HAND GESTURE RECOGNITION USING MULTILAYER PERCEPTRON NETWORK

S. Kajan, D. Pernecký, A. Hammad

Institute of Robotics and Cybernetics, Faculty of Electrical Engineering and Information Technology,
Slovak University of Technology in Bratislava, Slovak Republic

Abstract

This work is focused on an automatic recognition of hand gestures using depth sensors. Kinect for XBOX360 consists of depth sensors and was used in this work. The algorithm for the hand-tracking is based on fingertips detection method and on centers of palms detection method. For demonstration and evaluation purposes, the HandTrackerApp application was developed in C++ programming language. The application executes the whole process which consists of retrieving the image, filtering, hand-tracking and gesture recognition. Furthermore, the application consists of Graphical User Interface and is providing output classification visualization and data import/export. The multi-layer perceptron (MLP) network for gesture recognition is used. For training of this neural network Backpropagation of error was used. The training process of the MLP network was realized in Matlab using Neural Network Toolbox. For training and testing of MLP network a database of 12 gestures with 159 records was used.

1 Introduction

In the last years there has been great development in area of depth cameras. The use of these cameras, with their available prices, is found in control of video game consoles by moving body. For example Xbox 360 with the motion sensor Kinect from Microsoft. Besides the motion sensor Kinect there are other alternatives of different companies. Kinect sensor has been used in this work too.

Using the motion sensor Kinect many applications are developing and one of them is also tracking of hand and gesture hand recognition, with which this work deals too. Gesture recognition is very important in the area of robotics, which makes it possible to control and teach robots to perform certain actions. It is also a great tendency to create programs for gesture language recognition.

For gesture recognition several algorithms were created [1, 2, 6, 9, 11, 12, 13]. Proposed algorithms have to solve data processing from sensors, hand segmentation, hand tracking, obtaining necessary parameters for classification and the most important the gesture classification itself. In this paper we used the hand-tracking algorithm based on the fingertips detection method and on the centers of palms detection method [13]. Direct gesture classification by MLP network was used for normalized fingertips positions.

2 Hand tracking algorithm

Hand tracking algorithm has to filter out the hand from screen background at first. Then it is necessary to use such transformation, which produces an output as dataset of hand orientation description. These output data is afterwards used for gesture classification. In figure 1 is a block diagram for gesture recognition [11].

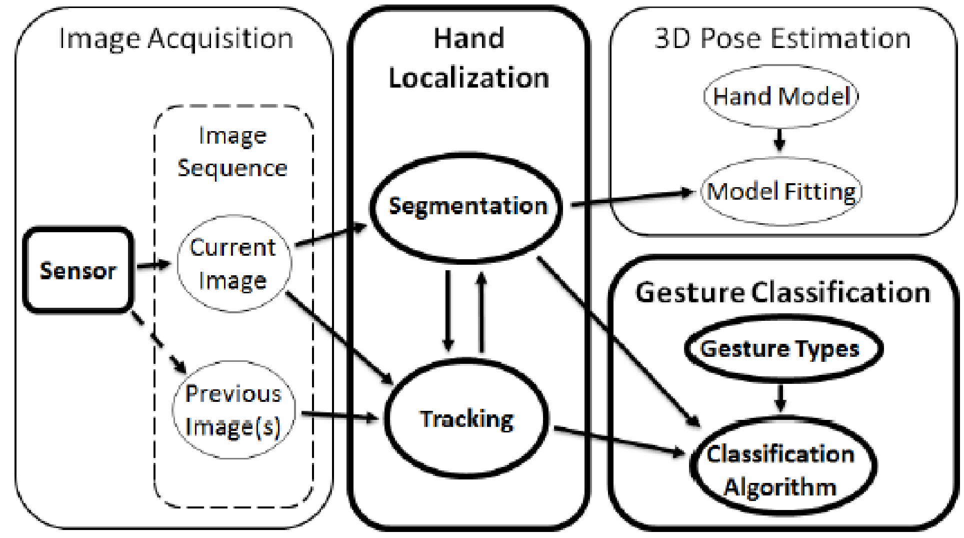


Figure 1: Block scheme of gesture recognition system

2.1 Hand segmentation

Hand segmentation from image was realized on basis of a depth map. The depth map and the depth filtered area are displayed in figure 2.

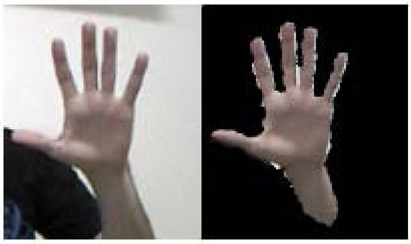


Figure 2: Segmentation on base of the depth map: left - original, right - depth filtered area

2.2 Hand tracking

When the area containing only the hand and part of the wrist is found, it is possible to analyze this filtered area further. A our algorithm was inspired by work of Zhi-Hua, K. Jung-Tae, L. Jianning, Z. Jing, and Y. Yu-Bo [13] but it was largely reworked and its final version is described below.

Because the output data from the depth sensor are noisy, it is necessary to filter out the output data. The first used filter was median filter, used to remove noise on the edges of the mask. The contours from binary image are obtained using findContours function from OpenCV library [13]. To find the center of the palm we used distance transformation. For each point of binary map a distance to nearest zero point of binary map is calculated. For distance calculating Euclidean norm was used. In figure 3 distance transformation is shown, where white color indicates the maximum distance value and black color indicates the minimum distance value. Center of the palm p_{palm} represents the point with maximum distance. The palm radius R_{palm} is calculated as the distance between center of the palm and the nearest point lying on contour palm (Figure 4).

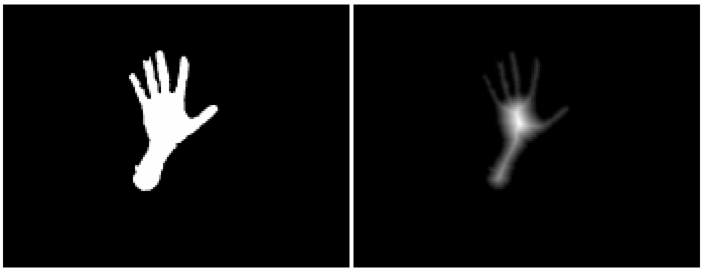


Figure 3: Left – hand mask, right – result of distance transformation into hand mask

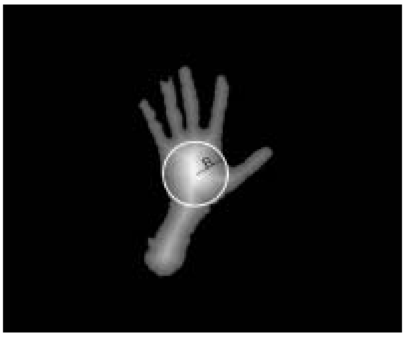


Figure 4: Depicting of circle with center in palm centre p_{palm} and radius R_{palm}

The palm isolation consists in creating the palm contour, which is obtained by created palm circle by the palm center p_{palm} and by the radius R_{palm} (Figure 4).

For fingertips positions tracking it is necessary to find wrist points c_j , which are used for creating a reference coordinate. To obtain the fingers we subtract palm mask from hand mask, as is shown in figure 5. Thus we get several objects among which are the fingers.

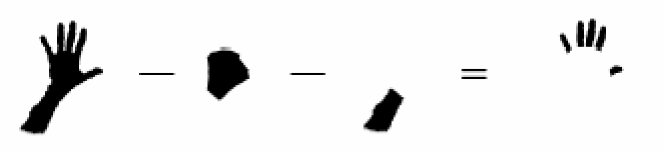


Figure 5: The procedure of obtaining fingers mask

Positions of fingertips ends are found using minAreaRect algorithm from library OpenCV for finding minimal rectangles describing fingers [15]. In figure 6 is shown final of the hand tracking algorithm.

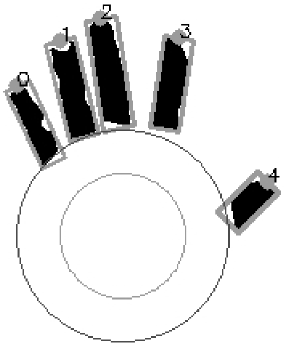


Figure 6: Final output of hand tracking algorithm

2.3 Gesture classification

The multi-layer perceptron (MLP) network for gesture classification was used [5, 7, 8, 10]. For positioning of fingertips v_i direct gesture classification using MLP network was used.

The fingertips position was normalized by position of wrist c_j. Normalized fingertips position u_i are calculated by equation (1), where o is centre of wrist position c_j.

$$u_i = \frac{v_i - o}{|c_2 - c_1|}, i \in \{1.2.3.4.5\} \tag{1}$$

Normalization by distance $|c_2 - c_1|$ enables the neural network to be invariant with respect to hand distance from camera and normalization by centre of wrist again enables to be invariant to position and orientation of the hand.

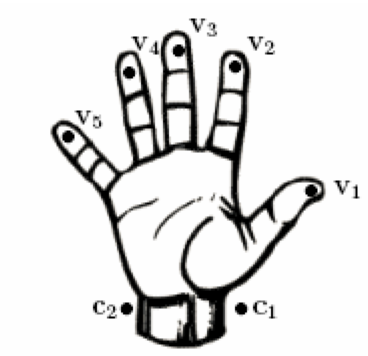


Figure 7: Input data for gesture recognition using MLP neural network

The structure of the MLP network used in gesture classification is displayed in Figure 8. In the hidden and output layers of the network the logical sigmoid function (logsig) was used. Network inputs in the database were represented as normalized positions of fingertips in a range of (0, 1) based on this data MLP realized classification into classes. Every network output had a value in a range of (0, 1) representing a group membership rate. For the MLP network training the modified back-propagation errors with adaptive learn rate and momentum parameter algorithm was used.

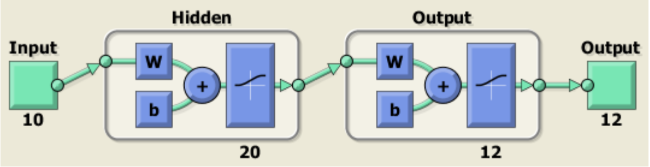


Figure 8: The structure of MLP neural network used for gesture recognition

3 Obtained results

The training of MLP network was realized in Matlab using Neural Network Toolbox [3]. For training and testing of MLP network a database of 12 gestures with 159 records (figure 9) was used. This gestures database was obtained from HandTrackerApp application, which was developed in C++ programming language [4]. The application executes the whole process which consists of retrieving the image, filtering, hand-tracking and gesture recognition. Furthermore, the application consists of Graphical User Interface and is providing output classification visualization and data import/export.

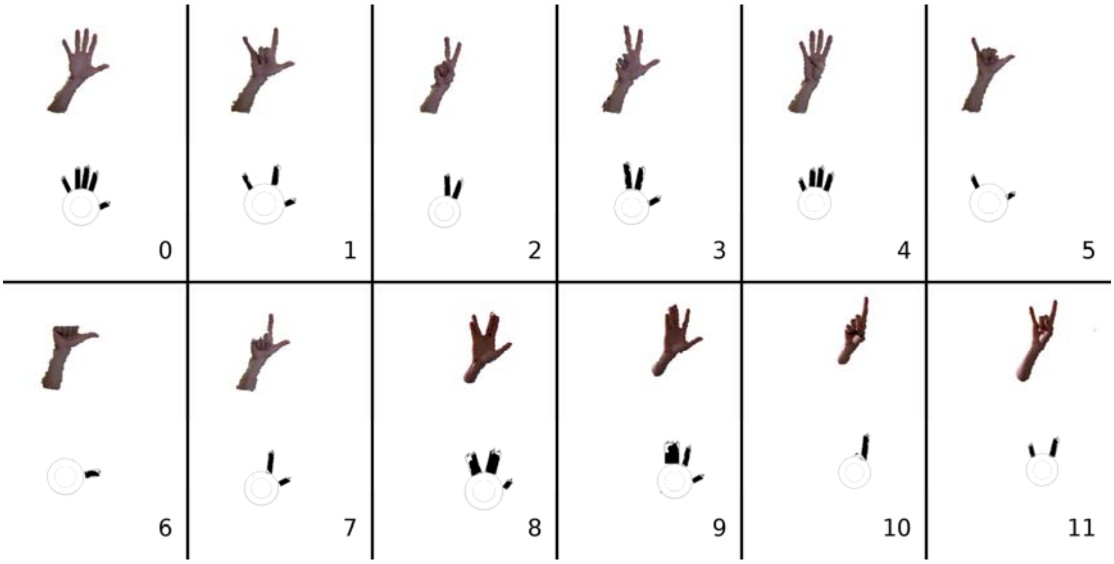


Figure 9: Gestures used for gesture recognition, using MLP network

In figure 10 input data of two gestures for training MLP network are shown. Data represented of the normalized fingertips positions vectors.

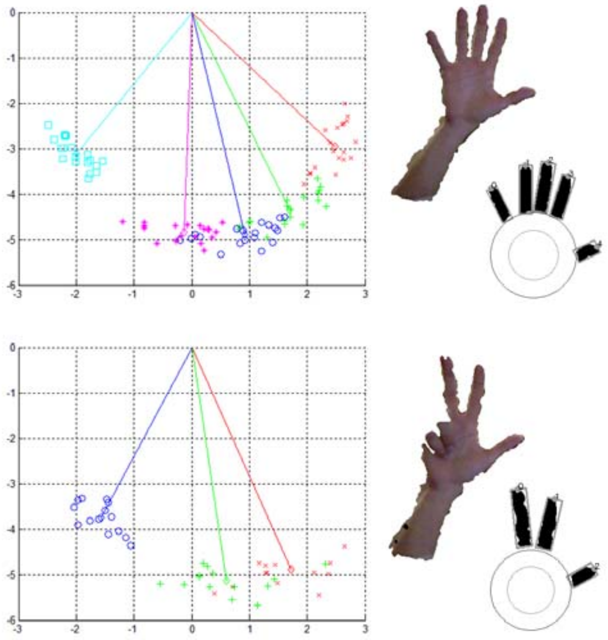
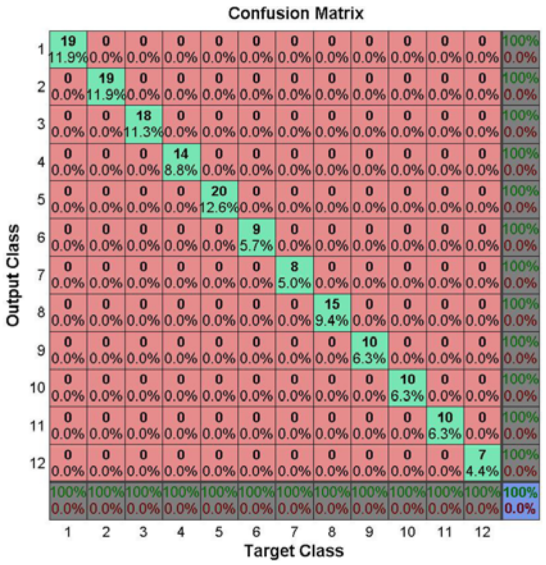


Figure 10: Input data of two gestures for training MLP network

Data for network training were divided into train (70%) and test (30%) data. For the MLP network training the modified back-propagation errors with adaptive learn rate and momentum parameter algorithm was used. The training algorithm was able to train MLP network to reach 100% success rate in classification for all 159 samples. Obtained results are shown as contingency table (confusion matrix) in figure 11.



Confusion Matrix

Output Class \ Target Class	1	2	3	4	5	6	7	8	9	10	11	12	
1	19 11.9%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	100% 0.0%
2	0 0.0%	19 11.9%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	100% 0.0%
3	0 0.0%	0 0.0%	18 11.3%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	100% 0.0%
4	0 0.0%	0 0.0%	0 0.0%	14 8.8%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	100% 0.0%
5	0 0.0%	0 0.0%	0 0.0%	0 0.0%	20 12.6%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	100% 0.0%
6	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	9 5.7%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	100% 0.0%
7	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	8 5.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	100% 0.0%
8	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	15 9.4%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	100% 0.0%
9	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	10 6.3%	0 0.0%	0 0.0%	0 0.0%	100% 0.0%
10	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	10 6.3%	0 0.0%	0 0.0%	100% 0.0%
11	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	10 6.3%	0 0.0%	100% 0.0%
12	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	7 4.4%	100% 0.0%
	100% 0.0%	100% 0.0%	100% 0.0%	100% 0.0%	100% 0.0%	100% 0.0%	100% 0.0%	100% 0.0%	100% 0.0%	100% 0.0%	100% 0.0%	100% 0.0%	100% 0.0%

Figure 11: Obtained results – contingency table (confusion matrix)

4 Conclusion

For demonstration and evaluation purposes, the HandTrackerApp application was developed in C++ programming language. MLP network was trained in Matlab and afterwards implemented to HandTrackerApp application. If the position of the fingertips was correctly detected by tracking algorithm, then the neural classification model showed a very good percentage rate. By successful testing of the MLP networks for classification problems we verified the suitability of their use in gesture recognition.

References

[1] A. Argyros, M. Loukaris. *Real-Time Tracking of Multiple Skin-Colored Objects with a Possibly Moving Camera. 2004,* [online].: http://www.ics.forth.gr/_publications/ 2004_05_ECCV04_hand_tracking_2d.pdf

[2] P. Breuer, C. Eckes, S. Muller. *Hand Gesture Recognition with a Novel IR Time-of-Flight Range Camera–A Pilot Study v Mirage* (Computer Vision/ Computer Graphics Collaboration Techniques), pp. 247-260, 2007

[3] H. Demuth, M. Beale, M. Hagan. *Neural network toolbox.* User's guide, 2008

[4] A. Hammad. *Rozpoznávanie pohybu ruky senzorom Kinect*, Diploma thesis, FEI STU Bratislava, 2015, (in Slovak)

[5] S. Kajan. *GUI for classification using multilayer perceptron network*, Technical Computing Prague, 2009

[6] C. Keskin, F. Kirac, E. Y. Kara,. L. Akarun, *Real time hand pose estimation using depth sensors v Computer* VisionWorkshops (ICCVWorkshops), pp. 1228-123, 2011

[7] V. Kvasnička, a kol. *Úvod do teórie neurónových sietí.* Bratislava: Iris, 1997. 285 s. ISBN 8088778301, (in Slovak)

[8] M. Negnevitsky, *Artificial Intelligence*. Pearson Education Limited, 2005

[9] I. Oikonomidis, N. Kyriazis, A. Argyros. *Efficient modelbased 3d tracking of hand articulations using kinect*, FORTH Institute of Computer Science, 2011.

[10]B. D. Ripley. *Pattern recognition and neural networks.* Cambridge university press, 1996. 403 s. ISBN 0-521-46086-7.

[11]J. Suarez, R. Murphy. *Hand Gesture Recognition with Depth Images*: A Review v The 21st IEEE International Symposium on Robot and Human Interactive Communication, France, 2012.

[12]P. Viola, M. Jones. *Rapid object detection using a boosted cascade of simple features v Computer Vision and Pattern Recognition* (CVPR), pp. I-511 -I-518, 2001

[13]CH. Zhi-Hua, K. Jung-Tae, L. Jianning, Z. Jing, Y. Yu-Bo. *Real-Time Hand Gesture Recognition Using Finger Segmentation,* The Scientific World Journal, Volume 2014 (2014), Article ID 267872

[14] OpenNI library, 2014 [Online]. Available: http://openni.org

[15] OpenCV library, 2014 [Online]. Available: http://opencv.org

Acknowledgments

The work has been supported by the grant agency KEGA No. 010STU-4/2014.

Ing. Slavomír Kajan, PhD.:
Institute of Robotics and Cybernetics, Faculty of Electrical Engineering and Information Technology, Slovak University of Technology in Bratislava, Slovak Republic, Ilkovičova 3, 812 19 Bratislava, E-mail: slavomir.kajan@stuba.sk

Ing. Daniel Pernecký:
Institute of Robotics and Cybernetics, Faculty of Electrical Engineering and Information Technology, Slovak University of Technology in Bratislava, Slovak Republic, Ilkovičova 3, 812 19 Bratislava, E-mail: daniel.pernecký@stuba.sk

Ing. Amir Hammad:
Institute of Robotics and Cybernetics, Faculty of Electrical Engineering and Information Technology, Slovak University of Technology in Bratislava, Slovak Republic, Ilkovičova 3, 812 19 Bratislava, E-mail: amir.ozk@gmail.com