Real-Time Head Pose Tracking with Online Face Template Reconstruction

Songnan Li, *Member, IEEE*,
King Ngi Ngan, *Fellow, IEEE*,
Raveendran Paramesran, *Senior Member, IEEE*,
and Lu Sheng

Abstract—We propose a real-time method to accurately track the human head pose in the 3-dimensional (3D) world. Using a RGB-Depth camera, a face template is reconstructed by fitting a 3D morphable face model, and the head pose is determined by registering this user-specific face template to the input depth video.

Index Terms—Head pose tracking, deformable face model, iterative closest point, RGB-Depth camera

1 Introduction

THE objective of head pose tracking is to estimate the 3D rigid head movement, i.e., translation and rotation, for each frame of a video sequence. It is a challenging computer vision problem with a wide range of applications such as human computer interaction, augmented reality, gaze estimation and free viewpoint TV.

There are a variety of approaches in the literature that exclusively used color information to estimate the human head pose in an image or across a video sequence. See [9] for an excellent survey. Since the launch of Microsoft Kinect, researchers have increasingly leveraged the RGB-Depth (RGBD) cameras to address challenging computer vision problems, such as human skeleton estimation, 3D reconstruction, facial expression tracking, and hand gesture tracking. Besides the color information, a RGBD camera can provide an additional depth channel of the scene at a video frame rate.

Recently Fanelli et al. [1] trained a random regression forest to directly map depth features to head poses. The performance of such learning based methods [1], [2], [17] depends heavily on the comprehensiveness of the training dataset, which should involve a wide range of head translations and rotations along all X, Y, and Z directions, a large number of subjects, facial expressions and inter-object occlusions (e.g., eyeglasses). Current head pose datasets, such as BIWI [19] and ICT-3DHP [20], cannot satisfy all these requirements. For example, both datasets cover limited range of head translations, especially for the Z (depth) direction. Another category of RGBD based methods [3], [4] tracked 3D facial landmarks and accordingly estimate the head pose. They can be considered as 3D extensions of 2D non-rigid facial tracking algorithms, such as AAM, ASM or CLM [10], by jointly using the color and depth information. Color-based landmark detection or feature tracking were performed which are computationally expensive and not robust to illumination changes. Furthermore, these methods are not suitable for applications which require only the head pose as input, since the time consumed by locating facial landmarks should be spared.

The proposed method belongs to another category of approaches, namely template matching. In [5], head pose was determined by comparing the matching errors between an input depth image and a bunch of template images which were pre-computed (offline) by diversely posing a 3D average face. Nose analysis was performed to locate nose candidates, and template images were selected according to the nose positions and orientations. Since there are a limited number of template images, the space of head poses is discrete, which may degrade the precision. In [6], Padeleris et al. selected one frame (typically the first one) of a sequence as a template image. For each subsequent frame, hypothesized images were rendered by applying different translations and rotations on the depth image. The head pose is determined by maximizing the similarity between the template image and the hypothesized images. Particle swarm optimization was employed to search the six-dimensional solution space. Therefore, a large number of hypothesized images need to be rendered increasing the computational cost. Rather than using 2D template images, in [7], [8] 3D face templates were directly matched to the 3D point cloud converted from the depth values. It turns the head pose tracking into a rigid 3D matching problem which can be time-efficiently implemented using algorithms such as the iterative closest point (ICP) [15]. In these works the 3D face template was reconstructed in a pre-processing step (offline). Our work differs in that the 3D face template is reconstructed online, i.e., during the real-time head pose tracking.

In this paper, a face template reconstruction algorithm is developed which is 20 times faster than existing similar approaches (based on the 18 seconds per frame reported in [11]). Rather than requiring a pre-processing step to reconstruct the face template offline, the reconstruction can be sequentially accomplished during the head pose tracking using up to nine frames which are automatically selected from the input video. The absence of an offline pre-processing step greatly enhances the user experience. The accuracy of our head pose tracking system outperforms the state-of-the-art methods by a large margin, as shown in Section 5. Furthermore, it has a low computational complexity. Without specific code optimization, processing one frame only takes about 21 ms, equivalent to a tracking speed of around 47 fps. Furthermore, it can track fast translational movement (0.3 meter per second), fast rotational movement (120 degree per second) and large rotation angles (50 degree for each of the pitch, yaw and roll rotations). The system is robust to large occlusion and can recover from tracking failure quickly. High accuracy, low complexity and good robustness make the system suited to many real-life applications.

The rest of the paper is structured as follows. Section 2 introduces the framework of our head pose tracking method. Section 3 proposes the online face template reconstruction algorithm. Section 4 presents how to use the reconstructed face template to track head poses in a RGBD video and use K-means clustering to identify outliers caused by hair or eyeglasses occlusion. Section 5 shows the experimental results. Finally, Section 6 draws the conclusion.

2 Overview

The general framework of our head pose tracking method is illustrated in Fig. 1. With known intrinsic camera parameters, the input color and depth video frames are converted into a 3D point cloud. To estimate the head pose, we leverage the time-efficient iterative closest point algorithm to register a face template to the point cloud in the 3D space. ICP is an iterative optimization process which needs a good initialization to converge to the global optimum. This initialization is derived from head pose prediction which uses the historical head poses to predict the next one.

To estimate the head pose in each video frame, the ICP algorithm needs to use a 3D face template. To enhance the precision, this face temple should be specific to the tracked subject (user). Our work differs from the prior similar studies [7], [8] in that we reconstruct this user-specific face template during the real-time head pose tracking (online) rather than using an offline pre-computation.

Before the user-specific template is ready, an average template will be used which will unavoidably deteriorate the tracking

- *S. Li, King N. Ngan, and L. Sheng are with the Department of Electronic Engineering, The Chinese University of Hong Kong. E-mail: {snli, knngan, lsheng}@ee.cuhk.edu.hk.*
- *R. Paramesran is with the University of Malaya, Malaysia. E-mail: ravee@um.edu.my.*

Manuscript received 29 Dec. 2014; revised 12 Oct. 2015; accepted 9 Nov. 2015. Date of publication 11 Nov. 2015; date of current version 11 Aug. 2016.
Recommended for acceptance by S. Sarkar.
For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below.
Digital Object Identifier no. 10.1109/TPAMI.2015.2500221

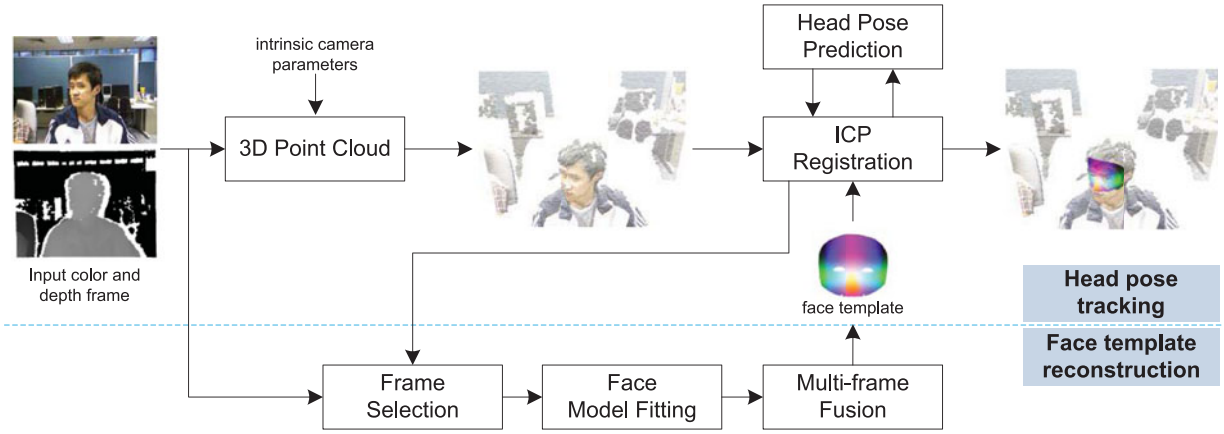

Fig. 1. General framework of the proposed head pose tracking method.

performance. Therefore, a fast reconstruction algorithm is required to ensure high accuracy as early as possible. We achieve fast reconstruction by tailoring the optimization process and using sequential reconstruction. To enhance the accuracy, multiple frames capturing different views of the subject's face are selected for reconstruction. Rather than requiring all these frames to be available before the reconstruction (causing a long time delay), our method reconstructs the face template frame-by-frame, and uses a time-efficient step to fuse reconstruction results from multiple frames. The reconstruction can start typically in the first frame of a video sequence, making our method suffer less from the reconstruction time delay. The quality of the face template is improved incrementally as more frames are used.

As can be seen from Fig. 1, head pose tracking and face template reconstruction are two loosely dependent processing stages. Information needs to be shared only occasionally, e.g., when the updated face template is ready. Therefore, we use two threads to implement the two stages, so that the computational costs can be allocated into multiple cores of the CPU, and the relatively time-consuming face template reconstruction will not slow down the foreground real-time head pose tracking.

3 Face Template Reconstruction

The face template reconstruction consists of three sequential processing stages, including frame selection, face model fitting, and multi-frame fusion, as shown in Fig. 1. Motivation and implementation details on each processing stage are given below.

3.1 Frame Selection

In the course of tracking, multiple frames are automatically selected from the video sequence to reconstruct the user-specific face template. Multiple frames can capture different views of the face, so as to handle the face self-occlusion. It can also suppress the depth noise, by averaging 3D positions of the same point in multiple frames.

Fig. 2 illustrates the fitting zones for frame selection. These fitting zones are bounded by the depth positions and the yaw rotations of the head pose. As shown in Fig. 1, the real-time head pose tracking results are monitored in the frame selection process. When the head is found to enter a fitting zone, the corresponding color and depth frames will be selected. There are totally nine fitting zones in our implementation. Once a fitting zone has been visited, it is marked as invalid. Thus, up to nine frames will be used for the face template reconstruction.

3.2 Face Model Fitting

Once a new video frame is selected, a morphable 3D face model will be fitted to it. We use the Basel Face Model (BFM) [13] which is a linear model trained on neural expression face scans of $N = 200$ subjects. BFM represents the face shape using a 3D mesh comprised of $M = 53{,}490$ vertices. Each face sample can be taken as a $3L$ dimensional vector $(x_1, y_1, z_1, \ldots x_L, y_L, z_L)$, i.e., x, y, z coordinates of all vertices. Principal Component Analysis (PCA) is performed on the 200 training face samples to estimate the mean $\boldsymbol{\mu} \in \Re^{3L}$, the standard deviations $\boldsymbol{\sigma} \in \Re^{N-1}$, and the orthonormal bases of principal components $\mathbf{P} \in \Re^{3L\times(N-1)}$, resulting in a parametric face model consisting of $\mathbf{M}$=($\boldsymbol{\mu}$, $\boldsymbol{\sigma}$, $\mathbf{P}$). A new face can be generated as a linear combination of the principal components

$$\mathbf{f} = \boldsymbol{\mu} + \mathbf{P}diag(\boldsymbol{\sigma})\boldsymbol{\alpha}, \tag{1}$$

where $\boldsymbol{\alpha} \in \Re^{N-1}$ are the principal component coefficients which are normally distributed with unit variance.

We fit BFM to the newly selected video frame by minimizing the following objective function

$$\min_{\mathbf{R},\mathbf{t},\boldsymbol{\alpha},\mathbf{d}_i} \sum_{i=1}^{L} w_i[(\mathbf{R}\mathbf{n}_i)\cdot(\mathbf{R}(\boldsymbol{\mu}_i + \mathbf{P}_i\boldsymbol{\alpha}) + \mathbf{t} - \mathbf{d}_i)]^2 + \lambda \sum_{j=1}^{K} \frac{\alpha_j^2}{\sigma_j^2}, \tag{2}$$

where $\boldsymbol{\alpha} \in \Re^K$ are the principal component coefficients of the BFM face model. The ith vertex of the new face is

$$\mathbf{f}_i = \boldsymbol{\mu}_i + \mathbf{P}_i\boldsymbol{\alpha}. \tag{3}$$

The first $K = 60$ principal components are used to generate this new face. Different from eq. (1) where the principal component coefficients have unit variance, in eq. (2) their values are explicitly regularized by the standard deviations σ_js, $j \in \{1, 2, \ldots K\}$. This

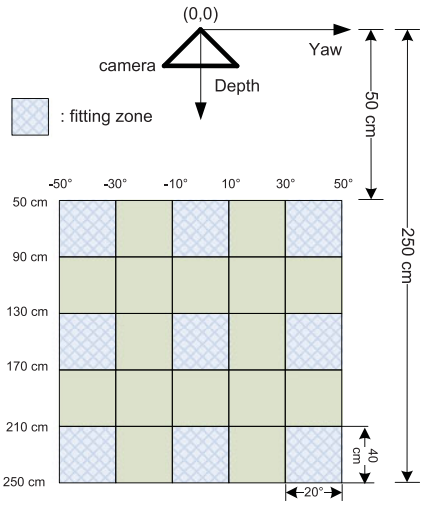

Fig. 2. Fitting zones for frame selection.

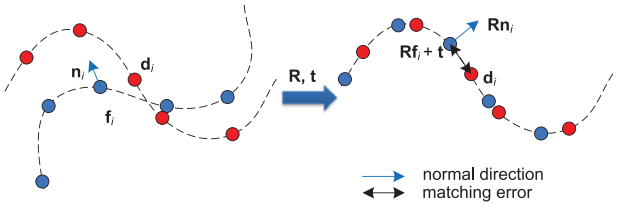

Fig. 3. An 2D illustration for the face model fitting process formulated by eq. (2).

regularization term constrains the Mahalanobis distance between the mean face $\boldsymbol{\mu}$ and the new face, so that the new face can be a realistic human face rather than being twisted. λ is an empirical scalar constant determining the regularization strength.

In eq. (2), $\mathbf{R}$ and $\mathbf{t}$ determine the head pose. $\mathbf{R} \in \Re^{3\times3}$ is the rotation matrix and $\mathbf{t} \in \Re^3$ is the translation vector. $\mathbf{d}_i$ is the matching point of $\mathbf{f}_i$ in the point cloud derived from the depth video frame; and $\mathbf{n}_i$ is the unit normal vector of $\mathbf{f}_i$. A (2D) illustration is given in Fig. 3. After rotation and translation, the matching error between $\mathbf{f}_i$ and $\mathbf{d}_i$ is projected onto $\mathbf{f}_i$'s normal direction using the inner product operation. The projection in the formulation is to solve the sampling problem in 3D registration. As illustrated in Fig. 3, although the two point clouds, indicated by the red and blue dots, respectively, are sampled from the same surface, the matching error can still be large when the two point clouds are perfectly registered, since their sampling patterns are quite different. But once projected onto the normal direction of the surface, the matching error is diminished.

Furthermore, w_i in eq. (2) is a scalar weighting factor for the ith vertex. The use of w_i has the following effects. First, by setting w_i to zero, a part of the face vertices can be excluded from the calculation. For example, vertices in the mouth, chin and eye regions are excluded in the face template, as shown in Fig. 1, because they can be easily affected by expressional deformations. To reduce computational complexity, we downsample the 3D face mesh. The total number of mesh vertices in use is trimmed to 1,100. Second, by assigning some vertices larger weights, we can emphasize their importance during the optimization. Specifically, we use a time-efficient facial landmarks detector (IntraFace [14]) to locate facial features such as the eye corners and the nose tip in the video frame. By assigning these feature points larger weights, the optimization of eq. (2) can converge more quickly, because the matching points $\{\mathbf{f}_i, \mathbf{d}_i\}$ found by the facial landmark detection tends to be more accurate compared with those found by projection and local search, especially in the first few iterations of the optimization. In the implementation, we use five facial landmarks (four canthi plus the nose tip) and set w_i to 40 for each of them.

The optimization of eq. (2) is carried out in an iterative process which consists of three major sequential steps:

(1) With constant $\mathbf{R}$, $\mathbf{t}$ and $\boldsymbol{\alpha}$, find the matching points $\{\mathbf{f}_i, \mathbf{d}_i\}$s;
(2) Given matching points and constant $\boldsymbol{\alpha}$, calculate $\mathbf{R}$ and $\mathbf{t}$;
(3) Given matching points, the updated $\mathbf{R}$ and $\mathbf{t}$, calculate $\boldsymbol{\alpha}$.

For fast face template reconstruction, each step is calculated using a time-efficient method.

In the first step, $\mathbf{f}_i$ is perspectively projected onto the depth image (projective data association [21]), followed by a local search around the projection point to find the closest $\mathbf{d}_i$. A pair of matching points $\{\mathbf{f}_i, \mathbf{d}_i\}$ is rejected if their distance is larger than a predefined threshold (30 mm), or the normal vector of $\mathbf{f}_i$ points away from the camera center (self-occluded).

The second step is equivalent to minimizing the following point-to-plane ICP objective function

$$c_1(\mathbf{R}, \mathbf{t}) = \sum_{i=1}^{L} w_i[(\mathbf{Rn}_i) \cdot (\mathbf{R}(\boldsymbol{\mu}_i + \mathbf{P}_i\boldsymbol{\alpha}) + \mathbf{t} - \mathbf{d}_i)]^2. \quad (4)$$

By assuming small head rotation angles across adjacent frames [4], [21], eq. (4) can be converted into a quadratic function and its minimization can be solved analytically.

Using the updated $\mathbf{R}$ and $\mathbf{t}$ in the third step, the BFM coefficients $\boldsymbol{\alpha}$ can be obtained by minimizing

$$c_2(\boldsymbol{\alpha}) = \sum_{i=1}^{L} w_i[(\mathbf{Rn}_i) \cdot (\mathbf{R}(\boldsymbol{\mu}_i + \mathbf{P}_i\boldsymbol{\alpha}) + \mathbf{t} - \mathbf{d}_i)]^2 + \lambda \sum_{j=1}^{K} \frac{\alpha_j^2}{\sigma_j^2} \quad (5)$$

which is also a quadratic function. The minimization solution of eq. (5) is given below

$$\boldsymbol{\alpha} = -\left(\sum_{i=1}^{M} \mathbf{P}_i{}^{\top}\mathbf{W}_i\mathbf{P}_i + \lambda\mathbf{Q}\right)^{-1}\left(\sum_{i=1}^{M} \mathbf{P}_i{}^{\top}\mathbf{W}_i{}^{\top}\mathbf{g}_i\right), \quad (6)$$

where

$$\mathbf{g}_i = \mathbf{R}^{\top}(\mathbf{t} - \mathbf{d}_i) + \boldsymbol{\mu}_i \quad (7)$$

$$\mathbf{W}_i = w_i\mathbf{n}_i\mathbf{n}_i{}^{\top} \quad (8)$$

$$\mathbf{Q} = \begin{bmatrix} \frac{1}{\sigma_1^2} & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \frac{1}{\sigma_K^2} \end{bmatrix}. \quad (9)$$

Limited by the paper length, details for minimizing eq. (4) and the derivation of eq. (6) are given in the supplementary material [18].

It should be noted, after the above analysis of eq. (2), that the benefits of using the morphable face model are mainly threefold. First, as elaborated at the beginning of this section, BFM was constructed using 200 high quality (noise-free) face scans. By weighting the regularization term of eq. (2) properly, we can use BFM to generate a smooth face template that is able to well approximate the noisy input. The noise robustness brought about by this morphable face model is especially beneficial to our problem, since the Kinect depth video is notoriously noisy. Second, as discussed above, the mouth and chin regions are excluded in the face template to relieve the negative effect of facial expression. Without using BFM, to identify these semantic regions in a point cloud is not so straightforward. Last, as shown in eq. (1), BFM is a linear model, which makes the optimization of eq. (2) and the subsequent multi-frame fusion step much more time-efficient than non-rigidly deforming a general face template to fit the input point cloud.

3.3 Multi-Frame Fusion

To handle face self-occlusion and depth noise, model fitting results from multiple frames are fused together time-efficiently in this stage.

In eq. (6), $\mathbf{g}_i$ is a frame dependent variable. To emphasize its frame-dependent nature, we denote it with a subscript

$$\mathbf{g}_{l,i} = \mathbf{R}_l{}^{\top}(\mathbf{t}_l - \mathbf{d}_{l,i}) + \boldsymbol{\mu}_i, \quad (10)$$

where l is the frame index; $\mathbf{R}_l$ and $\mathbf{t}_l$ are respectively the rotation matrix and the translation vector of frame l. Calculated in the lth frame, $\mathbf{g}_{l,i}$ is the 3D displacement of the ith vertex of the user's face from that of the mean face in the model coordinates. To fuse the BFM fitting results from multiple frames, we calculate the average value of $\mathbf{g}_{l,i}$s

$$\mathbf{g}_i^{F_i} = \frac{\sum_{l=1}^{F_i} \mathbf{g}_{l,i}}{F_i} \quad (11)$$

and use it in eq. (6) to determine the BFM coefficients $\boldsymbol{\alpha}$. Note that due to the face self-occlusion and depth holes, the number of frames F_i may be different for different vertices. Also note that we assume the face in the template region is static across the chosen

frames. Facial expression will violate this assumption, but since the template excludes the mouth, chin, and eyes, the influence from facial expression is largely diminished.

Multi-frame fusion is performed after the face model fitting for each selected frame. To reduce memory usage, the average value $\mathbf{g}_i^{F_i}$ and the number of frames F_i are recorded for each vertex. When a new frame is selected, the average value can be updated memory-efficiently by a weighted average

$$\mathbf{g}_i^{(F_i+1)} = \frac{F_i \mathbf{g}_i^{F_i} + \mathbf{g}_{F_i+1,i}}{F_i + 1}. \quad (12)$$

The normal vectors $\mathbf{n}_i$s are updated following the multi-frame fusion, and then a notification will be sent to the head pose tracking thread to inform that an updated face template is ready for use.

4 Head Pose Tracking

Given the user-specific face template, the head pose is tracked in a video sequence by minimizing the point-to-plane ICP registration function in eq. (4). The minimization is also implemented in an iterative process which consists of two major steps: (1) finding the matching points, and (2) calculating the rotation matrix $\mathbf{R}$ and the translation vector $\mathbf{t}$.

To find the matching points $\{\mathbf{f}_i, \mathbf{d}_i\}$s, we use perspective projection and local search as described in Section 3.2. And similarly, rejections are made based on Euclidean distances between the matching points, and the normal directions on the face template.

Furthermore, an additional rejection step is performed to handle outliers caused by, e.g., inter-object occlusions (hair, eyeglasses). To this end, we leverage the K-means clustering algorithm as follow

(1) For each $\mathbf{d}_i$, generate a six dimensional sample, $(\mathbf{r}_i, \mathbf{g}_i, \mathbf{b}_i, \gamma\mathbf{x}_i, \gamma\mathbf{y}_i, \gamma\mathbf{z}_i)$, where $\{\mathbf{r}_i, \mathbf{g}_i, \mathbf{b}_i\}$ are the three color values of $\mathbf{d}_i$ (between 0 and 255), and $\{\mathbf{x}_i, \mathbf{y}_i, \mathbf{z}_i\}$ are the x, y, z coordinates of $\mathbf{d}_i$ (in unit of mm), and γ is a weighting factor, whose value is set to 0.2;
(2) Use the K-means clustering algorithm [22] to get five clusters of samples in this six-dimensional space;
(3) For each cluster $c \in \{1, 2, \ldots 5\}$, calculate the average RGB values (r_c, b_c, g_c) of all its samples, and transform this triplet into the Lab color space (L_c, a_c, b_c). Compare the Lab centers (L_c, a_c, b_c)s of the five clusters. Find the cluster whose center is most distinctive among the five, by comparing the sum of Euclidean distances from the other centers in the Lab color space. Then reject the matching points $\{\mathbf{f}_i, \mathbf{d}_i\}$s whose $\mathbf{d}_i$s are used to generate this cluster.

The assumption is that the color appearance of the outliers is distinguishable from the face, which matches empirical observations well. A video demonstrating the robustness of our method to outliers can be found in [18]. Notice that in many cases, the use of a distance threshold can eliminate most outliers already. This is one of the many benefits from using the depth data. Also notice that a more comprehensive approach for color-based outlier rejection is to use Lab values rather than RGB values in step 1, since the Euclidean distance between Lab triplets can better represent the perceived color difference. The current implementation balances the time efficiency against accuracy. In addition, to further reduce the computational complexity, the K-means clustering is performed only once. The cluster centers are re-used in the following iterations of the ICP registration.

To calculate $\mathbf{R}$ and $\mathbf{t}$, the closed form solution of eq. (4) is used. It is noticed that the ICP registration can be easily trapped into a local minimum. Therefore, a good initialization of $\mathbf{R}$ and $\mathbf{t}$ should be provided. When head poses in the previous video frames are tracked successfully, they are used to predict the head pose in the current video frame, using multiple linear head pose predictors which are jointly trained using real-world head movement data. More details on the head pose prediction algorithm can be found in our previous work [12]. For the first frame of the video sequence, or when the tracking is lost (i.e., the located minimum of eq. (4) is larger than a threshold), we use a color-based face detector [16] to initialize $\mathbf{R}$ and $\mathbf{t}$. Specifically, we assume that the nose tip is the closest point to the camera in the face region, and the user's head faces towards the camera center.

TABLE 1
BIWI and ICT-3DPH Head Pose Databases

	Sequences	Frames	Subjects	Rotation range
BIWI	24	~15 K	14 males 6 females	~+-75° yaw ~+-60° pitch
ICT-3DHP	10	~14 K	6 males 4 females	~+-75° yaw ~+-45° pitch

5 Results

5.1 Databases

To quantitatively evaluate the head pose tracking accuracy, the BIWI [19] and ICT-3DHP [20] Kinect head pose databases are used. The general information is summarized in Table 1.

The BIWI head pose database contains 24 video sequences with around 15 thousand frames. Each video was recorded with a subject turning his/her head around in front of a Kinect camera (version 1) at about one meter distance. 20 subjects participated including 14 males and 6 females. The ground truth for the head pose of each frame was obtained using a face tracking software FACESHIFT [7], and provided in the form of 3D location of the head center and the rotation angles. Both color and depth images are available for each video frame, and camera parameters are given so that the color and depth registration can be performed. Since the BIWI database was not developed intentionally for tracking, 11 of the 24 sequences were not recorded continuously. On the other hand, all the 10 video sequences in the ICT-3DHP head pose database are sequential without missing frames. There are 10 subjects involved including six males and four females. Each subject was recorded sitting in front of a Kinect camera (version 1) at about one meter distance. Registered color and depth images are provided for each video frame. The ground truth was collected using the Polhemus Fastrack flock of birds tracker which is attached to a cap worn by each subject.

5.2 Quantitative and Qualitative Evaluation

Since the head center of the BFM face model is different from the one defined in either of the two databases and the displacement between the different head centers is unknown, like in the prior study [3], we only evaluate the head rotation accuracy.

Fig. 4 shows the success ratio, i.e., the percentage of correctly estimated images, depending on the angular error thresholds for both BIWI and ICT-3DHP databases. The angular error is calculated by L2 norm of the three-dimentional Euler rotation difference (including yaw, pitch and roll) between the estimated head rotation and the ground truth. The red and green lines represent performances of [1] and ours, respectively. On both databases, our method outperforms [1] by a large margin. But it should be noted that the method in [1] estimates the head pose using only the current frame, while our method needs to use the tracking results of the previous frames. In fact, we can use [1] to replace the color-based face detector [16] for initialization when the tracking loss occurs. As discussed in Section 5.1, near half of the sequences in BIWI have missing frames, which will unavoidably cause tracking

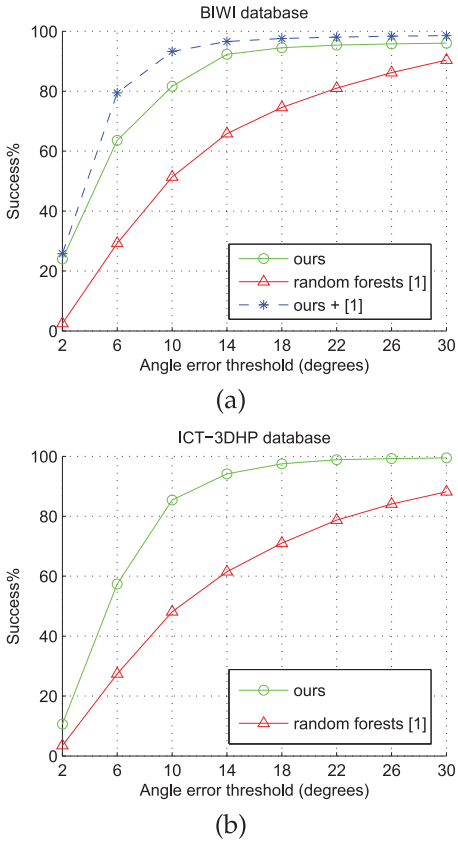

Fig. 4. Success ratio depending on the angular error thresholds for (a) BIWI and (b) ICT-3DHP. The red lines represent the performance of the random forests [1]. The green lines represent the performance of our method. The blue dashed line in (a) represents the performance of our method which replaces the face detection [16] with [1] for head pose initialization.

loss. Since we use a frontal face detector for initialization, once tracking loss happens, it may need a long time before the next frontal face is detected, during which the tracking accuracy will dramatically drops. On the other hand, the method in [1] can provide a coarse head pose estimation even if the frame contains a profile face. It explains the superior performance of our tracking method using [1] for initialization (denoted as "ours + [1]"), as shown in Fig. 4a. Notably, only 0.8 percent of the 15,699 frames in BIWI uses [1] for initialization, while the other frames are tracked using ICP. Combining ICP and detection based method like [1] can take merits from both approaches, specifically, accuracy, time-efficiency, occlusion robustness from ICP, and large rotation robustness from [1]. Note that the curve for "ours + [1]" is not plotted in Fig. 4b. Because our method generate no tracking loss on the ICP-3DHP database, it has the same performance as "ours + [1]". Furthermore, it should be emphasized that, if using an angular error threshold of 15 degree (as in [1]), the target loss rates of our method on the BIWI and ICT databases are 6.9 and 4.4 percent , respectively, in comparison to 31.6 and 35.9 percent of [1]. If using [1] to replace [16] as the initialization method, the target loss rate on BIWI can be further reduced to 3.1 percent.

TABLE 2
Mean Rotation Errors (in Unit of Degree) for Yaw, Pitch and Roll on the BIWI Database

	Yaw	Pitch	Roll
RF [2]	9.2	8.5	8.0
RF + Tensor [17]	5.0	7.4	6.6
CLM-Z [3]	14.8	12.0	23.2
CLM-Z [3] + GAVAM [23]	6.9	5.1	11.3
Ours	3.0	3.2	5.3
Ours + [1]	**2.2**	**1.7**	**3.2**

All results except for ours and ours + [1] are cited from [3] and [17].

TABLE 3
Mean Rotation Errors (in Unit of Degree) for Yaw, Pitch and Roll on the ICT-3DHP Database

	Yaw	Pitch	Roll
RF [2]	7.2	9.4	7.5
GAVAM [23]	3.0	3.5	3.5
CLM [10]	11.1	9.9	7.3
CLM-Z [3]	6.9	7.1	10.5
CLM-Z [3] + GAVAM [23]	**2.9**	**3.1**	3.2
Ours	3.3	**3.1**	**2.9**

All results except for ours are cited from [3].

Tables 2 and 3 list the mean head rotation errors for yaw, pitch and roll on the BIWI and ICT-3DHP databases, respectively. Among the methods in comparison, [2] and [17] are detection-based, while the others are tracking-based methods like ours. All results except for ours are directly cited from [3] and [17]. On the BIWI database, "ours + [1]" can achieve the state-of-the-art performance. Even without using [1] for initialization, our method still outperforms the state-of-the-art methods by a large margin. On the ICP-3DHP database, the yaw accuracy of our method is slightly lower than GAVAM [23] and CLM-Z [3] + GAVAM [23]. However, according to [23], GAVAM runs at 12 Hz on the Intel X535 Quad-core processor. Our method is much more time-efficient as discussed in Section 5.4.

According to our experiments with different subjects, the proposed method (without using [1]) can robustly track rotation angles within +-50° for yaw, row and pitch; a rotation speed around 4 degree per frame (120 degree per second); and a translational speed about 1 cm per frame (0.3 meter per second). Demo videos of the field tests can be found in our project website [18]. It can also be seen from these videos that our method is able to handle large occlusions and can recover from the tracking failure quickly.

5.3 Initial Error versus Estimation Error

Using the head pose in the n-1th frame to calculate the initial error, we plotted in Fig. 5a the relationship between the initial error and the estimation (final) error of the nth frame on the BIWI database (similar for ICT-3DHP), where the error is calculated as the L2 norm of the three-dimensional Euler rotation difference. The dotted lines indicate the standard deviation of the estimation error. As shown by the blue line which represents the result of frames containing mainly frontal faces (head rotation angle $< 20^o$), the estimation error linearly increases to around 4 degree, and then keeps roughly the same as the initial error goes to 9 degree, and surges up quickly thereafter. The shape of this curve can be explained by the three states of the ICP registration process: early-terminated, convergent and divergent, under subtle, moderate and fast head movements, respectively.

It can also be observed from Fig. 5a that the relationship between the initial error and the estimation error is greatly affected by the head rotation angle: the convergence range diminishes gradually when more frames with larger head rotation angles are included, as shown by the magenta and cyan curves in Fig. 5a. In other words, the performance of our method drops as the head rotation angle increases.

To see this more clearly, Fig. 5b shows the estimation errors computed for local angle ranges (15° pitch × 15° yaw) on the BIWI database (similar for ICT-3DHP). Apparently there is a trend that the performance of our method degrades for larger yaw and pitch

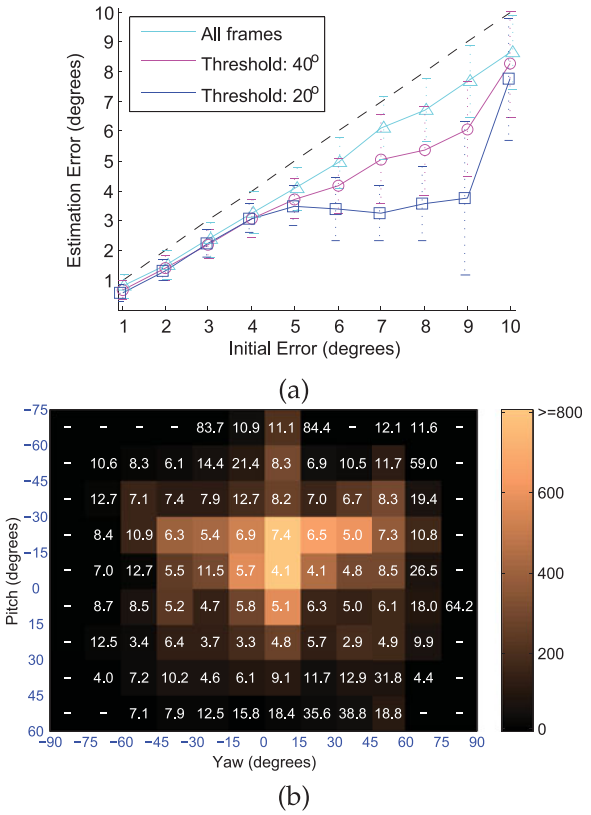

Fig. 5. (a) A plot of initial head pose errors versus estimation head pose errors on BIWI. Frames with head rotation angles under a specified threshold value are used to plot the corresponding lines. (b) Estimation error computed for local angle ranges (15° pitch × 15° yaw) on BIWI. The color encodes the number of images (side bar) falling into each region.

rotations. The reason is that as the rotation angles increases, less geometrical information can be used to determine the head pose, which is analogous to determining the 3D rotations of a globe using only its shape information.

5.4 Complexity

We empirically used 10 iterations for the optimization of eq. (2). On the Intel Core i7 processor at 2.93 GHz, the BFM model fitting time for each frame takes about 0.9 seconds, and there are up to nine frames used in the face template reconstruction process. The short reconstruction time benefits mostly from the time-efficient solution in each processing step as described in Section 3.2.

Including the time for the K-means clustering, head pose prediction and ICP registration (typically two or three iterations to converge), the head pose tracking needs around 21 ms to process one frame, or equivalently has a processing speed of around 47 fps.

5.5 Discussion

The advantage of using depth and 3D (shape) template in tracking comes from many aspects. While tracking methods using 2D (color) template usually suffers from illumination and viewpoint changes [24], the use of depth makes our method immune to illumination changes. Using a 3D template, the object pose can be explicitly formulated in our problem, making our method more robust to large pose/viewpoint changes than the 2D methods. Furthermore, using depth we can time-efficiently reject outliers by setting a distance threshold, as discussed in Section 4. To further reduce the influence of the remaining outliers, we can use a robust error metric as discussed in [10] to replace the quadratic one. However, how to balance the outlier robustness and the time complexity needs future investigation.

In order to achieve high time efficiency, we use the color data in face-detector-based initialization, landmark detection, and K-means clustering, but not in the main tracking process. Possible ways to fully exploit the color data for tracking include (1) using both color and depth to determine the matching points in the ICP registration, and (2) performing sparse feature matching or optical flow to find correspondences across adjacent frames. Furthermore, by using a skin-color detector to identity outliers, we can improve the outlier rejection algorithm as discussed in Section 4 in extreme cases, e.g., when the outliers are quite close to the face and occlude over 3/4 of the face region.

6 Conclusion

Using a RGBD camera, a head pose tracking method is proposed which can reconstruct a user-specific face template on-the-fly. During the real-time tracking, up to nine video frames are automatically selected for the reconstruction to handle problems such as the face self-occlusion and the depth noise. The BFM face model is fitted to each frame, which is formulated as an iterative optimization process. The optimization process consists of three processing steps, each of which has a time-efficient solution. Using the proposed approach, model fitting results from multiple frames can be fused together in one step. The face template reconstruction is sequential (frame-by-frame), so the head pose tracking process suffers less from the reconstruction delay. The head pose tracking is implemented using the ICP algorithm which registers the user-specific face template to the input depth video. A K-means clustering process is proposed to reject outliers based on colors which was found to be useful in practice. The proposed head pose tracking method has a low complexity and can achieve the state-of-the-art performances on two public head pose datasets.

Acknowledgments

This work was partially supported by the Hong Kong Innovation and Technology Commission (Project ITS/070/13) and the University of Malaya, Malaysia (Project UM.C/625/1/HIR/MOHE/ENG/42).

References

[1] G. Fanelli, M. Dantone, J. Gall, A. Fossati, and L. V. Gool, "Random forests for real time 3D face analysis," *Int. J. Comput. Vis.*, vol. 101, no. 3, pp. 437–458, 2013.
[2] G. Fanelli, T. Weise, J. Gall, and L. V. Gool, "Real time head pose estimation from consumer depth cameras," in *Proc. 33rd Int. Conf. Pattern Recog.*, 2011, pp. 101–110.
[3] T. Baltruaitis, P. Robinson, and L. P. Morency, "3D constrained local model for rigid and non-rigid facial tracking," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2012 pp. 2610–2617.
[4] Q. Cai, D. Gallup, C. Zhang, and Z. Zhang, "3D deformable face tracking with a commodity depth camera," in *Proc. 11th Eur. Conf. Comput. Vis.*, 2010, pp. 229–242.
[5] M. D. Breitenstein, D. Kuettel, T. Weise, L. V. Gool, and H. Pfister, "Real-time face pose estimation from single range images," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2008, pp. 1–8.
[6] P. Padeleris, X. Zabulis, and A. A. Argyros, "Head pose estimation on depth data based on particle swarm optimization," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recog. Workshops*, 2012, pp. 42–49.
[7] T. Weise, S. Bouaziz, H. Li, and M. Pauly, "Realtime performance-based facial animation," *ACM Trans. Graph.* vol. 30, no. 4, pp. 77:1–77:9, 2011.
[8] T. Bar, J. F. Reuter, and J. M. Zollner, "Driver head pose and gaze estimation based on multi-template ICP 3-D point cloud alignment," in *Proc. IEEE Conf. Intell. Transp. Syst.*, 2012, pp. 1797–1802.
[9] E. Murphy-Chutorian, and M. M. Trivedi, "Head pose estimation in computer vision: A survey," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 31, no. 4, pp. 607–626, Apr. 2009.
[10] J. Saragih, S. Lucey, and J. Cohn, "Deformable model fitting by regularized landmark mean-shift," *Int. J. Comput. Vis.*, vol. 91, no. 2, pp. 200–215, 2011.
[11] M. Zollhfer, M. Martinek, G. Greiner, M. Stamminger, and J. Smuth, "Automatic reconstruction of personalized avatars from 3D face scans," *Comput. Animation Virtual Worlds*, vol. 22, no. 2–3, pp. 195–202, 2011.
[12] S. Li, K. N. Ngan, and L. Sheng, ," A head pose tracking system using RGB-D camera," in *Proc. Comput. Vis. Syst.*, 2013, pp. 153–162.

[13] P. Paysan, R. Knothe, B. Amberg, S. Romdhani, and T. Vetter, "A 3D face model for pose and illumination invariant face recognition," in *Proc. IEEE Int. Conf. Adv. Video Signal Based Surveillance*, 2009, pp. 296–301.
[14] X. Xiong and F. De la Torre, "Supervised descent method and its applications to face alignment," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2013, pp. 532–539.
[15] P. Besl and H. McKay, "A method for registration of 3-d shapes," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 14, no. 2, pp. 239–256, Feb. 1992.
[16] P. Viola and M. J. Jones, "Robust real-time face detection," *Int. J. Comput. Vis.*, vol. 57, no. 2, pp. 137–154, 2004.
[17] S. Kaymak and I. Patras, "Exploiting depth and intensity information for head pose estimation with random forests and tensor models," in *Proc. Comput. Vis. Workshops*, 2013, pp. 160–170.
[18] Project website [Online], (2015). Available: http://www.ee.cuhk.edu.hk/snli/HeadTracking2.htm
[19] BIWI Kinect head pose database [Online], (2011). Available: http://www.vision.ee.ethz.ch/ gfanelli/head_pose/head_forest.html#db
[20] ICT-3DHP Kinect head pose database [Online], (2012). Available: http://projects.ict.usc.edu/3dhp/
[21] S. Izadi, D. Kim, O. Hilliges, D. Molyneaux, R. Newcombe, P. Kohli, J. Shotton, S. Hodges, D. Freeman, A. Davison, and A. Fitzgibbon, "Kinectfusion: Real-time 3D reconstruction and interaction using a moving depth camera," in *Proc. 24th Annu. ACM Symp. User Interface Softw. Technol.*, 2011, pp. 559–568.
[22] J. Hartigan, *Clustering Algorithms*. Hoboken, NJ, USA: Wiley, 1975.
[23] L. P. Morency, J. Whitehill, and J. R. Movellan, "Generalized adaptive view based appearance model: Integrated framework for monocular head pose estimation," in *Proc. IEEE Int. Conf. Autom. Face Gesture Recog.*, 2008, pp. 1–8.
[24] I. Matthews, T. Ishikawa, and S. Baker, "The template problem," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 26, no. 6, pp. 810–815, Jun. 2004.