Self-initializing Head Pose Estimation With a 2D Monocular USB Camera

Patrick Burger, Martin Rothbucher and Klaus Diepold

Technical Report

Self-initializing Head Pose Estimation With a 2D Monocular USB Camera

Patrick Burger, Martin Rothbucher and Klaus Diepold

March 30, 2014

Institute for Data Processing
Technische Universität München

Patrick Burger, Martin Rothbucher and Klaus Diepold . *Self-initializing Head Pose Estimation With a 2D Monocular USB Camera.* Technical Report, Technische Universität München, Munich, Germany, 2014.

Contents

Abstract

In this paper, we describe a method for estimating 3d pose of a human head from a 2d monocular web-cam image. We propose a novel system based on existing computer vision techniques for real-time head pose estimation, which can start and recover from failure automatically without any previous knowledge of the user's appearance or location and keeping the user free of any devices or wires. The system is robust to large pose and facial variations and to partial occlusions. With this setting a widespread use on different machines e.g. computers or laptops is guaranteed. The head pose will be estimated directly from the web-cam image appearance, leaving the user completely free of any artificial markers or special glasses. Furthermore, this system estimates the orientation in the 6 degrees freedom with an uncalibrated monocular camera. The system tracks a person's head pose at a reasonable distance without any camera calibration. Finally, the experimentation on the proposed systems shows that it is accurate, fast (23 fps, 43 ms) and is flexible to use in cases where classic approaches would fail.

1 Introduction

Head pose estimation have been studied in numerous works in the literature [1]. It is often required during runtime applications e.g. to detect and observe human-machine interaction (HMI), to analyse user attention while driving and many other tasks. Since human head orientation is an indication of a person's gaze direction, estimating head pose can provide critical feedback on commercial products and advertisement impact [1]. Most applications claim real-time pose estimation which is robust to wide pose variations [1].

Estimation from 2D images is delicate to shadows, illumination, lack of features etc. Nowadays, 3d depth camera technology e.g. Microsoft Kinect has reached a level of reliability to overcome these problems. However, the few pose estimation methods are often limited to a small pose range, need manual initialization, are not running in real-time because of hight computational time or do not work for images with multiple faces [2]. Additionally they often need a training phase, require manual interaction (e.g. key-frame selection) or a restart after complete occlusions of the field of interest [2]. Hence, a number of studies focused on combining head and eye information for gaze estimation are available in the literature [3] [4]. The work in [5] describes a real-time eye, gaze and head pose tracker for monitoring driver vigilance. The authors use IR-light to detect the pupils and derive the head pose by building a feature space from them. In addition, [6] improve the accuracy of head pose estimation by considering the advantage of multiple camera arrays and complex face models. They combine these two aspects in a probabilistic setting within a mixed-state particle filter framework [7].

However, no study is performed on the feasibility of an accurate appearance-only gaze estimator that considers both the head pose and the facial characteristics.

2 Methods

The head pose estimation (HPE) system used in this work is based on a combination of three different algorithms.
Firstly, before estimating the pose of the head, the user's face in the image of the camera has to be detected. This is solved by using OpenCV's implementation of the Viola-Jones detector [8] with an extended Haar-like feature set [9]. After having detected the face and face features in the image, its location and size defines a rectangular region of interest.
Secondly, we generate a face template with information of the eyes, nose and mouth feature location [10]. The advantage of this implementation is that these features are not greatly affected by facial expressions and have a low variation across users when compared to other features like e.g. face contours. Furthermore, with a simple facial model [10] we can do a first estimation of the head pose to do a full self auto initializing of the HPE system.
Thirdly, we use the POSIT [11] algorithm to determine the pose of the head using at least four non-coplanar 3D points on a model of the object and their two-dimensional projections onto the image plane.

2.1 Face feature detection

The subject's head orientation need to be detected automatically in the video stream. Therefore the eyes, nose and mouth features must be detected for sure. There are several different methods to detect the face by various image cues e.g. skin colour [12], [13], edge patterns, image motion, and ellipse fitting [14]. In our algorithm the face and feature locations are found by the Viola-Jones algorithm [15]. We used freely available trained cascades [16] and [17] to detect the face, eyes, nose and mouth regions. These detectors have been trained on a wide variety of training images, making the detection robust to a wide variety of people with varying skin colour and general appearance. At the beginning, the face detection is performed to constrain the face feature locations to areas inside the face region. This operation increases the computational efficiency of feature detection by limiting false-positive detections and by reducing the search area. The Viola-Jones method [15] is based on the values of simple Haar-like operators, and a cascade of weak classifiers. The required processing time needed to detect the face features is of approximately 3ms on a quadcore 2,4 Ghz computer.

2.2 Face feature tracking

The evaluation of the feature movement is based on matching the template image within a region of interest (ROI). Every time step the new feature location is updated to the position with the best match in the video stream. After the face features have been found, template images of the detected feature locations are captured and stored for subsequent matching. The trained Haar-cascades used in this work [16], [17] have been trained on frontal feature images. Consequently, they cannot be used to detect the features under rotation.
Therefore, to initialize the system correctly the user simply needs to look straight at the camera.To ensure that the template images are not initialized when the user's front is not facing the camera, the centroid of the triangle formed between the eyes and the mouth is compared to the position of the detected nose tip. If the nose tip is within a predefined distance of the centroid, then the person is considered to be in a frontal pose, and the tracking system may be initialised. The maximum distance from the centroid is set manually and allows for variations in human face appearance. Once the template images have been extracted, the feature motion is estimated by matching the template over an area of fixed size centred on the previous feature location. This method uses the simple normalized sum of squared differences (NSSD) to compute image patch similarity.

The typical proximity between corresponding face features across frames allows the template matching to be restricted to an area around the previous feature locations. From the feature tracking alone, estimates of the head translation, scale and roll may be extracted. The translation may be estimated from the average feature locations in the image, the scale may be estimated as a scale factor relative to the initial separation between the features, and the head roll may be estimated from the angle the eyes make with the vertical image axis.
In this work we consider the 3D method proposed by [10] since this is well suited for near-frontal head orientations, where all the face features are visible. Moreover, this method is more suitable for applications in which the user is directly facing the camera. The 3D face model is made up of three world lengths: L_f, L_n and L_m which denote the eye-mouth, nose-mouth, and the nose base-nose tip distances. The corresponding distances in the image are denoted by small case letters.

The face is assumed to a plane passing through the eye and mouth feature locations, and the facial normal is found by joining the nose base to the nose tip. The symmetry axis is located by joining the midpoint between the eyes to the mouth, and the nose base is located using the model ratio $R_m = L_m/L_f$. A further model ratio $R_n = L_n/L_f$ is required for the 3D model, however these two ratios needn't to be calibrated manually for each subject unless high accuracy is needed. The angle σ which the facial normal makes the vector d normal to the image plane which is called the slant. This angle may be found by using the image measurements Θ, l_n, ln, and the model ratio $R_n = L_n/L_f$. It can be shown that in camera centred coordinates, the facial normal n is given by [10]

$$n = [\sin(\sigma)\cos(\tau), \sin(\sigma)\sin(\tau), -\cos(\sigma)]. \tag{2.1}$$

With this method we provide a stable and accurate method to initialize the HPE system without any user interaction.

2.3 Head Pose Estimation

In this work we use the POSIT [11] [18] algorithm to determine the pose of an object using at least four non-coplanar 3D points on a model of the object and their two-dimensional projections onto the image plane. The feature points extraction method for suitable features in the ROI is proposed by [19]. Furthermore, we need to be able to reliably track the movement of all feature points over several successive frames.

Using the Pyramidal Lucas-Kanade Optical Flow algorithm [20], we can estimate the motion of the features of two successive frames recorded by the webcam.
The algorithm uses the feature points that have been found in the ROI of the camera image. The posit algorithm will only start if the initial process completely concluded. Therefore, the user should look nearly frontally into the camera. After that we estimate their 3D position in the coordinate system of a model of a human head. For ease and speed of the calculations, the model of the head can be as simple as a cylinder [21], however we have found a cylindrical model to yield poor results with significant errors in the estimation of the rotation angles. Instead, we modified the shape of the cylinder slightly so that an axial cut through it would form half of a sine wave [18].

$$X_i = \frac{x_i}{w} - 0.5, \tag{2.2}$$

$$Y_i = \frac{y_i}{h} - 0.5, \text{ and} \tag{2.3}$$

$$Z_i = 0.5 \cdot \sin(\frac{x_i}{w} \cdot \pi). \tag{2.4}$$

The model coordinates are calculated during the initialization phase using the coordinates of the feature points x_i, y_i in the image plane and the width w and height h of the face determined during face detection. We found this sinusoidal approximation of the front of the head to give nearly perfect pose estimation results in the desired range of movement when compared to our reference tracking system.

At some point, due to fast movements and due to occlusions or quick lighting variation the tracking system fails. In the case features has been lost and the confidence of the other features is high, the face symmetry and geometrical proportions are exploited. Therefore the system can detect tracking failure in two ways. Firstly, the best matching score between the template and the ROI is below a threshold or there is a significant change in the distance and ratios between the features. The current implementation can recover from single feature tracking failures and occlusions without having to go back to the feature detection routine, if the person has returned to a near frontal pose. In the event that more than one feature is lost or the model ratios between features has changed significantly, the algorithm will return to the feature detection step.

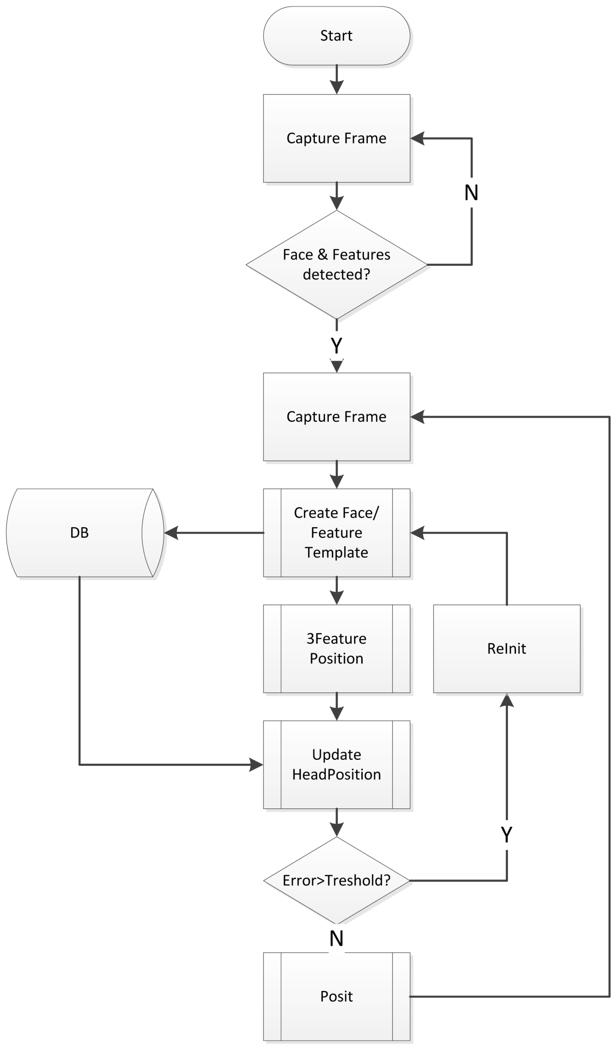

Figure 2.1: Sequence Diagram of HPE process

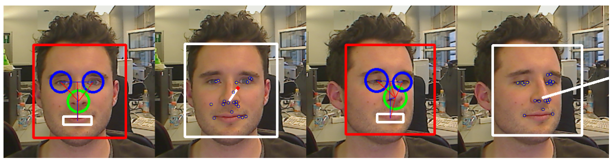

Figure 2.2: Face feature detection with head pose estimation. The pictures with the red boarder shows the face feature detection algorithm. The pictures with the white boarder shows the HPE algorithm results with the estimated view-vector.

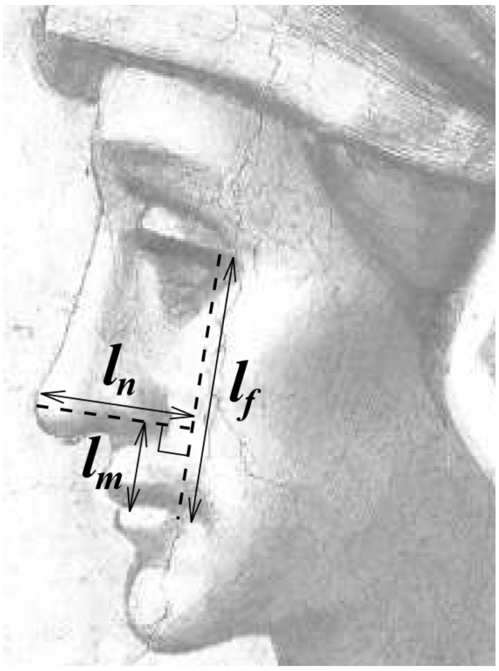

Figure 2.3: The facial model with the three world lengths L_f, L_n and L_m [10].

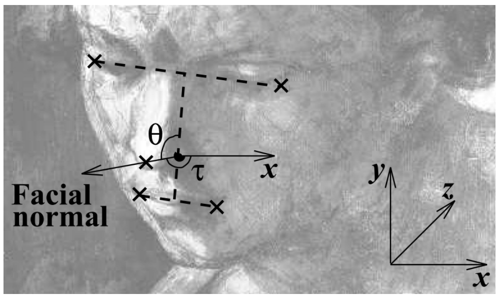

Figure 2.4: The facial model orientation in real world axes [10].

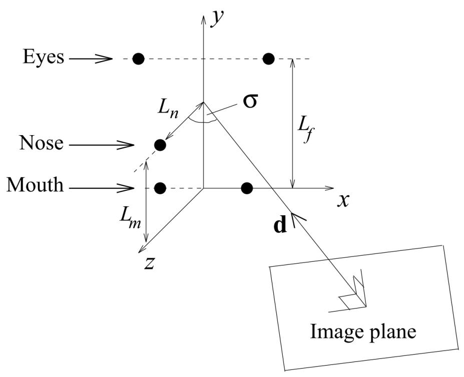

Figure 2.5: Slant Calculation of the face model [10].

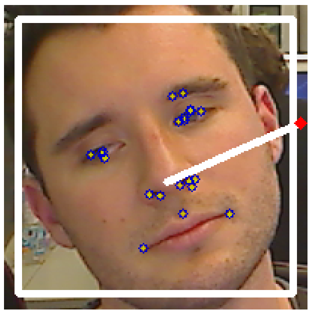

Figure 2.6: Feature Points, detected with the OPENCV framework [17].

3 Results

This straightforward implementation of our head pose estimation algorithm certainly has a few limitations. As we are tracking the movement of the feature points found during initialization, the range of movement from the initial pose is restricted to about $\pm30^\circ$ in azimuth and $\pm15^\circ$ in elevation. With the target scenario being a user sitting in front of a computer monitor, we found this restriction to be acceptable as we assume the user is looking onto the screen while using the system.
Furthermore, as the algorithm is relying on the accuracy of the estimation of the motion vectors, it is sensitive to occlusions, changes in illumination and rapid movements of the head. Again, we found this to be acceptable for our setting. For more demanding application scenarios, we are confident that these shortcomings might be overcome with more elaborate head pose estimation algorithms. Nevertheless, our implementation works with sufficient accuracy and is very fast so that we are able to process the camera frames in real time, i.e. faster than ($23\mathit{fps}$ and $43\mathit{ms}$), thus keeping the delay of the processing chain to a minimum ($7\mathit{fps}$).

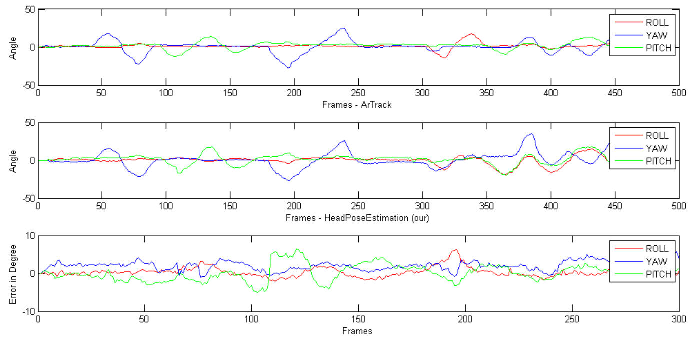

Figure 3.1: Mean Error in comparison with ar-track [22]

The results obtained from the professional ar-track [22] - pose estimation system compared with our scenario demonstrate its applicability to a real-world. The $Mean_{Errors}$ over a test sequence of 500 frames - 21 seconds

	ROLL	YAW	PITCH
$MEAN_{Error}$	1.4056	-0.9791	0.8163
STD_{Error}	5.2209	5.9485	3.0118

have been calculated.

4 Conclusion

The monocular image capture device does not require calibration and easily achieves real time performance, starts and recovers from failure automatically without any previous knowledge of the user's phenotype or location. It can operate in a wide range of constant lighting conditions, but handling with drastic changes it is still a problem since template images change considerably with varying illumination. In the current system, this problem is solved when our algorithm detects failure and automatically re-initialises to capture new template images under the changed illumination. In future work, a real head motion model can be used to improve the pose estimation by predicting the next pose position from past measurements. This would have the effect of smoothing the pose measurements and removing outliers. More robust feature tracking may be achieved by matching the template images against the regions of interest at various scales and orientations. This approach could be extended to any affine transformation which would further improve the possibility of finding a good match, thus enhancing the robustness of the tracker.

Bibliography

[1] E. Murphy-Chutorian and M.M. Trivedi. Head pose estimation in computer vision: A survey. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 31(4):607–626, 2009.

[2] M. D. Breitenstein, D. Kuettel, T. Weise, L. Van Gool, and H. Pfister. Real-time face pose estimation from single range images. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–8. IEEE, 2008.

[3] Y. Matsumoto and A. Zelinsky. An algorithm for real-time stereo vision implementation of head pose and gaze direction measurement. In *Automatic Face and Gesture Recognition, 2000. Proceedings. Fourth IEEE International Conference on*, pages 499–504. IEEE, 2000.

[4] R. Newman, Y. Matsumoto, S. Rougeaux, and A. Zelinsky. Real-time stereo tracking for head pose and gaze estimation. In *Automatic Face and Gesture Recognition, 2000. Proceedings. Fourth IEEE International Conference on*, pages 122–128. IEEE, 2000.

[5] Q. Ji and X. Yang. Real-time eye, gaze, and face pose tracking for monitoring driver vigilance. *Real-Time Imaging*, 8(5):357–377, 2002.

[6] S.O. Ba and JM. Odobez. A probabilistic framework for joint head tracking and pose estimation. In *Pattern Recognition, 2004. ICPR 2004. Proceedings of the 17th International Conference on*, volume 4, pages 264–267. IEEE, 2004.

[7] S Ba and JM Odobez. From camera head pose to 3d global room head pose using multiple camera views. In *Proc. Intel. Workshop Classification of Events Activities and Relationships*, 2007.

[8] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*, volume 1, pages I–511. IEEE, 2001.

[9] R. Lienhart and J. Maydt. An extended set of haar-like features for rapid object detection. In *Image Processing. 2002. Proceedings. 2002 International Conference on*, volume 1, pages I–900. IEEE, 2002.

[10] A. Gee and R. Cipolla. Determining the gaze of faces in images. *Image and Vision Computing*, 12(10):639–647, 1994.

[11] D.F. Dementhon and L.S. Davis. Model-based object pose in 25 lines of code. *International journal of computer vision*, 15(1-2):123–141, 1995.

[12] J. Chen and B. Tiddeman. A real-time stereo head pose tracking system. In *Signal Processing and Information Technology, 2005. Proceedings of the Fifth IEEE International Symposium on*, pages 258–263. IEEE, 2005.

[13] R.S. Feris, E. Teófilo, and R.C. Marcondes Jr. Detection and tracking of facial features in video sequences. In *MICAI 2000: Advances in Artificial Intelligence*, pages 127–135. Springer, 2000.

[14] K. Huang and M.M. Trivedi. Robust real-time detection, tracking, and pose estimation of faces in video streams. In *Pattern Recognition, 2004. ICPR 2004. Proceedings of the 17th International Conference on*, volume 3, pages 965–968. IEEE, 2004.

[15] P. Viola and M.J. Jones. Robust real-time face detection. *International journal of computer vision*, 57(2):137–154, 2004.

[16] M. Castrillón-Santana, O. Déniz-Suárez, L. Antón-Canalís, and J. Lorenzo-Navarro. Face and facial feature detection evaluation performance evaluation of public domain haar detectors for face and facial feature detection. 2008.

[17] G. Bradski and A. Kaehler. Learning opencv: Computer vision with the opencv library. 2008. *Oreilly & Associates Inc., USA*.

[18] M. Schreiber and M. Rothbucher. Erfassung der Kopfbewegung mit einer Webcam zur Verbesserung der HRTF Soundsynthese einer Telekonferenzanwendung. *Lehrstuhl für Datenverarbeitung, Technische Universitaet Muenchen*, 2010.

[19] J. Shi and C. Tomasi. Good features to track. In *Computer Vision and Pattern Recognition, 1994. Proceedings CVPR'94., 1994 IEEE Computer Society Conference on*, pages 593–600. IEEE, 1994.

[20] B.D. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *Proceedings of the 7th international joint conference on Artificial intelligence*, 1981.

[21] O. Kwon, J. Chun, and P. Park. Cylindrical model-based head tracking and 3d pose recovery from sequential face images. In *Hybrid Information Technology, 2006. ICHIT'06. International Conference on*, volume 1, pages 135–139. IEEE, 2006.

[22] Advanced realtime tracking gmbh, http://www.ar-tracking.com.