Finger Tracking for Gestural Interaction in Mobile Devices

Matti Matilainen[1], Jari Hannuksela[1], and Lixin Fan[2]

[1] University of Oulu,
Center for Machine Vision Research
Department of Computer Science and Engineering
P.O. Box 4500
FI-90014 University of Oulu, Finland
{matti.matilainen,jari.hannuksela}@ee.oulu.fi
http://www.cse.oulu.fi/CMV
[2] Nokia Research Center Tampere,
Visiokatu 3
33720 Tampere, Finland
lixin.fan@nokia.com
http://research.nokia.com

Abstract. In this paper we propose a finger tracking system that is suitable for gesture recognition in mobile devices. The initialisation of the system does not require the use of any I/O devices. The user covers the camera lense with his or her hand and then takes it to the operating distance. The statistical models used for hand segmentation are initialised from the first frames after the hand is removed from the lense. The hand segmentation does not need to be perfect because we do not use the hand contour in the recognition. In our method the fingertips are found using template matching. However, the template matching produces false detections. These false detections are pruned by searching a path from the fingertip to the estimated hand centre and discarding the paths that do not meet a predefined criteria. We evaluate the performance of the method against a fingertip detector proposed by Baldauf et al. [2] by using seven test subjects who initialise the system and then wave their hand in front of the camera. In testing we use one handheld USB camera that matches the image quality of most recent front cameras in mobile phones.

Keywords: computer vision, finger tracking, gesture recognition.

1 Introduction

Hand and finger tracking is an important preprocessing step in hand gesture based user interfaces. The finger tracker persented in this paper is going to be used in a multimodal user interface in a mobile device together with a touch screen. The multimodal user interface provides different functionalities depending on the distance between the hand and the device.

J.-K. Kämäräinen and M. Koskela (Eds.): SCIA 2013, LNCS 7944, pp. 329–338, 2013.
© Springer-Verlag Berlin Heidelberg 2013

The recent and most successful solutions to the problem of hand tracking have been either active systems (Kinect [15][8] or time-of-flight [13]) or passive multi camera systems [14] that produce a 3D point presentation of the hand. While these methods are very attractive for hand tracking, the imaging hardware required is not currently available in mobile devices.

Hand tracking can be done by using a single camera. Hand is not a rigid object. The tracker must be able to learn new poses for the hand online. The Flock of Features tracker [12] and TLD tracker [11], can be used to track any non-rigid objects. The simplest way to track the hand is the frame differencing method [10]. However, rame differencing needs a fixed camera and non-changin lighting to perform well. Colour information is often used to segment the hand. The hand orientation can be estimated by fitting an ellipse to the colour segmentation result. With the estimate the hand orientation can be normalized to an upright position. It is also possible to use template matching for gray-scale images to discriminate between hand poses [9]. After normalising the hand pose the fingertips can be detected by projecting the rest of the hand to the x-coordinate axis and finding the local peaks from the projection [3]. Template matching has also been used in a gesture controlled typing interface [17]. Another approach to the finger tracking is to find the first gradient peak (starting from the upper left edge of the image moving towards the lower right corner) from the red channel of a YCrCb colour space image [7]. A similar approach is presented in [1]. These methods can only detect one finger and its usefulness in a user interface utilizing gestural recognition is limited. It is also possible to use infrared sensors for the hand segmentation [16].

In this paper we propose methods for fingertip detection and finger tracking that use a pathfinding algorithm. The tracker works with a single low quality hand held camera. We also propose a novel way of initialising the tracker without any other I/O devices than the front camera of the mobile device. We compare our finger tracker against a modified version of fingertip detector presented in [2] by Baldauf et al. and show that our method performs more robustly when tested in an uncontrolled environment where the hand is very difficult to segment from the image sequence.

2 Proposed Finger Tracking System

The finger tracking system presented in this paper is suitable for mobile devices. The image processing operations used are computationally efficient and the system can be initialised using only one hand (while the other hand holds the device). Figure 1 presents the control flow of our system.

2.1 Initialisation

First when the program is started the statistical models used for the background in hand segmentation are updated with every input image. When the user wants to start using hand gestures to control the program he or she covers the camera

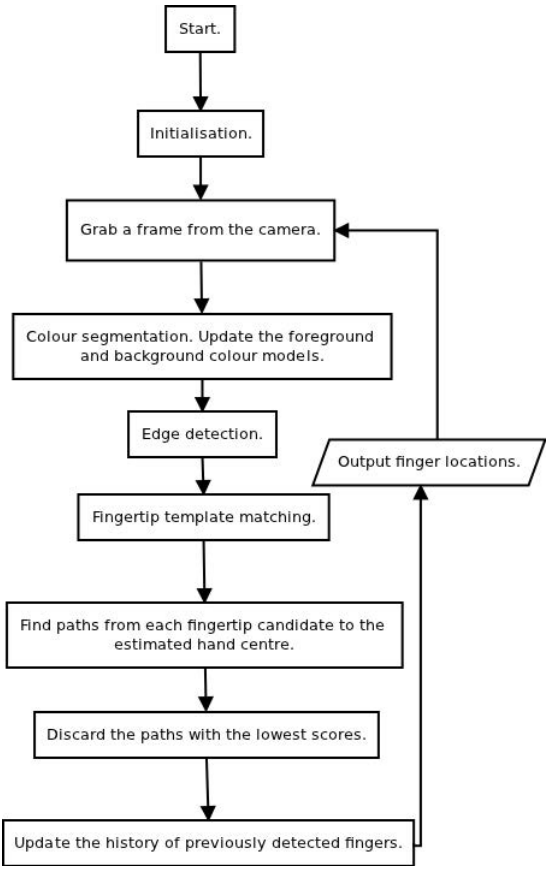

Fig. 1. The flowchart of the finger tracking system

lense with his or her hand or a finger and then takes the hand to the operating distance of $d = [20, 70]cm$ from the camera. The lense is considered covered when the sum over all the pixels in the input image is below a threshold. When the hand moves from the lense to the operating distance it is at the centre of the image for a few frames (we used the two first frames in our tests). The pixels at the centre of these images are used to initialise the models for the hand. When the models are initialised this way, the user does not need to use any other I/O devices than the front camera to initialise the tracker. The tracker can be reinitialized by covering the lense again with the hand or a finger.

2.2 Finger Detection

We assume that the palm faces the camera and that the hand is in an upright position. The hand is allowed to pivot 45 degrees in both directions. That is the limit that one can turn his or her hand with ease. These constraints are used

to reduce the computational cost of the method. For each of the input images, unless the user wants to re-initialise the tracker, the following image processing operations are performed: hand segmentation, edge detection and template matching. The hand segmentation is performed using colour segmentation. After the colour segmentation the colour models for the foreground and background are updated. We used the Canny method for edge detection [4]. The shortest line segments were removed from the edge detection result image because they are mostly noise. Template matching is performed in three different scales using a circular template where the bottom quarter is removed.

After the image processing is done, the hand centre point is estimated from the binary colour segmentation result image using image moments. Then a path is traced from each fingertip candidate to the hand centre. The pathfinding method we used works like a greedy optimisation algorithm. Starting from the fingertip candidate point, line segments with constant length and varying angle are aligned to the starting point. Then a score is given for each of the line segments as follows:

$$s = w_0 * s_{col} + w_1 * s_{dist} + w_2 * s_{edge} + w_3 * s_{mot} \quad (1)$$

where s_{col} is the amount of colour segmented pixels under the line segment, s_{dist} is the euclidean distance that this line segment gets the line closer to the hand centre, s_{edge} is the number of edge pixels that are crossed by the line segment and s_{mot} is the value in the Motion History Image (MHI) where the previously detected fingers are stored. The constants w_0,w_1,w_2 and w_3 set the weights for each of the scores. The constants are defined so that higher score means travelling through a lot of colour segmented foreground pixels, getting the line closer to the hand centre and trying not to cross any edges (the borders of the fingers are usually detected by the edge detection so trying to not to cross these borders keeps the line segment inside the finger) and staying where the previously detected fingers were. The segment with the highest score is selected and then the next search is started at the end point of the last line segment. The search ends when the hand centre is either found or passed by. Paths that have originated from a real fingertip have a high score while false detections have a low score. The paths with a low score are thresholded away and also, the paths that are too close to each other are combined to one. The pathfinding is illustrated in the Figure 2. In this Figure there are only five line segments sampled at five points for clarity. In the tests we used twenty 30 pixels long line segments sampled at six points.

Because we assume that the hand is in an upright position, the search has to be made only downwards and it is stopped when it passes the hand centre point. The resulting finger detections are saved to a MHI. This image is used during the next iteration of the program.

2.3 Hand Segmentation

The first step in the process of detecting the fingers is hand segmentation. For hand segmentation we tested colour segmentation using RGB colour space

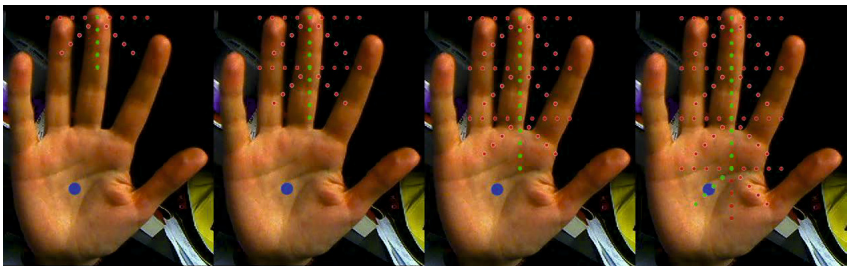

Fig. 2. Illustration of the pathfinding. The blue dot is the hand centre estimated from the colour segmented input frame. Starting from the image on the left, one set of line segments is laid over the hand and the chosen line segment is coloured green. The next three images are the three following iterations of the pathfinding. The search ends when the hand centre is found.

histogram with a varying number of bins and a tracking method by Collins [6]. Both methods can be initialised using our framework.

RGB Histogram. We use a three dimensional histogram in RGB space for both the foreground and the background. The sum over all the histogram bins is normalized to 1. The classification of a pixel $I(j,i)$ in the input image using foreground histogram ω_f and background histogram ω_b is performed as follows:

$$\begin{aligned} &\forall(y,x) \in I_s \\ &I_s(y,x) = \begin{cases} 1 \text{ ,if } p(I(y,x)|\omega_f) \geq p(I(y,x)|\omega_b) \\ 0 \text{ , otherwise} \end{cases} \end{aligned} \tag{2}$$

where the values 1 and 0 denote the foreground and background respectively in the result binary image I_s.

The foreground and background histograms are updated after each segmentation. Each pixel classified as foreground is used to update the foreground histogram ω_f and the rest of the pixels are used to update the background histogram ω_b. The histograms are updated using the following formula:

$$\begin{aligned} &\forall(i,j,k) \in \omega \\ &\omega(i,j,k) = \omega(i,j,k) * \alpha + \omega_n(i,j,k) * (N/N_n) * (1-\alpha) \end{aligned} \tag{3}$$

where ω is the original histogram, ω_n is the new histogram computed from the current frame, N andN_n are the sums over all the bins in original and new histograms respectively and α is the parameter that controls how much the new histogram contributes to the result histogram.

We ran the tests with the RGB histogram three times with 8, 9 and 10 histogram bins for each channel resulting in a histogram of size 8^3, 9^3 and 10^3 respectively. In every test we used $\alpha = 0.9$.

Online Selection of Discriminative Features. The tracking method by Collins [6] evaluates multiple feature spaces adaptively to get the best discrimination between the target object and background. The feature spaces used are the linear combinations (with varying weights) of the R, G and B channels of a colour image. For every frame the feature spaces are ranked using the previous segmentation. The best feature spaces are used to produce likelihood maps of the new frame. The location of the object is found using the mean shift process [5].

2.4 Fingertip Detection

The fingertips are detected from the edge image using template matching method in three different scales. The template we used is a circle where the bottom quarter is removed. This template gives high output inside the fingertip. The fingertip locations can be found by thresholding the image where the fingertip template was applied. The Figure 3 illustrates four frames from the Sequence 3 where the detected fingertips are marked with blue dots. After the detection, the fingertips are clustered and the centres of the clusters are used as seed points for the pathfinding algorithm.

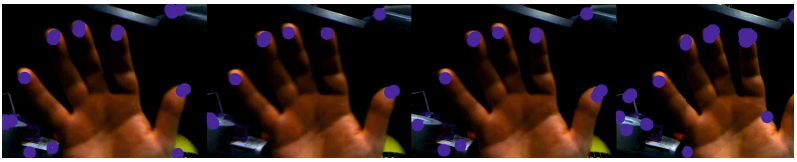

Fig. 3. Example frames from one of the test sequences. The fingertip candidates found with the template matching method are marked with the blue dots.

3 Experiments

The weight parameters w_0, w_1, w_2 and w_3 are the adjustable parameters of our method. The parameters used in our experiments were found by exhaustively searching through the parameter space. As a training set we used a sequence of 450 frames where the fingers were labelled in the last 351 frames. The first 99 frames included only background and the initialisation.

Image size for both the training sequence and the testing sequences was 640 x 480 pixels.

3.1 Reference Method

We compared our finger tracker against the fingertip detector presented by Baldauf et al. [2]. To segment the hand the same two colour segmentation methods were used as in our own tracker and also the same initialisation method was used.

Because the method by Baldauf et al. uses the hand contour, the colour segmentation had to be processed with morphological operations to connect the disconnected parts of the hand and to remove noise.

We made two simple modifications to the method. The hand contour was low pass filtered. The local maximum points can be found more robustly from the filtered countour. Also, when there were more than five local maximums, only the five points farthest from the hand center were used.

3.2 Test Setup

The experiments were made with a testing set of seven sequences. Each sequence was captured with a different user. The users were both male and female. The users held the camera in their left hand and covered the camera lense with their right hand to initialize the finger tracker and then took the hand to the operating distance of $d = [20, 70]cm$ from the camera. In the sequences there are a few seconds of only the background and then after the initialization on average 92 frames where the hand moves around the image. From these sequences the frames where the hand is shown completely each of the fingers were labelled with a bounding box to establish the ground truth.

3.3 Discussion

The average finger detection rates for both finger trackers using each of the segmentation methods are shown in Figure 4. The finger numbers from 1 to 5 are the fingers from little finger to thumb respectively. False detections per frame are shown in Figure 5.

As seen in the Figure 4 the method of Baldauf et al. finds the longer fingers in the middle more robustly compared to the fingers 1 and 5. The tips of the longer fingers are easier to detect because they are farther away from the hand center.

Our tracker has the lowest number of detections for the fingers 1 and 2. In the testing data these fingers were often very close to each other and they were combined to one in the fingertip detection phase.

As seen in Figure 5 the amount of false positives per frame is significantly lower with our method compared to the reference method. When the detection rates are taken into account, the overall accuracy of our tracker is higher. For example, when the RGB histogram with 8^3 histogram bins is used for segmentation, the average detection rates are for the reference method and our method are 0.51 and 0.62 respectively. The corresponding false positive rates are 0.44 and 0.02 respectively.

From the Figure 4 we can see that when using our tracker the performance is good using each of the segmentation methods. Because the computational complexity of the RGB histogram is lower than the Collins' method, the RGB histogram with 8^3 bins is the preferable segmentation method.

The Figure 6 shows a frame where the hand segmentation is performed and postprocessed with morphological close operation with different sized

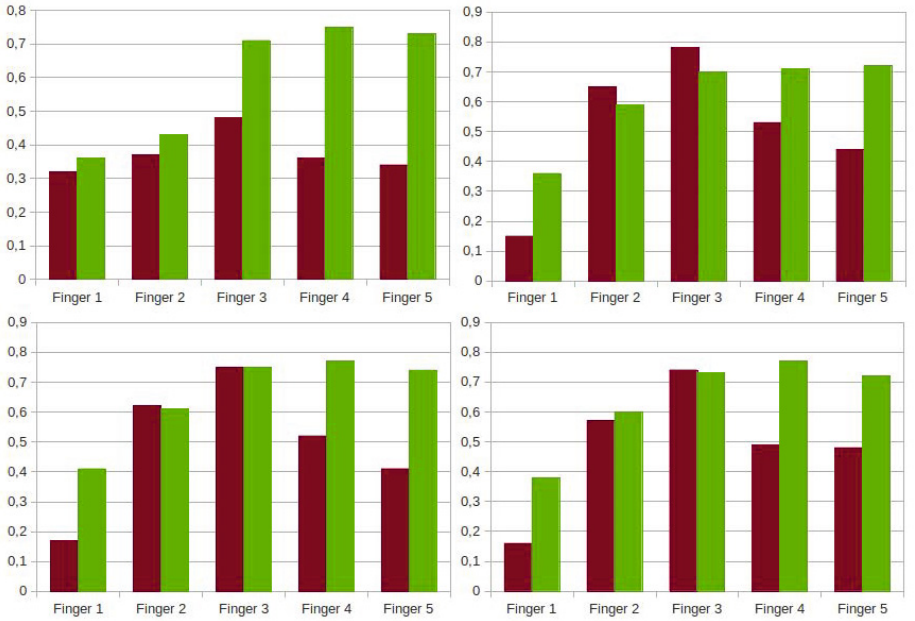

Fig. 4. The average finger detection rates over all the test sequences using each of the hand segmentation methods. The charts from the upper left corner to the lower right corner correspond to the Collins and RGB histogram with 8^3, 9^3 and 10^3 bins respectively. The results of the method by Baldauf et al. are marked in red and the results of our method are marked in green.

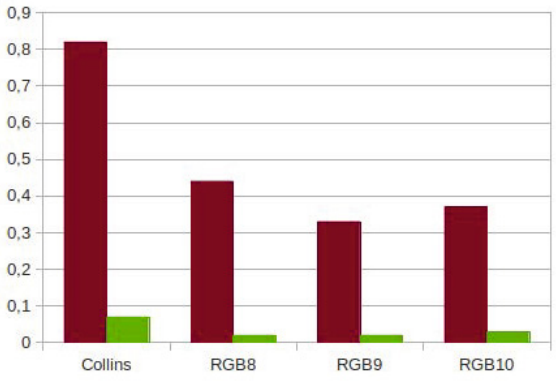

Fig. 5. The amount of false positives per frame using each of the hand segmentation methods. The results of the method by Baldauf et al. are marked in red and the results of our method are marked in green.

structuring elements. The morphological operations used improve the result but it is very difficult to set the size of the structuring element so that the separated components are connected without connecting the fingers to one big blob. In the second image, the fingertips are disconnected from the hand. In the third image each fingertip is conected to the hand but the three fingers in the middle contribute to only one local maximum peak in the resulting hand contour. The reference method [2] we used cannot find the fingers from this kind of data while our own method does not need the fingers to be segmented perfectly.

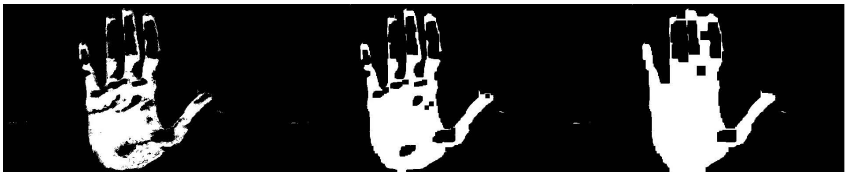

Fig. 6. Example of the hand segmentation after morphological close operation with different structuring element sizes. The first image is the original segmentation, in the next two images, a morphological close operation is perfrormed using rectangular structuring elements with width 10 and 20 pixels respectively.

Prototype of our finger tracker is currently implemented on a desktop PC. The computational cost of the pathfinding process on our prototype implementation is the same as the Gaussian filtering and Canny edge detection in the preprocessing step. It is feasible to say that the proposed tracker can be implemented on a modern mobile device.

4 Conclusion

In this paper we have proposed methods for fingertip detection and finger tracking that use a pathfinding algorithm. These methods are suitable for mobile devices. The imaging hardware required is already available in the recent mobile devices.

Easiness of use is important in practical mobile usage scenarios. The initialisation method proposed does not require the use of any other I/O devices than the front camera and it can be done with only one hand while the other hand holds the device.

The reference method [2] used requires perfect segmentation of the hand in order to detect the fingertips. Perfect hand segmentation is very hard to achieve without active imaging sensors. Our method does not use the hand contour and it can detect the fingers from a low quality image data. Experiments show that our method works more robustly than the reference method.

References

1. An, J.-H., Min, J.-H., Hong, K.-S.: Finger Gesture-Based Mobile User Interface Using a Rear-facing Camera. In: Park, J.J., Yang, L.T., Lee, C. (eds.) FutureTech 2011, Part II. CCIS, vol. 185, pp. 230–237. Springer, Heidelberg (2011)
2. Baldauf, M., Zambanini, S., Frölich, P., Reichl, P.: Markerless Visual Fingertip Detection for Natural Mobile Device Interaction. In: Proceedings of the 13th International Conference on Human Computer Interaction with Mobile Devices and Services, MobileHCI, pp. 539–544 (2011)
3. Ben Jmaa, A., Mahdi, W., Ben Hmadou, A.: A new approach for digit recognition based on hand gesture analysis. International Journal of Computer Science and Information Security, 108–115 (2009)
4. Canny, J.: A Computational Approach to Edge Detection. IEEE Transactions on Pattern Analysis and Machine Intelligence, 679–698 (1986)
5. Cheng, Y.: Mean shift, mode seeking, and clustering. IEEE Transactions on Pattern Analysis and Machine Intelligence, 790–799 (1995)
6. Collins, R.T., Zhou, X., Teh, S.K.: An Open Source Tracking Testbed and Evaluation Web Site. In: IEEE International Workshop on Performance Evaluation of Tracking and Surveillance (2005)
7. Davis, J.E., Gallo, O., Arteaga, S.M.: A camera-based pointing interface for mobile devices. In: IEEE International Conference on Image Processing, pp. 1420–1423 (2008)
8. Frati, V., Prattichizzo, D.: Using Kinect for hand tracking and rendering in wearable haptics. In: IEEE World Haptics Conference 2011, pp. 317–321 (2011)
9. Hasanuzzaman, M., Ampornaramveth, V., Zhang, T., Bhuyian, M.A., Shirai, Y., Ueno, H.: Real-time Vision-based Gesture Recognition for Human Robot Interaction. In: Proceedings of the IEEE International Conference on Robotics and Biomimetics, pp. 413–418 (2004)
10. Henrysson, A., Marshall, J., Billinghurst, M.: Experiments in 3d interaction for mobile phone AR. In: International Conference on Computer Graphics and Interactive Techniques in Australia and Southeast Asia, pp. 187–194 (2007)
11. Kalal, Z., Mikolajczyk, K., Matas, J.: Tracking-Learning-Detection. IEEE Transactions on Pattern Analysis and Machine Intelligence 6(1), 3789–3792 (2010)
12. Kölsch, M., Turk, M.: Fast 2D Hand Tracking with Flocks of Features and Multi-Cue Integration. In: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops, p. 158 (2004)
13. Li, Z., Jarvis, R.: Real time Hand Gesture Recognition using a Range Camera. In: Australasian Conference on Robotics and Automation (2009)
14. Malik, S., Laszlo, J.: Visual touchpad: a two-handed gestural input device. In: Proceedings of the 6th International Conference on Multimodal Interfaces, pp. 289–296 (2004)
15. Oikonomidis, I., Kyriazis, N., Argyros, A.: Efficient model-based 3D tracking of hand articulations using Kinect. In: Proceedings of the British Machine Vision Conference, pp. 101.1–101.11 (2011)
16. Oka, K., Sato, Y.: Real-Time Fingertip Tracking and Gesture Recognition. In: Computer Graphics and Applications, pp. 64–71 (2002)
17. Terajima, K., Komuro, T., Ishikawa, M.: Fast finger tracking system for in-air typing interface. In: Extended Abstracts on Human Factors in Computing Systems, pp. 3739–3744 (2009)