TRACKING OF FACIAL FEATURE POINTS BY COMBINING SINGULAR TRACKING RESULTS WITH A 3D ACTIVE SHAPE MODEL

Moritz Kaiser, Dejan Arsić, Shamik Sural and Gerhard Rigoll
Technische Universität München, Arcisstr. 21, 80333 Munich, Germany

Keywords: Facial feature tracking, 3D Active Shape Model, Face pose estimation.

Abstract: Accurate 3D tracking of facial feature points from one monocular video sequence is appealing for many applications in human-machine interaction. In this work facial feature points are tracked with a Kanade-Lucas-Tomasi (KLT) feature tracker and the tracking results are linked with a 3D Active Shape Model (ASM). Thus, the efficient Gauss-Newton method is not solving for the shift of each facial feature point separately but for the 3D position, rotation and the 3D ASM parameters which are the same for all feature points. Thereby, not only the facial feature points are tracked more robustly but also the 3D position and the 3D ASM parameters can be extracted. The Jacobian matrix for the Gauss-Newton optimization is split via chain rule and the computations per frame are further reduced. The algorithm is evaluated on the basis of three handlabeled video sequences and it outperforms the KLT feature tracker. The results are also comparable to two other tracking algorithms presented recently, whereas the method proposed in this work is computationally less intensive.

1 INTRODUCTION

Accurate 3D tracking of facial feature points from one monocular video sequence is a challenging task, since only 2D information is available from the video. The 3D tracking is appealing for many applications in human-machine interaction. In contrast to 2D tracking, information about the 3D head position and the 3D head movement can be extracted and it is also possible to perform pose independent emotion and expression recognition (Gong et al., 2009). Statistical models are a powerful tool for solving this task. Often, models that are based on the entire appearance of faces are employed (Cootes et al., 1998; Faggian et al., 2008). Nevertheless, for many applications it is sufficient to track only a certain number of selected facial feature points. The advantage of only tracking a sparse set of points is that it is less complicated and computationally less intensive.

1.1 Conception of this Work

For model-free 2D tracking the Kanade-Lucas-Tomasi (KLT) (Tomasi and Kanade, 1991) feature tracker generates reasonable results and it is computationally efficient. The KLT feature tracker is a local approach that computes the shift of each facial feature point separately. Often due to noise, illumination changes or weakly textured neighborhood the KLT feature tracker loses track of some feature points. An unrealistic constellation of points is the result as depicted in Figure 1. In this work we combine the tracking results for each single point with a 3D Active Shape Model (ASM). The 3D ASM guarantees a certain structure between the locally tracked points. Hence, the efficient Gauss-Newton optimization (Bertsekas, 1999) is not solving for the 2D shift of several facial feature points separately but for the parameters of a 3D ASM and rotation, translation and scale parameters which are the same for all feature points. Thereby, not only the facial feature points are tracked more robustly but also additional information like the parameters of the 3D ASM for emotion and expression recognition and the 3D head position and movement are extracted.

1.2 Previous Work

Model-free tracking algorithms are not suitable for extracting 3D information from a monocular video sequence. Thus, model-based methods are applied. A popular approach for tracking is to use statistical...
models based on the entire appearance of the face. The method described in (Cristinacce and Cootes, 2006a) uses texture templates to locate and track facial features. In (Cristinacce and Cootes, 2006b) texture templates are used to build up a constrained local appearance model. The authors of (Sung et al., 2008) propose a face tracker that combines an Active Appearance Model (AAM) with a cylindrical head model to be able to cope with cylindrical rotation. In (Fang et al., 2008) an optical flow method is combined with a statistical model, as in our approach. However, the optical flow algorithm by (Brox et al., 2004) is applied which is computationally more intensive than a Gauss-Newton optimization for only several facial feature points. All those methods rely on statistical models that are based on the entire texture of the face which is computationally more intensive than statistical models based on only a sparse set of facial feature points.

The authors of (Heimann et al., 2005) employed a statistical model which is based only on a few feature points, namely a 3D Active Shape Model, which is directly matched to 3D point clouds of organs for detection. The matching takes from several minutes up to several hours. (Tong et al., 2007) use a set of facial feature points whose spatial relations are modeled with a 2D hierarchical shape model. Multi-state local shape models are used to model small movements in the face.

This paper is organized as follows. In Section 2 our algorithm is presented, where first a tracking framework for a generic parametric model is explained and then the 3D parametric model we employ is described. Quantitative and qualitative results are given in Section 3. Section 4 gives a conclusion and outlines future work.

2 PROPOSED ALGORITHM

In this section, a tracking framework is illustrated, that can be applied if the motion can be described by a parametric model. The least-squares estimation of the parameters of a generic parametric model is explained. Subsequently, the parametric model that is employed in this work, namely a 3D ASM, is presented. It is explained how the parameter estimation, that has to be performed for each frame, can be carried out more efficiently. Furthermore, some constraints on the parameter estimation are added.

Following the ISO typesetting standards, matrices and vectors are denoted by bold letters ($\mathbf{I}, \mathbf{x}$) and scalars by normal letters ($I, x$).

2.1 Tracking Framework for Parametric Models

Assume that the first frame of a video sequence is acquired at time $t_0 = 0$. Vector $x_i = (x, y)^T$ describes the location of point $i$ in a frame. The location changes over time according to a parametric model $x_i(\mu_t)$, where $\mu_t$ is the parameter vector at time $t$. The set of points for which the parameter vector is supposed to be the same is denoted by $\mathcal{I} = \{x_1, x_2, \ldots, x_N\}$. It is assumed that the position of the $N$ points and thus also the parameters $\mu_0$ for the first frame are known. The brightness value of point $x_i(\mu_0)$ in the first frame is denoted by $I(x_i(\mu_0), t_0) = 0$.

The brightness constancy assumption implies that at a later time the brightness of the point to track is the same

$$ I(x_i(\mu_t), 0) = I(x_i(\mu_t), t). \quad (1) $$

For better readability we write $I_i(\mu_t, 0) = I(\mu_t, t)$. The energy function that has to be minimized at every time step $t$ for each location $i$ in order to obtain $\mu_t$ is

$$ E_i(\mu_t) = \left( I_i(\mu_t, t) - I_i(\mu_0, 0) \right)^2. \quad (2) $$

Knowing $\mu_t$ at time $t$, we only need to compute $\Delta \mu$ in order to determine $\mu_{t+\tau} = \mu_t + \Delta \mu$. The energy function becomes

$$ E_i(\Delta \mu) = \left( I_i(\mu_t + \Delta \mu, t + \tau) - I_i(\mu_0, 0) \right)^2. \quad (3) $$

If $\tau$ is sufficiently small, $I_i(\mu_t + \Delta \mu, t + \tau)$ can be linearized with a Taylor series expansion considering only the first order term and ignoring the second and higher order terms

$$ I_i(\mu_t + \Delta \mu, t + \tau) \approx I_i(\mu_t, t) + \frac{\partial I_i}{\partial \mu} I_i(\mu_t, t) \cdot \Delta \mu + \tau \frac{\partial^2 I_i}{\partial \mu^2} I_i(\mu_t, t), \quad (4) $$

with $\frac{\partial I_i}{\partial \mu} = (\frac{\partial I_i}{\partial \mu_1}, \frac{\partial I_i}{\partial \mu_2}, \ldots, \frac{\partial I_i}{\partial \mu_N})$. By approximating

$$ \tau \frac{\partial I_i}{\partial t} \approx I_i(\mu_t, t + \tau) - I_i(\mu_t, t), \quad (5) $$

Equation (3) becomes

$$ E_i(\Delta \mu) = \left( \frac{\partial I_i}{\partial \mu} I_i(\mu_t, t) \cdot \Delta \mu + \tau \frac{\partial^2 I_i}{\partial \mu^2} I_i(\mu_t, t), \Delta \mu + \tau \frac{\partial^2 I_i}{\partial \mu^2} I_i(\mu_t, t) \right)^2. \quad (6) $$

The error is defined as

$$ e_i(t + \tau) = I_i(\mu_t, t + \tau) - I_i(\mu_0, 0). \quad (7) $$

If we assume that $E_i$ is convex, the minimization problem can be solved by

$$ \frac{\partial}{\partial \Delta \mu} E_i = \left( \frac{\partial}{\partial \mu} I_i(\mu_t, t) \cdot \Delta \mu + e_i(t + \tau) \right) \cdot \frac{\partial}{\partial \mu} I_i(\mu_t, t) = 0. \quad (8) $$
Taking into account all points \( i \), the system of linear equations can be solved for \( \Delta \mu \) by
\[
\Delta \mu = -(J^T J)^{-1} J^T e(t + \tau),
\]
where
\[
e(t + \tau) = \begin{pmatrix} e_1(t + \tau) \\ \vdots \\ e_N(t + \tau) \end{pmatrix},
\]
\( J \) denotes the \( N \times N_p \) Jacobian matrix of the image with respect to \( \mu \):
\[
J = \begin{pmatrix} \frac{\partial I_1}{\partial \mu}(\mu_t, t) \\ \vdots \\ \frac{\partial I_N}{\partial \mu}(\mu_t, t) \end{pmatrix},
\]
where \( N \) is the number of points with equal parameters and \( N_p \) is the number of parameters.

For the KLT feature tracker there are only two parameters to estimate, namely the shift in \( x \)- and \( y \)-direction. \( \frac{\partial}{\partial \mu} I_i(\mu_t, t) \) can be estimated numerically very efficiently with e.g. a Sobel filter. The set \( T \) for which equal parameters are assumed is a squared neighborhood around the feature point that should be tracked. For the case of tracking several facial feature points, each facial feature point would have to be tracked separately. Although the KLT feature tracker works already quite well, single points to track might drift off due to a weakly textured neighborhood, noise, illumination changes or occlusion. Figure 1 (left) shows the first frame of a video sequence with 22 feature points and Figure 1 (right) illustrates the tracked points several frames later. Most of the feature points are tracked correctly but one (marked with a white cross) got lost by the feature tracker. The human viewer can immediately see that the spacial arrangement of the facial feature points (white and black crosses) is not typical for a face. The structure between the facial feature points can be ensured by a parametric model. In the case of landmarks on a face a 3D ASM seems appropriate.

\[ s = (x_1, y_1, z_1, \ldots, x_N, y_N, z_N). \]

For the KLT feature tracker there are only two parameters to estimate, namely the shift in \( x \)- and \( y \)-direction. \( \frac{\partial}{\partial \mu} I_i(\mu_t, t) \) can be estimated numerically very efficiently with e.g. a Sobel filter. The set \( T \) for which equal parameters are assumed is a squared neighborhood around the feature point that should be tracked. For the case of tracking several facial feature points, each facial feature point would have to be tracked separately. Although the KLT feature tracker works already quite well, single points to track might drift off due to a weakly textured neighborhood, noise, illumination changes or occlusion. Figure 1 (left) shows the first frame of a video sequence with 22 feature points and Figure 1 (right) illustrates the tracked points several frames later. Most of the feature points are tracked correctly but one (marked with a white cross) got lost by the feature tracker. The human viewer can immediately see that the spacial arrangement of the facial feature points (white and black crosses) is not typical for a face. The structure between the facial feature points can be ensured by a parametric model. In the case of landmarks on a face a 3D ASM seems appropriate.

\[ s = (x_1, y_1, z_1, \ldots, x_N, y_N, z_N). \]

For the KLT feature tracker there are only two parameters to estimate, namely the shift in \( x \)- and \( y \)-direction. \( \frac{\partial}{\partial \mu} I_i(\mu_t, t) \) can be estimated numerically very efficiently with e.g. a Sobel filter. The set \( T \) for which equal parameters are assumed is a squared neighborhood around the feature point that should be tracked. For the case of tracking several facial feature points, each facial feature point would have to be tracked separately. Although the KLT feature tracker works already quite well, single points to track might drift off due to a weakly textured neighborhood, noise, illumination changes or occlusion. Figure 1 (left) shows the first frame of a video sequence with 22 feature points and Figure 1 (right) illustrates the tracked points several frames later. Most of the feature points are tracked correctly but one (marked with a white cross) got lost by the feature tracker. The human viewer can immediately see that the spacial arrangement of the facial feature points (white and black crosses) is not typical for a face. The structure between the facial feature points can be ensured by a parametric model. In the case of landmarks on a face a 3D ASM seems appropriate.

\[ s = (x_1, y_1, z_1, \ldots, x_N, y_N, z_N). \]

For the KLT feature tracker there are only two parameters to estimate, namely the shift in \( x \)- and \( y \)-direction. \( \frac{\partial}{\partial \mu} I_i(\mu_t, t) \) can be estimated numerically very efficiently with e.g. a Sobel filter. The set \( T \) for which equal parameters are assumed is a squared neighborhood around the feature point that should be tracked. For the case of tracking several facial feature points, each facial feature point would have to be tracked separately. Although the KLT feature tracker works already quite well, single points to track might drift off due to a weakly textured neighborhood, noise, illumination changes or occlusion. Figure 1 (left) shows the first frame of a video sequence with 22 feature points and Figure 1 (right) illustrates the tracked points several frames later. Most of the feature points are tracked correctly but one (marked with a white cross) got lost by the feature tracker. The human viewer can immediately see that the spacial arrangement of the facial feature points (white and black crosses) is not typical for a face. The structure between the facial feature points can be ensured by a parametric model. In the case of landmarks on a face a 3D ASM seems appropriate.

\[ s = (x_1, y_1, z_1, \ldots, x_N, y_N, z_N). \]
where \( \mu = (x, \theta_x, \theta_z, t_x, t_z, \alpha) \) is the \((5 + N_e)\)-dimensional parameter vector and
\[
R(\theta_x, \theta_z) = \begin{pmatrix} \cos \theta_x \cos \theta_z & \cos \theta_x \sin \theta_z & -\sin \theta_x \\ -\sin \theta_x & \cos \theta_x & 0 \end{pmatrix}.
\] (16)

We directly insert the 3D ASM of Equation 15 into our tracking framework for parametric models of Equation 9 in order to estimate the parameters \( \mu \) in a least-squares sense for each frame.

### 2.3 Efficient Parameter Estimation

Since \( J \) of Equation 9 depends on time-dependent quantities, it must be recomputed for each frame. Especially for the model of Equation 15 the numerical computation of \( \frac{\partial}{\partial \mu} I(\mu, t) \) at each time step is time consuming. Therefore, the derivative of the image with respect to the parameters is decomposed via chain rule into an easily computable spatial derivative of the image and a derivative of the parametric model with respect to the parameters which can be solved analytically. Additionally, if, as in Equation 1, it is assumed that \( \frac{\partial}{\partial \mu} I(\mu, t) = \frac{\partial}{\partial \mu} I_0(\mu_0, 0) \), we obtain
\[
J = \begin{pmatrix} \frac{\partial}{\partial \mu} I_1(\mu_0, 0) \frac{\partial}{\partial \mu} x_1(\mu) \\ \vdots \\ \frac{\partial}{\partial \mu} I_N(\mu_0, 0) \frac{\partial}{\partial \mu} x_N(\mu) \end{pmatrix}.
\] (17)

The numerical derivative \( \frac{\partial}{\partial \mu} I(\mu_0, 0) \) and the analytical derivative \( \frac{\partial}{\partial \mu} x(\mu) \) can be computed offline. Thus, for each frame the current estimation of \( \mu \) has to be plugged in the Jacobian matrix and the new parameters are computed according to Equation 9.

### 2.4 Constraints on the 3D ASM

In order to prevent unrealistic results several constraints on the parameters are added. We require the rotation \( \theta_x, \theta_z \) and also the 3D ASM parameters \( \alpha \) not to become too large:
\[
\frac{\Delta \theta_x}{\sigma_{\theta_x}} = -\frac{\Delta \theta_z}{\sigma_{\theta_z}}; \quad \frac{\Delta \theta_z}{\sigma_{\theta_z}} = -\frac{\Delta \theta_x}{\sigma_{\theta_x}} \quad (18)
\]
\[
\frac{\Delta \alpha_j}{\sigma_{\alpha_j}} = \frac{\Delta \alpha_j}{\sigma_{\alpha_j}} \quad (19)
\]
We also require all parameters not to change too much from one frame to another
\[
\frac{\Delta \mu_j}{\sigma_{\Delta \mu_j}} = 0. \quad (20)
\]

The constraints can easily be appended at the bottom of \( J \) as further equations that the parameters \( \Delta \mu \) have to satisfy. In order to have the right balance between all equations, the equations of (9) are multiplied by \( 1/\sigma_{\mu_j}^2 \), where \( \sigma_{\mu_j}^2 \) is the variance of the error \( e_j \). Note that \( \sigma_{\Delta \mu_j}^2 \) can be directly taken from the principal component analysis performed for the 3D ASM, while the other variances have to be estimated.

### 2.5 Coarse-To-Fine Refinement

The coarse-to-fine refinement is a widely-used strategy to deal with larger displacements for optical flow and correspondence estimation. In practice, \( E_t \) is often not convex as we have assumed in Equation 8. Especially, if there is a large displacement of the facial feature points between two consecutive frames, the least-squares minimization converges to only a local minimum instead of the global minimum. This problem can be - at least partially - overcome by applying a Gaussian image pyramid that has to be created for each new frame. The computation of the parameter vector is performed for the coarsest level. Then, the parameters are taken as starting values for the next finer level for which the computation is performed again, and so on.

### 3 EXPERIMENTAL RESULTS

We built the 3D ASM from the Bosphorus Database (Savran et al., 2008) where we used 2761 3D images from 105 individuals. The images are labeled with \( N = 22 \) facial feature points (Figure 2). Four pyramid levels were employed for the Gaussian image pyramid. On a 3GHz Intel® Pentium® Duo-Core processor and 3GB working memory the computation of the parameters took on average 9.83 ms per frame.

#### 3.1 Quantitative Evaluation

The proposed algorithm was evaluated on the basis of three video sequences, each lasting roughly one minute. In each video sequence an individual performs motions, such as translation, rotation about the \( y \)- and \( z \)-axis and it also changes the facial expression. The image resolution of the video sequence is \( 640 \times 480 \) pixels. Every 10th frame was manually annotated with 22 landmarks and those landmarks were used as ground truth. For each labeled frame the pixel displacement \( d_i = \sqrt{\Delta x^2 + \Delta y^2} \) between the location estimated by our algorithm and the ground truth was computed. The pixel displacement was averaged over...
Figure 3: Pixel displacement for three test video sequences.

Figure 3 shows the results for the three video sequences. As a baseline system the KLT feature tracker was chosen. Observe that already for the first frame there is a small pixel displacement indicating that a portion of the displacement is owed to the fact that the annotation cannot always be performed unequivocally. For higher frame numbers the KLT feature tracker loses track of some points and thus the pixel displacement accumulates. Our algorithm is able to prevent this effect and shows robust performance over the whole sequence.

### 3.2 Qualitative Evaluation

Figure 4 shows two sample frames of video sequence no. 1. (The 3 video sequences are available together in a single file as supplementary material.) The information box on the lower left corner shows position details. For the left image it can be observed that the rotation about the z-axis is detected correctly and in the right image the rotation about the y-axis is estimated properly. Generally, it can be qualitatively confirmed that not only the points are tracked reliably in the 2D video sequence but also 3D motion and expressions can be extracted from the sequence.

It is also important to notice that our 3D ASM works with a relatively small number of facial feature points, since the 3D faces of the Bosphorus Database are labeled with only 22 landmarks. In contrast, current 2D face databases have more landmarks, some of them roughly hundred points. It is expected that a 3D ASM with more points would further improve the tracking and parameter estimation results.

### 4 CONCLUSIONS AND FUTURE WORK

A method for 3D tracking of facial feature points from a monocular video sequence is presented. The facial feature points are tracked with a simple Gauss-Newton estimation scheme and the results are linked with a 3D ASM. Thus, the efficient Gauss-Newton minimization computes the 3D position, rotation and 3D ASM parameters instead of the shift of each feature point separately. It is demonstrated how the amount of computations that must be performed for each frame can be further reduced. Results show that the algorithm tracks the points reliably for rotation, translations, and facial expressions. It outperforms the KLT feature tracker and delivers results comparable to two other methods published recently, while being computationally less intensive.

In our ongoing research we will analyze the effect of using gradient images and Gabor filtered images to further improve the tracking results. We have also planned to integrate a weighting matrix that depends on the rotation parameters to reduce the influence of facial feature points that might disappear.
REFERENCES


