A Face Tracking Algorithm for User Interaction in Mobile Devices

Abdullah Bulbul
Dep. of Computer Engineering
Bilkent University
Ankara, Turkey
bulbul@cs.bilkent.edu.tr

Zeynep Cipiloglu
Dep. of Computer Engineering
Bilkent University
Ankara, Turkey
zeynep@cs.bilkent.edu.tr

Tolga Capin
Dep. of Computer Engineering
Bilkent University
Ankara, Turkey
tcapin@cs.bilkent.edu.tr

Abstract—**A new face tracking algorithm, and a human-computer interaction technique based on this algorithm, are proposed for use on mobile devices. The face tracking algorithm considers the limitations of mobile use case – constrained computational resources and varying environmental conditions. The solution is based on color comparisons and works on images gathered from the front camera of a device. The face tracking system generates 2D face position as an output that can be used for controlling different applications. Two of such applications are also presented in this work; the first example uses face position to determine the viewpoint, and the second example enables an intuitive way of browsing large images.**

Keywords-face tracking; human computer interaction; mobile devices

I. INTRODUCTION

Recent advances in computer and mobile technology have made possible the investigation of new and efficient interaction techniques. As a result, gesture recognition, motion and object tracking, perceptual user interfaces are getting popular. As the input modalities are limited on a mobile device, the available resources should be used in an efficient and effective way. Keypad, joystick, stylus, camera and sensors are some of the input sources that are commonly used for interaction.

In this paper, we focus on the usage of camera input for interaction with the device. We propose to use the built-in camera to track the head position. Thus, our approach does not require any extra special hardware. In this method, we track the head movements by comparing the face positions through the neighboring frames.

By using this method, different gestures can be defined for different applications. Besides, face tracking enables enhancing the depth effect in the applications by supplying motion parallax. Motion parallax refers to the depth information provided by the optic flow of the visual field due to the sideways movement of the viewer [1].

In order to present the usage of this face tracking system, we have implemented two sample applications. The first application demonstrates the motion parallax effect where the face tracking system controls the camera position in a 3D application. The second application is a picture viewing tool in which the scroll events are controlled by the head movements in an intuitive way.

The Problem: Since the interaction method will run on a mobile device, and share the computational resources with the actual application, the solution should be real-time and should not consume much computational resources. Another challenging point of the mobility is highly varying camera input data, such as color, contrast, luminance, and the background and ambient properties.

II. RELATED WORK

In the literature, face detection, face tracking and other computer vision techniques have been heavily used for Human Computer Interaction. There are various interaction approaches based on egomotion, whose concern is the self movement of the device. Barnard et al. propose a vision based user interface for mobile devices where the camera input is used for egomotion calculation [2]. In this method, Hidden Markov Models are used to model the motion feature sequences. Then, the results are filtered by a likelihood ratio and the entropy of the sequence. Capin et al. suggest another camera based interaction solution in which incoming video is used to estimate the phone motion and then the physical motion of the phone is mapped to scroll or view direction according to the application. In this method, a feature based tracking algorithm is performed to analyze the motion of corner-like features between the consequent frames [3], [4]. Another feature based approach for controlling user interfaces on mobile devices is developed by Hannuksela et al. where motion analysis is performed using a sparse set of features and a Kalman filter is applied to smooth motion trajectories [5].

Finger tracking is another problem that has been addressed for user interaction on mobile devices. Hannuksela et al. combine Kalman filter and the Expectation Maximization (EM) algorithm for estimating the background and finger motions so as to deal with the changeable background conditions [6].

Face tracking is an important vision based tool to be used in human computer interaction. In these solutions, motion of the face is translated into a set of commands to interact with the computer. In order to track the face position, first step is to

978-0-7695-3791-7/09 $26.00 © 2009 IEEE
DOI 10.1109/CW.2009.9

detect the face in an image. Face detection algorithms are generally categorized as feature based and image based methods [7]. In feature based methods, facial features such as nose, eyes and lips are identified by performing geometrical analysis on their locations, proportions and sizes [8]. Color, motion and edges are the mostly used properties to extract the facial features. The second type, which contains image-based methods, is based on scanning the image through a window to find the face candidates. The methods in this category generally use template matching, support vector machines (SVM) or neural networks [9], [10]. Image based methods are popular in Computer Vision due to their robustness.

One example of feature based approaches for face tracking is developed by Bradski, where face tracking is used to control computer games and 3D navigation [8]. Flying over a 3D city model controlled by face movements is one example of these applications. The method extends a color based tracking method, called mean shift algorithm, which operates on color probability distributions. Hunke et al. proposes an image based neural networks algorithm that performs face tracking for human computer interaction [9]. Blink and Click project is another application, where traditional mouse is replaced by human facial actions [10]. For instance, nose location is used as the mouse pointer and left/right eye blinks correspond to left/right mouse clicks, in this system. This algorithm is a combination of feature based and image based approaches.

Several other researchers have proposed vision based solutions for HCI. We refer the reader to [11] for a further survey of these techniques.

All face tracking methods that have been referred are designed to operate on desktop or high-end mobile systems. In addition, these computer vision algorithms generally perform multiple passes over the image and require complex machine learning processes. Hence, they are computationally heavy to be applied on smart phone mobile devices, and there is a need for less expensive and lightweight solutions for use in combination with mobile applications.

III. Our Approach

As a human computer interaction solution, we are proposing a head tracking system which uses the input from the front camera to track the head position. In this system, we can make the assumptions that there is always a user in front of the display and there is only one user most of the time, most likely scenario for mobile devices. We target a lightweight face tracking algorithm that is suitable to use in mobile devices which has limitations in terms of CPU power.

Our camera based face tracking system can be used as an interaction technique for many applications in mobile devices. For instance, it is very suitable to be used for scrolling the scene or it can be used to render the scene according to the user's position. Two sample applications of the face tracking system are shown in the Applications section.

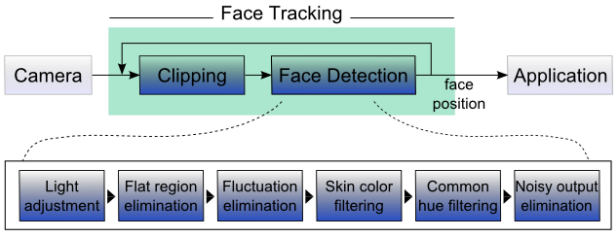

Figure 1: Overview of the System

An overview of our head tracking system is summarized in Figure 1. In this system, image is captured from the front camera of the device, a suitable region of the image is selected (clipping phase) according to the previous face position and the face is detected in the selected region of the image. According to the position of the head, an action specific to the application is performed, such as scrolling the picture to the right when the head moves to the left in a picture viewer application. We describe the details of the solution below.

A. Considerations and Limitations

First, we list our assumptions for our tracking method below:

- There is a single face on the image most of the time.
- The face covers most of the image space when it is fully in view. According to our observations on a horizontal slice of the image, the proportion of the image that belongs to face to the whole image is slightly above 1/2, and the face covers at least one third of the whole image.
- In a mobile environment, light is a highly varying factor, whereas the hue value of the face remains stable throughout the frames.
- Hue values belonging to skin colors of different people fall in a particular range.
- The RGB values will not fluctuate too much while going on a line on the face.
- Background changes frame to frame on a mobile environment, particularly while the user on the move.
- Background may be smoother than the face as well as the opposite.

The limitations of mobile devices are also important for our algorithm design. Most importantly, mobile devices have lower CPU powers relative to computers, and resolution of the captured images from the front camera is not assumed high. Our aim is keeping the face detected in real-time while background properties change. On the other hand, existing computer vision techniques require many passes on the image and have inferior results in mobile case.

Upon these design considerations, our algorithm is based on the following ideas:

- Scan not the whole image but a limited part of it. For example, only a horizontal and a vertical line that includes face.
- The line to scan may be determined according to previous position of the face.
- An adjustable solution is preferred that keeps performance and accuracy balanced.
- Complex vision techniques are not appropriate in our case.
- On start assume that the face is close to the center of the image.

B. Algorithm

We have developed an algorithm that only uses color comparisons, rather than geometric properties of the facial features in order to avoid high computations. When selecting color space to use in the algorithm we generally avoid using the RGB space since light changes significantly affect the R, G, B values. Therefore, we mostly prefer using the HSL color space as the hue value is the stable parameter against the varying environmental conditions.

The dark boxes in Figure 1 show the steps of our algorithm. In the clipping phase, a suitable region of the captured image is selected. Then the clipped image is given as an input to the face detection phase. The details of these phases are explained in the following subsections.

1) Clipping

Since scanning through the whole image is costly, we have designed an algorithm in which all calculations are performed upon a suitable region of the image. We found this suitable region using the face position on the previous frame. While determining the region that will be scanned:

- A horizontal scan detects the x position of the face and a vertical scan detects the y position.
- Which horizontal line or lines to use in calculations is determined by the y position of the face in the previous frame, and vice versa.
- Number of lines to scan can be increased for more accurate results. This allows a trade-off between accuracy and performance.

2) Face Detection

After determining the horizontal and vertical lines to be scanned in the clipping phase, for each line to scan, the following steps are applied:

a) Adjusting the Average Light

In a mobile environment the light is highly varying: The device can be used under sunlight, indoor environments, in a dark environment or in any different light conditions. Hence, the light value on the image may be in a narrow interval, such that in a dark environment all pixels on the image have low light values. A wider distribution of the color values is more preferable to work on. Thus, our algorithm starts with a light adjustment step. The light values are gathered from the HSL color space and after refinement of the L values new HSL values are replaced with the old ones on the image (1).

$$L_{p'} = \begin{cases} L_p \times \frac{Aim}{Avg'}, & L_p < Avg \\ 255 - (255 - L_p) \times \frac{255 - Aim}{255 - Avg'}, & L_p \geq Avg \end{cases} \quad (1)$$

Where $L_{p'}$ is the new light value of the pixel p, L_p is the old value, Avg is the average light value of the image, Aim is the target average light of the image and the light value is scaled between 0-255. The purpose of this adjustment is setting the average light of the image to a central value and widening the light distribution. (Figure 2)

b) Eliminating flat regions

This step in our algorithm eliminates the regions whose light values do not change for a long sequence of pixels because of the curvature on a human face. Pixels corresponding to a flat shape, for example a wall, have light values that are nearly the same since each part of a flat surface gets the light in a similar angle. On the other hand, human face is not flat and has a curvature; thus, the light values on corresponding frames are expected to differ slightly on a face. This part of the algorithm sums up the differences in a sequence of pixels and eliminates them if the change in light values is below a threshold (2).

$$\sum_{i \in n(i)} |L_i'| < th \Rightarrow eliminate \quad (2)$$

Where, $n(i)$ is the neighborhood of pixel i and th is the threshold value. We have empirically chosen a threshold of 25 for 30 consequent pixels (in 0-255 light value scale) (Figure 3).

c) Eliminating regions that have fluctuations

The next step of the algorithm is the elimination of the fluctuating parts. On the image, there may be a few textured parts that have a pattern of repeating colors or a non-regular region that have very different colors. These parts cannot be on a face since colors on a face generally changes towards a single direction. There is not much fluctuation of the color values on a face. Using R, G and B together is more suitable than using H, S and L since R, G and B are of similar types regarding the range and semantics; thus, we select using the RGB space in this step. Determining the fluctuating parts is done by taking the second order derivatives of the RGB values. Second order derivatives for R, G and B components are separately calculated and the magnitudes of the derivatives are summed up. If the summation of the fluctuation is greater than a threshold in a line of consequent pixels this part is eliminated (3).

$$\sum_{i \in n(i)} |R_i''| + |G_i''| + |B_i''| > th \Rightarrow eliminate \quad (3)$$

Where, $n(i)$ is the neighborhood of pixel i and th is the threshold value. In our case, the threshold is 750 for 30 consequent pixels (in a 0-255 RGB scale). This part of the algorithm uses a similar approach to edge detection algorithms and can be seen as elimination of the parts that have many edges. (Figure 4)

d) Finding skin color

There are a number of attempts to detect the skin color most of which suggest very restrictive methods [12]. These restrictions are not appropriate for mobile environment since they cause most of the face to be eliminated in the varying environmental conditions. Therefore, we use a less restrictive method to detect the skin color.

Based on the observations in [13], blue component in a skin color varies in a wide range, and hence, blue component is not effective on the overall color as much as red component. According to the analytical assessments of [13], red-green ratio of the skin colors changes between 1:1 to 3:1. Based on these assessments, we eliminate the regions with color that cannot be a skin color. (Figure 5)

e) Finding mostly used hue

One assumption is that face generally covers a large region in the display in a mobile device. The proportion of the face to the whole is generally above 1/2 and at least 1/3. Based on this assumption, it can be thought that the mostly used hue value belongs to the face. For this purpose, we divide the hue space into equal sized clusters. In this way, for each cluster, we find the total number of pixels that fall into that range. The cluster with the maximum number gives the mostly used hue and we assume that this is the hue value of the face. Therefore, we can eliminate other regions in the image. Cluster size of 40 generally gives the best results. (Figure 6)

f) Eliminating noisy output

After all the steps explained previously, there still exist some regions that seem as face candidate but are very small to be a face indeed. Therefore, we apply a median filter to remove the noisy regions in the image as the last step. (Figure 7)

IV. APPLICATIONS

In our initial experiments, we have used SONY VAIO UX device. In this part we present two applications that we integrated our face tracker into, to show example usages of our interaction technique.

Camera Application:

As a sample application that uses our face tracking algorithm, a 3D application was implemented. In this application, a 3D scene is rendered from different viewpoints using an OpenGL perspective camera. The position of the camera is determined by the input from our face tracking system. In other words, the position of the user is used as analogous to the position of the camera and the movement of the camera is controlled by the movement of the head. By this way, the user can view different portions of the scene by only moving his head or his device and this enhances the depth perception by providing *motion parallax* cue of depth. Hence, our face tracking system provides an elegant solution for such an application as well.

Figure 2: Left: Original, Right: Light Adjusted

Figure 3: Left: Original, Right: Flat regions are eliminated.

Figure 4: Left: Original, Right: Fluctuations are eliminated.

Figure 5: Left: Original, Right: Regions that are not skin color are eliminated.

Figure 6: Left: Original, Right: Effect of mostly used hue

Figure 7: Left: Without noise elimination, Right: After noise elimination

Picture Viewer Application:

Displaying a large picture in a small display area has always been a problem of mobile devices. Lack of mouse makes the usage of scroll bars difficult. For this purpose, an intuitive interaction solution is required. We propose to use our face tracking system in order to handle the inputs for scrolling events. Therefore, we have implemented a system to view large pictures in a small display. The usage of this system is quite intuitive because it can be thought as looking through a window as a peephole metaphor. So for instance, when you move your head to the right, you can see the area on the left from the window, and vice versa. This usage is much easier in mobile devices when compared to desktop systems, since there is also an opportunity of moving the device as well as the head.

V. DISCUSSION

Generally the system works as expected in different environments and under different lightings although there are some situations in which the system works problematically. In order to evaluate the success of the face tracker, we do not need the system to find the exact face region; instead, tracking an approximate position is sufficient for interaction.

The system works appropriately in different brightness conditions, even in a relatively dark environment, user's face can be differentiated by the system to some extent. Figure 8 shows sample execution results under various light and background conditions.

One advantage of the system is its adjustable usage. Since all calculations are done line by line in a horizontal or vertical fashion, it is possible to set how many lines to consider. All of the image can be used in the calculations as well as only a single vertical and a single horizontal line of the image. That provides flexibility needed for mobile devices which have different CPU powers and camera resolutions.

According to our observations the system cannot be used appropriately in some conditions. Firstly, the flat wooden parts such as many of the furniture cannot be differentiated from the face easily, since they have similar hue values and they have texture that is not flat or fluctuating enough to be eliminated. Also, the system cannot work correctly when there is a white background that is illuminated by yellow or pink light which creates color properties similar to a face. In these cases, using only color properties is not sufficient. A possible solution for these cases can be extending the algorithm to consider the geometrical features such as the proportions of nose, eyes and lips etc., in addition to the color based calculations.

We believe that face tracking is an intuitive and effective way of interaction. Also the adjustable behavior of the proposed face tracking system can be a significant facility for mobile devices which have different processors. In the meantime, it is possible to obtain more accurate results by improving the face detection part of the algorithm.

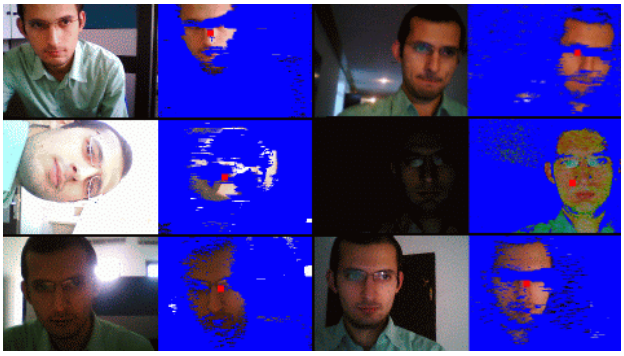

Figure 8: Results under different background conditions. First and third columns are the original images. Second and forth columns are the output of face detector: blue regions are the eliminated parts and the red point is the calculated face position.

VI. CONCLUSION

We have proposed a new face tracking based interaction method for mobile devices. The applied face tracking method is a color comparison based algorithm and designed to be appropriate for the limitations of the mobile devices and the highly varying mobile environment. The usage of the face tracking is presented with two example applications those simplify viewing images and 3D contents on mobile devices. The overall performance of the system is satisfactory although there are some problematic cases. Handling those situations also remains as a future work. Also, our face tracking algorithm which works on 2D can be extended to 3D by providing the information of how close the user to the device. A possible solution may be calculating the proportion of face pixels to the whole image in which an increase in this proportion means that user gets closer to the device.

ACKNOWLEDGMENT

We thank FP7-213349 All 3D Imaging Phone project for the hardware support.

REFERENCES

[1] P. Shirley. Fundamentals of Computer Graphics. A. K. Peters, Ltd., Natick, MA, USA, 2002.

[2] M. Barnard, J. Hannuksela, P. Sangi, and J. Heikkilä, "A Vision based Motion Interface for Mobile Phones," In: Proc. 5th International Conference on Computer Vision Systems (ICVS), Bielefeld, Germany, 2007.

[3] T. Capin, A. Haro, V. Setlur, and S. Wilkinson, "Camera-Based Virtual Environment Interaction on Mobile Devices," Lecture Notes in Computer Science, Vol. 4263/2006, October 2006, pp. 765-773, Springer, 9783540472421, Berlin.

[4] A. Haro, K. Mori, T. Capin, and S. Wilkinson, "Mobile camera-based user interaction," Proc. ICCV-HCI 2005, pages 79–89, October 2005.

[5] J. Hannuksela, P. Sangi, and J. Heikkilä, "A Vision-Based Approach for Controlling User Interfaces of Mobile Devices," In: Proc. IEEE Conference on Computer Vision and Pattern Recognition, Workshop on Vision for Human-Computer Interaction (V4HCI), 6 p., San Diego, CA, 2005.

[6] J. Hannuksela, S. Huttunen, P. Sangi, and J. Heikkilä, "Motion-based Finger Tracking for User Interaction with Mobile Phones," In: Proc. 4th European Conference on Visual Media Productio (CVMP), London, UK, 2007.

[7] B.K.L. Erik Hjelmas, "Face Detection: A Survey, Computer Vision and Image Understanding," vol. 3, no. 3, pp. 236-274, Sept. 2001.

[8] G. R. Bradski, “Computer vision face tracking for use in a perceptual user interface,” Intel Technology Journal, 1998.

[9] M. Hunke and A. Waibel, “Face locating and tracking for human-computer interaction”. In Proc. of the 28th Asilomar Conf. on Signals, Systems and Computers, pages 1277-1281, 1994.

[10] W. Siriluck, S. Kamolphiwong, T. Kamolphiwong, and S. Sae-Whong, “Blink and click,” In Proceedings of the 1st international Convention on Rehabilitation Engineering & Assistive Technology: in Conjunction with 1st Tan Tock Seng Hospital Neurorehabilitation Meeting (Singapore, April 23 - 26, 2007). i-CREATe '07. ACM, New York, NY, 43-46.

[11] A. Jaimes and N. Sebe, “Multimodal Human Computer Interaction: A Survey,” Proc. 11th IEEE Int'l Workshop Human Computer Interaction (HCI), 2005.

[12] V. Vezhnevets, V. Sazonov, and A. Andreeva, “A survey on pixel-based skin color detection techniques”, GRAPHICON03, pp. 85–92, 2003.

[13] J. Brand and J. Mason, “A Comparative Assessment of Three Approaches to Pixel-Level Human Skin Detection,” Proc. IEEE Int"l Conf. Pattern Recognition, vol. 1, pp. 1056-1059, Sept. 2000.