A PERCEPTUAL INTERFACE USING INTEGRAL PROJECTIONS

Ginés García-Mateos[1], Sergio Fructuoso-Muñoz[1]

[1] Dept. de Informática y Sistemas
University of Murcia, 30.071 Espinardo, Murcia, Spain
ginesgm@um.es

In this paper we describe a perceptual interface which allows to navigate in a virtual 3D environment by interpreting the movements of the face of human users. The core of the proposed approach consists of a face tracker, which is based on the computation and analysis of integral projections. Applying some predefined heuristics, the location and pose of the face are first estimated and, then, transformed in control signals for the navigation in the virtual world. The system has been implemented using OpenCV and DirectX 9. Some experimental results are shown, proving the feasibility of the proposed technique.

Introduction

The emerging field of perceptual interfaces aims at the development of non-intrusive and more natural ways of man/machine interaction [1]. Most research in this field has been done involving multimodal data [1], [2], [3]: face and gesture recognition, body movement, audio data, etc. Potential applications of perceptual interfaces in the future are far beyond their present uses, which are reduced due to practical reasons. In this sense, face based interfaces are among the most promising new ways of interaction. The human face represents a natural and feasible mode of input, and can provide computers with a wide range of information of the user, such as identity, facial expression, 3D location, orientation and head movement.

The perceptual interface described in this paper is based on the detection and tracking of human faces. It has been applied to the navigation in a virtual 3D environment, specifically created for this purpose. In this environment the user adopts the role of a character that can move freely through the world, in a similar way to first-person games. A global view of the system is depicted in Fig. 1.

A camera is situated in front of the user. The computer detects and tracks the location of the face in the video sequence, extracts information concerning the movement of the face and its 3D pose, and finally translates this data into a movement in the virtual world. This process is carried out in real-time in an off-the-shelf PC, using a low-cost webcam or a firewire camera for the acquisition.

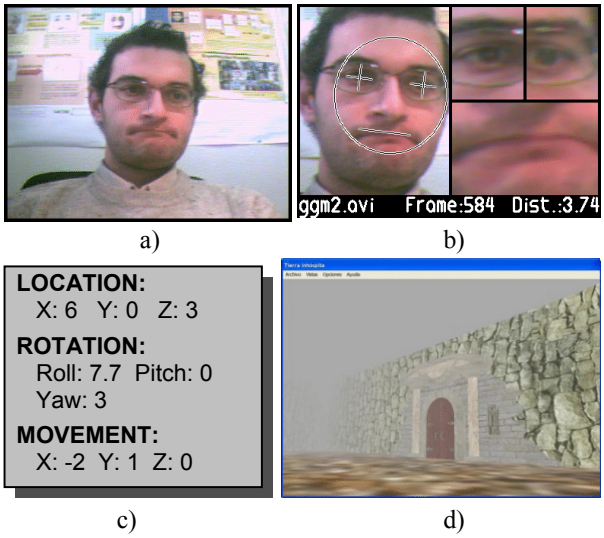


Fig. 1. Virtual world visually controlled by a face. a) Image captured from the camera. b) Face detected and facial features located. c) Estimated pose parameters. d) Movement translated into the virtual 3D environment.

In the following sections we will describe this system, centering our discussion on the perceptual part of the problem. We present the technique of integral projections, used for face tracking, and detail the extraction of 3D information from the tracked faces.

Face Tracking with Integral Projections

Face detection and tracking are previous and essential problems in the design and construction of a perceptual interface based on human faces. Integral projections have proved to be a very useful technique to tackle both problems, as discussed in [3] and [4]. Besides, they provide some valuable additional information that can be used in pose estimation, as described in the following section.

Intuitively, an integral projection is the average of gray levels along a row or column of pixels. Applied on human faces, typical patterns of dark and light regions appear. Moreover, the patterns of projections are stable and robust for a same individual along a video sequence [4]. We can make use of this property to construct a face tracking system, which is based on the alignment of projections. The structure of the process is depicted and summarized in Fig. 2.

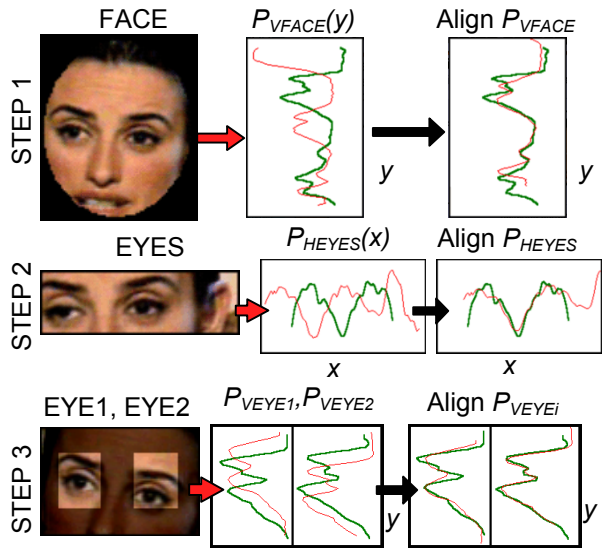


Fig. 2. A three-step face tracking process using integral projections. Left: input images for each step. Center: projection models (thick green lines) and instances (thin red lines) before alignment. Right: the same projections after alignment.

As indicated in Fig. 2, face tracking is decomposed into three main steps:

1. *Vertical alignment.* The vertical integral projection of the expected location of the face is computed. This projection is aligned with respect to a model of vertical projection of the face, obtaining the movement and scale in the vertical direction.
2. *Horizontal alignment.* After step 1, the region of eyes is segmented, and its horizontal projection is computed. Then, it is aligned with an adequate model, thus obtaining horizontal movement and scale.
3. *Orientation alignment.* Here orientation is considered in a planar sense, i.e., with respect to the plane of the image. In this step, the alignment is done by independently estimating the height of both eyes.

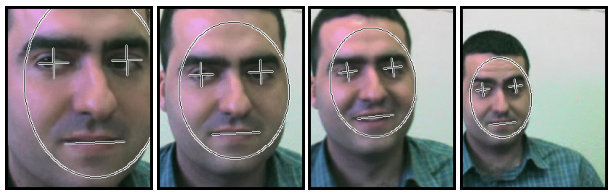
3D Pose Estimation

The tracking process provides the location of the face and facial features (eyes and mouth) along a video sequence. Using this information, 3D pose and movement of user's head are computed. The general problem of pose estimation –i.e., give values for 3D location (*x*,*y*,*z*) and angle (*roll*, *pitch*, *yaw*)– is computationally complex. However, we are not interested in a precise pose calculation but in an estimation suitable for the perceptual interface.

Thus, some heuristic methods have been defined to allow a fast approximation of pose parameters. These heuristics are explained in the following subsections.

Depth Estimation

The depth, or distance of user's head to the camera, is inversely proportional to the size of the head in the image. This holds as long as the face is considered a rigid object, which might not always be true, due to facial expressions. The size of the head is approximated with the distance from left to right eye. Some samples of depth estimation are shown in Fig. 3.

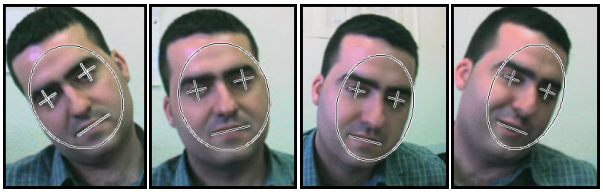


a) 0.674 b) 0.947 c) 1.184 d) 1.636

Fig. 3. Depth computation. Estimated values (relative to the initial distance) are shown below the pictures.

Roll Estimation

The estimation of roll is straightforward. Since roll involves a rotation along the plane of the camera, it can be computed using the perceived angle of the face. In particular, we use the angle of the line passing through both eyes with respect to the horizontal axis. Fig. 4 contains some samples of roll estimation.



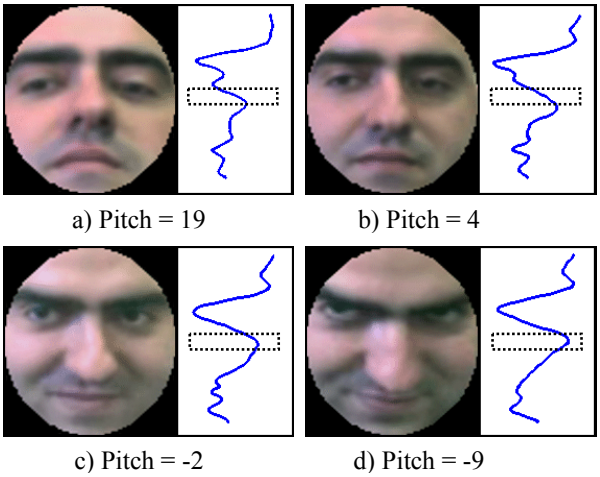
a) -33.4º b) -13.0º c) 13.5º d) 20.2º

Fig. 4. Roll computation. Estimated angles are shown below the pictures.

Obviously, when this rotation is combined with other kinds of rotation, the values obtained with this simple method are not very reliable.

Pitch Estimation

Pitch cannot be precisely estimated just using the location of eyes and mouth. Instead, the computed projections are used to extract some additional information. When the user raises or lowers his/her head, the vertical projection of the whole face changes smoothly, as we can observe in Fig. 5.



a) Pitch = 19 b) Pitch = 4

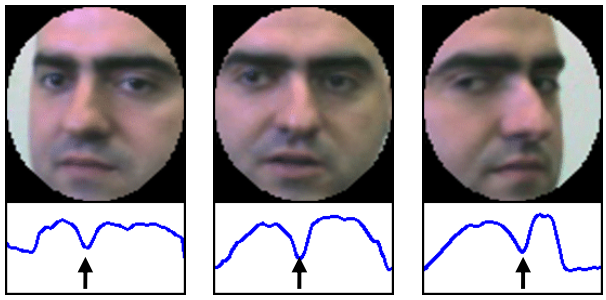
c) Pitch = -2 d) Pitch = -9

Fig. 5. Pitch computation. Vertical integral projections of the faces are shown on the right of the pictures, and estimated values below them.

The following *ad hoc* rule has been defined: estimated pitch is proportional to the value of the projection at a certain point between the maximum corresponding to the eyes and the minimum of the nose; the point is marked with a box in Fig. 5. This simple heuristic measure has a viable explanation: when the face is looking upwards, nostrils are more and more visible, making projections darker at that point.

Yaw Estimation

As before, yaw is also approximated with a fast heuristic method which might not be applicable in a general problem. In this case we consider the horizontal projection of the region of eyes. This projection usually contains a maximum corresponding to each eye and a minimum between them. When the user looks to the left or to the right, the horizontal projection changes accordingly. This effect is shown in Fig. 6.



a) -7 b) 0 c) 11

Fig. 6. Yaw computation. Horizontal integral projections of the eyes, and estimated yaw values are shown below the pictures.

Therefore, yaw is estimated computing the relative position of the minimum between the eyes in these horizontal projections, i.e., the locations of the arrows in Fig. 6.

Virtual Environment

We have developed a virtual 3D environment whose control is done using the perceptual interface described before. This environment is similar to a 3D game and consists of a number of cells, or rooms, where the user can move freely. The map of the world is shown in Fig. 7. The action is carried out in a first-person view. The user can move forward, backward, left, right, rotate left and right, and look up and down. These movements are controlled in the following way:

- *Move forward/backward.* Depth estimation is used to determine direction and magnitude of this movement.
- *Move left/right.* The location of the face, with respect to the image center, is used.
- *Rotate left/right.* In this case, two values are considered: roll and yaw. Both contribute to determine the quantity of rotation.
- *Look up/down.* The estimated pitch is applied in this control signal.

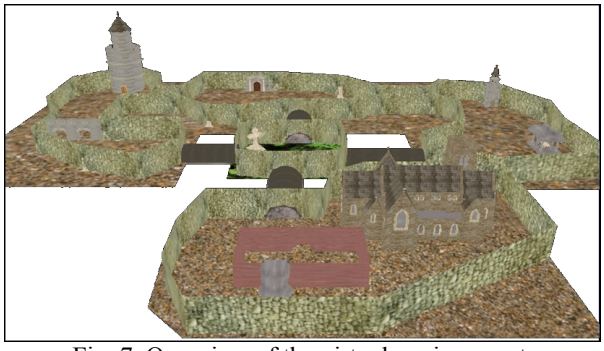


Fig. 7. Overview of the virtual environment.

To make the system more stable, a temporal smoothing is introduced in the estimated pose values. This causes a certain delay, of about 5 frames, but avoids trembling due to small errors in the estimations.

On the other hand, as we have mentioned before, not all of the estimated values are independent. For example, a rotation along one direction can result in a bad estimation in another parameter. Therefore, the movements have been prioritized to avoid this inter-dependence. In our case, the movement with most priority is pitch rotation. If a relevant pitch value is detected, all the other movements are omitted.

Experimental Results

The perceptual interface has been implemented using Intel's IPL and OpenCV libraries [5]. The rendering engine uses DirectX 9.0b, thus enabling hardware optimizations[1]. The program has been tested in a Pentium IV at 2.6 GHz and 512 Mbytes of RAM. For the acquisition we have used two cameras: a QuickCam Pro through USB, and a SONY DFW500 through firewire. We can point out some interesting results:

- The program performs in real-time, with an average about 30fps in rendering. Using the USB camera, image processing is done at 10fps, and with the firewire camera at 30fps.
- In general, the response and stability of the system are very good. The face tracker is able to work with typical speeds in the user's movement, and under a wide range of rotations. The method is also robust to facial expressions, like speaking, blinking, etc.
- Some parameters are more reliable than others. In particular, only a small rotation is admitted in yaw, and profile faces are not tracked. A wider variation is allowed in pitch. However, these estimated parameters are not very reliable for large rotations.

Conclusion

Integral projections has already been used in problems of face detection and tracking. They provide a fast, robust and stable method to track and locate human faces along a video sequence. Here we have shown that this approach can be applied in the design of a perceptual interface based on human faces. Using some simple computations, it has been extended to produce estimation for 3D location and rotation of user's head. Although the values obtained with this method are not very adequate if a precise 3D tracking is needed, they prove to be very useful in applications like perceptual interfaces.

References

1. Pentland, A.: Looking at People: Sensing for Ubiquitous and Wearable Computing. IEEE Trans. on PAMI. Vol 22, No 1. Jan. 2000. P. 107-119.
2. Bradski, G. R.: Computer Vision Face Tracking For Use in a Perceptual User Interface. Intel Technology Journal Q2'98 (1998).
3. Sobottka, K., Pitas, I.: Segmentation and Tracking of Faces in Color Images. Proc.of 2nd Intl. Conf. on Aut. Face and Gesture Recognition (1996). P. 236-241.
4. García-Mateos, G.: Refining Face Tracking with Integral Projections. Proc. of AVBPA'2003, LNCS 2086 (2003). P. 222-229.
5. Intel OpenCV: Open Source Computer Vision Library http://www.intel.com/research/mrl/research/opencv

[1] Some results and sample videos can be found at: `http://dis.um.es/~ginesgm/fip/percint.html`