# Vision-based hand segmentation techniques for human-robot interaction for real-time applications

Paulo Trigueiros
*paulo@trigueiros.org*

*Instituto Politécnico do Porto - ISCAP*

Fernando Ribeiro
*fernando@dei.uminho.pt*

*Universidade do Minho*

Gil Lopes
*lopes.gil@gmail.com*

*Universidade do Minho*

ABSTRACT

One of the most important tasks in hand recognition applications for human-robot interaction is hand segmentation. This work presents a method that uses a Microsoft Kinect camera, for hand localization and segmentation. One important aspect for robot control besides hand localization is hand orientation, which is used in this work to control robot heading direction (left or right) and linear velocity. The system first calculates hand position, and then a kalman filter is used to estimate displacement and linear velocity in a smoother way. Experimental results show that the system is easy to use, and can be applied on several different human-computer interface applications.

**Keywords:** human-robot interaction, hand segmentation, hand control, kalman filter, kinect

## 1 INTRODUCTION

Recent applications in Human-Computer Interaction (HCI) and Computer Vision (CV) are raising a great opportunity to improve human life.
Nowadays, the keyboard, the mouse and remote controls are used as the main interfaces for transferring information and commands to computerized equipment (Hassanpour and Shahbahrami 2010). In some applications involving three-dimensional information, such as visualization, computer games and control of robots, other interfaces based on trackballs, joysticks and data gloves (Garg, Aggarwal et al. 2009) (Murthy and Jadon 2009) (Chen, Georganas et al. 2007) are being used. In our daily life, however, humans use vision and hearing as main sources of information in an environment. Therefore, one may ask to what extent it would be possible to develop computerized equipment able to communicate with humans in a similar way, by understanding visual input.
Main advantages of using visual input in this context are that visual information makes it possible to communicate with computerized equipment at a distance, without the need for physical contact with the equipment to be controlled.
The purpose of this project is to develop a user interface able of extracting user hands position and orientation on each frame, and use that information to communicate remotely with a robot.
Several human-computer applications use skin-color as one of the basic features to detect and analyze human hands for human-computer or human-robot interaction. These applications have different aims and different constraints under which the human hands must be analyzed. One crucial point, which is common to most of these applications, is the accuracy of hand segmentation. Due to this important fact, it was decided to use a Kinect camera, and take advantage of the depth camera module to acquire the hand blobs for further processing, eliminating this way, one constant problem faced in this kind of applications being the illumination variation. The blob centroid and hand orientation are calculated to take control of linear velocity and direction as well as the heading direction for the robot. The method has the advantage of being robust and easy to use.
This paper describes the method implemented to solve the problem of hand recognition in order to achieve remote robot control and presents the software developed to prove its feasibility.
The paper is organized as follows: state of the art on related work in section 2, this new approach describing hand segmentation and detection of orientation in section 3, experiments and results of this method

are presented in section 4. The discussion of the proposed method is given in section 5, and conclusions are given in section 6.

## 2 RELATED WORK

Nowadays, human computer and human robot interaction using technology like Microsoft Kinect is giving its first steps. The solutions are starting to appear in many areas, and some examples of it are the work carried out at JAHIR - Joint-Action for Humans and Industrial Robots - in which a human co-worker is tracked using a Microsoft Kinect in a human-robot collaborative scenario (Robots)[1]. The solution that uses a kinect with OpenNI[2], is able to track a human and automatically send this tracking data to an interface between the Kinect and an industrial robot so that it will dynamically avoid contact with that human being.

During Health and Wellness Innovation 2011, the Microsoft Kinect was also used by Jin Joo Lee (Personal Robots group at the MIT Media Lab) to advance her research in modelling the dynamics of social interaction. The goal is to better understand the subtle cues in human-human communication in order to improve human-robot interactions. This project applies machine learning and gesture recognition algorithms to the motion capture data from the Kinect in order to detect nonverbal cues including mimicry and synchronous movement[3].

Another example of hand gesture detection using kinect is the work being done at the MIT Computer Science and Artificial Intelligence Lab (CSAIL) (MIT)[4]. The project uses ROS (Robot Operating System), a set of tools that help software developers create robot applications, libfreenect driver and MIT developed gesture recognition.

Hand segmentation for gesture recognition is a difficult task to solve, mainly because of different lighting conditions that normal cameras have to adapt to.

In the proposed approach this kind of problem is not a concern, since the kinect depth sensor, that consists of an infrared laser projector combined with a monochrome CMOS sensor and which captures video data in 3D, is not dependent on light and therefore works under any light conditions.

Our focus in the proposed method is to segment the obtained depth image to extract the hand coordinates, and calculate movement and heading to remotely control a robot.

[1] http://www6.in.tum.de/Main/ResearchJahir

[2] OpenNI is available from http://www.openni.org/

[3] http://newmed.media.mit.edu/blog/jom/2011/03/29/modeling-dynamics-social-interactions-kinect-better-healthcare

[4] http://www.ros.org/wiki/mit-ros-pkg/KinectDemos/MinorityReport

## 3 OUR APPROACH

The proposed approach uses the kinect sensor system to gather depth information. The depth information will be used to detect and segment hands using two planes that define the minimum and maximum thresholds. These thresholds are calculated taking into account the closest point to the camera. The extracted hand centroid (relative position) is then used to control the robot movement as indicated in Figure 1and Figure 2.

.

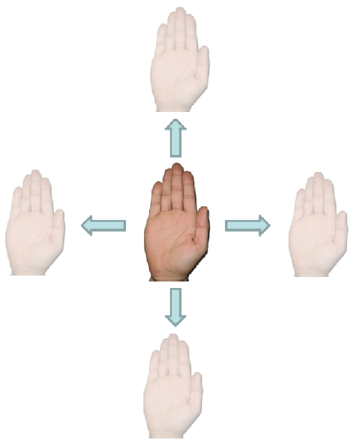


Figure 1. Hand movement used to control robot displacement.

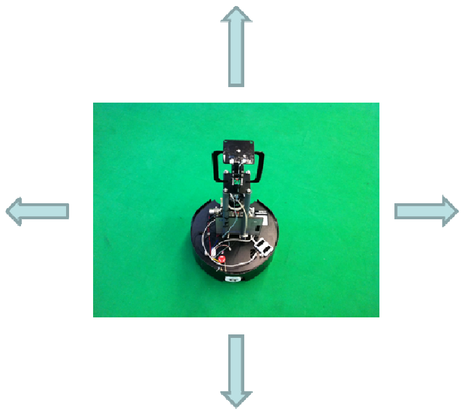


Figure 2. Robot movement relative to hand movement.

Hand rotation is used to control robot heading as illustrated in Figure 3

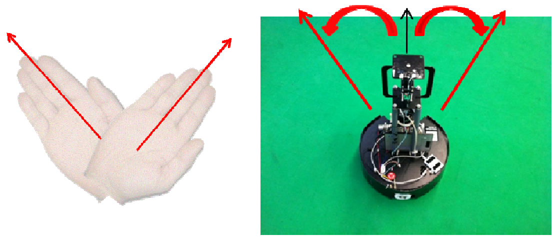


Figure 3. Robot heading dependent on hand rotation.

## 3.1 *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:

$$distMin = \min \begin{Bmatrix} I(x, y) : 0 \le x \le height(I) \\ and\, 0 \le y \le width(I) \end{Bmatrix} \tag{1}$$

Using this value, two parallel planes are defined to extract the hand binary mask and hand contours from the depth image.

**Algorithm 1 -** Nearest Point

Given the frame depth array (depth) and the image that is going to store the resulting binary mask (thImage) calculate:

```
Initialize minDistance = depth[0];
Initialize nrPixels = imageWidth * imageHeight;
for i = 0, …, nrPixels
    if minDistance = 0 and depth[i] > 0
        minDistance = depth[i];
    else if depth[i] > 0 and depth[i] < minDistance
        minDistance = depth[i];
    endif
end for

Initialize nearThreshold = minDistance – 5;
Initialize farThreshold = minDistance + 5;
for i = 0, … , nrPixels
    if depth[i] > nearThreshold and
depth[i] < farThreshold
        thImage[i] = 255;
    else
        thImage[i] = 0;
    endif
end for
```

The hand centroid (Equation 1 and 2) is then calculated, using the OpenCV method *findcontours*, being this value used for the relative hand position.

$$\bar{x} = \frac{1}{n}\sum x_i \tag{2}$$

$$\bar{y} = \frac{1}{n}\sum y_i \tag{3}$$

Using this centroid values $(\bar{x}, \bar{y})$, an estimated value for the new position is calculated using a kalman filter (Welch and Bishop 2001) (Press 1979), thereby enabling a smoother centroid movement, and respectively a smoother robot movement. This smoother robot movement not only makes it more visibly attractive but also increases the life expectancy of the robot motors.

The robot direction of movement is calculated according to the hand vector angle related to the image centre as illustrated in Figure 4

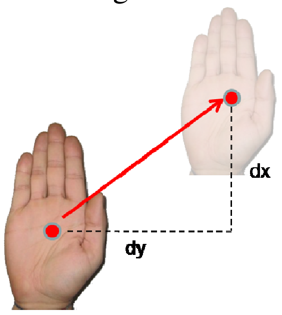


Figure 4. Direction of movement.

$$direction = \operatorname{atan2}(dx, dy) \tag{4}$$

The vector length represented by the distance between the image centre and the hand centroid, is then used to calculate the robot linear velocity, which is proportional to that value according to Equation 5.

$$linearV = \sqrt{dx^2 + dy^2} / 6 \tag{5}$$

where $linearV$ is the linear velocity transmitted to the robot and the constant value 6 was learned from the experiments to avoid sudden accelerations.
These values used to control the robot are then sent to the robot via a wireless client-server application.

## 3.2 *Orientation*

Hand orientation (θ) is calculated taking into account two vectors: one formed between the hand centroid and the farthest point from it, and another, a vector parallel to a horizontal line that passes through the centre of the image (Figure 5). The angle θ is then obtained by using the dot product[5] between the two vectors according to equation 6.

5 http://en.wikipedia.org/wiki/Dot_product

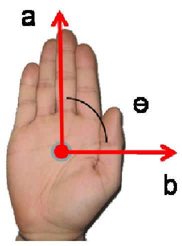


Figure 5. Vectors used for robot heading calculation.

$$\theta = \arccos\left(\frac{a \bullet b}{\|a\|\|b\|}\right) \tag{6}$$

being $a \bullet b$ the dot product between the two vectors and $\|a\|$ the norm of the vector.
The angle θ is used to control robot heading (left or right).

**Algorithm 2** - Orientation

Given the hand blob (blob) in the current frame and the center of the image (center) do:

```
Initialize distance = 0.0;
for i = 0,…, blob.nrPoints
    If blob.point[i].y < center.y
        v = (blob.point[i].x – center.x, blob.point[i].y – center .y);
    endif
    if distance < length(v)
        distance = length(v);
        maxPoint = (blob.point[i].x, blob.point[i].y);
    endif
end for

a = (maxPoint.x – center.x, maxPoint.y – center.y);
b = (imageWidth- center.x, 0);
normalize(a);
normalize(b);
theta = angle(a, b);

if theta < 80
    orientation = "Turn right";
else if theta > 100
    orientation = "Turn left";
else
    orientation = "Forward";
endif
```

## 4 EXPERIMENTS AND RESULTS

In order to validate the method, a soccer MSL robot from the Minho team was used to carry out a series of experiments.

A computer connected to a Kinect camera grabs hand movements and communicates that information through Wifi 802.11b to the robot on-board computer (Figure 6). The robot motion speed transmitted to the robot is proportional to the vector length that connects the hand centroid to the image centre point, and hand orientation gives us robot heading direction.

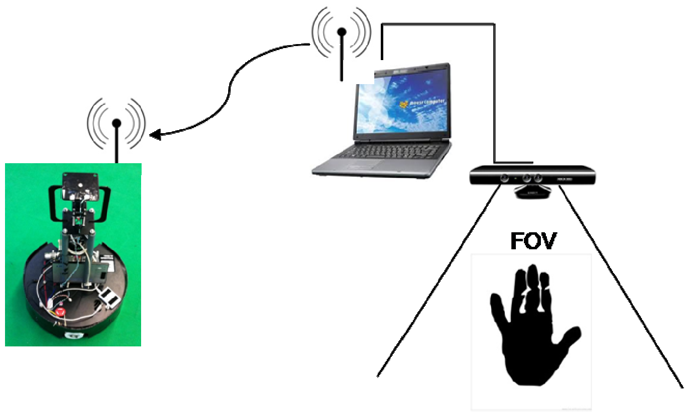


Figure 6. Computer with kinect for remote robot control.

The human-robot interface (Figure 7 ) was developed using the C++ language, Code::Blocks[6] IDE and the Openframeworks[7] toolkit with OpenCV[8](Bradski and Kaehler 2008) and libfreenect[9] under Ubuntu. OpenCV is used for vision, and libfreenect is used to interface with the kinect sensor unit. The computer used was a conventional notebook, with a 2GHz Core 2 Duo Processor.
The vision system operates at approximately 30 fps. It takes 4 ms to calculate the near point for segmentation, and takes around about 1 ms to detect the hand blob (Figure 7)
The communication with the robot is carried out through WiFi, and transmission speed is dependent on the network conditions.
Several scenarios were tried, with and without obstacles to test the easiness of robot driving. The locations chosen were all indoor environments since hand segmentation was not a problem with the use of a kinect depth sensor system. The obtained results were very good for the problem in hands.
The first contact with the software was a little bit strange, since all the users trying the system had no previous experience with this kind of situation – driving a robot with the hand – which led to comic results. After a while, everyone who has experienced it became confident, and everything started becoming more natural.

[6] Code::Blocks is available from http://www.codeblocks.org/
[7] Openframeworks is available from http://www.openframeworks.cc/
[8] OpenCV is available from http://opencv.willowgarage.com/wiki/
[9] libfreenect is available from http://www.openkinect.org

## 5 DISCUSSION

The proposed method consists of a new way to control a robot with a non contact method using the user hand, based on a Kinect sensor system to facilitate the extraction of useful information. It uses hand blobs and its centroids to calculate displacement, direction of movement from a starting point, and robot heading direction is given by the hand rotation relative to the vertical position.

One major advantage of this method consists on its simplicity of learning and usage, and proves to be very user friendly. Also, the use of inexpensive hardware makes it a solution that can easily be applied to many other applications where human-computer interface can improve the quality of human life.

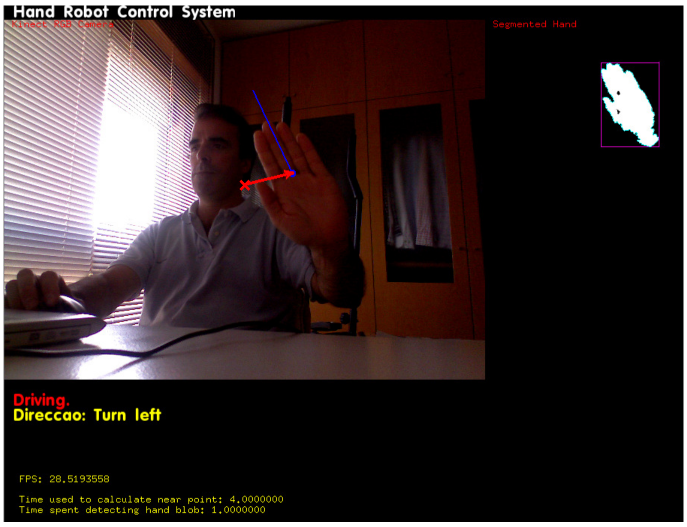
## 6 CONCLUSIONS

This paper presented a method able of remotely control the navigation of a robot using a kinect sensor system. One of the main advantages of the method is its robustness to light variations, since with the depth camera model there is no need to worry about. Another advantage is the ability to control the robot without wearing any kind of device, using only hand movements and a kinect sensor system which stands still on top of a table. The kind of hardware used is accessible, making this solution very attractive.

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.

This type of solution is easily adapted to robotics in general and is very useful in most kinds of situations. The implemented solution leads to a very fast algorithm, being able to be used in real-time systems.

There is however one disadvantage with the kinect sensor system, that has to do with the minimum distance required to operate with (approximately 0.5 m).



```
Driving.
Direccao: Turn left

FPS: 28.5193558

Time used to calculate near point: 4.0000000
Time spent detecting hand blob: 1.0000000
```

Figure 7. Camera image with user hand being detected.

**Acknowledgements**

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.

## 7 REFERENCES

Bradski, G. and A. Kaehler, Eds. (2008). Learning OpenCV: Computer vision with the OpenCV library, O'Reilly Media.

Chen, Q., N. D. Georganas, et al. (2007). Real-time Vision-based Hand Gesture Recognition Using Haar-like Features. Instrumentation and Measurement Technology Conference – IMTC 2007. Warsaw, Poland.

Garg, P., N. Aggarwal, et al. (2009). "Vision Based Hand Gesture Recognition." World Academy of Science, Engineering and Techonology: 972-977.

Hassanpour, R. and A. Shahbahrami (2010). "Human Computer Interaction Using Vision-Based Hand Gesture Recognition." Journal of Advances in Computer Research **2**: 21-30.

MIT. "ROS." from http://www.ros.org/wiki/mit-ros-pkg/KinectDemos/MinorityReport.

Murthy, G. R. S. and R. S. Jadon (2009). "A Review of Vision Based Hand Gestures Recognition." International Journal of Information Technology and Knowledge Management **2**(2): 405-410.

Press, A., Ed. (1979). Stochastic models, estimation, and control, Peter S. Maybeck.

Robots, J.-J.-A. f. H. a. I. "Kinect enabled robot workspace surveillance for multi-person collaboration."

Welch, G. and G. Bishop (2001). "An Introduction to the Kalman Filter." SIGGRAPH 2001.