Robust Model-based 3D Head Pose Estimation

Gregory P. Meyer[1,2] Shalini Gupta[2] Iuri Frosio[2] Dikpal Reddy[2] Jan Kautz[2]
[1]University of Illinois Urbana-Champaign [2]NVIDIA

Abstract

We introduce a method for accurate three dimensional head pose estimation using a commodity depth camera. We perform pose estimation by registering a morphable face model to the measured depth data, using a combination of particle swarm optimization (PSO) and the iterative closest point (ICP) algorithm, which minimizes a cost function that includes a 3D registration and a 2D overlap term. The pose is estimated on the fly without requiring an explicit initialization or training phase. Our method handles large pose angles and partial occlusions by dynamically adapting to the reliable visible parts of the face. It is robust and generalizes to different depth sensors without modification. On the Biwi Kinect dataset, we achieve best-in-class performance, with average angular errors of 2.1, 2.1 and 2.4 degrees for yaw, pitch, and roll, respectively, and an average translational error of 5.9 mm, while running at 6 fps on a graphics processing unit.

1. Introduction

Estimating the three dimensional (3D) pose (rotation and position) of the head is an important problem with applications in facial motion capture, human-computer interaction and video conferencing. It is a pre-requisite to gaze tracking, face recognition, and facial expression analysis. Head pose estimation has traditionally been performed on RGB images with rotation-specific classifiers or facial features [21], or by registering images to 3D templates [30, 9]. However, RGB-based head pose estimation is difficult when illumination variations, shadows, and occlusions are present. With the emergence of inexpensive commodity depth cameras, promising 3D techniques for body [29], hand [22], and head [15] pose estimation have been proposed.

We present an algorithm for accurate 3D head pose estimation for data acquired with commodity depth cameras. Our approach uses only 3D information, no manual intervention or training, and generalizes well to different 3D sensors. On the benchmark Biwi Kinect dataset [15], we achieve average angular errors of 2.1°, 2.1° and 2.4° for yaw, pitch, and roll, respectively, and an average translational error of 5.9 mm, while running at 6 fps on a graphics processing unit (GPU). To our knowledge, this is the best accuracy reported on this dataset up to now.

We achieve this high accuracy by combining a number of concepts together in an effective manner. We first detect the head using an adaptive 3D matched filter. Then, we register a morphable face model [6] to the measured facial data through a combination of particle swarm optimization (PSO) and the iterative closest point (ICP) algorithm. We demonstrate that together PSO and ICP simultaneously improve robustness, accuracy, and computational efficiency. Instead of creating a person-specific model during an initialization phase, we continuously adapt a morphable model to fit the subject's face on the fly. Additionally, we dynamically weight the vertices of the morphable model to give more importance to the useful visible parts of the face, and thus handle extreme poses and partial occlusions effectively.

2. Related Work

Notable techniques for 3D head pose estimation employ features, pose-specific classifiers, or registration to reference 3D head models.

Sun and Yin locate facial features using curvature properties to infer the head pose to within 5° [31]. Breitenstein et al. use the orientation of the nose as an initial estimate for head pose and refine it by comparing against pre-rendered depth images of an average face model in various poses 6° apart [7]. Papazov et al. introduce a triangular surface patch (TSP) descriptor to match facial point clouds to a gallery of synthetic faces and to infer their pose [24]. Feature-based techniques fail when the features cannot be detected, *e.g.* in the case of extreme rotations or partial occlusions.

Among the classifier-based techniques is the work of Seemann et al. where they detect faces in RGB images and estimate the head pose from the disparity map of a stereo camera using a neural network for each rotation [28]. Fanelli et al. train random classification and regression forests with range image patches for head detection and pose estimation [15]. Their technique achieves good accuracy on high and low quality depth data [14, 15]. Tulyakov et al. [32] use cascaded tree classifiers and achieve higher accuracies than Fanelli et al. Classifier-based techniques require extensive

training with large datasets. Moreover, classifiers trained on one 3D sensor do not generalize well to others.

An alternate approach registers a 3D head model to the measured data using the rigid/non-rigid ICP algorithm. Previous promising methods, *e.g.* [2, 26, 8, 10], employ 3D deformable model fitting to create person-specific models for head pose estimation. However, these existing methods require offline initialization with significant cooperation from the user to construct the subject-specific reference models. In contrast, we refine the morphable model's shape continuously to fit the subject while simultaneously estimating the head pose.

To ensure robustness to facial expressions, a number of the existing deformable model fitting based approaches, *e.g.* [35, 26], include only the less deformable eyes and nose regions of the face in the reference model. However, when the parts of the face that are included in the reference model are not visible, *e.g.* when the head is titled back the eyes and nose regions are not visible, the reference model matches poorly to the observed data, resulting in inaccurate pose estimation. In order to address this, we instead employ the entire morphable model and dynamically weight the regions of the model based on which parts of the face are visible.

Techniques for facial animation capture with high [38, 34, 37] and low [35, 18] quality 3D scans, also employ very precise morphable model fitting. These techniques require significant manual interaction to create very detailed person-specific models. Also, these studies do not directly report the accuracy of head pose estimation, but presumably perform sufficiently well to enable effective facial expression capture.

To avoid deformable model fitting, some methods directly use facial data from the 3D video sequence as a reference. For example, Padeleris et al. use the first frame [23], Bar et al. use frontal, left, and right profile faces [3], and Martin et al. employ 100 frames with faces in different poses [20]. When the absolute pose of the reference face is unknown, these techniques merely provide the pose of the face relative to the reference and not the absolute head pose. Furthermore, the quality of these references is low, as they often contain holes and noise. Generally, including more views of the face in the reference model, tends to improve accuracy. Tulyakov et al. achieve the best accuracy among these approaches by registering multiple frames and volumetrically averaging them to produce a higher quality reference model [32].

To register the reference model to the measured data, ICP and its variants are often used. However, ICP fails to converge to the correct solution when it is initialized poorly. To overcome this, Padeleris et al. employ the stochastic PSO algorithm [1] to register facial surfaces [23]. However, PSO also suffers from slow and/or premature convergence to a local optimum. Recently, Qian et al. proposed an optimization algorithm that combines PSO and ICP to overcome their individual limitations and successfully applied it to estimate the 26-dimensional pose of 3D hands [25]. Their work has inspired us to employ a combination of PSO and ICP to accurately estimate the 3D head pose. As an extension of their work, we provide a detailed analysis to understand the underpinnings and conditions for success of combining PSO and ICP for 3D surface registration.

Finally, 3D head detection is a pre-requisite to head pose estimation. When color and depth information is available, head detection is typically performed via face detection [33] in RGB, *e.g.* in [8, 26, 20, 28, 2]. Fanelli et al. [15] and Tulyakov et al. [32] trained classifiers to distinguish between face and non-face range image patches. However, these methods have limited reliability, and they do not generalize well to other depth sensors.

3. Method

3.1. Head Localization

The 3D head localization procedure identifies an area in a depth image which most likely contains a head. It employs an adaptive detection filter, whose size changes to match the expected size of an average human head at various depths.

The depth measured at pixel (i, j) is denoted by $d_o(i, j)$. We assume that the camera's focal length f is known and that only one subject is present in the scene. Furthermore, we assume that the subject is at a depth between d_m and d_M and that their silhouette can be reliably extracted, *e.g.* through thresholding. We define a binary mask

$$\varepsilon(i, j) = d_m < d_0(i, j) < d_M, \tag{1}$$

that identifies the pixels that are inside the boundary of the subject's silhouette, *i.e.* $\varepsilon(i, j) > 0$, which we refer to as *active* pixels. The expected pixel width $w(i, j)$ and height $h(i, j)$, of a head centered at (i, j), are obtained as:

$$w(i, j) = f\bar{w}/d_o(i, j), \quad h(i, j) = f\bar{h}/d_o(i, j) \tag{2}$$

where $\bar{w}$ and $\bar{h}$ are the width and height of an average human head, respectively [12].

For each active pixel location (i, j), we first resize the kernel in Fig. 1b relative to the approximated width, $w(i, j)$, and height, $h(i, j)$, of the head, and convolve it with $\varepsilon(\cdot, \cdot)$ to obtain a score $s(i, j)$. The head detection kernel (Fig. 1b) resembles the silhouette of a subject's head and shoulders, when their torso is roughly perpendicular to the optical axis of the camera.

The identified head region is centered on the pixel with the maximum score, (i_h, j_h), and has a size of $w(i_h, j_h) \times h(i_h, j_h)$ (red rectangle in Fig. 1d). We provide to the head pose estimation algorithm a slightly enlarged region around the detected head (shown in yellow in Fig. 1d) to ensure that the head is entirely contained in it. Since our head detection filter requires only the sums of rectangular regions of the image, we compute $s(i, j)$ efficiently using integral images.

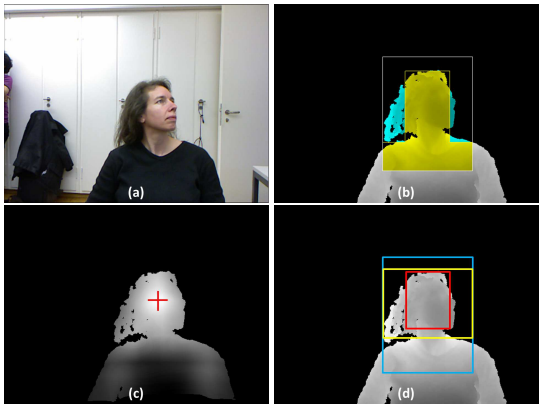

Figure 1. (a) A color image from the Biwi Kinect dataset [15]. (b) The corresponding depth map and the kernel centered on the identified head area; the yellow pixels have coefficients +1, and the cyan ones have a value of -1. (c) The score map $s(i,j)$; the red cross indicates the location (i_h, j_h) of the maximum of the score map. (d) The identified head area (red rectangle), enlarged head area (yellow rectangle), and kernel area (in blue).

3.2. Head pose estimation

3.2.1 Reference model

As the reference head model, we use the 3D Basel Face Model [6]. With this morphable model, a facial surface comprised of a set of 3D vertices $\mathbf{S} = (\mathbf{v}_1, \mathbf{v}_2, ..., \mathbf{v}_N)$ can be represented as a linear combination of an average 3D face (μ) and a set of 3D bases face shapes ($\mathbf{s}_i$):

$$\mathbf{S} = \mu + \sum_i \alpha_i \mathbf{s}_i. \tag{3}$$

Parts of the observed face may not match the morphable model (*e.g.*, due to facial hair, or eye-wear); therefore, we use a weight vector, $\mathbf{W} = (w_1, w_2, ..., w_N)$, to represent the confidence of each vertex in the reference model. For the initial frame, we set the morphable model's shape vector $\mathbf{S}_0$ to the average face μ and its weight vector $\mathbf{W}_0$ to unity for all vertices.

3.2.2 Cost function

The pose of the head is indicated by a 6-dimensional vector $\mathbf{x} = (\theta_x, \theta_y, \theta_z, t_x, t_y, t_z)$, where θ_i and t_i represent a rotation about and a translation along the axis i. We evaluate a hypothetical pose $\mathbf{x}$ for an observed depth image d_o by first rendering a depth image d_h and a weight image w_h of the morphable face model in the pose $\mathbf{x}$:

$$(d_h, w_h) = \text{Render}(\mathbf{x}, \mathbf{S}_k, \mathbf{W}_k, \mathbf{K}), \tag{4}$$

where $\mathbf{S}_k$ and $\mathbf{W}_k$ are the current shape and weight of the morphable model, and $\mathbf{K}$ is the camera's intrinsic calibration matrix. Each depth pixel at location (i,j) in d_o and d_h has a corresponding 3D vertex, $\mathbf{v}_o(i,j)$ and $\mathbf{v}_h(i,j)$, respectively. In addition, each vertex in $\mathbf{v}_h(i,j)$ has a normal vector $\mathbf{n}_h(i,j)$ computed using the relative position of its neighboring vertices.

To factor out the effect of outliers, which are commonly observed with low-cost depth cameras, we generate a subset, $\mathcal{P}$, of reliable vertices to be compared:

$$\mathcal{P} = \left\{(i,j) \middle| \, \|\mathbf{v}_o(i,j) - \mathbf{v}_h(i,j)\| < \tau, (i,j) \in \mathcal{O} \cap \mathcal{H}\right\}, \tag{5}$$

where $\mathcal{O}$ and $\mathcal{H}$ are the sets of valid (non-zero) pixels in the observed and hypothetical depth images, respectively. In our experiments, we empirically set $\tau = 3$ cm.

We then compute the following cost function to quantify the discrepancy between the observed and the hypothetical data:

$$E(\mathbf{x}) = E_v(\mathbf{x}) + \lambda E_c(\mathbf{x}) \tag{6}$$

where

$$E_v(\mathbf{x}) = \frac{\sum_{(i,j)\in\mathcal{P}} w_h(i,j) \left[(\mathbf{v}_o(i,j) - \mathbf{v}_h(i,j))^T \mathbf{n}_h(i,j)\right]^2}{\sum_{(i,j)\in\mathcal{P}} w_h(i,j)} \tag{7}$$

and

$$E_c(\mathbf{x}) = \left[1 - \sum_{(i,j)\in\mathcal{P}} w_h(i,j) \Big/ \sum_{(i,j)\in\mathcal{H}} w_h(i,j)\right]^2. \tag{8}$$

The term $E_v(\mathbf{x})$ measures the point-to-plane distance between corresponding vertices on the two surfaces, whereas $E_c(\mathbf{x})$ measures the extent to which the depth images coincide with each other (*i.e.*, it penalizes the hypothetical and observed depth images for not overlapping). The parameter λ designates the relative importance of the two terms, and it was empirically set to 350.

3.2.3 Optimization

In order to compute the pose, we employ a combination of particle swarm optimization (PSO) and the iterative closest point (ICP) algorithms.

PSO [16] uses a set of particles, that evolve through social interactions over a series of generations, to search for a global optimum in a non-convex parameter space. For head pose estimaton, each particle represents a head pose $\mathbf{x}$ and has a corresponding cost, $E(\mathbf{x})$, specified by Eq. (6). Each particle keeps track of the position $\mathbf{x}^*$ where it has observed the lowest cost, $E(\mathbf{x}^*)$, across all generations. The best position across all particles and generations is indicated by $\mathbf{x}_g^*$. At generation t, every particle stochastically updates its position $\mathbf{x}$ and velocity $\mathbf{u}$ based on its position relative to $\mathbf{x}^*$ and $\mathbf{x}_g^*$ [11]:

$$\begin{aligned} \mathbf{u}_{t+1} &= \gamma\left(\mathbf{u}_t + \alpha\xi_1\left(\mathbf{x}^* - \mathbf{x}_t\right) + \beta\xi_2\left(\mathbf{x}_g^* - \mathbf{x}_t\right)\right) \\ \mathbf{x}_{t+1} &= \mathbf{x}_t + \mathbf{u}_{t+1}, \end{aligned} \tag{9}$$

where the constants α, β, and γ are the cognitive, social, and constriction factors, respectively, and ξ_1 and ξ_2 are uniform random variables $\in [0,1]$. Based on [11], we set $\alpha = \beta = 2.05$ and $\gamma = 0.7298$.

During the first generation ($t = 0$), the particles' positions are initialized randomly, and their velocities are set to zero. For the initial frame, the particles' positions are generated by randomly sampling a normal distribution with the mean set to the frontal pose. For subsequent frames, half of the particles are initialized in this way, and the other half use a normal distribution with a mean set to the previous frame's pose estimate.

To prevent unlikely head poses, we bound the parameter space: $\theta_x \in [-60°, 60°]$ for pitch, $\theta_y \in [-90°, 90°]$ for yaw, and $\theta_z \in [-45°, 45°]$ for roll. For translation, we force the centroid of the morphable model to remain within a certain distance ($\sim$ 10 cm) from the head center that we detect during head localization.

For each particle and for each generation, we run 3 iterations of ICP [4] before the PSO update, Eq. (9). To efficiently identify point correspondences between the surfaces, we use projective data association, which finds corresponding points along camera rays [5]. Given the particle's current estimate of the head pose, we transform the vertices in the vertex map $\mathbf{v}_h$ and project them into the vertex map $\mathbf{v}_o$. Vertices in $\mathbf{v}_o$ and $\mathbf{v}_h$ that share the same pixel coordinate (i, j) and that are within a 3D Euclidean distance of 3 cm are considered corresponding points. We update the particle's position, $\mathbf{x}$, by minimizing the point-to-plane error metric, which has been shown to have improved convergence rates compared to the traditional point-to-point error metric [27], and it has a closed-form solution using the small angle approximation [19]. Note that the point-to-plane distance employed in ICP is also the first term in our cost function for PSO, Eq. (6).

As a trade-off between accuracy and computation time for our combined PSO and ICP optimization procedure, we used a set of 10 particles and 5 generations. After the optimization terminates, we provide the position of the best particle over all the generations, $\mathbf{x}_g^*$, as the final pose estimate for the face in the current frame.

3.2.4 Morphable model fitting

Once the head pose has been estimated, we update the shape and weights of the morphable model to match the observed face in the current frame. Utilizing the estimated pose $\mathbf{x}_g^*$, we identify point correspondences between the morphable face model and the observed data by transforming and projecting the vertices of the morphable model into the observed vertex map $\mathbf{v}_o$,

$$\begin{aligned} \begin{bmatrix} i & j & 1 \end{bmatrix}^T &= \mathbf{K}(\mathbf{R}\mathbf{v}_p + \mathbf{t}), \quad \mathbf{v}_p^o = \mathbf{v}_o(i,j), \\ m_p &= \begin{cases} 1 & \text{if } \left\| \mathbf{v}_p^o - (\mathbf{R}\mathbf{v}_p + \mathbf{t}) \right\| < \delta \\ 0 & \text{otherwise} \end{cases} \end{aligned} \tag{10}$$

where $\mathbf{R}$ and $\mathbf{t}$ are the rotation matrix and translation vector parameterized by $\mathbf{x}$, $\mathbf{v}_p$ is the p-th element in the morphable model's shape vector $\mathbf{S}_k$, $\mathbf{v}_p^o$ is a vertex in $\mathbf{v}_o$ which is the point corresponding to vertex $\mathbf{v}_p$, and $\delta = 1$ cm is a distance threshold for rejecting corresponding points that are too far apart.

We compute the new set of coefficients α^* of the morphable model by minimizing:

$$\alpha^* = \arg\min_{\alpha} \left\| \mathbf{M} \left(\left[\mu + \sum_i \alpha_i \mathbf{s}_i \right] - \mathbf{V} \right) \right\|^2, \tag{11}$$

where $\mathbf{V} = (\mathbf{v}_1^o, \mathbf{v}_2^o, ..., \mathbf{v}_N^o)$, and $\mathbf{M} = \text{diag}(m_1, m_2, ..., m_N)$. Afterwards, we updated the shape of the morphable model,

$$\mathbf{S}_{k+1} = \eta \left(\mu + \sum_i \alpha_i^* \mathbf{s}_i \right) + (1 - \eta)\mathbf{S}_k, \tag{12}$$

where $\eta = 0.1$ is a damping parameter introduced to prevent the shape from drastically changing between frames. In addition, we update the weights of the morphable model as:

$$w_p = \exp\left(-\left\| \mathbf{v}_p - \mathbf{v}_p^o \right\|^2 / \sigma_w \right), \tag{13}$$

where w_p and $\mathbf{v}_p$ are the p-th elements in the weight vector $\mathbf{W}_{k+1}$ and the shape vector $\mathbf{S}_{k+1}$, respectively, and $\sigma_w = 0.01$.

4. Results

We measured the performance of our method and compared it with state-of-the-art algorithms on two datasets. The Biwi Kinect Head Pose dataset, acquired with a Kinect sensor, contains over 15K RGB and depth images of 20 subjects recorded in 24 sessions [13]. It has large head rotations, long hair, and occlusions. For each frame, a ground truth binary mask of the face pixels, as well as, the 3D orientation of the head and the location of its center, are provided. On this dataset, we first coarsely locate the head using the method described in Sec. 3.1 and then estimate its pose.

The ETH Face Pose Range Image dataset by Breitenstein et al. contains 10K range images of 20 people [7]. These data are of higher quality than the Biwi Kinect data, and were acquired with a stereo enhanced structured light sensor [36]. In the ETH dataset, depth data is only available for the head region, thus we did not apply head localization on it.

4.1. Head localization

We evaluated the performance of our head localization method (Sec. 3.1) on the Biwi Kinect dataset [13]. Note that it provides only a coarse estimate of the 2D position of the head in depth images. We obtain a more precise estimate for the head center location after applying the head pose estimation procedure (Sec. 3.2).

The average distance of the centroid of the ground truth mask and the center of the head area identified by our algorithm was 26.1 pixels, with an approximate average face size of 90×120 pixels. The extended bounding box (in yellow in Fig. 1d), contained, on an average, 98.2% of the ground truth face pixels. For less than 1.0% of the the frames, the extended bounding box contained less than 50% of the ground truth head pixels. The score map $s(i, j)$ (Fig. 1c) was generally smooth and did not contain local maxima. Our method worked well for faces occluded by hair or when the subject's arms were raised around their head. It failed for only a few cases when the subject was looking down, and the shape of the adaptive filter did not match their silhouette. An optimized CUDA implementation of the head localization algorithm took less than 3 ms for a 640×480 sized image on a NVIDIA GeForce GTX 660 GPU.

These results suggest that although our head localization is not precise, it is fast and very reliable. By comparison, other approaches for 3D head detection, *e.g.* [15, 32] employ more detailed models and thus have a higher precision, but more missed detections. Note that for our algorithm, it is more important to reliably detect a face, since the precise 3D head location is estimated by the subsequent head pose estimation step.

4.2. Head pose estimation

Table 1 shows the average absolute errors for the yaw, pitch, and roll angles achieved by our algorithm on the Biwi Kinect dataset. It also contains the average positional errors for the head center and the accuracy of pose estimation. Accuracy is defined as the percentage of frames with an $L2$ norm of angular errors less than 10°.

On the Biwi Kinect dataset, we achieved angular errors of 2.1°, 2.1° and 2.4° for the yaw, pitch, and roll, a translational error of 5.9 mm and an accuracy of 94.6% with our proposed algorithm (row 1 in Table 1). In order to understand the contribution of each of the individual concepts employed in our algorithm, we additionally evaluated its performance with different configurations (Table 1).

Observe that between the morphable model (first row in Table 1) and the average face model (second row in Table 1), the morphable model consistently performed slightly better by allowing a better fit to each specific subject. Although, we did not fit the morphable model very precisely to the observed face. It seems that personalization of the face model is important for head pose estimation, but further investigation is required to establish this conclusively.

Next, we investigated the effect of combining PSO and ICP for optimization. Keeping all other parameters constant, we performed the optimization with 40 iterations of ICP only. In addition, we used PSO only with 25 particles and 40 generations. This configuration was previously shown to be effective [23]. Individually, PSO (fourth row in Table 1) performed the worst, considerably worse than ICP (third row in Table 1), plausibly because of PSO's slow convergence rate and susceptibility to premature convergence. ICP, by itself, performed better than PSO, but tended to fail for large angles of rotation. Lastly, analogous to what Qian et al. [25] observed for 3D hand pose estimation, we found that the combined PSO and ICP optimization method produced the most accurate results for 3D head pose estimation, as well.

The fifth row of Table 1 lists the performance of our algorithm for the case when we employed only the distance term E_v (Eq. 6) to measure the similarity between two 3D point clouds. On comparing these results with those in the first row of Table 1, where we used both the error terms in Eq. 6, it can be concluded that the 2D overlap term (E_c) helps to improve accuracy of head pose estimation considerably. Breintenstien et al. [7] made similar observations in their work where they found the overlap term to positively impact the accuracy of head pose estimation.

Lastly, the effect of dynamically re-weighting parts of the morphable model to best match the instantaneous appearance of the observed 3D face can be evaluated by comparing the first and sixth rows of Table 1. To obtain the values listed in the sixth row, we set all the weights of the morphable model to unity and kept then constant over time instead of dynamically varying them. Dynamically re-weighting the morphable model improves accuracy and helps to robustly handle partial occlusions of the face (*e.g.*, the first and sixth columns of Fig. 2). Recently, Tulyakov et al. [32] also achieved good pose estimation result by adopting a slightly different dynamic re-weighting scheme.

Our implementation of the proposed method runs in $\sim$ 160 ms on a NVIDIA GeForce GTX 660 GPU for a 640×480 sized frame, although it does not completely utilize the optimizations available in CUDA.

4.3. Comparison with existing methods

A number of recent algorithms [7, 15, 2, 23, 26, 20, 32, 24] for 3D head pose estimation have also been evaluated on the benchmark Biwi Kinect [15] and ETH [36] datasets. For all these methods, except for Padeleris et al.'s [23], we list the results reported by the authors in Tables 1 and 2, along with a summary of their methods. Padeleris et al. report average errors for only 91.4% of the frames on which their algorithm succeeded. Furthermore, when their algorithm failed, they re-initialized head pose tracking with the ground truth pose (personal communications with the authors). For

Method	Model	Weights	Optim.	Cost function	Errors				Accuracy
					Yaw [°]	Pitch [°]	Roll [°]	Location [mm]	
Proposed*	Morph	Dyn	PSO+ICP	All terms	2.1	2.1	2.4	5.9	94.6%
Proposed	Average	Dyn	PSO+ICP	All terms	2.2	2.2	2.6	6.2	92.3%
Proposed	Morph	Dyn	ICP	Distance	3.7	4.3	4.0	11.1	84.2%
Proposed	Morph	Dyn	PSO	All terms	7.7	8.4	6.0	23.3	73.7%
Proposed	Morph	Dyn	PSO+ICP	Distance	4.4	3.4	4.0	9.0	87.1%
Proposed	Morph	Fix	PSO+ICP	All terms	2.7	3.1	3.2	5.9	89.7%
Fanelli [15]	N/A	N/A	RF	N/A	8.9	8.5	7.9	14.0	79.0%
Tulyakov [32]	N/A	N/A	CT	N/A	4.7	7.6	5.3	9.1	-
Padeleris [23]	Frame 0	N/A	PSO	Distance	11.1	6.6	6.7	13.8	76.0%
Martin [20]	100 Frames	N/A	ICP	Distance	3.6	2.5	2.6	5.8	-
Rekik [26]	Morph	N/A	PF	Color + Distance	5.1	4.3	5.2	5.1	-
Baltrusaitis [2]	CLM-Z	N/A	RLMS	Color + Distance	6.3	5.1	11.3	7.56	-
Papazov [24]	TSP	N/A	N/A	TSP match	3.9	3.0	2.5	8.4	-

Table 1. The average absolute angular errors (in degrees) and the average translational error (in mm) on the Biwi Kinect dataset, for different configurations of our proposed algorithm, and for other existing state-of-the-art head pose estimation algorithms [15, 32, 23, 20, 26, 2, 24].

a fair comparison, we re-implemented their PSO-based algorithm and report its results for the entire Biwi Kinect dataset in Table 1.

On the Biwi Kinect dataset (Table 1), our proposed algorithm produced the lowest rotational errors, which were lower than those of the existing classifier-based [15, 32], rigid model-fitting-based [23, 20], non-rigid model-fitting-based [2, 26] and surface patch descriptor-based [24] approaches for head pose estimation. Despite using only the 3D information, our algorithm performed even better than the algorithms that employed both depth and color information [26, 2, 24].

We observed similar results on the ETH dataset (Table 2), where our algorithm outperformed the existing methods at estimating the yaw and pitch rotations (this dataset does not contain rotations about the roll axis). Since, we did not apply the head localization step (Sec. 3.1) on this dataset, these results demonstrate the superior performance of just our head pose estimation algorithm relative to the existing approaches, independently of head localization.

For all frames of the Biwi Kinect dataset, our algorithm also resulted in the smallest translational error (5.9 mm) of all the purely 3D-based head pose estimation algorithms (Table 1). Martin et al. report a similar translational error to ours (5.8 mm), but across only 97.6% of the frames on which their algorithm succeeded. Only Rekik et al.'s algorithm produced a lower translational error than ours (5.1 mm), but they used both color and depth data.

We also applied our head pose estimation algorithm to data acquired with a SoftKinetic DS325 time-of-flight camera by appropriately changing the depth de-noising parameters, the threshold τ in Eq. 5, and the camera's intrinsic parameters $\mathbf{K}$ in Eq. 4. We observed similar accuracy for head pose estimation with this sensor as well (Fig. 3a,b).

To further understand the effect of noise and resolution, we re-projected faces in the Biwi Kinect dataset, which are originally $\sim$ 1 m away from the sensor, to 1.5 m, 2 m and 2.5 m and added progressively increasing depth-dependent noise using the model proposed by Khoshelham and Elberink [17]. With increasing distance from the sensor our head pose estimation technique resulted in a linear increase in the yaw, pitch, and roll errors at a rate of 1.25°/m and a linear decrease in accuracy at a rate of 6.7%/m.

Method	Error		Accuracy
	Yaw [°]	Pitch [°]	
Breitenstein [7]	6.1	4.2	80.8%
Fanelli [14]	5.7	5.1	90.4%
Proposed	2.9	2.3	98.9%

Table 2. The average absolute angular errors for yaw and pitch, for the methods by Breitenstein et al. [7], Fanelli et al. [14], and for our method, on the ETH database [14]. No head localization was performed.

4.4. Combined PSO and ICP optimization

To better understand the peculiarities of the combined PSO and ICP algorithm, with different cost functions E and E_v, we considered the problem of registering (*i.e.* finding the optimal translation t_x) a 1D curve (in blue in Fig. 4a) with a set of sampled noisy points (in red in Fig. 4a). The first term of our cost function, $E_v(t_x)$, contains many local minima (Fig. 4b). Consequently ICP, which only optimizes $E_v(t_x)$, quickly converges toward one of these, depending upon the initialization. The trajectories for two different initializations are shown in red in Fig. 4b.

The overlap term $E_c(t_x)$ in our cost function, on the other hand, is quasi-convex (Fig. 4c). Although $E_v + E_c$ is still non-convex, adding E_c makes the global optimum more evident and local minima less pronounced (Fig. 4d); this explains the worse results reported in Table 1 (fifth row) for the use of E_v alone. Nonetheless, optimization with PSO

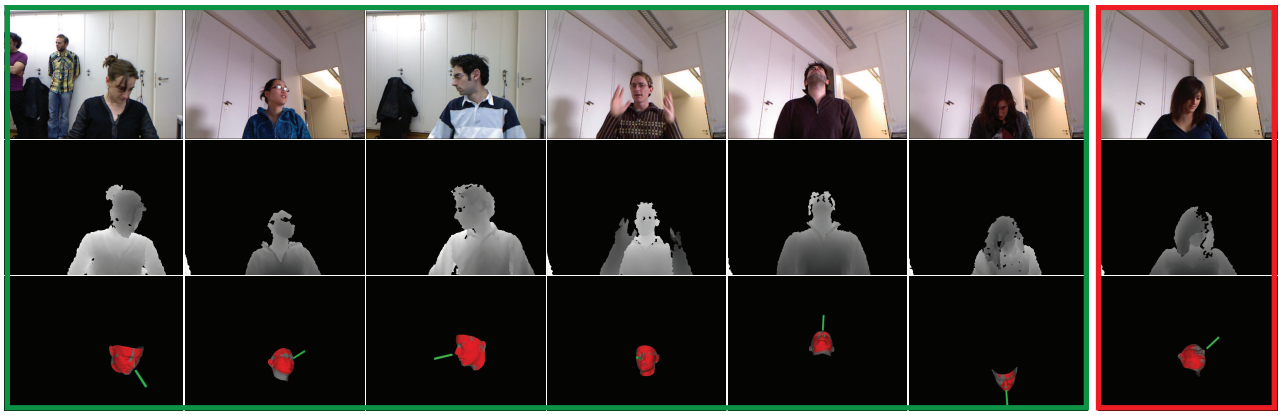

Figure 2. A set of RGB images (first row) and the corresponding depth images (second row) from the Biwi Kinect dataset [13]. The last row shows the head pose estimated by our method along with the dynamic weights assigned to different parts of the reference model. A failure case is shown in the rightmost column, where the subject's hair occludes a large part of her face.

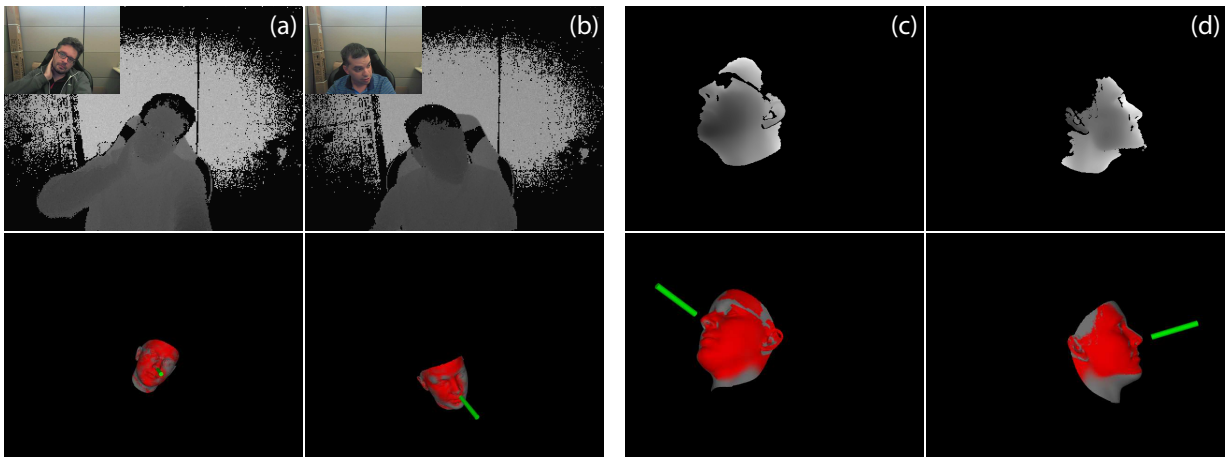

Figure 3. The left panels show images acquired by the SoftKinetic DS325 camera. The right panels show images from the ETH Face Pose Range Image dataset [7]. The head poses estimated by our algorithm are depicted in the bottom row (colored according to the adaptive weights).

remains problematic: particle 1 in Fig. 4d (on the right) moves towards particle 0 and misses the global optimum, leading to premature convergence into a local minimum of $E_v + E_c$. Additionally, since PSO randomly samples the cost function without using the gradient, convergence towards the global optimum is generally slow. This explains the poor accuracy for head pose estimation that we observed when we employed PSO only (Table 1).

Fig. 4e shows the trajectories of the combined PSO and ICP algorithm, where we apply ICP to each particle before the PSO update. ICP moves each particle to a local minimum of E_v (green triangle) thus making it less likely for PSO to skip over the basin of attraction of the global optimum. Note that, assuming that ICP converges, each particle is then constrained to lie in a local minimum of E_v. Since the local minima of E_v are generally only slightly offset with respect to the corresponding local minima of $E_v + E_c$, the combined optimization is generally more efficient and effective than PSO alone, as measured for our head pose estimation algorithm in Table 1. It is in fact sufficient for a particle to lie in the basin of attraction of the global optimum of E_v to quickly converge towards it (see the left particle in Fig. 4e). We also noted that the basins of attraction for E_v and $E_c + E_v$ are slightly different; as a consequence, ICP may contribute to move a particle out of a local basin of attraction of $E_c + E_v$ (see the right particle in Fig. 4f), thus potentially preventing premature convergence, increasing the overall mobility of the particles and favoring a wider exploration of the parameter space. On the other hand, we also noted that, because of ICP, the left particle in Fig. 4e oscillates around the same local minimum for two consecutive generations. This represents a potential drawback of any hybrid PSO algorithm, that can be mitigated by modifying the α, β and γ parameters in PSO.

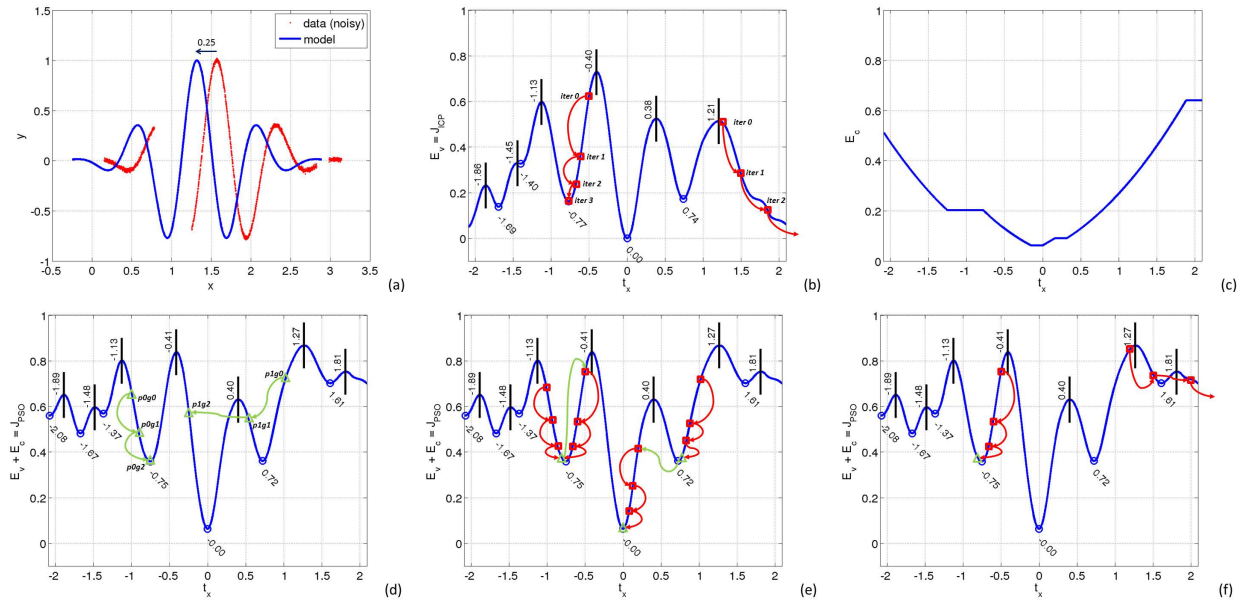

Figure 4. (a) A simple 1D registration problem: the reference model (shifted) and noisy sampled data. (b) ICP minimization of $E_v = E_v(t_x)$ (2 runs); t_x is the model shift along the x axis. Local minima are indicated by circles, the boundaries of the basins of attraction by vertical lines. (c) The overlap term $E_c = E_c(t_x)$. Optimization of $E_v(t_x) + E_c(t_x)$ with (d) PSO only, and with (e) the combined PSO and ICP algorithm. (f) ICP moves the left particle to the local minimum of E_v; the right one is moved away from the local basin of attraction of $E_v + E_c$.

5. Conclusion

We introduced a head pose estimation method for commodity depth cameras that results in best-in-class accuracy on benchmark datasets. It requires no initialization, handles extreme rotations and partial occlusions, and efficiently registers facial surfaces. Numerous factors contribute to the success of our algorithm: the overlap term (E_c) in the cost function, the combined PSO and ICP algorithm, dynamically adapting the weights of the face model, and the adoption of a morphable face model. While these concepts have each been introduced individually in previous studies, the contribution of our work lies in combining these disparate ideas in an effective manner to significantly improve the accuracy of 3D head pose estimation. Our work also presents for the first time a systematic quantitative assessment of the contribution of each of these various factors in improving the accuracy of head pose estimation. Building upon the work of Qian et al. [25] we also provide deeper insights into the workings of the combined PSO and ICP optimization for 3D surface registration.

To our knowledge, ours is also the first work to provide a comprehensive review and in-depth comparison of the existing state-of-the-art techniques for 3D head pose estimation on a common benchmark dataset. Our hope is that this work, besides advancing the-state-of-the-art in 3D head pose estimation, would also serve as a summary of the current understanding of the problem of 3D head pose estimation.

We see a number of directions along which we could extend our current work. Previous studies, *e.g.*, [2, 26] have shown that combining depth and color data improves the accuracy of head pose estimation. Relying on both the depth and color data can also help to extend the operating range of head pose estimation algorithms to illumination conditions, including bright sunlight where commodity depth sensors based on active infrared illumination fail. An obvious question then would be how best to combine information from the RGB images with 3D data with our current algorithm. Several authors have also noted the value of introducing temporal tracking mechanisms for head pose estimation in conjunction with per-frame head pose estimation [2, 8, 32, 26]. This can provide smoother head motion trajectories and the ability to predict the head pose in future frames. In our current algorithm, we use minimal temporal tracking, and hence this would be an interesting direction to explore in the future. While our study provides some evidence for the value of non-rigidly warping the reference model to the measured face, the issue is still worth exploring in depth in the future. Finally, a further analysis of the interaction between PSO, ICP and their cost functions may help to combine them even more effectively in the future.

References

[1] P. J. Angeline. Evolutionary optimization versus particle swarm optimization: Philosophy and performance differences. In *Evolutionary Programming VII*, pages 601–610, 1998. 2

[2] T. Baltrusaitis, P. Robinson, and L. Morency. 3d constrained local model for rigid and non-rigid facial tracking. In *CVPR*, pages 2610–2617, 2012. 2, 5, 6, 8

[3] T. Bar, J. F. Reuter, and J. Zollner. Driver head pose and gaze estimation based on multi-template icp 3-d point cloud alignment. In *ITS*, pages 1797–1802, 2012. 2

[4] P. J. Besl and N. D. McKay. Method for registration of 3-d shapes. In *Robotics-DL*, pages 586–606, 1992. 4

[5] G. Blais and M. D. Levine. Registering multiview range data to create 3d computer objects. *TPAMI*, 17(8):820–824, 1995. 4

[6] V. Blanz and T. Vetter. A morphable model for the synthesis of 3d faces. In *Conf. on Computer Graphics and Interactive Techniques*, pages 187–194, 1999. 1, 3

[7] M. D. Breitenstein, D. Kuettel, T. Weise, L. Van Gool, and H. Pfister. Real-time face pose estimation from single range images. In *CVPR*, pages 1–8, 2008. 1, 4, 5, 6, 7

[8] Q. Cai, D. Gallup, C. Zhang, and Z. Zhang. 3d deformable face tracking with a commodity depth camera. In *ECCV*, pages 229–242. Springer, 2010. 2, 8

[9] Y. Cai, M. Yang, and Z. Li. Robust head pose estimation using a 3d morphable model. *Mathematical Problems in Engineering*, 2015. 1

[10] C. Cao, Y. Weng, S. Lin, and K. Zhou. 3d shape regression for real-time facial animation. *ACM Trans. Graph.*, 32(4):41:1–41:10, 2013. 2

[11] M. Clerc and J. Kennedy. The particle swarm-explosion, stability, and convergence in a multidimensional complex space. *Trans. on Evolutionary Comp.*, 6(1):58–73, 2002. 3, 4

[12] P. de Leva. Adjustments to zatsiorsky-seluyanov's segment inertia parameters. *Journal of Biomechanics*, 29(9):1223 – 1230, 1996. 2

[13] G. Fanelli, M. Dantone, J. Gall, A. Fossati, and L. Van Gool. Random forests for real time 3d face analysis. *Int. J. Comp. Vision*, 101(3):437–458, 2013. 4, 5, 7

[14] G. Fanelli, J. Gall, and L. Van Gool. Real time head pose estimation with random regression forests. In *CVPR*, pages 617–624, 2011. 1, 6

[15] G. Fanelli, T. Weise, J. Gall, and L. Van Gool. Real time head pose estimation from consumer depth cameras. In *Pattern Recognition*, pages 101–110. 2011. 1, 2, 3, 5, 6

[16] J. Kennedy. Particle swarm optimization. In *Encyclopedia of Machine Learning*, pages 760–766. 2010. 3

[17] K. Khoshelham and S. O. Elberink. Accuracy and resolution of kinect depth data for indoor mapping applications. *Sensors*, 12(2):1437–1454, 2012. 6

[18] H. Li, J. Yu, Y. Ye, and C. Bregler. Realtime facial animation with on-the-fly correctives. *ACM Trans. Graph.*, 32(4):42, 2013. 2

[19] K. L. Low. Linear least-squares optimization for point-to-plane ICP surface registration. *Chapel Hill, University of North Carolina*, 2004. 4

[20] M. Martin, F. Van De Camp, and R. Stiefelhagen. Real time head model creation and head pose estimation on consumer depth cameras. In *3DV*, volume 1, pages 641–648, 2014. 2, 5, 6

[21] E. Murphy-Chutorian and M. M. Trivedi. Head pose estimation in computer vision: A survey. *TPAMI*, 31(4):607–626, 2009. 1

[22] I. Oikonomidis, N. Kyriazis, and A. A. Argyros. Efficient model-based 3d tracking of hand articulations using kinect. In *BMVC*, volume 1, page 3, 2011. 1

[23] P. Padeleris, X. Zabulis, and A. A. Argyros. Head pose estimation on depth data based on particle swarm optimization. In *CVPRW*, pages 42–49, 2012. 2, 5, 6

[24] C. Papazov, T. K. Marks, and M. Jones. Real-time 3d head pose and facial landmark estimation from depth images using triangular surface patch features. In *CVPR*, pages 4722–4730, 2015. 1, 5, 6

[25] C. Qian, X. Sun, Y. Wei, X. Tang, and J. Sun. Realtime and robust hand tracking from depth. In *CVPR*, pages 1106–1113, 2014. 2, 5, 8

[26] A. Rekik, A. Ben-Hamadou, and W. Mahdi. 3d face pose tracking using low quality depth cameras. In *VISAPP*, pages 223–228, 2013. 2, 5, 6, 8

[27] S. Rusinkiewicz and M. Levoy. Efficient variants of the icp algorithm. In *3-D Digital Imaging and Modeling*, pages 145–152, 2001. 4

[28] E. Seemann, K. Nickel, and R. Stiefelhagen. Head pose estimation using stereo vision for human-robot interaction. In *AFGR*, pages 626–631, 2004. 1, 2

[29] J. Shotton, T. Sharp, A. Kipman, A. Fitzgibbon, M. Finocchio, A. Blake, M. Cook, and R. Moore. Real-time human pose recognition in parts from single depth images. *Comm. ACM*, 56(1):116–124, 2013. 1

[30] M. Storer, M. Urschler, and H. Bischof. 3d-mam: 3d morphable appearance model for efficient fine head pose estimation from still images. In *CVPRW*, pages 192–199, 2009. 1

[31] Y. Sun and L. Yin. Automatic pose estimation of 3d facial models. In *ICPR*, pages 1–4, 2008. 1

[32] S. Tulyakov, R.-L. Vieriu, S. Semeniuta, and N. Sebe. Robust real-time extreme head pose estimation. In *ICPR*, pages 2263–2268, 2014. 1, 2, 5, 6, 8

[33] P. Viola and M. J. Jones. Robust real-time face detection. *Int. J. Comp. Vision*, 57(2):137–154, 2004. 2

[34] Y. Wang, X. Huang, C.-S. Lee, S. Zhang, Z. Li, D. Samaras, D. Metaxas, A. Elgammal, and P. Huang. High resolution acquisition, learning and transfer of dynamic 3-d facial expressions. In *Computer Graphics Forum*, volume 23, pages 677–686, 2004. 2

[35] T. Weise, S. Bouaziz, H. Li, and M. Pauly. Realtime performance-based facial animation. *ACM Trans. Graphics*, 30(4):77, 2011. 2

[36] T. Weise, B. Leibe, and L. Van Gool. Fast 3d scanning with automatic motion compensation. In *CVPR*, pages 1–8, 2007. 4, 5

[37] L. Zhang, N. Snavely, B. Curless, and S. M. Seitz. Spacetime faces: High-resolution capture for˜ modeling and animation. In *Data-Driven 3D Facial Animation*, pages 248–276. 2008. 2

[38] Y. Zhu and K. Fujimura. 3d head pose estimation with optical flow and depth constraints. In *3-D Digital Imag. and Modeling*, pages 211–216, 2003. 2