Vision-based Hand Wheel-chair Control

Paulo Trigueiros
Departamento de Informática Instituto Politécnico do Porto Porto, Portugal
paulo@trigueiros.org
Fernando Ribeiro
Departamento de Electrónica Industrial Universidade do Minho Guimarães, Portugal
Fernando@dei.uminho.pt

ABSTRACT

Several studies have shown that people with disabilities benefit substantially from access to a means of independent mobility and assistive technology. Researchers are using technology originally developed for mobile robots to create easier to use wheelchairs. With this kind of technology people with disabilities can gain a degree of independence in performing daily life activities. In this work a computer vision system is presented, able to drive a wheelchair with a minimum number of finger commands. The user hand is detected and segmented with the use of a kinect camera, and fingertips are extracted from depth information, and used as wheelchair commands.

Keywords – independent mobility, machine vision, wheelchair control, hand segmentation, finger control

I. INTRODUCTION

Several studies have shown that people with disabilities benefit substantially from access to a means of independent mobility and assistive technology [1], being independent mobility an important aspect of self-esteem [2]. Assistive devices such as powered wheelchairs improve one's quality of life. While the needs of many individuals with disabilities can be satisfied with traditional manual wheelchairs, some find it difficult to use. To accommodate this population and even other segments, several researchers have used technologies originally developed for mobile robots to create user friendly easier to use wheelchairs and "smart wheelchairs". Smart wheelchairs typically consist of either a standard power wheelchair to which a computer and a collection of sensors have been added or a mobile robot base to which a seat has been attached [2]. This work presents a simple and effective HCI (human computer interface) giving the user the ability to easily control a robotic wheelchair with a minimum number of finger commands. The main goal consists of giving the user the capability to control it without touching any physical device. For that purpose, a computer vision interface was developed, able to detect fingertips and able to use that information for driving the wheelchair.

To extract the hand and fingertip localization a kinect [3] camera is mounted on the back of the wheelchair pointing down to the user's hand. Machine vision is a promising sensor technology. With nowadays cameras, smaller than a lot of other sensors, they can be mounted in multiple locations, giving larger sensor coverage. Also, the cost of machine vision hardware has fallen significantly, and the solutions based on computer vision continue to improve.

II. RELATED WORK

Several wheelchair control devices were studied as alternatives to traditional input methods. Within the analogue control systems the joystick is by far the most common drive control [4] and it can be mounted for either right or left hand use. The joystick usually consists of a metal stick with a hard plastic head [5] that the user use to command the chair, an on/off switch, battery gauge, maximum speed control and sometimes a drive mode switch. With the wheelchair chin control, the gimble is mounted on a swingaway mount of some sort and positioned slightly below and forward of the chin (Figure 1). Chin controls work much the same as conventional joysticks in that the user simply pushes the gimble the direction they want to move and control their speed with the distance they push the gimble. This system is designed for a user with good head control. [4].

Figure 1. Chin control device, from MEYRA[1].

When set up to be actuated by the head, the gimble is mounted behind the head and attached to a headrest. The user pushes the headrest left to go left, right to go right and back to go forward (Figure 2). One drawback of this system is that the user cannot actually use the headrest, as a headrest, unless power to the chair is turned off. Another drawback of this set up is that the user must activate a switch to be able to move backwards, and activate the switch again to move forward. Normally this is not a serious drawback, but if the user is in a situation where several back and forward movements are required to get through a doorway or enter an elevator etc., it can be quite annoying to have to activate the forward/reverse switch so often.

Figure 2. Head array controller, from Adaptive Switch Lab. Inc.

1 http://atwiki.assistivetech.net/index.php/Alternative_wheelc hair_control

As an alternative to these systems, there is the finger wheelchair drive control and the touch pad wheelchair drive control. The first one consists of a small square box about 3"x3" x 1 1/2" with a 2" hole on top. The finger control box can be mounted just about anywhere the user can comfortably reach. To drive the chair with a finger control box, the user places one finger through the hole on the top of the box and moves the finger in the direction they want the power wheelchair to move. This system is basically the same principle as a joystick in that it is a proportional drive but instead of moving a gimble, the user moves a finger (Figure 3).

Figure 3. Finger wheelchair control.

The touch pad also drives the power wheelchairs with a finger and can be mounted on several places of the wheelchair depending on the ability of the user to access it. Because touch pads are also proportional analogue drives, the user can determine and control the speed of the wheelchair while moving simply by a small movement of the finger. Maskeliunas et al.[6] developed a HCI that tries to combine a traditional input with speech and video recognition technologies into one multimodal control "package". The authors think that the creation of a multimodal control interface combining various input and output modalities is a reasonable choice to fit the targeted audience with limited capabilities. Carlson et al. [7], proposed a method that integrates a brain computer interface (BCI) with a vision system that interprets the high-level BCI commands given the experimental environmental context. The proposed method gives the user the ability to effectively drive the wheelchair without any collisions around an office. More exotic input methods that have been implemented include detection of the wheelchair user's sight path (i.e., where the user is looking) through electrooculographic (EOG) activity [8] or the use of machine vision to calculate the position and orientation of the wheelchair user's head. Reis et.al. [9] [10] developed a platform for intelligent wheelchairs called IntellWheels composed of a control software, simulator/supervisor and a real prototype of the intelligent wheelchair . The simple multimodal human-robot interface developed allows the connection of several input modules, enabling the wheelchair control through flexible input sequences of distinct types of inputs (voice, facial expressions, head movements, keyboard and, joystick). The system is capable of storing of storing user defined associations, of input's sequences and corresponding output commands. In the proposed approach, hand segmentation is carried out through the use of a kinect[3] depth image in order to extract the hand blob and detect fingertips position to control the wheelchair. One advantage of the proposed solution compared to similar ones (finger drive), is the ability to control the wheelchair without touching any physical devices.

III. APPROACH DESCRIPTION

The proposed approach uses a kinect camera system mounted on the back of a wheelchair. The kinect is used to gather depth information for hand segmentation and finger detection. The user hand is segmented with the help of two planes that define the minimum and maximum thresholds of the extracted depth array. After hand segmentation the fingertips are extracted using the k-curvature algorithm [11]. The index finger is used to control the forward movement and turning left or right (Figure 4 -a, b, c) and the thumb control the lateral displacement to the left or right (Figure 4 - e, f). Movement to the rear is controlled with two fingers in 'V' (Figure 4 - d). Closed hand is the stop command.

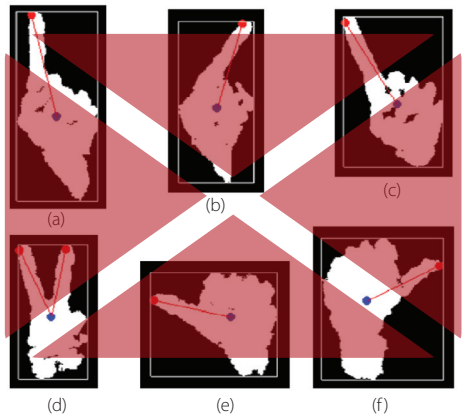

Figure 4. Finger commands used to control the wheelchair. (a) move forward, (b) turn right, (c) turn left, (d) move backwards, (e) move left, (f) move right.

A. Hand Segmentation

In order to segment the hand region, the nearest point to the camera is calculated on each frame. For each time t, the closest point on the depth image I is calculated according to the formula:

$$distMin = \min \left\{ \begin{matrix} I(x,y) if 0 \le x \le height(I) \\ and 0 \le y \le width(I) \end{matrix} \right\} \quad (1)$$

Using this value, two parallel planes [distMin-15, distMin+15] are defined to extract the hand blob from which the contour is calculated. The hand contour is then used to detect fingertips using the k-curvature algorithm.

B. K-Curvature

The k-curvature is an algorithm that attempts to find pixels that represent peaks along the contour perimeters [11] as shown in Figure 5.

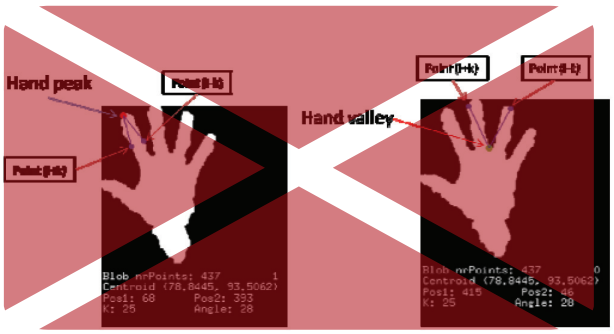

Figure 5. Hand peak and valley point detection.

At each pixel i in an hand contour C, the k-curvature is calculated and consists on the angle between the vectors A=[C(i), C(i-k)] and B=[C(i), C(i+k)], where k is a constant set equal to 30 in our implementation. The angle can be easily calculated using the dot product between the two vectors as illustrated in Figure 6.

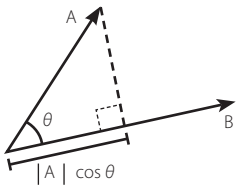

Figure 6. Dot product between vectors A and B.

A value of θ=35º is used, such that only points below this angle will be considered further. In order to classify the points as peaks or valleys, the cross product between the vectors is calculated (Figure 7). If the sign of the z component is positive, the point is labeled as a peak and stored; otherwise the point is a valley and is discarded.

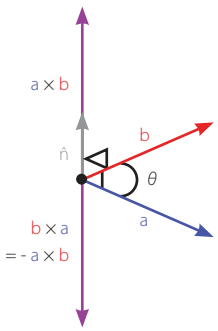

Figure 7. Cross product calculation.

Finally, an average of all peak points detected on each finger is calculated, since it was found that a set of peaks were detected in the neighborhood of the strongest locations.

The following is the algorithm used to calculate fingertip locations:

Algorithm 1.

```
// blob = hand blob binary image
fingerK = 30;
for(int i=0; I < (blob.nPts) - fingerK; i++) {
      //calculating angle between k-vectors
      //first the case where we are in the first k points....
      if(I < fingerK)
          v1.set(blob.pts[i].x-blob.pts[blob.nPts+i -fingerK].x,
                 blob.pts[i].y - blob.pts[blob.nPts+I - fingerK].y);
      else
          v1.set(blob.pts[i].x - blob.pts[I - fingerK].x,
                 blob.pts[i].y - blob.pts[I - fingerK].y);

      v2.set(blob.pts[i].x - blob.pts[i+fingerK].x,
             blob.pts[i].y - blob.pts[i+fingerK].y);

// 3D vectors lying in the XY-plane for cross product calculation
      if(i<fingerK)
          v1_3D.set(blob.pts[i].x-blob.pts[blob.nPts+i-fingerK].x,
                    blob.pts[i].y-lob.pts[blob.nPts+i-fingerK].y,0);
      else
          v1_3D.set(blob.pts[i].x-blob.pts[i-fingerK].x,
                     blob.pts[i].y-blob.pts[i-ingerK].y,0);
      v2_3D.set(blob.pts[i].x - blob.pts[i+fingerK].x,
                blob.pts[i].y - blob.pts[i+fingerK].y,0);
      vXv = v1_3D.cross(v2_3D);
      v1.normalize();
      v2.normalize();

      theta=v1.angle(v2);
      // if theta < 35 then we are at a peak or valey ( /\ or \/ )
      if(fabs(theta) < 35) {
          if(vXv.z> 0)
              fingerFound = true;
      }
}
```

C. Direction of movement and turning

The wheelchair moving direction is calculated by the dot product between the control vector, vector between the hand centroid and the fingertip, and a horizontal vector parallel to a line that crosses the image centre as illustrated in Figure 8.

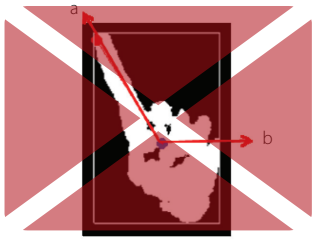

Figure 8. Vectors used in the calculation of finger orientation.

The angle θ is calculated by using the dot product between the two vectors according to equation 2.

$$\theta = \arccos\left(\frac{a \bullet b}{\|a\|\|b\|}\right) \quad (2)$$

IV. EXPERIMENTS AND RESULTS

In order to validate the method, a series of experiments were made with an MSL robot from the Minho Team (from University of Minho). The finger commands calculated according to the

above-described method are transmitted to the robot computer, allowing the user to control the wheelchair.

Several scenarios have been tried, with and without obstacles to test the easiness of wheelchair driving. The obtained results were very good for the problem in hands, showing that the solution is able to give people with disabilities an easy and inexpensive way to control a wheelchair.

The HCI (Figure 9) was developed using the C++ language, and the Openframeworks toolkit with two addons: the OpenCV[12] addon (ofxOpenCv) and the kinect addon (ofxKinect) under Ubuntu. OpenCV is used for some of the vision algorithms, and the fingertip extraction algorithm used, has been implemented by the same authors as an addon for the openframeworks. ofxKinect is used to control the kinect camera and to extract depth information. The computer used was a conventional notebook, with a 2GHz Core 2 Duo Processor.

The vision system operates at approximately 30 fps, and is able to correctly detect fingertips, which gives the possibility to have stable finger commands.

It takes 4 ms to calculate the near point and extract hand information, and takes about 2 ms to calculate fingertips for command classification.

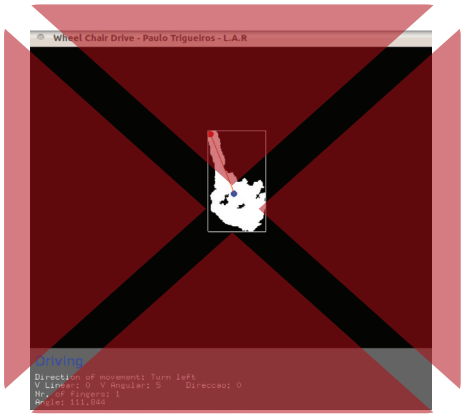

Figure 9. User interface showing segmented hand and vector information.

V. DISCUSSION

The proposed method consists of a new way to control a robotic wheelchair with the use of fingertip information, based on a Kinect sensor system to facilitate the extraction of useful hand features. One major advantage of this method consists on its simplicity, which leads to rapid learning rates, and gives the user the needed independent mobility. Also, the use of inexpensive hardware and open source tools make it a solution that can easily be applied to many other applications where human-computer interface can improve the quality of human life. The solution is not intended to be the best solution, but an alternative to the many solutions that exist in the market at the moment.

VI. CONCLUSIONS

This paper presents a method for controlling a robotic wheelchair using a kinect sensor system. One of the main advantages of the method is its simplicity in the number of commands the user has to use to drive the wheelchair. The user has just a number of finger commands. Another advantage is the ability to control the robot without wearing or touching any kind of device, using only finger movements and a kinect sensor which is mounted on the back of the wheelchair. The hardware used is very accessible, making this kind of solution very attractive. The solution is not universal – every disabled person is different and has different needs.

More tests, with the wheelchair developed in the laboratory of automation and robotics from the University of Minho, will be carried out to verify the robustness of the system and possible improvements.

The availability of open source tools to develop applications for this type of hardware is a very important aspect to take into account. These open source tools are quickly becoming widely used. The type of solution used is easily adaptable to robotic equipment in general and is very useful in most kinds of situations. The software achieves very good performances, which gives the possibility to drive the wheelchair in a very natural way. The use of depth information is sufficient to extract hand features, and allows the system to operate in environments with varying light intensity.

There is however a drawback with this type of camera that has to do with the minimum distance required to operate it (approximately 0.5 m), which requires a special adapter to install the camera in the wheelchair.

ACKNOWLEDGMENT

The authors wish to thank all members of the Laboratório de Automação e Robótica, at University of Minho, Guimarães. Also special thanks to the ALGORITMI Research Centre for the opportunity to develop this research work.

REFERENCES

[1] Lab, U.L.R., Vision-Based Interaction for Robot Arms and Wheelchairs in Human Environments, UMass Lowell Robotics Lab: Lowell, MA 01854.

[2] Simpson, R.C., Smart wheelchairs: A literature review. Journal of Rehabilitation Research & Development, 2005. 42(4): p. 423-346.

[3] Wikipedia. Kinect. [cited 2011; Available from: http://en.wikipedia.org/wiki/Kinect.

[4] Lange, M.L. Driving with Your Head: head controlled power wheelchair access methods. 2001 May 2001; Available from: http://quickplace.medgroup.com/QuickPlace/nrn/PageLibrary86256D21 004D93AC.nsf/h_05C223C63E65046486256F1D00569435/F55541397 7D54DB386256F17005AFA5F/?OpenDocument.

[5] Mahmood Ashraf, M.C., Towards Natural Interaction Using Alternate Wheelchair Controllers. International Journal on New Computer Architectures and Their Applications, 2011. 1(3): p. 1000-1013.

[6] R. Maskeliunas, R.S., Multimodal Wheelchair Control for the Paralyzed People, in Electronics and Electrical Engineering2011. p. 81-84.

[7] Tom Carlson, G.M., José del R. Millán, Vision-Based Shared Control for a BCI Wheelchair. International Journal of Bioelectromagnetism, 2011. 13(1): p. 20-21.

[8] Yanco, H.A., ed. Wheelesley, a Robotic Wheelchair System: Indoor Navigation and User Interface. Lecture notes in Artificial Intelligence: Assistive Technology and Artificial Intelligence, ed. M. VO, et al.1998, Springer-Verlag. 12.

[9] Reis, L.P., et al., IntellWheels MMI: A Flexible Interface for an Intelligent Wheelchair, in RoboCup 2009: 13th annual RoboCup Int. Symposium, Springer, Editor 2009: Graz, Austria. p. 296-307.

[10] Braga, R.A.M., et al., IntellWheels: A Modular Development Platform for Intelligent Wheelchairs. JRRD -Journal of Rehabilitation Research and Development, 2011. 48(9): p. 1061-1076.

[11] Malik, S., Real-time Hand tracking and Finger Tracking for Interaction, 2003.

[12] Bradski, G. and A. Kaehler, eds. Learning OpenCV: Computer vision with the OpenCV library. 2008, O'Reilly Media.