3D Deformable Face Tracking with a Commodity Depth Camera

Qin Cai[1], David Gallup[2], Cha Zhang[1], and Zhengyou Zhang[1]

[1] Communication and Collaboration Systems Group, Microsoft Research
One Microsoft Way, Redmond, WA 98052 USA
[2] Dept. of Computer Science, UNC at Chapel Hill
Sitterson Hall, UNC-Chapel Hill, Chapel Hill, NC 27599 USA

Abstract. Recently, there has been an increasing number of depth cameras available at commodity prices. These cameras can usually capture both color and depth images in real-time, with limited resolution and accuracy. In this paper, we study the problem of 3D deformable face tracking with such commodity depth cameras. A regularized maximum likelihood deformable model fitting (DMF) algorithm is developed, with special emphasis on handling the noisy input depth data. In particular, we present a maximum likelihood solution that can accommodate sensor noise represented by an arbitrary covariance matrix, which allows more elaborate modeling of the sensor's accuracy. Furthermore, an ℓ_1 regularization scheme is proposed based on the semantics of the deformable face model, which is shown to be very effective in improving the tracking results. To track facial movement in subsequent frames, feature points in the texture images are matched across frames and integrated into the DMF framework seamlessly. The effectiveness of the proposed method is demonstrated with multiple sequences with ground truth information.

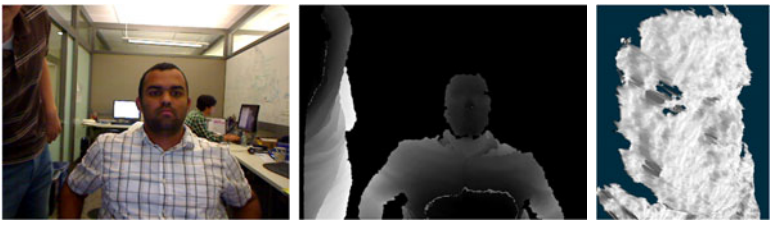
1 Introduction

Tracking non-rigid objects, in particular human faces, is an active research area for many applications in human computer interaction, performance-driven facial animation, and face recognition. The problem is still largely unsolved, as usually for 3D deformable face models there are dozens of parameters that need to be estimated from the limited input data.

A number of works in the literature have focused on 3D deformable face tracking based only on videos. There are mainly two categories of algorithms: (1) appearance based, which uses generative linear face appearance models such as active appearance models (AAMs) [1] and 3D morphable models [2] to capture the shape and texture variations of faces, and (2) feature based, which uses active shape models [3] or other features [4] for tracking. Appearance based algorithms may suffer from insufficient generalizability of AAMs due to lighting and texture variations, while feature based algorithms may lose tracking due to the lack of semantic features, the occlusions of profile poses, etc.

Another large body of works considered fitting morphable models to 3D scans of faces [5,6,7,8,9]. These 3D scans are usually obtained by laser scanners or

K. Daniilidis, P. Maragos, N. Paragios (Eds.): ECCV 2010, Part III, LNCS 6313, pp. 229–242, 2010.



(a) (b) (c)

Fig. 1. Data captured by a commodity depth camera. (a) Texture image; (b) depth image; (c) enlarged face region rendered from another viewpoint.

structured light systems, which have very high quality. Fitting these high quality range data with a morphable face model usually involves the well-known iterative closest point (ICP) algorithm [10] and its variants [11], and the results are generally very good. The downside, however, is that these capturing systems are usually very expensive to acquire or operate.

Recently, depth cameras based on time-of-flight or other principles became available at commodity prices, such as 3DV systems and Canesta. Fig. 1 shows some captured data from our test depth camera, which derives depth information from infrared light patterns and triangulation. The camera is capable of recording both texture and depth images with 640×480 pixels resolution at 30 frames per second (fps). In general the depth information is very accurate, though a closer look at the face region (Fig. 1(c)) shows that it is still much noisier than laser scanned results.

In this paper, we propose a regularized maximum likelihood deformable model fitting (DMF) algorithm for 3D face tracking with a commodity depth camera. Compared with existing approaches, this paper has two major contributions. First, unlike most previous works on DMF, we do not assume an identity covariance matrix for the depth sensor noise. This leads to a more general maximum likelihood solution with arbitrary noise covariance matrices, which is shown to be effective for our noisy depth data. Second, the noisy depth data also require regularization in the ICP framework. We propose a novel ℓ_1 regularization scheme inspired by the semantics of our deformable face model, which improves the tracking performance significantly.

2 Related Work

There is a large amount of literature in facial modeling and tracking. We refer the reader to the survey by Murphy-Chutorian and Trivedi [12] for an overview.

Many models have been explored for face animation and tracking. Parametric models use a set of parameters to describe the articulation of the jaw, eyebrow position, opening of the mouth, and other features that comprise the state of the face [13]. Physics-based models seek to simulate the facial muscle and tissue [14]. Blanz and Vetter [2] discovered that the manifold of facial expression

and appearance can be effectively modeled as a linear combination of exemplar faces. This morphable model is computed from a large database of registered laser scans, and this approach has proven useful for face synthesis [2], expression transfer [8], recognition [5], and tracking [15]. For tracking, a subject-specific morphable model can be constructed [9], which requires each subject to undergo an extensive training phase before tracking can be performed. In contrast, we use a generic morphable model constructed by an artist, which is first fit to the subject during initialization. Only a few frames with neutral faces are required to automatically compute the subject-specific appearance parameters before tracking.

Several approaches have used range data for face modeling and tracking. Zhu and Fujimura [6] used range data as an additional image channel in optical flow-based tracking. Methods that rely solely on visual appearance will be sensitive to lighting conditions and changes, whereas many ranging techniques are unaffected by lighting conditions. Many methods, such as that of Zhang et al. [7], used structured light or other active ranging techniques. The structured light systems in [7,8,9] required a camera, a projector, and in some cases synchronization circuitry. This hardware is not uncommon, but still expensive to acquire and operate. This paper will study deformable face tracking with a commodity depth camera, which is projected to cost under $100 in the next few years, and has lower resolution and less accuracy than structured light systems. A key part of our method is thus to model the sensor noise and add regularization to improve the tracking performance. Note uncertainty on measurements has been considered in other contexts such as motion analysis for mobile robot navigation [16], though we are not aware of similar work in the context of deformable face tracking.

Iterative closest point (ICP) is a common approach for aligning shapes, such as range images of faces. Besl et al. [10] proposed the ICP algorithm for rigid shape alignment, and variants have been proposed for nonrigid alignment [11]. Lu and Jian [17] used ICP for face matching, and applied ICP in deformable model fitting as an intermediate step assuming the deformation is fixed. ICP has also been used in face recognition [18] and real-time tracking [9]. Note in model fitting and tracking applications, regularization is a common technique to stabilize the final results [11,9]. However, the ℓ_1 regularization that will be introduced in Section 4.5 has not be used in previous works, and its performance improvement is rather significant.

3 Linear Deformable Model

We use a linear deformable model constructed by an artist to represent possible variations of a human face [19], which could also be constructed semi-automatically [2]. The head model is defined as a set of K vertices $\mathcal{P}$ and a set of facets $\mathcal{F}$. Each vertex $\mathbf{p}_k \in \mathcal{P}$ is a point in $\mathbb{R}^3$, and each facet $f \in \mathcal{F}$ is a set of three or more vertices from the set $\mathcal{P}$. In our head model, all facets have exactly 3 vertices. In addition, the head model is augmented with two artist-defined deformation matrices: the static deformation matrix $\mathbf{B}$ and the action

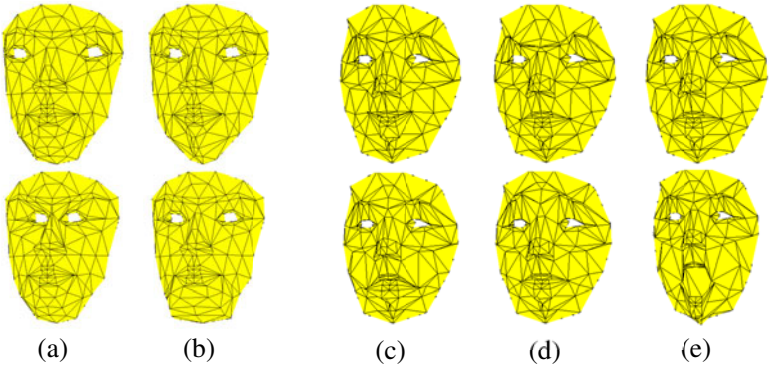


Fig. 2. Example deformations of our 3D face model. (a)(b) Static deformations; (c)(d)(e) action deformations.

deformation matrix $\mathbf{A}$. According to weighting vectors $\mathbf{s}$ and $\mathbf{r}$, they transform the mesh linearly into a target head model $\mathcal{Q}$ as follows:

$$\begin{bmatrix} \mathbf{q}_1 \\ \vdots \\ \mathbf{q}_K \end{bmatrix} = \begin{bmatrix} \mathbf{p}_1 \\ \vdots \\ \mathbf{p}_K \end{bmatrix} + \mathbf{A} \begin{bmatrix} r_1 \\ \vdots \\ r_N \end{bmatrix} + \mathbf{B} \begin{bmatrix} s_1 \\ \vdots \\ s_M \end{bmatrix}, \tag{1}$$

where M and N are the number of deformations in $\mathbf{B}$ and $\mathbf{A}$, $\alpha_m \leq s_m \leq \beta_m, m = 1, \cdots, M$, and $\theta_n \leq r_n \leq \phi_n, n = 1, \cdots, N$ are ranges specified by the artist. The static deformations in $\mathbf{B}$ are characteristic to a particular face, such as enlarging the distance between eyes, or extending the chin. The action deformations include opening the mouth, raising eyebrows, etc. Some example deformations of our model are shown in Fig. 2.

4 Regularized Maximum Likelihood DMF

4.1 Problem Formulation

Let $\mathcal{P}$ represent the vertices of our head model, and $\mathcal{G}$ represent the 3D points acquired from the depth camera. We want to compute the rotation $\mathbf{R}$ and translation $\mathbf{t}$ between the head model and the depth camera, as well as the deformation parameters $\mathbf{r}$ and $\mathbf{s}$. We formulate the problem as below.

Following the procedure of ICP [10], let us assume that in a certain iteration, a set of point correspondences between the deformable model and the depth image is available. For each correspondence $(\mathbf{p}_k, \mathbf{g}_k)$, $\mathbf{g}_k \in \mathcal{G}$, we have the equation:

$$\mathbf{R}(\mathbf{p}_k + \mathbf{A}_k\mathbf{r} + \mathbf{B}_k\mathbf{s}) + \mathbf{t} = \mathbf{g}_k + \mathbf{x}_k \tag{2}$$

where $\mathbf{A}_k$ and $\mathbf{B}_k$ represent the three rows of $\mathbf{A}$ and $\mathbf{B}$ that correspond to vertex k. $\mathbf{x}_k$ is the depth sensor noise, which can be assumed to follow a zero

mean Gaussian distribution $\mathcal{N}(0, \mathbf{\Sigma}_{\mathbf{x}_k})$. The maximum likelihood solution of the unknowns $\mathbf{R}$, $\mathbf{t}$, $\mathbf{r}$ and $\mathbf{s}$ can be derived by minimizing:

$$J_1(\mathbf{R}, \mathbf{t}, \mathbf{r}, \mathbf{s}) = \frac{1}{K} \sum_{k=1}^{K} \mathbf{x}_k^T \mathbf{\Sigma}_{\mathbf{x}_k}^{-1} \mathbf{x}_k, \tag{3}$$

where $\mathbf{x}_k = \mathbf{R}(\mathbf{p}_k + \mathbf{A}_k\mathbf{r} + \mathbf{B}_k\mathbf{s}) + \mathbf{t} - \mathbf{g}_k$. $\mathbf{r}$ and $\mathbf{s}$ are subject to inequality constraints, namely, $\alpha_m \leq s_m \leq \beta_m, m = 1, \cdots, M$, and $\theta_n \leq r_n \leq \phi_n, n = 1, \cdots, N$. Additional regularization terms may be added to the above optimization problem, which will be discussed further in Section 4.5.

A useful variation is to substitute the point-to-point distance with point-to-plane distance [20]. The point-to-plane distance allows the model to slide tangentially to the surface, which speeds up convergence and makes it less likely to get stuck in local minima. Distance to the plane can be computed using the surface normal, which can be computed from the head model based on the current iteration's head pose. Let the surface normal of point $\mathbf{p}_k$ in the head model coordinate be $\mathbf{n}_k$. The point-to-plane distance can be computed as:

$$y_k = (\mathbf{R}\mathbf{n}_k)^T \mathbf{x}_k, \tag{4}$$

The maximum likelihood solution is thus obtained by minimizing:

$$J_2(\mathbf{R}, \mathbf{t}, \mathbf{r}, \mathbf{s}) = \frac{1}{K} \sum_{k=1}^{K} \frac{y_k^2}{\sigma_{y_k}^2}, \tag{5}$$

where $\sigma_{y_k}^2 = (\mathbf{R}\mathbf{n}_k)^T \mathbf{\Sigma}_{\mathbf{x}_k} (\mathbf{R}\mathbf{n}_k)$, and $\alpha_m \leq s_m \leq \beta_m, m = 1, \cdots, M$, and $\theta_n \leq r_n \leq \phi_n, n = 1, \cdots, N$.

Given the correspondence pairs $(\mathbf{p}_k, \mathbf{g}_k)$, since both the point-to-point and the point-to-plane distances are nonlinear, we resort to a solution that solves for $\mathbf{r}$, $\mathbf{s}$ and $\mathbf{R}$, $\mathbf{t}$ in an iterative fashion. For ease of understanding, we present the solution for identity noise covariance matrix in Section 4.2 first, and extend it to arbitrary covariance matrix in Section 4.3.

4.2 Iterative Solution for Identity Noise Covariance Matrix

We first assume the depth sensor noise covariance matrix is a scaled identity matrix, i.e., $\mathbf{\Sigma}_{\mathbf{x}_k} = \sigma^2 \mathbf{I}_3$, where $\mathbf{I}_3$ is a 3×3 identity matrix. Further, let $\tilde{\mathbf{R}} = \mathbf{R}^{-1}$, $\tilde{\mathbf{t}} = \tilde{\mathbf{R}}\mathbf{t}$, and

$$\mathbf{y}_k = \tilde{\mathbf{R}}\mathbf{x}_k = \mathbf{p}_k + \mathbf{A}_k\mathbf{r} + \mathbf{B}_k\mathbf{s} + \tilde{\mathbf{t}} - \tilde{\mathbf{R}}\mathbf{g}_k. \tag{6}$$

Since $\mathbf{x}_k^T \mathbf{x}_k = (\mathbf{R}\mathbf{y}_k)^T (\mathbf{R}\mathbf{y}_k) = \mathbf{y}_k^T \mathbf{y}_k$, the likelihood function can be written as:

$$J_1(\mathbf{R}, \mathbf{t}, \mathbf{r}, \mathbf{s}) = \frac{1}{K\sigma^2} \sum_{k=1}^{K} \mathbf{x}_k^T \mathbf{x}_k = \frac{1}{K\sigma^2} \sum_{k=1}^{K} \mathbf{y}_k^T \mathbf{y}_k. \tag{7}$$

Similarly, for point-to-plane distance, since $y_k = (\mathbf{R}\mathbf{n}_k)^T\mathbf{x}_k = \mathbf{n}_k^T\mathbf{R}^T\mathbf{R}\mathbf{y}_k = \mathbf{n}_k^T\mathbf{y}_k$, and $\sigma_{y_k}^2 = (\mathbf{R}\mathbf{n}_k)^T\mathbf{\Sigma}_{\mathbf{x}_k}(\mathbf{R}\mathbf{n}_k) = \sigma^2$, we have:

$$J_2(\mathbf{R},\mathbf{t},\mathbf{r},\mathbf{s}) = \frac{1}{K\sigma^2}\sum_{k=1}^{K}\mathbf{y}_k^T\mathbf{N}_k\mathbf{y}_k, \tag{8}$$

where $\mathbf{N}_k = \mathbf{n}_k\mathbf{n}_k^T$.

We may decompose the rotation matrix $\tilde{\mathbf{R}}$ into an initial rotation matrix $\tilde{\mathbf{R}}_0$ and an incremental rotation matrix $\Delta\tilde{\mathbf{R}}$, where the initial rotation matrix can be the rotation matrix of the head in the previous frame, or an estimation of $\tilde{\mathbf{R}}$ obtained in another algorithm. In other words, let $\tilde{\mathbf{R}} = \Delta\tilde{\mathbf{R}}\tilde{\mathbf{R}}_0$. Since the rotation angle of the incremental rotation matrix is small, we may linearize it as:

$$\Delta\tilde{\mathbf{R}} \approx \begin{bmatrix} 1 & -\omega_3 & \omega_2 \\ \omega_3 & 1 & -\omega_1 \\ -\omega_2 & \omega_1 & 1 \end{bmatrix}, \tag{9}$$

where $\omega = [\omega_1,\omega_2,\omega_3]^T$ is the corresponding small rotation vector. Further, let $\mathbf{q}_k = \tilde{\mathbf{R}}_0\mathbf{g}_k = [q_{k1},q_{k2},q_{k3}]^T$, we can write the variable $\mathbf{y}_k$ in the form of unknowns $\mathbf{r}$, $\mathbf{s}$, $\tilde{\mathbf{t}}$ and ω as:

$$\mathbf{y}_k = \mathbf{p}_k + \mathbf{A}_k\mathbf{r} + \mathbf{B}_k\mathbf{s} + \tilde{\mathbf{t}} - \Delta\tilde{\mathbf{R}}\mathbf{q}_k \approx (\mathbf{p}_k - \mathbf{q}_k) + [\mathbf{A}_k,\mathbf{B}_k,\mathbf{I}_3,[\mathbf{q}_k]_\times]\begin{bmatrix}\mathbf{r}\\ \mathbf{s}\\ \tilde{\mathbf{t}}\\ \omega\end{bmatrix} \tag{10}$$

where $[\mathbf{q}_k]_\times$ is the skew-symmetric matrix of $\mathbf{q}_k$:

$$[\mathbf{q}_k]_\times = \begin{bmatrix} 0 & -q_{k3} & q_{k2} \\ q_{k3} & 0 & -q_{k1} \\ -q_{k2} & q_{k1} & 0 \end{bmatrix}. \tag{11}$$

Let $\mathbf{H}_k = [\mathbf{A}_k,\mathbf{B}_k,\mathbf{I}_3,[\mathbf{q}_k]_\times]$, $\mathbf{u}_k = \mathbf{p}_k - \mathbf{q}_k$, and $\mathbf{z} = \left[\mathbf{r}^T,\mathbf{s}^T,\tilde{\mathbf{t}}^T,\omega^T\right]^T$, we have:

$$\mathbf{y}_k = \mathbf{u}_k + \mathbf{H}_k\mathbf{z}. \tag{12}$$

Hence,

$$J_1 = \frac{1}{K\sigma^2}\sum_{k=1}^{K}\mathbf{y}_k^T\mathbf{y}_k = \frac{1}{K\sigma^2}\sum_{k=1}^{K}(\mathbf{u}_k + \mathbf{H}_k\mathbf{z})^T(\mathbf{u}_k + \mathbf{H}_k\mathbf{z}) \tag{13}$$

$$J_2 = \frac{1}{K\sigma^2}\sum_{k=1}^{K}\mathbf{y}_k^T\mathbf{N}_k\mathbf{y}_k = \frac{1}{K\sigma^2}\sum_{k=1}^{K}(\mathbf{u}_k + \mathbf{H}_k\mathbf{z})^T\mathbf{N}_k(\mathbf{u}_k + \mathbf{H}_k\mathbf{z}) \tag{14}$$

Both likelihood functions are quadratic with respect to $\mathbf{z}$. Since there are linear constraints on the range of values for $\mathbf{r}$ and $\mathbf{s}$, the minimization problem can be solved with quadratic programming [21].

The rotation vector ω is an approximation of the actual incremental rotation matrix. One can simply insert $\Delta\tilde{\mathbf{R}}\tilde{\mathbf{R}}_0$ to the position of $\tilde{\mathbf{R}}_0$ and repeat the above optimization process until it converges.

4.3 Solution for Arbitrary Noise Covariance Matrix

When the sensor noise covariance matrix is arbitrary, again we resort to an iterative solution. Note since $\mathbf{y}_k = \tilde{\mathbf{R}}\mathbf{x}_k$, we have $\mathbf{\Sigma}_{\mathbf{y}_k} = \tilde{\mathbf{R}}\mathbf{\Sigma}_{\mathbf{x}_k}\tilde{\mathbf{R}}^T$. A feasible solution can be obtained if we replace $\tilde{\mathbf{R}}$ with its estimation $\tilde{\mathbf{R}}_0$, i.e.,

$$\mathbf{\Sigma}_{\mathbf{y}_k} \approx \tilde{\mathbf{R}}_0\mathbf{\Sigma}_{\mathbf{x}_k}\tilde{\mathbf{R}}_0^T, \tag{15}$$

which is known for the current iteration. Subsequently,

$$J_1 = \frac{1}{K}\sum_{k=1}^{K}\mathbf{y}_k^T\mathbf{\Sigma}_{\mathbf{y}_k}^{-1}\mathbf{y}_k = \frac{1}{K}\sum_{k=1}^{K}(\mathbf{u}_k + \mathbf{H}_k\mathbf{z})^T\mathbf{\Sigma}_{\mathbf{y}_k}^{-1}(\mathbf{u}_k + \mathbf{H}_k\mathbf{z}) \tag{16}$$

$$J_2 = \frac{1}{K}\sum_{k=1}^{K}\frac{\mathbf{y}_k^T\mathbf{N}_k\mathbf{y}_k}{\mathbf{n}_k^T\mathbf{\Sigma}_{\mathbf{y}_k}\mathbf{n}_k} = \frac{1}{K}\sum_{k=1}^{K}\frac{(\mathbf{u}_k + \mathbf{H}_k\mathbf{z})^T\mathbf{N}_k(\mathbf{u}_k + \mathbf{H}_k\mathbf{z})}{\mathbf{n}_k^T\mathbf{\Sigma}_{\mathbf{y}_k}\mathbf{n}_k} \tag{17}$$

We still have quadratic likelihood functions with respect to $\mathbf{z}$, which can be solved via quadratic programming. Again, the minimization will be repeated until convergence by inserting $\Delta\tilde{\mathbf{R}}\tilde{\mathbf{R}}_0$ to the position of $\tilde{\mathbf{R}}_0$ in each iteration.

4.4 Multi-frame DMF for Model Initialization

In our tracking system, the above maximum likelihood DMF framework is applied differently in two stages. During the initialization stage, the goal is to fit the generic deformable model to an arbitrary person. We assume that a set of L ($L \leq 10$ in the current implementation) neutral face frames are available. The action deformation vector $\mathbf{r}$ is assumed to be zero. We jointly solve the static deformation vector $\mathbf{s}$ and the face rotations and translations as follows.

Denote the correspondences as $(\mathbf{p}_{lk}, \mathbf{g}_{lk})$, where $l = 1, \cdots, L$ represents the frame index. Assume in the previous iteration, $\tilde{\mathbf{R}}_{l0}$ is the rotation matrix for frame l. Let $\mathbf{q}_{lk} = \tilde{\mathbf{R}}_{l0}\mathbf{g}_{lk}$; $\mathbf{H}_{lk} = [\mathbf{B}_k, \mathbf{0}, \mathbf{0}, \cdots, \mathbf{I}_3, [\mathbf{q}_{lk}]_\times, \cdots, \mathbf{0}, \mathbf{0}]$, where $\mathbf{0}$ represents a 3×3 zero matrix. Let $\mathbf{u}_{lk} = \mathbf{p}_{lk} - \mathbf{q}_{lk}$, and the unknown vector $\mathbf{z} = \left[\mathbf{s}^T, \tilde{\mathbf{t}}_1^T, \omega_1^T, \cdots, \tilde{\mathbf{t}}_L^T, \omega_L^T\right]^T$. Following Eq. (16) and (17), we may rewrite the overall likelihood function as:

$$J_{\text{init}1} = \frac{1}{KL}\sum_{l=1}^{L}\sum_{k=1}^{K}(\mathbf{u}_{lk} + \mathbf{H}_{lk}\mathbf{z})^T\mathbf{\Sigma}_{\mathbf{y}_{lk}}^{-1}(\mathbf{u}_{lk} + \mathbf{H}_{lk}\mathbf{z}) \tag{18}$$

$$J_{\text{init}2} = \frac{1}{KL}\sum_{l=1}^{L}\sum_{k=1}^{K}\frac{(\mathbf{u}_{lk} + \mathbf{H}_{lk}\mathbf{z})^T\mathbf{N}_{lk}(\mathbf{u}_{lk} + \mathbf{H}_{lk}\mathbf{z})}{\mathbf{n}_{lk}^T\mathbf{\Sigma}_{\mathbf{y}_{lk}}\mathbf{n}_{lk}}, \tag{19}$$

where $\mathbf{n}_{lk}$ is the surface normal vector for point $\mathbf{p}_{lk}$, $\mathbf{N}_{lk} = \mathbf{n}_{lk}\mathbf{n}_{lk}^T$, and $\mathbf{\Sigma}_{\mathbf{y}_{lk}} \approx \tilde{\mathbf{R}}_{l0}\mathbf{\Sigma}_{\mathbf{x}_{lk}}\tilde{\mathbf{R}}_{l0}^T$. $\mathbf{x}_{lk}$ is the sensor noise for depth input $\mathbf{g}_{lk}$.

The point-to-point and point-to-plane likelihood functions are used jointly in our current implementation. A selected set of point correspondences is used

for J_{init_1} and another selected set is used for J_{init_2} (see Section 5.1 for more details). The overall target function is a linear combination:

$$J_{\text{init}} = \lambda_1 J_{\text{init}_1} + \lambda_2 J_{\text{init}_2}, \tag{20}$$

where λ_1 and λ_2 are the weights between the two functions. The optimization is conducted through quadratic programming.

4.5 Regularization for Tracking

After the static deformation vector $\mathbf{s}$ has been initialized, we track the face frame by frame, estimating the action deformation vector $\mathbf{r}$ and face rotation and translation $\mathbf{R}$ and $\mathbf{t}$, while keeping $\mathbf{s}$ fixed. Although our maximum likelihood solution above can incorporate arbitrary sensor noise covariance matrices, we found the expression tracking results are still very unstable. Therefore, we propose to add additional regularization terms in the target function to further improve the results.

A natural assumption is that the expression change between the current frame and the previous frame is small. In our case, let the previous frame's face action vector be $\mathbf{r}^{t-1}$, we can add an ℓ_2 regularization term as:

$$J_{\text{track}} = \lambda_1 J_1 + \lambda_2 J_2 + \lambda_3 ||\mathbf{r} - \mathbf{r}^{t-1}||_2^2, \tag{21}$$

where J_1 and J_2 follow Eq. (16) and (17). Similar to the initialization process, J_1 and J_2 use different sets of feature points (see Section 5.2 for more details); $||\mathbf{r} - \mathbf{r}^{t-1}||_2^2 = (\mathbf{r} - \mathbf{r}^{t-1})^T(\mathbf{r} - \mathbf{r}^{t-1})$ is the squared ℓ_2 norm of the difference between the two vectors.

The ℓ_2 regularization term works to some extent, though the effect is insignificant. Note as shown in Fig. 2, each dimension of the $\mathbf{r}$ vector represents a particular action a face can perform. Since it is hard for a face to perform all actions simultaneously, we believe in general that the $\mathbf{r}$ vector shall be sparse. This inspires us to impose an additional ℓ_1 regularization term as:

$$J_{\text{track}} = \lambda_1 J_1 + \lambda_2 J_2 + \lambda_3 ||\mathbf{r} - \mathbf{r}^{t-1}||_2^2 + \lambda_4 ||\mathbf{r}||_1, \tag{22}$$

where $||\mathbf{r}||_1 = \sum_{n=1}^{N} |r_n|$ is the ℓ_1 norm. This regularized target function is now in the form of an ℓ_1-regularized least squares problem, which can be reformulated as a convex quadratic program with linear inequality constraints [21], which can again be solved with quadratic programming methods.

Note for PCA-based deformable face models, the ℓ_1 regularization term may not be applied directly. One can identify a few dominant facial expression modes, and still assume sparsity when projecting the PCA coefficients to these modes.

5 Implementation Details

5.1 Deformable Model Initialization

As described in Section 4.4, we use multiple neutral face frames for model initialization, as shown in Fig. 3. Note the likelihood function J_{init} contains both

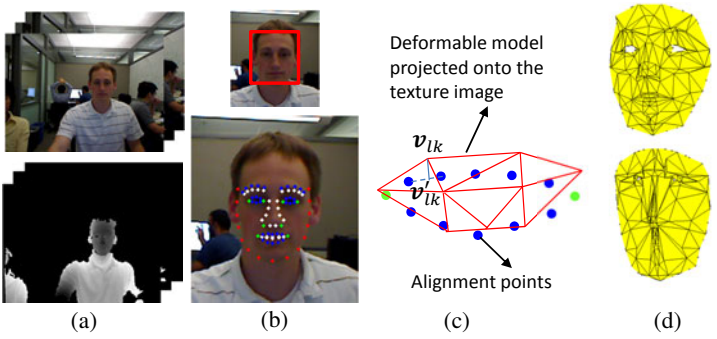


Fig. 3. The process of multi-frame deformable model initialization. (a) Multiple slightly rotated frames with neutral faces as input; (b) face detection (top) and alignment (bottom); (c) define correspondences for edge points around eyebrows, lips etc; (d) DMF with both point-to-point and point-to-plane terms (top) and DMF with point-to-plane term only (bottom).

point-to-point and point-to-plane terms (Eq. (20)). For the point-to-plane term J_{init_2}, the corresponding point pairs are derived by the standard procedure of finding the closest point on the depth map from the vertices on the deformable model [20]. However, the point-to-plane term alone is not sufficient, because our depth images are very noisy and the vertices of the deformable model can drift tangentially, leading to unnatural faces (Fig. 3(d)). In the following we discuss how to define the point-to-point term J_{init_1}.

For each initialization frame, we first perform face detection and alignment on the texture image. The results are shown in Fig. 3(b). The alignment algorithm provides 83 landmark points of the face, which are assumed to be consistent across all the frames. These landmark points are separated into four categories. The first category contains the green points in Fig. 3(b), such as eye corners, mouth corners, etc. These points have clear correspondences $\mathbf{p}_{lk}$ in the linear deformable face model. Given the calibration information between the depth camera and the texture camera, we simply project these landmark points to the depth image to find the corresponding 3D world coordinate $\mathbf{g}_{lk}$.

The second category contains the blue points on eyebrows and upper/lower lips. The deformable face model has a few vertices that define eyebrows and lips, but they do not all correspond to the 2D feature points provided by the alignment algorithm. In order to define correspondences, we use the following steps illustrated in Fig. 3(c):

1. Use the previous iteration's head rotation $\mathbf{R}_0$ and translation $\mathbf{t}_0$ to project the face model vertices $\mathbf{p}_{lk}$ of eyebrows/lips to the texture image, $\mathbf{v}_{lk}$;
2. Find the closest point on the curve defined by the alignment results to $\mathbf{v}_{lk}$, let it be $\mathbf{v}'_{lk}$;
3. Back project $\mathbf{v}'_{lk}$ to the depth image to find its 3D world coordinate $\mathbf{g}_{lk}$.

The third category contains the red points surrounding the face, which we refer as silhouette points. The deformable model also has vertices that define these

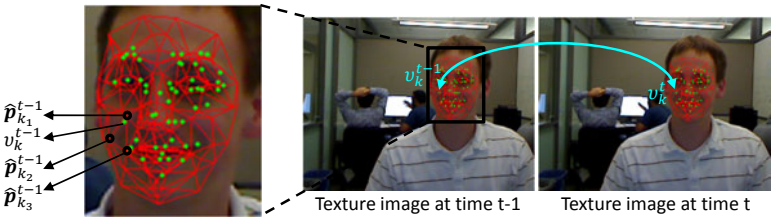


Fig. 4. Track feature points to build correspondences for the point-to-point function

boundary points, but there is no correspondence between them and the alignment results. Moreover, when back projecting the silhouette points to the 3D world coordinate, they may easily hit a background pixel in the depth image. For these points, we follow a similar procedure as the second category points, but ignore the depth axis when computing the distance between $\mathbf{p}_{lk}$ and $\mathbf{g}_{lk}$.

The fourth category includes all the white points in Fig. 3(b), which are not used in the current implementation.

5.2 Tracking

During tracking, we again use both point-to-point and point-to-plane likelihood terms, with additional regularization as in Eq. (22). The point-to-plane term is computed similarly as during model initialization. To reliably track face expressions, the point-to-point term is still crucial. We rely on feature points detected and tracked from the texture images to define these point correspondences, as shown in Fig. 4. Similar schemes have been adopted in deformable surface tracking applications such as [22].

The feature points are detected in the texture image of the previous frame using the Harris corner detector. These points are then tracked to the current frame by matching patches surrounding the points using cross correlation. One issue with such detected and tracked feature pairs is that they may not correspond to any vertices in the deformable face model. Given the previous frame's tracking result, we first represent the feature points with their barycentric coordinates. Namely, as shown in Fig. 4, for 2D feature point pair v_k^{t-1} and v_k^t, we obtain parameter η_1,η_2 and η_3, such that:

$$v_k^{t-1} = \eta_1 \hat{\mathbf{p}}_{k_1}^{t-1} + \eta_2 \hat{\mathbf{p}}_{k_2}^{t-1} + \eta_3 \hat{\mathbf{p}}_{k_3}^{t-1}, \tag{23}$$

where $\eta_1 + \eta_2 + \eta_3 = 1$, and $\hat{\mathbf{p}}_{k_1}^{t-1}$, $\hat{\mathbf{p}}_{k_2}^{t-1}$ and $\hat{\mathbf{p}}_{k_3}^{t-1}$ are the 2D projections of the deformable model vertices $\mathbf{p}_{k_1}$, $\mathbf{p}_{k_2}$ and $\mathbf{p}_{k_3}$ onto the previous frame. Similar to Eq. (2), we can have the following equation:

$$\mathbf{R} \sum_{i=1}^{3} \eta_i \left(\mathbf{p}_{k_i} + \mathbf{A}_{k_i} \mathbf{r} + \mathbf{B}_{k_i} \mathbf{s} \right) + \mathbf{t} = \mathbf{g}_k + \mathbf{x}_k, \tag{24}$$

where $\mathbf{g}_k$ is the back projected 3D word coordinate of 2D feature point v_k^t. Let $\bar{\mathbf{p}}_k = \sum_{i=1}^{3} \eta_i \mathbf{p}_{k_i}$, $\bar{\mathbf{A}}_k = \sum_{i=1}^{3} \eta_i \mathbf{A}_{k_i}$, and $\bar{\mathbf{B}}_k = \sum_{i=1}^{3} \eta_i \mathbf{B}_{k_i}$. Eq. (24) will be in

identical form as Eq. (2), thus tracking is still solved with Eq. (22). Results on the tracking algorithm will be reported in Section 6.

5.3 Noise Modeling

Due to the strong noise in the depth sensor, we find it is generally beneficial to model the actual sensor noise with the correct $\mathbf{\Sigma}_{\mathbf{x}_k}$ instead of using an identity matrix for approximation. The uncertainty of 3D point $\mathbf{g}_k$ has two major sources: the uncertainty in the depth image intensity, which translates to uncertainty along the depth axis, and the uncertainty in feature point detection/matching in the texture image, which translates to uncertainty along the imaging plane.

Assuming a pinhole, no-skew projection model for the depth camera, we have:

$$z_k \begin{bmatrix} u_k \\ v_k \\ 1 \end{bmatrix} = \mathbf{K}\mathbf{g}_k = \begin{bmatrix} f_x & 0 & u_0 \\ 0 & f_y & v_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_k \\ y_k \\ z_k \end{bmatrix} \tag{25}$$

where $\mathbf{v}_k = [u_k, v_k]^T$ is the 2D image coordinate of the feature point k in the depth image, and $\mathbf{g}_k = [x_k, y_k, z_k]^T$ is the 3D world coordinate of the feature point. $\mathbf{K}$ is the intrinsic matrix, where f_x and f_y are the focal lengths, and u_0 and v_0 are the center biases.

For the depth camera, the uncertainty of u_k and v_k is generally caused by feature point uncertainties in the texture image, and the uncertainty in z_k is due to the depth derivation scheme. These two uncertainties can be considered as independent to each other. Let $\mathbf{c}_k = [u_k, v_k, z_k]^T$, we have:

$$\mathbf{\Sigma}_{\mathbf{c}_k} = \begin{bmatrix} \mathbf{\Sigma}_{\mathbf{v}_k} & \mathbf{0} \\ \mathbf{0}^T & \sigma_{z_k}^2 \end{bmatrix}. \tag{26}$$

It is easy to find that:

$$\mathbf{G}_k \triangleq \frac{\partial \mathbf{g}_k}{\partial \mathbf{c}_k} = \begin{bmatrix} \frac{z_k}{f_x} & 0 & \frac{u_k - u_0}{f_x} \\ 0 & \frac{z_k}{f_y} & \frac{v_k - v_0}{f_y} \\ 0 & 0 & 1 \end{bmatrix}. \tag{27}$$

Hence as an approximation, the sensor's noise covariance matrix shall be:

$$\mathbf{\Sigma}_{\mathbf{x}_k} \approx \mathbf{G}_k \mathbf{\Sigma}_{\mathbf{c}_k} \mathbf{G}_k^T. \tag{28}$$

In our current implementation, to compute $\mathbf{\Sigma}_{\mathbf{c}_k}$ from Eq. (26), we assume $\mathbf{\Sigma}_{\mathbf{v}_k}$ is diagonal, i.e., $\mathbf{\Sigma}_{\mathbf{v}_k} = \sigma^2 \mathbf{I}_2$, where $\mathbf{I}_2$ is a 2×2 identity matrix, and $\sigma = 1.0$ pixels in the current implementation. Knowing that our depth sensor derives depth based on triangulation, following [23], the depth image noise covariance $\sigma_{z_k}^2$ is modeled as:

$$\sigma_{z_k}^2 = \frac{\sigma_0^2 z_k^4}{f_d^2 B^2}, \tag{29}$$

where $f_d = \frac{f_x + f_y}{2}$ is the depth camera's average focal length, $\sigma_0 = 0.059$ pixels and $B = 52.3875$ millimeters based on calibration. Note since σ_{z_k} depends on z_k, its value depends on each pixel's depth value and cannot be pre-determined.

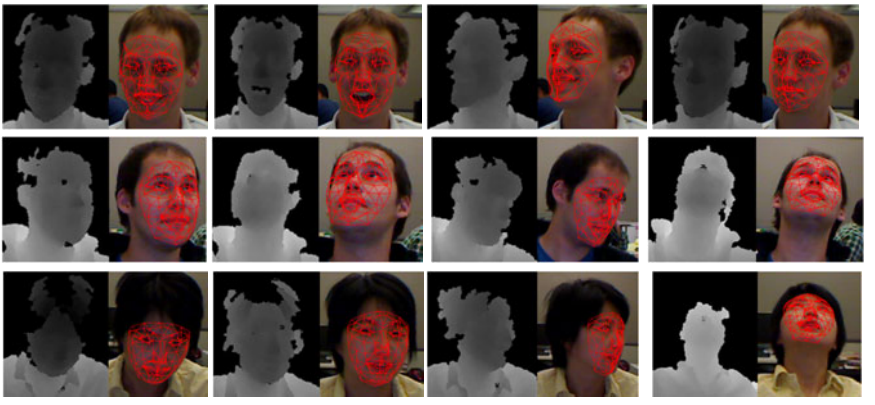


Fig. 5. Example tracking results using the proposed algorithm. From top to bottom are sequence #1 (810 total frames), #2 (681 total frames) and #3 (300 total frames), respectively.

6 Experimental Results

We tested the proposed algorithm with three sequences captured by our depth camera. Both the color and the depth images are at 640×480 pixels resolution and 30 fps. In each sequence the user sat about 3 ft from the depth camera, and moved around with varying expressions. The head sizes in the images are about 100×100 pixels. Throughout the experiments, we set the weights of different terms in J_{init} and J_{track} to be $\lambda_1 = \lambda_2 = 1$, $\lambda_3 = 10^{-6}$ and $\lambda_4 = 10$. All sequences are initialized fully automatically and accurately with the multi-frame DMF algorithm presented in Section 4.4 and 5.1. Initialization from 10 input frames takes about 20 iterations and 6.7 seconds on an Intel 2.66 GHz computer, while tracking usually converges in 2 iterations and can be done at about 10-12 fps without much code optimization.

We first show a few example tracking results using the proposed algorithm in Fig. 5, which demonstrate the robustness of the proposed algorithm despite large face pose and expression variations.

To provide some quantitative results, we manually labeled 12 feature points around the eye and mouth regions of each face in every 3-5 frames of the three sequences, as shown in Fig. 6(a). We then computed the average Euclidian distance from the 2D projections of their corresponding deformable model vertices to the labeled positions. We compared various combinations of algorithms with and without noise modeling, with and without the ℓ_2 regularization, and with and without the ℓ_1 regularization. The results are summarized in Table 1. Note because some combinations could not track the whole sequence successfully, we reported the *median* average error of all the labeled frames in Table 1. It can be seen that all three components improved the tracking performance. More specifically, compared with the traditional scheme that adopts an identity covariance matrix for sensor noises and ℓ_2 regularization (ID+ℓ_2), the proposed scheme (NM+ℓ_2+ℓ_1) reduced the median average error by 25.3% for sequence

Table 1. Comparison of median tracking error (in pixels) for various algorithms. The suffix "L" indicates that the tracking algorithm lost the face and never recovered. "ID" stands for using the identity covariance matrix for sensor noises, and "NM" stands for using the proposed noise modeling scheme.

	ID+ℓ_2	ID+ℓ_1	ID+ℓ_2+ℓ_1	NM+ℓ_2	NM+ℓ_1	NM+ℓ_2+ℓ_1
Seq#1 (164 labeled frames)	3.56	2.88	2.78	2.85	2.69	2.66
Seq#2 (164 labeled frames)	4.48	3.78	3.71	4.30	3.64	3.55
Seq#3 (74 labeled frames)	3.98L	3.91	3.91	3.92L	3.91	3.50

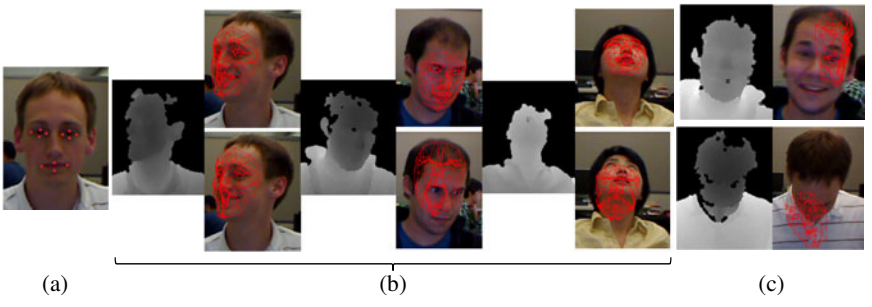


Fig. 6. (a) Face labeled with 12 ground truth feature points; (b)a few successfully tracked frames with NM+ℓ_2+ℓ_1 (top) which were failed using the traditional approach ID+ℓ_2 (bottom); (c) two failure examples for the proposed algorithm.

#1 and by 20.8% for sequence #2. The traditional ID+ℓ_2 scheme lost tracking for sequence #3 after about 100 frames, while the proposed scheme successfully tracked the whole sequence.

Fig. 6(b) shows a few examples where the proposed algorithm tracked the face successfully, while the traditional scheme failed. Nonetheless, our algorithm may also fail, as shown in Fig. 6(c). In the top frame, the head moved very fast and the color image was blurry. In addition, the proposed algorithm is an iterative scheme, and fast motion can cause poor initialization of the estimated parameters. In the bottom frame, the face turned downward, which caused problems in tracking facial features in the color image. Currently we have not built any recovery mechanism in the system such as adding key frames or occasional re-initialization, which will be part of our future work.

7 Conclusions and Future Work

In this paper, we presented a regularized maximum likelihood DMF algorithm that can be used to track faces with noisy input depth data from commodity depth cameras. The algorithm modeled the depth sensor noise with an arbitrary covariance matrix, and applied a new ℓ_1 regularization term that is semantically meaningful and effective. In future work we plan to work on 3D face alignment that can re-initialize the tracking process at arbitrary face poses, thus further improving the performance of the overall system.

References

1. Xiao, J., Baker, S., Matthews, I., Kanade, T.: Real-time combined 2d+3d active appearance models. In: CVPR (2004)
2. Blanz, V., Vetter, T.: A morphable model for the synthesis of 3D faces. In: SIGGRAPH (1999)
3. Vogler, C., Li, Z., Kanaujia, A., Goldenstein, S., Metaxas, D.: The best of bothworlds: Combining 3d deformable models with active shape models. In: ICCV (2007)
4. Zhang, W., Wang, Q., Tang, X.: Real time feature based 3-D deformable face tracking. In: Forsyth, D., Torr, P., Zisserman, A. (eds.) ECCV 2008, Part II. LNCS, vol. 5303, pp. 720–732. Springer, Heidelberg (2008)
5. Blanz, V., Vetter, T.: Face recognition based on fitting a 3d morphable model. IEEE Trans. on PAMI (2003)
6. Zhu, Y., Fujimura, K.: 3d head pose estimation with optical flow and depth constraints. In: 3DIM (2003)
7. Zhang, L., Snavely, N., Curless, B., Seitz, S.M.: Spacetime faces: high-resolution capture for modeling and animation. In: SIGGRAPH 2004 (2004)
8. Wang, Y., Huang, X., Lee, C.S., Zhang, S., Li, Z., Samaras, D., Metaxas, D., Elgammal, A., Huang, P.: High resolution acquisition, learning and transfer of dynamic 3-D facial expressions. In: EUROGRAPHICS 2004 (2004)
9. Weise, T., Li, H., Gool, L.V., Pauly, M.: Face/off: Live facial puppetry. In: Symposium on Computer Animation (2009)
10. Besl, P.J., McKay, N.D.: A method for registration of 3-D shapes. IEEE Trans. on PAMI 14, 239–256 (1992)
11. Amberg, B., Romdhani, S., Vetter, T.: Optimal step nonrigid ICP algorithms for surface registration. In: CVPR (2007)
12. Murphy-Chutorian, E., Trivedi, M.M.: Head pose estimation in computer vision: A survey. IEEE Trans. on PAMI (2009)
13. Cohen, M.M., Massaro, D.W.: Modeling coarticulation in synthetic visual speech. In: Models and Techniques in Computer Animation (1993)
14. Sifakis, E., Selle, A., Robinson-Mosher, A., Fedkiw, R.: Simulating speech with a physics-based facial muscle model. In: Proc. of SCA 2006 (2006)
15. Munoz, E., Buenaposada, J.M., Baumela, L.: A direct approach for efficiently tracking with 3D morphable models. In: ICCV (2009)
16. Zhang, Z., Faugeras, O.D.: Determining motion from 3d line segment matches: a comparative study. Image and Vision Computing 9, 10–19 (1991)
17. Lu, X., Jain, A.K.: Deformation modeling for robust 3D face matching. IEEE Trans. on PAMI 30, 1346–1357 (2008)
18. Bowyer, K.W., Chang, K., Flynn, P.: A survey of approaches and challenges in 3D and multi-modal 3D+2D face recognition. CVIU (2006)
19. Zhang, Z., Liu, Z., Adler, D., Cohen, M.F., Hanson, E., Shan, Y.: Robust and rapid generation of animated faces from video images: A model-based modeling approach. IJCV 58, 93–119 (2004)
20. Chen, Y., Medioni, G.: Object modelling by registration of multiple range images. Image and Vision Computing 10, 145–155 (1992)
21. Boyd, S., Vandenberghe, L.: Convex Optimization. Cambridge Univ. Press, Cambridge (2004)
22. Salzmann, M., Pilet, J., Ilic, S., Fua, P.: Surface deformation models for nonrigid 3d shape recovery. IEEE Trans. on PAMI 29, 1481–1487 (2007)
23. Gallup, D., Frahm, J.M., Mordohai, P., Pollefeys, M.: Variable baseline/resolution stereo. In: CVPR (2008)