Head Pose Estimation by Perspective-n-Point Solution Based on 2D Markerless Face Tracking

François Rocca, Matei Mancas, and Bernard Gosselin

University of Mons (UMONS), Faculty of Engineering (FPMs),
20, Place du Parc, 7000 Mons, Belgium
{francois.rocca,matei.mancas,bernard.gosselin}@umons.ac.be

Abstract. In this paper, we present an optimized implementation of automatic head direction extraction for a person placed in front of his webcam. The aim is to compute the different rotation angles of the head with a non-invasive and continuous tracking. Our method is based on 2D features tracking of the face with a low cost webcam. This information is associated to a set of points from a 3D head model by perspective-n-point solution to obtain pitch, roll and yaw. These results are then compared with a reference acquired with faceLAB, a robust markerless head tracker and eye tracking system.

Keywords: Face tracking, head pose estimation, markerless, perspective-n-point solution, faceLAB.

1 Introduction

Faces play a crucial role in human communication and we are increasingly brought to communicate using new technology, particularly in front of a screen (computer, television, tablet, smartphone, etc.). The real-time visual detection and tracking of faces and their pose is a topic of particular interest in the analysis of social interaction. Analyzing the behaviors of people requires to answer a set of typical questions: who are the people? Where are they? What are they doing? What are they looking at? And how are they interacting? Locating people and their faces allows us to address the first two questions and is usually the first step before answering the other questions.

Concerning the analysis of interactions between people, progress has been achieved through the development of 2D and 3D markerless tracking algorithms. For the whole body, cheap 3D cameras and software have been developed to automatically determine the position of the skeleton of the people.

However, the markerless analysis of real-time faces remains an open problem. Analysis systems to recognize facial expressions or emotions exist but they are generally very expensive. Most of them are marker-based and not easy to use. Therefore it is interesting to solve the problem of 2D face tracking using methods combining 2D and 3D information. Performing facial analysis based on cheap hardware easy to use is also important to make it accessible to everyone.

To achieve this goal, we have implemented a solution of head detection and pose estimation using a low-cost camera. This choice was made due to the number of electronic devices equipped with a camera. Moreover, TV manufacturers begin to integrate cameras into their new systems.

This paper is structured as follows. The second section provides information about the related work, section 3 details the implemented algorithm in two parts, the first one explains the face tracking and the second one the perspective-n-point solution. Section 4 relates the results of the experiment with a reference comparison. Finally we conclude at the fifth section.

2 Related Work

In the animation industry, head movements are almost exclusively captured with physical sensors and optical analyses. Physical sensors such as accelerometer, gyroscope and magnetometer are placed on the head to compute the head rotation [1] [2]. The other way is marker-based optical motion capture systems that are able to capture the subtlety of the motions. These systems enable us to obtain high definition facial motion data. Accurate tracking like the OptiTrack system requires multiple expensive cameras (at least three) and triangulation software to compute 3D facial tracking therefore these systems are very expensive [3]. Some cheaper methods like the Zigntrack equipment use colored dots and only one classical camera to track the face [4]. But all these methods are considered as invasive because they require to place a set of sensors on the person, which complicates their utilization.

Markerless tracking is another approach to face motion capture. A wide range of methods exists for markerless motion capture. Some markerless equipment uses infrared cameras to compute tracking of characteristic points. For example, faceLAB gives the head orientation and the position of lips, eyes and eyebrows [5]. But there are also algorithms using only a webcam. We can cite faceAPI [6] from the same company as faceLAB. A robust "FaceTracker" algorithm has been developed by Jason Saragih [7] based on the OpenCV library. We will use this algorithm to compute the markerless face tracking in section 3.

More recently, with the arrival of low cost depth sensor [8], more accurate solutions have emerged [9] [10]. Based on the use of depth maps, those methods are able to overcome known problems on 2D images as illumination or low contrast backgrounds. Many of these techniques are not real-time or require expensive hardware to obtain real-time results [11]. Another approach, based on 2D and 3D analysis as the method developed by Weise [12], provides robust and impressive results.

3 Head Pose Estimation

The aim of this work is to calculate the three angles of rotation of the head: pitch, roll and yaw (Figure 1). Our first step is to detect the user face and to follow specific features on this face. The coordinates of these features will be associated to a set of points from a 3D head model. Solving a PnP (Perspective-n-Point) problem based on the 2D-3D correspondence will be the next step to obtain the 3DoF (degrees of freedom).

Firstly we will explain the utilization of the face tracking algorithm, then we will explain the PnP problem and we will finish with the combination of these two methods to obtain the values of the 3DoF.

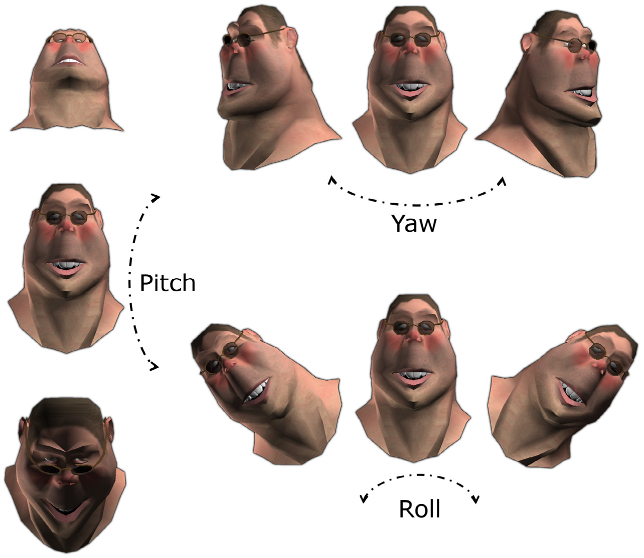


Fig. 1. The figure shows the three different degrees of freedom: pitch, roll and yaw. All the motions of head rotation can be obtained by combining these three basic movements.

3.1 Face Tracking

FaceTracker is a CLM-based C/C++ API for real-time generic non-rigid face alignment and tracking. The approach is an instance of the constrained local model with the subspace constrained mean-shifts algorithm as an optimization strategy [7]. FaceTracker allows the identification and localization of 66 landmarks. These 66 points can be assimilated to a facial mask allowing to track facial features like the edge of lips, facial contours, nose, eyes and eyebrows (Figure 2).

The advantage is that FaceTracker does not require specific training by the user before utilization: the algorithm makes an automatic detection of the user face based on a model pre-trained on database. FaceTracker is based on the OpenCV library. It is compatible with any camera. In our setup we use a 480X640 pixel webcam. The initialization of the algorithm is based on the Haar classifiers [13], thus the face tracking is optimal if the face is centered in front of the camera. We can also observe significant perturbations when an object starts occluding some landmarks or when head rotation is rapidly done with a wide angle.

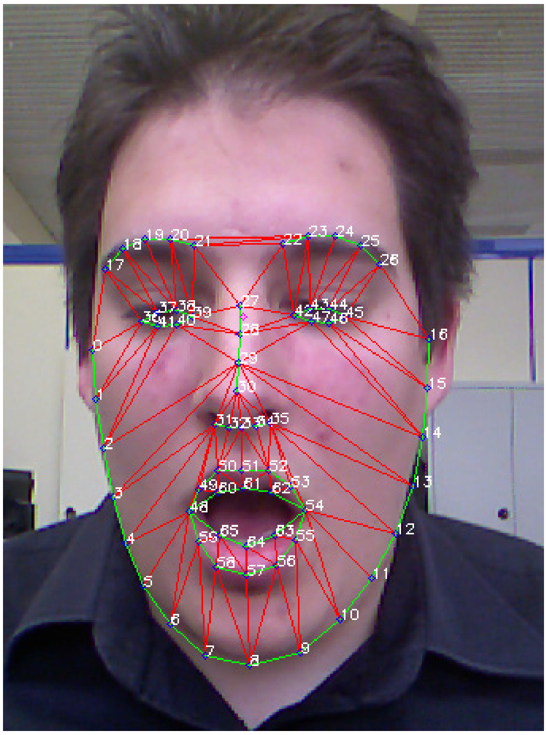


Fig. 2. FaceTracker detects in real-time the user face and sets a 66 points mask. Points 0 to 16 are used for lower facial contours, 17 to 21 and 22 to 26 for the right and left eyebrows, 27 to 35 for the nose, 36 to 41 and 42 to 47 for the right and left eyes, and 48 to 65 for the edge of lips.

3.2 Perspective-n-Point Problem

The perspective-n-point problem is a recurrent problem. Knowing a number of points (N) from an object in a 3D coordinate system and knowing the projection of these points on a plane in another coordinate system (2D image), it is possible to find the transformation between the two coordinate systems. The relation, between the 2D and the 3D coordinate systems for one point, is given as follows [14]:

$$m' = A[R|t]M' \tag{1}$$

Or,

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \tag{2}$$

- (u, v) are the coordinates of the projection point in pixels
- A is a camera matrix, or a matrix of intrinsic parameters
- (cx, cy) is a principal point that is usually at the image center
- fx, fy are the focal lengths expressed in pixel units
- (X, Y, Z) are the coordinates of a 3D point in the world coordinate space
- And [R|t] is the rotation and transformation matrix

With a set of N 2D coordinates and the 3D correspondences, and knowing the camera matrix A, it is possible to find rotation-translation matrix. The PnP has already been used to find the 3DoF of the head where the 2D coordinates are extracted manually from an image [15].

The 3D values are based on a 3D model of a head. Results are computed with OpenCV library and the system provides a visualization of the 3D projection on the 2D source image. In the next step we will explain how we combine this method with the face tracking to compute markerless real-time head pose estimation.

3.3 Solving PnP Problem Using 2D Coordinates from the Face Tracking

For the 2D coordinates, we take 7 points among the 66 points from FaceTracker: points 0 and 16 on the temples, 39 and 42 for the inner corners of the eyes, 48 and 54 for the corners of the lips and 30 for the nose. These points were chosen because they are far enough and stable regardless of the expressions and movements of the face.

The geometry of the 3D model should be similar to the one of the person in front of the camera. Indeed, a large difference in geometry does not allow correctly solving the PnP and minimizing the error between the projections coordinates of the 3D model on the plane and the values obtained with the face tracking. The 3D model used during our experiment is similar to the user head geometry. A display window shows the 2D face tracking and the 3D head model superposed on the user face by OpenGL (Figure 3).

Once the seven 2D and 3D coordinates are set, and the camera matrix found, we can calculate the matrix of rotation and translation of 3D model by reporting the data from the face tracking. The pitch, roll and yaw can directly be extracted from the rotation matrix.

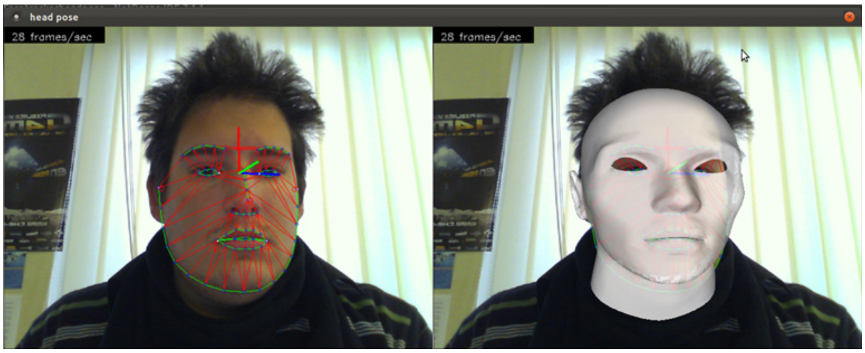
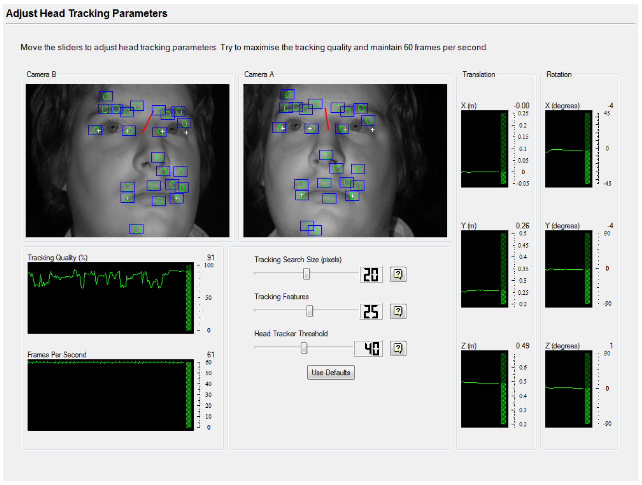


Fig. 3. On the left we have the face tracking and on the right the projection of the 3D model is correctly superposed on the points from the face tracking. The whole system works in real-time.

4 Experiment

In this section we will describe the results and compare them with a reference obtained with the faceLAB system [5]. FaceLAB produces markerless accurate data in real-time for eyes and head tracking at about 60 frames per second (Figure 4). The system works with stereoscopic vision by two infrared cameras and one infrared projector, which makes the system more robust and less sensitive to lighting conditions. We have chosen faceLAB system as a reference because the accuracy measurement is about +/- 1° of rotational error and tracking range for head rotation is about +/-90° around the y-axis and +/- 45° around the x-axis. But this choice has been done especially to compare two markerless and real-time methods.


Adjust Head Tracking Parameters

Move the sliders to adjust head tracking parameters. Try to maximise the tracking quality and maintain 60 frames per second.

Tracking Search Size (pixels) 20

Tracking Features 25

Head Tracker Threshold 40

Fig. 4. FaceLAB shows the features tracking for each infrared camera. The software gives also the tracking quality index, the frames per second (about 60fps), and the 3 rotation angles.

4.1 Experimental Protocol

We have chosen to perform the recording of the two methods in parallel under normal conditions: correct face lighting without infrared component to not disturb faceLAB.

We made several recordings for a total duration of 5 minutes. Our head pose estimation system is slower than faceLAB with about 20fps (from 12 to 28fps). The computing time per frame is about 50ms by single thread on a Linux OS with Intel Core i7 2.3GHz and 8GB of RAM. Two users have participated in recording and for each user a 3D model associated with the geometry of the user head was used.

Movements performed are conventional rotations when we are facing a screen (pitch, roll, and yaw; combination of these movements; slow and fast rotation) (Figure 5).

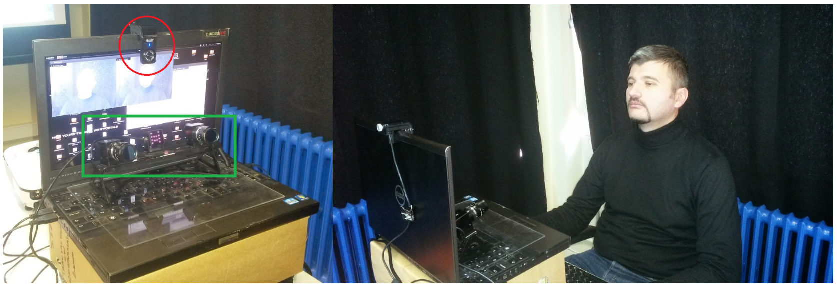


Fig. 5. On the left we have the camera for our face tracking system (in red circle) and the faceLAB system of cameras (in green square). On the right, we have a user in front of the system, ready for recording.

4.2 Results

After having synchronized the results obtained by our system and our reference, as the two sampling frequencies are different, we have interpolated reference values to obtain points at the same times as the points of our system. To make the comparison between our system and the reference computed with faceLAB, we use two tools: the Root Mean Square Error (RMSE) and the correlation.

Root Mean Square Error is given by:

$$\sqrt{\sum \frac{(y_{pred} - y_{ref})^2}{N}} \tag{3}$$

With the predicted values obtained by our system y_{pred}, the values from the reference y_{ref} and the number of values N.

The correlation, based on the Pearson's coefficient, is given by:

$$\frac{\sum (y_{ref} - \overline{y}_{ref})(y_{pred} - \overline{y}_{pred})}{\sqrt{\sum (y_{ref} - \overline{y}_{ref})^2}\sqrt{\sum (y_{pred} - \overline{y}_{pred})^2}} \tag{4}$$

The figure below shows the result by the superposition of our values in red and the reference in blue for one recording session. The X rotation is the pitch, the Y is the yaw and the Z is the roll. We can see that the two curves are similar for each rotation.

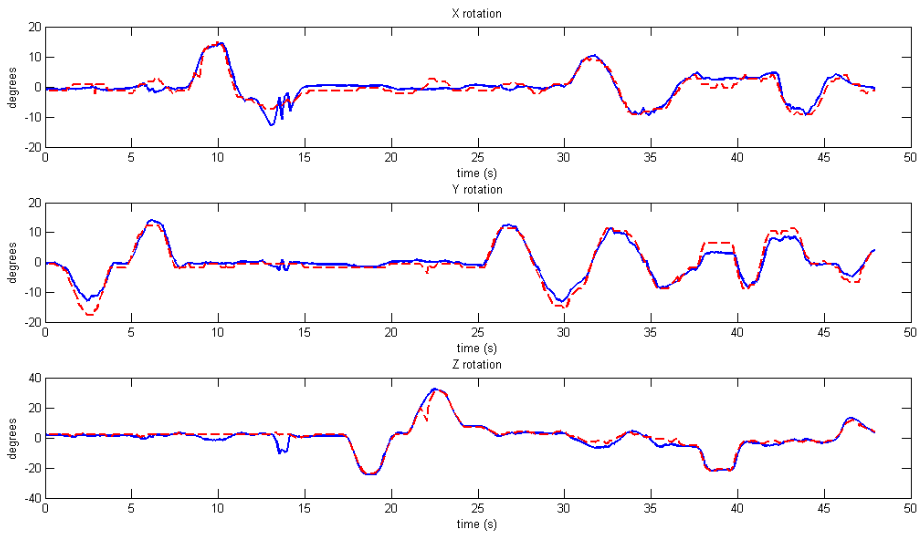


Fig. 6. Reference value in blue and our head pose value in dashed red are displayed on the same figure. First is pitch, second is yaw and last is roll.

The Values of the RMSE and the correlation are given in the Table 1:

Table 1. Root Mean Square Error and correlation from comparison of signals from Figure 6

	X-rotation / Pitch	Y-rotation / Yaw	Z-rotation / Roll
RMSE (in degrees)	1.62	1.83	2.09
Correlation	0.94	0.97	0.97

These results show that the RMSE are very small, between 1 and 2 degrees of error. This is very low considering that the precision of our reference is +/- 1°. The values of the correlation are very close to 1, which shows that the curves are highly correlated.

Then we have analyzed the whole recorded sessions. The other sessions have different duration, bigger angular differences (Figure 7) and variation of the speed movement (Figure 8). The results are given in Table 2 for the RMSE and Table 3 for the correlation.

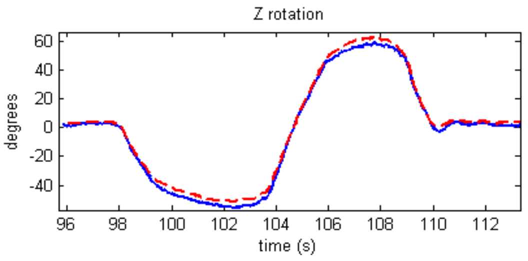


Fig. 7. Sample of Z rotation from -55° to 62 °, with reference value in blue and our head pose value in dashed red from session 3

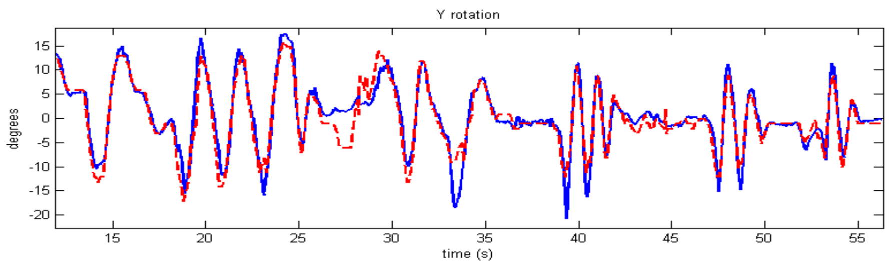


Fig. 8. Sample of Y rotation with rapid rotation, with reference value in blue and our head pose value in dashed red from session 4

Table 2. Root Mean Square Error for each recorded session

	X-rotation / Pitch	Y-rotation / Yaw	Z-rotation / Roll
Session 1(~60 s)	1.62	1.83	2.09
Session 2(~36s)	2.42	0.92	1.59
Session 3(~116 s)	5.08	6.64	2.8
Session 4(~62 s)	3.92	2.36	3.9

Table 3. Correlation for each recorded session

	X-rotation / Pitch	Y-rotation / Yaw	Z-rotation / Roll
Session 1(~60 s)	0.94	0.97	0.97
Session 2(~36s)	0.80	0.92	0.98
Session 3(~116 s)	0.75	0.83	0.99
Session 4(~62 s)	0.72	0.94	0.94

The results of the second session are similar to the first session with a lower correlation for the pitch. The last two recording sessions have mainly a decrease in the pitch correlation and an increase of the Root Mean Square Error. This is explained by the fact that the movements produced in sessions 3 and 4 (large angles and bigger speed) become more difficult to follow by the face tracking. Errors are mainly due to face tracking errors and tracking losses. The global RMSE is about 2.9°, not far from the faceLAB accuracy (+/-1°) and the global correlation is equal to 0.9 and is therefore quite good.

5 Conclusions

In this paper, we have presented an implementation of head pose estimation to compute the different rotation angles of the head of a person placed in front of his webcam. Our real-time non-invasive method is based on 2D features tracking of the face with a low cost webcam. We solve a perspective-n-point solution to obtain pitch, roll and yaw. These results are then compared with a reference obtained from faceLAB with relatively good results: a low Root Mean Square Error near to 2.9° and a correlation near to 0.9.

Acknowledgments. This work is supported by the Integrated Project LinkedTV (www.linkedtv.eu) funded by the European Commission through the 7th Framework Programme (FP7-287911).

References

1. Persa, S.-F.: Sensor fusion in head pose tracking (2006)
2. Emotiv EPOC headset Features, `http://emotiv.com/epoc/features.php`
3. OptiTrack, Optical motion tracking solutions, `http://www.naturalpoint.com/optitrack/`
4. ZignTrack. facial motion capture solution, `http://www.zigncreations.com/zigntrack.htm`
5. faceLAB 5. Face and eye tracking application, `http://www.seeingmachines.com/product/facelab/`
6. faceAPI. Markerless face tracking application, `http://www.seeingmachines.com/product/faceapi/`
7. Saragih, J.M., Lucey, S., Cohn, J.F.: Deformable model fitting by regularized landmark mean-shift. International Journal of Computer Vision 91(2), 200–215 (2011)
8. Microsoft Kinect sensor, `http://www.xbox.com/kinect`
9. Fanelli, G., Weise, T., Gall, J., Van Gool, L.: Real time head pose estimation from consumer depth cameras. In: Mester, R., Felsberg, M. (eds.) DAGM 2011. LNCS, vol. 6835, pp. 101–110. Springer, Heidelberg (2011)
10. Fanelli, G., Dantone, M., Gall, J., Fossati, A., Van Gool, L.: Random forests for real time 3d face analysis. International Journal of Computer Vision 101(3), 437–458 (2013)
11. Leroy, J., Rocca, F., Mancaş, M., Gosselin, B.: 3D Head Pose Estimation for TV Setups. In: Mancas, M., d' Alessandro, N., Siebert, X., Gosselin, B., Valderrama, C., Dutoit, T. (eds.) INTETAIN 2013. LNICST, vol. 124, pp. 55–64. Springer, Heidelberg (2013)
12. Weise, T., Bouaziz, S., Li, H., Pauly, M.: Realtime performance-based facial animation. ACM Trans. Graph. 30(4), 77 (2011)
13. Viola, P., Jones, M.J.: Robust real-time face detection. International Journal of Computer Vision 57(2), 137–154 (2004)
14. OpenCV documentation, `http://docs.opencv.org/modules/calib3d/doc/calib3d.html`
15. Shil, R.: Head Pose Estimation with OpenCV & OpenGL Revisited, `http://www.morethantechnical.com/2012/10/17/head-pose-estimation-with-opencv-opengl-revisited-w-code/`