Available online at www.sciencedirect.com
SCIENCE DIRECT
Image and Vision Computing 22 (2004) 931–942
www.elsevier.com/locate/imavis

Nouse 'use your nose as a mouse' perceptual vision technology for hands-free games and interfaces

Dmitry O. Gorodnichy*, Gerhard Roth

Institute for Information Technology, National Research Council of Canada, M-50 Montreal Rd, Ottawa. Ont., Canada K1A 0R6

Received 31 July 2003; received in revised form 2 December 2003; accepted 22 March 2004

Abstract

Due to recent increase of computer power and decrease of camera cost, it became very common to see a camera on top of a computer monitor. This paper presents the vision-based technology which allows one in such a setup to significantly enhance the perceptual power of the computer. The described techniques for tracking a face using a convex-shape nose feature as well as for face-tracking with two off-the-shelf cameras allow one to track faces robustly and precisely in both 2D and 3D with low resolution cameras. Supplemented by the mechanism for detecting multiple eye blinks, this technology provides a complete solution for building intelligent hands-free input devices. The theory behind the technology is presented. The results from running several perceptual user interfaces built with this technology are shown.

Keywords: Computer-human; Stereo-tracking; Face-tracking

1. Introduction

We consider the problem of designing vision-based perceptual user interfaces, which are the systems that use video-cameras to perceive the visual cues of the user, such as the motion of the face, to control a program (see Fig. 1). The main applications of these systems are seen in human–computer interaction, teleconferencing, entertainment and industry for disabled. In particular, face-tracking-based program control can be used as a hands-free alternative to tangible pointing devices such as mouse, joystick, track pad, etc. It can be used, for example, to switch a focus of attention in windows environment. It can also be used to control commercial computer games, immersive 3D worlds and avatar-like computer-generated communication programs.

Perceptual user interfaces provide a way for multiple-user interaction—several users can be tracked at the same time with several cameras. They also can be applied to video-conferencing, where it can be used to correct the gaze direction, and also in video-coding, content-based image retrieval and security industry.

To be operational, perceptual user interfaces require face-tracking to be fast, affordable and, most importantly, precise and robust. In particular, the precision should be sufficient to control a cursor, while the robustness should be high enough to allow a user the convenience and the flexibility of head motion.

A few hardware companies have developed hands-free mouse replacements. Their systems, however, either use dedicated software or structured environment (e.g. markings on the user's face) to simplify the tracking process. At the same time, recent advances in hardware, invention of fast USB and USB2 interfaces, falling camera prices, and increase of computer power brought a lot of attention to the real-time face-tracking problem from the computer vision community. The obtained vision-based solutions though still do not exhibit the desirable precision and/or robustness. Let us review these solutions.

The approaches to vision-based face-tracking are generally divided into two classes: image-based (global) and feature-based (local) approaches [1–3]. Image-based approaches use global facial cues such as skin colour, head geometry and motion. They are robust to head rotation and scale and do not require high quality images.

* Corresponding author.
E-mail addresses: dmitry.gorodnichy@nrc-cnrc.gc.ca (D.O. Gorodnichy), gerhard.roth@nrc-cnrc.gc.ca (G. Roth).

0262-8856/$ - see front matter © 2004 Elsevier B.V. All rights reserved.
doi:10.1016/j.imavis.2004.03.021

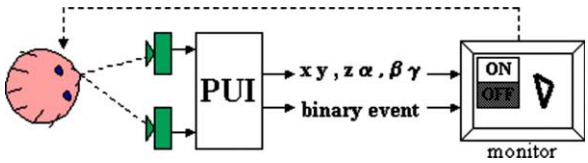

Fig. 1. Vision-based perceptual user interface: the system diagram.

These approaches, however, lack precision and therefore cannot be used to control the cursor precisely.

In order to achieve precise and smooth face-tracking, feature-based approaches are used. These approaches are based on tracking individual facial features and theoretically should be able to track faces with pixel-size precision. In practice, however, they are not, because commonly used edge-based features such as corners of brows, mouth, etc. are not rotation and/or scale invariant and are therefore easily lost in unconstrained head motion. These approaches usually also require high-resolution cameras so that edges can be well detected.

Recently a concept of a curvature-based convex-shape nose feature has been introduced [4], which is shown to be rotation and scale invariant. In the present paper, we make use of this feature to build a perceptual vision system which is able to track faces both robustly and precisely.

We also show how tracking the rotation and scale invariant nose feature allows one to track other facial features in 3D using two uncalibrated cameras.

While stereo-tracking of faces using two cameras is performed by several other authors [5–8], our approach differs from that of the others in two important aspects. First, we do not use dedicated stereo setups, but rather use two liberally positioned cameras. Using recent advances in the projective vision theory, we compute the relationship between the cameras on-the-fly; this relationship, represented by the fundamental matrix, is naturally obtained while observing a face with both cameras. Second, while being able to track a face in 3D for up to 40° of head rotation around all three axes of rotation, our approach is shown to be very suitable for low-resolution and low-quality cameras, such as common USB webcams.

The organization of the paper is the following. After reemphasizing the importance of the nose for vision-based face-tracking and defining the convex-shape nose feature in Section 2, we show how to compute the fundamental matrix for two cameras so that the face can be tracked in 3D. Section 4 describes the applications of the proposed technology, which became known as the *Nouse* (Nose as Mouse) perceptual vision technology [9–12]. To complete the technology with a face-operated version of mouse click, we introduce the *Doubleblink* event, which, as shown, can be detected even when a face moves. Conclusions end the paper.

2. Importance of nose for face-tracking

Due to human physiology, our nose, as the most protruding and also the furthest from the axes of head rotation part of our face, has the largest degree of motion freedom, as compared to other parts of the face. As such, when in need to point at something hands-free, we would probably use nose as a replacement for a finger; eye pupils are less suitable for the task, since, because of the large amount of involuntary saccadic motion they exhibit [13], they are much more difficult to control. Let us show now that nose is also a feature which is extremely useful for vision-based face-tracking.

Within the framework of template matching, which remains to be one of the fastest and most commonly used techniques in feature tracking [14–20] in order to build robust and precise face-tracking-based interfaces, we introduce the following propositions.

Proposition 1. *One feature only should be used for the final decision on the head position in video.*

This eliminates the jitter problem which arises when tracking several features due to the fact that facial features move non-rigidly. This also provides a user with an intuitive way of controlling with the head by simply visualizing the feature as a chalk or a tip of a joystick.

Proposition 2. *The tracked feature should always be clearly visible for all face positions and expressions, including the cases with the users who wear eyeglasses, mustaches or beards.*

It follows from Propositions 1 and 2 that the problem of robust tracking is the problem of finding such a facial feature which would stay invariant during the motion of the user's face.

In image processing, a feature is often thought of as a point on the object surface which has large change of intensity gradient in the image [21]. This explains why the most commonly used in face-tracking facial features are corners and edges of mouth, brows, nostrils and eye pupils. These edge-based facial features are difficult to track, however, if the face rotates or changes the facial expression.

In order to select a facial feature suitable for precise and smooth tracking, the concept of a convex-shape feature has been introduced in Ref. [4] and the rotation invariant *nose feature* has been defined as the point on the tip of the nose closest to the camera. More generally, the nose feature may lie slightly off from the point closest to the camera, in which case it is the closest to another fixed object in space.

It can be shown using the shape-from-shading theory [22] that under approximately the same lighting condition the intensity pattern around the thus defined nose feature is not affected by the orientation of the face and the distance from the face to the camera, provided that face motion is much less than the distance to the camera and the light source. Based on this, we make the nose template vector out

Table 1
Applicability of different face sizes for face processing tasks

Face size	80 × 80	40 × 40	20 × 20	10 × 10	Colour	Motion	Intensity
i.o.d.	40	20	10	5			
Eye size	20	10	5	2			
Nose size	10	5	–	–			
FS	X	X	X	m	+[34,35]	+[36]	+
FD	X	X	m	–	+[35,37]	+	+[1,2,38,39]
FT	X	X	m	–	+[40]	+	+[3]
FL	X	m	–	–		+[41]	+[4]
FER	X	X	m	–	+	+	+[20,42]
FC	X	X	m	–			+43
FM/I	X	X	–	–			+[44]

FS, FD, FT, FL, FER, FM/I refer to face segmentation, detection, tracking, localization, expression recognition and memorization/identification. The distinction between FD and FT is that FT uses the past information about the face location, whereas FD does not, and between FT and FL that FT detects an approximate face location, while FL provides the exact position of a face or facial feature(s). The face size, defined as twice the intra-ocular distance (i.o.d) squared, is given in pixels. 'X' indicates that for a given face size the task can be executed; 'm' signifies that the size is marginally acceptable for the task. The representative work done in the area is given in brackets.

of the gray-scale intensities centered around the extremum of the nose surface. The template vector size is chosen so that it covers the spherical surface around the tip of the nose. Table 1 shows the size of the nose convex-shape area for different face sizes, as compiled from our experiments.

The defined nose feature is always visible in camera and can therefore be always located. This is a very important property of the nose, which as mentioned earlier, does not hold for other facial features. It gives a user the desired flexibility and convenience of head motions.

It should be noted that the convex-shape nose feature is not associated with a physical point on a nose. Instead, it is associated with the extremum of the nose curvature, which *moves* on the nose surface. In the camera centered coordinate system with Oz axis perpendicular to the image plane going from the camera, the nose feature corresponds to the extremum of the nose surface $z = z(x, y)$. This is seen from Fig. 2, which shows the range of motion within which the convex-shape nose feature is tracked, and also from the video-sequences, snapshots of which are shown in Figs. 3 and 4. In the first sequence, a person performs the 'no' (left–right) motion, followed by the 'yes' (up–down) motion, and then followed by the circular motion, holding his shoulders still at all times; the position of the tracked convex-shape nose feature is overlaid on the top of the image.

In the second sequence, which takes less than 3 s to perform, a user draws a musical 'G' clef sign holding his shoulders still. In both cases, a user rotates his head at normal fast speed as far he can, provided that he can still see the screen. More results showing the precision and robustness of nose tracking are shown in Section 4 (Fig. 13).

The experiments conducted with different people, lighting conditions and different USB cameras show that the nose tip is tracked for up to about 40° of rotation of head in all three directions. This range of allowable motion practically covers all possible head motion a user may exhibit while looking at the screen. Scale-wise, for a camera resolution of 160 × 120, the nose is robustly tracked when a user sits within 30–60 cm distance from the camera. This is the most common case in computer-user setups, which shows that low-resolution cameras can be used for the task.

By observing the figures, one can notice another important property of the convex-shape nose feature,

Fig. 2. The figure shows the range of motion within which the convex-shape nose feature is tracked.

Fig. 3. Tracking 'Yes', 'No' and circular motion. Shoulders are held still.

Fig. 4. Drawing the G-clef sign with head motion. Shoulders are held still.

which is the smoothness of its motion. This smoothness is due to the fact that the function, which maps the position of the pixel in the image to the feature vector, is continuous in the area around the extremum of the nose surface. By other words, the closer in the image pixel u is to the nose feature pixel f, the smaller is the distance between vectors $\vec{V}_u$ and $\vec{V}_f$, where $\vec{V}_{(\cdot)}$ denotes the vector obtained by centering a peephole mask on pixel $(\cdot)$.

The continuity property is very important not only for smooth tracking but also for precise tracking, as it allows one to use the weighted average of pixels around the pixel of the best match instead of the position of the pixel itself. In particular, the following convolution filter can be applied to refine the position of the feature

$$\hat{u} = \sum_{k \in \Omega} \omega_f^k u^k, \tag{1}$$

where Ω designates the area around the pixel of the best match u, and weights ω_f^k are set proportional to the normalized correlation between vector $\vec{V}_u^k$ and the template vector $\vec{V}_f$. Due to the continuity property of the mapping function in the vicinity of the nose feature, these weights can be considered monotonically decreasing with the distance to the nose feature, and thus the applied convolution filter is guaranteed to refine the position of the feature.

Using the continuity property and the averaging convolution filter is analogous to the approach taken in Ref. [19] where the adaptive logic network (ALN) [23] is used to detect eye pupils in the photographs. In that work, however, the continuity is achieved by designing the output scheme which allows the continuity of the feature mapping and it is the ALN that does the filtering.

Finally, worth mentioning is that applying the weighted average not only makes tracking more reliable due to using a collective decision instead of the decision based on a single pixel, but it also allows one to compute the position of the nose feature with one extra digit behind the decimal. This becomes very useful for mapping the position of the feature in a low resolution image to the position of the cursor or other virtual object in high-resolution screen.

3. Stereo-tracking

Below we describe a stereo-tracking system which makes use of the convex-shape nose feature, in order to allow 3D face-tracking with the aid of two ordinary USB web-cameras. The cameras do not need to be aligned; they are simply mounted on top of the computer monitor so that the user's face is seen by both of them (see Fig. 5).

The system operates in three stages as follows. First, the user has to perform the self-calibration procedure to acquire the relationship between the cameras. For this purpose the user captures his head at the same time with both cameras at the distance to be used in tracking (Fig. 6).

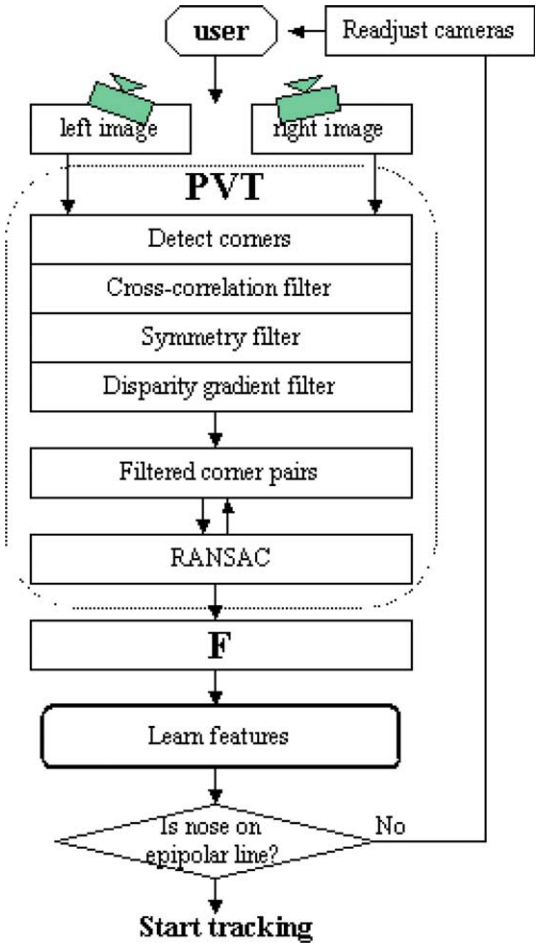

Fig. 5. Overview of the calibration procedure for the face-tracking-based user interfaces.

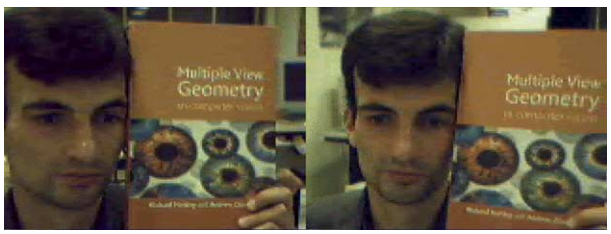

Fig. 6. Images captured by two cameras to be used in self-calibration. They should have enough visual features.

Then the correctness of the calibration information is verified during the feature learning stage (Fig. 7). Finally, the computed calibration information is used for tracking the features (Fig. 15).

3.1. Stereo self-calibration

When a 3D point in space is observed by two cameras, it is projected to the image plane of each camera [24]. This generates two vectors $\mathbf{x}$ and $\mathbf{x}'$ starting from the origin of each camera. These vectors are related to each other through the equation

$$\mathbf{x}^{\mathrm{T}}\mathbf{t}\times R\mathbf{x}' = 0, \tag{2}$$

where $\mathbf{t}$ is the translation vector between the camera positions, R is the rotation matrix. In computer vision, this equation is known as the *epipolar constraint* and is usually written as $\mathbf{x}^{\mathrm{T}}E\mathbf{x}' = 0$, where $E \equiv \mathbf{t}\times R$ is the *essential matrix*.

If cameras are not calibrated, which is the case with hand-made stereo setups made of USB cameras, then the epipolar constraint is written using a concept of the *fundamental matrix* F as follows

$$\tilde{\mathbf{u}}^{\mathrm{T}}F\tilde{\mathbf{u}}' = 0, \tag{3}$$

where $\tilde{\mathbf{u}} = [u, v, 1]^{\mathrm{T}}$ and $\tilde{\mathbf{u}}' = [u', v', 1]^{\mathrm{T}}$ define the raw pixel coordinates of the vectors $\mathbf{x}$ and $\mathbf{x}'$.

Given a point $\mathbf{u} = (x, y)$ in one image, the fundamental matrix allows one to compute a line in the other image on which the matching point must lie. This line is called the *epipolar line* and is computed as

$$\mathbf{l}_{\mathbf{u}} = F\mathbf{u}. \tag{4}$$

It is this piece of extra information that makes tracking with two cameras much more robust than tracking with one camera. In order to filter out bad matches, the *epipolar error*, defined as the sum of squares of distances of points to their epipolar lines, can be used [24]. Using Eq. (4), the relationship between epipolar error ρ and fundamental matrix F can be derived as

$$\rho \equiv \mathrm{distance}(\mathbf{u}, \mathbf{l}_{\mathbf{u}'})^2 + \mathrm{distance}(\mathbf{u}', \mathbf{l}_{\mathbf{u}})^2$$

$$= (\mathbf{u}'^{\mathrm{T}}F\mathbf{u})^2\left(\frac{1}{(F\mathbf{u})_1^2 + (F\mathbf{u})_2^2} + \frac{1}{(F^{\mathrm{T}}\mathbf{u}')_1^2 + (F^{\mathrm{T}}\mathbf{u}')_2^2}\right) \tag{5}$$

$(\cdot)_i$ designates the ith coordinate of vector $(\cdot)$, and the following proposition can be used to make tracking more robust.

Proposition 3. *Provided that fundamental matrix F of the stereo is known, the best pair of matches $\mathbf{u}$ and $\mathbf{u}'$ corresponding to a 3D feature is the one that minimizes the epipolar error defined by Eq. (5).*

Using this constraint allows us to relax the matching constraint, which is used to enforce the visual similarity between the observed and the template feature vectors. The matching constraint is the main reason why feature-based tracking with one camera is not robust, and relaxing this constraint makes tracking much more robust.

As seen from Proposition 3, it is the quality of the stereo setup calibration that is important for the success of stereo-tracking, and it is the computation of the fundamental matrix that makes the calibration process.

3.2. Computing the fundamental matrix

The calibration of the hand-made stereo consists of two parts: computing the fundamental matrix and inspecting the quality of the computed fundamental matrix. Fig. 5 shows the entire procedure.

Computing of the fundamental matrix is done using the public domain projective vision toolkit (PVT) [25,26] and is based on finding the correspondences in two images captured with both cameras observing the same static scene. Because off-the-shelf USB cameras are usually of low quality and resolution, extra care is taken to deal with the bad matches by applying a set of filters and using robust statistics. The description of each step follows.

Finding interest points. After two images of the same scene are taken, the first step is to find a set of corners or interest points in each image. These are the points where there is a significant change in intensity gradient in both the x and y direction. A local interest point operator [27] is used and a fixed number of corners is returned. The final results are not particularly sensitive to the number of corners. Typically there are in the order of 200 corners found in each image.

Fig. 7. Examining the quality of stereo calibration. The features to be tracked must lie on the epipolar lines.

Matching corners and symmetry filter. The next step is to match corners between the images. A local window around each corner is correlated against all other corner windows in the adjacent image that are within a certain pixel distance [28]. This distance represents an upper bound on the maximum disparity and is set to 1/3 of the image size. All corner pairs that pass a minimum correlation threshold are then filtered using a symmetry test, which requires the correlation be maximum in both directions. This filters out half of the matches and forces the remaining matches to be one-to-one.

Disparity gradient filter. The next step is to perform local filtering of these matches. We use a relaxation-like process based on the concept of *disparity gradient* [29] which measures the compatibility of two correspondences between an image pair. It is defined as the ratio

$$d_{\mathrm{gr}} = |\mathbf{d_a} - \mathbf{d_b}|/|\mathbf{d_{a+b/2}}|, \tag{6}$$

where $\mathbf{d_a}$ and $\mathbf{d_b}$ are the disparity vectors of two corners, $\mathbf{d_{a+b/2}}$ is the vector that joins midpoints of these disparity vectors, and $|\cdot|$ designates the absolute value of a vector. The smaller the disparity gradient, the more the two correspondences in agreement with each other. This filter is very efficient. At a very low computational cost, it removes a significant number of incorrect matches.

Using random sampling. The final step is to use the filtered matches to compute the fundamental matrix. This process must be robust, since it still cannot be assumed that all filtered correspondences are correct. Robustness is achieved by using a random sampling algorithm. This is a 'generate and test' process in which a minimal set of correspondences, the smallest number necessary to define a unique fundamental matrix (seven points), are randomly chosen [30,31]. The number of random samples is dynamically adjusted to a given confidence level.

A fundamental matrix is then computed from this best minimal set using Eq. (3) by the SVD algorithm [32]. The set of all corners that satisfy this fundamental matrix, in terms of Eq. (5), is called the support set. The fundamental matrix with the largest support set is used in stereo-tracking. Before it can be used, however, it needs to be evaluated, because robust and precise tracking is possible only under the assumptions that the computed fundamental matrix correctly represents the current stereo-setup.

Evaluating the quality of calibration. The evaluation is done in two ways: analytically—by examining the size of the support set, and visually—by visual examination of the epipolar lines.

It has been empirically obtained that for the fundamental matrix to be correct it should have at least 35 matches in the support set. If their number is less, it means that either (i) there were not enough visual features present in the images, or (ii) cameras are located too far from each other. In this case, cameras have to be repositioned or some extra objects, in addition to the head, should be added in the camera field of view and the calibration procedure should be repeated.

3.3. Visual examination and feature learning

At this stage, the user is required to manually choose the features to be tracked. At least three features have to be selected to allow 3D tracking. One of these features must be the nose feature as defined in Section 2. Other features may include conventional edge-based features such as inner corners of the brows and corners of the mouth. These features are not invariant to the 3D motion of the head. Therefore, in order to make the tracking of these features robust, the rigidity constraint, which relates their positions to the nose position, is imposed. This constraint is computed while selecting the features.

By clicking with a mouse on a feature in one image, the user has to verify that the epipolar line generated in the other image according to Eq. (4) passes correctly through each feature. Fig. 8 shows the epipolar line (on the right side) corresponding to the tip of the nose (on the left side). As can be seen, it passes correctly through the nose, thus verifying the correctness of the fundamental matrix and presenting a useful constraint for locating the nose in the tracking stage.

It is the major advantage of stereo-tracking that it provides an additional epipolar constraint which ensures that the observed features belong to a rigid body.

If generated epipolar lines do not pass through the selected features within a given precision, then the fundamental matrix has to be recomputed. Once it is

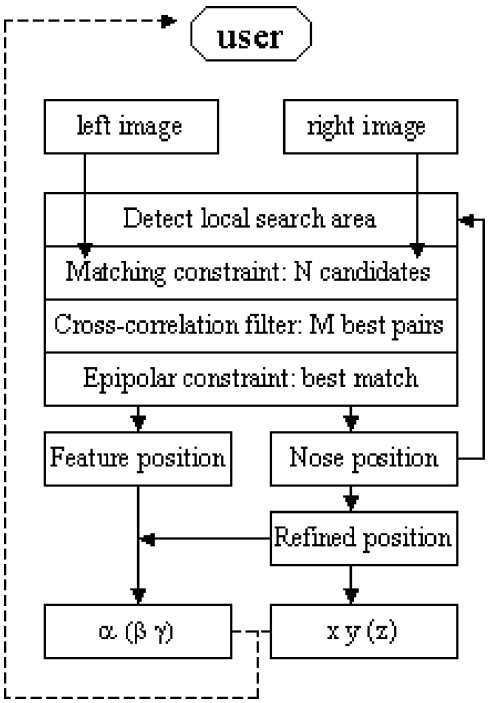

Fig. 8. Overview of the tracking procedure for the face-tracking-based user interfaces.

computed with the sufficient precision (e.g. less than 5 pixels, as in our experiments), the template vectors of both the selected features and their best matches in the other image lying on the epipolar lines are stored. At the same time the rigidity constraint is computed.

3.4. Tracking procedure

The hierarchy of all face processing tasks performed by the perceptual vision system is presented in Section 4. Below we describe the last of these tasks—face-tracking, as it is performed in the case of tracking with two cameras. It consists of the following four steps (see also Fig. 8).

Step 1. For each tracked facial feature, the area for the local search is obtained using the rigidity constraint and the knowledge of the nose tip position. When the feature is the nose itself, the local search area is set around the previous position of the feature. If the previous position of the nose is not known, then the search area is set to the area where the face was detected.

Step 2. A set of N best candidate matches $\{\mathbf{u_f}\}$ for each feature is generated in both images using the matching constraint by scanning the local search area with template vector $\vec{V}_f$ learnt in the training. In order to be valid, all candidate matches are required to have the correlation with the template feature larger than a certain threshold. In our experiments, the allowable minimum correlation is set to 0.9. Normalized correlation is used.

Step 3. Out of N^2 possible match pairs of each feature between two images, the best M pairs are selected using the cross-correlation between the images. In our experiments, N is set to 10 and M is set to 5. Increasing N or M increases the processing time, but makes tracking more robust.

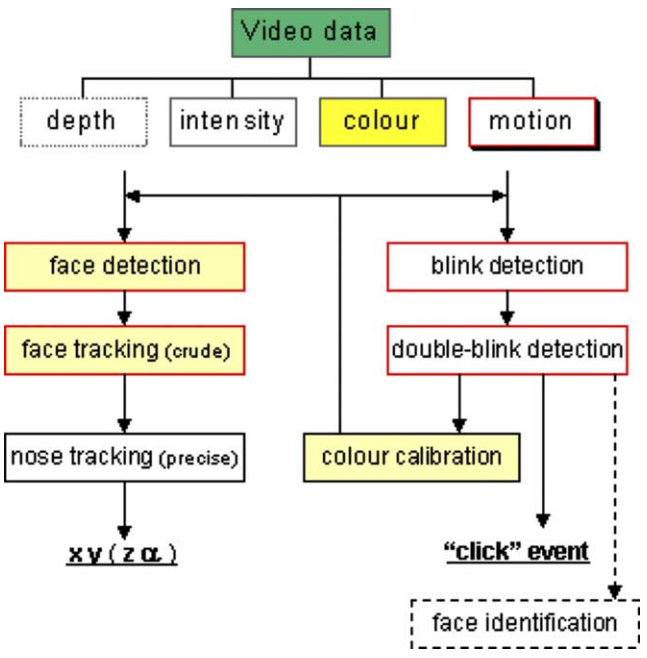

Fig. 9. Hierarchy of tasks in the *Nouse* perceptual user interface system.

Step 4. Finally, Proposition 3 is used and the match pair that minimizes the epipolar error ρ defined by Eq. (5), is returned by the stereo tracking system. If the returned match has the epipolar error less than a certain threshold, than the feature is considered successfully tracked. Otherwise, the feature is considered mistracked. The value of the maximum allowable epipolar error $\rho_{\max}$ depends on the quality of stereo self-calibration and should be determined during the learning stage by observing the epipolar lines at different parts of the image. In the experiments presented in this paper it is set equal to 5 pixels.

The position of the nose feature is then refined as described in Section 2 and its position (x, y) is returned. The z coordinate of the face can be calculated, if required, using the essential matrix E as in Ref. [24]. The distance between the cameras can be either measured by hand or set equal to one. In the latter case, the reconstruction will be known up to a scale factor, which is sufficient for many applications.

Other features provide the information about α (roll), β (tilt), γ (pan) rotation of the face. It should be noted, however, that with low resolution cameras, such as USB webcams, the pan and tilt rotation of the head (β, γ) cannot be retrieved well. This is because of the high error in depth calculation due to the image quantization and warping errors, which can be as high as 10% of the measured depth distance [33]. Nevertheless, retrieving other four degrees of freedom of the face are quite sufficient for designing many face-tracking-based hand-free user interfaces.

4. Nouse perceptual vision systems

Nouse, which stands for *N*ose as M*ouse*, is the name of the perceptual vision technology which uses the ideas described above to enable vision-based hands-free interaction with computer. The architecture of a perceptual user interface system built with the Nouse technology is shown in Fig. 9.

The system takes a video sequence as an input, and splits it into the channels corresponding to the motion, colour and intensity components of video. As discussed in Ref. [44], this is how video information is processed by biological vision systems, and this is also, in fact, how face processing tasks are commonly approached by computer scientists (from Table 1), which shows the representative work done in the area, we see that most solutions described in the literature are single-channel solutions.

The first tasks to perform by the system are face segmentation and detection. They provide the initial estimate of where a face, if any, is located in video. Originally, these tasks have been assigned to the colour and motion channels. The colour channel uses the combination of the perceptually uniform colour space (UPS) with the non-linearly transformed YCrCb colour space [34,35] to compute the binarized skin image, while the motion channel uses the non-linear change detection [36] to obtain

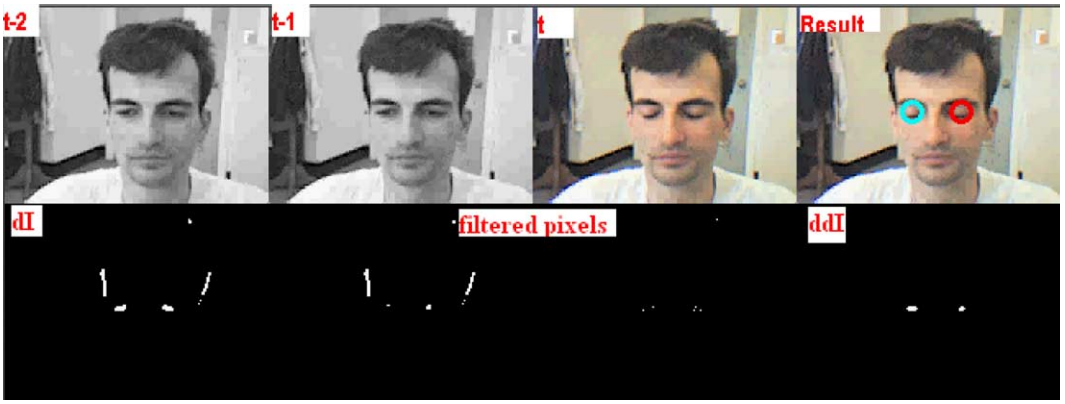

Fig. 10. Detection of eye blinks using the first order (bottom left image) and the second order change (bottom right image) detection. The pixels change due to head motion are shown at the bottom center images.

the binary change image. Analyzing the first and second moments of these binary images yields the initial estimate for size and location of skin coloured and moving regions in video. Presently, we also use the intensity channel, which can locate front-faced horizontally aligned faces using the recently proposed public domain face classifier trained on binary Haar-like binary wavelets [43].

After a face has been detected by either of the channels, the tracking of the nose feature proceeds as described in the previous sections.

The video is processed at the 160×120 resolution. This resolution allows one to perform video processing in real-time and, as mentioned above, is quite sufficient for many face processing tasks. In particular, from Table 1 we see that for most common setups when a camera is mounted in front of the user's face on top of the computer monitor, which result in the face occupying more than one quarter of the image, one can perform such tasks as nose tracking and eye localization from blinking.

4.1. Blink detection for 'clicking' and initialization

For face-operated perceptual user interfaces to be fully hands-free operational, besides tracking the face position, they should be able to detect a facial expression event, which a user would use in order to send a binary 'on/off' command, analogous to the mouse 'click' event (see Figs. 1 and 11).

In the Nouse technology, such a clicking ability is performed by detecting double (or multiple) eye blinks of the user. It is the motion channel which performs this task.

It is known that detection of eye blinks from eye lid motion is difficult when the head is not still, because there are also many pixels which move around the head boundary, mouth, brows and other parts of the face. This problem can be significantly alleviated, however, if instead of the commonly used first-order change, computed by using two frames from the video sequence, the second order change (i.e. the change of the change) proposed in Ref. [41], computed from three consecutive video frames is used. This allows one to discriminate the local (most recent) change in image, such as blink of the eyes, from the global (long lasting) change, such as the motion of head (see Fig. 10).

As opposed to a single blink, which happens involuntary, a doubleblink or several blinks performed within short lapse of time, is a deliberate action that can be easily performed by most people. Another big advantage of using this facial visual cue, which we termed the *Doubleblink* event, is that it provides an additional spatio-temporal constraint for blink verification. This constraint becomes very valuable for the situations, when false positives are undesirable.

Besides providing a mechanism for switching the Nouse on and off, detection of blinks also allows one to initialize the nose template. One way of doing it is to ask a user to position his nose in the middle of the image and then to Doubleblink. The other is to ask a user to blink several times

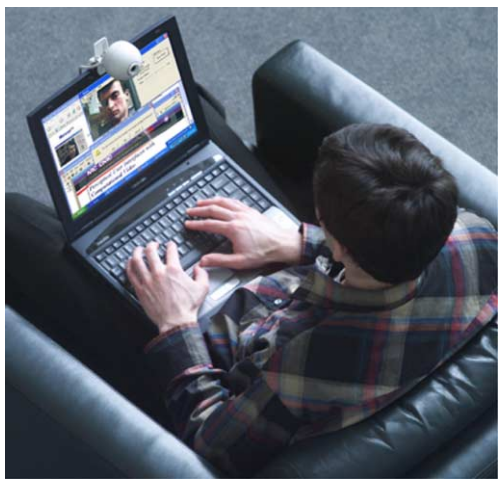

Fig. 11. Vision-based perceptual user interface at work: a laptop with a webcam mounted on its top allows a user to use Nouse and Doubleblink to select hands-free a 'Yes' button in a pop-up 'Continue? ‖ Yes | No | Cancel ‖ window.

and then to detect the nose automatically in the area under the eyes from the set of prestored templates.

If a user feels at any time that the tracking quality has deteriorated, he can invoke the initialization routine by performing several blinks. Our experiments show, however, that under approximately the same lighting condition, users rarely need to reinitialize the Nouse. Even more, in most cases, the same nose template can be used for different users.

Let us also mention that the ability to retrieve the precise positions of the eyes from blinking allows our systems to retrieve the entire face. Using the associative memory principles, the retrieved faces can be memorized and recognized when needed to enhance the perceptual ability of the vision system [44].

Finally, eye detection allows one to make colour-based face detection more robust, by recomputing the skin colour model in the area below the eyes each time the person doubleblinks.

4.2. Demo programs

In order to show the potential of the Nouse perceptual vision technology, we have applied them to a few applications. While the paper presents the snapshots of our experiments, full mpeg videos are available at our website [12]. The website also provides the binary code of our programs, so that they can be evaluated by the public. In order to run the programs, a user will only need to have one or two USB webcams connected to a computer. In our experiments, we used Intel USB web-cameras. Other USB webcams, such as Logitech, Creative and 3Com, were also tried. Accessing and synchronizing the images is done using the DirectX interface. Image preprocessing is done using the Intel Open Source CV library [45]. On a Pentium IV 2 GHz processor, all processing takes less than 50 ms per frame.

Navigation in Windows. The Nouse technology makes it possible to use the nose to control the Windows cursor (see Fig. 11). One of two control modes can be used for this purpose. Joystick mode operates in a fashion similar to an analog joystick: offsetting the nose from the center of the screen causes the mouse cursor to move in a similar direction. The speed of the cursor is determined based on the offset amount. Mouse mode attempts to mimic an actual mouse more closely: offsetting the nose from the center position causes the cursor to move in a similar direction, but the movement of the nose back to the center position has no effect. This allows the user to simulate the common continuous dragging process that computer users frequently perform, when, for example, run out of drag space on their mouse pad.

It has been found that for screen resolution of 640×480, Nouse provides sufficient robustness and precision for users to be able to select an item in Windows menu hands-free using the nose. For larger resolution, however, because the size of the computer screen (in pixels) is much larger than the size of the image in which the nose moves, it takes considerable amount of time and energy for the user to get to the desired item.

NousePaint. This program is written to allow users to paint hands-free using their nose, thinking of it as a pen or a chalk. Having been tried by many users, it demonstrated that robustness and precision of Nouse allows one to write quite long phrases and to draw hands-free as fast as it is convenient. For example, it took 25 s to write a phrase of Fig. 12. Switching the drawing on and off can be done by either *Doubleblink* or key stokes.

Aim-n-shoot BubbleFrenzy game. This game traditionally involves simple left/right mouse movements or key presses in order to aim a bubble turret in the desired direction. Once properly targeted, the user then presses the space bar to launch a coloured bubble in the turret direction. We used Nouse to replace mouse, allowing a player to point the direction of shooting with the nose. The precision of Nouse is such that very slight rotations of head left and right are sufficient to cover the entire 180° range of the turret aim. Fig. 13 shows a user playing *BubbleFrenzy* with Nouse, using one of the two webcams attached to the computer.

The users of the game were given a choice of switching back and forth between the Nouse and the standard mouse modes. This helped them to evaluate the new hand-free interface technology. Among more than 50 people who tried the game, it has been agreed that playing the game hands-free with Nouse is not only more fun, but is also less tiring than playing the game with mouse. Some users experienced

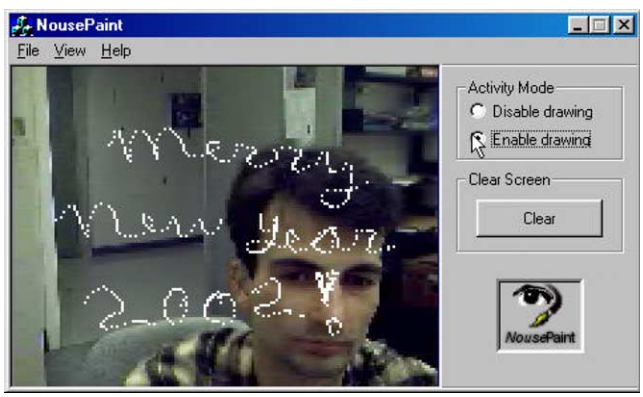

Fig. 12. A new year greeting written hands-free using *NousePaint*.

Fig. 13. Setup for stereotracking. A user plays an aim-n-shoot game aiming with his nose.

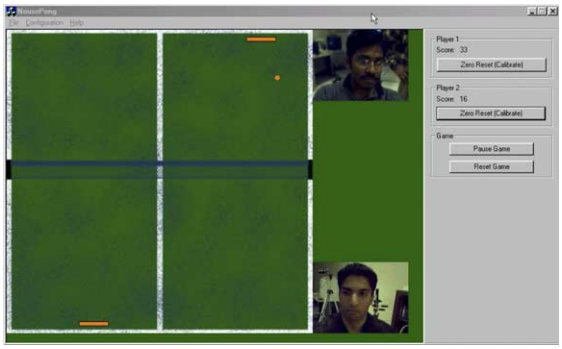

Fig. 14. Two users are playing a pong game using their heads to bounce a ball.

severe wrist fatigue when they played the game holding a mouse in their hands for longer than 15 min. This does not happen with Nouse. Using the nose to aim the turret was found very natural, while the precision of aiming with the nose was as good as with a standard mouse.

NousePong. This game has been written to demonstrate another advantage of vision-based interfaces, which is multiple-user interaction. Two web-cameras are mounted to face the heads of the players as shown in Fig. 14. Because of the robustness of the convex-shape nose feature to rotation, the cameras do not have to be aligned with the horizon or each other. The game consists in bouncing the ball back and forth over a virtual table using the head. Each camera tracks the motion of a player's head in order to convert it to the motion of the paddle. This game showed that with the aid of the Nouse technology computer games may become much more physical.

4.3. 3D hands-free interfaces

The tracked facial features provide information about the position and the orientation of the head with respect to the cameras. This information can be used to control a 2D object on the screen, as presented above, or it can be used to control a 3D object. The results below demonstrate the applicability of the stereotracking technique proposed in Section 3 for designing 3D hands-free user interfaces.

Fig. 15 shows a virtual man which is controlled by the motion of the user's head. The scale of the man is proportional to the distance between the features and the camera and the roll rotation of the man coincides with the roll rotation of the head. Yellow boxes around the features show the facial features used for tracking, which are the nose and the inner corners of the brows. These features are found to be most optimal for recovering the orientation of the head.

The importance of having the convex-shape nose feature among the features becomes very apparent, when by switching the rigidity constraint on and off we observe that in many cases, e.g. such as shown in the figure, brow features are lost.

In a similar fashion, by switching on and off the epipolar constraint, we were also able to observe that

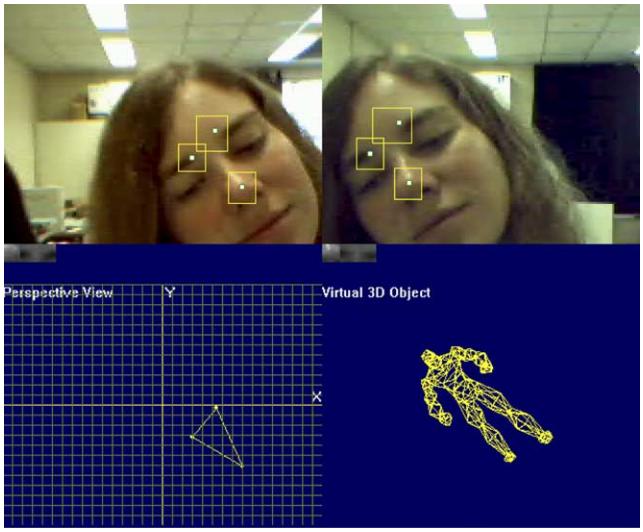

Fig. 15. Stereotracking with two web-cameras: the orientation and scale of the virtual man is controlled by the position of the observed face.

using two cameras instead of one does make tracking more robust; with the epipolar constraint switched on, the detected features always lie on a face.

We have experimented with the different baselines between the cameras, and the distance of about 10–20 cm appears to be the most optimal. Larger baselines result in too few matches needed to compute the fundamental matrix, while smaller baselines cause large epipolar errors.

4.4. Mistracking problem

One has to realize that, unlike hand-operated interfaces, hands-free interfaces do not have a feedback connection. By holding a mouse, a user not only controls the program, but he also keeps the knowledge of where the mouse is. No matter how robust the perceptual user interface is, it can loose the user; it might be even more appropriate to say that a user looses the interface. This problem, which can happen to Nouse too, can be resolved, however, by providing users with the missing feedback, as shown in Figs. 1 and 8. A user can get this type of feedback from the image captured by the video-camera. By visually verifying that he is still observed by the camera and that the face is still being tracked properly, a user can ensure that he has a good 'grasp' of Nouse. This is done in NousePaint nose-drawing program and also in NousePong game, where the images captured with both cameras are shown on the screen at all times.

Another way of having the desired feedback is to know where the camera is, and as soon as a user feels that he has lost control with Nouse, he puts his face in front of the camera and sends a signal to Nouse, e.g. by *Doubleblink* or by using a keyboard, to reset the search of the face.

5. Conclusions

Because of the low prices and the ease of installation, USB cameras have become very popular nowadays. This paper presented a few computer vision techniques which allow one to use such cameras for designing perceptual user interfaces.

We show that a human face possesses a very unique feature—the nose; because of its prominence and convex-shape, the nose feature can be seen at all times during the interaction with the computer screen, regardless of the orientation of the head and the camera. This makes it possible to track faces both robustly and precisely even with low quality cameras such as USB webcams. This also allows us to extend robust face-tracking from 2D to 3D. We have proposed a stereotracking technique, which makes use of convex-shape nose feature and the epipolar constraint to build 3D face-tracking-based user interfaces which do not require expensive dedicated stereo setups.

Combined with other perceptual mechanisms, such as blink detection, our face-tracking technology allows one to build complete intelligent hands-free alternatives, or extensions, to conventional tangible input devises such mouse or joystick.

Finally, as noted in Ref. [44], the systems presented in the paper can be combined with face recognition techniques, in order to improve the performance of the latter.

Acknowledgements

BubbleFrenzy game is provided by Extended Reality Ltd. The authors are also thankful to Shazad Mallik for his help in developing the programs.

References

[1] E. Hjelmas, B.K. Low, Face detection: a survey, Computer Vision and Image Understanding 83 (2001) 236–274.
[2] M. Yang, N. Ahuja, D. Kriegman, Detecting faces in images: a survey, IEEE Transaction on Pattern Analysis and Machine Intelligence 24 (1) (2002) 34–58.
[3] M. Turk, C. Hu, R. Feris, F. Lashkari, A. Beall, TLA based face tracking, in: Proceedings of International Conference on Vision Interface (VI'2002), Calgary, May, 2002, Online at www.visioninterface.net/vi2002.
[4] D.O. Gorodnichy, On importance of nose for face tracking, in: Proceedings of IEEE International Conference on Automatic Face and Gesture Recognition (FG 2002),Washington, DC, May 20–21, 2002, pp. 188–196.
[5] M. Xu, T. Akatsuka, Detecting head pose from stereo image sequence for active face recognition, in: Proceedings of IEEE International Conference on Automatic Face and Gesture Recognition (FG'98), 1998.
[6] G. Loy, E. Holden, R. Owens, 3D head tracker for an automatic lipreading system, in: Proceedings of Australian Conference on Robotics and Automation (ACRA 2000), Melbourne, Australia, August, 2000.
[7] Y. Matsumoto, A. Zelinsky, An algorithm for real-time stereo vision implementation of head pose and gaze direction measurement, in: Proceedings of IEEE International Conference on Automatic Face and Gesture Recognition (FG 2000), 2000.
[8] R. Newman, Y. Matsumoto, S. Rougeaux, A. Zelinsky, Real-time stereo tracking for head pose and gaze estimation, in: Proceedings of IEEE International Conference on Automatic Face and Gesture Recognition (FG 2000), 2000.
[9] M. Picardi, T. Jan, Recent advances in computer vision, The Industrial Physicist 9 (1) (2003).
[10] La nariz utilizada como mouse, in: La Nacion Line, Argentina, June 3, 2003.
[11] Esperanca contra a excusao, in Planeta Digital, Brasil, August 6, 2003, pp. 4–5.
[12] Website, Nouse perceptual vision technology, 2001, http://www.cv.iit.nrc.ca/research/Nouse (www.perceptual-vision.com).
[13] L. Itti, C. Koch, Computational modeling of visual attention, Nature Reviews Neuroscience 2 (3) (2001) 194–203.
[14] L.C. De Silva, K. Aizawa, M. Hatori, Detection and tracking of facial features, in: SPIE Visual Communications and Image Processing'95 (VCIP'95), vol. 2501, 1995, pp. 2501/1161–2501/1172.
[15] A. Gee, R. Cipolla, Fast visual tracking by temporal consensus, Image and Vision Computing 14 (2) (1996) 105–114.
[16] A. Colmenarez, B. Frey, T. Huang, Detection and tracking of faces and facial features, in: ICIP Proceedings, 1999.
[17] J. Yang, R. Stiefelhagen, U. Meier, A. Waibel, Real-time face and facial feature tracking and applications, in: Proceedings of AVSP'98, Terrigal, Australia, 1998, pp. 79–84.
[18] T. Darrell, B. Moghaddam, A. Pentland, Active face tracking and pose estimation in an interactive room, in: IEEE Conference on Computer Vision and Pattern Recognition, 1996, pp. 67–72.
[19] D.O. Gorodnichy, W.W. Armstrong, X. Li, Adaptive logic networks for facial feature detection, in: Proceedings of International Conference on Image Analysis and Processing (ICIAP'97), vol. II, LNCS-131, Springer, Berlin, 1997, pp. 332–339.
[20] Y.-L. Tian, T. Kanade, J.F. Cohn, Recognizing action units for facial expression analysis, Pattern Analysis and Machine Intelligence 23 (2) (2001) 97–115.
[21] J. Shi, C. Tomasi, Good features to track, in: IEEE Conference on Computer Vision and Pattern Recognition (1994) 593–600.
[22] B.K.P. Horn, Understanding image intensities, Artificial Intelligence 8 (1977) 201–231.
[23] W.W. Armstrong, M.M. Thomas, Adaptive logic networks, Handbook of Neural Computation, Section C1.8, IOP Publishing and Oxford University Press, New York, 1996, ISBN 0 7503 0312 3.
[24] R. Hartley, A. Zisserman, Multiple view geometry in computer vision, Cambridge University Press, Cambridge, 2000.
[25] G. Roth, A. Whitehead, Using projective vision to find camera positions in an image sequence, in: Proceedings of International Conference on Vision Interface (VI 2000), Montreal, May, 2000, pp. 87–94.
[26] Website, PVT (Projective Vision Toolkit), http://www.cv.iit.nrc.ca/research/PVT, 2000.
[27] S.M. Smith, J.M. Brady, Susan—a new approach to low level image processing, International Journal of Computer Vision (1997) 45–78.
[28] Z. Zhang, R. Deriche, O. Faugeras, Q.-T. Luong, A robust technique for matching two uncalibrated images through the recovery of the unkown epipolar geometry, Artificial Intelligence Journal 78 (1995) 87–119.
[29] R. Klette, K. Schluns, A. Koschan, Computer vision: three-dimensional data from images, Springer, Berlin, 1996.
[30] P.J. Rousseeuw, Least median of squares regression, Journal of American Statistical Association 79 (388) (1984) 871–880.
[31] G. Roth, M.D. Levine, Extracting geometric primitives, Computer Vision, Graphics and Image Processing: Image Understanding 58 (1) (1993) 1–22.

[32] H.P. Press, B.O. Flannery, Numerical Recipes in C, Cambridge University Press, Cambridge, 1988.
[33] J. Rodriguez, J. Aggarwal, Stochastic analysis of stereo quantazation error, IEEE Transactions on Pattern Analysis and Machine Intelligence 12/5 (1990) 467–470.
[34] J.-C. Terrillon, M. Shirazi, H. Fukamachi, S. Akamatsu, Comparative performance of different skin chrominance models and chrominance spaces for the automatic detection of human faces in color images, in: Proceedings of Fourth International Conference on Automatic Face and Gesture Recognition (FG 2000), 2000.
[35] R.-L. Hsu, M. Abdel-Mottaleb, A. Jain, Face detection in color images, IEEE Transactions on Pattern Anaylis and Machine Intelligence 24 (5) (2002) 696.
[36] E. Durucan, T. Ebrahimi, Change detection and background extraction by linear algebra, in: IEEE Proceedings on Video Communications and Processing for Third Generation Surveillance Systems 89 (10) (2001) 1368–1381.
[37] R. Feraund, O.J. Bernier, J.E. Viallet, M. Collobert, A fast and accurate face detector based on neural network, IEEE Transactions on Pattern Analysis and Machine Intelligence 23 (1) (2001).
[38] H. Rowley, S. Baluja, T. Kanade, Neural network-based face detection, IEEE Transactions on Pattern Analysis and Machine Intelligence 20 (1) (1998) 23–38.
[39] B. Heisele, T. Poggio, M. Pontil, Face detection in still gray images, MIT technical report (ai.mit.com), 2000.
[40] G. Bradski, Computer vision face tracking for use in a perceptual user interface, Intel Technology Journal (2) (1998).
[41] D.O. Gorodnichy, Second order change detection, and its application to blink-controlled perceptual interfaces, in: Proceedings of IASTED Conference on Visualization, Imaging and Image Processing (VIIP 2003), Benalmdena, Spain, September 8–10, 2003, pp. 140–145.
[42] M. Pantic, L. Rothkrantz, Automatic analysis of facial expressions: the state of the art, IEEE Transactions on PAMI 22 (12) (2000) 1424–1445.
[43] G. Shakhnarovich, P.A. Viola, B. Moghaddam, A unified learning framework for realtime face detection and classification, in: International Conference on Automatic Face and Gesture Recognition (FG 2002), USA, 2002, pp. 10–15.
[44] D.O. Gorodnichy, Face recognition from video, in: Proceedings of International Conference on Audio and Video-Based Biometric Person Authentication (AVBPA'03), LNCS 2688, Guildford, UK, 2003, pp. 505–514.
[45] G. Bradski, OpenCV: examples of use and new applications in stereo, recognition and tracking, in: Proceedings of Conference on Vision Interface (VI'2002), Calgary, May, 2002, p. 347.