Kornél Bertók, Levente Sajó & Attila Fazekas:
A robust head pose estimation method based on POSIT algorithm
Argumentum 7 (2011), 348-356
Debreceni Egyetemi Kiadó

Working paper

Kornél Bertók, Levente Sajó & Attila Fazekas

A robust head pose estimation method based on POSIT algorithm

Abstract

Estimating the head pose is an important ability of a computer when interacting with humans because the head pose usually indicates the focus of attention (Bennewit, Faber, Joho, Schreiber & Behnke 2005). In this paper, we present a novel method to estimate head pose over low-resolution web camera images. Our approach proceeds in three stages. First, a face detector localizes faces on the input image. Then, classifiers trained with AdaBoost using Haar-like features, detect distinctive facial features namely the mouth, the nose tip and the eyes. Based on the positions of these features, finally the POSIT algorithm estimates the three continuous rotation angles and the translation vector, what we use later fore head pose modeling. Since we have a compact representation in case of faces – using only few distinctive features –, so thus our approach is computationally highly efficient. As we show in experiments with standard databases as well as with real-time image data, our system locates the distinctive features with a high accuracy and provides robust estimating of head pose.
Keywords: Head pose estimation, facial features detection, human-computer interaction

1 Introduction

Successful interaction with humans requires robust and accurate perception and tracking of their body parts to infer nonverbal signals of attention and intention. In order to establish a joint focus of attention, estimating the head pose is critical since it usually is the same with the gaze direction. Furthermore, head pose estimation is also essential for analyzing complex meaningful gestures such as pointing gestures or head nodding and shaking.

In this paper, we present a novel method to estimate head pose over low-resolution web camera images. Modeling of head pose performs in the three-dimensional space by three Euler angles of rotation around three axes orthogonal to each other, and three orthogonal translation directions (see Figure 1). While most of the existing approaches for head pose estimation deal only with poses that vary around the vertical axis, our system provides continuous head pose estimates in all three rotations and translation directions. We propose a method which is based on distinctive features namely the eyes, the nose tip and the center of mouth. Our approach proceeds in a hierarchical structure. Starting with face detection, followed by the extraction of the distinctive facial features within the bounding box of the detected face. Finally, it refines the pose estimate by considering the positions of the features. By using few local features instead of the whole sub-image containing the face, we achieve a compact representation that allows for training a computationally efficient estimator. Using

Kornél Bertók, Levente Sajó & Attila Fazekas:
A robust head pose estimation method based on POSIT algorithm
Argumentum 7 (2011), 348-356
Debreceni Egyetemi Kiadó

facial components also contributes to the robustness of the system since the appearance of a single facial feature varies less under different illumination conditions than, the appearance of the whole face.

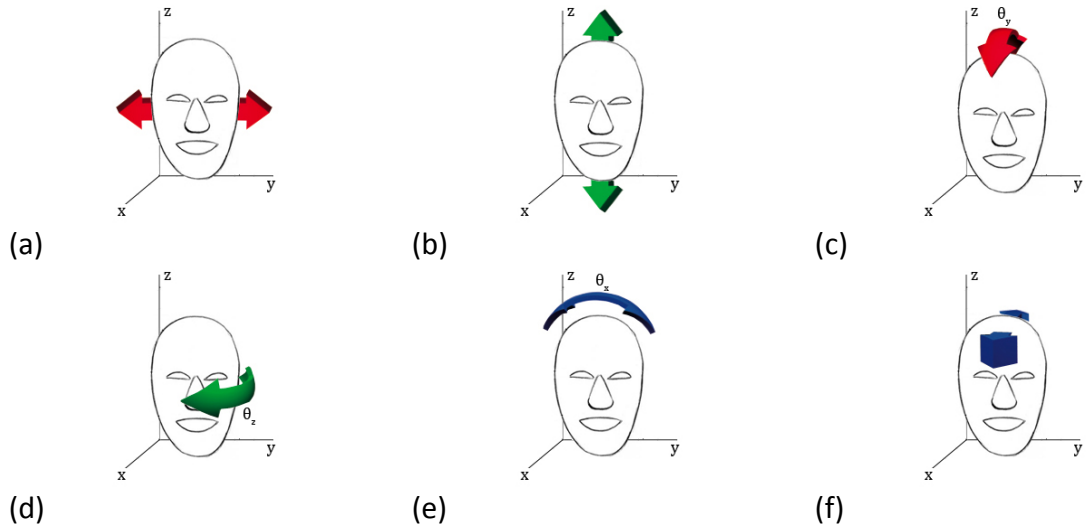

Figure 1: he six degrees of freedom of the head pose. (a) moving left and right, (b) moving up and down, (c) pitch angle denoted as , (d) yaw angle denoted as , (e) roll angle denoted as , (f) moving forward and backward.

We construct the feature detectors using AdaBoost with Haar-like features (Viola & Jones 2001). As we show in our experiments, our detectors accurately locate the distinctive facial features. Based the position of the features the POSIT algorithm was used to estimate the three rotation and the translation vector of the head pose.

This paper is organized as follows. The next section reviews related work. Section 3. presents the AdaBoost algorithm which we use to train our feature detectors. Section 4. defines the facial features and their search areas and describes the training data and the application of the detectors. Section 5. to the POSIT algorithm and finally, Section 6. presents the experimental results.

2 Related work

In the last few years, many research has been focused on head pose estimation based on low-resolution images. Existing methods can be categorized in appearance-based and model-based methods. Our head pose estimation system presented in this paper, combines ideas from these approaches mentioned below and combines their advantages.

Appearance-based techniques use the whole sub-image containing the face. Most of them concentrate on face detection and consider the pose estimation problem as a classification problem. The range of head orientations is divided into a limited number of classes and classifiers for each class are trained. The number of the classes defines the accuracy of the final pose estimation that can be achieved. Among these techniques: the statistical method of Schneiderman and Kanade (Schneiderman & Kanade 2004), and the approach proposed by Meynet et al. (Meynet, Arsan, Mota & Thiran 2007) who train a tree of classifiers by

hierarchically sub-sampling the pose space, and the technique of Li and Zhang (Li & Zhang 2004) who apply a detection pyramid which contains classifiers with increasingly finer resolution. Using only a limited number of classes for the head orientation, however, is not sufficient to recognize head gestures like nodding or shaking. Further appearance-based approaches are the systems developed by Stiefelhagen (Stiefelhagen 2004) and Rae and Ritter (Rae & H. 1998) which are based on neural networks. Appearance-based techniques are quite efficient regarding computation time. However, the mentioned approaches do not yield estimates for all three rotation angles. They concentrate on estimating the yaw and the pitch angle only. Our system, in contrast, provides accurate and continuous head pose estimates in all three rotations and translation directions.

Model-based approaches for head pose estimation use a geometric model of the face. For example, the methods proposed by Stiefelhagen et al. (Stiefelhagen, Yang & Waibel 1996) and Gee and Cipolla (Gee & Cipolla 1994) extract a set of facial features such as eyes, mouth, and nose and map the features onto the 3D model using perspective projection. Schödl et al. (Schödl & Haro 1998) combine the 3D model with a texture. They transform an extracted texture of the face to the frontal view and project it onto the model. Dornaika and Ahlberg (Dornaika & Ahlberg 2004) apply an active appearance model and use a very detailed description of the contours and features of the face. The disadvantage of model-based approaches, however, is that they are computationally more expensive and furthermore, most of them need to be hand-initialized. The head pose estimation system presented in this paper, combines ideas from several approaches mentioned above and combines their advantages. Our approach first localizes faces in the image using classifiers for frontal faces. Then, we refine the pose estimate using a feature-based technique. Instead of explicitly finding correspondences with a 3D head model, we learn the correlations between the positions of facial features.

Facial feature detection itself is a difficult task. In contrast to other methods that use low-level edge features or simple grayscale features (Gee & Cipolla 1994; Stiefelhagen, Yang & Waibel 1996) to locate facial features, we apply a reliable and fast appearance-based technique.

3 Object detection with AdaBoost

We trained classifiers for the facial features using AdaBoost, which is a supervised learning algorithm. Boosting refers to the concept of building a strong, highly accurate classifier by combining weak, not very accurate classifiers. In this work, we apply the adaptive boosting variant of the algorithm AdaBoost which was proposed by Freund and Schapire (Freund & Schapire, 1997).

The algorithm takes as input a training set $x_1, y_1, \dots, (x_m, y_m)$ where each x_i belongs to some domain or instance space X and each label y_i is in some label set Y. In this paper, we assume $Y = \{0; 1\}$. AdaBoost calls a given weak or base learning algorithm repeatedly in a series of rounds $t = 1, \dots T$. One of the main ideas of the algorithm is to maintain a distribution or set of weights over the training set. The weight of this distribution on training example i on round t is denoted $D_t(i)$. Initially, all weights are set equally, but on each round, the weights of incorrectly classified examples are increased so that the weak learner is forced to focus on the hard examples in the training set. The AdaBoost selects a weak classifier

Kornél Bertók, Levente Sajó & Attila Fazekas:
A robust head pose estimation method based on POSIT algorithm
Argumentum 7 (2011), 348-356
Debreceni Egyetemi Kiadó

$h_t: X \rightarrow \{0,1\}$ that best separates the positive and the negative examples with respect to the current distribution D_t. The goodness of a weak hypothesis is measured by its error ε_t.

Notice that the error is measured with respect to the distribution D_t on which the weak learner was trained. In practice, the weak learner may be an algorithm that can use the weights D_t on the training examples. Alternatively, when this is not possible, a subset of the training examples can be sampled according to D_t, and these (unweighted) resampled examples can be used to train the weak learner. In this way, in the next round the algorithm is forced to concentrate on the difficult examples which have been incorrectly classified before. The final strong classifier h is a weighted majority vote of the T best weak classifiers. The weight of a hypothesis h_t is larger the smaller its error ε_t is. We apply the variant of Viola and Jones (Viola & Jones, 2001) in which a weak classifier h_j is built from a single scalar feature f_j:

$$h_j x = \begin{matrix} 1 if p_j f_j x < p_j \theta_j \\ 0 otherwise \end{matrix},$$

where, θ_j is a threshold and the parity p_j represents the direction of the inequality. For each weak classifiers h_j optimal values θ_j and p_j are determined so that a minimum number of training examples is misclassified.

For the construction of the classifiers, we use Haar-like features (Lienhart & Maydt, 2002; Viola & Jones, 2001). When applied to an image patch, the value of each Haar-like feature is computed very efficiently using the differences of the sum of the pixel values within neighboring rectangular regions. In order to reduce computation time, Viola and Jones proposed to use a cascade of classifiers. To account for the fact that the majority of sub-windows in an image are negatives, the detector is constructed so as to process negative instances as efficiently as possible. Examples that are evaluated as positives at some stage of the cascade are processed at the next stage, while examples that are classified as negatives are immediately rejected. To find a trade-off between efficiency and accuracy and to meet given detection rates, constraints are imposed on the individual stage classifiers. New layers are added to the cascade until the overall target detection rate is reached.

4 Facial feature detection

We apply a face detection system that is also based on the AdaBoost algorithm and uses a boosted cascade of the same Haar-like features. The system works fast, has high detection rates, and yields accurate positions of the faces. We use one trained cascade (for frontal faces) that is provided by Intel's OpenCV library (Intel® IPP 2010).

Given a detected face in the image, we localize distinctive facial features within the corresponding bounding box. We use the following four features: left eye, right eye, nose tip, center of the mouth. If the yaw angle gets bigger, a part of the face becomes occluded. Therefore, for faces, we use only the features from the entirely visible part of the face, i.e., one of the eyes, the nose, one of the mouth, and additionally, the ear (if not covered by hair). Due to the symmetry of faces, we train feature detectors only for the left part of the face. By flipping the image patches and applying the detectors for the left features, the right parts can be detected.

To focus the search for a feature to a small region of the face which is most likely to contain the particular feature, we define individual search areas for the features in a scale-invariant face bounding box. Figure 2 shows results from the facial feature detectors for the frontal views. As explained above, we restrict the search to the expected location areas of the specific features in the face. We shift the search window by one pixel each time so that the detector examines all locations within the particular search region. The size of the facial features is correlated to the face size, given the extracted face bounding box. Therefore, during the search, we only consider a small number of scales of the image patch.

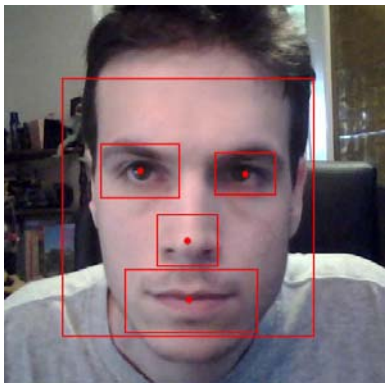

Figure 2: The detected face and the facial feature points with their search windows.

5 Pose estimation

Computation of the position and orientation of an object (object pose) using images of feature points when the geometric configuration of the features on the object is known (a model) has important applications, such as calibration, cartography, tracking and object recognition. In this chapter, we describe a method for finding the pose of faces from a single image. We assume that we can detect and match in the image four or more non-coplanar feature points (i.e. the eyes, the nose, and the mouth) of the faces, and that we know their relative geometry on the faces.

The method combines two algorithms, the first algorithm: POS (Pose from Orthography and Scaling) approximates the perspective projection with a scaled orthographic projection and finds the rotation matrix and the translation vector of the faces by solving a linear system. The second algorithm: POSIT (POS with ITerations) uses in its iteration loop the approximate pose found by POS in order to compute better scaled orthographic projections of the feature points, and then applies POS to these projections instead of the original image projections. POSIT converges to accurate pose measurements in a few iterations, POSIT can be used with many feature points at once for added insensitivity to measurement errors and image noise. Compared to classic approaches making use of Newtons method, POSIT does not require starting from an initial guess, and computes the pose using an order of magnitude fewer floating point operations, it may therefore be a useful alternative for real-time operation.

5.1 POS algorithm

The method relies on linear algebra techniques and is iterative like the methods of Tsai (Tsai 1987), Lowe (Lowe 1991; Lowe 1985) and Yuan (Yuan 1989), but it does not require an initial pose estimate and does not require matrix inversions in its iteration loop. At the first iteration step, the method finds an approximate pose by multiplying an object matrix (which depends only on the distribution of feature points on the object and is precomputed) by two vectors (which depend only on the coordinates of the images of these feature points). The two resulting vectors, once normalized, constitute the first two rows of the rotation matrix, and the norms of these vectors are equal to the scaling factor of the projection, which provides the translation vector. These operations amount to assuming that the involved image points have been obtained by a scaled orthographic projection (SOP in the following). We refer to this part of the algorithm as POS (Huttenlocher & Ullman 1988; Tomasi 1991; Ullman & Basri 1991).

5.2 POSIT algorithm

The next iterations applies exactly the same calculations, but with corrected image points. The basic idea is that since the POS algorithm requires an SOP image instead of a perspective image to produce an accurate pose, we have to compute SOP image points, using the pose found at the previous iteration. The process consists in shifting the feature points of the object in the pose just found, to the lines of sight (where they would belong if the pose was correct) and obtains a scaled orthographic projection of these shifted points. We call this iterative algorithm POSIT (DeMenthon & Davis 1995). Four or five iterations are typically required to converge to an accurate pose.

For N matching between object points and image points, POSIT requires around 24N arithmetic operations and two square root calculations per iteration. For 8 feature points and four iteration steps, around 800 arithmetic operations and 8 square roots are needed.

6 Results and Conclusion

We evaluated the performance of our pose estimation system on a separate test set. The faces and facial features were automatically detected by our detectors. For testing POSIT algorithm were used for pose estimation, we used all correctly detected faces with a full set of facial features. Because the development is not finished yet, we could not make full performance test, but we used a face database with 50 images for evaluation of the pose estimator. We experienced that the head pose of the faces seen on the pictures is correspond to all intents and purposes more than 90% with the head pose that compute the application (see Figure 3). The only limitation of the algorithm is that the profile poses are not approximated with a high accuracy, because we only use a face detector for frontal faces.

Additionally, we will perform qualitative experiments to evaluate the capability of our approach to estimate the head pose from images in real-time. Using a standard webcam with a resolution of 640×480 pixels, we currently achieve a rate of 10 fps on a standard PC. To improve the robustness of our system and to smooth the estimated angles, we will apply independent Kalman filters to track each facial feature and each rotation angle over time.

In this paper, we presented a feature-based system that estimates the head pose from low-resolution images. Our system works in three stages. First, a face-detector detects the faces on the input image. Then, classifiers detect distinctive features within the face. The classifiers for

the individual facial features are learned using AdaBoost with Haar-like features. Given the positions of the individual facial features, the POSIT algorithm finally estimates the three rotation angles and the two translation directions of the head pose. Since we use local features instead of the whole sub-image containing the face, we have a compact representation which results in a computationally efficient estimator. Our system yields continuous estimates of all three rotation angles and two directions.

Figure 3: Qualitative facial features detection and pose estimation results.

References

Bennewit, M., Faber, F., Joho, D., Schreiber, S. & Behnke, S. (2005): Towards a humanoid museum guide robot that interacts with multiple persons. *IEEE/RSJ Int. Conf. on Humanoid Robots (Humanoids)* .

DeMenthon, D. & Davis, L.S. (1995) : Model-Based Object Pose in 25 Lines of Code. *International Journal of Computer Vision* 15(1-2), 123-141.

Dornaika, F. & Ahlberg, J. (2004): Fast and reliable active appearance model search for 3D face tracking. *IEEE Transactions on Systems, Man and Cybernetics, Part B* 34(4), 1838-1853.

Freund, Y. & Schapire, R. (1997): A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences* 55(1), 119-139.

Gee, A.H. & Cipolla, R. (1994): Determine the gaze of faces in images. *Image and Vision Computing* 12(10), 639-647.

Huttenlocher, D.P., & Ullman, D.S. (1990): Recognizing Solid Objects by Alignment. *International Journal of Computer Vision* 5(2), 195-212.

Intel® IPP. (2010). Forrás: Open Source Computer Vision Library: http://sourceforge.net/projects/opencvlibrary/

Li, S.Z. & Zhang, Z.Q. (2004): Floatboost learning and statistical face detection. *IEEE Transactions Pattern Analysis and Machine Intelligence* 26(9), 1112-1123.

Lienhart, R. & Maydt, J. (2002): An Extended Set of Haar-like Features for Rapid Object Detection. *IEEE International Conference on In Image Processing (2002)*, 900–903.

Lowe, D.G. (1991): Fitting Parameterized Three-Dimensional Models to Images. *IEEE Transactions Pattern Analysis and Machine Intelligence* 13(5), 441–450.

Lowe, D.G. (1985): Perceptual Organization and Visual Recognition. *Kluwer Academic Publisher*.

Meynet, J., Arsan, T., Mota, J.C. & Thiran, J. (2007): Fast multiview tracking with pose estimation. *Proceeedings of the 16th European Signal Processing Conference (2008)*, 1-12.

Rae, R., & Ritter, H.J. (1998). Recognition of human head orientation based on artificial neural nets. *IEEE Computational Intelligence Society* 9(2), 257–265.

Schneiderman, H. & Kanade, T. (2004): Object detection using the statistics of parts. *International Journal of Computer Vision* 56(3), 151-177.

Schödl, A. & Haro, A.E. (1998): Head tracking using a textured polygonal model. *Technical Report GIT-GVU-98-24, Georgia Institute of Technology* .

Stiefelhagen, R. (2007): Estimating head pose with neural networks. *Lecture Notes in Computer Science* 4122/2007, 291-298.

Stiefelhagen, R., Yang, J. & Waibel, A. (1996): A model-based gaze tracking system. *IEEE International Joint Symposia on Intelligence and Systems*, 304-310.

Tomasi, C. (1991): Shape and Motion from Image Streams: A Factorization Method. *Technical Report CMU-CS-91-172 Carnegie Mellon University*.

Tsai, R.Y. (1987): A versatile Camera Calibration Technique for High-Accuracy 3D Machine Vision Metrology Using Off-the-Shelf TV Cameras and Lenses. *IEEE J. Robotics and Automation* 3, 323-344.

Ullman, S. & Basri, R. (1991): Recognition by Linear Combinations of Models. *IEEE Trans. on Pattern Analysis and Machine Intelligence* 13, 992-1006.

Viola, P. & Jones, M. (2001): Rapid object detection using a boosted cascade of simple features. *In Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition* 1, 511-518.

Yuan, J.S. (1989): A General Photogrammetric Method for Determining Object Position and Orientation. *IEEE Trans. on Robotics and Automation* 5, 129-142.

Kornél Bertók
University of Debrecen, Faculty of Informatics
Department of Computer Graphics and Image Processing
H-4010 Debrecen, P.O Box 12
bertok.kornel@inf.unideb.hu

Levente Sajó
University of Debrecen, Faculty of Informatics
Department of Computer Graphics and Image Processing
H-4010 Debrecen, P.O Box 12
sajolevente@inf.unideb.hu

Dr. Attila Fazekas
University of Debrecen, Faculty of Informatics
Department of Computer Graphics and Image Processing
H-4010 Debrecen, P.O Box 12
attila.fazekas@inf.unideb.hu