Automatic modelling and efficient tracking of deformable objects

E. Muñoz[1] A. Del Bue[2] J.M. Buenaposada[3] L. Baumela[1] L. Agapito[2]

[1] Facultad de Informática. Universidad Politécnica. 28660 Boadilla del Monte, Madrid, Spain
[2] Dept. Computer Science. Queen Mary, University of London. E1 4NS, London. UK
[3] Dpto. de Informática, Estadística y Telemática. Univ. Rey Juan Carlos. 28933 Móstoles, Madrid.

Email: {kike,lbaumela}@dia.fi.upm.es {alessio,lourdes}@dcs.qmul.ac.uk jmbuenaposada@escet.urjc.es

Keywords: non-rigid tracking, non-rigid factorization, direct methods, morphable models

Abstract

This paper presents a system for efficiently tracking a deformable object in 3D. It is based on a model of the target represented as a set of textured features in 3D space and a set of shape bases, which encode the non-rigid modes of deformation. The model is constructed from a monocular sequence of 2D feature tracks using a non-rigid factorization algorithm followed by a non-linear optimization of the model parameters. Once the model is available, it is used for efficiently tracking the target using a 3D extension of the Inverse Compositional Algorithm which uses a projective camera model. In the experiments we show the performance of the system on synthetic and real sequences of a human face undergoing different facial expressions.

1 Introduction

Non-rigid face modelling and tracking are currently research topics of great interest to the computer vision and graphics communities for their application to the construction of advanced computer interfaces and to achieving realistic human models for animation. In this paper we present an efficient model-based tracking system which tracks the rigid and non-rigid motion of a human face. The model is generated automatically from a sequence of images and it consists of a set of shape bases, which encode the principal modes of deformation of the face, and a set of small textured patches centred around some feature points on the 3D model. Each patch is tangent to the 3D volume of the face at a different point. The texture of the patch is the result of projecting the underlying grey levels of the face orthogonally onto a small plane. This set of patches effectively acts as a sparse model of face appearance. Once the model has been built a new version of the Inverse Compositional Alignment (ICA) algorithm – modified to improve its efficiency – is used to track the face by relating changes in appearance with face motion.

Recent work in non-rigid factorization [5, 3, 17] has proved that under weak perspective viewing conditions it is possible to infer the principal modes of deformation of an object alongside its 3D shape, within a structure from motion estimation framework. Crucially, these new factorization methods work purely from video in an unconstrained case: a single uncalibrated camera viewing an arbitrary 3D surface which is moving and articulating. In this paper we have used an extension of these algorithms which includes a non-linear minimization step to optimize the deformable 3D shape and motion [7].

Our model-based tracking procedure is based on the ICA algorithm, which was devised for minimising image-based (2D) cost functions that were invertible and closed under composition (e.g. 2D affine or projective warps) [2]. In its original formulation it was used for fitting a Flexible Appearance Model. Recently, a 3D extension has been proposed for fitting a 3D Morphable Model [14]. One of the limitations of this approach is the assumption of affine camera projection, in order to simplify the estimation of the rigid component of motion. In this paper we present a 3D extension which computes the rigid and non-rigid motion components for a projective camera.

The problem of non-rigid face modelling and tracking has been previously addressed by different authors. Most approaches to face tracking are based on very precise models. Blanz and Vetter [19] use hundreds of scanned faces in order to model the face and two kinds of deformations: those caused by facial expressions and those due to morphological differences among humans. Decarlo and Metaxas [8] use a hand-crafted model and optical flow data for tracking. Eisert and Girod [9] use a similar model-based approach. Our approach is most closely related to the work by Gokturk et. al. [10] however, it differs from it both in the way the model is built and in the tracking process. In this paper, we use a non-rigid factorization approach to generate the 3D deformable model using a set of 2D feature tracks from an uncalibrated monocular sequence as input [7]. Gokturk et. al. on the other hand, used a stereo camera setup to obtain a set of 3D tracks on which PCA is applied to obtain the shape bases. Finally, their tracking procedure is based on the Lucas and Kanade alignment algorithm. Instead, we used a new version of ICA, which has been modified to improve its efficiency [1]. Brand and Bhotika also used a 3D morphable model generated using non-rigid factorization to perform model-based tracking [4]. However their tracking algorithm is not incremental like the one presented here but instead estimates the new set of parameters at each frame.

The paper is organized as follows: in section 2 we describe the non-rigid factorization algorithm used to generate the 3D deformable model used in the tracking. Section 3 describes the inverse compositional algorithm (ICA) while in section 4 we introduce the efficient version of this algorithm, extended to deal with 3D data and with a projective camera model. In section 5 we present some results with synthetic and real sequences and finally we present some final conclusions and future work.

2 Automatic 3D model building

Tomasi and Kanade's factorization algorithm for rigid structure [16] has recently been extended to the case of non-rigid deformable 3D structure [5, 3, 17]. Here, the 3D shape of any configuration of the non-rigid object is expressed as a linear combination of a set of K basis-shapes $\mathtt{B}_i$ plus a mean component $\mathtt{X}$ in the following way:

$$\mathtt{S} = \mathtt{X} + \sum_{i=1}^{K} l_i \mathtt{B}_i \quad \mathtt{S}, \mathtt{X}, \mathtt{B}_i \in \Re^{3\times N} \quad l_i \in \Re,$$

where N are the number of points describing the object and l_i are the configuration weights. If we assume a scaled orthographic projection model for the camera, the coordinates of the 2D image points observed at each frame f are related to the coordinates of the 3D points according to the following equation:

$$\mathtt{W}_f = \begin{bmatrix} u_{f,1} & \dots & u_{f,N} \\ v_{f,1} & \dots & v_{f,N} \end{bmatrix} = c_f \mathtt{R}_f \left(\mathtt{X} + \sum_{i=1}^{K} l_i \mathtt{B}_i \right) + \mathbf{T}_f \tag{1}$$

where c_f is the scale parameter, $\mathtt{R}_f$ is a 2×3 orthonormal matrix which contains the first and second rows of the camera rotation matrix and $\mathbf{T}_f$ contains the first two components of the camera translation vector, which may be eliminated by registering image points to the centroid in each frame. If all N points can be tracked throughout an image sequence we may stack all the point tracks from frame 1 to F into a $2F \times N$ measurement matrix W and we may write:

$$\mathtt{W} = \begin{bmatrix} \mathtt{W}_1 \\ \mathtt{W}_2 \\ \vdots \\ \mathtt{W}_F \end{bmatrix} = \begin{bmatrix} c_1\mathtt{R}_1 & c_1 l_{11}\mathtt{R}_1 & \dots & c_1 l_{1K}\mathtt{R}_1 \\ \vdots & & \vdots & \\ c_f\mathtt{R}_F & c_f l_{F1} R_F & \dots & c_f l_{FK}\mathtt{R}_F \end{bmatrix} \begin{bmatrix} \mathtt{X} \\ \mathtt{B}_1 \\ \vdots \\ \mathtt{B}_K \end{bmatrix} = \mathtt{MS} \tag{2}$$

Since $\mathtt{M}$ is a $2F \times 3(K+1)$ matrix and $\mathtt{S}$ is a $3(K+1) \times N$ matrix, the rank of $\mathtt{W}$ must be $r \leq 3(K+1)$.

The rank constraint on the measurement matrix $\mathtt{W}$ can be easily imposed by truncating the SVD of $\mathtt{W}$ to rank 3(K+1). This will factor $\mathtt{W}$ into a motion matrix $\tilde{\mathtt{M}}$ and a shape matrix $\tilde{\mathtt{S}}$. Note that in the non-rigid case the matrix $\tilde{\mathtt{M}}$ needs to be further decomposed into the 3D pose matrices $\mathtt{R}_f$ and the deformation weights l_{fk} since their values are mixed inside the motion matrix $\tilde{\mathtt{M}}$.

A further issue is that the result of the factorization of $\mathtt{W}$ into $\tilde{\mathtt{M}}$ and $\tilde{\mathtt{S}}$ is not unique since any invertible $3(K+1) \times 3(K+1)$ matrix $\mathtt{Q}$ can be inserted in the decomposition leading to the alternative factorization: $\mathtt{W} = (\tilde{\mathtt{M}}\mathtt{Q})(\mathtt{Q}^{-1}\tilde{\mathtt{S}})$. The problem is to find a transformation matrix $\mathtt{Q}$ that renders the appropriate replicated block structure of the motion matrix $\tilde{\mathtt{M}}$ shown in (2) and that removes the affine ambiguity upgrading the reconstruction to a metric one. Whereas in the rigid case the problem of computing the transformation matrix $\mathtt{Q}$ to upgrade the reconstruction to a metric one can be solved linearly [16], in the non-rigid case imposing the appropriate repetitive structure to the motion matrix $\tilde{\mathtt{M}}$ results in a non-linear problem. Various methods to recover the transformation matrix $\mathtt{Q}$ have been proposed so far in the literature [3, 5, 17] but they fail to provide a completely satisfactory solution.

2.1 Bundle adjustment

Our approach is to obtain an initial solution for the non-rigid shape and 3D pose and then to perform a non-linear optimization step by minimizing image reprojection error.

The goal is to estimate the camera matrices $\mathtt{R}_i$ and the 3D structure parameters $\mathtt{X}, \mathbf{B}_k, c_i, l_{ik}$ such that the distance between the measured image points $\mathbf{p}_{ij}$ and the estimated image points $\hat{\mathbf{p}}_{ij}$ is minimized:

$$\min_{\mathtt{R}_i \mathtt{X} \mathbf{B}_k c_i l_{i,k}} \sum_{i,j} \parallel \mathbf{p}_{ij} - \hat{\mathbf{p}}_{ij} \parallel^2 = \tag{3}$$

$$\min_{\mathtt{R}_i \mathtt{X} \mathbf{B}_k c_i l_{i,k}} \sum_{i,j} \parallel \mathbf{p}_{ij} - (c_i \mathtt{R}_i (\mathtt{X} + \sum_k l_{i,k} \mathbf{B}_k)) \parallel^2 \tag{4}$$

This method, generically termed bundle-adjustment, provides a Maximum Likelihood estimate provided that the noise can be modelled with a Gaussian distribution. The non-linear optimization of the cost function was achieved using a Levenberg-Marquadt minimization scheme modified to take advantage of the sparse block structure of the matrices involved [18].

The initial estimate for the bundle adjustment minimization could be provided for example by Brand's non-rigid factorization algorithm [3]. However, we have found that an alternative procedure that provides a satisfactory initial estimate is to compute the motion associated to the rigid component and to initialize the configuration weights to small values close to zero. A prior on the 3D shape has been added to the cost function to avoid the non-linear optimization leading to a solution corresponding to a local minimum. Our prior states that the depth of the points on the object surface will not change significantly from one frame to the next, adding the term $\sum_{i=2,j=1}^{i=F,j=N} \parallel \mathtt{S}_z^{i-1,j} - \mathtt{S}_z^{i,j} \parallel^2$ to the cost function. Similar regularization terms have also been reported in [17, 3].

In the case where the subject is only performing non-rigid deformations we suggest the use of a stereo factorization approach to obtain the initial estimate [6].

In the following sections we present our novel model-based efficient 3D tracker which uses the 3D model described in this section and is an extension of the Incremental Compositional Alignment algorithm of Baker and Matthews which we explain in the next section.

3 Inverse compositional algorithm

In this section we describe the Inverse compositional tracking algorithm proposed by Baker and Matthews [2].

Let $\mathbf{x}$ represent the location of a point in an image and $\mathtt{I}[\mathbf{x}, t]$ represent the brightness value of that location in the image acquired at time t. Let $\mathcal{R} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N\}$ be a set of N image points of the object to be tracked (*target region*), whose brightness values are known in a reference image $\mathtt{T}[\mathbf{x}]$. These image points together with their brightness values at the reference image represent the *reference template* to be tracked.

Assuming that the *brightness constancy* assumption holds, then

$$\mathtt{T}[\mathbf{x}] = \mathtt{I}[f(\mathbf{x}, \boldsymbol{\mu}_t), t] \forall \mathbf{x} \in \mathcal{R}, \tag{5}$$

where $\mathtt{I}[f(\mathbf{x}, \boldsymbol{\mu}_t), t]$ is the image acquired at time t rectified with motion model $f(\mathbf{x}, \boldsymbol{\mu})$ and motion parameters $\boldsymbol{\mu} = \boldsymbol{\mu}_t$.

Tracking the object means recovering the motion parameter vector of the target region for each image in the sequence. This can be achieved by minimising the difference between the template and the rectified pixels of the target region for every image in the sequence

$$\min_{\boldsymbol{\mu}} \sum_{\forall \mathbf{x} \in \mathcal{R}} [\mathtt{I}(f(\mathbf{x}, \boldsymbol{\mu}), t] - \mathtt{T}[\mathbf{x}]]^2 \tag{6}$$

This minimisation problem has been traditionally solved linearly by computing $\boldsymbol{\mu}$ incrementally while tracking. We can achieve this by making a Taylor series expansion of (6) and computing the increment in the motion parameters between two time instants by Gauss-Newton iterations. Different solutions to this problem have been proposed in the literature, depending on which term of (6) the Taylor expansion is made on and how the motion parameters are updated [12, 11, 15, 2].

This problem was first solved in the seminal work of Lucas and Kanade [12]. The computational cost of tracking with this approach is due mainly to the cost of estimating the Jacobian of the image grey values w.r.t. the motion model's parameters and its pseudoinverse, which are needed to make the Gauss-Newton iterations. Two efficient tracking extensions to the Lucas and Kanade algorithm have been proposed, which overcome this problem, the *Factorisation* approach of Hager and Belhumeur [11] and the *Inverse Compositional Algorithm* (ICA) of Baker and Matthews [2]. Here we will present the second approach, which is the one used in our tracker.

The minimisation solved for tracking with ICA is the following

$$\min_{\delta\boldsymbol{\mu}} \parallel \mathbf{I}[f(\mathbf{x}, \boldsymbol{\mu}_t), t + \delta t] - \mathbf{T}[f(\mathbf{x}, \delta\boldsymbol{\mu})] \parallel^2, \tag{7}$$

where $\mathbf{T}[\mathbf{x}]$ and $\mathbf{I}[\mathbf{x}, t]$ are vectors formed by scanning the grey levels in $\mathtt{T}[\mathbf{x}]$ and $\mathtt{I}[\mathbf{x}, t]$. This algorithm rectifies the reference template, $\mathbf{T}[f(\mathbf{x}, \delta\boldsymbol{\mu})]$ in order to compensate for the error produced when rectifiying the current image with the motion parameters of the previous one, $\mathbf{I}[f(\mathbf{x}, \boldsymbol{\mu}_t), t + \delta t]$. Baker and Matthews called this minimization *inverse* because it exchanged the role that the template and the rectified image had in the original work of Lucas and Kanade.

In order to solve the minimisation in (7) with a Gauss-Newton procedure, we make a first order Taylor expansion of the reference template term,

$$\mathbf{T}[f(\mathbf{x}, \delta\boldsymbol{\mu})] = \mathbf{T}[f(\mathbf{x}, \mathbf{0})] + \mathtt{M}\delta\boldsymbol{\mu} = \mathbf{T}[\mathbf{x}] + \mathtt{M}\delta\boldsymbol{\mu}, \tag{8}$$

where

$$\mathtt{M} = \left.\frac{\partial \mathbf{T}[f(\mathbf{x}, \boldsymbol{\mu})]}{\partial \boldsymbol{\mu}}\right|_{\boldsymbol{\mu}=\mathbf{0}},$$

is the Jacobian of the grey levels in the reference template w.r.t. the motion parameters and function $f(\mathbf{x}, \boldsymbol{\mu})$ is chosen such that $f(\mathbf{x}, \mathbf{0}) = \mathbf{x}$.

Introducing (8) into (7) the minimisation can be rewritten as

$$\min_{\delta\boldsymbol{\mu}} \parallel \mathbf{I}[f(\mathbf{x}, \boldsymbol{\mu}_t), t + \delta t] - \mathbf{T}[\mathbf{x}] - \mathtt{M}\delta\boldsymbol{\mu} \parallel^2,$$

which can be solved by least squares

$$\delta\boldsymbol{\mu} = (\mathtt{M}^\top \mathtt{M})^{-1}\mathtt{M}\mathcal{E}(t + \delta t), \tag{9}$$

where $\mathcal{E}(t + \delta t) = \mathbf{I}[f(\mathbf{x}, \boldsymbol{\mu}_t), t + \delta t] - \mathbf{T}[\mathbf{x}]$. Note that matrix $\mathtt{M}$ is constant, since it does not depend on $\boldsymbol{\mu}_t$. So $(\mathtt{M}^\top\mathtt{M})^{-1}\mathtt{M}$ and can be precomputed off-line. This is the key for the efficiency of this algorithm.

Note also that the Jacobian of pixel $\mathbf{x}$ with respect to the model parameters in the reference template, $\mathtt{M}$, is a matrix whose values are our *a priori* knowledge about the target structure, that is, how the grey value of each pixel in the reference template changes as the object moves infinitesimally. It represents the information provided by each template pixel to the tracking process. When $\mathtt{M}^\top\mathtt{M}$ is singular the motion parameters cannot be recovered, this would be a generalisation of the so called *aperture problem* in the estimation of optical flow.

Once $\delta\boldsymbol{\mu}$ is known, the last step is to update $\boldsymbol{\mu}_t$. Introducing $\mathbf{x}' = f(\mathbf{x}, \delta\boldsymbol{\mu})$ into (7) we get the equivalent minimisation

$$\min_{\delta\boldsymbol{\mu}} \parallel \mathbf{I}[f(f^{-1}(\mathbf{x}', \delta\boldsymbol{\mu}), \boldsymbol{\mu}_t), t + \delta t] - \mathbf{T}[\mathbf{x}'] \parallel^2,$$

from where we can conclude that $f(\mathbf{x}', \boldsymbol{\mu}_{t+\delta t}) = f(f^{-1}(\mathbf{x}', \delta\boldsymbol{\mu}), \boldsymbol{\mu}_t)$. Which means that the update of the motion parameters is compositional.

The online computation performed by this tracking procedure is quite small and consists of the warping of N pixels (a fast operation using conventional software) the subtraction of N pixels to compute $\mathcal{E}(\mathbf{x}, t + \delta t)$, and the multiplication of this result by the $n \times N$ matrix $(\mathtt{M}^\top\mathtt{M})^{-1}\mathtt{M}^\top$, where $n = \dim(\boldsymbol{\mu})$.

4 Efficient non-rigid 3D tracking

In this section we will describe how to extend the Inverse Compositional Alignment Algorithm to compute efficiently the 3D rigid and non-rigid motion of a deformable object using the model provided by the procedure described in section 2. In order to improve the robustness of the tracker we attach a small textured planar patch to each point of the 3D model. These patches (of size p pixels) are tangent to the 3D volume of the model at each point X_i. The texture of each patch is the result of projecting orthogonally the texture of the model around the point onto the patch (see Fig. 1).

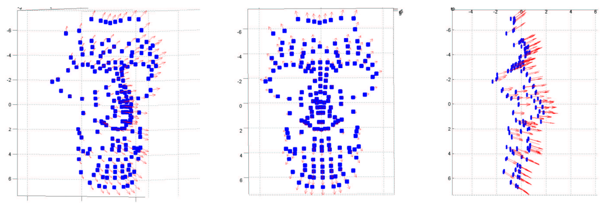

Figure 1: Points in the rigid component of the model with texture patches attached to them.

4.1 Motion model

Let Ω=$\{\mathbf{X}_1, \cdots, \mathbf{X}_{N_p}\}$ be the set of 3D points in all N patches, let $\mathtt{T}[\mathbf{X}_i]$, be the grey level of point $\mathbf{X}_i$. The 3D motion of $\mathbf{X}_i$ can be described by the composition of a rigid, $f_r(\mathbf{X}_i, \boldsymbol{\mu}_r) = \mathtt{R}\mathbf{X}_i + \mathbf{T}$, and a non-rigid, $f_n(\mathbf{X}_i, \boldsymbol{\mu}_n) = \mathbf{X}_i + \mathtt{B}\boldsymbol{\mu}_n$, motion model,

$$f_r(f_n(\mathbf{X}_i, \boldsymbol{\mu}_n), \boldsymbol{\mu}_r) = \mathtt{R}(\mathbf{X}_i + \mathtt{B}\boldsymbol{\mu}_n) + \mathbf{T},$$

where $\mathtt{B}_{3\times K}$ is a matrix storing the basis shapes of $\mathbf{X}_i$, $\mathtt{R}(\alpha, \beta, \gamma)$ a rotation matrix, $\mathbf{T}(t_x, t_y, t_z)$ a translation vector, $\boldsymbol{\mu}_r = (\alpha, \beta, \gamma, t_x, t_y, t_z)^\top$ the vector of rigid parameters and $\boldsymbol{\mu}_n = (l_1, l_2, \ldots, l_K)^\top$ the vector of configuration weights.

4.2 Incremental alignment

In order to simplify the equations in this section we will use vector notation. Let $\mathbf{X}_{3N_p\times 1} = (\mathbf{X}_1^\top, \mathbf{X}_2^\top, \ldots, \mathbf{X}_{N_p}^\top)^\top$ be the result of stacking in a vector the coordinates of the N_p 3D points and let $\mathbf{f}_r(\mathbf{X}, \boldsymbol{\mu}_r)$ and $\mathbf{f}_n(\mathbf{X}, \boldsymbol{\mu}_n)$ be the vectorial forms of $f_r(\mathbf{X}_i, \boldsymbol{\mu}_r)$ and $f_n(\mathbf{X}_i, \boldsymbol{\mu}_n)$. The *reference template*, $\mathbf{T}(\mathbf{X})$, is a $N_p \times 1$ column vector containing the grey level of all the pixels in all the patches associated to the N_p points in Ω.

As described in section 3, tracking using the ICA algorithm consists of minimising the following cost function

$$\min_{\delta\boldsymbol{\mu}_r, \delta\boldsymbol{\mu}_n} ||\mathbf{I}[\mathbf{f}_r(\mathbf{f}_n(\mathbf{X}, \boldsymbol{\mu}_{n_t}), \boldsymbol{\mu}_{r_t}), t + \delta t] - \mathbf{T}[\mathbf{f}_r(\mathbf{f}_n(\mathbf{X}, \delta\boldsymbol{\mu}_n), \delta\boldsymbol{\mu}_r)]||^2. \tag{10}$$

We will estimate the minimum of (10) in two steps. First we will minimise the rigid component of motion assuming $\delta\boldsymbol{\mu}_n \approx 0$, then, minimise the non-rigid component assuming $\delta\boldsymbol{\mu}_r \approx 0$. This will provide us with initial estimates of $\delta\boldsymbol{\mu}_{n_{t+\delta t}}$ and $\delta\boldsymbol{\mu}_{r_{t+\delta t}}$, which can be used to rectify $\mathbf{I}[\mathbf{x}, t + \delta t]$. This procedure can be repeated until $\delta\boldsymbol{\mu}_r \approx 0$ and $\delta\boldsymbol{\mu}_n \approx 0$.

4.2.1 Estimation of rigid motion

If we assume $\delta\boldsymbol{\mu}_n \approx 0$, then (10) can be rewritten as

$$\min_{\delta\boldsymbol{\mu}_r} ||\mathbf{I}[\mathbf{f}_r(\mathbf{f}_n(\mathbf{X}, \boldsymbol{\mu}_{n_t}), \boldsymbol{\mu}_{r_t}), t + \delta t] - \mathbf{T}[\mathbf{f}_r(\mathbf{X}, \delta\boldsymbol{\mu}_r)]||^2.$$

Now the rigid Jacobian matrix is

$$\mathtt{M}_r = \left.\frac{\partial \mathbf{T}[\mathbf{f}_r(\mathbf{X}, \boldsymbol{\mu})]}{\partial \boldsymbol{\mu}}\right|_{\boldsymbol{\mu}=\mathbf{0}}$$

and the incremental rigid motion can be written as $\delta\boldsymbol{\mu}_r = (\mathtt{M}_r^\top \mathtt{M}_r)^{-1}\mathtt{M}_r\mathcal{E}(t + \delta t)$. Finally, considering that $\mathbf{X} = \mathbf{f}_r^{-1}(\mathbf{Y}, \delta\boldsymbol{\mu}_r) = \delta\mathtt{R}^\top\mathbf{Y} - \delta\mathtt{R}^\top\delta\mathbf{T}$ we may write:

$$\mathbf{f}_r(\mathbf{f}_n(\mathbf{f}_r^{-1}(\mathbf{Y}, \delta\boldsymbol{\mu}_r), \boldsymbol{\mu}_{n_t}), \boldsymbol{\mu}_{r_t}) = \mathtt{R}_t\delta\mathtt{R}^\top(\mathbf{Y} + \delta\mathtt{R}\mathtt{B}\boldsymbol{\mu}_{r_t}) + \mathbf{T}_t - \mathtt{R}_t\delta\mathtt{R}^\top\delta\mathbf{T}.$$

So the new rigid motion parameters are:

$$\mathtt{R}_{t+\delta t} = \mathtt{R}_t\delta\mathtt{R}^\top; \quad \mathbf{T}_{t+\delta t} = \mathbf{T}_t - \mathtt{R}_t\delta\mathtt{R}^\top\delta\mathbf{T}. \tag{11}$$

4.2.2 Estimation of non-rigid motion

Assuming now $\delta\boldsymbol{\mu}_r \approx 0$, then the cost function (10) can be rewritten as

$$\min_{\delta\boldsymbol{\mu}_n} ||\mathbf{I}[\mathbf{f}_r(\mathbf{f}_n(\mathbf{X}, \boldsymbol{\mu}_{n_t}), \boldsymbol{\mu}_{r_t}), t + \delta t] - \mathbf{T}[\mathbf{f}_n(\mathbf{X}, \delta\boldsymbol{\mu}_n)]||^2,$$

and the ICA algorithm can be immediately used. In this case the error can be written as:

$$\mathcal{E}(t + \delta t) = \mathbf{I}[\mathbf{f}_r(\mathbf{f}_n(\mathbf{X}, \boldsymbol{\mu}_{n_t}), \boldsymbol{\mu}_{r_t}), t + \delta t] - T[\mathbf{X}]$$

, and the non-rigid Jacobian matrix takes the form

$$\mathtt{M}_n = \left.\frac{\partial \mathbf{T}[\mathbf{f}_n(\mathbf{X}, \boldsymbol{\mu})]}{\partial \boldsymbol{\mu}}\right|_{\boldsymbol{\mu}=\mathbf{0}}.$$

The increment of the configuration weights can then be computed using the expression:

$$\delta\boldsymbol{\mu}_n = (\mathtt{M}_n^\top \mathtt{M}_n)^{-1}\mathtt{M}_n\mathcal{E}(t + \delta t)$$

. Finally, considering that $\mathbf{X} = \mathbf{f}_n^{-1}(\mathbf{Y}, \delta\boldsymbol{\mu}_n) = \mathbf{Y} - \mathtt{B}\delta\boldsymbol{\mu}_n$ then,

$$\mathbf{f}_n(\mathbf{f}_n^{-1}(\mathbf{Y}, \delta\boldsymbol{\mu}_n), \boldsymbol{\mu}_{n_t}) = \mathbf{Y} + \mathtt{B}(\boldsymbol{\mu}_{n_t} - \delta\boldsymbol{\mu}_n).$$

So, the new configuration weight vector is

$$\delta\boldsymbol{\mu}_{n_{t+\delta t}} = \boldsymbol{\mu}_{n_t} - \delta\boldsymbol{\mu}_n. \tag{12}$$

4.3 The tracking algorithm

The final algorithm is as follows:

- **Offline:**
  1. Compute $\mathtt{M}_r$ and $\mathtt{M}_n$.
  2. Compute and store $\Lambda_r = (\mathtt{M}_r^\top \mathtt{M}_r)^{-1}\mathtt{M}_r^\top$.
  3. Compute and store $\Lambda_n = (\mathtt{M}_n^\top \mathtt{M}_n)^{-1}\mathtt{M}_n^\top$.
- **Online:**
  1. Repeat Until $\delta\boldsymbol{\mu}_r \approx 0$ and $\delta\boldsymbol{\mu}_n \approx 0$.
     - 1.1. Rectify $\mathbf{I}[\mathbf{f}_r(\mathbf{f}_n(\mathbf{X}, \boldsymbol{\mu}_{n_t}), \boldsymbol{\mu}_{r_t}), t + \delta t]$.
     - 1.2. $\mathcal{E} = \mathbf{I}[\mathbf{f}_r(\mathbf{f}_n(\mathbf{X}, \boldsymbol{\mu}_{n_t}), \boldsymbol{\mu}_{r_t}), t + \delta t] - T[\mathbf{X}]$.
     - 1.3. Compute $\delta\boldsymbol{\mu}_n = \Lambda_n\mathcal{E}$.
     - 1.4. Compute $\delta\boldsymbol{\mu}_r = \Lambda_r\mathcal{E}$.
     - 1.5. Update $\boldsymbol{\mu}_{n_t}$ using (12).
     - 1.6. Update $\boldsymbol{\mu}_{r_t}$ using (11).
  2. $\boldsymbol{\mu}_{n_{t+\delta t}} = \boldsymbol{\mu}_{n_t}$.
  3. $\boldsymbol{\mu}_{r_{t+\delta t}} = \boldsymbol{\mu}_{r_t}$.

Let $n = 6 + K$ be the number of motion parameters and N_p the number of pixels. Taking into account that usually $N_p >> n$, the complexity of our algorithm is shown in tables 1 and 2. The computation time is dominated by the image warping in step 1.1. In the case of minimising (10) by Lucas-Kanade the offline part of our algorithm would be performed on-line. Then the computation of the Jacobians and its pseudo-inverses would be the bottle neck of the minimisation.

Step (1)	Step (2)	Step (3)	Total
$O(nN_p)$	$O(6^2N_p + 6^3)$	$O(K^2N_p + K^3)$	$O((6^2 + K^2)N_p)$

Table 1: Complexity of the offline part of the algorithm in number of operations

Step (1.1)	Step (1.2)	Step (1.3)	Step (1.4)
$O(nN_p)$	$O(N_p)$	$O(KN_p)$	$O(6N_p)$

Step (1.5)	Step (1.6)	Step (2)	Step (3)	Total
$O(K)$	$O(6)$	$O(K)$	$O(6)$	$O(nN_p)$

Table 2: Complexity of the online part of the algorithm in number of operations

5 Experiments

We validate our tracking framework with two different sets of experiments designed to show the performance of the tracker with synthetic and real sequences of a human face performing different expressions. Notice that the

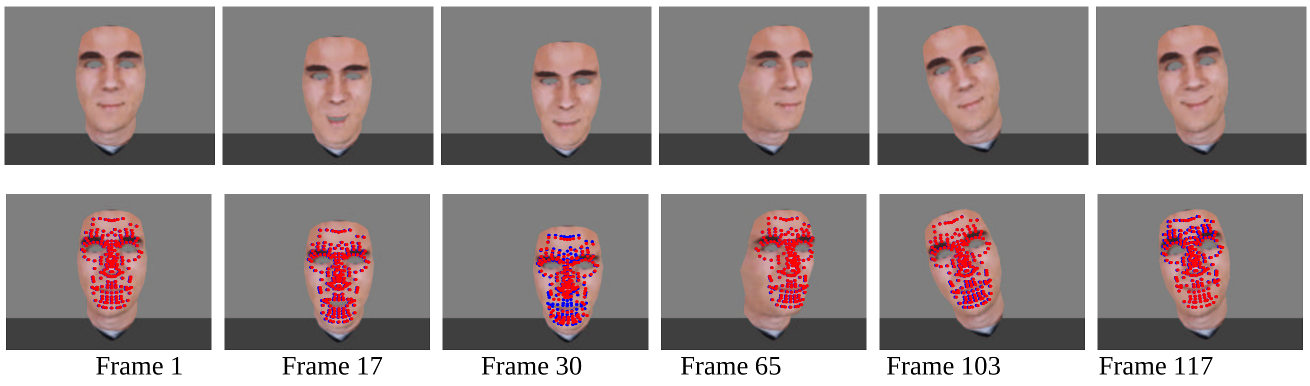

Figure 2: Synthetic experiment results for some frames. Blue dots stand for ground truth projections whereas red dots stand for actual tracked positions.

approach is valid for generic non-rigid objects given a valid description of the shape in terms of a linear combination of basis shapes. Synthetically generated data is used to compare the output of the tracking algorithm with the ground truth position of the 2D points.

5.1 Synthetic data

We have generated a sequence using a synthetic face model originally developed by Parke et. al. [13]. This is a 3D model which encodes 18 different muscles of the face. Animating the face model to generate facial expressions is achieved by actuating on the different facial muscles. Then, the generated 3D shape is projected and rendered onto the image plane using a free tool for ray-tracing [1]. The head translates along the x axis while it rotates around its three canonical axes for the entire 125 frame sequence. Deformations occur twice between frames 1–50 and frames 100–125 and the non-rigid motion is mainly located in the mouth and eyebrows region. Key frames of the output sequence are shown in the first row of Figure (2).

The non-rigid 3D model is directly computed from 194 image point projections of the described synthetic face at each frame. We apply the non-linear optimization approach described in Section 2 with a number of basis shapes that has been fixed to $K = 9$ by considering a dimensionality enough to contain 95% of the total energy encoded in the SVD singular values. The motion associated with the rigid component was used to initialize the bundle adjustment minimization, while the configuration weights were initialized to a small value, and the algorithm converged smoothly after 16 iterations.

The generated 3D model is directly applied to the efficient non-rigid tracking framework on a sequence with the same set of face deformations but different rigid motion (in particular, we emphasize the effect of projective distortions). Note that here we want to test the performance of the tracker in the case where the model describes perfectly the range of facial deformations. An initialization process is required to align the model with the first frame. This issue arises because the non-rigid factorization uses an affine camera model while the efficient tracking assumes a full projective one. As initial guess for this initialization we use a rough projection matrix computed from the 3D model and the ground truth projections in the first frame of the sequence. We subsequently optimize the initial solution by carrying out a non-linear optimization essentially similar to our approach for model generation.

Once the model is aligned in the first frame of the sequence, our efficient tracking algorithm is applied to the images. For each one of the 194 points of the model, a tangent 3D square patch of 0.3 units size is generated. From each patch we sample 9 points which extend our model up to 1746 points. These points are projected onto the first image of the sequence giving us the intensity values for our *reference template*.

Figure (2) shows the actual tracked points (red dots) compared with the ground truth projections (blue dots) for certain key frames. The RMS of the reprojection error for each frame of the sequence is computed as well (see Figure. 3). As expected, the reprojection error for the initial frame is different from zero. This is due to the alignment error introduced by the affine model. A peak in the error occurs roughly at frame 30 as the model opens its mouth and rotates around the x axis.

[1] See http://www.povray.org

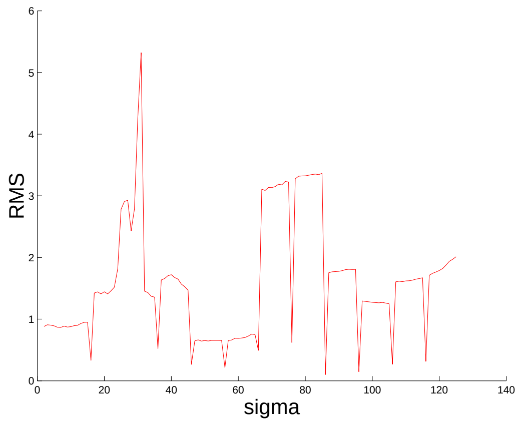

Figure 3: Reprojection error (distance in pixels) vs. frame number for the sequence.

5.2 Real Data

Our experiments with real data show the performance of the non-rigid model generation and efficient tracking algorithms. We generated a reliable 3D non-rigid model by using a set of 2D points manually tracked from an uncalibrated real video sequence of a subject performing different facial expressions and head rotations. This 3D model is subsequently used to initialize the tracking algorithm on a different video sequence but with the same subject. We perform the initial alignment by matching manually the 3D points and their projections on the image plane.

Two different sequences are used to show the performance of our tracker. First we use a 173 frames sequence where only rigid motion is shown (see Figure 4) where the motion is restricted to a rotation around y axis. Figure 4 shows the actual tracked positions for each point of the model.

A 190 frames sequence showing only face deformations is used to verify how the points are tracked when the model deforms. Results for several key frames showing expressions can be seen in Figure 5. The efficient tracking can cope with different degrees of rigid motion as shown in Figure (4). In the second sequence, where only non-rigid deformations are present, the tracker experiences some difficulties in following the non-rigid motion as shown in Figure (5).

6 Conclusions

In this paper we have presented an efficient model-based 3D tracking algorithm. The model is represented as a linear

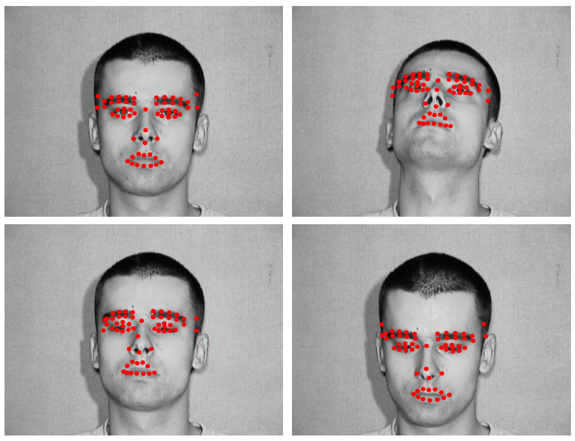

Figure 4: Real experiment results for the rigid motion tracking. The key frames 1, 59, 129 and 173 shows the algorithm tracking different head poses. Red dots stand for the actual tracked positions.

combination of shape bases generated automatically from a set of 2D correspondences in an uncalibrated monocular video sequence. Once the model is generated it can be subsequently used for tracking using our efficient 3D tracker which is an extension of the Inverse Compositional Alignment algorithm to the case of a projective camera model. We have demonstrated the performance of the model generation and the 3D tracking in synthetic and real image sequences. Future work includes the automatic update of the 3D deformable model during tracking to extend the range of deformations that the 3D model is able to cope with.

Acknowledgment

The authors would like to thank the Royal Society European Science Exchange Programme, EPSRC Grant GR/S61539/01 and the Spanish Ministry of Science project TIC2002-00591 for financial support. Alessio Del Bue holds a Queen Mary Studentship award. Enrique Muñoz was funded by a FPU grant from the Spanish Ministry of Education.

References

[1] S. Baker and I. Matthews. Lukas-kanade 20 years on: A unifiying framework. *Int. Journal of Computer Vision*, 56(3):221–255, 2004.

[2] Simon Baker and Ian Matthews. Equivalence and efficiency of image alignment algorithms. In *Proc. of International Conference on Computer Vision and*

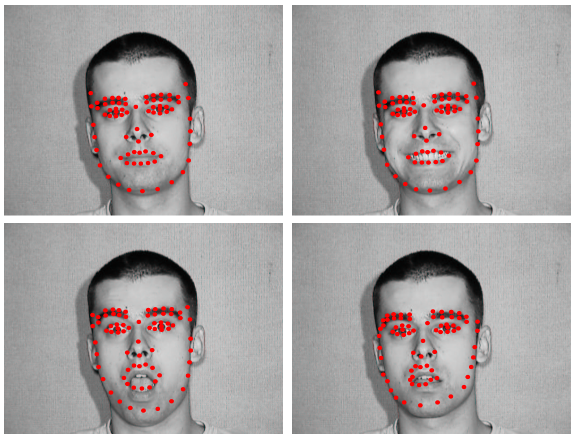

Figure 5: Real experiment results for deformation tracking. The key frames 1, 30, 138 and 190 shows different facial expressions of the subject. Red dots stand for the actual tracked positions.

Pattern Recognition, volume 1, pages I–1090–I–1097. IEEE, 2001.

[3] M. Brand. Morphable models from video. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition, Kauai, Hawaii*, December 2001.

[4] M. Brand and Bhotika R. Flexible flow for 3d nonrigid tracking and shape recovery. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition, Kauai, Hawaii*, pages 315–22, December 2001.

[5] C. Bregler, A. Hertzmann, and H Biermann. Recovering non-rigid 3d shape from image streams. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition, Hilton Head, South Carolina*, pages 690–696, jun 2000.

[6] A. Del Bue and L. Agapito. Non-rigid 3d shape recovery using stereo factorization. In *Asian Conference of Computer Vision*, volume 1, pages 25–30, Jeju, South Korea, January 2004.

[7] A. Del Bue, F. Smeraldi, and L. Agapito. Non-rigid structure from motion using non-parametric tracking and non-linear optimization. In *IEEE Workshop on Articulated and Nonrigid Motion, CVPR2004, Washington DC, USA*, 2004.

[8] Douglas DeCarlo and Dimitri Metaxas. Optical flow constraints on deformable models with applications to face tracking. *International Journal of Computer Vision*, 38(2):99–127, 2000.

[9] P. Eisert and B. Girod. Analyzing facial expressions for virtual conferencing. *IEEE Computer Graphics and Applications: Special Issue: Computer Animation for Virtual Humans*, 18(5):70–78, 1998.

[10] ”S. Gokturk, J. Bouget, and R. Gzreszocuk”. A data-driven model for monocular face tracking. In *Proc. 8th International Conference on Computer Vision, Vancouver, Canada*, 2001.

[11] Gregory D. Hager and Peter N. Belhumeur. Efficient region tracking with parametric models of geometry and illumination. *IEEE Transactions on Pattern Analisys and Machine Intelligence*, 20(10):1025–1039, 1998.

[12] Bruce D. Lucas and Takeo Kanade. An iterative image registration technique with an application to stereo vision. In *Proc. of Imaging Understanding Workshop*, pages 121–130, 1981.

[13] Frederick I. Parke and Keith Waters. *Computer Facial Animation*. AK Peters Ltd, 1996.

[14] S. Romdhani and T. Vetter. Efficient, robust and accurate fitting of a 3d morphable model. In *IEEE International conference on Computer Vision 2003*, 2003.

[15] Heung-Yeung Shum and Richard Szeliski. Construction of panoramic image mosaics with global and local alignment. *International Journal of Computer Vision*, 36(2):101–130, 2000.

[16] C. Tomasi and T. Kanade. Shape and motion from image streams: a factorization method. *International Journal in Computer Vision*, 9(2):137–154, 1991.

[17] L. Torresani, D. Yang, E. Alexander, and C. Bregler. Tracking and modeling non-rigid objects with rank constraints. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition, Kauai, Hawaii*, 2001.

[18] Bill Triggs, Philip McLauchlan, Richard Hartley, and Andrew Fitzgibbon. Bundle adjustment – A modern synthesis. In W. Triggs, A. Zisserman, and R. Szeliski, editors, *Vision Algorithms: Theory and Practice*, LNCS, pages 298–375. Springer Verlag, 2000.

[19] T. Vetter and V. Blanz. A morphable model for the synthesis of 3d faces. In *Proceedings of the ACM SIGGRAPH Conference on Computer Graphics*, pages 187–194, 1999.