Detecting and Tracking of 3D Face Pose for Human-Robot Interaction

Fadi Dornaika
French National Geographic Institute (IGN)
94165 Saint-Mandé, France
fadi.dornaika@ign.fr

Bogdan Raducanu*
Computer Vision Center, UAB,
08193 Bellaterra, Barcelona, Spain
bogdan@cvc.uab.es

Abstract— **Faces play a major role in many HCI systems, because they represent a rich source of information. Being able to estimate the 3D face pose in real-time, we can get a clue about user's intentions or to assess which object become his/her focus of attention. This paper has two main contributions. First, we propose an automatic 3D face pose initialization scheme for our real-time tracker by adopting a 2D face detector and an eigenface system. Second, we use the proposed methods - the initialization and tracking - for controlling the orientation of an AIBO camera. We show how the changes in user's face movement can be imitated by the robot's camera and how it can be applied to map an indoor scene.**

Keywords: face detection, 3D face pose estimation, real time 3D face pose tracking, human-robot interaction, AIBO robot

I. INTRODUCTION

The ability to detect and track human faces and facial features in video sequences is useful in a great number of applications, such as human-computer interaction and gesture recognition [1], [2]. Vision-based tracking systems represent an attractive solution since vision sensors are not an invasive technology. To this end many systems and methods have been developed. Of particular interest are vision-based markerless head and/or face trackers. Since these trackers do not require any artificial markers to be placed on the face, comfortable and natural movements can be achieved. On the other hand, building robust and real-time markerless trackers for face and facial features is a difficult task due to the high variability of the face and the facial features in videos.

In general, there are two main approaches for face-pose estimation: feature-based approaches and view-based (holistic) approaches. Feature-based approaches refer to the extraction of salient facial characteristics (eyes, nose, mouth) which are used to compute the face pose based on their spatial relations. In the past, many feature-based approaches have been proposed (e.g, [3], [4], [5], [6]).

The main drawback of the feature-based approaches is that it could happen that not all the points are visible during a video sequence. Moreover, reliable feature tracking is often difficult, since minor changes between frames can lead to very different segmentation in consecutive frames—the drift problem.

A solution to overcome the drawbacks of feature-based approaches is given by view-based approaches (appearance-based approaches), which try to analyze the whole facial appearance in order to infer the 3D face pose. To overcome the problem of appearance changes recent works on faces adopted statistical facial textures. For example, Active Appearance Models received a lot of attention recently. The Active Appearance Models have been proposed as a powerful tool for analyzing facial images [7]. Deterministic and statistical appearance-based tracking methods have been proposed [8], [9], [10]. These methods can successfully tackle the image variability and drift problems by using deterministic or statistical models for the global appearance of a special object class: the face.

Recently, we have developed a face and facial feature tracking method based on Online Appearance Models (OAMs) [11]. Unlike the Active Appearance Models, the OAMs offer a lot of flexibility and efficiency since they do not require any facial texture model that should be computed beforehand. Instead the texture model is built online from the tracked sequence.

This paper extends our previous work [11] in two directions. First, we propose a method for the automatic initialization of the 3D face pose. In [11], the initialization is performed manually. Second, we use the proposed tracker in a human-robot interaction application in which the gaze of a robotics vision sensor is controlled by the user's gaze. The proposed scheme for estimating and tracking the 3D face pose methods are automatic. The remainder of the paper is organized as follows. Section II briefly describes the deformable 3D face model that we use to create shape-free facial patches from input images. Section III states the problem we propose to solve. Section IV describes the proposed automatic 3D face pose initialization from one single image. Section V presents the real-time face and facial action tracker. Section VI describes the proposed human-robot interaction scenario that is based on controlling the AIBO camera orientation through the use of the tracked user's face pose parameters. Section VII concludes the paper.

II. MODELING FACES

a) ***A deformable 3D model:*** In our study, we use the 3D face model *Candide* [12]. This 3D deformable wireframe model was first developed for the purpose of model-based image coding and computer animation. The 3D shape of this wireframe model is directly recorded in coordinate form. It is given by the coordinates of the 3D vertices $\mathbf{P}_i, i = 1, \ldots, n$ where n is the number of vertices. Thus, the shape up to a global scale can be fully described by the $3n$-vector $\mathbf{g}$; the concatenation of the 3D coordinates of all vertices $\mathbf{P}_i$. The

*This work is supported in part by MEC Grant TIN2006-15308-C02 and CONSOLIDER-INGENIO 2010 (CSD2007-00018), and the Ramon y Cajal research program.

vector $\mathbf{g}$ is written as:

$$\mathbf{g} = \mathbf{g}_s + \mathbf{A}\,\boldsymbol{\tau_a} \tag{1}$$

where $\mathbf{g}_s$ is the static shape of the model, $\boldsymbol{\tau_a}$ the animation control vector, and the columns of $\mathbf{A}$ are the Animation Units. In this study, we use seven modes for the facial Animation Units (AUs) matrix $\mathbf{A}$. We have chosen the following AUs: lower lip depressor, lip stretcher, lip corner depressor, upper lip raiser, eyebrow lowerer, outer eyebrow raiser. These AUs are enough to cover most common facial animations. Moreover, they are essential for conveying emotions.

In equation (1), the 3D shape is expressed in a local coordinate system. However, one should relate the 3D coordinates to the image coordinate system. To this end, we adopt the weak perspective projection model. We neglect the perspective effects since the depth variation of the face can be considered as small compared to its absolute depth. Thus, the state of the 3D wireframe model is given by the 3D face pose parameters (three rotations and three translations) and the internal face animation control vector $\boldsymbol{\tau_a}$. This is given by the 12-dimensional vector $\mathbf{b}$:

$$\mathbf{b} \;=\; [\theta_x,\;\; \theta_y,\;\; \theta_z,\;\; t_x,\;\; t_y,\;\; t_z,\;\; \boldsymbol{\tau_a}^T\;]^T \tag{2}$$

Note that if only the aspect ratio of the camera is known, then the component t_z is replaced by a scale factor having the same mapping role between 3D and 2D. In this case, the state vector is given by (s denotes the scale factor):

$$\mathbf{b} \;=\; [\theta_x,\;\; \theta_y,\;\; \theta_z,\;\; t_x,\;\; t_y,\;\; s,\;\; \boldsymbol{\tau_a}^T\;]^T \tag{3}$$

*b) **Shape-free facial patches:*** A facial patch is represented as a shape-free image (geometrically normalized raw-brightness image). The geometry of this image is obtained by projecting the static shape $\mathbf{g}_s$ (neutral shape) using a centered frontal 3D pose onto an image with a given resolution. The texture of this geometrically normalized image is obtained by texture mapping from the triangular 2D mesh in the input image (see figure 1) using a piece-wise affine transform, $\mathcal{W}$ (see [12] for more details). The warping process applied to an input image $\mathbf{y}$ is denoted by:

$$\mathbf{x}(\mathbf{b}) = \mathcal{W}(\mathbf{y}, \mathbf{b}) \tag{4}$$

where $\mathbf{x}$ denotes the shape-free patch and $\mathbf{b}$ denotes the geometrical parameters. Several resolution levels can be chosen for the shape-free patches. Regarding photometric transformations, a zero-mean unit-variance normalization is used to partially compensate for contrast variations.

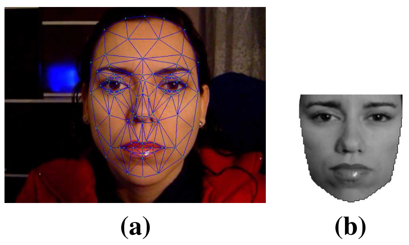


Fig. 1. **(a)** an input image with correct adaptation. **(b)** the corresponding shape-free facial patch.

III. PROBLEM FORMULATION

Given a video sequence depicting a moving face, we would like to recover, for each frame, the 3D face pose and the facial actions encoded by the control vector $\boldsymbol{\tau_a}$. In other words, we would like to estimate the vector $\mathbf{b}_t$ (3) at time t given all the observed data until time t, denoted $\mathbf{y}_{1:t} \equiv \{\mathbf{y}_1, \ldots, \mathbf{y}_t\}$. In a tracking context, the model parameters associated with the current frame will be carried over to the next frame. For each input frame $\mathbf{y}_t$, the observation is the shape-free facial patch associated with the geometric parameters $\mathbf{b}_t$. We use the HAT symbol for the tracked parameters and patches. For a given frame t, $\hat{\mathbf{b}}_t$ represents the computed geometric parameters and $\hat{\mathbf{x}}_t$ the corresponding shape-free patch, that is,

$$\hat{\mathbf{x}}_t = \mathbf{x}(\hat{\mathbf{b}}_t) = \mathcal{W}(\mathbf{y}_t, \hat{\mathbf{b}}_t) \tag{5}$$

The estimation of the initial parameters $\hat{\mathbf{b}}_1$ corresponding to the first frame will be described in Section IV—3D face pose initialization. The estimation of the current parameters $\hat{\mathbf{b}}_t$ from the previous ones $\hat{\mathbf{b}}_{t-1}$ and from the sequence of images will be presented in Section V—simultaneous face and facial action tracking.

Figure 2 depicts our proposed 3D face tracker. The initialization part relies on a 2D face detector and on a statistical facial texture. The tracking part relies on image registration based on the principles of Online Appearance Models.

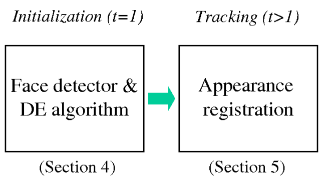


Fig. 2. A full automatic 3D face and facial feature tracker.

IV. 3D FACE POSE INITIALIZATION

As can be seen, the tracker requires the knowledge of the state vector (the 3D face pose parameters and the facial actions) associated with the first frame in the monocular video sequence. Note that even though the static shape of the user's face model is known inferring its 3D pose (face pose parameters) with respect to the camera using a single image is a challenging task since there is no correspondence between the 3D wireframe model and the raw image. Previous works adopted a simple scheme where the user is asked to align his/her face position and orientation with respect to the camera such that the actual 3D face pose becomes equal to a predefined face pose. The alignment can be controlled and assessed by the projection of some facial features (nose tip, eye corners) on their predefined locations corresponding to the predefined 3D face pose.

In our work, we relax the use of a predefined 3D face pose in order to get a very flexible 3D face tracker. In order to compute the 3D face pose parameters associated with the first frame, we will use a statistical facial texture model which is built offline. The 3D face pose parameters are then estimated by minimizing the distance between the input image texture and a learned face space—eigenface system. Reaching a global minimum can be achieved through the use of the differential evolution algorithm. In the current implementation, we assume that the first frame in the video captures a face with a neutral configuration[1]. Therefore, the state vector will reduce to six parameters describing the 3D face pose, that is, $\mathbf{b}_t = [\theta_x, \theta_y, \theta_z, t_x, t_y, s, \mathbf{0}^T]^T = [\mathbf{h}^T, \mathbf{0}^T]^T$. The vector $\mathbf{h}$ encodes the six 3D face pose parameters.

The use of a statistical facial texture model has already been used by some works in order to track the face pose parameters in monocular video sequences (e.g. [12]). However, in [12], estimating the 3D face pose parameters for the first frame is performed manually. In our work, those parameters are automatically estimated whereas most of the proposed tracking methods use a manual initialization.

A. Statistical facial texture

To build a statistical facial texture model we use our appearance based tracker [11] (outlined in Section 5). This tracker provides the time-varying 3D face pose and facial actions together with the corresponding shape-free patches $\hat{\mathbf{x}}$. Using these training patches one can easily build a statistical facial texture model. We assume that we have K shape-free patches. Applying a classical Principal Components Analysis (PCA) on the training patches we can compute the mean and the principal modes of variations. Thus, the parameters of the facial texture model will be given by the average texture $\overline{\mathbf{x}}$ and the M principal texture modes encoded by a $d \times M$ matrix denoted by $\mathbf{X}$. The columns of $\mathbf{X}$ (whose size is d) represent the principal modes. The number of principal modes M is set such that their corresponding variation is equal to a high percentage of the total variation. Note that each training patch has undergone a zero-mean unit-variance normalization.

If the model instance, $\mathbf{h}$, is a good fit to the input image (i.e., the 3D mesh is aligned with the actual 3D face pose), then the residual error between the shape-free patch $\mathbf{x}$ and its projection onto the PCA space $\widetilde{\mathbf{x}}$ is small since the remapped texture will be consistent with the statistical model of a face texture. Thus, a reliable measure of the goodness of any fit, $\mathbf{h}$, can be given by the norm of the associated residual image between the shape-free patch and its PCA approximation:

$$e(\mathbf{h}) \quad = \quad \|\mathbf{r}\|^2 = \|\mathbf{x}(\mathbf{h}) - \widetilde{\mathbf{x}}(\mathbf{h})\|^2 \tag{6}$$

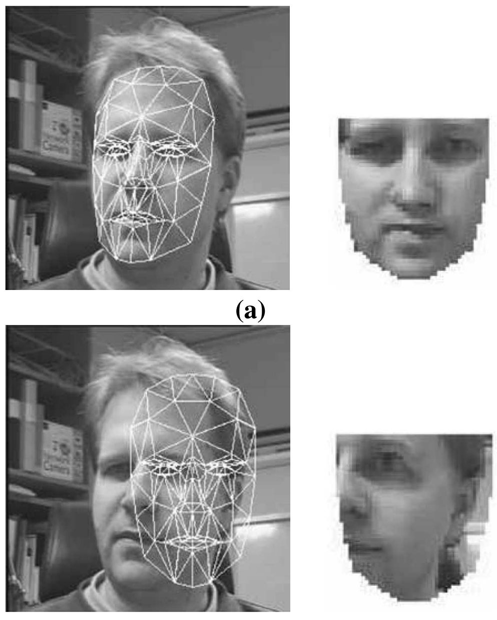
The projection of the texture $\mathbf{x}(\mathbf{h})$ onto the space spanned by the texture modes is given by:

$$\widetilde{\mathbf{x}}(\mathbf{h}) \quad = \quad \overline{\mathbf{x}} + \mathbf{X}\,\mathbf{X}^T\,(\mathbf{x}(\mathbf{h}) - \overline{\mathbf{x}})$$

[1]This assumption is very realistic since the neutral state is usually the user's emotion state.

In the literature, the error (6) is known under the name of the reconstruction error or DFFS (Distance From Feature Space).

Figure 3 illustrates the principle of the technique. Figure 3.**a** displays a good model adaptation. Both the input image and the corresponding shape-free patch are shown. In this case, the residual error (6) corresponds to a minimum. Figure 3.**b** displays a bad model adaptation. In this case, the error (6) does not correspond to the minimum. Thus, the basic idea is to estimate the 3D face pose parameters, i.e. the vector $\mathbf{h}$, such that the associated shape-free patch will be as close as possible to a facial texture.



(b)

Fig. 3. Example of model fitting. **(a)** corresponds to correct 3D face pose parameters $\mathbf{h}$. **(b)** corresponds to bad 3D face pose parameters $\mathbf{h}$.

B. 3D face pose initialization using the Differential Evolution algorithm and a face detector

As we have mentioned above, the initial 3D face pose parameters, $\mathbf{h}$, are recovered by minimizing the residual error (6):

$$\mathbf{h} = arg \min_{\mathbf{h}} e(\mathbf{h})$$

To this end, we use the Differential Evolution (DE) algorithm [13], [14] in order to minimize the error (6) with respect to the 3D face pose parameters. The DE algorithm is a practical approach to global numerical optimization that is easy to implement, reliable and fast [15].

This is carried out using generations of solutions—population. The population of the first generation is randomly chosen around a rough solution $\mathbf{h}^\star$. Thus, the first population is centered on a solution formed by $\mathbf{h}^\star = (0, 0, 0, t_x^\star, t_y^\star, s^\star)^T$. The 2D translation $(t_x^\star, t_y^\star)$ is set to the center of the rectangle found by Viola & Jones face detector [16]. The scale $s^\star$ is directly related to the size of the detected rectangle.

The optimization adopted by the DE algorithm is based on a population of N solution candidates $\mathbf{h}_{n,i}$ $(n = 1, \ldots, N)$ at iteration (generation) i where each candidate has six components. Initially, the solution candidates $\mathbf{h}_{n,0}$ are randomly generated within the provided intervals of the search space. The population improves by generating new solutions iteratively for each candidate.

We stress the fact that the use of the face detector can be relaxed on the expense of a very large range for the solutions belonging to the first population.

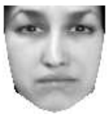


Fig. 4. The mean texture associated with 500 training images.

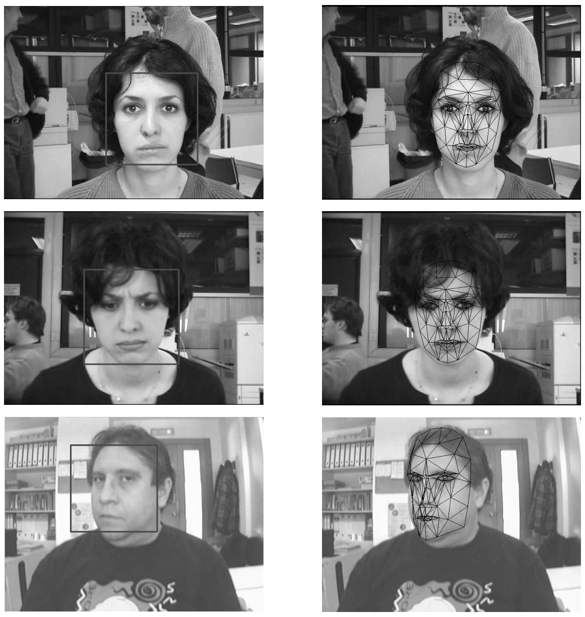


Fig. 5. Automatic 3D face pose initialization. **Left column:** Three unseen images together with the 2D face detector results. **Right column:** The corresponding 3D face pose using the Differential Evolution algorithm.

C. Results

The general scheme adopted for building a facial texture model is to use training images belonging to several subjects. However, since we are interested in a specific-user interface application, we will adopt a more flexible scheme. Within this scheme, each subject will have his/her own facial texture model. Thus, every subject is asked to perform face movements together with some facial expressions. The appearance-based tracker is then run to get the training shape-free facial patches associated with each image in the training video (the first frame in this video is manually adapted). Figure 4 illustrates the average texture obtained with a sequence of 500 training patches. In this example, we found that the first 10 principal modes corresponds to 87% of the total variation associated with the training images. The first 20 principal modes correspond to 93% of the total variation.

Figure 5 illustrates the automatic 3D face pose initialization associated with three unseen images. The PCA space was built using 500 training images. The number of principal modes was set to 20. The left column displays the original image together with the 2D face detector results. The right column displays the corresponding estimated 3D face pose using the DE algorithm. The 3D mesh is projected according to the estimated 3D face pose. As can be seen, even though the face was not in a frontal view the corresponding 3D pose parameters are correctly estimated using the Differential Evolution algorithm. Recall that the recovered parameters are richer than those provided by the 2D face detector since they depicts the 3D pose of the face with respect to the camera.

Figure 6 illustrates the evolution of the best error obtained by the Differential Evolution algorithm associated with the image shown in Figure 5 (top). The DE algorithm was run with three population sizes: $N = 60$, $N = 120$, and $N = 180$. Figure 7 illustrates the estimated 3D face pose obtained at convergence when the population size is set to 60, 120, and 180 (from left to right). As can be seen, the best fitting results were obtained when the population size was 180.

The CPU time associated with the automatic initialization ranges from one second to a few seconds. This computing time depends on many factors such as the number of generations and the number of texture modes used. It is worth noting that this initialization does not prohibit the real-time performance of the 3D face tracker (presented in Section V) since this initialization is performed only on the first video frame as the subject lets the system captures the 3D pose of his/her face.

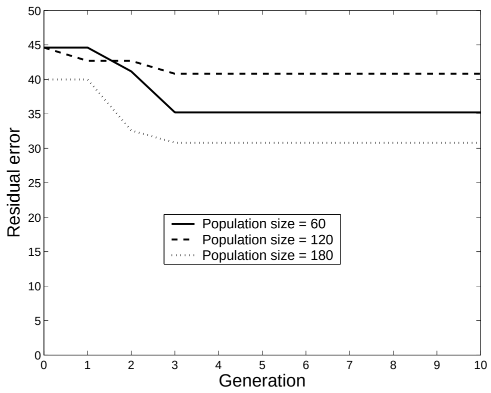


Fig. 6. The evolution of the best error obtained by the Differential Evolution algorithm associated with the image shown in Figure 5 (top). The DE algorithm was run with three population sizes: 60, 120, and 180.

V. SIMULTANEOUS FACE AND FACIAL ACTION TRACKING

In the previous section, we have addressed the initialization problem, i.e., the estimation of the state vector (the 3D face pose) for the first video frame. In this section, we will describe the tracking process, i.e., the estimation of the state vector (the 3D face pose and the facial actions) for every

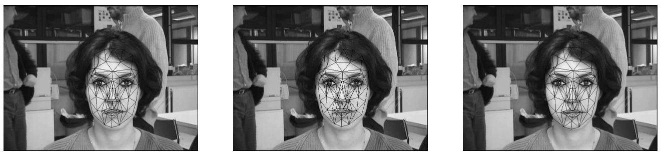


Fig. 7. Automatic 3D face pose initialization using the Differential Evolution algorithm associated with the image shown in Figure 5 (top). The estimated 3D face pose obtained at convergence when the population size is set to 60, 120, and 180 (from left to right).

subsequent video frame. Certainly, one can use the same initialization process for estimating the state vector for every frame in the video. However, using this scheme has three major disadvantages: (i) it cannot run in real-time, (ii) the 2D face detector may fail when the face undergoes under significant out-of-plane movements, and (iii) the statistical facial texture model is fixed in the sense that it does not take into account possible appearance changes during the whole video sequence. For these reasons, we will use our tracker based on Online Appearance Models [11]. This appearance-based tracker aims at computing the 3D face pose and the facial actions, i.e. the vector **b**, by minimizing a distance between the incoming warped frame and the current *shape-free* appearance of the face. This minimization is carried out using a gradient descent method. The statistics of the *shape-free* appearance as well as the gradient matrix are updated every frame. This scheme leads to a fast and robust tracking algorithm. On a 3.2 GHz PC, a non-optimized C code of the approach computes the 3D face pose and the facial actions in 50 ms.

VI. A HUMAN-ROBOT INTERACTION SCENARIO

The proposed methods for estimating the user's face pose (described in sections IV and V) are applied in a human-robot interaction scenario, for mapping an indoor environment. By mimicking user's face movement, a robot's camera can take periodically snapshots of the current perceived region. At the end of the process, the panoramic image of the region of interest is built, from the extracted snapshots by applying an image mosaicking technique [17].

The experimental setup is depicted in Figure 8. The input to the system consists of a video stream capturing user's face from a fixed camera. The corresponding pitch and yaw angles of the user's face are encoded and sent to the robot using a wireless network. Without any loss of generality, we used in our experiments Sony's AIBO robot, which has the advantage of being especially designed for interaction with persons. Thus, our application can be considered as a natural extension of AIBO's built-in behaviours. The orientation of robot's head (the robot's camera) is updated online according to the desired direction imposed by the user's face pose. Due to the motors response (as well the communication through the wireless network), there is a very small delay between the desired orientation and robot's response. The inertia associated with the motors is most visible when a change in direction has to be performed. However, if user's movement is reasonably slow, but continuous, a real-time response from the robot can be expected.

Figure 9 depicts the data flow between the real-time 3D face pose tracker and AIBO. Figure 10 illustrates the results of face movement imitation associated with a 691-frame sequence. In this video, the person looks around without any restriction. Only eight frames are shown in the figure. The left column displays the user's face pose and the right column shows the corresponding snapshot of the scene as seen by the AIBO's camera. Figure 11 illustrates a panoramic image computed from the captured individual snapshots. For this purpose, we used the AutoStitch^{TM} application [17], developed by M. Brown and D. Lowe from UBC, Canada. In a broader context, the user and the robot can be very distant from each other. This case corresponds to a telepresence application where the user is exploring a remote (dangerous or inaccessible) spot by only changing his face pose. This is the case, for instance, for rescue robots, which can be sent to areas affected by earthquakes or fires in order to perform an exhaustive search of the environment to find survivors. The use of a special display device that visualizes the scene as it is viewed by the remote camera will boost the telepresence feeling. If we assume that the orientation of the fixed camera is aligned with that of the robot camera (in its reference position) then its gaze direction will be equivalent to that of the user.

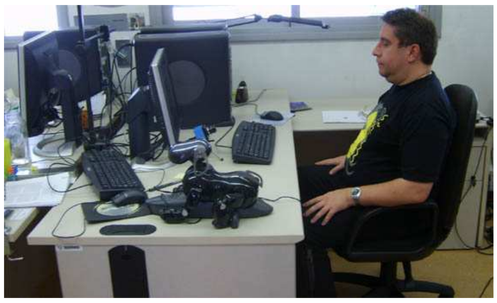


Fig. 8. The experimental setup.

VII. CONCLUSION

This paper described two main contributions. First, we proposed an automatic 3D face pose initialization scheme for a real-time appearance-based tracker by adopting a 2D face detector and an eigenface system. Second, we used the proposed methods—the initialization and tracking—for controlling the movements of AIBO's camera. Applications such as telepresence, virtual reality can directly use the proposed techniques.

ACKNOWLEDGEMENT

The authors thank Dr. Franck Davoine from CNRS, Compiegne, France, for providing the video sequences shown in Figures 5 and 7.

REFERENCES

[1] A. Bakhtari and B. Benhabib, "An active vision system for multitarget surveillance in dynamic environments," *IEEE Transactions on Systems, Man and Cybernetics, part B*, vol. 37, no. 1, pp. 190–198, 2007.

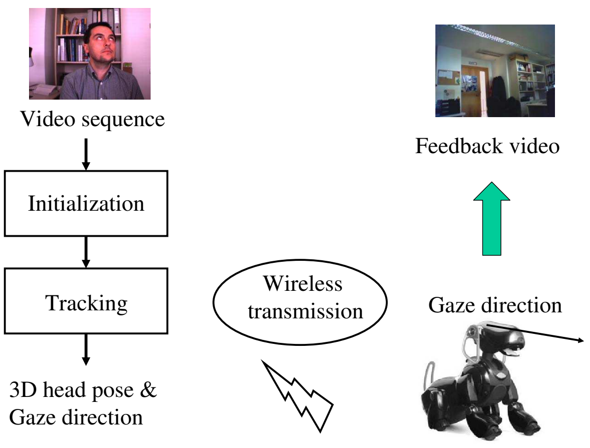


Fig. 9. Data flow for a user's gaze imitation using a monocular camera. The user's gaze direction is continuously controlling the gaze of the AIBO's camera which is capturing the remote scene.

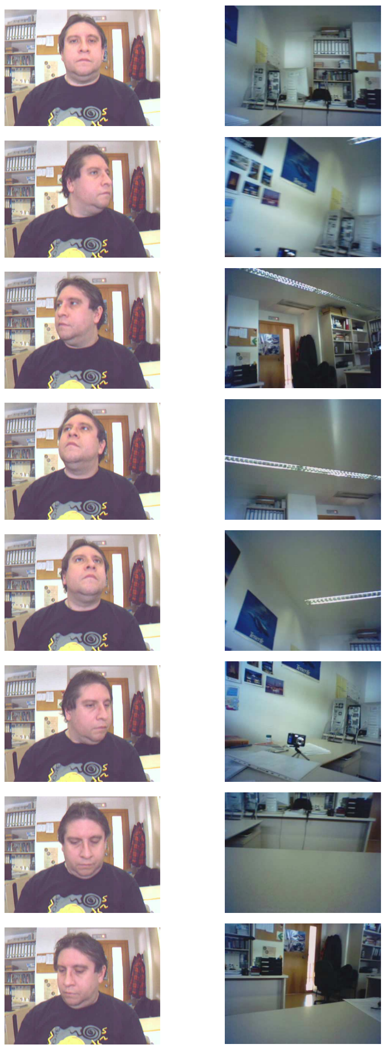


Fig. 10. **Left column:** Some input images from the original video. **Right column:** The corresponding snapshot acquired by the controlled robot's camera.

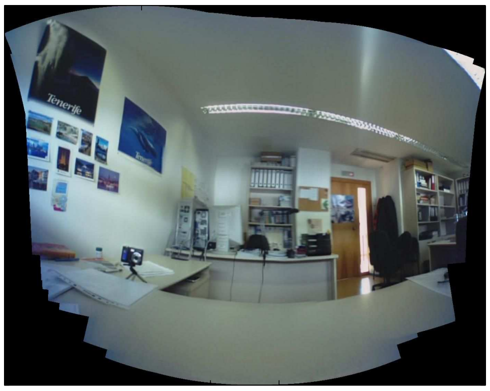


Fig. 11. A panoramic view obtained from the individual snapshots shown in Figure 10.

[2] J. Wang and E. Sung, "Study on eye gaze estimation," *IEEE Transactions on Systems, Man and Cybernetics, part B*, vol. 32, no. 3, pp. 332–350, 2002.
[3] M. Cordea, E. Petriu, N. Georganas, D. Petriu, and T. Whalen, "3D head pose recovery for interactive virtual reality avatars," in *IEEE Instrumentation and Measurement Technology Conference*, 2001, pp. 72–77.
[4] Y. Matsumoto and A. Zelinsky, "An algorithm for real-time stereo vision implementation of head pose and gaze direction measurement," in *IEEE Conference on Automatic Face and Gesture Recognition*, 2000, pp. 499–505.
[5] Y. Nakanishi, T. Fujii, K. Kiatjima, Y. Sato, and H. Koike, "Vision-based face tracking system for large displays," in *UbiComp2002*, G. Borriello and L. E. Holmquist, Eds. LNCS 2498, 2002, pp. 152–159.
[6] J. Wang, E. Sung, and R. Venkateswariu, "EM enhancement of 3D head pose estimates by perspective invariance," in *ECCV 2004 Workshop on Human Computer Interaction*, N. S. et al., Ed. LNCS 3058, 2004, pp. 187–199.
[7] T. Cootes, G. Edwards, and C. Taylor, "Active appearance models," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 23, no. 6, pp. 681–684, 2001.
[8] M. Cascia, S. Sclaroff, and V. Athitsos, "Fast, reliable head tracking under varying illumination: An approach based on registration of texture-mapped 3D models," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 4, pp. 322–336, 2000.
[9] F. Dornaika and J. Ahlberg, "Fast and reliable active appearance model search for 3-D face tracking," *IEEE Transactions on Systems, Man and Cybernetics, part B*, vol. 34, no. 4, pp. 1838–1853, 2004.
[10] I. Matthews and S. Baker, "Active appearance models revisited," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 135–164, 2004.
[11] F. Dornaika and F. Davoine, "On appearance based face and facial action tracking," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 16, no. 9, pp. 1107–1124, September 2006.
[12] J. Ahlberg, "Model-based coding: Extraction, coding, and evaluation of face model parameters," Ph.D. dissertation, No. 761, Linköping University, Sweden, September 2002.
[13] S. Das, A. Konar, and U. Chakraborty, "Two improved differential evolution schemes for faster global search," in *Genetic and Evolutionary Computation*, 2005.
[14] R. Storn and K. Price, "Differential evolution – A simple and efficient heuristic for global optimization over continuous spaces," *Journal of Global Optimization*, vol. 11, pp. 341–359, 1997.
[15] K. V. Price, J. A. Lampinen, and R. M. Storn, *Differential Evolution: A Practical Approach To Global Optimization*. Springer, 2005.
[16] P. Viola and M. Jones, "Robust real-time object detection," *International Journal of Computer Vision*, vol. 57, no. 2, pp. 137–154, 2004.
[17] M. Brown and D. Lowe, "Automatic panoramic image stitching using invariant features," *International Journal of Computer Vision*, vol. 74, no. 1, pp. 59–73, 2007.