A Robust Composite Metric for Head Pose Tracking using an Accurate Face Model

Philippe Phothisane*, Erwan Bigorgne**, Laurent Collot**, Lionel Prévost***

***Abstract*— We present a method for full-motion recovery of the head pose from a monocular video input based on an accurate head model and textures. We build our face model using a distribution of high resolution 3D face scans. The cost of computations makes us select parts of this full model. To address the difficult task of initializing the model position and tracking its motion, we use a composite metric using face texture samples from three different face databases, following a positive face detection. The algorithm is subsequently able to track the head in a wide pose range with great accuracy. We also provide test video sequences with an independent accurate ground truth (with estimated RMS error of 0.1 degrees).**

I. INTRODUCTION

Head pose tracking is used in many applications, from recreative purposes such as augmented reality, to safety protocols like face recognition, but also human-computer interaction.

Many approaches have been explored to retrieve the full 3 dimensional head motion. Some methods track facial features to recover the head pose. A facial feature is the description of a specific part of the face. This part, such as the eye corners, the tip of the nose or the mouth edges exist in each face. However, they have a different appearance and a different location depending on the subject. To locate and track these feature for unknown faces is a difficult task. Tong et al. based their face tracker [1] using a hierarchical Active Shape Model [2]. Landmarks are represented by Gabor wavelets and gray-level profiles [2]. Many other 2D model-based head pose trackers exist. However, assuming all facial features are visible constrains these trackers' pose and scale ranges. 3-dimensional approaches were developed to include much larger head rotations and scaling.

These 3D head models can be relatively simple geometrical shapes like planes [3], cylinders [4], [5] or ellipsoids [6], [7]. More complex 3D models use actual face shapes. This shape can have a relatively raw appearance when only a hundred vertices describe the face, or have a photo-realistic look. These models are called 3D Active Appearance Models [2], [8] or 3D Morphable Models (3DMM) [9], [10].

Basu et al. [7] developed a system aimed to retrieve the rigid motion of the head with an ellipsoid model. The full 3D model permits non visible points of the head to be disqualified in the tracking process. The texture model was described using optical flow, while the optimal pose parameters search used the simplex algorithm.

[4], [5], proposed a similar approach using a cylindrical head model (CHM). La Cascia et al. chose to optimize illumination and motion parameters to retrieve the pose. Xiao et al. applied a form of the iteratively re-weighted least squares [11] to update their texture template, considering possible occlusions and illumination shifts.

According to [6], the ellipsoid model is a better fit for the curve of the forehead and thus provides better results w. r. t. pitch rotations compared to the cylindrical head model. They also added a particle filter and an online appearance model to adapt to the short-term and long term appearance changes during a video sequence.

[10] presents a head tracker using Candide-3, a more complex deformable head model, with main vertices matching facial feature points. The deformations include facial expressions as hand-made shape modes.

We believe a 3D head model extracted from actual human faces will always be a better fit than relatively simple shapes. This is why we propose an alternative method for head pose tracking. Although our approach is based on [7], [4], [5] publications, we use an accurate head model (AHM) extracted from real faces scans. As of today, an accurate face model such as the Basel Face Model (BFM) [12] is not appliable in its original form for head pose tracking implementations. The number of vertices implies a high dimensionality and thus calculations which cannot be handled with common hardware.

However, we propose to decimate the initial model into a simpler one, while keeping specific vertices matching face features such as eyes corners or the tip of the nose. This method decreases the aforementioned high dimensionality and using this decimated model for pose tracking becomes viable. The other main issue with accurate face models is that they imply too many degrees of freedom. Fitting the head model to the subject without any prior knowledge of its appearance is a challenging task. We propose to use a composite metric initially introduced in [13] to model, detect and recognize faces. This metric is learned on the visual appearance of faces to converge more robustly, even on unknown subjects.

We also produced video sequences containing an accurate ground truth using a motion capture process. Most available pose test videos use a magnetic sensor. While easy to use, we believe it lacks accuracy. The main reason is that magnetic interferences destabilize the measurements. Others use more accurate inertial measurement units which nevertheless suffer

This work was supported by Majority Report

* P. Phothisane is with Institut des Systèmes intelligents et de Robotique, UPMC, France phothisane@isir.upmc.fr

** E. Bigorgne & L. Collot are with Majority Report, France erwan.bigorgne,laurent.collot@majority-report.com

*** L. Prevost is with Université des Antilles et de la Guyane (UAG), France lionel.prevost@univ-ag.fr

from integration errors. This is why we propose an accurate method using a simple circular shapes detector. This paper is organized as follows. Section II will describe our method thoroughly, beginning with the 2D tracking and our method to build a sparse face model from an originally dense face model for head pose tracking purposes. We later describe Xiao et al.'s method for rigid motion recovery and how we intend to make it more robust with the composite metric. We then explain the implementation of the face deformations and finally discuss about the inverse compositional algorithm [14]. Section III provides test results on the Boston data set [4] and our video sequences set. We also present the replicable design of our motion capture process. Finally, section IV draws a conclusion.

II. METHODOLOGY

A. 2D tracking

The first stage of the overall pose tracker is a fast 2D-model-based tracker which aims to handle large scale *and/or* fast head motions. As a precise description of the algorithm is beyond the scope of this paper, we only present it briefly. In [15], D. Ross *et al.* propose a particle filter implementation that allows an online learning of the target appearance : a Sequential Karhunen-Loeve [16] extracts incrementally a low dimensional basis representation of an affine-transform view of the target while it undergoes pose, illumination and appearance changes.

This tracker performs well, but as noted by the authors in the conclusions of their study, it occasionally drifts from the target object. In particular, a certain care has to be devoted to the tuning of a forgetting factor impacting both the stability and adaptivity of the tracker. Thus we modified the original algorithm in two ways :

- In order to minimize the drift, the appearance model is not anymore updated by views based on the current best particle state. It is only fed by views detected by a Viola-Jones [17] face detection routine, selected by proximity and size evidences.
- To preserve the robustness of the tracker outside of the Viola-Jones viewpoint sensitivity cone - which apex angle is typically 40°- each particle likelihood is now formed by a probabilistic mixture model that combines its similarity to the appearance model and its auto-similarity to its last observation, a temporary model that evolves independently for each particle. The weighting of the mixture is proportional to the probability density scattering towards the appearance model similarity.

B. An accurate 3D model

Our face model is extracted from the Basel Face Model (BFM) [12], which is a 3D Morphable Model (3DMM) [9]. It was constructed using 200 face scans including 100 male and 100 female persons with a neutral expression. Their average age is 25 while the average weight is 67 kilograms, ages going from 8 to 62 years. Each face scan is formed with m = 53490 vertices sharing topology. Each vertex is thus described by its 3D position $(x_j, y_j, z_j)^T \in \mathbb{R}^3$, and its color $(r_j, g_j, b_j)^T \in [0,1]^3, j \in \{1, ..., m\}$. Shape and texture distributions were independently modeled using Principle Component Analysis (PCA). The resulting parametric face model is thus described as

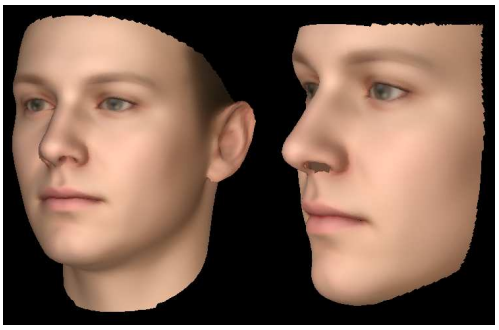

Fig. 1. The left face is the full Basel Face Model. On the right, the same after the first selection.

$$M_s = (\nu_s, \sigma_s, U_s) \tag{1}$$

and

$$M_t = (\nu_t, \sigma_t, U_t) \tag{2}$$

where $\nu_{s,t}$ are the mean, $\sigma_{s,t}$ are the standard deviations and $U_{s,t}$ are orthonormal bases of principal components of shape and texture. This mainly describes the BFM, which was developed for face recognition purposes. In our case, we use only parts of the model. The shared topology implies that the vertices corresponding to the MPEG4 landmarks [18] are available in a list of indices. These specific points to which we added manually picked vertices were explicitly not decimable in the following selection algorithm. These 108 points are used to have a common candide-like representation of the model [19].

As computing over 50,000 vertices for the head pose tracking is not necessary for the resolutions we work at and implies tremendous calculation costs anyways, we have to select a subset of vertices within the original m. The m vertices cover the ears, the neck and a large surface of the front. Because these parts will be often occluded by hair or clothes in real video sequences, we decided to exclude them from our face model. After this first selection, about 26,000 vertices remain (Fig. [1]). We can then select subsets of vertices using their local density.

The needed number of vertices depends on the face resolution in the application images. For instance, if a tracked face has an actual resolution of 30×30, having it tracked by a $N = 2,000$ would be a computational waste. Our final implementation uses a multi-scale approach to optimize calculations costs. For these vertices selections we use an iterative random decimation algorithm. At each iteration, each vertex has a low probability to be removed from the final selection. This decimation probability depends on the local vertices density, high density meaning high decimation probability, we have in the end a semi-uniform vertices selection. There are no remaining patterns and the symmetry is destroyed (see Fig. [2]). Using a lower resolution

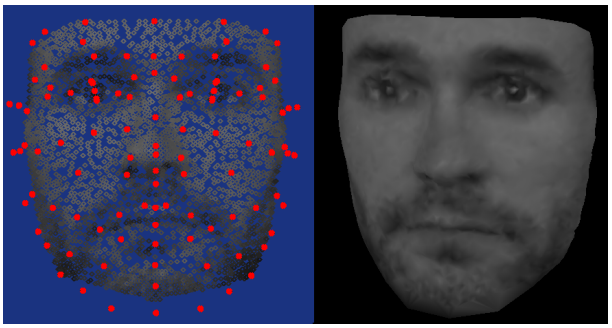

Fig. 2. An example of texture projection on a 3,000 vertices model. The red vertices are the MPEG4 and hand-picked vertices used for display purposes. On the right, the 3d rendering.

implies the original shape deformation vectors not to be valid anymore. Simply taking the matching components to build a new set of vectors destroys the basis' orthogonality. In order to recreate an orthonormal basis for the selected vertices, we generated random full resolution faces using the original distribution. For each intended sub-resolution, we selected the right vertices in each newly generated face example and computed another model using PCA, resulting in a new orthonormal basis. We do not use the texture distribution as the BFM albedo measurements would not match classical camera sensors. To construct our texture model, we manually match our lower resolution models on frontal face pictures from many face databases (FacePix [20], CMU PIE [21], Yale face database [22]). Our texture PCA is extracted from 474 different textures captured in classical lighting conditions. It is used to initialize the 3D head pose after the face detection (Viola-Jones face detector) [17].

C. 3D tracking method

1) Least squares optimization: Our tracking method is based on Xiao et al. dynamics templates and re-registration techniques [5]. They applied a form of the iteratively re-weighted least squares (IRLS) [11] while we choose another metric using distance in feature space and distance from feature space [13].

With the current model having N vertices, we measure each vertex' gray level by projecting its 3D coordinates $\hat{u}_n = (x_n, y_n, z_n), n \in \{1, ..., N\}$ onto the image plane *I*. We suppose this projection depends only on the focal length. For a rigid motion of any considered vertex $\hat{u}_n$ we update it with the matrix Q to get updated coordinates $\hat{u}'_n$:

$$(\hat{u}'_n, 1) = Q.(\hat{u}_n, 1) \tag{3}$$

$$Q = \begin{pmatrix} 1 & -r_z & r_y & t_x \\ r_z & 1 & -r_x & t_y \\ -r_y & r_x & 1 & t_z \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{4}$$

We can combine this transformation function with the perspective projection to obtain the function *F*:

$$F(\hat{u}_n, \mu) = \begin{pmatrix} x_n - y_n r_z + z_n r_y + t_x \\ x_n r_z + y_n - z_n r_x + t_y \end{pmatrix} \cdot \frac{f_L}{-x r_y + y r_x + z + t_z} \tag{5}$$

with $\mu = (r_x, r_y, r_z, t_x, t_y, t_z)$, the rigid motion parameters vector and f_L being the video camera's focal length. $F(\hat{u}_n, \mu)$ is the projection of the current 3D vertex $\hat{u}_n$ on the image plane after the rigid motion $Q(\mu)$. Given the current observation, the goal of the algorithm is to find the vector μ which will iteratively move the current head model towards the solution, the actual head pose. To achieve this difficult task, Xiao et al. proposed to minimize the sum of squared differences (SSD) $E(\mu)$ between the template T and the current observation $I(F(\hat{u}, \mu))$ with the Lucas-Kanade algorithm [14].

$$E(\mu) = \sum_{n=1}^{N} (I(F(\hat{u}_n, \mu)) - T_n)^2 \tag{6}$$

The template T is an average face texture for the initialization frame, and an updated texture for the following frames. Depending on the pose and occlusions, some vertices may not be visible. Moreover, noise or lighting changes can perturbate the 3D tracking. To take these issues into account, Xiao et al. used a form of the iteratively re-weighted least squares (IRLS) robust technique [11]. It affects to each vertex several weights. The w_I value depends on each vertex' current gray level observation compared to the matching template gray level. The weight is maximized when the current gray level is strictly equal to the template's, and decreases following a normal law with a standard deviation depending on the whole template's median value. This weight allows us to disqualify outliers. The w_G value depends on the vertex' gradient absolute value, giving importance to pixel on edges. The last one, w_D depends on the 3D vertex coordinates and its normal orientation. This last weight regularizes the vertices density on the image plan and permits to disqualify occluded vertices at the current pose and gives importance to the facing vertices. In the end, the total weight w is affected to the vertex. See [5] for more details.

$$w = w_D(w_I + w_G) \tag{7}$$

The L-K algorithm finds a solution in the least squares sense. We propose an alternative optimization method by projecting the texture error $I(F(\hat{u}, \mu)) - T$ and in a composite metric using Distance In Feature Space (DIFS) and Distance From Feature Space (DFFS)[13].

2) Composite metric: To easily understand both DIFS and DFFS, please imagine a 2D gaussian distribution. By using PCA, you find the two principal components vectors. The feature space is always defined by the most important vectors, whereas its orthogonal space is defined by the complement vectors. With only two available vectors, the first component vector describes our feature space, while the second one describes the non feature space. For a new observation, the feature space is the axis described by the reference observation (used to compute this PCA) and the first vector. The distance in feature space is the distance between the observation projection on this axis and the reference data. The distance from feature space is the distance between an observation and its projection onto the feature space.

In our case of face textures, for each resolution, we computed a texture PCA with samples from face databases [20], [21], [22]. Having a limited number of texture samples compared to the dimensionality of the full eigenspace, only the first eigenvectors contain useful information. The following vectors are basically noise. We thus select the first eigenvectors to build our DIFS and DFFS.

After ranking the texture PCA's eigenvectors $\{\Phi_i\}_{i=1}^{L}$ and the diagonal matrix Λ containing the matching eigenvalues λ_i, we select the *K* first principal components of the vector space $\mathbb{R}^L$. We form the feature space described by the orthogonal basis $\Phi_K = \{\Phi_i\}_{i=1}^{K}$. With its orthogonal complement described by $\bar{\Phi}_K = \{\Phi_i\}_{i=K+1}^{L}$, they constitute two mutually exclusive and complementary subspaces. For any texture vector $\mathbf{x}$, we can obtain its component vector $\mathbf{y} = \Phi^T(\mathbf{x} - \bar{\mathbf{x}})$. We can express the Mahalanobis distance in terms of the sum:

$$d(\mathbf{x}) = \sum_{i=1}^{K} \frac{\mathbf{y}_i^2}{\lambda_i} + \sum_{i=K+1}^{L} \frac{\mathbf{y}_i^2}{\lambda_i} \tag{8}$$

The first summation is computed by projecting $\mathbf{x}$ onto the K-dimensional principal subspace *D*. The second one is not computed explicitly because of high-dimensionality and because the implied eigenvectors are mostly noise. The sum of this term is however available as

$$d(\mathbf{x}) = \sum_{i=1}^{L} (\mathbf{x}_i - \bar{\mathbf{x}}_i)^2 = \sum_{i=1}^{L} \mathbf{y}_i^2 \tag{9}$$

$$\epsilon^2(\mathbf{x}) = \sum_{i=K+1}^{L} \mathbf{y}_i^2 = ||(\mathbf{x} - \bar{\mathbf{x}})||^2 - \sum_{i=1}^{K} \mathbf{y}_i^2 \tag{10}$$

we formulate an estimator $\hat{d}(\mathbf{x})$:

$$\hat{d}(\mathbf{x}) = \sum_{i=1}^{K} \frac{\mathbf{y}_i^2}{\lambda_i} + \frac{\epsilon^2(\mathbf{x})}{\rho} \quad \text{with} \quad \rho = \frac{1}{L-K} \sum_{K+1}^{L} \lambda_i \tag{11}$$

ρ being the arithmetic average of the eigenvalues in the orthogonal space, so that the expected value of the estimator is equal to the expected value of the Mahalanobis distance. So instead of optimizing an euclidian distance between our template *T* and our observed texture vector $I(\hat{u})$, we intend to minimize the estimator $\hat{d}(\mathbf{x})$:

$$\begin{aligned} \hat{d}(\mathbf{x}) = (\mathbf{x} - \bar{\mathbf{x}})^T \Phi_K \Lambda^{-1} \Phi_K^T (\mathbf{x} - \bar{\mathbf{x}}) + \\ (\mathbf{x} - \bar{\mathbf{x}})^T \frac{I_d - \Phi_K \Phi_K^T}{\rho} (\mathbf{x} - \bar{\mathbf{x}}) \end{aligned} \tag{12}$$

The first term of (12) is the DIFS (equivalent to the Mahalanobis distance on the first K components), while the second term is the DFFS. In the end, minimizing $\hat{d}(\mathbf{x})$, is equivalent to minimizing the new error $E(\mu)$, which we will use as our trust metric during the head tracking.

$$E(\mu) = (I(F(\hat{u}, \mu)) - T)^T \mathbf{A} (I(F(\hat{u}, \mu)) - T) \tag{13}$$

with

$$\mathbf{A} = \mathbf{I_d} - \Phi_K \Phi_K^T + \rho \Phi_K \Lambda^{-1} \Phi_K^T \tag{14}$$

The L-K algorithm needs to compute the Gauss-Newton approximation to the Hessian matrix :

$$H = \sum_{n}^{N} \left[I_n \frac{\partial F(\hat{u}_n, 0)}{\partial \mu} \right]^T \mathbf{A} \left[I_n \frac{\partial F(\hat{u}_n, 0)}{\partial \mu} \right] \tag{15}$$

with

$$\frac{\partial F}{\partial \mu}|_{\mu=0} = \begin{pmatrix} -xy & x^2+z^2 & -yz & z & 0 & -x \\ -(y^2+z^2) & xy & xz & 0 & z & y \end{pmatrix} \cdot \frac{f_L}{z^2} \tag{16}$$

The Lucas-Kanade solution we obtain is:

$$\mu^* = -H^{-1} \sum_{n}^{N} w_n \left[\hat{I}_n \frac{\partial F(\hat{u}_n, 0)}{\partial \mu} \right]^T \mathbf{A} (I(F(\hat{u}_n, 0)) - T_n) \tag{17}$$

With $\hat{I}_n = \nabla I(\hat{u}_n)$, the value of the gradient image for the vertex n.

3) A Morphable model: For now, the model is an average rigid 3d face we extracted from the mean shape of the BFM. By implementing the shape PCA we calculated beforehand, we aim to adapt the face model to the tracked face. Obviously, this system will only be able to adapt to neutral faces.

The shape eigenvectors are computed in the original face referential. For any given pose, rotation matrix *R* which rotates our head model in its referential to the actual pose in the camera referential will define the calculation needed for the morphological adaptation. We do not perform simultaneous pose and shape optimization as it implies converging in a much higher dimension space. Also, the shape optimization is triggered only with frontal view poses. Let us reshape the $3N$ components shape vectors $\mathbf{x}_s$ and the first *K'* shape eigenvectors $\{\Psi_i\}, i \in \{1, ..., K'\}$ into $\tilde{\mathbf{x}}_s$ and $\left\{\tilde{\Psi}_i\right\}, i \in \{1, ..., K'\}$, $3 \times N$ matrices. We can now formulate an elementary shape deformation with the parameters vector $\alpha = (\alpha_1, ..., \alpha_{K'})$:

$$\tilde{\mathbf{x}}_s' = \tilde{\mathbf{x}}_s + R \sum_{i=1}^{K'} \tilde{\Psi}_i \alpha_i \tag{18}$$

The 3d coordinates and rotation matrices can be defined in separate lines : $\tilde{\mathbf{x}}_s = (\tilde{x}_s, \tilde{y}_s, \tilde{z}_s)$ and $R = (R_1, R_2, R_3)$. As previously, we combine the shape deformation with the image projection function :

$$G(\tilde{\mathbf{x}}_s, \alpha) = \begin{pmatrix} \tilde{x}_s + R_1 \sum_{i=1}^{K'} \tilde{\Psi} \alpha i \\ \tilde{y}_s + R_2 \sum_{i=1}^{K'} \tilde{\Psi} \alpha i \end{pmatrix} \cdot \frac{f_L}{\tilde{z}_s + R_3 \sum_{i=1}^{K'} \tilde{\Psi} \alpha i} \tag{19}$$

For the Lucas-Kanade algorithm, we have to compute $\frac{\partial G(\tilde{\mathbf{x}}, \alpha)}{\partial \alpha}$ at $\alpha = 0$:

$$\frac{\partial G}{\partial \alpha_i}|_{\alpha_i=0} = \begin{pmatrix} \tilde{z}_s R_1 \tilde{\Psi}_i - \tilde{x}_s R_3 \tilde{\Psi}_i \\ \tilde{z}_s R_2 \tilde{\Psi}_i - \tilde{y}_s R_3 \tilde{\Psi}_i \end{pmatrix} \cdot \frac{f_L}{\tilde{z}_s^2} \tag{20}$$

For the morphological adaptation, the Hessian matrix is :

$$H_m = \sum_{n}^{N} w_n \left[\hat{I}_n \frac{\partial G(\tilde{\mathbf{x}}_s, 0)}{\partial \alpha} \right]^T \mathbf{A} \left[\hat{I}_n \frac{\partial G(\tilde{\mathbf{x}}_s, 0)}{\partial \alpha} \right] \tag{21}$$

TABLE I

AVERAGE ERRORS ON THE BOSTON DATA SET.

tracker	pitch (°)	yaw (°)	roll (°)
cylindrical head model	4.4	5.2	2.5
ellipsoid head model	3.9	4.0	2.8
c-metric AHM	**3.7**	3.8	2.4
c-metric AHM + Morph. Adapt.	4.4	**3.5**	**2.1**

The Lucas-Kanade solution for the morphologic adaptation is :

$$\alpha^* = -H_m^{-1} \sum_n^N w_n \left[\hat{I}_n \frac{\partial G(\tilde{\mathbf{x}}_s, 0)}{\partial \alpha} \right]^T \mathbf{A}(I(F(\hat{u}_n, 0)) - T_n) \quad (22)$$

4) Inverse Compositional Algorithm (ICA): In our case, diminishing the computation costs by implementing the ICA [14] consisted first in pre-computing the gradient ∇T of the template T to inverse the template's and the current observation's role. Then for each new frame, pre-compute the Jacobian $\frac{\partial F(\hat{u},0)}{\partial \mu}$, but also the regularization weight w to finally have a constant hessian matrix for the next iterations of the L-K algorithm. These last two approximations imply the fact using the ICA is only valid in a very local neighborhood around the initial position. While using the ICA with our composite metric provided acceptable results, the SSD version was destabilized. The computational gain is not worth the lessened results quality. In the case where using a significant computation acceleration (with a graphics processing unit) is out of the question, one may want to update the Jacobian and weights only every few iterations.

III. EXPERIMENTATION

A. Boston Data set

We tested our system on the Boston video data set described in [4]. The head tracker resolution ranged from $N = 400$ to $3,000$ in the test implementations. It is worth noting that our reported errors are not computed w. r. t. a manually referenced pose (first frames of each videos), as we keep our original and absolute referential. These measures thus contain an offset error which inevitably occurs during the tracking initialization. As a corollary, the absolute initial fitting error computed with the average face texture as template can be limited to our results in Table I. However, we have better results than the cylindrical and ellipsoid head model tests provided by Choi and Kim [6] (see Table I). Our pitch error with the morphological adaptation can be explained. The pitch rotation changes the 2D appearance in a similar way the shape deformations do and they are thus linked. We have to take the relatively low angular accuracy of the ground truth into account. There is at least a 0.5 degrees admitted indetermination but given the data noise, we estimate it at around 3 degrees. This is most certainly due to interferences with external elements such as metal furniture and computers [4]. For further experimentation, we decided to make our own test videos with a more accurate ground truth.

B. Motion capture data set

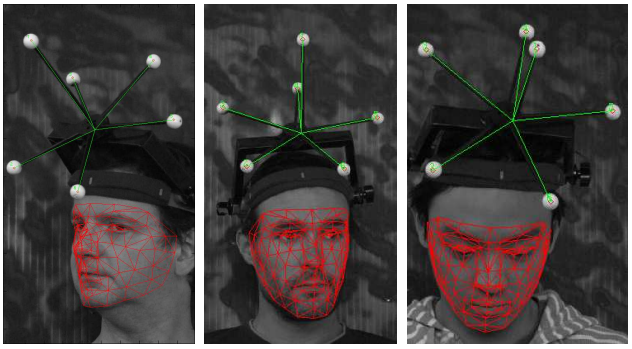

Fig. 3. Example of our hat's motion capture (green) and our pose tracker (red).

We designed a motion capture "hat" worn by every test subjects. The hat was crafted to be easily trackable in the videos. We precisely measured the white spherical markers' 3D coordinates in the hat's referential. By detecting the visible markers in each frame with a circular Hough transform and matching the results with the real coordinates via the Posit algorithm [23] (see Fig. [3]), we are able to track the hat's pose with a high accuracy. The markers detection error is estimated in the 1-3 pixels range, corresponding to a 0.6-1.8 mm error. Given the hat's width (around 20cm), it implies a 0.2-0.6 degrees error in the worst case scenario, depending on the subject's attitude. We estimate the actual indetermination to be about 0.1 degrees. This method avoids many ground truth issues like measure and video synchronizing, or possible physical interferences diminishing the measurement quality. For each sequence, we manually selected the reference frame which is not necessarily the first one. All the hat poses are computed w. r. t. the reference image hat pose. The 8 video sequences count from 400 to over 800 vertical HD frames shot at 25 frames per second. The subjects were first asked to watch the camera then look around. The sequences contain wide pose range with yaw and pitch angles above 45 and 20 degrees, respectively (see Fig. [4], [5]). The total number of test frames is over 4,300.

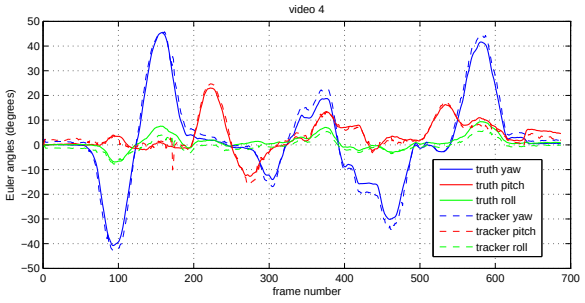

Fig. 4. Ground truth and our tracker's results on a test video.

For the sequence where the SSD implementation delivers its best results (see Table II), the composite metric is still

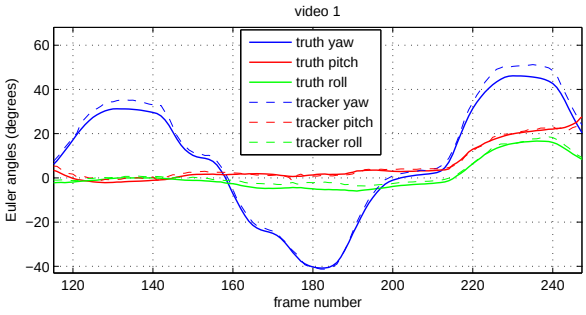

Fig. 5. Zoom on a ground truth and our tracker's results on a test video.

more accurate. As you can see in Table III, the tracker achieves great accuracy in a wide pose range.

TABLE II

AVERAGE ERRORS ON VIDEO 1 (BEST SSD RESULTS).

tracker / mean	pitch (°)	yaw (°)	roll (°)
SSD + MA	4.6	4.8	4.0
c-metric AHM	3.1	3.4	3.2
c-metric AHM + MA	**2.5**	**2.6**	**2.2**

TABLE III

AVERAGE ERRORS ON THE MOTION CAPTURE HAT SET.

tracker	pitch (°)	yaw (°)	roll (°)
c-metric AHM	4.6	3.8	2.4
c-metric AHM + MA	**3.6**	**2.2**	**1.9**

IV. CONCLUSIONS

We propose an effective 3-D head tracker using a head model extracted from real face scans while most methods use simpler approximations using fiducial points or geometrical shapes. We created our head model manifolds from the BFM. By decimating the original set of vertices, we managed to create a lighter head model with representative shape deformations. The regression process is triggered after a face detection and it tracks continuously the subject's rigid motion and its morphology across whole video sequences, over a wide pose range. It is also quite insensitive to perturbations. We also built a composite metric with face textures from many databases and showed that its implementation improved the L-K algorithm. The good results on the Boston data set and our owns show that our method is more accurate than the cylindrical or the ellipsoid model-based trackers.

We also propose a simple yet accurate method to obtain ground truth data, using an optical motion capture system based on "hat" mounted white spherical markers, easy to track with a circular shape detector. While using trivial principles, it shows great accuracy and is easy to set up.

V. ACKNOWLEDGMENTS

We provide our test videos including the ground truth measures on request. Please contact us by mail to receive our data. The authors gratefully acknowledge the contribution of the Agence National de la Recherche and reviewers' comments.

REFERENCES

[1] Y. Tong, Q. Ji, Y. Wang, and Z. Zhu. Robust facial feature tracking under varying face pose and facial expression. *Pattern Recognition*, 2007.

[2] T.F. Cootes, G.J. Edwards, and C.J. Taylor. Active appearance models. *Computer Vision-ECCV*, 1998.

[3] M. J. Black and Y. Yacoob. Tracking and recognizing rigid facial motions using local parametric models of image motion. *ICCV*, pages 374–381, 1995.

[4] M. La Cascia, S. Sclaroff, and V. Athitsos. Fast, reliable head tracking under varying illumination: an approach based on registration of texture-mapped 3d models. *PAMI*, 2000.

[5] J. Xiao, T. Kanade, and J. Cohn. Robust full-motion recovery of head by dynamic templates and re-registeration techniques. *FG*, pages 156–162, 2002.

[6] S. Choi and D. Kim. Robust head tracking using 3d ellipsoidal head model in particle filter. *Pattern Recognition*, pages 241(9):2901–2915, 2008.

[7] S. Basu, I. Essa, and A. Pentland. Motion regularization for model-based head tracking. *ICPR*, 3:611–616, 1996.

[8] S. Baker, I. Mathews, J. Xiao, R. Gross, T. Ishikawa, and T. Kanade. Real-time non-rigid driver head tracking for driver mental state estimation. *11th World Congress Intelligent Transportation Systems*, 2004.

[9] V. Blanz and T. Vetter. A morphable model for the synthesis of 3d faces. *PAMI*, 2003.

[10] F. Dornaika and J. Ahlberg. Fitting 3d face models for tracking and active appearance model training. *Image and Vision Computing 24*, pages 1010–1024, 2006.

[11] M. Black. *Robust incremental optical flow*. PhD thesis, 1992.

[12] P. Paysan, R. Knothe, B. Amberg, S. Romdhani, and T. Vetter. A 3d face model for pose and illumination invariant face recognition. *6th IEEE International Conference on Advanced Video and Signal based Surveillance (AVSS) for Security, Safety and Monitoring in Smart Environments*, 2009.

[13] B. Moghaddam and A. Pentland. Probabilistic visual learning for object representation. *Early Visual Learnin*, pages 99–130, 1996.

[14] S. Baker and I. Matthews. Lucas-kanade 20 years on: A unifying framework. *IJCV*, pages 221–255, 2004.

[15] D.A. Ross, J. Lim, R.S. Lin, and M.H. Yang. Incremental learning for robust visual tracking. *IJCV*, 77:125–141, 2008.

[16] A. Levy and M. Lindenbaum. Sequential karhunen-loeve basis extraction and its application to images. *IEEE Trans. on Image Processing*, 9:1371–1374, 2000.

[17] P. Viola and M. Jones. Robust real-time object detector. *International Journal on Computer Vision*, pages 57:137–154, 2004.

[18] J. Ostermann. Animation of synthetic faces in mpeg-4. *Computer Animation*, pages 49–51, 1998.

[19] J. Ahlberg. Candide-3 - an updated parameterized face, 2001.

[20] G. Little, S. Krisgna, J. Black, and S. Panchanatha. A methodology for evaluating robustness of face recognition algorithms with respect to variations in pose angle and illumination angle. *Int. Conf. Acoustics, Speech and Signal Processing*, 2005.

[21] T. Sim, S. Baker, and M. Bsat. The cmu pose, illumination and expression (pie) database. *FG*, 2002.

[22] A.S. Georghiades, P.N. Belhumeur, and D.J. Kriegman. Illumination cone models for face recognition under variable lighting and pose. *PAMI*, 23:643–660, 2001.

[23] R. Duraiswami P. David, D. DeMenthon and H. Sametl. Softposit: Simultaneous pose and correspondance determination. *IJCV*, pages 259–284, 2004.