NOSE TIP DETECTION AND TRACKING IN 3D VIDEO SEQUENCES

Xiaoming Peng [1, 2] and Mohammed Bennamoun [2]
[1] *School of Automation Engineering, University of Electronic Science and Technology of China*
No. 4, Section 2, North Jianshe Road, Chengdu, Sichuan 610054, China
[2] *School of Computer Science and Software Engineering, The University of Western Australia*
M002, 35 Stirling Highway, Crawley, WA 6009, Australia

Keywords: Facial point cloud data, Geometric computing, 3D video.

Abstract: Point cloud data processing is an important topic in geometric computing. One promising application of point cloud data processing is 3D face recognition. With the recent developments of 3D scanning technology, the emergence in the near future of 3D face recognition from 3D video sequences is eminent. Face tracking is a necessary step before the recognition of a face. In this paper, we propose the integration of a nose tip detection method into the process of tracking the face in a 3D video sequence. The nose tip detection method which does not require training nor does it rely on any particular model, can deal with both frontal and non-frontal poses, and is quite fast. Combined with the Iterative Closest Point (ICP) algorithm and a Kalman filter, the nose-tip-detection-based method achieved robust tracking results on real 3D video sequences. We have also shown that it can be used to coarsely estimate the roll, yaw and pitch angles of the face poses.

1 INTRODUCTION

Face recognition is an active research area in biometrics with much of the work being performed in the 2D domain. With the rapid development of advanced range imaging devices, 3D face recognition which uses facial point cloud data has lately received a growing attention from researchers and industries. Compared with 2D face recognition, 3D face recognition is less sensitive to illumination, pose variations, facial expressions, and makeup (Mian et al., 2007). Currently, most 3D face recognition research is performed on still range images (Queirolo et al., 2010; Pears et al., 2009; Chang et al., 2006; Al-Osaimi et al., 2009; Lu and Jain, 2008). Although these methods work very well on neutral-expression and frontal faces, great challenges are met when they deal with non-neutral-expression and/or non-frontal faces (Queirolo et al., 2010; Al-Osaimi et al., 2009; Lu and Jain, 2008). One possible strategy for confronting such challenges is to utilize temporal information. As has been observed in 2D video-based face recognition, since successive frames in a video sequence are continuous in the temporal dimension, "such continuity, coming from facial expression, geometric continuity related to head and/or camera movement, or photometric continuity related to changes in illumination, provides an additional constraint for modeling face appearance" (Zhou et al., 2009). Although existing 3D video scanners, e.g., SwissRanger 3000/4000, are not comparable with 2D video cameras in terms of image resolution and capture range, we can envision that with the development of 3D scanning technology, high-quality 3D video scanners will emerge in the near future and can be used for 3D face recognition.

In order to recognize a face in a video sequence, one needs to first locate the face in each frame of the video sequence, which leads to the face tracking problem. In this paper, we propose a nose-tip-detection-based face tracking method which tracks the face by tracking the nose tip in a 3D video sequence. We argue that compared with other facial features, such as inner or outer eye corners, the nose tip is more robust to pose changes; it is also more invariant to expression changes compared with features on the cheeks or on the chin. The nose tip detection method adopted in this paper, combined with the Iterative Closest Point (ICP) algorithm and

Peng X. and Bennamoun M..
NOSE TIP DETECTION AND TRACKING IN 3D VIDEO SEQUENCES.
DOI: 10.5220/0003314900130022
In *Proceedings of the International Conference on Computer Graphics Theory and Applications* (GRAPP-2011), pages 13-22
ISBN: 978-989-8425-45-4

a Kalman filter, helps to build a robust face tracking method in that although the nose tip detection method may sometimes fail, tracking can still go ahead with the help of ICP.

The rest of the paper is organized as follows. In Section 2, related work is reviewed. In Section 3, we describe the proposed method in detail. Section 4 presents the experimental results of the proposed method applied on real range image sequences. Finally, in Section 5 we provide the conclusions and discussions.

2 RELATED WORK

In 3D face recognition the nose region plays a very important role particularly in face normalization, pose correction, and nose region based matching. More often, nose localization is achieved by successfully detecting distinctive facial features (the nose tip, nose ridge, eye corners, etc.) in face scans.

Most existing methods are 3D-based. They rely on the direct detection of facial features from range images. Some researchers (Malassiotis et al., 2005; Colombo et al., 2006) use curvature analysis, including the mean (H) and Gaussian (K) curvatures classification (HK Classification) and the principal curvatures to help confine the search scope of particular facial features. The computation of the H and K curvatures and the principal curvatures involves estimating the second order derivatives of a range image, which is error-prone because of the noise in the image. In addition, these methods are not pose invariant.

Xu et al. first use the "effective energy" defined in their paper to filter out points in non-convex areas (Xu et al., 2006). Because the effective energy condition is very weak and many points in other areas like the cheeks and chin also meet this condition. A support vector machine (SVM) is then trained using the means and variances of the effective energy sets as input to further filter out the non-nose tip points. However, the effective energy set is not pose invariant, meaning that even for a same point, the means and variances of the various effective energy sets corresponding to various poses could be different. Chew et al. also use the effective energy condition to select nose tip candidates, but circumvent the SVM-training stage by trying to find the mouth-and-eyes regions in the input image, which is not always easy. The nose tip detection rates were moderate as reported in their paper—93% and 68%—for frontal and non-frontal faces, respectively (Chew et al., 2009). In Pears et al's method a nose tip candidate has to pass a four-level filtering scheme at the 3D vertex level in order to be identified (Pears et al., 2009). Their method is computationally expensive (requires about one minute on a AMD Athlon 64×2 Dual core 4200+ 2.20 Ghz, 4 Gb RAM machine running MATLAB), and cannot deal with pure profile facial views.

Anuar et al. propose to use point signature (Chua and Jarvis, 1997) for nose tip representation (Anuar et al., 2010). However, the point signature representation is not pose-invariant. Bevilacqua et al. propose to extend Generalized Hough Transform (GHT) to nose tip detection in 3D facial data (Bevilacqua et al., 2007). However, they assume that the region around the nose tip can be modelled as a series of spheres, an assumption not accurate.

In Breitenstein et al.'s method (Breitenstein et al., 2008), for a pixel to be a nose candidate its aggregated signature must have more local directional maxima than a given threshhold and the center of the set of the local directional maxima has to be part of the pixel's single signature. A nose tip and the associated pose is validated by finding the best match between the input range image and a set of reference pose range images whose orientations are close to that of the pose associated with the nose tip candidate. The reference pose range images are rendered by rotating a particular 3D head model under many poses, which is not always available in practice. More importantly, we argue that a particular model generated from a quite limited number of training data is unlikely to accurately represent any input face.

In contrast to the majority of the literature on 3D facial feature detection, only very few methods are able to detect the nose tip from 2D face profiles. Lu et al. assume that the nose tip has the largest z value (called "directional maximum" in their paper) if projected onto the corrected pose direction (Lu et al., 2008). The major drawback of their method is that it does not take the pitch into consideration. For this reason, Yang et al.'s method is also likely to fail because it relies on the directional maximum to firstly gain some nose tip candidates (Yang et al., 2009). Faltemier et al. propose to first get 37 profiles by rotating a face scan around the y-axis from the range of [0°, 180°] in a step size of 5°. Then for each profile they search for the nose tip by translating two nose profile models along the profile to search for the points that best match the nose tips of the two models, respectively (Faltemier et al., 2008). This translation-based model matching is obviously scale sensitive. In addition, just as in the case of 3D model, the generalization of their model is limited.

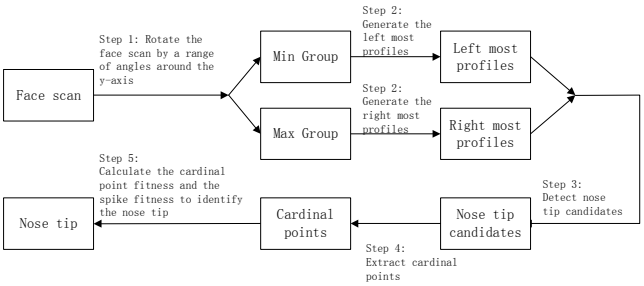


Figure 1: Pipeline of the nose tip detection algorithm.

We came up with a novel method for nose tip detection across a wide range of face poses (Peng et al., 2011). Compared with other existing nose tip detection methods, our method has the following three characteristics: First, it does not require any training nor rely on any particular model. Therefore, compared with training-based and model-based methods, it has the advantage of being exempt from laborious training and the pose limitations that are associated with training. Second, it works equally well for frontal and non-frontal poses, which is very important in practice because the pose of the face changes constantly in a sequence. Finally, it is relatively fast, requiring only a few seconds to process a facial range image of 100-200 pixels (in both x and y dimensions) with a MATLAB implementation.

Ren et al. propose a method for nose tip detection and tracking in 2D image sequences (Ren et al., 2007). In their method, the nose tip is detected in the first frame using GentleBoost and then tracked through the sequence using an improved Lucas-Kanade optical flow method. To the best of our knowledge, there is currently no publicly available literature on 3D face tracking in range image sequences.

3 NOSE TIP DETECTION AND TRACKING

In this section, we first describe the nose tip detection method followed by the nose-tip-detection-based face tracking scheme.

3.1 Nose Tip Detection Algorithm

To make this paper self-contained, we briefly describe below the main steps of the nose tip detection algorithm. Details can be found in (Peng et al., 2011). The input to the algorithm is assumed to be a 3D face region coarsely located by some face detection method. The reported parameters of the algorithm have been adjusted based on a resolution of 1mm/pixel. For a different resolution, these parameters will have to be adjusted accordingly. The pipeline of the algorithm is presented in Fig. 1 followed by its description.

The Nose tip Detection Algorithm

Input: one face scan containing N 3D face points with larger z-value points closer to the viewer.

Steps:

1. Rotate the Input Face Scan around the y-axis by an angle β within the range of [-90°, 90°] in a step size of 3°. Divide the 61 new point sets, each consisting of the rotated points of the original input face scan by a particular angle, into two groups called *the Min Group* and *the Max Group*.

The Min Group consists of 30 point sets corresponding to the rotations in the range of [-90°, -3°] and the Max Group contains the remaining 31 point sets corresponding to the rotations in the range of [0°, 90°]. Rotated around the y-axis by an angle of β, a face point (x_i, y_i, z_i) ($i = 1,2,...N$) in the original input face scan takes the new 3D coordinates $(x_i^\beta, y_i^\beta, z_i^\beta)$ as

$$\begin{pmatrix} x_i^\beta \\ y_i^\beta \\ z_i^\beta \end{pmatrix} = \begin{pmatrix} \cos\beta & 0 & \sin\beta \\ 0 & 1 & 0 \\ -\sin\beta & 0 & \cos\beta \end{pmatrix} \begin{pmatrix} x_i \\ y_i \\ z_i \end{pmatrix} \quad (1)$$

2. Generate the 2D *left most and right most face profiles* from the Min Group and the Max Group.

For a point set that corresponds to rotation β_j ($j = 1,2,...,30$) in the Min Group, its 2D *left most* face profile S_j is a 2D curve on the xy-plane

$$S_j = \{(x_k^j, y_k^j)_{k=1,2,...,N_j} \mid y_k^j \in \{0,1,...,R-1\}, \\ x_k^j = \min(\{x_i^{\beta_j} \mid (x_i^{\beta_j}, y_i^{\beta_j} = y_k^j, z_i^{\beta_j})_{i\in\{1,2,...,N\}}\})\} \quad (2)$$

where N_j is the total number of 2D points in S_j

and R the number of rows of the face scan. x_k^j is rounded to its nearest integer value in the case that it is non-integer. The 2D face profile S_j ($j = 31, 32, ..., 61$) for a point set in the Max Group can be defined analogously.

3. Detect Nose Tip Candidates. To this end, move the center of a circle of radius r (We use $r = 10$ which works very well in the experiments.) along each 2D face profile generated in step 2, and count the numbers of the 180 points (which are uniformly positioned along the perimeter of the circle) that are "inside" and "outside" the face profile when the circle's center is moved to each point on the face profile of interest (see figure 2). Let n^+ and n^- denote, respectively, the numbers of the 180 points inside and outside the face profile when the circle's center moves to a particular point on the face profile. For a point on the face profile to be a possible nose tip candidate, we require that $D = n^+ - n^- \leq -50$. This threshold is experimentally obtained and works well in the experiments.

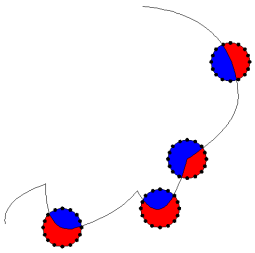


Figure 2: Nose tip detection on a Max Group face profile. A circle's center is moved along the face profile. At different positions on the face profile the areas of the circle enveloped "inside" the face profile (denoted by blue) are different. Smaller areas are likely to correspond to the nose tip. Actually, n^+ and n^- are counted instead of computing the areas directly.

4. Extract *cardinal points* from the nose tip candidates detected in step 3. Do the following: a) Initialize a histogram with each entry of the histogram corresponding to each y-coordinate value of the 2D face profiles. For each point (x_k^j, y_k^j) [$k = 1, 2, ..., N_j$, where N_j is the total number of points in S_j, refer to Eq. (2)] where a possible nose tip candidate is detected, increase the entry corresponding to y_k^j by one. Since the nose tip candidates cluster into spikes in the histogram, we only consider those points whose y-coordinates correspond to the peak value of the related spike. We call these points the "*cardinal points*" in this paper. We abandon all short spikes whose heights are either less than 30% of the highest spike or less than five to avoid false detections caused by noise.

5. Calculate two metrics, the *cardinal point fitness* and the *spike fitness* described below to identify the nose tip.

We observed that in a large number of cases the nose shape in a face profile can be fitted using a triangle. Thus, we use the cardinal point fitness to measure the degree of similarity between a triangle and a face profile segment corresponding to a nose shape candidate. For this purpose, we fit the face profile segment around a cardinal point S using connected straight line segments. For a fit to be valid we require that the connected straight line segments must form a maximum convex polygon (see Fig. 3). For example, the polygon (B2, B1, S, A1, A2, A3) in Fig. 3 is valid and adding more points, such as B3 and A4, will violate the requirement. In our implementation we use a line segment to approximate a curve segment if all the points in the curve segment are within distance d of the line segment. We empirically found that optimal values for d are two or three pixels.

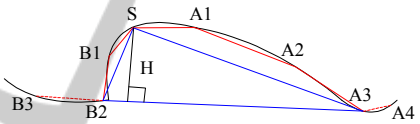


Figure 3: The fitness of the cardinal point S is calculated as $CF_S = 2 - S_p / S_t$, where S_p is the area of the polygon (B2, B1, S, A1, A2, A3) and S_t is the area of the triangle (B2, S, A3).

Obviously, a large CF_S value indicates that the nose shape candidate with S as the nose tip is similar to a triangle. Although a larger value of CF_S means that the face profile segment is closer to a triangle in shape, it does not necessarily mean that the segment represents a nose. Thus we introduce the definition of a *valid cardinal point*. A valid cardinal point satisfies all three heuristic conditions below (called the "*nose triangle conditions*" in this paper): a) The distance of the nose tip to the nose ridge must be larger than that of the nose tip to the nose bottom. In our implementation we require that the former is at least 1.2 times larger than the latter. b) The distance of the nose tip to the nose ridge must be within the range of 20-60 pixels. This heuristic helps to eliminate too large or too small triangle-like shapes which are not likely to be nose shapes. c) The altitude H perpendicular to the nose tip-nose bottom side of the triangle must be larger than 5 pixels. This heuristic helps to eliminate too flat triangle-like

shapes which are not likely to be nose shapes. Note that the parameters in conditions b) and c) are derived from observations of a large number of face samples and are set according to the resolution of the face scan (which is approximately 1mm/pixel).

Assume that there are M spikes to be considered, then the spike fitness of the m 'th ($m = 1, 2, ..., M$) spike is

$$SF_m = c_m \sum_s CF_s - D_m \tag{3}$$

where c_m is the ratio of the height of this spike to that of the highest spike, CF_s is the fitness of a valid cardinal point associated with this spike, and D_m is the maximum dimension of the cluster formed by all the valid cardinal points that contribute to the summation in Eq. (3). D_m is computed as

$$D_m = \begin{cases} 200e^{(-0.6D_m)}, & \text{if } D_m \le 5 \\ D_m, & \text{otherwise} \end{cases} \tag{4}$$

We use an exponent term to penalize small D_m values. The spike that has the largest fitness value is regarded as corresponding to the real nose tip candidates cluster. The valid cardinal point in this cluster with the highest altitude perpendicular to the nose tip-nose bottom side of the nose shape triangle is taken as the nose tip.

A MATLAB implementation of the above algorithm takes 2-4 seconds to detect the nose tip in an image of 121×121 pixels.

A by-product of the proposed method is that it can be used to coarsely estimate the roll, yaw and pitch angles of the face pose. Assume that the nose tip is detected at face profile S_j ($j = 1, 2, ..., 61$), which corresponds to an angle of β in the range [-90°, 90°], then the yaw rotation is estimated by

$$Yaw = \begin{cases} -(90° + \beta), & \text{if } \beta < 0 \\ 90 - \beta, & \text{if } \beta \ge 0 \end{cases} \tag{5}$$

The pitch rotation can be estimated by computing the angle between the y-axis of the profile S_j and the nose ridge-nose bottom line (the line connecting A3 and B2 in Fig. 3).

Assume that the positions of the detected nose ridge and nose bottom on the range image plane is (x_r, y_r) and (x_b, y_b), respectively. Let the angle between the vector $[x_r - x_b, y_r - y_b]$ and the y-axis of the range image plane be α, then the roll rotation can be estimated using

$$Roll = \begin{cases} -\alpha, & \text{if } x_r > x_b \\ \alpha, & \text{otherwise} \end{cases} \tag{6}$$

Experimental results show that the proposed method is robust to many scenarios that are encountered in common face recognition applications (e.g., surveillance). A high detection rate of 99.43% was obtained on FRGC v2.0 data set (Peng et al., 2011). The detection rate of our method is comparable to that of a most recent nose tip detection method (Pears et al., 2009) which achieves 99.6% detection rate on FRGC v2.0 data set. However, the method in (Pears et al., 2009) is much more computationally expensive (see Section 2 Related Work). A detailed complexity analysis of the above nose tip detection method is also presented in (Peng et al., 2011), which shows that most of the computations involved are simple. Thus, if implemented in hardware (such as a GPU implementation), the proposed method should be able to work in real time.

3.2 Nose-tip-Detection-based Face Tracking

In order to efficiently track the nose tip of a face in every frame of a 3D video sequence, it is required that the nose tip detection algorithm in Section 3.1 has a very high detection rate (ideally 100%). However, due to noise and missing data in the range image, this perfect detection rate is hard to achieve. In view of this, we combine our nose tip detection algorithm with the ICP algorithm (Besl and McKay, 1992) and a linear Kalman filter in a novel tracking scheme (shown in Fig. 4). Currently, our approach deals with the case of only one face appearing in a frame.

The ICP algorithm (Besl and McKay, 1992): Given two data clouds, the ICP algorithm establishes correspondence between each point of one data cloud with the closest point of the other data cloud. A transformation is then derived by minimizing the sum of the squared distances between these corresponding points. This transformation is applied to align the two data clouds and a new set of correspondences is established between their closest points. The algorithm is repeated iteratively until the sum of the squared distances between the points of the two data clouds reaches a threshold or there is no significant change between iterations.

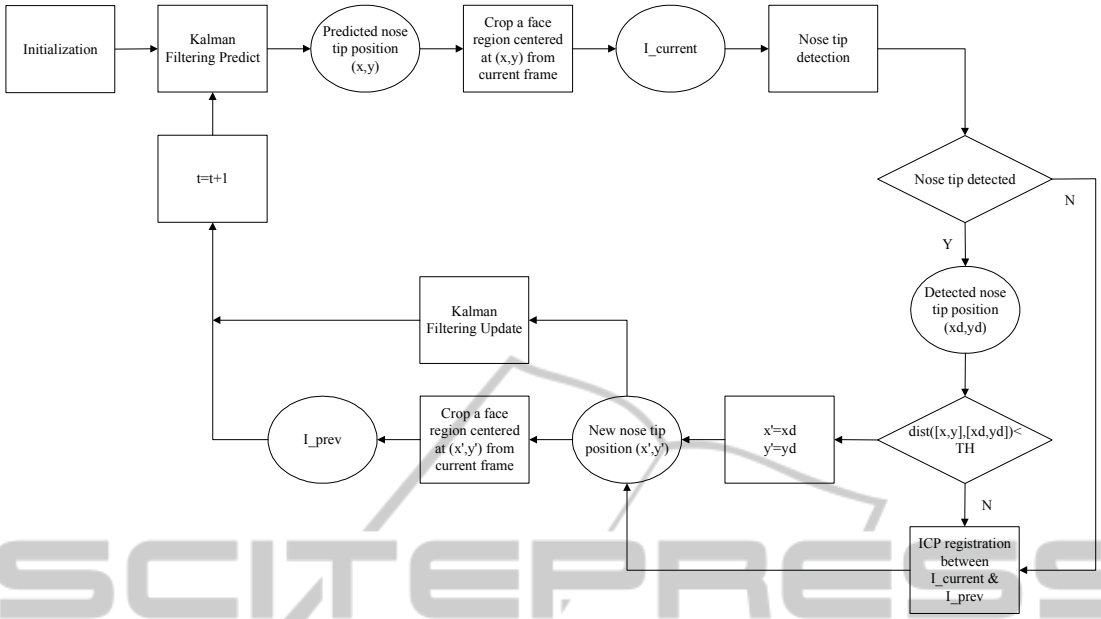


Figure 4: The nose-tip-detection-based face tracking scheme.

The linear Kalman filter: The linear Kalman filter is an optimal solution to the state estimation problem under the linear and Gaussian assumption. It consists of two steps, the prediction step and the update step. The advantage of the linear Kalman filter is that it does not require keeping a history of the states—only the previous state is needed to estimate the next state.

One loop of this iterative tracking process works as follows.

At iteration t, we first predict the position of the nose tip (x, y) in the current frame using Kalman filtering prediction. Then the face region is cropped at the current frame centered at (x, y); the result is denoted as $I\_current$. Subsequently, the nose tip detection algorithm of Section 3.1 is applied on $I\_current$. There are three possible cases for the nose tip detection results. Case (a): the detected nose tip position (x_d, y_d) is correct. According to the trajectory continuity characteristic, the nose tip positions in the immediately subsequent frames should be close to each other in the trajectory. In our implementation, we regard (x_d, y_d) as correct if it is within distance TH of the predicted position (x, y). Case (b): the detected nose tip position (x_d, y_d) is far away from the predicted position (x, y) due to false detections (based on the assumption that there is only one face). Case (c): no nose tip position is detected. This is usually because the quality of the image is so bad that there are not enough nose tip candidates detected. For the latter two cases, we register $I\_current$ with a face scan cropped from the previous frame, denoted as I_prev, using the ICP algorithm (Besl and McKay, 1992). Given the nose tip position in I_prev and the rotation matrix and translation vector obtained from the ICP algorithm, the nose tip position (x', y') in the current frame can be easily obtained. For the first case we simply set $x' = x_d$ and $y' = y_d$. We can then use (x', y') to update the Kalman filtering and crop I_prev at the current frame centered at (x', y'). The tracking process is thereafter propagated to iteration $t+1$.

We use the following linear Kalman filter in the tracking scheme:

$$\begin{cases} X(t) = \Phi(\Delta t)X(t-\Delta t) + w(t-\Delta t) \\ Y(t) = H(t)X(t) + v(t) \end{cases} \quad (7)$$

where $X(t)$ and $Y(t)$ are the state and observation vectors, respectively; $\Phi(\Delta t)$ and $H(t)$ are the state

$$\Phi(\Delta t) = \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \; H(t) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}, \text{ and}$$

$\Delta t = 1$.

The prediction and update equations of the Kalman filter are

$$\begin{cases} X^{-}(t) = \Phi(\Delta t)X^{\sim}(t-\Delta t) \\ P^{-}(t) = \Phi(\Delta t)P^{\sim}(t-\Delta t)\Phi^{T}(\Delta t) + Q(t-\Delta t) \end{cases} \quad (8)$$

and

$$\begin{cases} K(t) = P^{-}(t)H^{T}(t)[H(t)P^{-}(t)H^{T}(t) + R(t)]^{-1} \\ P^{\sim}(t) = [I - K(t)H(t)]P^{-}(t) \\ X^{\sim}(t) = X^{-}(t) + K(t)[Y(t) - H(t)X^{-}(t)] \end{cases} \quad (9)$$

respectively. In Eqs. (8) and (9), "-" and "~" represent the *a priori* and *a posteriori* estimations of a random variable, respectively; $Q(t) = E(w(t)w(t)^{T})$ and $R(t) = E(v(t)v(t)^{T})$ are the covariance matrices of w and v, respectively. In all our implementation $R(t)$ and $P^{-}(0)$ are identity matrices **Id**, and $Q(t) = 3*\mathbf{Id}$. These three parameters work well in the experiments.

When the nose tip detection algorithm fails, we use the ICP algorithm to obtain the nose tip position in the current frame. It may seem to the reader that the ICP algorithm is more "reliable" than the nose tip detection algorithm and one may wonder why we do not just use ICP for the tracking scheme. The main reason is that if we at every iterative step only use ICP to register adjacent frames and propagate the nose tip position from the previous frame to the next frame, then at each step small registration errors are generated and accumulated; when the accumulation of the registration errors builds up as the tracking proceeds, the nose tip position will gradually drift away. This situation is similar to the drift problem in template-based tracking in 2D image sequences (Matthews et al., 2004).

4 EXPERIMENTAL RESULTS

The method was implemented in MATLAB R2007a and ran on a desktop PC (3.2GHz Pentium CPU, 1GB RAM, Microsoft Windows XP Professional OS). The data used in our experiments is from the ETH Face Pose Range Image Data Set (Breitenstein et al., 2008). Robust tracking results were obtained on a dozen of 3D video sequences containing about 5000 frames in total. These sequences were recorded using a range scanner at a frame rate of 28 fps. All frames in the sequences are 640×480 pixels in size, and contain only the head part of a subject. The head pose range covers about ± 90° in the yaw and ± 45° in the pitch. A face typically occupies about 150×200 pixels in a frame (approximately 1mm/pixel). The ground truth of the nose tips is also available in the data set. Due to the space limit of this paper, here we only present the tracking results of three typical video sequences from the data set, M16, M5 and M5g. When naming the video names, the "M" denotes male subject, the "g" denotes that the subject wears glasses, and the numbers are assigned to distinguish different subjects.

Even though the spike noise has been removed from the data beforehand, they generally exhibit a poor quality. There are many holes in the data and the boundaries of the faces are zigzagged. In addition, in the data smaller z-value points are closer to the viewer—contrary to our assumption that larger z-value points are closer to the viewer. Therefore, in order to apply the proposed method, we first pre-processed the data in the following four steps: First, we replaced the original z-values in each frame by subtracting them from the largest valid z-value. Second, we applied a morphological close operation on the frame followed by a morphological open operation. The aim of this step is to smooth the boundaries of the face. The disk structuring elements used for the close and open operations have radii of 7 and 2 pixels, respectively. Third, we filled in the holes in the frame using the "imfill" MATLAB command. Finally, we applied a 3×3 averaging filter to the frame. This step helps to remove small spurs on the face. Fig. 5 presents two data pre-processing examples.

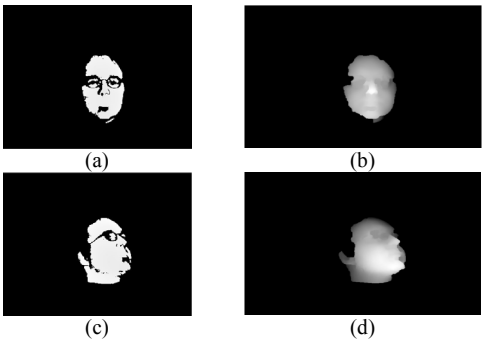


Figure 5: Two data pre-processing examples. (a) An original face range data. (b) The result of pre-processing (a). (c) Another original face range data. (d) The result of pre-processing (c).

The parameters used for the tracking process is as follows: the dimensions of $I\_current$ and I_prev are 121×121 pixels, and TH is empirically chosen based on visual inspection of the maximum

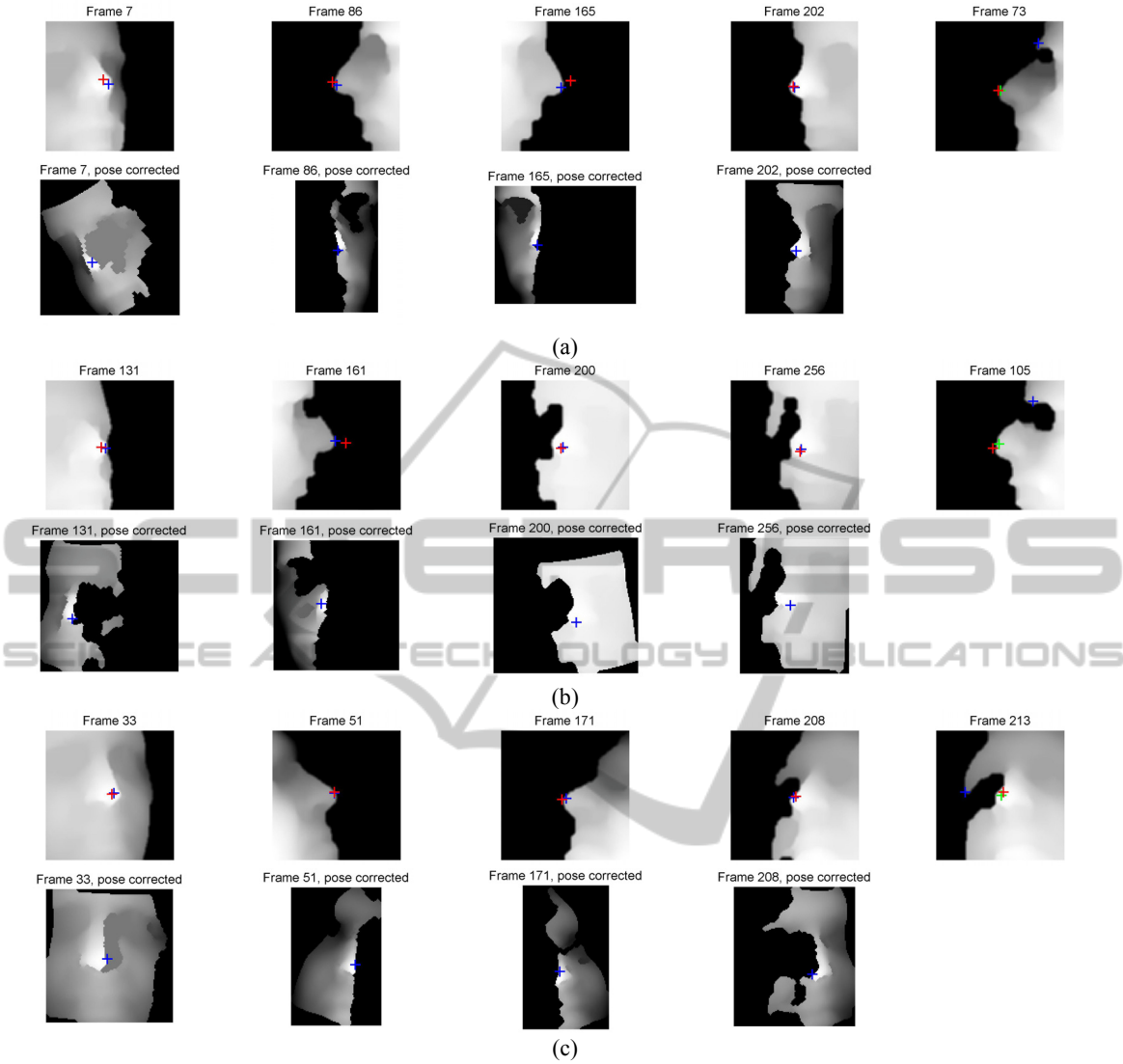


Figure 6: Tracking results obtained using our method. (a) The first row corresponds to M5 and the second row to corrected poses from the estimated pose angles. (b) The first row corresponds to M5g and the second row to corrected poses from the estimated pose angles. (c) The first row corresponds to M16 and the second row to corrected poses from the estimated pose angles. Red crosses correspond to the ground truth (Breitenstein et al., 2008) of nose tip positions; blue crosses correspond to the detected nose tips using the nose tip detection method in Section 3.1; green crosses correspond to the detected nose tips using the ICP algorithm (figure best seen in color).

movement of the nose tip in adjacent frames. However, in real applications *TH* should be chosen otherwise, e.g., based on a number of training sequences. To initialize the tracking process, an initialization step is required. Since our nose tip detection algorithm assumes that the input to the algorithm is a facial region, we assume that the initialization step can be accomplished using another algorithm that is able to detect the face region from a range image. For the sake of this paper, we manually extract a window of 121×121 pixels from the first frame centered on the ground truth nose tip and the tracking thereafter begins. The tracking results along with corrected face poses are presented in Fig. 6. For each video we use two rows to show the results, one row for the tracked nose tip and the other for the corrected poses. The last column corresponding to the first rows justifies our choice of ICP as a complement to the nose tip detection method—in these cases the ICP were activated when the nose tip detection failed.

As shown in Fig. 6, despite detection failures in

Table 1: Description of the image sequences used in the experiments and tracking results.

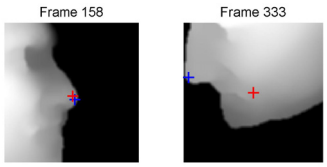
Sequence name	M5	M5g	M16
no. of frames	361	331	543
Ave. running time (s) [a)]	2.78	2.77	3.05
Detection rate [b)]	98.34%	88.52%	99.82%
Error mean (pixels) [c)]	4.49	3.81	2.56
Error std. deviation (pixels) [c)]	2.00	1.93	1.68
Frame no. when tracking failed by using ICP only	8	14	37

[a)] The computation was carried out on a window of 121×121 pixels.
[b)] The detection rate is the ratio of the number of frames in which the nose tip was successfully detected out of the total number of frames in the sequence.
[c)] Calculated only for the frames in which the nose tip was successfully detected and in the case where the ground truth (Breitenstein et al., 2008) is correct.

some frames, all three sequences were successfully tracked until the end. Note that in Fig. 6 some pose-corrected faces have much less face points than others. This is because these face poses are non-frontal and a large portion of the faces was missing due to occlusion. In Table 1 we also summarize the average running time of our method to detect a nose tip in one frame, the nose tip detection rates for each sequence, and the average errors between the detected results and the ground truth.

It can be seen from Table 1 that the detection rate of the nose tip detection method deteriorated when the subject had glasses on. This is because glasses leave more zigzagged boundaries in a face scan. Zigzagged boundaries generate more candidates in the nose tip detection step and increase the chances of false alarms and thus lead to lower detection rates. We believe that better data pre-processing techniques will further improve the results. One interesting fact in Table 1 is that although the detection rate of sequence M5g is the lowest compared with the other two sequences, the tracking accuracy for this sequence in terms of error mean and error standard deviation is comparable to those of the other two sequences. This fact partly demonstrates the robustness of the proposed nose tip detection algorithm to noise.

We observed in some frames that the ground truth (Breitenstein et al., 2008) is visually far from the real nose tip position. Consequently, two ways have been used to judge whether our detection is considered successful. If the visual inspection reveals that the nose tip of the ground truth is inaccurate, we rely on our visual inspection to decide on whether the nose tip was accurately detected. Otherwise, a detection result is considered accurate if the detected nose tip is within 10 pixels from the ground truth. Two examples are shown in Fig. 7.



(a) (b)

Figure 7: Two examples (from sequence F7g) where the ground truth (Breitenstein et al., 2008) can be judged accurate based on visual inspection. Red crosses correspond to the ground truth (Breitenstein et al., 2008) of nose tip positions, while blue crosses correspond to the detected nose tips using the proposed method. Obviously, the ground truth for the nose tip position in (b) is inaccurate.

Since there is currently no publicly available literature on 3D face tracking in range image sequences, for the purpose of comparison we used ICP only for tracking the three sequences. In these experiments, when the tracked nose tip entirely drifted away from the nose region, we judged the tracking as having failed. The results are also included in Table 1, which show that using ICP only for tracking performed poorly.

5 CONCLUSIONS AND FUTURE WORK

This paper addresses an important application of point cloud data processing in 3D facial feature detection and tracking. In particular, in this paper we propose a nose-tip-detection-based face tracking method which combines a nose tip detection method with the Iterative Closest Point (ICP) algorithm and a Kalman filter. The proposed approach tested on real 3D sequences achieved promising tracking results. We envision that our approach has two potential real applications. One possible application is for a non-contact interface for disabled users. In this case, the user's nose is used as a pointing device in a human-computer interaction manner in 3D space. The other possible application is for 3D video-based face recognition for authentification or surveillance purposes. In this case our method works

as a prerequisite step before the recognition of a face can happen.

However, there is still some substantial future work that needs to be done. Some possible directions are as follows:

Firstly, the proposed approach can fail in the case where the nose tip detection method and the ICP algorithm fail simultaneously. This could happen when two consecutive views of a face are both very noisy (which can lead to false alarms) and change rapidly in pose/position. The ICP algorithm may fail when the movement of the face between two adjacent frames is large to the extent that the ICP algorithm is unable to register the current and previous frames correctly. Therefore, how to improve the tracking scheme so that these two components can better complement each other is one direction of our future work.

Secondly, the pose correction in the proposed approach is coarse. We believe that there is potential to improve this step in the future.

Finally, currently our approach deals with the case of only one person appearing in a frame. It requires extending the approach to tracking multiple faces simultaneously appearing in a frame and dealing with appearing faces and disappearing faces. This will be the third direction of our future work.

REFERENCES

Al-Osaimi, F., Bennamoun, M., Mian, A., 2009. *An expression deformation approach to non-rigid 3d face recognition,* International Journal of Computer Vision, 81: 302-316.

Anuar L. H., Mashohor S., Mokhtar M., and Wan Adnan W.A. 2010. *Nose tip region detection in 3D facial model across large pose variation and facial expression,* IJCSI International Journal of Computer Science Issues, 7(4): 1-9.

Besl, P. J., McKay, N. D., 1992. *Reconstruction of Real-World Objects via Simultaneous Registration and Robust Combination of Multiple Range Image,*. IEEE Transactions on Pattern Analysis and Machine Intelligence, 14(2), 239-256.

Bevilacqua V., Casorio P., and Mastronardi G. 2008. *Extending Hough Transform to a points' cloud for 3D-face nose-tip detection,* Lecture Notes in Artificial Intelligence, 5227: 1200-1209.

Breitenstein, M. D., Kuettel, D., Weise, T., VanGool L. J., Pfister, H., 2008. *Real-timeface pose estimation from single range images,* In CVPR'08, pp.1-8.

Chang, K. I., Bowyer, K. W., Flynn, P. J., 2006. *Multiple nose region matching for 3D face recognition under varying facial expression*, IEEE Transactions on Pattern Analysis and Machine Intelligence, 28(10): 1695-1700.

Chew W. J., Seng K. P., and Ang L. M. 2009. *Nose tip detection on a three-dimensional face range image invariant to head pose,* In IMCES 2009, pp. 858-862.

Chua C. S., and Jarvis R. 1997. *Point signatures: a new representation for 3D object recognition,* International Journal of Computer Vision, 25(1): 63-85.

Colombo A., Cusano C., and Schettini R. 2006. *3D face detection using curvature analysis,* Pattern Recognition, 39(3): 444-455.

Faltemier T. C., Bowyer K. W., and Flynn P. J. 2008. *Rotated profile signatures for robust 3D feature detection,* In the 8th IEEE International Conference on Automatic Face & Gesture Recognition, pp.1-7.

Lu, X., Jain, A. K., 2008. *Deformation modeling for robust 3d face matching*, IEEE Transactions on Pattern Analysis and Machine Intelligence, 30(8): 1346-1356.

Malassiotis, S. and Strintzis, M. G. 2005. *Robust real-time 3D head pose estimation from range data,* Pattern Recognition, 38(8): 1153-1165.

Matthews, I., Ishikawa, T., Baker S., 2004. *The template update problem*, IEEE Transactions on Pattern Analysis and Machine Intelligence, 26: 810-815.

Mian, A. S., Bennamoun, M., Owens, R., 2007. *An efficient multimodal 2D-3D hybrid approach to automatic face recognition,* IEEE Transactions on Pattern Analysis and Machine Intelligence, 29(11): 1927-1943.

Pears, N., Heseltine, T., Romero, M., 2009. *From 3D point clouds to pose-normalized depth map,* International Journal of Computer Vision, DOI 10.1007/s11263-009-0297-y.

Peng, X., Bennamoun, M., and Mian, A.S. 2011. *A training-free nose tip detection method from face range images.* Pattern Recognition, 44(3): 544-558.

Queirolo, C. C., Silva, L., Bellon, O. R. P., Segundo, M. P., 2010. *3D face recognition using simulated annealing and the surface interpenetration measure,* IEEE Transactions on Pattern Analysis and Machine Intelligence, 32(2): 206-219.

Ren X., Song J., Ying H., Zhu Y., and Qiu X. 2007. *Robust nose detection and tracking using GentleBoost and improved Lucas-Kanade optical flow algorithms,* Lecture Notes in Computer Science, 4681: 1240-1246.

Xu C., Tan T., Wang Y., Quan L. 2006. *Combining local features for robust nose location in 3D facial data,* Pattern Recognition Letters, 27(13): 1487-1494.

Yang J., Liao Z., Li X., and Wu Z. 2009. *A method for robust nose tip location across pose variety in 3D face data,* In 2009 International Asia Conference on Informatics in Control, Automation and Robotics, pp. 114-117.

Zhou, S. K., Chellappa, Aggarwal, R., G., 2009. *Face tracking and recognition from video*, Book Chapter from The Essential Guide to Video Processing, A. Bovic (Ed.), Academic Press.