Face Tracking for Spatially Aware Mobile User Interfaces

Jari Hannuksela, Pekka Sangi, Markus Turtinen, and Janne Heikkilä

Machine Vision Group, Infotech Oulu
Department of Electrical and Information Engineering
P.O. Box 4500, FIN-90014 University of Oulu, Finland
jari.hannuksela@ee.oulu.fi

Abstract. This paper introduces a new face tracking approach for controlling user interfaces in hand-held mobile devices. The proposed method detects the face and the eyes of the user by employing a method based on local texture features and boosting. An extended Kalman filter combines local motion features extracted from the face region and the detected eye positions to estimate the 3-D position and orientation of the camera with respect to the face. The camera position is used as an input for the spatially aware user interface. Experimental results on real image sequences captured with a camera-equipped mobile phone validate the feasibility of the method.

Keywords: facial feature extraction, motion analysis, pose estimation.

1 Introduction

Modern mobile communication devices are attractive platforms for various new applications as their multimedia capabilities are improving together with their computational resources. As more and more applications are available for these devices, the user interfaces are becoming overloaded, potentially confusing the user who needs to learn to use each invidual application. Particularly, the small displays set the primary restrictions for the usability. A viable alternative to improve user interaction with mobile terminals are spatially aware displays [1]. The solution is to provide a window on a larger virtual workspace where the user can access more information by moving the device around.

Based on this approach, Yee [2] presented a peephole display technique to control, for example, the 3-D image viewer and the 3-D calendar applications on a hand-held computer using several additional sensors providing positional information. Recent work on small sized devices such as mobile phones has focused on employing motion data measured using the accelerometer [3] or the camera [4] for this same purpose. The motion can be integrated in order to obtain a positional input for controlling the spatially aware display. However, when integrating motion data, errors typically accumulate over time. Therefore, motion based techniques should be used only when a precise position is not mandatory. Our idea is to use a built-in camera as a sensor to determine the position and orientation (pose) of the device with respect to the user face and utilize this information for browsing the virtual workspace of the mobile terminal in its display. The pose is measured continuously, which means that the user interface is spatially aware of the user all the time.

© Springer-Verlag Berlin Heidelberg 2008

A number of face and head pose estimation methods have been proposed mainly for personal computers. Prior research related to our method include early work of Azarbayejejani et al. [5]. They used recursive estimation to recover a head structure and motion from image sequences of rigid motion. The extended Kalman filter (EKF) was applied to track distinct features corresponding to the corners of the eyes and nostrils. Black and Yacoob [6] developed a regularized optical flow method that used an eight parameter 2-D planar model. Based on their work, Basu et al. [7] presented a system to track heads with a large amount of head motion. Instead of using a planar model they used a 3-D ellipsoidal model of the head. The algorithm starts by computing optical flow for the image and then it estimates the 3-D motion of the rigid model. The system was not real-time due to slow computation of the optical flow. La Cascia et al. [8] presented an approach using a texture mapped 3-D rigid surface model for the head and formulated tracking as an image registration problem.

In this paper, we propose a method for 3-D face tracking that can be used to control spatially aware user interfaces of mobile devices. Unlike many other methods proposed in the literature, the low computational cost of our method makes it practical for mobile platforms where high computational resources are not available. Also, we wish to emphasise the point here that our applications differs from the usual case since the device is moved with respect to the face. The proposed system consists of two stages. In the initialization stage, the users face and eyes are detected automatically. Whenever a face is detected the tracking is started. During tracking, an extended Kalman filter estimates the camera pose utilizing a novel combination of motion features and eye positions detected from the face region.

2 Facial Feature Extraction

The automatic facial feature extraction starts with a face detection process. After the face of the user is found, the eyes are searched for within the face region. We use the detected eye positions to initialise the face model for the tracker. In the tracking phase, we detect eyes using the same method and also extract motion features which provide information about local motion for different parts of the facial region.

2.1 Face Detection

Our object detection approach uses efficient gray-scale invariant texture features [9] and boosting [10]. The solution is based on the local binary pattern (LBP) texture methodology [9] for facial represenation. The AdaBoost learning is then applied to find the most discriminative features for distinguishing the face patterns from the background.

The LBP features can detect local texture primitives such as spots, edges or corners from the images [9]. They have been found to be very discriminative in facial image analysis [11]. Similar to the approach of Hadid et al. [12], we use the facial representation where the LBP feature histograms are separately computed over the sparse set of local image regions and the whole face area. These are then concatenated for creating the final face descriptor. Fig. 1 illustrates the idea of LBP based facial representation.

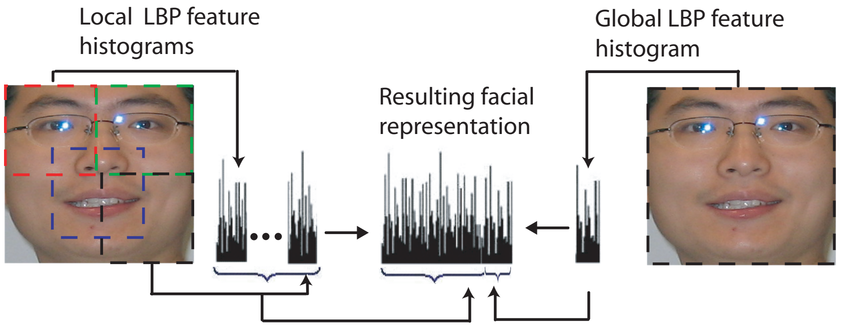
Local LBP feature histograms
Resulting facial representation
Global LBP feature histogram

Fig. 1. Facial representation with LBP texture features: a face image is with set of local and global LBP histograms

The original 20×20 pixels face image is divided maximally into nine overlapping regions of 10×10 pixels. The basic 4-neighbors LBP operator with 16-bin histogram is used resulting totally of 144-bin histogram of local features. In addition to that the global 16-bin histogram is concatenated to the local features in order to create the full 160-bin face histogram.

The AdaBoost algorithm is a discriminative learning methdod that has been widely used in different object detection tasks. The idea is to combine several relatively weak features into one stronger hypothesis [10]. In our case the weak features are the LBP histogram bin values calculated over certain image regions. These positions and corresponding LBP values are learned off-line with the set of labeled face samples. Few example training images are shown in Fig. 2 a.

Inspired by the well known Viola and Jones [13] object detection approach, we built a cascaded classifier structure to speed up the detection. In the early cascade levels only a few histogram positions were considered to rapidly reject majority of classified image regions. The face like image regions were classified with more features in order to make robust detection. We applied the trained cascade to the image pyramid using a sliding window approach to classify image regions in different scales. As an result, the rectangular coordinates of each detected face was obtained. Example output from the face detection algorithm is shown in Fig. 3.

2.2 Eye Detection

Once a possible face of the user is detected, the eyes are searched for within the facial region. The approach for detecting the eyes is similar to the face detection algorithm except the size of the template image is 16×20 pixels to more accurately cover the eye region. Some example training samples are shown in Fig. 2 b. Face orientation and size were used to restrict the search window and scales in the eye detection phase. In this sense, the actual detection is carried out with very low computational costs. The centers of the detected windows are used as eye coordinates for the tracker.

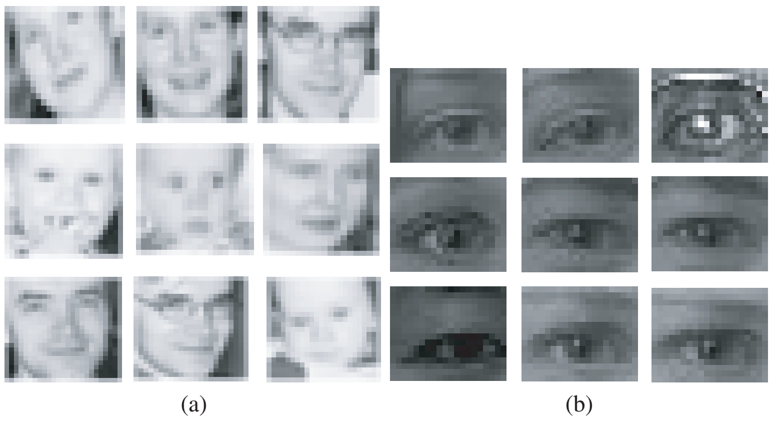

Fig. 2. Example training images for face (a) and eye (b) detector

2.3 Motion Features

In motion feature extraction, we apply an efficient method developed in our earlier work [14]. First, the face region of the previous frame is split to 16 subregions, and one 8x8 pixel block is selected from each region based on analysis of spatial gradient information. Displacement of selected blocks is estimated using zero mean sum of squared differences (ZSSD) criterion which is applied over some range of candidate displacements (e.g. 16 pixels). Refinement to subpixel precision is done in the neighborhood of the displacement minimizing the ZSSD measure using fitting of the second order polynomials.

The ZSSD values are also used to analyse uncertainty information related to the local displacement estimate. This information is used in RANSAC style outlier analysis which provides reliable motion features for 3-D pose estimation. Each motion feature consists of (1) block position in the previous frame $\mathbf{p}_i$ (2) block displacement estimate $\mathbf{d}_i$ and (3) displacement error estimate encoded as a 2×2 covariance matrix $\mathbf{C}_i$. This information in addition to the eye coordinates is used as measurements in the tracking phase.

3 Face Tracking and Pose Estimation

The 3-D camera pose with respect to the face can be estimated based on a face model and corresponding 2-D image observations. In our method, the eyes and motion features detected from the face region are used as measurements. We apply extended Kalman filtering (EKF) to estimate the camera position and to track the eyes as well as the motion features.

3.1 Face Pose and Motion Model

We model the face as a rigid plane which is not an accurate description for the face, but considering our application it provides sufficient pose estimates to control the user

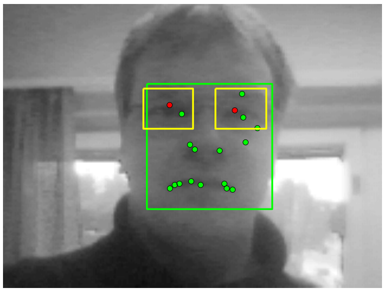

Fig. 3. Facial features: face detection result (green rectangle), eye detection result (yellow rectangles and red dots), centers for the eyes (red dots) and motion features (green dots)

interface. The center of the face coordinate system is set to the center of the detected face region in the first frame. Initially the face plane is parallel to the image plane and its distance to the camera is some predefined constant. The face model includes the eye positions ($\mathbf{P}_1$ and $\mathbf{P}_2$) and the motion feature positions ($\mathbf{P}_i, i = 3, 4, ..., N + 2$). $\mathbf{P}_1$ and $\mathbf{P}_2$ are the backprojected face plane coordinates of the eyes detected in the first image. $\mathbf{P}_i, i = 3, 4, ..., N + 2$ are the backprojected face plane coordinates of the motion feature positions $\mathbf{p}_i$ where projection uses the pose estimate associated with the previous frame. The model points for motion features are updated for each incoming frame, but the eye positions are fixed after initialization.

The first order dynamic model for the system is represented as

$$\mathbf{x}_{k+1} = \mathbf{\Phi}_k \mathbf{x}_k + \mathbf{\Gamma}_k \varepsilon_k, \tag{1}$$

where the pose and the velocity terms are included in the state vector $\mathbf{x}_k$ at time instant k, and $\mathbf{\Phi}_k$ is the corresponding state transition matrix. Specifically, the state vector $\mathbf{x}_k$ consists of the position (x_k, y_k, z_k) and orientation $(\omega_k, \varphi_k, \kappa_k)$ of the face model with respect to the camera and the corresponding velocities. It is defined as follows

$$\mathbf{x}_k = [x_k, \dot{x_k}, y_k, \dot{y_k}, z_k, \dot{z_k}, \omega_k, \dot{\omega_k}, \varphi_k, \dot{\varphi_k}, \kappa_k, \dot{\kappa_k}]^T.$$

$\mathbf{\Gamma}_k \varepsilon_k$ models the uncertainty of the dynamic model. $\mathbf{\Gamma}_k$ is the process noise transition matrix. We model arbitrary accelerations as process noise ε_k which is assumed to be Gaussian distributed with zero-mean and the covariance matrix $\mathbf{Q}_k = E\{\varepsilon_k \varepsilon_k^T\}$. The pose parameters are initialized according to the initial face model. In the beginning, the velocities are set to zero. We approximate the variances of the process noise from the maximum accelerations assumed.

3.2 Measurements

The measurement model is needed to relate the 3-D pose parameters (state) to the 2-D image observations. We use a perspective camera model to transform the object coordinates $\mathbf{P}_i$ to image coordinates (u_i, v_i) and the model is

$$\mathbf{z}_k = \mathbf{h}(\mathbf{P}_i, \mathbf{x}_k) + \eta_k, \tag{2}$$

where $\mathbf{h}(\mathbf{P}_i, \mathbf{x}_k)$ is a non-linear observation function that uses the perspective camera model to transform object coordinates $\mathbf{P}_i$ to image coordinates $\mathbf{z}_k$ using state $\mathbf{x}_k$. The observation model is linearized using its partial derivatives with respect to the state variables $\mathbf{x}_k$ to obtain the Jacobian matrix.

The actual measurement $\mathbf{z}_i, i = 1, ..., 2$ for the eyes are extracted in a region around the predicted locations. The size for the region is adjusted dynamically by projecting the estimation uncertainty to the image coordinates. The eye detection is performed as described in Sec. 2.2 and (u, v)- coordinates are obtained as a result. If there is too much deviation from the prediction, then the predicted position is chosen instead of the actual measurement. In the case of motion features, the measurement is $\mathbf{z}_i = \mathbf{p}_i + \mathbf{d}_i, i = 3, ..., N + 2$, where $\mathbf{p}_i$ is motion feature position in the previous frame and $\mathbf{d}_i$ is the detected displacement in the current frame. The measurement noise η_k for the eyes is assumed to be Gaussian with zero-mean and variance of 9 pixels. The error covariance matrix $\mathbf{C}_i$ is derived for each feature separately using the method described in [14].

3.3 Tracker

The EKF algorithm estimates the pose recursively repeating prediction and correction stages. In the first stage, the pose and locations of the features at the next time instant are predicted based on the previous pose estimate and the dynamical model. In the correction stage the predicted pose is adjusted by using facial feature measurements. Also, the face region is updated based on the estimated pose allowing motion feature extraction from the correct image region. The measurements can occasionally cause errors that make the tracking fail. If the feature position is not within the uncertainty limits derived from the error covariance matrix or the feature is lost then the prediction is used instead of the measurement. The error cannot exist longer than three frames or the initialization is started again.

4 Experiments

We evaluated the feasibility of the method for controlling the spatially aware user interface with a camera-equipped mobile phone. The phone used was Nokia N95 containing ARM11 based processor running at 330 MHz. It has two built-in cameras, one for high resolution imaging and other low resolution camera pointing to the users' face. In experiments, we used the latter with an image resolution of 320x240 pixels and a maximum obtainable frame rate of 15 fps.

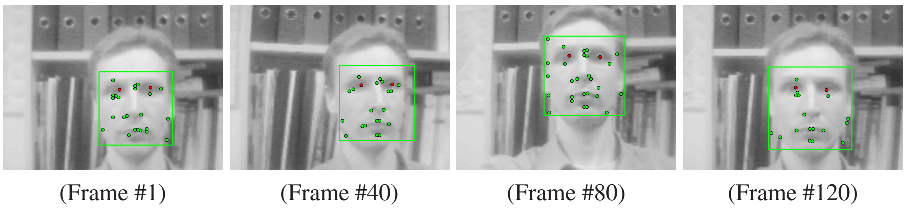

Fig. 4. Facial feature tracking example

In the experiments, we asked a subject to perform rectangular hand movement representing hypothetical input for the user interface. In order to test repeatability of the method, a rectangle (120x90 mm) was first drawn on the board. The user followed the outline of the rectangle and repeated the experiment 10 times. The trajectory for the movement was obtained by solving the camera position from the estimated 3-D pose of the face. Fig. 5 (a) shows the obtained 3-D trajectories of the device as a projection in the $x - y$ plane and Fig. 5 (b) illustrates the same result in 3-D coordinates. The trajectories are uniformly scaled and aligned with each other. As illustrated in the figure, the repeatability achieved for the method is reasonably good and it allows the position based control of user interfaces. The results can be further improved by adding more fixed points other than the eyes to the face model providing enhanced positional information. Possible additional points are, for example, the mouth, the mouth corners and the nostrils.

We also assessed the feature tracking subjectively because the ground truth for the pose was not known. Fig. 4 shows an example result of the extracted facial feature locations during tracking. The tracking was successfully completed without interruptions in all of the sequences. There were sometimes large errors in the eye measurements or rarely the eye was not detected at all. In these situations the tracker used the predicted measurements and tracking did not fail. Although the lighting changed continuously throughout the sequences, the system still worked properly.

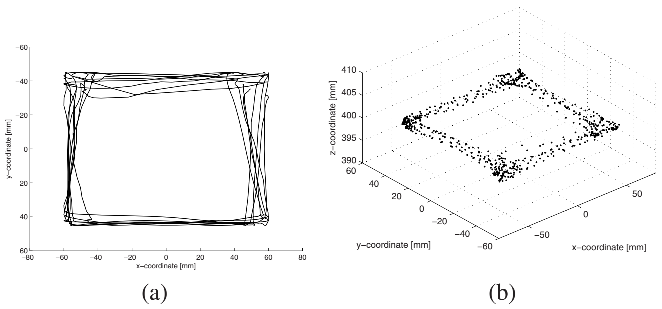

Fig. 5. Trajectories for the repeability test: (a) trajecrory in x-y plane (b) trajectory in 3-D coordinates

5 Conclusions

We have presented a new feature based head tracking method for controlling user interfaces of mobile devices. It automatically detects the face and eyes of the user employing local texture features and boosting. The 3-D position of the device with respect to the user face is estimated with extended Kalman filtering using a novel combination of motion features and eyes extracted from the face region. The camera position obtained can be used to control the spatially aware user interface of camera-equipped mobile phone. The advantage of the method is that real-time performance on a resource limited mobile device can be easily achieved, which makes it useful for practical applications.

Acknowledgments

The financial support of the Academy of Finland (project no. 110751) is gratefully acknowledged.

References

1. Fitzmaurice, G.W.: Situated information spaces and spatially aware palmtop computers. Communications of the ACM 36(7), 38–49 (1993)
2. Yee, K.P.: Peephole displays: pen interaction on spatially aware handheld computers. In: Proc. of the SIGCHI conference on human factors in computing systems, pp. 1–8 (2003)
3. Hinckley, K., Pierce, J., Sinclair, M., Horvitz, E.: Sensing techniques for mobile interaction. In: Proc. 13th ACM symposium on User Interface Software and Technology, pp. 91–100 (2000)
4. Haro, A., Mori, K., Capin, T., Wilkinson, S.: Mobile camera-based user interaction. In: Proc. of IEEE Workshop on Human-Computer Interaction, pp. 79–89 (2005)
5. Azarbayejani, A., Starner, T., Horowitz, B., Pentland, A.: Visually controlled graphics. IEEE Transactions on Pattern Analysis and Machine Intelligence 15(6), 602–605 (1993)
6. Black, M.J., Yacoob, Y.: Tracking and recognizing rigid and non-rigid facial motions using local parametric models of image motion. In: Proc. IEEE International Conference on Computer Vision, pp. 374–381 (1995)
7. Basu, S., Essa, I., Pentland, A.: Motion regularization for model-based head tracking. In: Proceedings of the 13th IEEE International Conf. on Pattern Recognition, pp. 611–616 (1996)
8. La Cascia, M., Sclaroff, S., Athitsos, V.: Fast, reliable head tracking under varying illumination: An approach based on registration of texture-mapped 3d models. IEEE Trans. on Pattern Analysis and Machine Intelligence 22(4), 322–336 (2000)
9. Ojala, T., Pietikäinen, M., Mäenpää, T.: Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. IEEE Transactions on Pattern Analysis and Machine Intelligence 24(7), 971–987 (2002)
10. Freund, Y., Schapire, R.: A decision-theoretic generalization of on-line learning and an application to boosting. In: Europ. Conf. on Computational Learning Theory, pp. 23–37 (1995)
11. Hadid, A., Zhao, G., Ahonen, T., Pietikäinen, M.: Face analysis using local binary patterns. In: Mirmehdi, M. (ed.) Handbook of Texture Analysis. Imperial College Press (2007)
12. Hadid, A., Pietikäinen, M., Ahonen, T.: A discriminative feature space for detecting and recognizing faces. In: IEEE Conference on Computer Vision and Pattern Recognition (2004)
13. Viola, P., Jones, M.: Rapid object detection using a boosted cascade of simple features. In: IEEE International Conf. on Computer Vision and Pattern Recognition, pp. 511–518 (2001)
14. Sangi, P., Hannuksela, J., Heikkilä, J.: Global motion estimation using block matching with uncertainty analysis. In: 15th European Signal Processing Conference (2007)