Head Pose Estimation from Passive Stereo Images

M.D. Breitenstein[1], J. Jensen[2], C. Høilund[2], T.B. Moeslund[2], and L. Van Gool[1]

[1] ETH Zurich, Switzerland
[2] Aalborg University, Denmark

Abstract. We present an algorithm to estimate the 3D pose (location and orientation) of a previously unseen face from low-quality range images. The algorithm generates many pose candidates from a signature to find the nose tip based on local shape, and then evaluates each candidate by computing an error function. Our algorithm incorporates 2D and 3D cues to make the system robust to low-quality range images acquired by passive stereo systems. It handles large pose variations (of $\pm 90^\circ$ yaw and $\pm 45^\circ$ pitch rotation) and facial variations due to expressions or accessories. For a maximally allowed error of 30°, the system achieves an accuracy of 83.6%.

1 Introduction

Head pose estimation is the problem of finding a human head in digital imagery and estimating its orientation. It can be required explicitly (*e.g.*, for gaze estimation in driver-attentiveness monitoring [11] or human-computer interaction [9]) as well as during a preprocessing step (*e.g.*, for face recognition or facial expression analysis).

A recent survey [12] identifies the assumptions of many state-of-the-art methods to simplify the pose estimation problem: small pose changes between frames (*i.e.*, continuous video input), manual initialization, no drift (*i.e.*, short duration of the input), 3D data, limited pose range, rotation around one single axis, permanent existence of facial features (*i.e.*, no partial occlusions and limited pose variation), previously seen persons, and synthetic data. The vast majority of previous approaches are based on 2D data and suffer from several of those limitations [12]. In general, purely image-based approaches are sensitive to illumination, shadows, lack of features (due to self-occlusion), and facial variations due to expressions or accessories like glasses and hats (*e.g.*, [14,6]). However, recent work indicates that some of these problems could be avoided by using *depth information* [2,15].

In this paper, we present a method for robust and automatic head pose estimation from *low-quality range images*. The algorithm relies only on 2.5D range images and the assumption that the nose of a head is visible in the image. Both assumptions are weak. Two color images (instead of one) are sufficient to compute depth information in a passive stereo system, thus, passive stereo imagery is

A.-B. Salberg, J.Y. Hardeberg, and R. Jenssen (Eds.): SCIA 2009, LNCS 5575, pp. 219–228, 2009.

cheap and relatively easy to obtain. Secondly, the nose is normally visible whenever the face is (in contrast to the corners of both eyes, as required by other methods, *e.g.*, [17]). Furthermore, our method particularly does not require any manual initialization, is robust to very large pose variations (of $\pm 90\,^\circ$ yaw and $\pm 45\,^\circ$ pitch rotation), and is identity-invariant.

Our algorithm is an extension of earlier work [1] that relies on high-quality range data (from an active stereo system) and does not work for low-quality passive stereo input. Unfortunately, the need for high-quality data is a strong limitation for real-world applications. With active stereo systems, users are often blinded by the bright light from a projector or suffer from unhealthy laser light. In this work, we generalize the original method and extend it for the use of low-quality range image data (captured, *e.g.*, by an off-the-shelf passive stereo system).

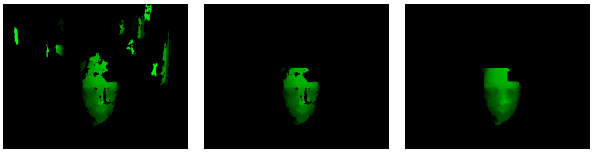
Our algorithm works as follows: First, a region of interest (ROI) is found in the color image to limit the area for depth reconstruction. Second, the resulting range image is interpolated and smoothed to close holes and remove noise. Then, the following steps are performed for each input range image. A pixel-based signature is computed to identify regions with high curvature, yielding a set of *candidates for the nose position*. From this set, we generate *head pose candidates*. To evaluate each candidate, we compute an error function that uses pre-computed *reference pose range images*, the ROI detector, motion direction estimation, and favors temporal consistency. Finally, the candidate with the lowest error yields the *final pose estimation* and a confidence value.

In comparison to our earlier work [1], we substantially changed the error function and added preprocessing steps. The presented algorithm works on single range images, making it possible to overcome drift and complete frame drop-outs in case of occlusions. The result is a system that can directly be used together with a low-cost stereo acquisition system (*e.g.*, passive stereo).

Although a few other face pose estimation algorithms use stereo input or multi-view images [8,17,21,10], most do not explicitly exploit depth information. Often, they need manual initialization, have limited pose range, or do not generalize to arbitrary faces. Instead of 2.5D range images, most systems using depth information are based on complete 3D information [7,4,3,20], the acquisition of which is complex and thus of limited use for most real-world applications. Most similar to our algorithm is the work of Seemann *et al.* [18], where the disparity and grey values are directly used in Neural Networks.

2 Range Image Acquisition and Preprocessing

Our head pose estimation algorithm is based on depth, color and intensity information. The data is extracted using an off-the-shelf stereo system (the Point Grey Bumblebee XB3 stereo system [16]), which provides color images with a resolution of 640×480 pixels. The applied stereo matching algorithm is a sum-of-absolute-differences correlation method that is relatively fast but produces mediocre range images. We speed it up further by limiting the allowed disparity range (*i.e.*, reducing the search region for the correlation).



(a) Input. (b) ROI only. (c) Interpolated.

Fig. 1. a) The range image, b) after background noise removal, c) after interpolation

The data is acquired in a common office setup. Two standard desk lamps are placed near the camera to ensure sufficient lighting. However, shadows and specularities on the face cause a considerable amount of noise and holes in the resulting depth images.

To enhance the quality of the range images, we remove background and foreground noise. The former can be seen in Fig. 1(a) in form of the large, isolated objects around the head. These objects originate from physical objects behind the user's head or due to erroneous 3D estimation. We handle such background noise by computing a region of interest (ROI) and ignoring all computed 3D points outside (see result in Fig. 1(b)). For this purpose, we apply a frontal 2D face detector [6]. As long as both eyes are visible, it detects the face reliably. When no face is detected we keep the ROI from the previous frame. In Fig. 1(b), foreground noise is visible, caused by the stereo matching algorithm. If the stereo algorithm fails to compute depth values, *e.g.*, in regions that are visible for one camera only, or due to specularities, holes appear in the resulting range image. We fill such holes by linear interpolation to remove large discontinuities on the surface (see Fig. 1(c)).

3 Finding Pose Candidates

The overall strategy of our algorithm is to find good candidates for the face pose (location and orientation) and then to evaluate them (see Sec 4). To find pose candidates, we try to locate the nose tip and estimate its orientation around object-centered rotation axes as local positional extremities. This step needs only local computations and thus can be parallelized for implementation on the GPU.

3.1 Finding Nose Tip Candidates

One strategy to find the nose tip is to compute the curvature of the surface, and then to search for local maxima (like previous methods, *e.g.*, [3]). However, curvature computation is very sensitive to noise, which is prominent especially in passively acquired range data. Additionally, 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. Instead, our algorithm is based on a signature to approximate the local shape of the surface.

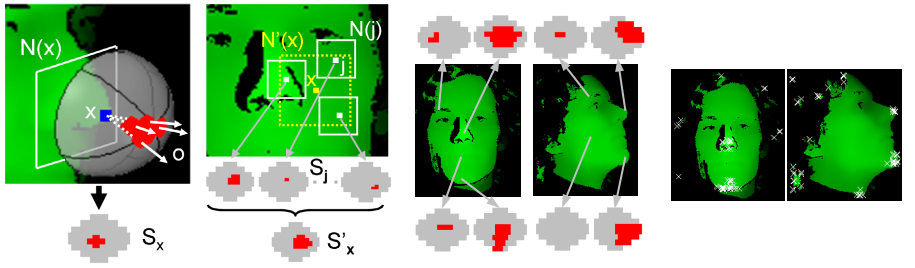


Fig. 2. a) The *single signature* $S_{\boldsymbol{x}}$ is the set of orientations $\boldsymbol{o}$ for which the pixel's position $\boldsymbol{x}$ is a maximum along $\boldsymbol{o}$ compared to pixels in the neighborhood $N(\boldsymbol{x})$. b) Single signatures $S_{\boldsymbol{j}}$ of points $\boldsymbol{j}$ in $N'(\boldsymbol{x})$ are merged into the final signature $S'_{\boldsymbol{x}}$. c) The resulting signatures for different facial regions are similar across different poses. The signatures at nose and chin indicate high curvature areas compared to those at cheek and forehead. d) Nose candidates (white), generated based on selected signatures.

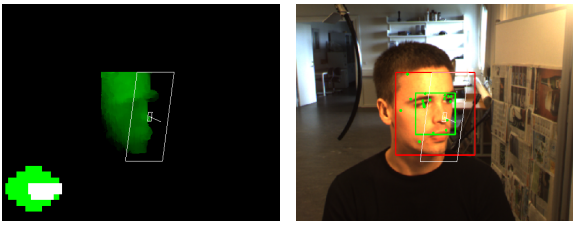
To locate the nose, we compute a 3D shape signature that is distinct for regions with high curvature. In a first step, we search for pixels $\boldsymbol{x}$ whose 3D position is a maximum along an orientation $\boldsymbol{o}$ compared to pixels in a local neighborhood $N(\boldsymbol{x})$ (see Fig. 2(a)). If such a pixel (called a *local directional maximum*) is found, a *single* signature $S_{\boldsymbol{x}}$ is stored (as a boolean matrix). In $S_{\boldsymbol{x}}$, one cell corresponds to one orientation $\boldsymbol{o}$, which is marked (red in Fig. 2(a)) if the pixel is a local directional maximum along this orientation. We only compute $S_{\boldsymbol{x}}$ for the orientations on the half sphere towards the camera, because we operate on range data (2.5D).

The resulting *single* signatures typically contain only a few marked orientations. Hence, they are not distinctive enough yet to reliably distinguish between different facial regions. Therefore, we merge single signatures $S_{\boldsymbol{j}}$ in a neighborhood $N'(\boldsymbol{x})$ to get signatures that are characteristic for the local shape of a whole region (see Fig. 2(b)).

Some resulting signatures for different facial areas are illustrated in Fig. 2(c). As can be seen, the resulting signatures reflect the characteristic local curvature of facial areas. The 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 that are flat, such as the cheek or forehead. Furthermore, the signature for a certain facial region looks similar if the head is rotated.

3.2 Generating Pose Candidates

Each *pose candidate* consists of the location of a *nose tip candidate* and its respective *orientation*. We select points as nose candidates based on the signatures using two criteria: first, the whole area around the point has a convex shape, *i.e.*, a large amount of the cells in the signature has to be marked. Secondly, the



(a) (b)

Fig. 3. The final output of the system: a) the range image with the estimated face pose and the signature of the best nose candidate, b) the color image with the output of the face ROI (red box), the nose ROI (green box), the KLT feature points (green), and the final estimation (white box). *(Best viewed in color)*

point is a "typical" point for the area represented by the signature (*i.e.*, it is in the center of the convex area). This is guaranteed if the cell in the center of all marked cells (*i.e.*, the *mean orientation*) is part of the pixel's *single* signature. Fig. 2(d) shows the resulting nose candidates based on the signatures of Fig. 2(c). Finally, the 3D positions and mean orientations of selected nose tip candidates form the set of final *head pose candidates* $\{P\}$.

4 Evaluating Pose Candidates

To evaluate each pose candidate P_{cur} corresponding to the nose candidate N_{cur}, we compute an error function. Finally, the candidate with the lowest error yields the final pose estimation:

$$P_{final} = \arg\min_{P_{cur}}(\alpha e_{nroi} + \beta e_{feature} + \gamma e_{temp} + \delta e_{align} + \theta e_{com}) \quad (1)$$

The error function consists of several error terms e (and their respective weights), which are described in the following subsections. The final error value can also be used as a (inverse) confidence value.

4.1 Error Term Based on Nose ROI

The face detector used in the preprocessing step (Sec. 2) yields a ROI containing the face. Our experiments have shown that the ROI is always centered close to the position of the nose in the image, independent of the head pose. Thus, we compute ROI_{nose}, a region of interest around the nose, using 50% of the size of the original ROI (see Fig. 3(b)). Since we are interested in pose candidates corresponding to nose candidates inside ROI_{nose}, we ignore all the other candidates.

In practice, instead of a hard pruning, we introduce a penalty value χ for candidates outside and no penalty value for candidates inside the nose ROI:

$$e_{nroi} = \begin{cases} \chi & \text{if } N_{cur} \notin ROI_{nose} \\ 0 & \text{otherwise} \end{cases} \quad (2)$$

This effectively prevents candidates outside of the nose ROI from being selected as long as there is one other candidate within the nose ROI.

4.2 Error Term Based on Average Feature Point Tracking

Usually, the poses in consecutive frames don't change dramatically. Therefore, we further evaluate pose candidates by checking the temporal correlation between two frames. The change of the nose position between the position in the last frame and the current candidate is defined as a motion vector V_{nose} and should be similar to the overall head movement in the current frame, denoted as V_{head}. However, this depends on the accuracy of the pose estimation in the previous frame. Therefore, we apply this check only if the confidence value of the last estimation is high (*i.e.*, if the respective final error value is below a threshold).

To implement this error term, we introduce the penalty function

$$e_{feature} = \begin{cases} |V_{head} - V_{nose}| & \text{if } |V_{head} - V_{nose}| > T_{feature} \\ 0 & \text{otherwise.} \end{cases} \quad (3)$$

We estimate V_{head} as the average displacement of a number of feature points from the previous to the current frame. Therefore, we use the Kanade-Lucas-Tomasi (KLT) tracker [19] on color images to find feature points and to track them (see Fig. 3(b)). The tracker is configured to select around 50 feature points. In case of an uncertain tracking result, the KLT tracker is reinitialized (*i.e.*, new feature points are identified). This is done if the number of feature points is too low (in our experiments, 15 was a good threshold).

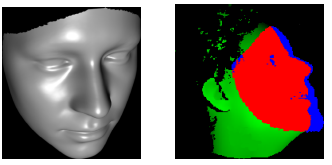
4.3 Error Term Based on Temporal Pose Consistency

We introduce another error term e_{temp}, which punishes large differences between the estimated head pose P_{prev} from the last time step and the current pose candidate P_{cur}. Therefore, the term enforces temporal consistency. Again, this term is only introduced if the confidence value of the estimation in the last frame was high.

$$e_{temp} = \begin{cases} |P_{prev} - P_{cur}| & \text{if } |P_{prev} - P_{cur}| > T_{temp} \\ 0 & \text{otherwise.} \end{cases} \quad (4)$$

4.4 Error Term Based on Alignment Evaluation

The current pose candidate is further assessed by evaluating the alignment of the corresponding *reference pose range image*. Therefore, an average 3D face model was generated from the mean of an eigenvalue decomposition of laser scans from 97 male and 41 female adults (the subjects are not contained in our test dataset for the pose estimation). In an offline step, this average model (see Fig. 4(a)) is then rendered for all possible poses, and the resulting *reference pose range images* are directly stored on the graphics card. The possible number of poses depends on the memory size of the graphics card; in our case, we can



(a) (b)

Fig. 4. a) The 3D model. b) An alignment of one reference image and the input.

store reference pose range images with a step size of 6 ° steps within ±90 ° yaw and ±45 ° pitch rotation. The error e_{align} consists of two error terms, the depth difference error e_d and the coverage error e_c

$$e_{align} = e_d(M_{\boldsymbol{o}}, I_{\boldsymbol{x}}) + \lambda \cdot e_c(M_{\boldsymbol{o}}, I_{\boldsymbol{x}}), \tag{5}$$

where e_{align} is identical with [1]; we refer to this paper for details. Because e_{align} only consists of pixel-wise operations, the alignment of all pose hypotheses is evaluated in parallel on the GPU.

The term e_d is the normalized sum of squared depth differences between reference range image $M_{\boldsymbol{o}}$ and input range image $I_{\boldsymbol{x}}$ for all foreground pixels (*i.e.*, pixels where a depth was captured), without taking into account the actual 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). Therefore, the second error term e_c favors those alignments where all pixels of the reference model fit to foreground pixels of the input image.

4.5 Error Term Based on Rough Head Pose Estimate

The KLT feature point tracker used for the error term $e_{feature}$ relies on motion, but does not help in static situations. Therefore, we introduce a penalty function that compares the current pose candidate P_{cur} with the result P_{com} from a simple head pose estimator.

We apply the idea of [13], where the center of the bounding box around the head (we use the ROI from preprocessing) is compared with the center of mass *com* of the face region. Therefore, the face pixels S are found using an ad-hoc skin color segmentation algorithm ($x_{r,g,b}$ are the values in the color channels)

$$S = \{\mathbf{x} | x_r > x_g \wedge x_r > x_b \wedge x_g > x_b \wedge x_r > 150 \wedge x_g > 100\}. \tag{6}$$

The error term e_{com} is then computed as follows:

$$e_{com} = \begin{cases} |P_{com} - P_{cur}| & \text{if } |P_{com} - P_{cur}| > T_{com} \\ 0 & \text{otherwise} \end{cases} \tag{7}$$

The pose estimation P_{com} is only valid for the horizontal direction and not very precise. However, it provides a rough estimate of the overall viewing direction that can be used to make the algorithm more robust.

Fig. 5. Pose estimation results: good (top), acceptable (middle), bad (bottom)

5 Experiments and Results

The different parameters for the algorithm are determined experimentally and set to $[T_{feature}, T_{temp}, T_{com}, \chi, \lambda] = [40, 25, 30, 10000, 10000]$. The weights of the error terms are chosen as $[\alpha, \beta, \gamma, \delta, \theta] = [1, 10, 50, 1, 20]$. None of them is particularly critical. To obtain test data with ground truth, a magnetic tracking system [5] is applied with a receiver mounted on a headband each test person wears. Each test person used to evaluate the system is first asked to look straight ahead to calibrate the magnetic tracking system for the ground truth. However, this initialization phase is not necessary for our algorithm. Then, each person is asked to freely move the head from frontal up to profile poses, while recording 200 frames. We use 15 test persons yielding 3000 frames in total[1].

We first evaluate the system qualitatively by inspecting each frame and judging whether the estimated pose (superimposed as illustrated in Fig. 5) is acceptable. We define acceptable as whether the estimated pose has correctly captured the general direction of the head. In Fig. 5 the first two rows are examples of acceptable poses in contrast to the last row. This test results in around 80% correctly estimated poses. In a second run, we looked at the ground truth for the acceptable frames and found that our instinctive notion of acceptable corresponds to a maximum pose error of about $\pm 30°$. We used this error condition in a quantitative test, where we compared the pose estimation in each frame with the ground truth. This results in a recognition rate of 83.6%.

We assess the isolated effects of the different error terms (Sec. 4) in Table 1, which shows the recognition rates when only the alignment term and one other

[1] Note that outliers (*e.g.*, a person looks backwards *w.r.t.*the calibration direction) are removed before testing. Therefore, the effect of some of the error terms is reduced due to missing frames, hence the recognition rate is lowered – but more realistic.

Table 1. The result of using different combinations of error terms

Error term	Error $\leq 15°$	Error $\leq 30°$
Alignment	29.0%	61.4%
Nose ROI	36.7%	75.7%
Feature	36.4%	68.7%
Temporal	37.7%	73.4%
Center of Mass	34.0%	66.4%
All	47.3%	**83.6%**

term is used. In [1], a success rate of 97.8% is reported, while this algorithm achieves only 29.0% in our setup. The main reason is the very bad quality of the passively acquired range images. In most error cases, a large part of the face is not reconstructed at all. Hence, special methods are required to account for the quality difference, as done in this work by using complementary error terms.

There are mainly two reasons for the algorithm to fail. First, when the nose ROI is incorrect, nose tip candidates far from the nose could be selected (especially those at the boundary, since such points are local directional maxima for many directions); see middle image of last row in Fig. 5. The nose ROI is incorrect when the face detector breaks for a longer time period (and the last accepted ROI is used). Secondly, if the depth reconstruction of the face surface is too flawed, the alignment evaluation will not be able to distinguish the different pose candidates correctly (see right and left image of the last row in Fig. 5). This is mostly the case if there are very large holes in the surface, which is mainly due to specularities or uniformly textured and colored regions.

The whole system runs with a frame-rate of several fps. However, it could be optimized for real-time performance, *e.g.*, by consistently using the GPU.

6 Conclusion

We presented an algorithm for estimating the pose of unseen faces from low-quality range images acquired by a passive stereo system. It is robust to very large pose variations and for facial variations. For a maximally allowed error of 30°, the system achieves an accuracy of 83.6%. For most applications from surveillance or human-computer interaction, such a coarse head orientation estimation system can be used directly for further processing.

The estimation errors are mostly caused by a bad depth reconstruction. Therefore, the simplest way to improve the accuracy would be to improve the quality of the range images. Although better reconstruction methods exist, there is a tradeoff between accuracy and speed. Further work will include experiments with different stereo reconstruction algorithms.

Acknowledgments. Supported by the EU project HERMES (IST-027110).

References

1. Breitenstein, M.D., Kuettel, D., Weise, T., Van Gool, L., Pfister, H.: Real-time face pose estimation from single range images. In: CVPR (2008)
2. Chang, K.I., Bowyer, K.W., Flynn, P.J.: An evaluation of multimodal 2D+3D face biometrics. PAMI 27(4), 619–624 (2005)
3. Chang, K.I., Bowyer, K.W., Flynn, P.J.: Multiple nose region matching for 3d face recognition under varying facial expression. PAMI 28(10), 1695–1700 (2006)
4. Colbry, D., Stockman, G., Jain, A.: Detection of anchor points for 3d face verification. In: A3DISS, CVPR Workshop (2005)
5. Fastrak, `http://www.polhemus.com`
6. Jones, M., Viola, P.: Fast multi-view face detection. Technical Report TR2003-096, Mitsubishi Electric Research Laboratories (2003)
7. Lu, X., Jain, A.K.: Automatic feature extraction for multiview 3D face recognition. In: FG (2006)
8. Matsumoto, Y., Zelinsky, A.: An algorithm for real-time stereo vision implementation of head pose and gaze direction measurement. In: FG (2000)
9. Morency, L.-P., Sidner, C., Lee, C., Darrell, T.: Head gestures for perceptual interfaces: The role of context in improving recognition. Artificial Intelligence 171(8-9) (2007)
10. Morency, L.-P., Sundberg, P., Darrell, T.: Pose estimation using 3D view-based eigenspaces. In: FG (2003)
11. Murphy-Chutorian, E., Doshi, A., Trivedi, M.M.: Head pose estimation for driver assistance systems: A robust algorithm and experimental evaluation. In: Intelligent Transportation Systems Conference (2007)
12. Murphy-Chutorian, E., Trivedi, M.M.: Head pose estimation in computer vision: A survey. PAMI (2008) (to appear)
13. Nasrollahi, K., Moeslund, T.: Face quality assessment system in video sequences. In: Workshop on Biometrics and Identity Management (2008)
14. Osadchy, M., Miller, M.L., LeCun, Y.: Synergistic face detection and pose estimation with energy-based models. In: NIPS (2005)
15. Phillips, P.J., Flynn, P.J., Scruggs, T., Bowyer, K.W., Chang, J., Hoffman, K., Marques, J., Min, J., Worek, W.: Overview of the face recognition grand challenge. In: CVPR (2005)
16. Point Grey Research, `http://www.ptgrey.com/products/bumblebee/index.html`
17. Sankaran, P., Gundimada, S., Tompkins, R.C., Asari, V.K.: Pose angle determination by face, eyes and nose localization. In: FRGC, CVPR Workshop (2005)
18. Seemann, E., Nickel, K., Stiefelhagen, R.: Head pose estimation using stereo vision for human-robot interaction. In: FG (2004)
19. Tomasi, C., Kanade, T.: Detection and tracking of point features. Technical report, Carnegie Mellon University (April 1991)
20. Xu, C., Tan, T., Wang, Y., Quan, L.: Combining local features for robust nose location in 3D facial data. Pattern Recognition Letters 27(13), 1487–1494 (2006)
21. Yao, J., Cham, W.K.: Efficient model-based linear head motion recovery from movies. In: CVPR (2004)