ELECTRONICS AND ELECTRICAL ENGINEERING
ISSN 1392 – 1215 2007. No. 8(80)
ELEKTRONIKA IR ELEKTROTECHNIKA

MEDICINE TECHNOLOGY
T 115
MEDICINOS TECHNOLOGIJA

Estimating Characteristic Points of Human Body for Automatic Hand Pointing Gesture Recognition

P. Serafinavičius
Department of Electronics Engineering, Kaunas University of Technology,
Studentų str. 50, Kaunas, Lithuania, e-mail: paulius.se@gmail.com

Introduction

Multimodal user interfaces enable users to communicate with computers using the modality that best suits their requirements. Besides conventional mouse or keyboard input, these modalities include handwriting, speech, or gesture. The set of gestures, performed by humans when communicating with each other, includes pointing gestures. These are especially useful for kind of applications like smart rooms, virtual reality or household robots.

Let us consider that pointing gesture in the context of this paper is a movement of the arm towards a pointing target. Human perform the gesture in the communication with others to mark a specific object, location or direction. Two main tasks exist in the recognition of the pointing gestures: the detection of the gesture occurrence in natural arm movements and the estimation of the pointing direction.

Usually human tend to look at the objects with which they interact. There were investigated how people use speech and gaze when interacting with "an office of the future". They report that the subjects nearly always looked at a speech enabled office device before addressing it [1]. Similar results are reported. There were investigated how people use different interfaces to control room lights. They also report that subjects typically looked at the lights they wanted to control [2].

There are many approaches for the extraction of the body features by means of one or more cameras. In [3], the system that uses a statistical model of color and shape to obtain a 2D representation of the head and hands, was described. [4] Describes a 3D head and hands tracking system that calibrates automatically from watching a moving person. An integrated head and silhouette tracking approach based on color, dense stereo processing and face pattern detection is proposed in [5]. Similar fields were researched by Lithuanian authors. In [6], model-based method for estimation 3D head position is proposed. By method OpenCV characteristic points of face are detected and continuous real time tracking was realized by normalized correlation.

This paper presents a pointing gesture recognition system based on stereo vision and implemented using open source computer vision library (OpenCV). The conception of the system was reported in [7]. The system is able to detect pointing gestures and to determine the 3D pointing direction in real-time. To obtain input features for gesture recognition, at first, we detect a person's head and pointing hand, mark some feature points in detected regions of the head and pointing hand and then performed its tracking in 3D. Two cascades of boosted classifiers were used to detect the occurrence of the head and pointing hand. One has been trained on human face detection, the other - on pointing hand detection with different directions and illumination of sample pointing gestures.

In comparison to related works, our approach characterizes human body feature detection in combination using cascades of boosted classifier and detected feature points tracking in 3D.

Detection of the head and hand regions

Detection of the head was implemented using OpenCV function *cvHaarDetectObjects()*. The function finds rectangular regions in the given image that are likely to contain objects the cascade has been trained for and returns these regions as a sequence of rectangles. The function scans the image several times at different scales. Each time it considers overlapping regions in the image and applies the classifiers to the regions. It may also apply some heuristics to reduce the number of analyzed regions, such as Canny pruning. After it has proceeded and collected the candidate rectangles (regions that passed the classifier cascade), it groups them and returns a sequence of average rectangles for each large enough group. A trained classifier cascade for face detection has been taken from OpenCV samples.

The same function was used in implementation of the hand pointing gesture detection. A classifier used for pointing hand gesture was trained using separate OpenCV application called *haartraining*. It can train a cascade of boosted classifiers from a set of samples. There were prepared 1000 positive (hand pointing gesture) (Fig. 1) and

4000 negative (background) samples for training. Various photos were taken as background samples where similar objects like pointing gestures can not be located. A cascade of boosted classifiers was trained; it consists of 20 stages of simple stump classifiers in each stage. The type of haar features' set used in training was full upright and 45 degree rotated feature set. The size of training samples – 24x24 pixels.

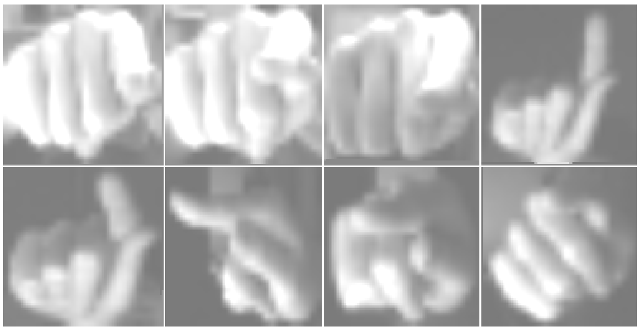
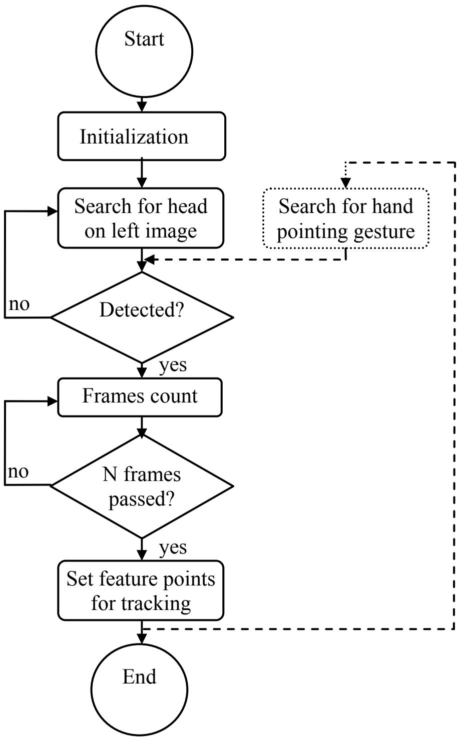


Fig. 1. Some samples of the hand pointing gesture which were used for training of the classifier

Our pointing gesture recognition system consists of two USB web cameras with a fixed baseline and connected to a standard portable PC. A special algorithm was developed for detecting and tracking the hand pointing gestures and estimating the pointing direction in 3D.



Start
Initialization
Search for head on left image
Search for hand pointing gesture
no
Detected?
yes
Frames count
no
N frames passed?
yes
Set feature points for tracking
End

Fig. 2. Flow chart diagram of setting feature points needed for head and hand regions' tracking in the camera's images

Detection process starts with initialization of cameras, video format setting, loading both cascades of boosted classifiers from files. The search of the head is performed on one of the camera's images. In our case, it is left camera. In order to achieve a robust detection of the head region we count a defined number of frames N. While the head region is located permanently, the number of frames will pass. After these frames have passed, the feature points, which will be used while tracking the head, are set. N=6 is enough to achieve good detection results for the head region.

Detection of pointing gesture process is similar to the one marked with dotted lines in the flow chart diagram (Fig. 2). Hand pointing gesture does not have such a rigid shape with certain features as the head, and can not be detected robustly enough. In such a case, more frames are needed to ensure robust hand pointing gesture detection. We estimated experimentally that 15 frames are enough for this. The bigger number of frames would increase robustness of detection, but it is not reasonable due to significant slow down of the detection process. If detection fails, while the number of frames is not passed, the counting of frames will restart from 0. This prevents from setting feature points to incorrect regions of the image.

Tracking of feature points

When both regions of the feature points are prepared the tracking of them starts. Flow chart diagram of feature points' tracking during one frame is shown in Fig. 3. It starts with a memory allocation of temporary buffers needed for calculation of optical flow in pyramids. If some counted feature points exist from previous frame, Lukas-Kanade optical flow calculation in pyramids iterative process starts. It is implemented using OpenCV function *CalcOpticalFlowPyrLK()* which is able to calculate optical flow for a sparse feature set using iterative Lucas-Kanade method in pyramids for two images (in our case for a stereo pair). After the calculation of optical flow, a loop starts for processing founded new feature points. The processing consists of the following:

1. Evaluating the distance between externally added point and existing ones:
$$D = \sqrt{dx^2 + dy^2}\ , \text{[px]} \tag{1}$$
while $dx = x_a - x_i$ and $dy = y_a - y_i$ are the differences between added point and the *i-th* point *(x, y)* coordinates.
2. If the distance is less than 2 pixels the added point is removed.
3. Calculation of centroid coordinates:
$$C(x, y) = \frac{\sum P_i(x_i, y_i)}{i}\ , \tag{2}$$
while *i* is the count of all available feature points, $P_i(x_i, y_i)$ is the *i-th* available feature point.
4. Evaluating the distance between centroid and the existing feature points. It is similar evaluation to (1).
5. If the evaluated distance is larger than 70 pixels, the feature point will be removed during the next frame. In such a case, if any point was removed, a new point with a coordinates of centroid from the last frame will be added.
6. Start a search for corresponding feature points on the right image.

Finding correspondent points on the right image is similar to left frame processing. It starts after the left frame feature points' processing is finished. It checks if the count of feature points is not equal to zero and then optical flow calculation is performed on the right image, using the same function *CalcOpticalFlowPyrLK()*. Calculation of centroid is performed on the right image separately from the left one, but the evaluation of distance, addition or removal of points are not performed because all these things lie on corresponding points' calculation from the left image.

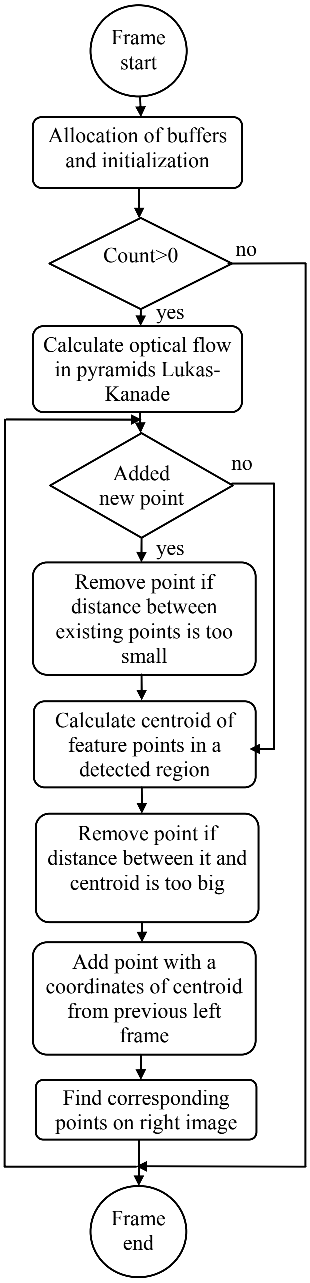

Frame start
Allocation of buffers and initialization
Count>0
no
yes
Calculate optical flow in pyramids Lukas-Kanade
Added new point
no
yes
Remove point if distance between existing points is too small
Calculate centroid of feature points in a detected region
Remove point if distance between it and centroid is too big
Add point with a coordinates of centroid from previous left frame
Find corresponding points on right image
Frame end

Fig. 3. Flow chart diagram of tracking feature points per frame

Experimental evaluation

Our system was evaluated experimentally. There are two main criteria of the evaluation:

1. The rate P of the correctly detected head and hand regions:

$$P(H) = h/n; \quad (3)$$

$$P(G) = g/n; \quad (4)$$

$$P = P(H)P(G); \quad (5)$$

while *P(H)* – probability of head region correct detection, *P(G)* – probability of pointing gesture correct detection, n – number of total tries, h – number of tries with correctly detected head region, g – number of tries with correctly detected hand pointing gesture, P – total probability of correct head and hand detection.

2. Robustness while tracking detected pointing gesture. It can be defined as ratio of successfully pointed targets with the number of available targets:

$$R = T_S / T_A; \quad (6)$$

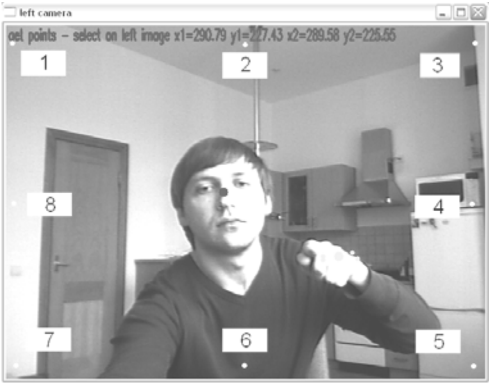
while T_S – successfully pointed targets, T_A – available targets.

Table 1. The results of correct detection rates of the head and pointing hand gesture regions. 3 people were asked to test the system detection rate. Each of them made n=10 tries

	P, %	P(H)	P(G)
1.	100	1,0	1,0
2.	90	1,0	0,9
3.	81	0,9	0,9
Avg. P	90,33		

Table 2. The results of robustness while tracking a pointing gesture. R – robustness in %. T_1-T_8 – targets (Fig. 4), 1 – target successfully pointed from the center of image and returned back to it, 0 – during pointing or returning back were failures of tracking.

	R, %	T_1	T_2	T_3	T_4	T_5	T_6	T_7	T_8
1.	62,5	1	0	1	0	1	1	0	1
2.	50,0	1	0	0	0	1	0	1	1
3.	62,5	1	0	1	0	1	0	1	1
Avg. R	58,33								



oel points – select on left image x1=290.79 y1=227.43 x2=289.58 y2=225.55

Fig. 4. These eight target points were used for evaluating the robustness of pointing gesture tracking

Discussion and conclusion

Detection rate is good enough while using proposed algorithm. Average rate of correctly detected head and hand regions is 90%.

While robustness of tracking algorithm is 58%, it still needs improvements. Especially it needs to increase robustness when pointing gesture is tracked over a background with similar color to human skin and big contrasts in the background result in decreased robustness of tracking. Therefore, current pointing gesture detection and tracking system can only be suitable in environments where backgrounds have low contrast colors which differ from human skin color.

References

1. **Maglio P. P., Matlock T., Campbel C. S., Zhai S., Smith B. A.** Gaze and speech in attentive user interfaces // Proceedings of the International Conference on Multimodal Interfaces. – Springer-Verlag, 2000.
2. **Brumitt B., Krumm J., Meyers B., Shafer S.** Let There Be Light: Comparing Interfaces for Homes of the Future // IEEE Personal Communications, 2000.
3. **Wren C., Azarbayejani A., Darrell T., Pentland A.** Pfinder: Real-Time Tracking of the Human Body // IEEE Transaction on Pattern Analysis and Machine Intelligence. – 1997. – Vol. 19, No. 7. – P. 780–785.
4. **Azarbayejani A., Pentland A.** Real-time Self-Calibrating Stereo Person Tracking Using 3-D Shape Estimation from Blob Features // Proceedings of 13th ICPR. – 1996. – Vol. 3. – P. 627.
5. **Darrell T., Gordon G., Harville M., Woodfill J.** Integrated Person Tracking Using Stereo, Color, and Pattern Detection // Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. – Santa Barbara, CA: IEEE Computer Society, 1998. – P. 601.
6. **Dervinis D.** Head Orientation Estimation using Characteristic Points of Face // Electronics and Electrical Engineering. – Kaunas: Technologija, 2006. – No. 8(72). – P. 61–64.
7. **Serafinavičius P., Daunys G.** Detection of Hand Position using 3-D Computer Vision // Electronics and Electrical Engineering. – Kaunas: Technologija, 2006. – No. 7(71). – P. 63–66.

Submitted for publication 2007 05 17

P. Serafinavičius. Estimating Characteristic Points of Human Body for Automatic Hand Pointing Gesture Recognition // Electronics and Electrical Engineering. – Kaunas: Technologija, 2007. – No. 8(80). – P. 83–86.

The pointing gesture recognition system based on stereo vision and implemented using open source computer vision library (OpenCV) is presented. The system is able to detect pointing gestures and to determine the 3D pointing direction in real-time. To obtain input features for gesture recognition, at first, detect a person's head and pointing hand, mark some feature points in detected regions of the head and pointing hand and then perform its tracking in 3D. Two cascades of boosted classifiers were used to detect the occurrence of the head and pointing hand. Results of experimental evaluation prove that the rate of correct detection is quite high, while tracking still needs improvements. Ill.4, bibl. 7 (in English; summaries in English, Russian and Lithuanian).

П. Серафинавичюс. Автоматическое нахождение точек человеческого тела для опознания показательных гестов // Электроника и электротехника. – Каунас: Технология, 2007. – № 8(80). – С. 83–86.

Используя OpenCV библиотеку компьютерного зрения отрытого кода, на основе стереозрения представлена система опознания показательных гестов руки. Система может обнаружить показательные гесты и установить их направление в трехмерном пространстве в реальное время. Для обнаружения головы и руки, используются две разные каскады класификаторов. В произведённом эксперименте подтверждается, что система достаточно хорошо находит голову и руку человека, но алгорифм слежения за характерными точками требует совершенствования. Ил. 4, библ. 7 (на английском языке; рефераты на английском, русском и литовском яз.).

P. Serafinavičius. Būdingųjų žmogaus kūno taškų radimas automatiniam rodomųjų gestų atpažinimui // Elektronika ir elektrotechnika. – Kaunas: Technologija, 2007. – Nr. 8(80). – P. 83–86.

Pateikiama rodomųjų rankos gestų atpažinimo sistema, sukurta stereoregos pagrindu naudojant atvirojo kodo kompiuterinės regos biblioteką (OpenCV). Sistema gali aptikti rodomuosius gestus ir nustatyti rodymo kryptį trimatėje erdvėje realiu laiku. Pirmiausia vaizde aptinkama žmogaus galva, po to – rodančioji ranka, pažymimi būdingieji taškai, kurie sekami trimatėje erdvėje. Galvai ir rankai aptikti naudojamos dvi skirtingos klasifikatorių kaskados. Atliktas eksperimentas patvirtina, kad sistema gana gerai aptinka žmogaus galvą ir rankas, bet būdingųjų taškų sekimo algoritmas dar yra tobulintinas. Il. 4, bibl. 7 (anglų kalba; santraukos anglų, rusų ir lietuvių k.).