Real-Time Face Pose Estimation from Single Range Images

Michael D. Breitenstein, Daniel Kuettel, Thibaut Weise, Luc van Gool
Computer Vision Laboratory, ETH Zurich, Switzerland
{breitenstein, weise, vangool}@vision.ee.ethz.ch, dkuettel@student.ethz.ch

Hanspeter Pfister
School of Engineering and Applied Sciences, Harvard University, Cambridge, MA, USA
pfister@seas.harvard.edu

Abstract

We present a real-time algorithm to estimate the 3D pose of a previously unseen face from a single range image. Based on a novel shape signature to identify noses in range images, we generate candidates for their positions, and then generate and evaluate many pose hypotheses in parallel using modern graphics processing units (GPUs). We developed a novel error function that compares the input range image to precomputed pose images of an average face model. The algorithm is robust to large pose variations of $\pm 90^\circ$ *yaw,* $\pm 45^\circ$ *pitch and* $\pm 30^\circ$ *roll rotation, facial expression, partial occlusion, and works for multiple faces in the field of view. It correctly estimates* 97.8% *of the poses within yaw and pitch error of* 15° *at* 55.8 *fps. To evaluate the algorithm, we built a database of range images with large pose variations and developed a method for automatic ground truth annotation.*

1. Introduction

The estimation of head pose (location and orientation) is often required during runtime (*e.g.*, human-robot interaction or monitoring driver-attentiveness) or during a preprocessing step (*e.g.*, multi-view face recognition or facial expression analysis). Most applications require real-time pose estimation which is robust to large pose variations.

Face pose estimation from 2D images is sensitive to illumination, shadows, and lack of features (*e.g.*, due to occlusions). Lately, 3D acquisition systems reached a level of reliability such that range images can be used to overcome these problems. However, the few pose estimation methods for range images are often limited to a small pose range, need manual initialization, are not running in real-time, or do not work for images with multiple faces. Furthermore, they often use pose tracking over multiple frames. Tracking algorithms may suffer from drift or jitter, need a training phase, require manual interaction (*e.g.*, for key-frame selection), and need a restart after complete occlusions of the field of view.

We present an algorithm for automatic and real-time face pose estimation for previously unseen faces in single range images. It is robust to large pose changes (from profile to frontal view), facial variations (*e.g.*, expressions), partial occlusions (*e.g.*, due to glasses or hair), and to frame dropouts (*e.g.*, due to complete occlusions). It also can handle multiple faces in the field of view. As far as we know, this is the first real-time method with all these features.

Fig. 1 shows an overview of the algorithm. In an offline

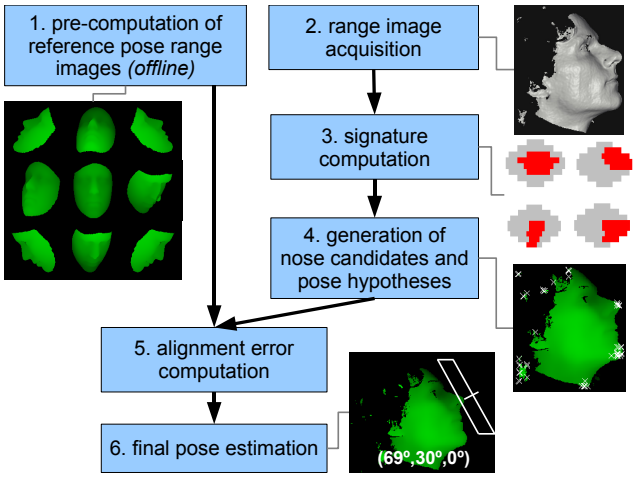


Figure 1. Overview of the algorithm

step, range images of an average face are rendered for many poses, and the resulting *reference pose range images* are saved on the graphics card (1). During runtime, range images of faces are continuously acquired using the real-time active light system of Weise *et al.* [27] (2). For each input range image, the following steps are performed in parallel on the GPU. For each pixel we compute *signatures* that are

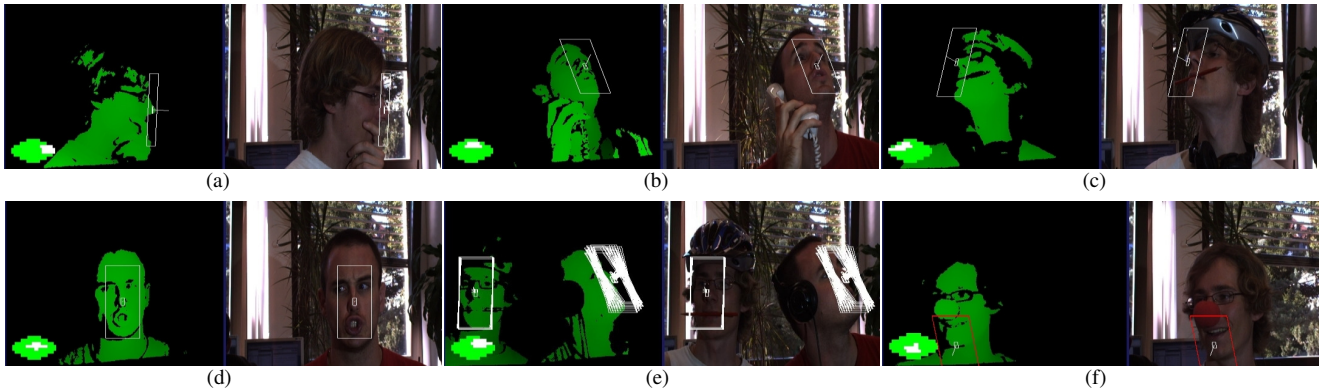


Figure 2. Pose estimation results: The left part of each image shows the range image and the signature of the nose candidates (lower left corner). The right part shows the field of view of the range acquisition system. The estimated face poses are shown in white. The algorithm is robust to large pose variations, partial occlusions (2(a)-2(c)), and expressions (2(d)). It handles multiple faces in the field of view by accepting several hypotheses (2(e)), and detects bad estimations based on the confidence value (2(f)).

distinct for regions with high curvature, such as the nose tip (3). This yields a set of candidate nose positions and orientations (4) that we use as head pose hypotheses. We then compute the error between the reference pose range images corresponding to the pose hypotheses and the input range image using a novel error function (5). The match with the lowest error yields the final pose estimation and a confidence value (6). To evaluate the performance of our algorithm we acquired a database of annotated face range images with systematic, large pose variations (*c.f*. [9, 19]). Some pose estimation results are shown in Fig. 2.

Our approach follows a general tendency towards massively parallel computations that replace piecemeal analysis on the basis of sophisticated feature extraction. The method compares many pose hypotheses in parallel using modern GPUs (see Germann *et al*. [8] for a similar example in the area of bin picking for robotics). This strategy makes the system far more robust: Global rather than local optima are detected, and complete frame drop-outs (*e.g*., when the field-of-view gets completely occluded) are overcome without a problem. The speed of these computations surpasses real-time due to the massive parallelism of modern GPUs.

2. Related Work

Face Tracking: Face tracking in video involves pose estimation, but its accuracy is mostly not explicitly measured. An impressive system is presented by Vacchetti *et al*. [26], which integrates key-frames with recursive tracking. However, tracking suffers from intrinsic disadvantages such as drift, jitter, and the need for manual initialization or offline key-frame selection. It is also limited to one face in the field of view. In contrast, our approach deals with multiple faces and works independently on each frame, which makes it robust to occlusions and frame drop-outs.

Pose Estimation from 2D Images: Image-based pose estimation approaches analyze the entire image but require exact localization of faces [5], or require pose-dependent features. Often, separate detectors for a limited number of pose classes are built and applied in turn [12, 21, 24]. Such methods usually require a very large number of labeled training examples. For their state-of-the-art system, Osadchy *et al*. [18] used 52,850 face images to train a Convolutional Neural Network which correctly classifies 80% of the yaw rotations within 15 ° error at 5 fps. Image-based methods can be enhanced by aligning a generic or person-specific 3D model to the input image (see [4] for a survey), *e.g*., by aligning a 3D deformable model to 2D images [3, 10].

Some systems use stereo or multi-view images, but do not explicitly use depth information for pose estimation. They either match 2D feature patches (*e.g*., [16, 23]) or feature points (*e.g*., [29]) to a user-specific or generic head model. Morency *et al*. [17] build 3D view-based eigenspaces and use a Kalman filter to calculate the pose change between frames. Since most of these methods are part of a larger system (*e.g*., face recognition), pose accuracy is usually not explicitly evaluated. Some systems need manual initialization, have limited pose range, or do not generalize to arbitrary faces. In general, purely image-based approaches are sensitive to illumination, shadows, lack of features, and occlusions.

Pose Estimation from Range Images: Recently, the use of range images (*i.e*., images with per-pixel depth) has become attractive due to the decreasing cost and improved quality of range scanners. Since 3D alignment methods like ICP [2] need a good initialization, 3D features – *e.g*., the nose – are often used for coarse pose estimation. Lu and Jain [15] create hypotheses for the nose position in range images by computing features based on directional maxima. For verification, they compute the nose profile using PCA and a curvature-based shape index. Neither accuracy nor performance are reported. Likewise, Colbry *et al*. [7] generate six nose hypotheses based on global extremal 3D points and the shape index, and compare them using person-

specific 3D models in 15 s. Chang *et al*. [6] match multiple overlapping regions around the nose using curvature information to find eye cavities, nose saddle and nose tip. Xu *et al*. [28] look for extremal points and cap-like shapes. Actual nose tips are found using a Support Vector Machine. Most of these methods are not robust to large pose variations, and do not lend themselves to real-time processing.

There have been a few real-time systems for pose estimation from range images. Seemann *et al*. [25] present an automatic system which runs at 10 fps and operates on each frame separately (similar to our system). After face detection by a skin color detector, the pose is estimated using one fully connected three-layer neural network for each rotation. For the initialization of skin color histograms they use a face detector [12] that requires the faces to be frontal. Our system does not make any assumptions about the initial face pose.

Germann *et al*. [8] developed a GPU pose estimation method for bin picking of rigid objects. We adopt their overall strategy of comparing pre-computed reference range images with an error function to the input range image. However, we focus on the more difficult case of deformable faces. In contrast to their method, we use a novel 3D shape signature for rough pose initialization, a novel error function for fine pose alignment, no distance maps, and no downhill simplex optimization. We also developed a fully functioning system, including real-time range image acquisition, and evaluate its performance on thousands of range images with ground truth annotations.

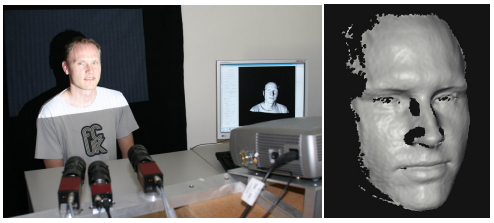
3. Pose Estimation Algorithm

We discuss our algorithm in detail according to the numbering in Fig. 1. The choice of algorithm parameters is discussed in Sec. 4.3.

1. Reference pose range images We generate an average 3D face model from the mean of an eigenvalue decomposition of laser scans of 97 male and 41 female adults (similar to Lee *et al*. [14]). The subjects are not contained in our test dataset for the pose estimation. The average model is rendered for many poses, and the resulting *reference pose range images* are directly stored on the graphics card.

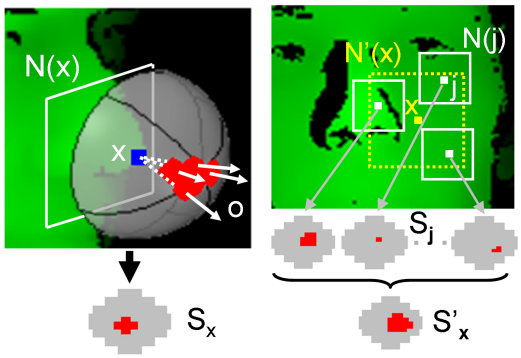
2. Range image acquisition We use the real-time stereo-enhanced structured-light method developed by Weise *et al*. [27], running at 28 fps. The range images are processed for noise and outlier removal using discontinuity-aware Gaussian smoothing and morphological operators. The setup is shown in Fig. 3(a), and a resulting face reconstruction result is shown in Fig. 3(b).

3. 3D shape signature computation Finding the nose tip and its orientation as local positional extremities is a good strategy to roughly estimate head pose (*c.f*., Lu and Jain [15]). To that end, we use a novel 3D shape signature



(a) (b)

Figure 3. a) Scanning setup. b) Range image after noise and outlier removal.



(a) (b)

Figure 4. a) The *single signature* $S_{\boldsymbol{x}}$ is the set of orientations $\boldsymbol{o}$ for which the pixel's position $\boldsymbol{x}$ is a maximum along $\boldsymbol{o}$ compared to pixels in the neighborhood $N(\boldsymbol{x})$. b) Single signatures $S_{\boldsymbol{j}}$ of points $\boldsymbol{j}$ in $N'(\boldsymbol{x})$ are merged into the *aggregated signature* $S'_{\boldsymbol{x}}$.

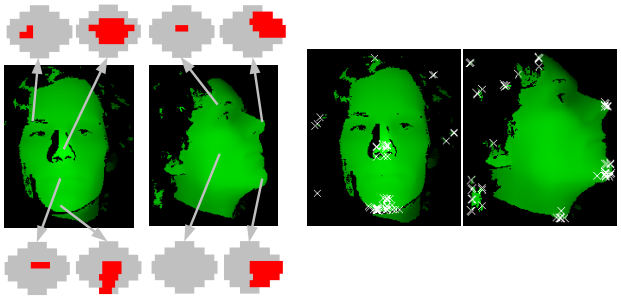
(which is computed for each pixel) instead of computing the surface curvature like previous methods (e.g., Chang *et al*. [6]), because curvature computation is sensitive to noise which mostly is present in range images. Secondly, nose detection in profile views based on curvature is not reliable because the curvature of the visible part of the nose significantly changes for different poses. Furthermore, our signature is more efficient to compute on the GPU than an approximated curvature.

We define the orientation parameters as yaw (ϕ), pitch (θ) and roll (ψ) around object-centered rotation axes (see Fig. 7(b) for an illustration of the rotation angles and axes). The reference frame is aligned with the camera. The *single signature* $S_{\boldsymbol{x}}$ for pixel $\boldsymbol{x}$ is obtained as:

$$S_{\boldsymbol{x}} = \{\boldsymbol{o} = (\phi, \theta) \mid \forall \boldsymbol{i} \in N(\boldsymbol{x}) : d(\boldsymbol{i}, \boldsymbol{o}) \leq 0\}, \quad (1)$$

where $d(\boldsymbol{x}, \boldsymbol{o})$ is the distance to the plane through $\boldsymbol{x}$ with normal orientation $\boldsymbol{o} = (\phi, \theta)$. Because we operate on range data (2.5D), we only compute $S(\boldsymbol{x})$ for the orientations on the half sphere towards the camera.

As shown in Fig. 4(a), $S_{\boldsymbol{x}}$ corresponds to a boolean matrix, where each cell corresponds to an orientation $\boldsymbol{o}$. A cell is marked (red in the figure) if the pixel's 3D position is a maximum along this orientation compared to pixels in a neighborhood $N(\boldsymbol{x})$. Such a pixel is called a *local directional maximum*.



(a) (b)

Figure 5. a) Some aggregated signatures for different face regions. They are similar, *e.g.*, for nose and chin, or for cheek and forehead. b) Nose candidates generated from the signatures (white crosses).

The resulting single signatures are sparse (*i.e.*, contain only few orientations) because one pixel is a local directional maximum only for a few orientations. Therefore, single signatures are not distinctive enough to distinguish different facial regions. In order to arrive at signatures that are more characteristic for local shape, single signatures $S_{\boldsymbol{j}}$ of points $\boldsymbol{j}$ in a neighborhood $N'(\boldsymbol{x})$ are merged to an *aggregated signature* $S'_{\boldsymbol{x}}$ by a boolean OR operation:

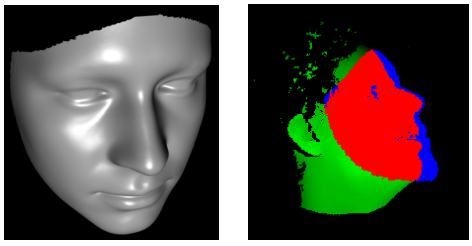
$$S'_{\boldsymbol{x}} = \bigcup_{\boldsymbol{j} \in N'(\boldsymbol{x})} S_{\boldsymbol{j}} \tag{2}$$

Thus, a cell of the aggregated signature is marked if any point $\boldsymbol{j} \in N'(\boldsymbol{x})$ is a local directional maximum for the neighborhood $N(\boldsymbol{j})$ (see Fig. 4(b)). The influence and choice of N and N' are discussed in Sec. 4.3.

As can be seen in Fig. 5(a), the resulting signatures reflect the characteristic curvature of facial areas. The aggregated signatures are distinct for large, convex extremities, such as the nose tip and the chin. Their marked cells typically have a compact shape and cover many adjacent cells compared to those of facial regions which are flat, such as the cheek or forehead. Furthermore, the aggregated signatures look similar if the head is rotated.

4. Generating nose candidates and pose hypotheses We select points as nose candidates based on two criteria: first, at least T of the cells of an aggregated signature have to be marked (see Sec. 4.3 for the choice of T). Second, the cell in the center of all marked cells, called the *mean orientation*, has to be part of the pixel's *single* signature. This ensures that the pixel is a "typical" point for the area represented by the aggregated signature. Fig. 5(b) shows the resulting nose candidate pixels based on the aggregated signatures of Fig. 5(a). The 3D positions and mean orientations of nose candidate pixels form a set of *head pose hypotheses*.

5. Alignment error computation In order to compute the *fine head pose estimation*, an error function evaluates the alignment of reference pose range images $M_{\boldsymbol{O}}$ and the input range image $I_{\boldsymbol{x}}$. In the first *rough alignment* pass, all reference pose range images that correspond to the head pose hypotheses from the shape signatures are being evaluated. The best matching reference pose range images (*i.e.*, with the smallest error) are being passed on to the second pass.



(a) (b)

Figure 6. a) The average 3D face model. b) An alignment of one reference pose range image and the input image. The red and blue areas correspond to the pixel sets $\mathcal{V}$ and $\mathcal{V}^{-1}$ in Eq. 6 and 7.

The orientations of the reference pose range images have been sampled more densely than the orientations of the shape signatures (see Sec. 4.3). Therefore, in the second *fine alignment* pass, the best matching reference pose range images are augmented with additional adjacent reference pose range images. This new set of head pose hypotheses is then compared to the input image using the error function, and the best match is output as the final pose estimate. Fig. 6(a) shows the average 3D face model used for the reference pose range images, and Fig. 6(b) shows an example of the alignment errors.

The nose and chin positions have been annotated by hand in the reference range images $M_{\boldsymbol{O}}$. First, the input image is translated such that each nose candidate position $\boldsymbol{x}$ is matched to the annotated nose position in $M_{\boldsymbol{O}}$. The size of $M_{\boldsymbol{O}}$ is scaled according to the z-value at the nose candidate position $\boldsymbol{x}$. We then compute a per-pixel error function:

$$e(M_{\boldsymbol{O}}, I_{\boldsymbol{x}}) = e_d(M_{\boldsymbol{O}}, I_{\boldsymbol{x}}) + \lambda \cdot e_c(M_{\boldsymbol{O}}, I_{\boldsymbol{x}}) + C, \tag{3}$$

where e_d is a *depth difference error term*, e_c is a *coverage error term*, and λ controls their influence. The constant C is added if no signatures are found in the chin area that has been marked in $M_{\boldsymbol{O}}$. This penalizes cases where a nose candidate position $\boldsymbol{x}$ is actually at the chin, which is then mistakenly being placed at the nose of the reference image.

Depth Difference Error Term: The error term e_d computes the normalized sum of squared depth differences between reference and input range image for all foreground pixels (*i.e.*, pixels where a depth was captured). The functions $M_{\boldsymbol{O}}(\boldsymbol{u})$ and $I_{\boldsymbol{x}}(\boldsymbol{u})$ return the depth at pixel position $\boldsymbol{u}$ and $-\infty$ for background pixels for the reference image and the input range image, respectively. The sets $\mathcal{V}_{M_{\boldsymbol{O}}}$ and $\mathcal{V}_{I_{\boldsymbol{x}}}$ consist of the foreground pixels returned by $M_{\boldsymbol{O}}(\boldsymbol{u})$ and $I_{\boldsymbol{x}}(\boldsymbol{u})$:

$$\mathcal{V}_{M_{\boldsymbol{O}}} = \{\boldsymbol{u} \mid M_{\boldsymbol{O}}(\boldsymbol{u}) \neq -\infty\} \tag{4}$$

$$\mathcal{V}_{I_{\boldsymbol{x}}} = \{\boldsymbol{u} \mid I_{\boldsymbol{x}}(\boldsymbol{u}) \neq -\infty\} \quad (5)$$

The set $\mathcal{V} = \mathcal{V}_{M_{\boldsymbol{O}}} \cap \mathcal{V}_{I_{\boldsymbol{x}}}$ contains all pixels which are foreground in the input image as well as the reference image (the red area in Fig. 6(b)). The depth difference error term is defined by:

$$e_d(M_{\boldsymbol{O}}, I_{\boldsymbol{x}}) = \frac{\sum_{\boldsymbol{u}\in\mathcal{V}} (M_{\boldsymbol{O}}(\boldsymbol{u}) - I_{\boldsymbol{x}}(\boldsymbol{u}))^2}{\mid \mathcal{V} \mid} \quad (6)$$

Coverage Error Term: The depth difference error term only computes an error over foreground pixels, without taking into account the number of pixels. Hence, it does not penalize small overlaps between input and model (*e.g.*, the model could be perfectly aligned to the input but the overlap consists only of one pixel). Hence, this error term is necessary to boost the prominence of alignments where all foreground pixels of the reference model fit to foreground pixels of the input image.

The set $\mathcal{V}^{-1} = \mathcal{V}_{M_{\boldsymbol{O}}} \backslash \mathcal{V}_{I_{\boldsymbol{x}}}$ contains all foreground pixels of the reference image that have no correspondence in the input image (the blue area in Fig. 6(b)). The coverage error term is the squared ratio of pixels defined by $\mathcal{V}^{-1}$:

$$e_c(M_{\boldsymbol{O}}, I_{\boldsymbol{x}}) = \left(\frac{\mid \mathcal{V}^{-1} \mid}{\mid \mathcal{V}_{M_{\boldsymbol{O}}} \mid}\right)^2 \quad (7)$$

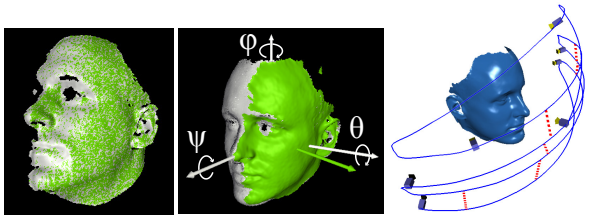
6. Final pose estimation The comparison of the input range image against many reference pose range images is being performed in parallel on the GPU. The pose of the reference pose range image with the smallest error value is output as the final pose estimate. Additionally, its error value can be used as a confidence value (see Sec. 5).

4. Implementation

4.1. Optimization for the GPU

We implemented the pose estimation algorithm completely on the GPU (NVIDIA GeForce 8800 GTX), making use of the Compute Unified Device Architecture of NVIDIA [1] to exploit the GPU as a generic data-parallel computing device. The GPU comprises 16 multi-CPUs (mCPUs), where each of them can execute one or more blocks of up to 512 threads each. Threads in the same block can communicate efficiently using the shared memory, whereas threads on different mCPUs can only communicate over the global GPU memory, which is a few magnitudes slower.

To compute the single signatures, the input range image is divided into adjacent patches of size $N(x)$ (see Eq. 1). A patch is loaded into shared memory only once and the threads jointly access the shared memory. To compute the signatures at locations of a directional maximum, each thread checks a neighborhood of size $N(x)$ for its directional maximum (see Eq. 2). Since we chose $N = N'$ (see Sec. 4.3), a thread aggregates its signatures by processing the list of single signatures of neighboring patches.



(a) (b) (c)

Figure 7. a) Two scans, registered using sequential ICP. b) Illustration of rotation angles and axes for pose estimation (white), and relative frame pose annotation (transformation of ψ-axis to green arrow). Previously registered frames are colored grey, the current scan green. c) Camera trajectory for one acquisition sequence. Dotted red lines depict additional multi-view registration pairs.

For the pixel-wise computation of the error function (Eq. 3), every block evaluates one pose hypothesis using 100 threads that compare different parts of the reference range images. Because loading one contiguous block is several magnitudes faster than random access, blocks containing as many contiguous pixels of the range image as there are threads are loaded together. Each thread then simultaneously processes its pixel within the block, *i.e.*, in an interleaved fashion. Intermediate errors are stored in the shared memory and eventually collected by the last thread.

4.2. Data Set and Ground Truth Annotation

We created a test data set consisting of 10,545 range images from 26 people (male and female).[1] Each person freely turned her head (see Fig. 7(c) for a typical camera trajectory) while the scanner captured range images at 28 fps. The resulting range images have a resolution of 640×480, and a face typically consists of about 150×200 depth values. Usually, range data is annotated with ground truth by one the following methods (*e.g.*, see the survey by Gross [9]): capturing images from several viewpoints at the same time with cameras at known positions (*e.g.*, measured by a theodolite), using one camera and asking the subject to look to hand-marked points in the room, computing the angle by manually clicking to unique locations or markers on the face, or using magnetic sensors (flock of birds).

Instead, we capture ground truth data by determining the relative pose difference between a frontal anchor pose and the captured range images in the sequence. We measure the pose difference between subsequent frames using ICP [2, 22, 11] and concatenate the consecutive transformations. Fig. 7(a) shows an example ICP registration of two consecutive scans, and Fig. 7(b) shows one frame in relation to the previous scans. Since sequential registration leads to an accumulation of errors [22], we use a combination of

[1] www.vision.ee.ethz.ch/~bremicha/cvpr2008/

the methods of Krishnan *et al*. [13] and Pulli [20] to substantially reduce the error over all scans. To achieve this, additional pair-wise registration constraints between non-consecutive scans are added to the ICP procedure (see the dotted red lines connecting different parts of the camera trajectory in Fig. 7(c)).

With our method, a recorded range image stream of about thirty seconds is automatically annotated in a few seconds, very precisely and without manual interaction, assuming the face in the first frame is frontal.

4.3. Algorithm Parameters

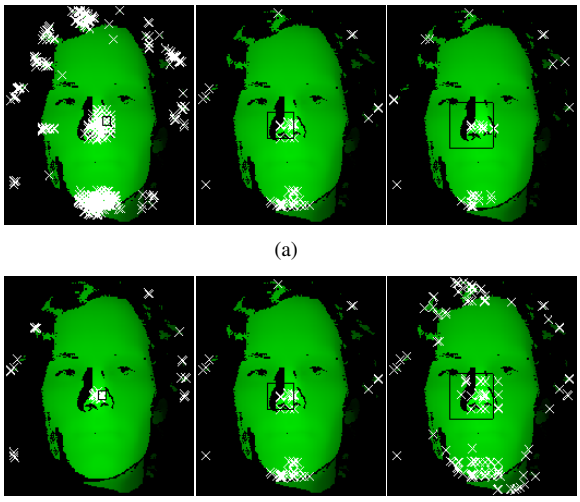
The parameters of the algorithm depend on the resolution of the range images and are independent of the specific hardware setup. In this section, we discuss the choice of these parameters.

We rendered the 3D average face model for the pose orientations $\phi \leq \pm 90\,^\circ$, $\theta \leq \pm 45\,^\circ$, and $\psi \leq \pm 30\,^\circ$ with step sizes of $6\,^\circ$ for ϕ and θ, and $15\,^\circ$ for ψ. This defines the set of reference pose range images. Note that it also defines the minimum error for the final pose estimate.

Each signature is computed for 56 orientations resulting from a regular angular sampling of the hemisphere. After evaluation of the error function during the rough alignment pass, we select the 5 best rough pose hypotheses together with their 5×5 neighbouring orientations from the reference range image set. This set of up to 125 poses is evaluated in the fine alignment pass, and the best match is output as the final pose estimate. We experimentally chose those orientation sampling densities to achieve a good balance between accuracy, memory footprint, and speed of our method.

The neighborhood areas $N \times N$ and $N' \times N'$ (see Eq. 1 and 2) control the size of the neighborhood for which a point has to be a directional maximum, and the neighborhood in which single signatures are aggregated, respectively. For both N and N' we experimentally chose a size of 31. Fig. 8 shows the nose candidates generated from the resulting signatures when N or N' is varied between 11, 31 or 51 pixels, while keeping the other one fixed to 31 pixels. If N is small, a pixel is more probable to be a directional maximum for many orientations (see Fig. 8(a)). Therefore, the signatures of different pixels look more similar, which results in more nose candidates, many of which are wrong. In contrast, the nose candidates that result from a very large N are pixels that are global directional maxima, *i.e*., which are the maximum for an orientation compared to *all* other points. This is very sensitive to outliers or occlusions.

Fig. 8(b) demonstrates the influence of N'. If N' is larger, single signatures in a larger area are included. Therefore, if N' is too large, all aggregated signatures contain many orientations and look similar for pixels from different facial regions. However, if N' is too small, the signatures are not distinctive because they represent not the characteristics of a facial area but only of a few pixels.



(a)

(b)

Figure 8. The resulting nose candidates (white crosses) for a) N and b) N' equal to 11, 31, and 51 pixels (black squares).

To determine λ, which controls the influence of the error terms in Eq. 3, we analyzed the number of correct orientation estimates for yaw and pitch with an error smaller than $10\,^\circ$, $12\,^\circ$, $15\,^\circ$, and $20\,^\circ$. We call the percentage of orientations satisfying this criterion the *pose estimation success rate*. For all our experiments, we consider the estimated nose position correct if its Euclidean Distance to the true position is less than 20 mm. For these tests we used five subjects who are not part of the ground truth data set. The highest pose estimation success rate is achieved for $\lambda = 10,000$, independent of the error criterion (see Fig. 9(a)). The constant C in Eq. 3 is experimentally determined and turned out to be robust for all test scenes.

Fig. 9(b) shows the pose estimation success rate and fps for different resolutions of the reference range images. For each resolution, we varied the signature threshold T for nose candidate selection (see Sec. 3). T controls which and how many nose candidates are evaluated by the error function. Even though we only show values for T for one example resolution in the figure, they all follow the same trend, and $T = 6$ achieves maximum success rate independent of the resolution. Surprisingly, the pose estimation success rate is still good for very low reference range image resolutions. Based on these experiments, we chose a resolution of 32×32 pixels for the reference images and a signature threshold $T = 6$.

5. Results and Discussion

In the ROC plot in Fig. 9(c), the performance of the algorithm is evaluated for different error criteria (see Sec. 4.3). For a pose error of within $15\,^\circ$, our algorithm classifies 97.8% of all test data correctly, 98.4% for $20\,^\circ$, and 80.8%

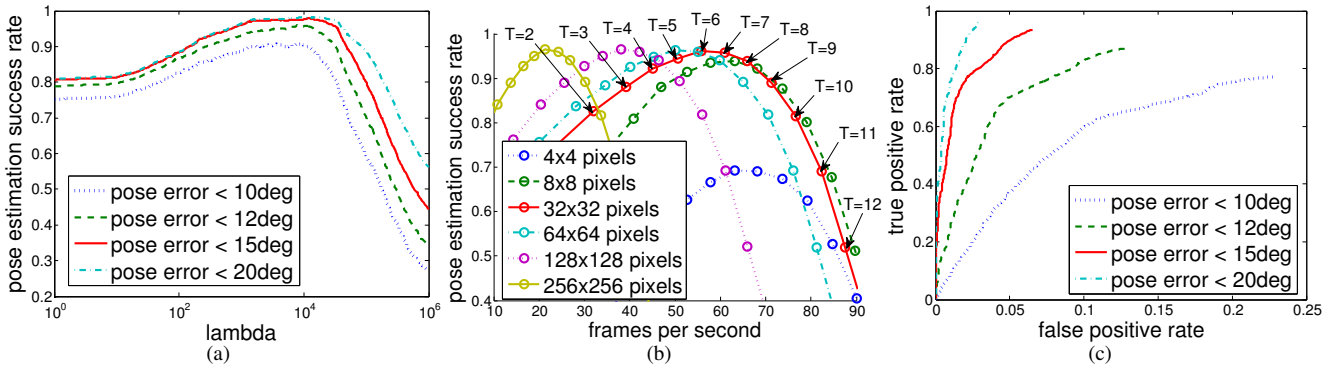


Figure 9. a) Pose estimation success rate for different values of λ in Eq. 3. We chose $\lambda = 10,000$. b) Pose estimation success rate for different reference range image resolutions. For each resolution we vary T (see Sec. 3), as shown for one example. Even for very low resolutions the success rate is still high. c) The ROC curves demonstrate the pose estimation performances for different error criteria.

for $10\,^\circ$, respectively. Compared to other systems that work for such large pose variations, like the one of Seemann *et al.* [25] (95.1% pose estimation success rate for an error smaller than $20\,^\circ$, and 75.2% for $10\,^\circ$), or the state of the art 2D system of Osadchy *et al.* [18] (80% correctly estimated yaw-rotations within $15\,^\circ$ error), our method appears to be more accurate (and is faster, see below). However, no direct comparison is possible because different datasets were used. With a conservative threshold for a true positive rate of 80% and false positive rate of 3%, the mean error and standard deviation of the pose estimation are $(\mu_\phi = 6.1\,^\circ, \sigma_\phi = 10.3\,^\circ)$, $(\mu_\theta = 4.2\,^\circ, \sigma_\theta = 3.9\,^\circ)$, and $(\mu_{nose} = 9$ mm, $\sigma_{nose} = 14$ mm), respectively. Because our test dataset does not include roll rotations, we quantitatively evaluated only the estimation of yaw and pitch rotation. However, our method is able to robustly estimate arbitrary roll rotations by including appropriate reference pose range images, as shown in the companion video. The test data set covers a pose range of $\pm 90\,^\circ$ yaw and $\pm 45\,^\circ$ pitch rotation. Since the pose coverage is not uniform for different poses (because the persons were allowed to freely move their head), we normalize the pose error over the number of available test images per pose range. In Fig. 10, the test images are divided into pose areas of $15\,^\circ \times 15\,^\circ$, and the number of images per area is color-encoded. For each pose area, the pose estimation success rate (see Sec. 4.3) is indicated. We only show estimation results for areas where at least 10 test images were available. This evaluation demonstrates that even for large pose variations, such as in profile views, the results of the algorithm are good. To the best of our knowledge, the pose estimation accuracy has not been evaluated for other methods that work for such large pose variations. Therefore, no direct comparisons are possible.

The resulting computation time for different parts of the algorithm are shown in Table 1. On average, about 481 pose hypotheses are evaluated. The performance for the pose estimation algorithm is about 55.8 fps, or about 42.5 fps including data transfer from memory to GPU. This is

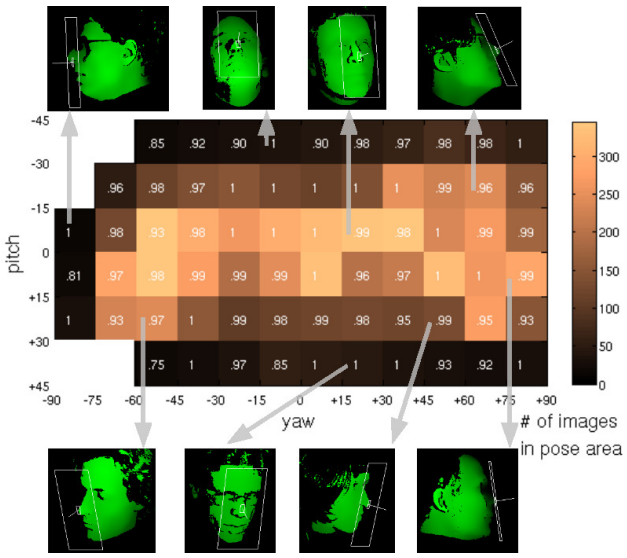


Figure 10. Pose estimation success rate dependent on pose area: The data set is divided into pose areas of $15\,^\circ \times 15\,^\circ$ and colored proportional to the number of test images in this area (see color scale). The number in each area is the success rate.

Signature Computation	3.5 ms
Signature Selection	2.1 ms
Selection of 5 Pose Hypotheses	9.6 ms
Error Comput. of 125 Hypotheses	2.8 ms
Total Pose Estimation	18 ms (55.8 fps)
With Data Loading Memory - GPU	23.5 ms (42.5 fps)

Table 1. Computation time for different parts of the algorithm.

some factors faster than other comparable methods (*e.g.*, Seemann *et al.* [25] report a runtime of 10 fps). The range image acquisition system runs at 28 fps. Together, the complete system runs at about 15 fps on one machine with an Intel Core Duo 2 CPU and a NVIDIA GeForce 8800 GTX.

Because of the low resolution of the reference images (see Sec. 4.3), the average face model generalizes well, and

face variations due to differences between persons, expression, or glasses etc. can be neglected (see Figs. 2(a)-2(c)). Although our algorithm fails if the nose in the image is completely occluded, wrong estimations can be detected based on the low confidence (*i.e.*, high error) value (see Fig. 2(f)). Furthermore, our method estimates the pose of several faces in the same image without additional computation time (see Fig. 2(e)). We simply output either the n best pose hypotheses, or those whose error is below a certain threshold.

6. Conclusion

Our real-time pose estimation algorithm is scalable, robust to large pose changes and facial variations, works for previously unseen persons and multiple faces in the field of view, does not require manual initialization or interaction, does not suffer from the disadvantages of tracking, and does not require a costly training procedure with a lot of data. While previous approaches might be almost as fast or precise, our algorithm combines the advantages of different methods. This makes it unique and possible to generalize for other applications. Future work will include experiments with different range data acquisition systems, and systematic evaluation of roll rotation estimation.

Acknowledgments: We thank B. Leibe, M. Germann, and I.-K. Park for valuable discussions, B. Bickel for help with the video, and Mitsubishi Electric Research Labs and NVIDIA for their support of this project. M. Breitenstein gratefully acknowledges support by the EU project HERMES (IST-027110) and Swiss NSF NCCR project IM2.

References

[1] NVIDIA Compute Unified Device Architecture (CUDA). http://developer.nvidia.com/cuda.

[2] P. Besl and N. McKay. A method for registration of 3-d shapes. *PAMI*, 14(2):239–256, 1992.

[3] V. Blanz and T. Vetter. Face recognition based on fitting a 3d morphable model. *PAMI*, 25(9):1063–1074, 2003.

[4] K. W. Bowyer, K. Chang, and P. Flynn. A survey of approaches and challenges in 3d and multi-modal 3d + 2d face recognition. *CVIU*, 101(1):1–15, 2006.

[5] L. M. Brown and Y.-L. Tran. Comparative study of coarse head pose estimation. In *MOTION*, 2002.

[6] K. I. Chang, K. W. Bowyer, and P. J. Flynn. Multiple nose region matching for 3d face recognition under varying facial expression. *PAMI*, 28(10):1695–1700, 2006.

[7] D. Colbry, G. Stockman, and A. Jain. Detection of anchor points for 3d face verification. *A3DISS, CVPR Workshop*, 2005.

[8] M. Germann, M. D. Breitenstein, I. K. Park, and H. Pfister. Automatic pose estimation for range images on the gpu. *3DIM*, 2007.

[9] R. Gross. Face databases. In S.Li and A.Jain, editors, *Handbook of Face Recognition*. Springer, 2005.

[10] L. Gu and T. Kanade. 3d alignment of face in a single image. *CVPR*, 2006.

[11] T. Jaeggli, T. Koninckx, and L. V. Gool. Online 3d acquisition and model integration. *PROCAMS, ICCV Workshop*, 2003.

[12] M. Jones and P. Viola. Fast multi-view face detection. Technical Report TR2003-096, Mitsubishi Electric Research Laboratories, 2003.

[13] S. Krishnan, P. Y. Lee, J. B. Moore, and S. Venkatasubramanian. Global registration of multiple 3D point sets via optimization-on-a-manifold. *Symposium on Geometry Processing*, pages 187–196, 2005.

[14] J. Lee, B. Moghaddam, H. Pfister, and R. Machiraju. Silhouette-based 3D face shape recovery. *Graphics Interface*, 2003.

[15] X. Lu and A. K. Jain. Automatic feature extraction for multiview 3d face recognition. *FG*, 2006.

[16] Y. Matsumoto and A. Zelinsky. An algorithm for real-time stereo vision implementation of head pose and gaze direction measurement. *FG*, 2000.

[17] L.-P. Morency, P. Sundberg, and T. Darrell. Pose estimation using 3d view-based eigenspaces. *FG*, 2003.

[18] M. Osadchy, M. L. Miller, and Y. LeCun. Synergistic face detection and pose estimation with energy-based models. *NIPS*, pages 1017–1024, 2005.

[19] P. J. Phillips, P. J. Flynn, T. Scruggs, K. W. Bowyer, J. Chang, K. Hoffman, J. Marques, J. Min, and W. Worek. Overview of the face recognition grand challenge. *CVPR*, 2005.

[20] K. Pulli. Multiview registration for large data sets. *3DIM*, 1999.

[21] H. A. Rowley, S. Baluja, and T. Kanade. Rotation invariant neural network-based face detection. *CVPR*, 1998.

[22] S. Rusinkiewicz, O. A. Hall-Holt, and M. Levoy. Real-time 3D model acquisition. *TOG*, 21(3):438–446, 2002.

[23] P. Sankaran, S. Gundimada, R. C. Tompkins, and V. K. Asari. Pose angle determination by face, eyes and nose localization. *FRGC, CVPR Workshop*, 2005.

[24] H. Schneiderman and T. Kanade. A statistical method for 3d object detection applied to faces and cars. *CVPR*, 2000.

[25] E. Seemann, K. Nickel, and R. Stiefelhagen. Head pose estimation using stereo vision for human-robot interaction. *FG*, 2004.

[26] L. Vacchetti, V. Lepetit, and P. Fua. Fusing online and offline information for stable 3d tracking in real-time. *CVPR*, 2003.

[27] T. Weise, B. Leibe, and L. V. Gool. Fast 3d scanning with automatic motion compensation. *CVPR*, 2007.

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

[29] J. Yao and W. K. Cham. Efficient model-based linear head motion recovery from movies. *CVPR*, 2004.