technique allows a non-linear mapping one signal to another by minimizing the distance between the two. Figure 2.37.(a) shows two signals that are similar but locally out of phase, and DTW (Figure 2.37.(b)) can overcome that problem. It is a good solution for modeling sequences that are not aligned in the time axis.

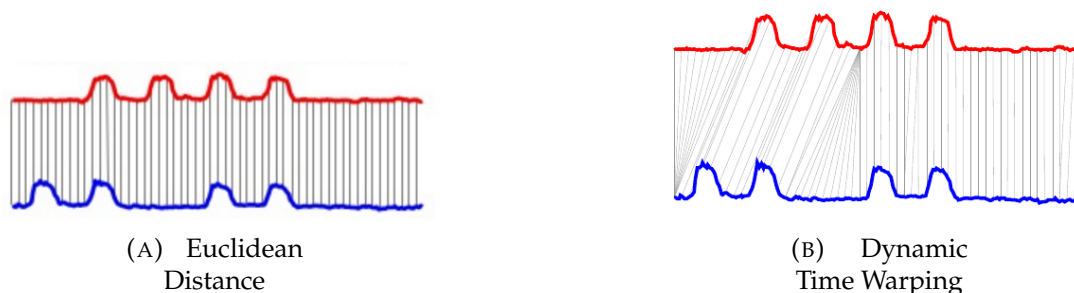


FIGURE 2.37: Differences between the Euclidean Distance and the Dynamic Time Warping applied to the same signal [49][50]

However, in spite of the great accuracy, the processing time is 100 to 1000 slower than Euclidean distance, depending on the length of the sequences. DTW finds the optimal path between the two series sequences, regarding the sequence T and W , the algorithm must ensure their best alignment, making their discrepancy small. Nonetheless there are a variety of possible solutions via the warping matrix (see Figure 2.38), thereby some restrictions must be considered such as the monotonicity, continuity, boundary conditions, warping window and slope constraint. DTW is considered to be computationally expensive, therefore new variations such as SparseDTW [51], the FastDTW [52] and UCR-DTW [53] have emerged.

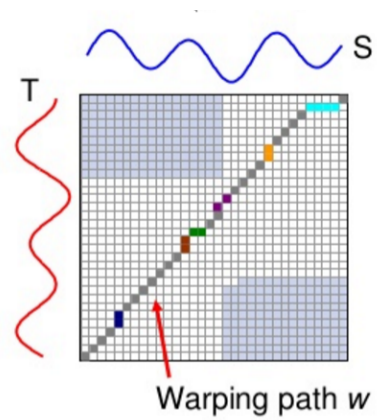


FIGURE 2.38: Dtw path [54]

Chapter 3

Analysis and System Specification

This chapter begins by specifying the methodology used for the development of this research and then it describes the system architecture of the overall system and of each individual subsystem, presenting the system requirements, and also his functionalities, as well as the role and relevance of the selected methods for the developed system.

3.1 Methodology

The conducted research follows the waterfall methodology. This methodology is composed by the phases: Analysis, Design, Implementation, Verification and Maintenance (Figure 3.1).

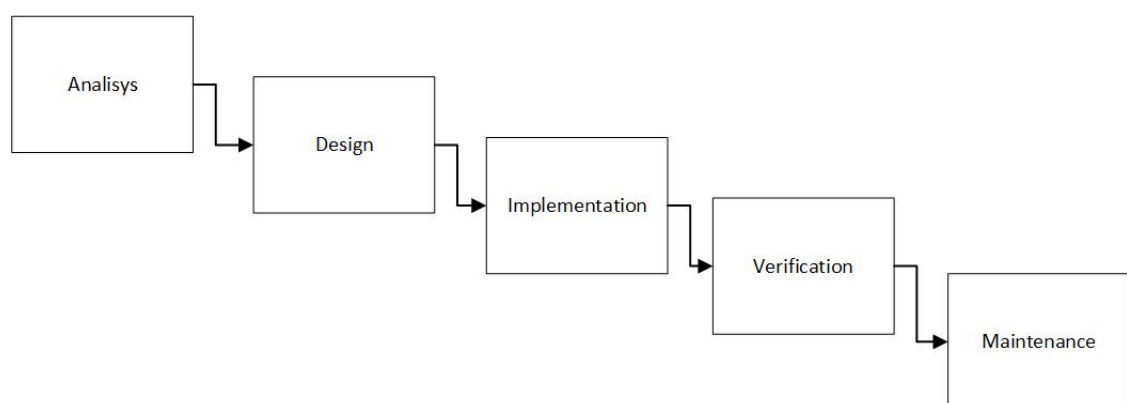


FIGURE 3.1: Waterfall model

In the analysis phase, where the project planning begins, it is important to understand the problems, needs and objectives to achieve.

The second phase is where hardware and software specification, models and methodologies are defined . After that definition and the methodology for implementation and testing, the system implementation begins.

Lastly, the system validation phase, where all the accomplished work is evaluated through tests. These tests will verify if the system works as expected and obtain an idea of the overall performance.

3.2 Project Requirements

The automotive brand is always striving to be leading company. Therefore, they attempt to support innovation and change that drivers are demanding. Whereas, the new car functionalities are a great marketing tool also is to have a competitive price. Subsequently, when they attempt to incorporate gesture recognition into their automobiles, currently, the two economically feasible solutions are infrared and capacitive technologies. Therefore, it was also the two solutions adopted by the project "Project Bosch INNOVCAR: The cockpit of the future". The capacitive technology is a good solution due to the sensors itself can be applied unobtrusively because the generated electric field propagates through any non-conductive material; requires small energy consumption and processing power, which facilitates the integration into embedded systems.

For the selection of system architecture, the functional requirements and non-functional requirements are defined.

The systems has the following functional requirements:

- Use of the capacitive technology for the gesture recognition system;
- Interface with the devices to obtain gesture recognition data;
- Develop the gesture recognition algorithms;
- Communicate the recognized gesture through an Ethernet connection.

The non-functional requirements are:

- Recognize gestures with different hand positions and speeds;
- Communicate the gesture to a central system under 1 second;

- For the different type of gestures, achieve an accuracy of 70% or higher;
- Create a functional prototype to be integrated in a simulated environment;

3.3 System Overview

This dissertation aims to develop a gesture recognition system, where common drivers or passengers can perform a hand gesture to access easily to in-vehicle functionalities such as radio, HVAC, navigation, among others. To accomplish that, a G.R (Gesture Recognition) module is developed to allow the users to perform a set of predefined gesture, those who the G.R. system is capable of recognizing, within a certain area.

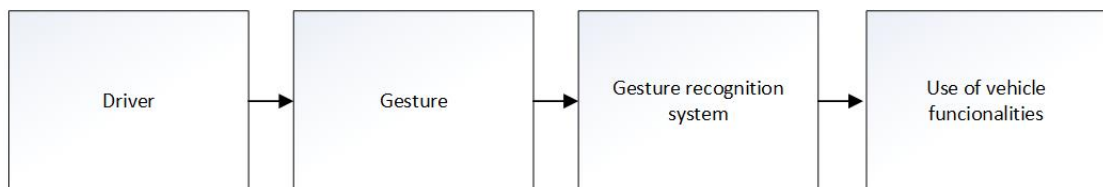


FIGURE 3.2: System Overview

3.3.1 Detailed Overview

The overall overview of the system can be seen in Figure 3.3. The system is composed by three subsystem: sensing devices, microcontroller unit (MCU) and Cluster.

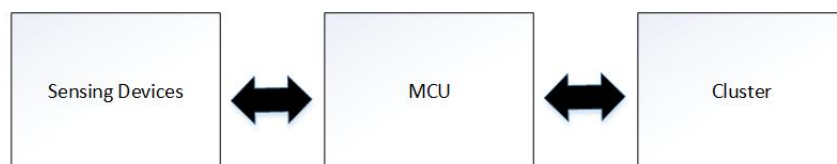


FIGURE 3.3: Detailed overview of the overall system

By analyzing the diagram is possible to identify three steps: the sensing device interface, the gesture detection algorithms and the data communication to the cluster.

3.4 Hardware Specification

The following section will present the different hardware elements used in this dissertation as well as the criterion used for their selection.

3.4.1 Criteria to select the Sensing Devices

The initial step in the project planning was to select which gesture recognition devices, based on capacitive sensing, could be used for the development of automotive HMI concepts. During this step some sensors must be select through an analysis of the market. Since the final system will be only for the validation of the HMI concepts, the sensors validation for automotive grade were considered not essential. The selected of the sensing devices were selected based on the following criteria:

- Detection range up;
- Functionalities;
- Protocol of communication;
- Available in the market;
- Cost;
- Resolution;
- Time-to-prototype.

3.4.2 Selection of the Sensing Devices

For the development of the present research, three gesture recognition devices were selected: Texas Instruments FDC1004, Hover and Microchip MGC3130 according to the previous criteria.

Texas Instruments FDC1004 sensor (capacitance to digital converter) has four channels for measure the electrodes capacitance, that will be used for detect proximity and subsequently gestures. This sensor has low power consumption and use the I2C interface to communicate the capacitance. Moreover, it is inexpensive and it is available on the market in a form of FDC1004EVM development board (Figure 3.5.(b)).

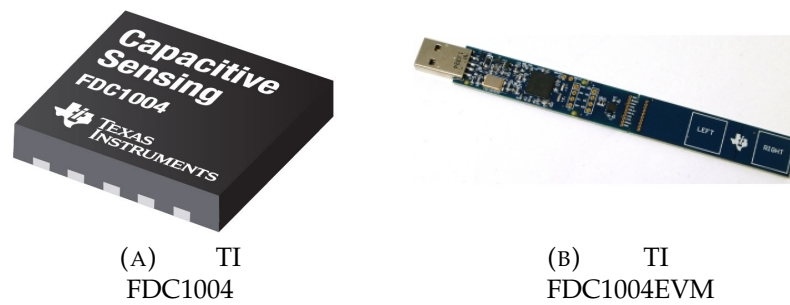


FIGURE 3.4: Texas Instruments FDC1004 [55], [56]

Microchip MGC3130 allows to acquire three dimensional position of the hand with a detection range up to 10 cm. The Hover sensor already recognize the left, right, up and down swipe gestures by activating the respective gesture flag. The Microchip MGC3130 and Hover devices can be seen, respectively, in Figure 3.5.(a) and in Figure 3.5.(b).

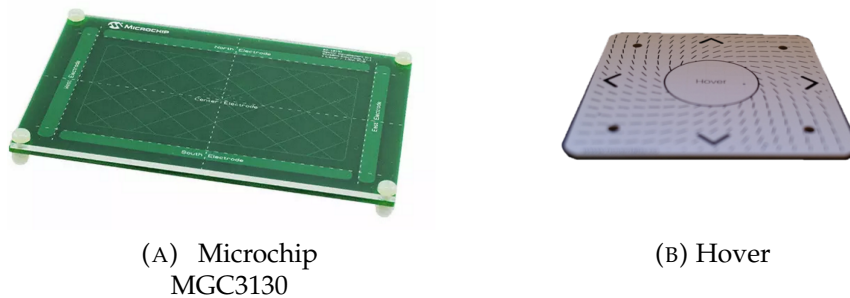


FIGURE 3.5: Microchip MGC3130 device on the left and Hover device on the right [55], [56]

3.4.3 Criteria to select the Development Board

The selection of the microcontroller unit is critical for the well system functioning since it interfaces with all the others subsystems. The microcontroller unit should take into consideration the following criteria:

- Cost;
- Performance;
- Size;

- Communication protocols (Ethernet, SPI, I2C, CAN and UART);
- Mbed development platform support;
- Availability.

3.4.4 Selection of the Development Board

The development board mbed-enable LPC4088 Quick Board form Embedded (Figure 3.6) was selected owing to a high clock speed, Arm Cortex M4F running up to 120 MHz; support for different standard protocols as UART, I2C, SPI and CAN. It also has low-power and is small witch facilitates the car integration.

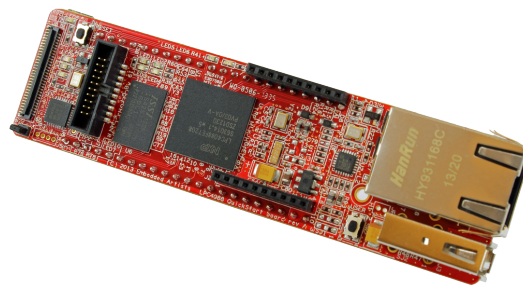


FIGURE 3.6: Embedded Artists - LPC4088 QuickStart Board [57]

Furthermore, a similar development board, the NXP LPC1768, was immediately available for development. The change to a new development board was owing to the fact that the EALPC4088 has an Ethernet Port (RJ45), which will be used to communicate the executed gesture.

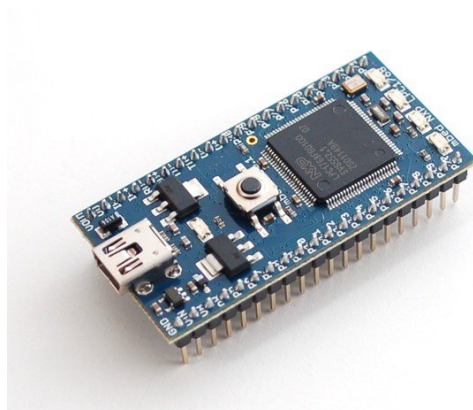


FIGURE 3.7: NXP LPC1768 Board [58]

3.5 Software Specification

For the development of the gesture algorithms, we study different developing tools. This study lead to a selection of the Mbed platform and the Keil embedded development tools for developing the algorithms, Qt for developing the GUI applications and Matlab for simulation purposes.

3.5.1 Programming Language

In this dissertation, the selected programming language was C and C++ because it is supported by most development platform and the target one and has a large and good community.

3.5.2 Software Platforms

Mbed

Mbed is a platform developed by ARM for devices based on 32-bit ARM Cortex-M, and it was selected to perform the interface between the sensors, development board as well as to implement the gesture recognition algorithms. This software allows great portability between different platforms and reduced time-to-prototype due to higher level of implementation.

Keil Embedded Development tools

Keil was selected to implement the gesture recognition algorithms as well as to do the interface between the sensors, development board. This software facilitates the configuration the development tools for the target microcontroller and shorten the learning curve. It is worth noting that this is not a replacement of MBED online editor are complements in this dissertation and not substitutes.

Qt

Qt framework was selected for the creation of GUI applications to facilitate the interface with the sensor allowing to visualize the signals in real time and the gesture data for analysis. This software seems to be a good selection since it has a variety of functionalities, good official support and code portability for different hardware targets.

Matlab R2015a and Kevin Murphy HMM toolbox

MATLAB (MATrix LABoratory) is a software towards numerical calculation. For the training, the Kevin Murphy Hidden Markov Model (HMM) Toolbox for Matlab was selected because it is widely used for training HMM, since it supports learning for HMMs with discrete outputs (dhmm's), Gaussian outputs (ghmm's), and mixtures of Gaussians output (mhmm's).

3.5.3 Algorithm Development

After literature review, two approaches were selected for algorithm development. The two approaches were: Threshold model and Hidden Markov Models. The first considered approach was a threshold-based one, since it is simple and allows a reliable detection of basic dynamic gestures. When following the machine learning approach, the HMM sounds the best option since dynamic gestures are time-varying processes, therefore HMM was a plausible choice to modeling them. Moreover, it is capable of modeling spatio-temporal time series where the same gesture can differ in shape and duration. HMM is widely used in hand writing, speech and character recognition [59].

3.6 The system

The overall overview of the system can be seen in Figure 3.8. The system is composed by three subsystem: Gesture Recognition Devices, HMI System and Monitoring Applications.

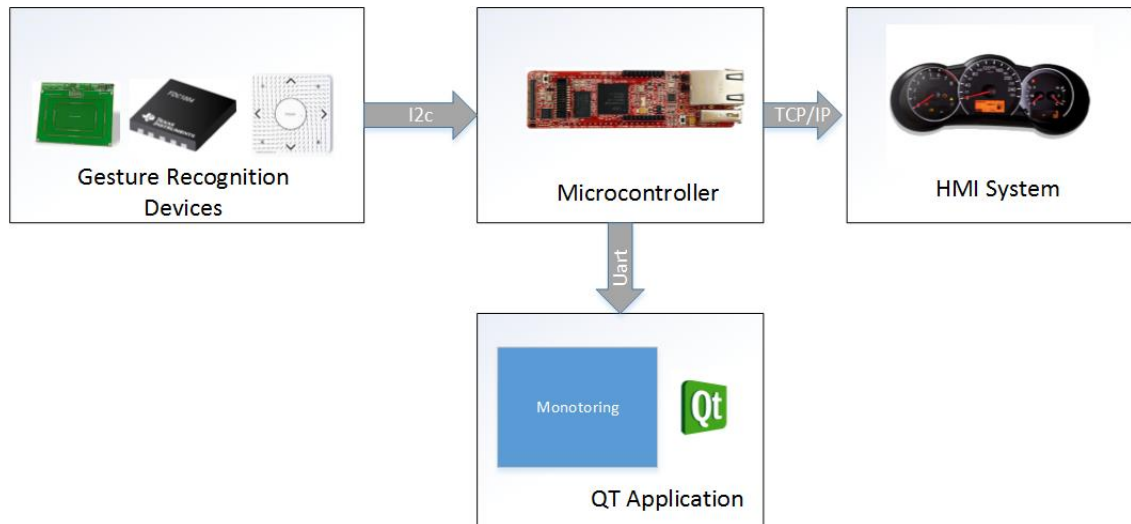


FIGURE 3.8: Detailed overview of the overall system

The gesture recognition devices interface with the microcontroller unit via I2C allowing it to acquire the raw data and perform the gesture recognition algorithms. After the gesture being recognized, a command is sent by TCP/IP to the HMI system. A GUI application, will be developed to test the devices as well as to monitor in real-time the data communication via serial interface, facilitating the sensor calibration and to have instant feedback on the recognized gesture. The three sensing devices have different specifications and characteristics, therefore this section will explain the interface of each device.

3.6.1 Texas Instruments FDC1004 Interface

The TI FDC1004 sensor allows four individual sensing channels; therefore, the created interactive device based on capacitive proximity sensing uses four individual electrodes. With the four individual electrodes it will be possible to recognize the right, left, up and down swipe gestures. The sensing device communicates with a microcontroller LPC4088 via I2C interface, then, the gesture recognition algorithms will be developed to detect gestures. To infer the hand movement it uses a threshold based model. Figure 3.9 depicts the system where each individual electrode's capacitance is measured and converted to digital by the FDC1004 sensor. The cross layout seems to be a good solution because it is intuitive, as it can be easily perceived how the user will have to perform a gesture.

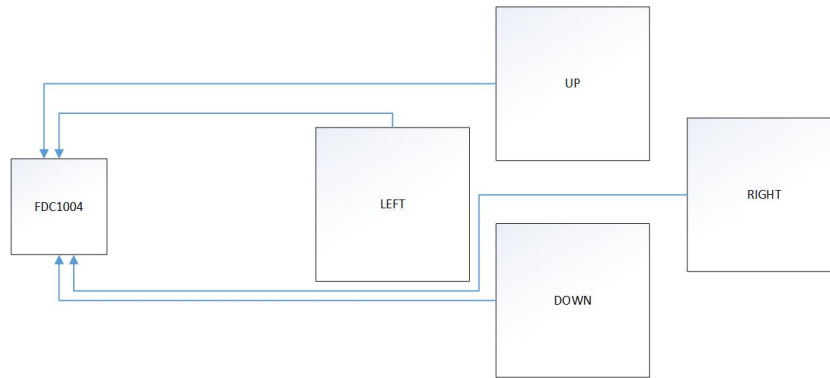


FIGURE 3.9: The interactive device - four electrodes distributed in a cross layout

The process of recognize the gestures can be described in several steps. Firstly, the sensor noise influence is minimized. Secondly, the algorithm is applied to recognize linear gestures. Lastly, the system is recalibrated to compensate for the sensor natural drift.

Raw Data Processing

The acquired signal will pass through an IIR Filter to remove the high frequency noise. This is necessary to smooth the data before sending to the gesture recognition algorithm. The IIR filter is similar to a moving average with the exception that the previous values do not have to be stored and shifted, becoming the process more computational efficient. The filtered value can calculated as follows:

$$Avg[i] = \frac{(Avg[i - 1]) * N - Avg[i - 1] + NewValue}{N}$$

Proximity Detection

When attempting to detect proximity, the environmental interferences must be considered since they affect the way whereby the electric field propagates. The environmental parameters such as temperature and humidity changes occur over a longer period of time, thus they can be compensated using a drift factor. Two approaches will be followed: differential and derivative. For both

approaches the calibration process involves calculating the maximum, minimum and average values of a set of measurements in order to detect the noise level and thus assigning a threshold value according to it.

For the first approach, the algorithm calculates the difference between the up and down and between the left and right electrodes and verifies if the difference is above or below a shift value, which is calculated as follows:

$$ShiftValue = AverageValue + \frac{MaximumValue - MinimumValue}{2}$$

If this is the case, the correspondent proximity flag is activated.

For the derivative approach, it tracks the rate of change between the current measurement and the previous measurement. When the derivative value, the rate of change, overcomes the derivative threshold value, an integrator begins accumulating the derivative values. If it achieves the integrator threshold, an object is considered to be in near proximity. It is worth noting that the derivative threshold should be very low; nonetheless, if it is unreasonably low the noise can be detected as proximity. Conversely, the integration threshold value should be high to avoid false triggering.

Algorithm

The gesture recognition algorithm begins by initializing the sensing device. If it is correctly initialized, the algorithm will continuously measure the four channel and determinate when an object is near. Then, when a proximity is detected, the corresponding proximity flag is activated and if it is the first proximity flag, the algorithm activates a begin gesture flag and initiates a timeout. After, if it detects another proximity flag in the same direction within the time gesture frame, a gesture is recognized. This means that when a proximity from the left electrode is detected, the algorithm waits to receive a proximity flag from the right electrode, disabling the proximity recognition from the up and down electrode. The same concept applies to the up and down swipe gesture. The gesture recognition algorithm can be seen in Figure 3.10.

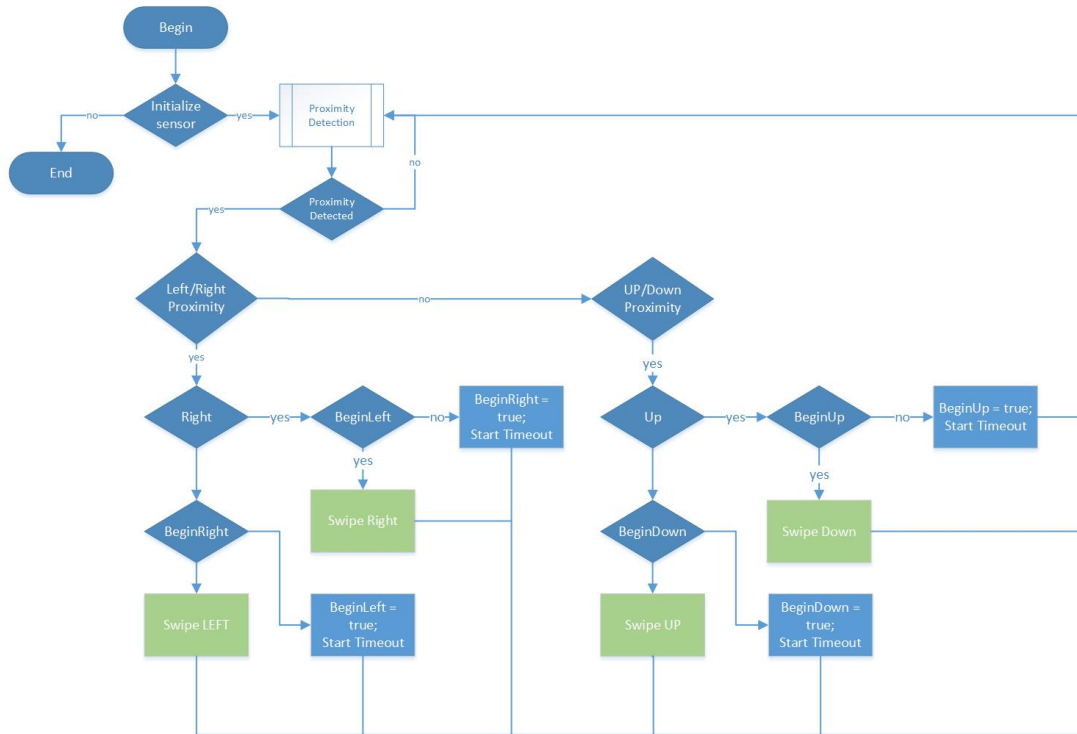


FIGURE 3.10: TI FDC1004 - Swipe Left, Swipe Right, Swipe Up, Swipe Down recognition flowchart

Recalibration

As specified previously, the sensor needs to be recalibrated to compensate for external interferences. This can be achieved by calculating a new threshold value after proximity is detected, as Figure 3.11 demonstrates.

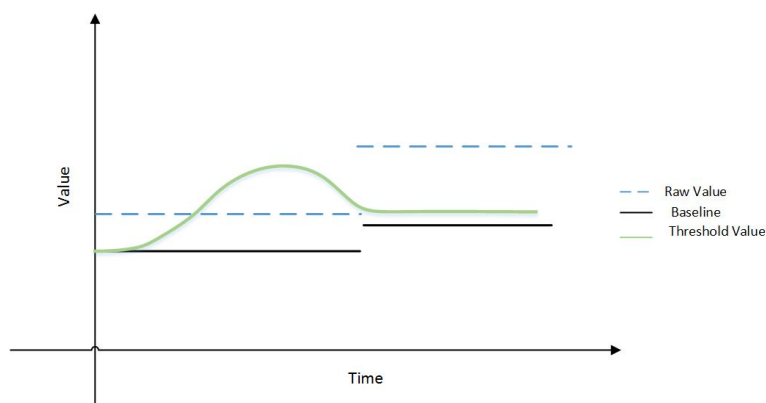


FIGURE 3.11: Adaptive baseline

The recalibration process, as previously explained, consist of collecting some capacitance values and obtain the maximum, minimum and average value. With those values, the new threshold value can be extrapolated.

3.6.2 Hover Interface

The gestures are already recognized by the sensing device, therefore the approach will be to extend the already recognizable gesture. Since it detects left, right, up and down swipe gestures, it will be extended to detect sequence gestures. The bidirectional gesture, where the user must perform a gesture, and shortly after another gesture in the opposite direction. As the flowchart (Figure 3.12) shows, when a gesture is detected for the first time, it is saved and a timer is activated. If within the time frame specified another gesture is recognized, it will be considered a sequence gesture, otherwise it will be considered a non-sequence gesture.

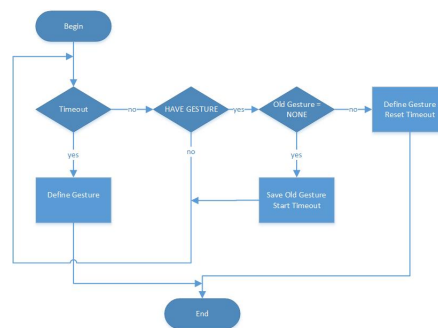


FIGURE 3.12: Hover extended gesture flowchart

3.6.3 Microchip MGC3130 Interface

This system is comprised of 3 independent modules: hand tracking, feature extraction and gesture recognition. The high-level architecture of the system is illustrated in Figure 3.13, where the interaction between the different modules is shown.

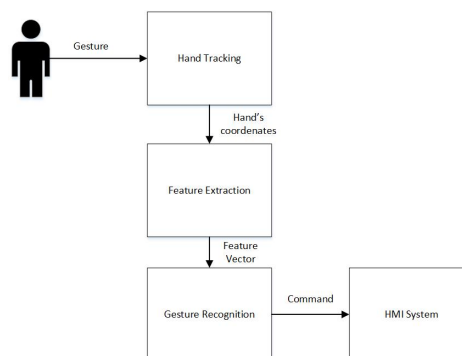


FIGURE 3.13: The system's high-level architecture

The interaction will occur once the human is within the recognizable sensing area of approximately 10 cm above the sensing device. As the user performs a gesture, the device will track the motion of the hand and pass it into the feature extraction module that will create the gesture's feature vector. Once the user finishes the gesture, the feature vector will be used to recognize the performed gesture and issue a command to the HMI system for it to perform the desired task.

Feature Extraction

The feature vector must encapsulate the trajectory of the hand's motion as the gesture is performed; therefore, the beginning and ending of a gesture must be determined. This problem is called segmentation, the capability of determining when a gesture ends and a new one begins. The feature vector will store the three-dimensional hand's coordinates when the first data is sent until there is enough data for the gesture to be recognized.

Gesture Recognition

This module is responsible for interpreting the executed gesture. The gesture is recognized by analyzing the feature vector. The principle of recognizing a gesture can be explained using a dx-dy plane (see Figure 3.14).

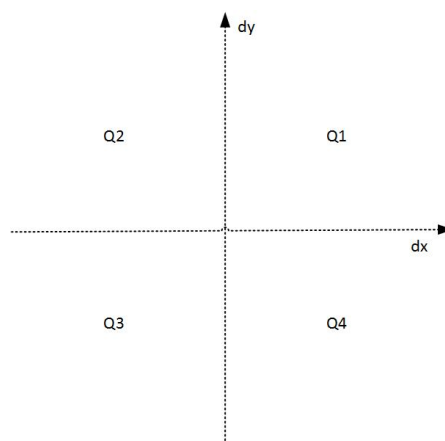


FIGURE 3.14: Quadrants in the dx-dy plane

Linear Gestures

For linear gestures (left, right, up, down, in and out swipes) the displacement occurs in one of four possible regions in dx - dy . For example, if the gesture to be recognized is a vertical motion, there are only two possible regions where it can be, near the dy axis. If the motion is directed upward, it will end near the positive side of the dy axis. On the contrary, if the gesture is directed downward, it will end near the negative side of the dy axis. Thereby, it is possible to recognize a gesture through its direction. Similarly, the left and right swipes as well as the in and out swipe gestures can be recognized.

Circular Gestures

Quadrant I is where both dx and dy are positive ($++$), quadrant II is where dx is negative and dy is positive ($-+$), quadrant III is where dx and dy are both negative, quadrant IV is where dx is positive and dy is negative ($+ -$). Therefore, analyzing the coordinate's displacements, the circular gestures can be determined. The quadrant in which the circular gesture begins is not relevant. The process of determining the direction of the circular motion involves scanning the signs of both dx and dy . After noting the signs, the repetitive sets are eliminated from the feature vectors. Then, the sequence is compared with a time-sequence model for the clockwise and another for anti-clockwise. The time-sequence model for the clockwise motion is $(-+, ++, +-, -)$ and for the anti-clockwise is $(++, +-, -, +)$. The classification of the motion will be determined by matching.

3.6.4 Graphical user interface

Two GUI applications will be developed, the first to visualize the capacitance values of the electrodes and another GUI application to visualize the hand's trajectories (x, y). Although the GUI applications have a similar objective of recognizing gestures, the different interfaces lead to two different applications.

3.6.5 GUI Application - CapGUI

This graphical application will allow to visualize, in real time, the capacitance value of each electrode. Thereby, the interface with the device sensor can be tested and the threshold values can be defined. Moreover, the executed gesture is displayed for immediate feedback on the recognized gesture. Figure 3.15 shows the GUI layout and his functionalities.

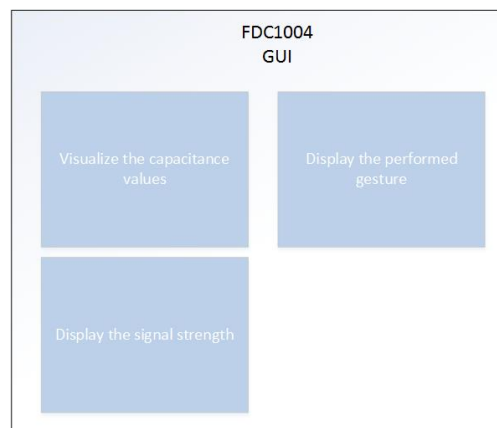


FIGURE 3.15: CapGUI layout

Main Block

The GUI application consists of four routines: Read Data, Graphic window, Gesture window and Strength window (Figure 3.16). Read Data handles the reception of data with the specified protocol, the Graphic window deals with the visualization of the capacitance values in real time, the Gesture window displays the recognized gesture, and strength windows shows the signal strength. Each routine is explained in the following sections.

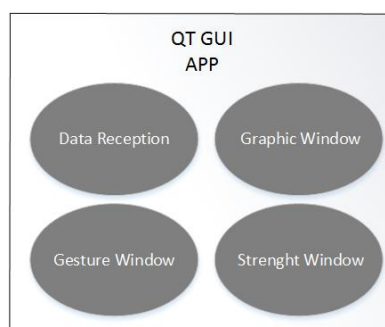


FIGURE 3.16: Graphical User Interface main routines: Data reception, Graphic window, Gesture window and Strength window

Read Data

In this applications there are two types of messages. The first is the update of the graphic values, where the microcontroller sends the four measurements of the four electrodes (Left, Right, Up and Down). The second is a message that occurs when a gesture is recognized, the corresponded gesture is sent to be displayed in the GUI. The communication protocol has the following message:

- $\langle *value \backslash n \rangle$ - channel one measurement;
- $\langle \#value \backslash n \rangle$ - channel two measurement;
- $\langle !value \backslash n \rangle$ - channel three measurement;
- $\langle +value \backslash n \rangle$ - channel four measurement;
- $\langle ?gest \backslash n \rangle$ - recognizable gesture.

where, value is a 23 bit data of each electrode measurement, gest corresponds to the gesture recognized.

When a message is received it will be decoded by its first character. When it receives one of the following first characters: '*', '#', '!', '+', the capacitance values will be updated. On the other hand, if the first characters is '?', the gesture will be used to update the Gesture window.

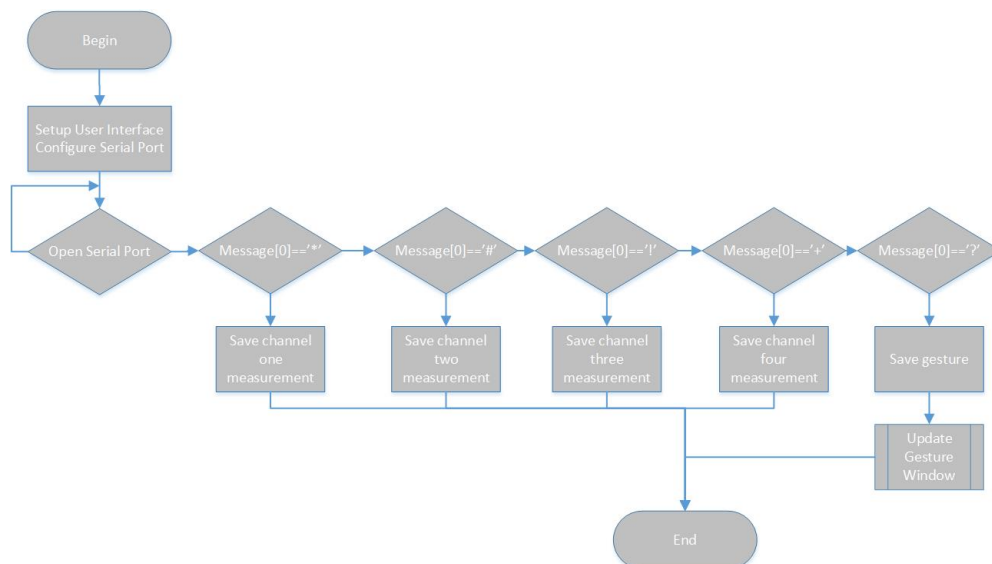


FIGURE 3.17: Data Reception Flowchart

Graphic Window

Graphic window will plot four graph line, in real-time, each one corresponding to the capacitance value of the electrodes. This routine has two main components: graphic setup and graphic update. The former will configure the number of graphics as well as their colors. It will also set the axis labels and make the range adjustable while new values are added. Finally, it will set a timer to call the graphic update routine. The latter, will update the capacitance values and redraw the Graphic window. Figure 3.18 shows the process of configuring and updating the graphic.

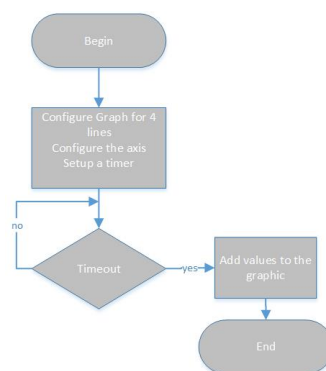


FIGURE 3.18: Update Graph Flowchart

Gesture Window

This routine aims to display the recognized gesture for immediate feedback. When the recognized gesture is sent by the microcontroller unit and received by graphical application, the corresponded gesture picture is displayed and a timer is initiated. As the flowchart 3.19 demonstrates, when a timeout occurs the default image is set.

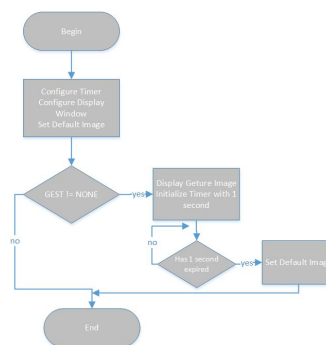


FIGURE 3.19: Gesture Window Flowchart

Strength Window

This routine is responsible for displaying the signal strength of the four electrodes. If the user's hand is in near proximity to one of the electrodes, the correspondent electrode signal increases drastically and the other electrodes could also slightly increase if they are near that electrode. While the microcontroller unit sends the capacitance values of the four electrodes, these values will be displayed in a form of a bar graph.

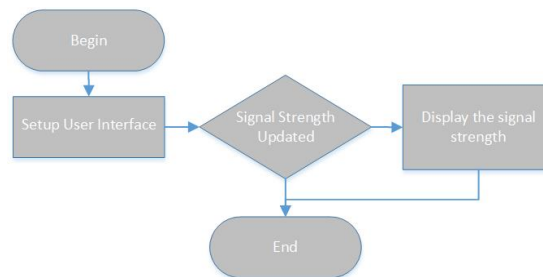


FIGURE 3.20: Strength Window Flowchart

3.6.6 GUI Application - CoorGUI

Another graphical user interface will be developed to visualize the hand's trajectory in real time. Thereby, the algorithms can be performed based on the hand's motion. In a similar way the recognized gesture by the microcontroller unit is displayed for immediate feedback. Figure 3.22 shows the GUI layout and his functionalities.

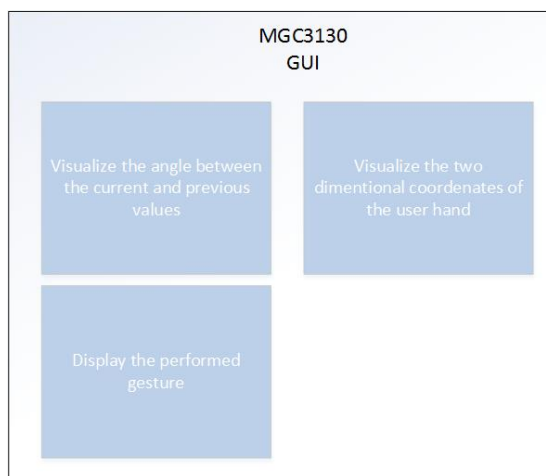


FIGURE 3.21: CoorGUI Layout

Main Block

The GUI applications consists of four subsystems: Read Data, Arrow window, Gesture window and 2D Track window (Figure 3.22). Read Data handles the reception of data with the specified protocol. The second routine consists of a two dimensional grid plane with a point above moving according to the device coordinates sent. The third is used to display the recognized gestures. The last routine displays an arrow, its angle corresponds to the angle between the two last coordinates. The flowcharts of each subsystem can be seen in the next sections.

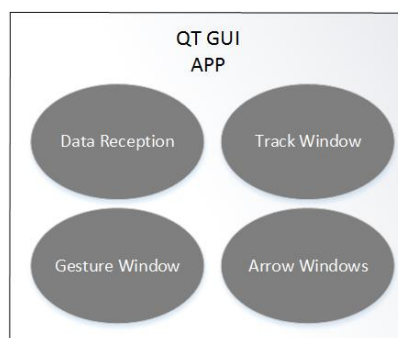


FIGURE 3.22: Graphical User Interface main routines: Data reception, Track window, Gesture window and Arrow window

Read Data

This applications contains two types of messages. The first consists of the coordinates(x,y) and the angle between the current point and the last point. The second message occurs when a gesture is performed. The communication protocol has the following message:

- $\langle Mangle, posx, posy \setminus n \rangle$ - the angle value, x position and y positions;
- $\langle Ggest \setminus n \rangle$ - recognizable gesture, where gest is the word to differentiate the gestures;

When a message is received it will be decoded by its first character. When it receives one of the following first characters: 'M', the capacitance values will be updated. On the other hand, if the first characters is 'G', the gesture will be used to update the Gesture window. Figure 3.25 illustrates the flowchart of the data reception.

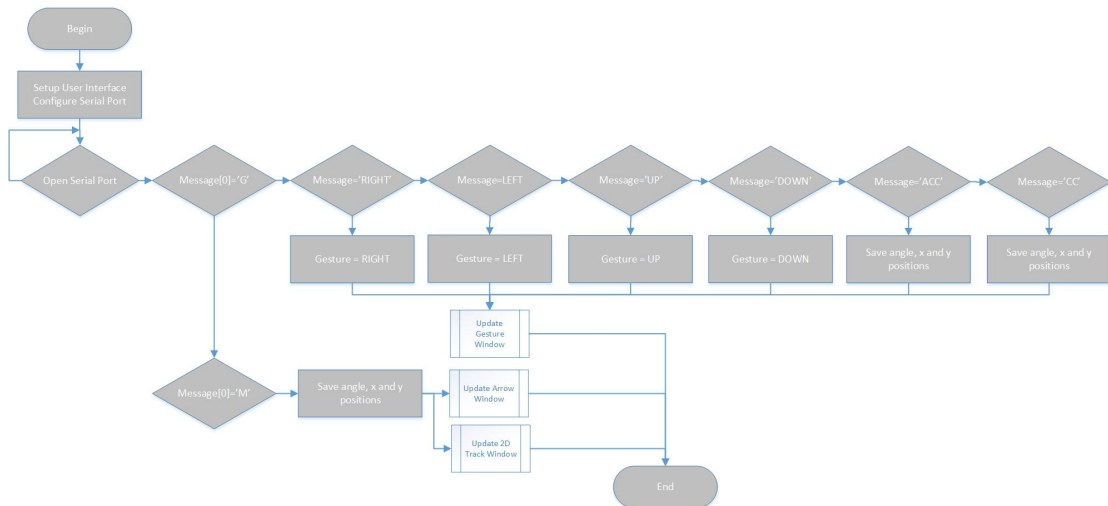


FIGURE 3.23: Data Reception Flowchart

Arrow Window

The next routine to be implemented was the Arrow window. The idea was to develop a concept where the performed number of circles could be determined. Instead of detecting a clock wise or anti-clockwise gesture, where the gesture is only detected when the gesture is completed, half-gestures or two circles can be detected. After the angle being issued by the microcontroller unit, this routine will have to rotate the image and display the quantity of executed circles. In the Qt coordinate system the y axis grows upwards, therefore the rotation is with the opposite number. Two translation are needed, the first is for the arrow rotate around its center and the second to move back the arrow to its original position. Figure 3.24 shows this routine flowchart.

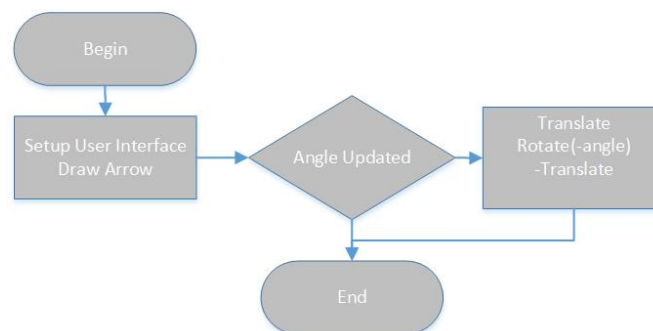


FIGURE 3.24: Arrow Display Flowchart

2D Track Window

This routine starts by creating a two dimensional grid plane, allowing a more precise visualization of the hand position relative to the sensing device. When an object is near the device, i.e. user hand, it starts sending the positional data (x and y coordinates), that will be used to update a point that is in front of the grid plane.

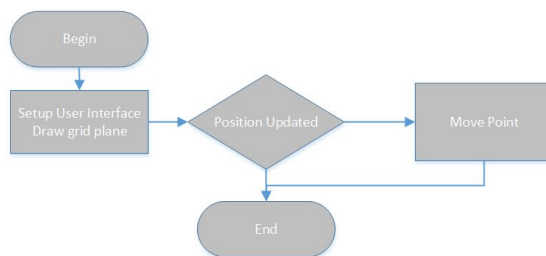


FIGURE 3.25: 2D Tracking Flowchart

Gesture Window

The gesture window routine is similar to the other graphical user application, where it displays a gesture for a second, after the gesture being recognized by the sensing device and received by the GUI.

3.6.7 Cluster

Once the gestures are recognized, the microcontroller issue a TCP/IP command to HMI system for it to perform the desired task (Figure 3.26). The HMI system is essentially a local server that accepts requests from different gesture recognition devices.

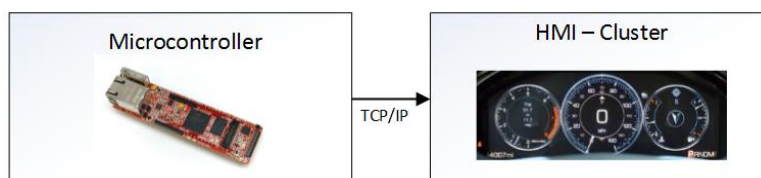


FIGURE 3.26: Server Subsystem

Communication Protocol

The protocol can be seen as a two steps procedure: registration, where the device is registered in the HMI System; gesture transmission, where the HMI system verifies if the device was already declared and if so a gesture can be sent. For registration, the data message, witch specifies the sensor devices ID (identification), is:

– "registernode = Gesture_Sensor_x\n"

Sending a gesture can be accomplished by: define a Destination ID, for which part of the HMI System is the destination; define a Gesture Device ID, which must be already registered; define a message, where the recognized gesture is specified.

– "to:DestinationID from:Gesture_Sensor_x message=<Gesture_Y>\n"

How it is possible to verify in the sequential diagram (Figure 3.27), the algorithm needs first to register the device sensor ID. After the successful registration, the algorithm will send the gestures, when those happens, to the HMI System.

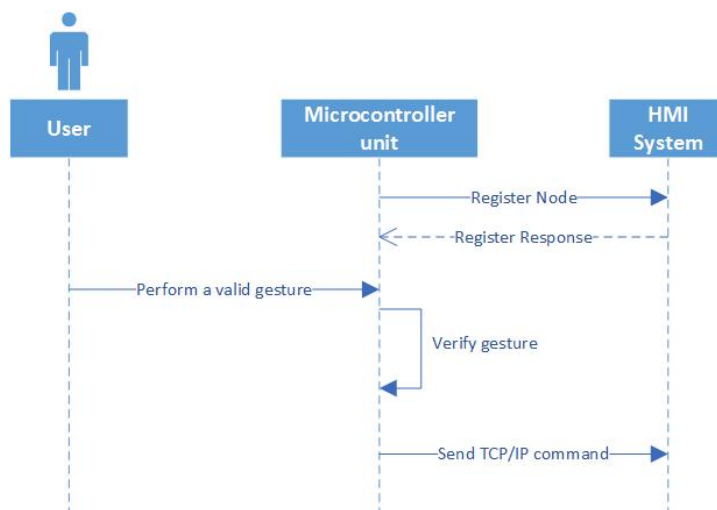


FIGURE 3.27: Sequence diagram to send the gesture command to the HMI System

Client

All devices must communicate with the HMI system, thereby, all the algorithms begin in the same way. It starts by establishing an Ethernet connection to microcontroller IP address. After being connected to the network, it attempts to open a TCP/IP socket to communicate with the HMI system. After the TCP/IP communication setup, a TCP/IP message is sent when a gesture is performed by the user. Figure 3.28

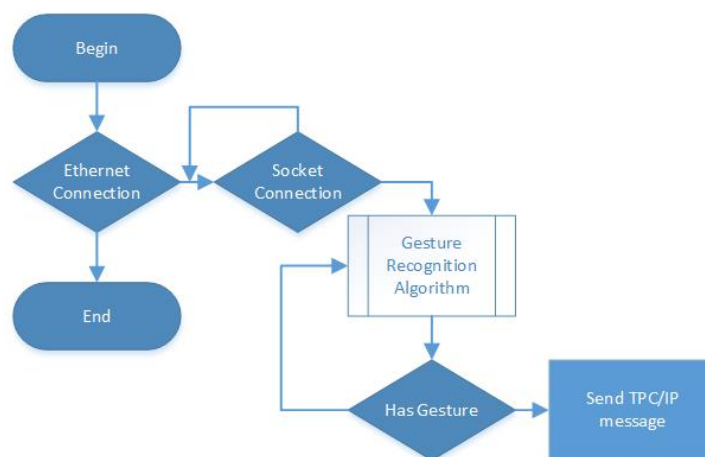


FIGURE 3.28: Sending a gesture via TCP/IP Flowchart

Server

In order to test the integration between the microcontroller unit and the HMI system a local server will be developed. This server has only the purpose of validating the data reception with the previously explained data structure that will be used to communicate with the cluster in DSM. Therefore, the server will not perform any functionality, it will only simulate the cluster.

3.7 HMI Concepts

This section summarize the recognizable gestures in an overall perspective. Every device sensor recognize the linear gestures (left, right, up and down swipes) and some the sequence gestures (left-right, right-left, up-down and down-up swipes). Additionally, the linear gesture (in and out swipes) and the

rotational gestures (clockwise and anti-clockwise gestures) can be recognized by the Microchip MGC3130.

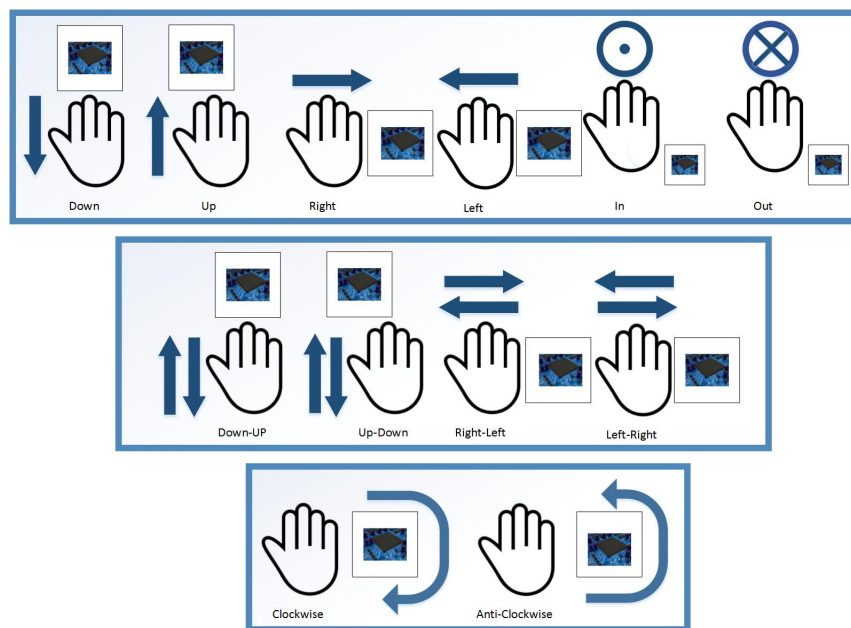


FIGURE 3.29: Supported Gestures

The system has one actor (the driver or the passenger), where the actions are the different hand gestures that can be used to control different in-vehicle infotainment functionalities (Figure 3.30).

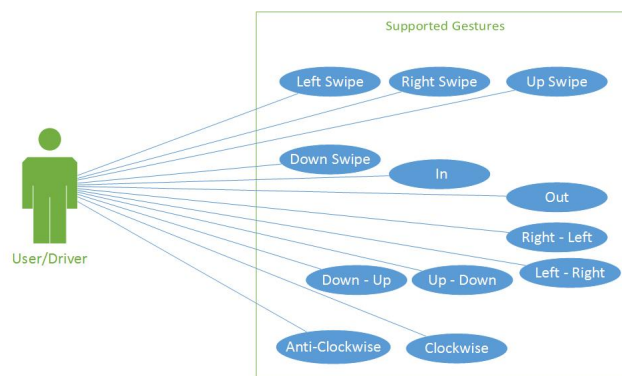


FIGURE 3.30: System Overview

These functionalities can be multimedia controls, navigation controls, steering wheel controls, phone and warning interaction, HVAC controls, general HMI controls, control of roof, hood, windows and doors, among others.

3.8 Data Processing

Figure 3.31 shows a typical simplified processing pipeline for capacitive proximity sensing. It begins by converting the capacitance to a digital value that will be processed to minimize external influence. Then, the new signal can be used for several applications.

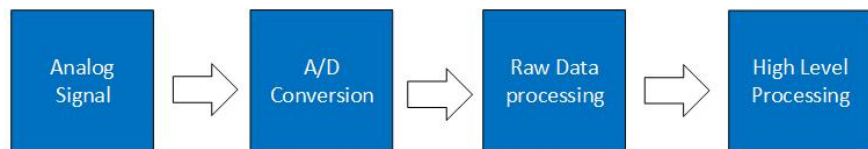


FIGURE 3.31: Typical data capacitive sensing pipeline

Raw Data Processing

This phase attempts to reduce or compensate sensor noise that can influence the change of capacitance, such as temperature, humidity, composition of the air or grounded objects in close proximity.

The most common form of noise in capacitive proximity is high-frequency signals, therefore it is necessary to reduce this influence by using a low-pass filter. The most commonly used low-pass filters are average and medium filters.

Since the parameters change over time due to electronic components heating up, the environmental temperature changing or humidity changing, it is essential to have an adaptive baseline. One simple method is to have a threshold level that triggers the baseline calibration when the object (user hand) is removed.

High Level Processing

After the normalization and calibration of the sensor, the purpose of any capacitive application is to use the raw data to perform a task. For gesture recognition purposes, two methods were selected: Threshold and HMM.

Threshold Model

Threshold model is a good solution for simple dynamic gesture recognition because of its simplicity and intuitiveness. For each gesture a set of threshold values must be defined in order to the correct gesture classification. Typically, for simple dynamic gesture recognition (horizontal and vertical swipes), two methods are used: position-based and phase-based.

The former is related to the estimation of the position to recognize the gesture. This means that the raw data must first be converted to distance data and thus determine the position of the target object. Thereafter, the algorithm verifies the movement of the position for a certain period of time. If the position moves steadily during the defined time frame, a swipe gesture has occurred.

On the contrary, for the latter method, the location of the target is never calculated. This method involves solely analyzing the raw data from the proximity measurement. The swipe gesture can be determined by the verifying which proximity flag raised first.

HMM Model

As shown in Figure 3.32, in the HMM approach, each gesture has his own HMM model.

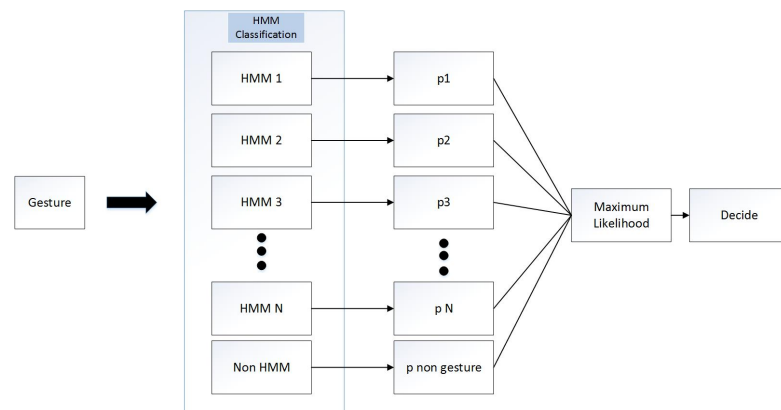


FIGURE 3.32: Evaluate the gesture recognition probability of each model

When the user performs a gesture, the sequential data acts as the input to the HMM classifier. Then, each model will give a likelihood, with the higher

being the recognized gesture. Since the model will always return the highest gesture, a simple threshold for the likelihood or the creation of HMM model for gestures that should not be considered is needed.

Learning Procedure

HMM is considered supervised learning, therefore a training set must be given that will allow the discovery of the parameters of each HMM model, thereby constructing the gesture database. To obtain the training database, a few people need to perform the same gesture a few times on the gesture recognition device. In order for the robustness of the classifier, the performed gestures must be variously performed with different hand positions, levels of distance and speed. Then, the training data will be saved in a SD card in a comma-separated values (CSV) file. This file will be imported by Matlab to train the HMM models. For the training, the Kevin Murphy Hidden Markov Model (HMM) Toolbox for Matlab will be selected because it is widely used for training HMM, since it supports learning for HMMs with discrete outputs (dhmm's), Gaussian outputs (ghmm's), and mixtures of Gaussians output (mhmm's). Lastly, the HMM model for each gesture is obtained.

Algorithm Validation Metrics

The algorithms will be evaluated by its accuracy, sensitivity and specificity. These attributes can be obtained by a confusion matrix (or confusion table), being a detailed representation of correct and incorrect classification for each class. As Figure 3.33 shows for each class, it describes the TP (number of true positives), FP (number of False positives), TN (number of true negatives) and FN (Number of False Negatives).

		Prediction	
		0	1
Actual	0	TN	FP
	1	FN	TP

FIGURE 3.33: Confusion matrix Concept[60]

The accuracy for each class can be obtained by:

$$Accuracy = \frac{TP + TN}{n} \quad (3.1)$$

The sensitivity to evaluate the likelihood of correctly labeling members of the target class is determined by:

$$Sensitivity = \frac{TP}{TP + FN} \quad (3.2)$$

The specificity to evaluate how well correctly identifies the negative cases:

$$specificity = \frac{TN}{TN + FP} \quad (3.3)$$

3.9 System Validation

Tests play a crucial component in the development of a prototype, as they ensure the validation of the implemented functionalities. These tests will be classified as initial tests, integration tests and final tests. The initial tests are responsible for dealing with the devices communication, allowing the raw data to be read by the microcontroller unit.

3.9.1 Final Tests

The final system will be evaluated by 15 to 25 participants. The selection of participants shall be diversified, to simulate the demands of the real world, therefore the users that will be interacting with the sensing devices should have different ages and sexes. Each individual will be required to execute the same gesture (30 times), ideally with different hand positions and speeds. This evaluation will take place in a laboratory-like environment, this means that the conditions will be controlled, such as temperature, humidity and others distracting factors.

Chapter 4

Implementation and Results

The previous chapter allowed to understand the system architecture, their subsystems and functionalities. It was specified the algorithms for gesture recognition, described the structure that it is used for the implementation of the graphical user interfaces and the HMI system as well as the process of communication between the different subsystems.

This chapter describes how the different modules were implemented. More specifically, what was possible to fulfill in relation to the planned. It begins with the required hardware for development, then it presents the implement algorithms. Thereafter, it will address the graphical user interfaces and the HMI system. Finally, it presents the results of the implementation of the gesture recognition algorithms and the interaction with both the monitoring graphical applications and the HMI system.

4.1 The system

The overall overview of the system can be seen in Figure 4.1. The system is composed by four subsystem: the Gesture Recognition Devices, the Microcontroller Unit, the Cluster and the Monitoring Applications.

The same presentation used in last chapter used in last chapter will be used. Therefore, the implementation topics will approach first the sensing devices, next GUI applications and finally the GUI applications.

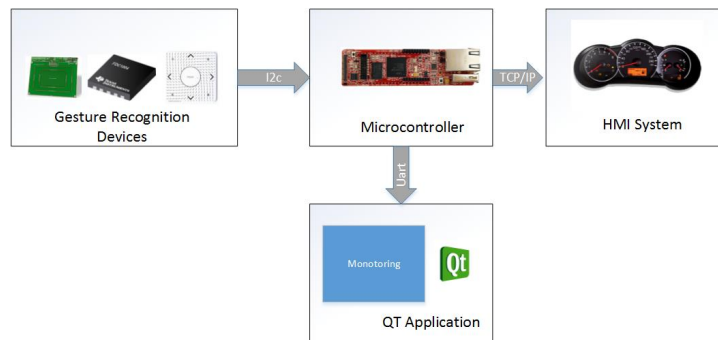


FIGURE 4.1: Implementation View

4.1.1 Sensing Devices

In the implementation phase the developed algorithms were implemented for each gesture recognition device. The following section present how the different algorithms were implemented for the TI FDC1004, Hover and Microchip MGC3130 sensing devices.

Hardware

The majority of the necessary hardware for the development of this research was bought as a COTS (Commercial off-the-shelf) solution with no alteration. Thus, the only hardware implementation was the electrode design of the TI FDC1004 sensor.

Electrode Design

When designing capacitive sensing applications the two major factors are electrodes geometry and material, as previously described. To evaluate the required trade-off between sensing range and electrode size an experiment was conducted. This experiment used squared electrodes, to determinate how the electrode size affect the sensitivity. The electrodes were made of copper since it has the best proprieties for capacitive sensing and it was easy to obtain.

The following electrodes sizes : 2 cm^2 , 3 cm^2 , 4 cm^2 , 5 cm^2 , 6 cm^2 and 7 cm^2 (Figure 4.2.(a)) were tested to detect the maximum detection. The experimental

test setup, presented in Figure 4.2.(b), consist of acrylic glass with written intervals to determinate proximity distance. The measurement electrode and the sensor are connected via a coaxial cable to minimize any external interference.

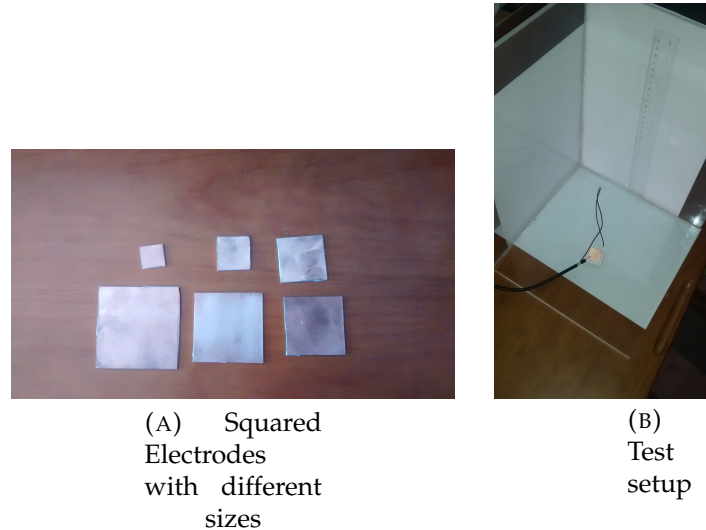


FIGURE 4.2: Experimental setup

The results obtained, as expected, demonstrated the proportionality between the size of the electrodes and the maximum distance detection. This test allowed to conclude that a 5 cm^2 electrode size is suitable for the desired application. The results are presented in the table below:

TABLE 4.1: Correlation between the size of the electrodes and the maximum distance detection

Electrode size (cm^2)	Maximum Distance (cm)
1	2
2	4
3	6
4	8
5	10
6	12
7	16

4.1.2 Texas Instruments FDC1004

The device usage consists of (1) configuring the measurements, (2) trigger a measurement, (3) wait for measurement completion and (4) read measurement data. Firstly, the measurements are configured to enable the desired input

channels to be read, to simplify each measurement is equal to the corresponded input channel; the capdac value, which is an offset to the measurements, is defined as zero because it is only used when the value is over 15 pF. After the measurement being triggered, the reading method verify if the measurement is completed, if so the data is valid to be used.

```
int main() {
    sensor = new FDC1004(FDC1004_100HZ);
    sensor->configureMeasurementSingle(0, 0); //Enable channel 0 with a capdac value of 0
    ...
    sensor->measureChannel(0, 0, raw_value); //Measure the 23-bit data measurement
    int32_t left_capacitance = raw_value[0] << 16 | raw_value[1];
    left_capacitance /= 256;
    ...
}

uint8_t FDC1004::measureChannel(uint8_t channel, uint8_t capdac, uint16_t * value) {
    uint8_t measurement = channel;
    triggerSingleMeasurement(measurement, this->_rate);
    wait_ms(SAMPLE_DELAY[this->_rate]);
    return readMeasurement(measurement, value);
}
```

LISTING 4.1: Measuring the capacitance value using the TI FDC1004

Filtering

The implementation of the low pass filter consists of adding the current value, the 23 bits measurement of one electrode, to the average multiplied by N, then subtracting the average and then dividing all by N. The N is 16, which is the size of the window, is a power of two to facilitate the division operation.

```
uint32_t Moving_Average::Average(uint32_t value)
{
    uint32_t avg =0;
    avg = this->old_avg * N - this->old_avg + value;
    this->old_avg = avg;
    return avg >> this->divide_value;
}
```

LISTING 4.2: Moving average code

The acquired signal, shown in 4.4.(a), passed through an IIR filter to remove the high frequency noise. A Matlab script was implemented to analyze how different window sizes affect the raw signal. The raw signal is captured by the sensing device and exported to Matlab. After considering the problem

in detail, the number of 16 points seemed to be a good solution as is possible to see in the Figure 4.4.(b).

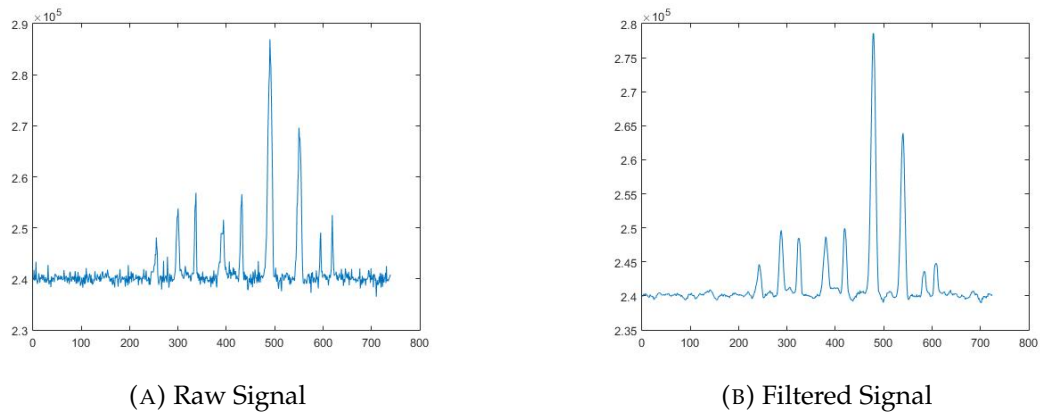


FIGURE 4.3: Raw capacitance value acquired using the FDC1004 sensor and the same signal passing through a IIR filter of 16 samples

Calibration

The calibration process consist of collection 50 sample points to obtain the maximum, minimum and average values. This procedure allows to verify the noise level present in the system and thus permits to define the proximity threshold values according to it.

```
threshold_value = (stats->max_value + stats->min_value) / 2 + stats->min_value;
```

LISTING 4.3: Calibration Code

Proximity

The device read the measurement of the four channels, and then calculates the difference between the up and down electrodes measurements and between the left and right electrodes. On the basis of that difference, it is possible to detect proximity. If it has three consecutive proximity values of the same electrode, it is considered valid. If it exceeds the maximum number of proximity detections, it means that, possibly, an external source is interfering with the system, thereby, the threshold values must be recalibrated.

```

diff_value_lr = left_avg - right_avg;
if( diff_value_lr < (baseline_value_diff_lr - Thres_lr) ) {
    if(counter_right >= 3) right_prox = true;
    if(counter_right >= 30) right_prox = false;
    counter_right ++;
}else if(diff_value_lr > baseline_value_diff_lr + Thres_lr){
    if(counter_left >= 3) left_prox = true;
    if(counter_left >= 30) left_prox = false;
    counter_left ++;
}else{
    counter_left = 1;
    counter_right = 1;
    right_prox = false;
    left_prox = false;
}

```

LISTING 4.4: Proximity detection code

Proximity - Derivative approach

The derivative approach calculates the rate of change between the current and previous measurements. After the signal is acquired, it passes through a low pass filter and when the rate of change is above a threshold value of 100 picofarad, an integrator begins to accumulate those values. If the integrator value overcame the integrator threshold, it means that proximity was detected. Figure 4.4 illustrates the original signal and the correspondent integrations value.

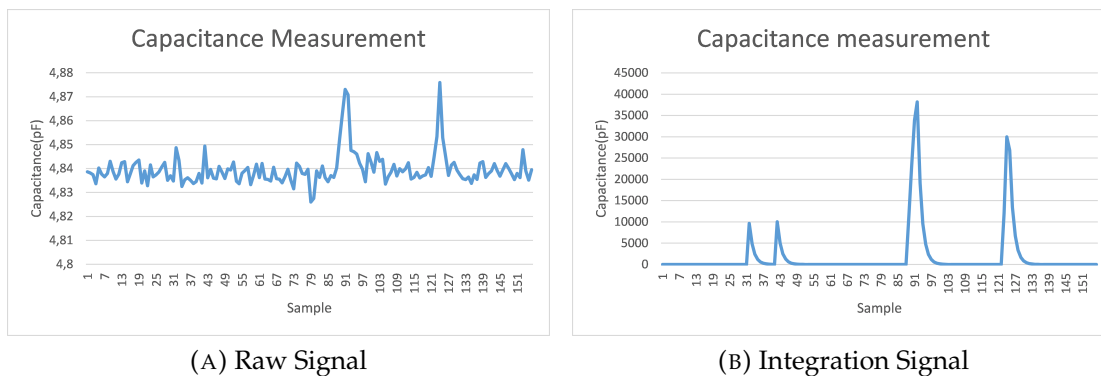


FIGURE 4.4: Raw capacitance value acquired using the FDC1004 sensor and the integration algorithm for proximity detection

Gesture Recognition

The gesture recognition algorithm uses the proximity data to recognize gestures. When a proximity flag is raised, the algorithm waits 700 ms until the opposite flag is raised. If it succeeds, the gesture is recognized and the respective command is sent to the system responsible for handling it. The extract of code below shows how the left and right swipes are recognized.

```
if(left_prox == true || right_prox == true ){
  if( right_prox == true){
    if(begin_left){
      sock.send_all("to:DashboardUpperStack from:FDC1004 message=Right",49);
      ...
    }else{
      if(begin_right == false) end_time.attach(&gest_time, GESTURE_DURATION);
      begin_right = true;
    }
  }else{
    if(begin_right){
      sock.send_all("to:DashboardUpperStack from:FDC1004 message=Left",48);
      ...
    }else{
      if(begin_left == false) end_time.attach(&gest_time, GESTURE_DURATION);
      begin_left = true;
    }
  }
  ...
}else{
  if(has_timeout){
    ...
    has_timeout = false; begin_left = false; begin_right = false;
  }
}
```

LISTING 4.5: Left and right swipe recognition using the FDC1004EVM

4.1.3 Hover Gesture Device

The device sensor provides the positional data of the user's hand, nonetheless it was not fast enough to be used for gesture recognition. Therefore, the strategy was to recognize 4 additional gestures (Left-Right, Right-Left, Up-Down, Down-Up) in addition to the already recognized left, right, up and down swipe gestures. The main routine begins by initializing the sensor and then verifies if a gesture flag is raised. When it does, if another gesture is recognized before timeout, it will be considered a sequential gesture.

```
int main(void) {
GESTURE::GESTURE gesture = GESTURE::DIRECTION_NONE;
if(hover.Initialize())
pc.printf("Hover Initialization Completed!\n\r");
else
pc.printf("Hover Initialization Problem!\n\r");

while(1) {
if(hover.IsGestureAvailable()){
gesture = hover.ReadGesture();
DecodeGesture(gesture);
}
Sequentialdecode();
wait_ms(35);
}
```

LISTING 4.6: Left and right swipe recognition using the Hover device

4.1.4 Microchip MGC3130

Hand Tracking

The MGC3130 device provides the hand's coordinates in three-dimensions when the user hand is within the recognizable sensing area. The electrode layout consists of one transmitting electrode at middle and the receivers are placed on the edges. Thereby, calculating the variation in the signals at the receiver electrodes when the user absorbs some of the radiated field, it is possible to determine where the user's hand is located. The coordinates (x,y,z) goes from $(0,0,0)$, on the bottom left corner, until $(65535, 65535, 65535)$, on the upper right corner.

Feature Extraction

From the initial moment that the device sends the first positional, the feature vector is created storing the hand's trajectory until there is enough information to be processed.

Gesture Recognition

For the linear gesture it detects the direction of the gesture by calculating the difference between the first and last values. If this value is above a threshold value, the swipe gesture can be determined. This type of gestures include left, right, up and down swipes. For the circular gestures, the first step is to determine the respective quadrants in which the gesture was performed and eliminate the repetitive quadrants. Then, by matching with the clockwise motion (-+,++,+,-) or anti-clockwise motion (++,+,-,-) the gesture can be recognized.

4.2 GUI Applications

4.2.1 CoorGUI

Read Data

When the application receives the data message from the microcontroller unit, sometimes all the content was not received, thereby it was necessary to accumulate that short messages until the new line character is received. Only then the message can be processed to change the visual interface. This process involves decoding the message to update the two dimensional hand's position, angle and gestures.

```
QStringList buffer_split = serialBuffer.split('\n');
    if(buffer_split.length() < 2){
        this->serialData = this->serial->readAll();
        this->serialBuffer = this->serialBuffer +
            QString::fromStdString(serialData.toStdString());
        serialData.clear();
        if(this->serialBuffer.contains('\n')) this->DataUpdate();
    }else{
        this->DataUpdate();
    }
}
```

LISTING 4.7: Data reception handler

Arrow Window

The next routine to be implemented was the Arrow window. It used QPaint to create the arrow shape, as addition of a rectangle with a triangle. In the Qt coordinate system the y axis grows upwards. Therefore, the rotation is with the opposite number. Two translation are needed, the first is for the arrow rotate around its center and the second to move back the arrow to its original position.

```

QPainterPath OuterPath;
OuterPath.setFillRule(Qt::WindingFill);
OuterPath.addRect(100, 200, 100, 20);
QPolygon polygon;
polygon << QPoint(200, 210) << QPoint(200, 240)
        << QPoint(240, 210) << QPoint(200, 180)
        << QPoint(200, 210);
OuterPath.addPolygon(polygon);
...
QPainterPath FillPath = OuterPath;
QPainter Painter(this);
Painter.translate(150,210);
Painter.rotate(-this->angle);
Painter.translate(-150,-210);
...
}

```

LISTING 4.8: Creation of an arrow and updating its rotation

2D Track Window

To facilitate the visualization of the user's hand trajectory, a two-dimensional grid plane was created using the QGraphicsScene object. The hand's coordinates are normalized to fit into the plane and represented by a black point

```

QGraphicsScene* scene = new QGraphicsScene;
for (int x=0; x<=250; x+=50)
    scene->addLine(x,0,x,250, QPen(Qt::black));
for (int y=0; y<=250; y+=50)
    scene->addLine(0,y,250,y, QPen(Qt::black));
...
QBrush blueBrush(Qt::blue);
QPen outlinePen(Qt::black);
outlinePen.setWidth(2);
scene->addEllipse(this->posx*250/65536, (65536 - this->posy)
                *250/65536,5,5,outlinePen, blueBrush);

```

LISTING 4.9: Creation of an two dimensional grid plane

Gesture Window

This routine aims to display the recognized gesture for immediate feedback. When the recognized gesture is received by the GUI, it is displayed. The gestures include the liner gestures left, right, up, down, in and out swipes as well as the rotation gesture clockwise and anti-clockwise. After the gesture is displayed a timer is initiated, and when it finishes the image is reset to his default image.

```
void MainWindow::selectGestImage(swipe_gesture g) {
int w = ui->labelGestImage->width(); // get label dimensions
int h = ui->labelGestImage->height();
QPixmap pix("./Pictures/Arrow Left.jpg");
QPixmap pix2("./Pictures/Arrow Right.jpg");
QPixmap pix3("./Pictures/Arrow Up.jpg");
QPixmap pix4("./Pictures/Arrow Down.jpg");
QPixmap pix5("./Pictures/In.jpg");
QPixmap pix6("./Pictures/Out.jpg");
QPixmap pix7("./Pictures/Arrow CC.jpg");
QPixmap pix8("./Pictures/Arrow ACC.jpg");
switch(g) {
case LEFT:
ui->labelGestImage->setPixmap(pix.scaled(w,h,Qt::KeepAspectRatio));
break;
case RIGHT:
ui->labelGestImage->setPixmap(pix2.scaled(w,h,Qt::KeepAspectRatio));
break;
case UP:
ui->labelGestImage->setPixmap(pix3.scaled(w,h,Qt::KeepAspectRatio));
break;
case DOWN:
ui->labelGestImage->setPixmap(pix4.scaled(w,h,Qt::KeepAspectRatio));
break;
case IN:
ui->labelGestImage->setPixmap(pix5.scaled(w,h,Qt::KeepAspectRatio));
break;
case OUT:
ui->labelGestImage->setPixmap(pix6.scaled(w,h,Qt::KeepAspectRatio));
break;
case CC:
ui->labelGestImage->setPixmap(pix7.scaled(w,h,Qt::KeepAspectRatio));
break;
case ACC:
ui->labelGestImage->setPixmap(pix8.scaled(w,h,Qt::KeepAspectRatio));
break;
default: break;
}
imageTimer.start(1000); // Reset picture in 1 second
}
```

LISTING 4.10: Code to display the recognized gesture

4.2.2 CapGUI

Graphic Window

The graphic routine has two main components: graphic setup and graphic update. Using the QCustomPlot widget, four graphics are created, each one with a different colour. Then when it receive the four measurements from the microcontroller unit, the graphic lines are updated.

```
void MainWindow::setupRealtimeData(QCustomPlot *customPlot){

    ui->customPlot->addGraph(); // blue line
    ui->customPlot->graph(0)->setPen(QPen(Qt::blue));
    ui->customPlot->graph(0)->setName(QString(" Cap 1"));

    ui->customPlot->addGraph(); // red line
    ui->customPlot->graph(1)->setPen(QPen(Qt::red));
    ui->customPlot->graph(1)->setName(QString(" Cap 2"));

    ui->customPlot->addGraph(); // green line
    ui->customPlot->graph(2)->setPen(QPen(Qt::green));
        ui->customPlot->graph(2)->setName(QString(" Cap 3"));

        ui->customPlot->addGraph(); // yellow line
        ui->customPlot->graph(3)->setPen(QPen(Qt::yellow));
    ui->customPlot->graph(3)->setName(QString(" Cap 4"));

    connect(customPlot->xAxis, SIGNAL(rangeChanged(QCPRange)),
            customPlot->xAxis2, SLOT(setRange(QCPRange)));
    connect(&dataTimer, SIGNAL(timeout()),
            this, SLOT(realTimeDataSlot()));
    dataTimer.start(10);
}
```

LISTING 4.11: Code to setup the graphic window

Strength Window

To represent the strength signal, each capacitance value is converted to percentage value and plotted in a bar graph. When the user's hand is in very close to the electrode the bar graph is full, diminishing its value when the user's hand move away from the electrode.

4.3 Server

4.3.1 Client

All the gesture recognition algorithms begins in a similar way. The extract of code shows the configuration of TCP/IP communication, as it may be noted that the selected IP address for the microcontroller was ("192.168.1.90"). If the network communication succeeds, the microcontroller attempts to create a TCP/IP socket to issue the command. When the communication is established, the microcontroller unit register the device in the destination system. Then, when an executed gesture is recognized it is sent by defining the destination identification, the registered device and which the recognized gesture.

```
#define eth_init "registernode=X_gesture_sensor\n"
#define eth_up_gesture "to=upperstack from=X_gesture_sensor message=\"<GESTURE_UP>\"\n"

static const char*      mbedIp      = "192.168.1.100"; //IP
static const char*      mbedMask    = "255.255.255.0"; // Mask
static const char*      mbedGateway = "0.0.0.0";     //Gateway
int main() {
bool EthernetValid = false;
eth.init(mbedIp,mbedMask,mbedGateway);
while(EthernetValid == false){
    if(sock.connect("192.168.1.73",6969) == -1) EthernetValid = false;
    else EthernetValid = true;
}
...
}
```

LISTING 4.12: TCP/IP Initialization Code

4.3.2 Server

The server application is configured to accept any connection. A linked list is implemented to store the deviceID, allowing to verify if it already exists. If it exists the device sensor can send the recognized gesture. When the incoming connection is accepted, a thread is assigned to handle the communication for each client.

4.4 Results

This section will present the experimental results obtained after the implementation and integration of the system.

4.4.1 Interface with the Gesture Devices

When a user perform a gesture, it will be interacting with one of the following devices: TI FDC1004, Microchip MGC3130 and Hover.

Texas Instruments FDC1004

The device sensor is used to acquire hand's trajectory. In order to detect the direction of the motion the proximity values must be read to know from which electrode it began and ended. When the user's hand is near the electrodes its value is increased. As the Figure 4.5 illustrates, the application can successfully receive the four electrodes measurements (up, down, left and right electrodes, respectively). In the first test no objects where in near proximity, therefore all of the values were similar, around 1252000 raw data measurement (23-bit), which corresponds approximately to 2.38 pF. In the following test the hand was hovering the left electrode causing the same to increase its capacitance.

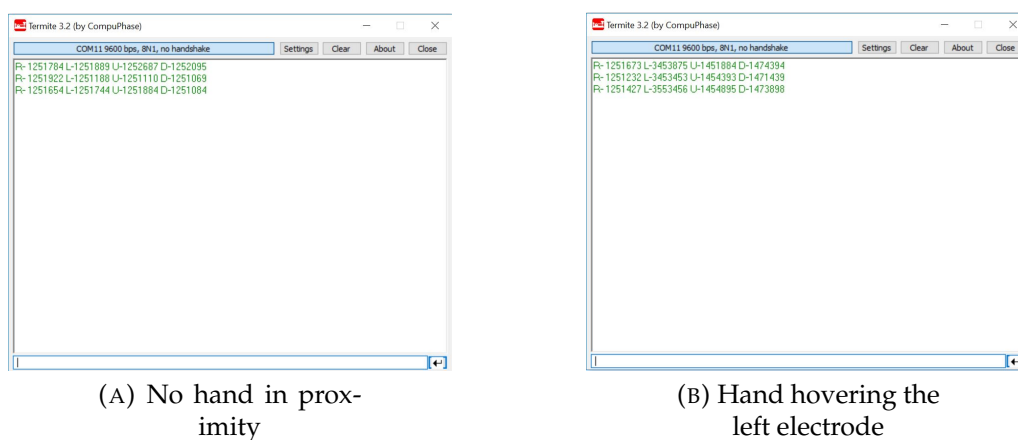


FIGURE 4.5: Result of reading the four channels of the Texas Instruments FDC1004

Hover

In the case of the Hover sensor, the gesture flags were read to test the system functioning. Figure 4.6 shows the results of a user performing the following sequence of gestures: down and down-up.

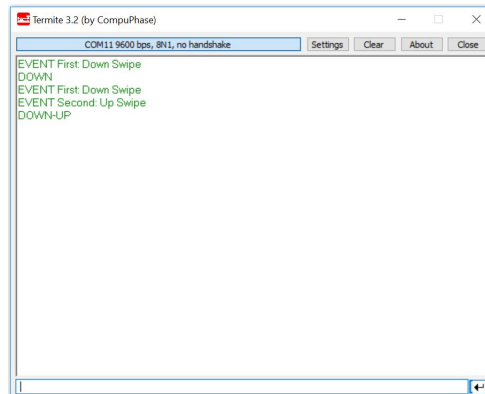


FIGURE 4.6: Result of reading the gestures from the Hover sensing device

Microchip MGC3130

The device sensor is used to obtain the hand's trajectory that is expressed through the positional data of the hand. Figure 4.7 shows the sequence of points of a right swipe.

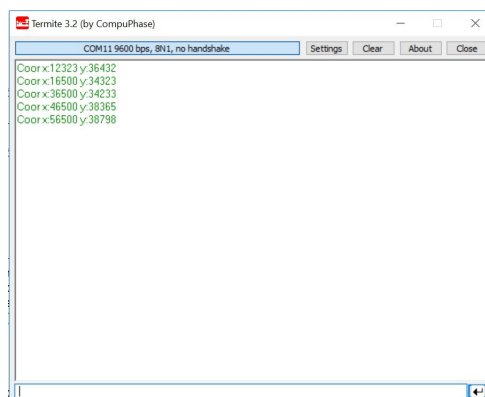


FIGURE 4.7: Result of reading the gestures hand's trajectory with the MGC3130 sensing device

4.4.2 Graphical User Interfaces

After the successful interface with the three sensing devices, two applications were developed. Due to a lack of configurations in the Hover, the development of a graphical application was not considered essential.

CapGUI

The application is always capturing and plotting the four capacitance values of the four electrodes as well as their signal strength. When a gesture is recognized the microcontroller issue a command via serial port, this command is recognized by the application and changes the visual interface to select the corresponded image to be displayed.

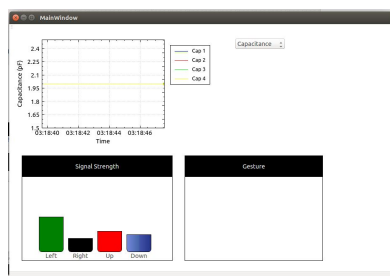


FIGURE 4.8: Graphical user interface for TI FDC1004

CoorGUI

The application is always waiting for an user interaction with device. When that occurs, the two dimensional data (represented by a dot) is shown in a grid plane. And if the performed sequence corresponds to a recognizable gesture, it will be displayed on the gesture window for a second. The final window shows the corresponded angle between the current data and the previous one, updating the number of executed circles when those are executed. Figure 4.9 shows the result when a user executes a right swipe. Thus, the right swipe picture is displayed, the angle is around the zero degrees and the final positional data in the sequence is near the right edge of the grid plane.

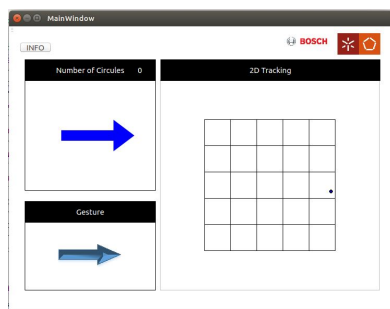
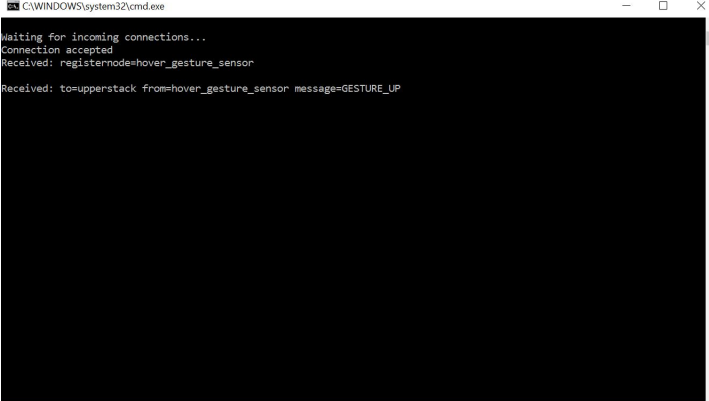


FIGURE 4.9: Graphical user interface for Microchip MGC3130 device

4.5 HMI System

The TCP/IP commands sent by the gesture recognition devices where design to make possible the integration in Bosch DSM. Firstly, the local server acknowledge the existence of the device sensor and thereafter the device sensor sends the correspondent gesture. As Figure 4.10 shows the HMI system receives the initial command to register the device in the server and then receives another command indicating that the driver or passenger performed a right swipe and the correspondent gesture must be interpreted to trigger an car functionality.



```
C:\WINDOWS\system32\cmd.exe
Waiting for incoming connections...
Connection accepted
Received: registernode=hover_gesture_sensor
Received: to=upperstack from=hover_gesture_sensor message=GESTURE_UP
```

FIGURE 4.10: Result of receiving the TCP/IP commands from the microcontroller unit

4.6 Algorithm evaluation

Due to a unavailability of a driving simulator mockup, the tests were performed in a laboratory-like environment with controlled conditions of environmental temperature (approximately of 20 degrees) and humidity. The gesture recognition systems were studied by 12 people in order to evaluate the gestures detection rate. The study consisted of performing the same gesture 30 times with different hand positions and speed. The individual passed through a learning period, where the recognized gestures and sensing device position and orientation where explained.

4.6.1 Texas Instruments FDC1004

The two developed proximity algorithms were tested; difference approach - based on the difference between the left and right electrodes and between the up and down electrodes and derivative approach - based on the difference between the current measurement and the previous measurement. The last approach is indicated to detect proximity from a single electrode, achieving an accuracy greater than 78%, having difficulty when the two electrodes are close to each other. The other approach was used to detect the linear gestures (left, right, up, down swipes). Table 4.2 shows that the results were very similar with each gesture accuracy ranging from 71.3% to 75.3%.

TABLE 4.2: Texas Instruments FDC1004 - Evaluation of linear gesture recognition performance

Recognition Rate	Left (%)	Right (%)	Up (%)	Down (%)
Maximum	75.3	74.3	75.2	74.3
Minimum	71.3	72.3	73.1	72.3
Average	72.3	72.6	71.2	73.3

4.6.2 Hover Gesture

How is possible to verify in Table 4.3 shows that the results were very similar with each gesture ranging from 87.2% to 88.3%. When the gesture is performed over 7 cm the detecting rate lowers significantly.

TABLE 4.3: Hover - Evaluation of linear gesture recognition performance

Recognition Rate	Left (%)	Right (%)	Up (%)	Down (%)
Maximum	88.2	88.3	88.2	88.3
Minimum	87.3	87.3	87.2	87.3
Average	87.9	87.9	87.8	87.9

Table 4.4 show that the algorithm is capable of distinguish well between the a linear gesture and sequential gesture.

TABLE 4.4: Hover - Evaluation of linear gesture recognition performance

Recognition Rate	Left-Right (%)	Right-Left (%)	Up-Down (%)	Down-Up (%)
Maximum	88.1	88.3	88.2	88.3
Minimum	86.3	87.3	86.2	86.3
Average	87.9	88	87.9	88.1

4.6.3 Microchip MGC3130

Table 4.5 shows that the results are very similar with each gesture ranging from 84.6% to 80.2%. In the case of the in and out motion the results were slightly worst.

TABLE 4.5: Microchip MGC3130 - Evaluation of linear gesture recognition performance

Recognition Rate	Left (%)	Right (%)	Up (%)	Down (%)	In (%)	Out (%)
Maximum	84.7	84.6	83.7	84.3	83.3	83.4
Minimum	82.4	82.2	82.4	82.4	80.2	80.3
Average	83.9	83.8	83.9	84	82.2	82.3

Table 4.6 shows that the algorithm is capable of distinguish well between the a linear gesture and sequential gesture.

TABLE 4.6: Microchip MGC3130 - Evaluation of sequential gestures recognition performance

Recognition Rate	Left-Right (%)	Right-Left (%)	Up-Down (%)	Down-Left (%)
Maximum	84.6	84.3	84.2	84.3
Minimum	81.3	81.3	81.2	81.3
Average	82.5	82.5	82.3	82.2

In the case of rotational gestures the detection rate (Table 4.7) was considerably lower, which suggests that other methods should be experimented, probably one based on machine learning.

TABLE 4.7: Microchip MGC3130 - Evaluation of rotational gestures recognition performance

Recognition Rate	Clockwise (%)	Anti-clockwise (%)
Maximum	61.3	61.1
Minimum	20.3	23.6
Average	37.6	36.3

4.7 Results Analysis

The results suggests that this technology can be applied to cars because of the versatility they can be applied. Electric fields are not perceived by users because they propagate through many materials in the environment such as plastics, wood and leather. Thus, capacitive sensing can be applied invisibly with minimum effect on the measurement. Moreover, the frequency range in which the sensor operates of a few kHz is not an interval that usually disturbs others electronic systems. This technology can be used in a range of automotive applications: keyless door entry control, initiating the car unlock process when the user approaches the door handle; wake up and illuminate the touchscreen when the user hand is near; turn on/off systems to control car lights, roof and hood; and detection of dynamic gestures to control vehicles infotainment.

Chapter 5

Conclusions and Future Work

This chapter presents a critical analysis about the developed work and propose some alterations to be implemented as future work.

5.1 Conclusions

Technology is evolving rapidly and as a consequence the automotive industry is changing in the ways drivers and passengers interact with their cars. The list of features includes HVAC system, multimedia system, navigation system, among others. Most of these new features are no longer an exclusive domain of luxury cars, being currently available at family cars. Delivering such a huge quantity of infotainment features in a safe and non-distracting way is not easy task, even more if these features continue to multiply at current rates.

This thesis presented a gesture recognition system, which satisfies the need for a natural HMI's between the driver and the car. Demonstrating a hand gesture recognition system for interpreting 2D dynamic hand gestures. The system consists of three main modules: sensing devices, monitoring graphical applications, and HMI system. Firstly, three sensing devices were selected, those who offer most functionalities and sensing range. For each device was developed a gesture library to recognize simple linear gestures, sequence gestures and if possible rotational gestures. The monitoring applications allowed to track the signals in real time as well as to have a visual feedback on the recognized gesture. Lastly, the HMI system was implemented as a local server to simulate the communication between the sensing devices and the Bosch DSM.

Finally, the most of the proposed requests were accomplished, with the devices recognizing the performed gestures allowing a more directly communication between the driver and the vehicle infotainment. The developed gesture recognition systems are capable of recognizing gestures with different hand and finger positions, speeds, distances and execution times. The system communicate to the server in a period of time of less than 0.8 seconds, which is under the requirement of a second. The system can detecting more than 70% of the gestures, with a variable percentage depending on the sensing device, except for rotational gestures that due to the variability of ways of performing the same gesture, which suggests that other approaches should be used, probably one in which a data base is created.

5.2 Future Work

The work presented a preliminary study on gesture recognition using capacitive sensing with space for improvement.

In addition to the capacitive technology, other technologies, i.e. IR and TOF cameras, should be evaluated their differences, and analyze for which automotive HMI concepts they should be used.

The second suggestion concerns to new electrode design that could be developed using different material and shapes. These new electrode designs could be develop so the system can be fitted into the car. For example, using just only one electrode, proximity can be detected; using two electrodes, the bidirectional swipe gestures can be recognized.

The developed algorithms could be improved and also different gesture recognition algorithms could be tested to increase the recognition rate. The system was developed using threshold base models, nonetheless HMM modals were studied, thereby, a comparison between these two models could be presented.

Due to time constrains, it was not possible the integration on a simulated driving environment to test the develop system in a more realist way, i.e. Bosch DSM. These tests will be performed by a Human Factor Team, to study different concepts and evaluate their usability.

References

- [1] Technavio, *Global automotive human machine interface (hmi) market 2015-2019*. Technavio, 2015, pp. 1–74.
- [2] C. Chung and E. Rantanen, *Gestural Interaction with In-Vehicle Audio and Climate Controls*, October. 2010, pp. 1406–1410, ISBN: 1071-1813. DOI: 10.1177/154193121005401911. [Online]. Available: <http://pro.sagepub.com/content/54/19/1406.short>.
- [3] A. Riener, A. Ferscha, F. Bachmair, P. Hagmüller, A. Lemme, D. Mutterthaler, D. Pühringer, H. Rogner, A. Tappe, and F. Weger, *Standardization of the in-car gesture interaction space*. 2013, pp. 14–21, ISBN: 9781450324786. DOI: 10.1145/2516540.2516544. [Online]. Available: <http://dl.acm.org/citation.cfm?doid=2516540.2516544>.
- [4] C. f.D. C. and Prevention. (2011). *Distracted Driving in the United States and Europe*, [Online]. Available: <http://www.cdc.gov/Features/dsDistractedDriving/>.
- [5] C. J. D. Patten, *COGNITIVELoad AND THE DRIVER Understanding the Effects of Cognitive Workload on Driving from a Human Information Processing Perspective*. 2007, p. 144, ISBN: 9789171554093.
- [6] D. Bannach, O. Amft, K. S. Kunze, E. A. Heinz, G. Tr, and P. Lukowicz, “Waving Real Hand Gestures Recorded by Wearable Motion Sensors to a Virtual Car and Driver in a Mixed-Reality Parking Game”, [Online]. Available: <http://kaikunze.de/papers/bannach2007waving.pdf>.
- [7] IHS. (2013). *Sales of automotive proximity and gesture recognition systems shift into high gear*, [Online]. Available: <http://press.ihs.com/press-release/design-supply-chain-media/sales-automotive-proximity-and-gesture-recognition-systems>.
- [8] A. Kendon, “Current issues in the study of gesture”, pp. 101–134, 1989.

- [9] D. Saffer, *Designing Gestural Interfaces*, ISBN: 9780596518394.
- [10] A. Riener and P Wintersberger, *Natural, intuitive finger based input as substitution for traditional vehicle control*. 2011, pp. 159–166, ISBN: 9781450312318. DOI: 10.1145/2381416.2381442. [Online]. Available: <http://dl.acm.org/citation.cfm?id=2381442>.
- [11] N. Hobbs and L. Chi. (2013). Gesture-based automotive controls. US Patent App. 13/437,730, [Online]. Available: <https://www.google.com/patents/US20130261871>.
- [12] Nielsen, J., *Usability engineering*. Boston: Academic Press, 1993, ISBN: 978-0-12-518406-9.
- [13] P. W. Jordan, “Human factors for pleasure in product use”, *Applied Ergonomics*, pp. 25–33, 1998, ISSN: 00036870. DOI: 10.1016/S0003-6870(97)00022-7. [Online]. Available: <http://linkinghub.elsevier.com/retrieve/pii/S0003687097000227>.
- [14] BMW. (). Bmw idrive concept, [Online]. Available: <http://s2.paultan.org/image/2015/01/bmw-iDrive-Touchscreen-0003.jpg>.
- [15] —, (). Bmw i8 ivision concept, [Online]. Available: http://cdn1.autoexpress.co.uk/sites/autoexpressuk/files/styles/article_main_image/public/2016/01/bmw_i_vision_future_interaction_on_loc-026.jpg?itok=8vQg39rp.
- [16] Hyundai. (). Hyundai genesis concept, [Online]. Available: <https://i.ytimg.com/vi/EPQYDWjawdE/maxresdefault.jpg>.
- [17] VW. (). Vw golf r touch concept, [Online]. Available: `\tinySource : \url{https://s.yimg.com/ny/api/res/1.2/I2VwvpyFKTNMaELnbvuvIQ--/YXBwaWQ9aGlnaGxhbmRlcjtzbt0xO3c9ODAw/http://1.yimg.com/cd/resizer/2.0/original/UIQPExHHOd8HVWHB_Yym-dcolzM}`.
- [18] —, (). Vw golf e touch concept, [Online]. Available: http://i.auto-bild.de/ir_img/1/2/4/2/6/1/6/VW-Golf-e-Golf-Touch-Sitzprobe-1200x800-06ceb497aac0ee21.jpg.
- [19] —, (). Vw budd-e concept, [Online]. Available: <https://pictures.dealer.com/m/mcgrathgroup/1409/ccd6421836d7bd7a9dc304acd3c0d33dx.jpg>.

- [20] Visteon. (). Visteon concept, [Online]. Available: <https://cnet2.cbsistatic.com/hub/i/r/2016/01/06/026c986e-5dae-4f6c-b605-45813059cc50/resize/970xauto/caccb35ff156e8cbbbf3c2c313b796/visteongesturecontrol-01.jpg>.
- [21] (1927). Leon termen plays the "theremin" in paris 1927, [Online]. Available: <http://i2.wp.com/120years.net/wordpress/wp-content/uploads/2013/09/Termen-plays-the-thereminvox.-Paris-1927.jpg>.
- [22] (2014), [Online]. Available: http://lghttp.50970.nexcesscdn.net/8029F77/vk/media/catalog/product/cache/1/image/9df78eab33525d08d6e5fb8d27136e95/m/o/moog_music_theremini_front.jpg.
- [23] D. Cohen, "Magnetic fields around the torso: Production by electrical activity of the human heart", vol. 156, pp. 652–654, 1967.
- [24] J. R. Smith, N. Gershenfeld, and S. A. Benton, "Electric Field Imaging", 1999. [Online]. Available: <http://cba.mit.edu/docs/theses/99.02.smithphd.pdf>.
- [25] D. W. F. Andreas Braun, "Capacitive proximity sensing in smart environments capacitive proximity sensing in smart environments", 2015.
- [26] T. Grosse-Puppendahl, Y. Berghoefler, A. Braun, R. Wimmer, and A. Kuijper, "Opencapsense : A rapid prototyping toolkit for pervasive interaction using capacitive sensing", pp. 151–158, 2013.
- [27] A. Braun and D. Germany, "Using the human body field as a medium for natural interaction", 2009.
- [28] A. Braun and P. Hamisu, "Designing a multi - purpose capacitive proximity sensing input device", pp. 1–8, 2011.
- [29] Microchip. (2015). Mgc3030/3130 3d tracking and gesture controller data sheet.
- [30] A. Braun, "Towards interactive car interiors : the Active Armrest Towards Interactive Car Interiors - the Active Armrest", no. October, 2014. DOI: 10.1145/2639189.2670191.
- [31] R. Wimmer, P. Holleis, M. Kranz, and A. Schmidt, "Thracker-using capacitive sensing for gesture recognition", *Distributed Computing Systems Workshops, 2006. ICDCS Workshops 2006. 26th IEEE International Conference on*, p. 64, 2006, ISSN: 1545-0678.

- [32] T. Grosse-Puppendahl, A. Braun, F. Kamieth, and A. Kuijper, "Swiss-Cheese Extended: An Object Recognition Method for Ubiquitous Interfaces based on Capacitive Proximity Sensing", *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems - CHI '13*, pp. 1401–1410, 2013. DOI: 10.1145/2470654.2466186. [Online]. Available: <http://dl.acm.org/citation.cfm?doid=2470654.2466186>.
- [33] A. Nelson, G. Singh, R. Robucci, C. Patel, and N. Banerjee, "Adaptive and Personalized Gesture Recognition using Textile Capacitive Sensor Arrays", 2015.
- [34] T Grosse-Puppendahl, S. Beck, and D Wilbers, "Rainbowfish: visual feedback on gesture-recognizing surfaces", *CHI'14 Extended Abstracts ...*, pp. 427–430, 2014. DOI: 10.1145/2559206.2574787. [Online]. Available: <http://dl.acm.org/citation.cfm?id=2574787>.
- [35] M. Le Goc, S. Taylor, S. Izadi, and C. Keskin, "A Low-cost Transparent Electric Field Sensor for 3D Interaction on Mobile Devices", *Proceedings of the 32nd annual ACM conference on Human factors in computing systems - CHI '14*, pp. 3167–3170, 2014. DOI: 10.1145/2556288.2557331. [Online]. Available: <http://dl.acm.org/citation.cfm?doid=2556288.2557331> <https://www.youtube.com/watch?v=DThJDAPxtxM>.
- [36] (). Stereo vision concept, [Online]. Available: <http://www.vision-systems.com/content/dam/VSD/print-articles/2012/04/leadf2.jpg>.
- [37] (). Structured light concept, [Online]. Available: <http://home.lagoa.com/wp-content/uploads/2014/04/structuredlightbands.jpg>.
- [38] (). Time of flight concept, [Online]. Available: <http://ca.mouser.com/images/microsites/time-of-flight-roboticsfig1.png>.
- [39] (). Depth image, [Online]. Available: <https://forum.libcinder.org/viewImage.do?fileId=23286000001372143&forumGroupId=23286000000003001>.
- [40] Y.-t. Liu, C.-y. Wu, P.-h. Shih, H.-s. Liang, and M. Y. Chen, "SoundSense : 3D Gesture Sensing using Ultrasound on Mobile Devices", 2011. [Online]. Available: <http://mrorz.github.io/files/soundsense.pdf>.

- [41] (). Ultrasonic concept, [Online]. Available: https://commons.wikimedia.org/wiki/File:Sonar_Principle_EN.svg1.
- [42] (). Infrared concept, [Online]. Available: http://education.rec.ri.cmu.edu/content/electronics/boe/ir_sensor/images/409px-IR_Sensor_Principles.png.
- [43] (). Controlled based gestures concept, [Online]. Available: http://ichef-1.bbci.co.uk/news/660/media/images/68831000/png/_68831329_arm.png.
- [44] (). Radarconcept, [Online]. Available: http://cdn2.expertreviews.co.uk/sites/expertreviews/files/styles/article_main_wide_image/public/2015/05/project_soli_google_io_2.png?itok=zQAoToNX.
- [45] (). Threshold, [Online]. Available: <https://svi.nl/wikiimg/SeedAndThreshold02.png>.
- [46] Y. S. Kim and K.-H. Baek, "A motion gesture sensor using photodiodes with limited field-of-view", [Online]. Available: <https://www.osapublishing.org/oe/fulltext.cfm?uri=oe-21-8-9206&id=252485#articleCitations>.
- [47] J. A. B. Adrianoa, E. A. Aquino, C. J. V. Cabael, B. E. D. Castro, and K. N. S. Jamoralin, "Gesture-based computer interaction through infrared motion sensing", [Online]. Available: <http://fs.mapua.edu.ph/MapuaLibrary/Thesis/Gesture-Based%20Computer%20Interaction%20Through%20Infrared%20Motion%20Sensing%20FULL%20TXT.pdf>.
- [48] F. Camastra and A. Vinciarelli, "Machine learning for audio, image and video analysis", [Online]. Available: <http://www.dcs.gla.ac.uk/~vincia/textbook.pdf>.
- [49] (). Euclidean distance, [Online]. Available: <http://image.slidesharecdn.com/eage2012dtwhvdb-140322204609-phpapp01/95/automated-seismictowell-ties-17-638.jpg?cb=1395521331>.
- [50] (). Dynamic time warping, [Online]. Available: <https://computersciencesource.files.wordpress.com/2010/01/conmat.png>.
- [51] G. Al-naymat, "SparseDTW: A Novel Approach to Speed up Dynamic Time Warping", [Online]. Available: <http://arxiv.org/pdf/1201.2969v1.pdf>.

- [52] S. Salvador and P. Chan, "FastDTW : Toward Accurate Dynamic Time Warping in Linear Time and Space", [Online]. Available: <https://gi.cebitec.uni-bielefeld.de/teaching/2007summer/jclub/papers/Salvador2004.pdf>.
- [53] T. Rakthanmanon, B. Campana, A. Mueen, G. Batista, B. Westover, Q. Zhu, J. Zakaria, and E. Keogh, "Searching and Mining Trillions of Time Series Subsequences under Dynamic Time Warping", [Online]. Available: http://www.cs.ucr.edu/~eamonn/SIGKDD__trillion.pdf.
- [54] (). Dtw path, [Online]. Available: <https://computersciencesource.files.wordpress.com/2010/01/conmat.png>.
- [55] T. Instruments. (). Fdc1004 capacitive sensing, [Online]. Available: http://www.electronicproducts.com/uploadedImages/Analog_Mixed_Signal_ICs/Sensors/FDC1004.jpg.
- [56] TI. (). Ti fdc1004, [Online]. Available: http://www.ti.com/diagrams/med_tida-00506_tida-00506_board_photo.jpg.
- [57] E. Artists. (). Ea lpc4088 quickstart board, [Online]. Available: https://www.embeddedartists.com/sites/default/files/image/product/mbed_lpc4088_diag_0.png.
- [58] NXP. (). Lpc1768 board, [Online]. Available: <https://blog.adafruit.com/wp-content/uploads/2012/05/window-186.jpg>.
- [59] M. Elmezain, A. Al-hamadi, and B. Michaelis, "Real-Time Capable System for Hand Gesture Recognition Using Hidden Markov Models in Stereo Color Image Sequences", pp. 65–72, [Online]. Available: <https://otik.uk.zcu.cz/bitstream/handle/11025/1315/Elmezain.pdf?sequence=1>.
- [60] (). Confusion matrix concept, [Online]. Available: <https://computersciencesource.files.wordpress.com/2010/01/conmat.png>.
- [61] Microchip. (). Microchip mg3130, [Online]. Available: <http://media.digikey.com/Photos/Microchip\%20Tech\%20Photos/DM160226.jpg>.
- [62] Hover. (). Hover gesture sensor, [Online]. Available: http://www.linuxuser.co.uk/wp-content/uploads/2015/01/Hover2_Web-copy_Web.jpg.

- [63] (). Time of flight concept, [Online]. Available: <https://forum.libcinder.org/viewImage.do?fileId=23286000001372143&forumGroupId=23286000000003001>.
- [64] E. Artists. (). Lpc4088 quickstart board, [Online]. Available: https://www.embeddedartists.com/sites/default/files/image/product/mbed_lpc4088_diag_0.png.
- [65] (). C and c++ programming languages, [Online]. Available: http://medo3g.16mb.com/gallery_gen/97e73ca6bc655ea9fe46e3d08129761c_492x480.png.
- [66] Matlab. (). Matlab logotype, [Online]. Available: http://3.bp.blogspot.com/-Sx8r0kxHI6k/UFMLf_QBGzI/AAAAAAAAAAM/FB0NDQht_Cc/s1600/Matlablogo.png.
- [67] QT. (). Cross-platform software, [Online]. Available: http://i1-linux.softpedia-static.com/screenshots/Qt_1.jpg.
- [68] Mbed. (). Arm mbed logo, [Online]. Available: <http://www.wiznet.co.kr/wp-content/uploads/2015/06/mbed-only.jpg>.
- [69] (). Confusion matrix, [Online]. Available: http://ir.sdu.edu.cn/bbs/attachments/month_1012/10121720336a39fac646afefd5.jpg.
- [70] TI, *Time of flight an introduction*, 2014. [Online]. Available: <http://www.ti.com/lit/wp/sloa190b/sloa190b.pdf>.