Computer Vision and Image Understanding 180 (2019) 13–22

Contents lists available at ScienceDirect

Computer Vision and Image Understanding

journal homepage: www.elsevier.com/locate/cviu

Robust and accurate 2D-tracking-based 3D positioning method: Application to head pose estimation

Mikel Ariz *, Arantxa Villanueva, Rafael Cabeza *

Department of Electrical and Electronic Engineering, Public University of Navarra, Pamplona 31006, Spain

ARTICLE INFO

Communicated by Y. Tian

Keywords:
Pose estimation
Head tracking
Facial point detection and tracking
Outlier correction
POSIT

ABSTRACT

Head pose estimation (HPE) is currently a growing research field, mainly because of the proliferation of human–computer interfaces (HCI) in the last decade. It offers a wide variety of applications, including human behavior analysis, driver assistance systems or gaze estimation systems. This article aims to contribute to the development of robust and accurate HPE methods based on 2D tracking of the face, enhancing performance of both 2D point tracking and 3D pose estimation. We start with a baseline method for pose estimation based on POSIT algorithm. A novel weighted variant of POSIT is then proposed, together with a methodology to estimate weights for the 2D–3D point correspondences. Further, outlier detection and correction methods are also proposed in order to enhance both point tracking and pose estimation. With the aim of achieving a wider impact, the problem is addressed using a global approach: all the methods proposed are generalizable to any kind of object for which an approximate 3D model is available. These methods have been evaluated for the specific task of HPE using two different head pose video databases; a recently published one that reflects the expected performance of the system in current technological conditions, and an older one that allows an extensive comparison with state-of-the-art HPE methods. Results show that the proposed enhancements improve the accuracy of both 2D facial point tracking and 3D HPE, with respect to the implemented baseline method, by over 15% in normal tracking conditions and over 30% in noisy tracking conditions. Moreover, the proposed HPE system outperforms the state of the art on the two databases.

1. Introduction

People have always been able to provide and receive information from the movements of the head (i.e. position and orientation variations), which are a nonverbal form of communication that very often accompanies the speech. The interpretation of these head movements is crucial for understanding the intentions of other people in everyday human interactions, since the head pose gives direct information of people's attention target. In the computer vision field, head pose estimation (HPE) is understood as the computation of the head position and orientation with respect to a given coordinate system, usually a camera that is considered to be the origin of the world coordinate system (WCS). This computation is thus accomplished using the two-dimensional images obtained from that camera. Full orientation in 3D is determined by six degrees of freedom, namely the three rotation angles (roll, yaw, pitch) and the three translations (T_x, T_y, T_z) along the three spatial axes that define the WCS.

HPE offers a wide range of applications such as human behavior analysis (Ba and Odobez, 2011; Subramanian et al., 2013), driver assistance systems (Tawari et al., 2014) or gaze estimation systems (Valenti et al., 2012). There is a great amount of information contained in head gestures, and HPE is a rich form of communication considered an important bridge for the interaction between humans and computers (Murphy-Chutorian et al., 2009). Human–computer interaction (HCI) has experienced an important rise in the past decade due to its multidisciplinary nature and its application in a vast number of fields, such as assistive technologies, artificial intelligence or control of mobile devices. Lately, research on HCI has focused on developing control methods without the need of touch, such as hand gesture recognition (Wang et al., 2013; Valstar et al., 2017), head tracking (Belhumeur et al., 2013; Qiao et al., 2008), or gaze estimation (Navallas et al., 2011), among others. HCI based on eye-control (Yuan et al., 2013) is rapidly evolving and has a great potential, mainly due to the enormous spread of mobile devices. Gaze tracking systems suffer in unconstrained environments because they are very sensitive to head motion, and HPE has become a key point for successful gaze estimation. It has already been demonstrated that the head pose is as important as the eye location in order to determine the gaze direction (Langton et al., 2004), and gaze tracking and HPE are often combined in the search for better gaze estimation accuracy without constraining user movements, which also increases the application range (Valenti et al., 2012; Cazzato et al., 2014). When doing

* Corresponding authors.
E-mail addresses: mikelarizgalilea@gmail.com (M. Ariz), rcabeza@unavarra.es (R. Cabeza).

https://doi.org/10.1016/j.cviu.2019.01.002
Received 8 November 2017; Received in revised form 21 November 2018; Accepted 6 January 2019
Available online 15 January 2019
1077-3142/© 2019 The Authors. Published by Elsevier Inc. This is an open access article under the CC BY-NC-ND license (http://creativecommons.org/licenses/by-nc-nd/4.0/).

gaze estimation using off the shelf cameras, where the irises are usually not represented in great detail, it becomes of critical importance to have an accurate HPE system (Valenti et al., 2012). Moreover, an automatic estimation of the orientation of the head, eventually strengthened by a more accurate gaze estimation, is a key point in various applications of assistive technologies, such as autism diagnosis, monitoring of social development, depression detection and human behavior analysis (Leo et al., 2017).

HPE systems usually estimate the head pose from a set of 2D–3D facial point correspondences, what is known as the Perspective-n-Point (PnP) problem when the intrinsic camera parameters are known. There are several geometric algorithms in the literature to estimate the pose from such set of correspondences. Pose from Orthography and Scaling with Iterations (POSIT) (DeMenthon and Davis, 1995) requires a set of at least four non-coplanar points to estimate the pose of an object using projective geometry. POSIT consists of two steps: it initially assumes a weak perspective model for the pose calculation, in which the points from the object are projected onto the image as Scaled Orthographic Projections (SOP), and then iterates by shifting the points of the object to the lines of sight in order to better approximate the perspective projection, until it reaches convergence. It is a fast algorithm that does not require an initial pose estimate. Lu et al. (2000) presented another iterative method that formulates the PnP problem as the minimization of collinearity error in the object space, achieving high accuracy. The popular 'Iterative PnP' method that is available in the OpenCV libraries implements the Direct Linear Transform (DLT) algorithm to algebraically solve the system of equations determined by projective geometry. It then uses a Levenberg–Marquardt optimization to iteratively find the pose that minimizes the reprojection error. Another common approach consists in combining the Iterative PnP method with the Random Sample Consensus (RANSAC) algorithm to detect outliers in the set of point correspondences and estimate the pose with the best subset. The main disadvantage of this approach is the increase in computation time required by RANSAC. Lepetit et al. (2009) presented a non-iterative approach to solve the PnP problem, known as Efficient PnP (EPnP). They proposed a computationally efficient solution that can handle both planar and non-planar configurations by expressing the 3D points as a weighted sum of four virtual control points, and they achieved little loss of accuracy with respect to iterative approaches while being much faster.

Most of gaze estimation systems require performing HPE based on 2D facial point tracking, estimating the incremental head pose that accounts for the variation in the image observation. One of the main problems of tracking-based approaches is the accumulated error that may appear as the appearance model updates, also known as *drifting*. Many papers have addressed this issue with different success. Xiao et al. (2002) used templates of the head image and the corresponding head pose that were dynamically updated along the tracking, which was performed using iteratively re-weighted least squares (IRLS). When a new image was found to be in a pose close to that in one of the templates, the image was re-registered to the template to minimize error accumulation. Particle filters based on the appearance of the head have also been largely used in the literature to improve the tracking by maximizing the posterior probability using a motion history model. Once the head dynamics have previously been modeled, virtual views of the head can be rendered and compared with the current observation in order to update the weights of the particle filter (Choi and Kim, 2008; Tu et al., 2006). Kalman filters have also been introduced in tracking-based approaches, trying to use information from predicted head poses in order to improve the tracking itself (Yu and Gang, 2011). Wang et al. (2012) developed a method based on keypoint matching, learning keypoint descriptors invariant to different viewpoints, nonrigid deformations and illumination changes. Color information was used in order to eliminate keypoints lying outside the face, and optical flow applied in order to remove motion jitter. Tran et al. (2013) trained their method using synthetic data generated using a parametric 3D model and manually annotated facial landmarks, applying SIFT descriptors to learn an appearance model. Tracking was then performed and the head pose estimated using posterior probability, updating the tracking model through SVM. Recently, Jeni et al. (2017) developed a 3D cascade regression approach that fits a dense 3D shape for each frame in real time with good results in many different imaging scenarios, and Diaz Barros et al. (2018) proposed combining the accuracy in HPE given by the detection of facial landmarks with the ability to handle extreme poses given by the detection of salient features in the head region.

Most of these solutions require complex training, or are specifically designed for the tracking of the head, making their adaptation to other application fields difficult. This article aims to contribute to the development of robust and accurate HPE methods based on 2D tracking of the face, enhancing performance of both 2D point tracking and 3D pose estimation and addressing especially the problem of drifting of points that lose track. With the aim of achieving a wider impact and going one step further, the problem is addressed using a global approach: all the methods proposed are generalizable to the tracking of any object for which an approximate 3D model is available. The range of application of the techniques presented in this paper thus extends to generic object tracking and pose estimation.

Section 2 describes the proposed contributions in detail. It starts with the baseline 2D-feature-based 3D pose estimation system based on POSIT. Then, the novel weighted POSIT (wPOSIT) algorithm is proposed, which aims to enhance 3D pose estimation by applying weights to the 2D–3D point correspondences based on the tracking accuracy. Moreover, a method for obtaining a tracking accuracy index (TAI) is described based on two invariant shape metrics that we introduce, which allows us to calculate weights for wPOSIT. The method is naturally extended for outlier detection and correction using combined information from 2D and 3D. The adaptations of all these generic-object methods to the problem of HPE are specified in detail. Section 3 presents 2D tracking and 3D HPE results for two different head pose databases, focusing on the improvement given by the techniques proposed and comparing our HPE accuracy results with 29 other state-of-the-art methods. Section 4 presents the discussion of the results, and Section 5 closes the article with the concluding remarks and future lines of work.

2. Methods

The process of performing HPE in a video sequence using a 2D-tracking-based method can be divided in three different steps according to the approach proposed in this article: (1) 2D facial feature point detection and tracking; (2) 3D pose estimation; and (3) 2D tracking and 3D pose estimation enhancement through wPOSIT, outlier detection and outlier correction. This section is organized according to those steps. As it has already been stated, an important added value of this work is that the proposed contributions are valid for 2D tracking and 3D pose estimation of any object with very simple adaptations. Each of the following subsections makes a short reference to this point.

2.1. 2D feature detection and tracking

A set of 12 characteristic facial landmarks has been chosen for the 2D face tracking, as shown in Fig. 1. 2D feature tracking algorithms are prone to drifting errors, and they usually perform best when the selected points are the most characteristic ones, which typically means choosing well-defined corners. Adding more points to this 12-point model would imply including points that, instead of being corners, are located on contours defined by the mouth, nose, eyes or eyebrows. For tracking algorithms, these points usually perform worse because the point may drift along the contour during tracking, due to the appearance similarity of image patches along it.

The points are automatically detected in the initial frame of a sequence using Active Shape Models (ASM) (Cootes et al., 1995). Specifically, an ASM implementation based on the work by Cerrolaza

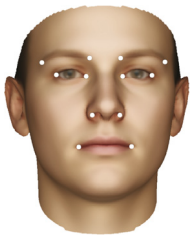

Fig. 1. Illustration of the 12 facial features tracked.

et al. (2012) has been used. This study proposes to work with multiple resolutions, looking for the learned shape gradually from a coarser resolution to the finest possible one. Once the 12 features are detected in the initial frame, they are tracked along the complete sequence using a pyramidal implementation of the popular Lucas–Kanade (LK) optical flow algorithm (Lucas and Kanade, 1981). The pyramidal implementation deals better with large displacements and the accuracy-robustness tradeoff that comes with the choice of the search window.

The main reason for the choice of LK is that this 2D point tracking method is universal for any kind of 3D object without the need of any training, whereas other popular methods such as IntraFace (IF) (Xiong and De Torre, 2013), the already mentioned ASM, or Active Appearance Models (AAM) (Cootes et al., 2001) require previous training. In the case of IF, the public version has only been trained for faces and the code is not available to train the method differently. In the case of ASM and AAM, performing an exhaustive training that includes views of the object in any pose is a time-consuming task, often tedious because of the difficulty to find labeled datasets, and there is little guarantee of achieving a good performance.

Training ASM for the initial feature detection though, as in our method, is an easier task, since it only needs frontal views of the object, in this case the face. We have carried out this training using the publicly available 2D-labeled dataset created by Ariz et al. (2016). However, LK may also be combined with methods for the automatic detection of strong corners in an initial image patch, such as the Shi–Tomasi corner detector (Shi and Tomasi, 1994), which makes its implementation for any kind of object tracking very simple.

2.2. Baseline method for 3D pose estimation

The head pose is estimated using POSIT algorithm (DeMenthon and Davis, 1995), which is a widely used generic object 3D pose estimation method based on projection geometry. It has been considered fit for the purpose of this work: it is fast, there are public implementations in C++ or Matlab among others, and it estimates the pose of any object from 2D images without the need of an initial estimate. The main disadvantage of POSIT against some other state of the art PnP methods is that it may fail in planar configurations. However, we showed in previous work (Ariz et al., 2016) that it achieves a high accuracy in the typical scenarios of our target applications (e.g. gaze estimation). If we have ideal 2D–3D correspondences, the intrinsic HPE error of POSIT is around 0.03°, which demonstrates that the non-coplanarity requirement is met by the facial landmark configuration with respect to the camera in these scenarios. Nevertheless, we include other state of the art PnP methods in the HPE performance comparison in the results section to validate further the correct choice of POSIT. We will thus call the *baseline method* to applying POSIT to a set of 2D–3D correspondent points that describe the shape of the face, in order to obtain its 3D pose in a sequence of images.

A 3D head model is necessary in order to match the 2D points given by the tracker, which can be achieved by different Structure from Motion (SfM) methods. The Basel Face Model (BFM) created by Paysan et al. (2009) allows us to generate different 3D head models on a PCA basis. The mean shape of the PCA has been used to initialize a Sparse Bundle Adjustment (SBA) algorithm (Triggs et al., 2000), which fits a sparse 3D model by using different views in 2D and minimizing the reprojection error between the observed and predicted image points, applying a Levenberg–Marquardt non-linear least squares algorithm. In order to ensure a coherent head model, this 3D fitting has then been refined by finding the closest PCA observation to it in the BFM.

Both POSIT and SBA are methods that can be applied to any 3D object. The following subsections describe in detail the techniques proposed in order to enhance both the 2D tracking and 3D pose estimation.

2.3. Weighted POSIT

Weighted POSIT (wPOSIT) is a novel variant of POSIT algorithm, in which a certain weight is applied to each 2D–3D point correspondence. The manner in which these weights are applied consists in repeating each point as many times as its weight indicates. That is, the list of 2D–3D point correspondences passed to the POSIT algorithm contains repeated points according to the weights assigned. Therefore, the weights are integer numbers and the maximum weight is a parameter of the algorithm that, in turn, determines the number of weight levels available.

The optimal way to determine the weight of a point in a certain frame would be to calculate a tracking accuracy index (TAI), which should indicate the accuracy with which the point has been located in that frame; the more accurately the point has been tracked, the higher the weight that should be applied, and vice versa. Obtaining a TAI for each point in every frame is a challenging task that is addressed in the next section.

The feasibility of wPOSIT was assessed in advance in a HPE simulation environment that was built using the simulator tool and the UPNA Synthetic Head Pose Database presented by Larumbe et al. (2017). Noisy instances of the synthetic database were created introducing Gaussian 2D tracking error in the simulations, thus generating databases with increasing average 2D error ranging from 0 to 8 pixels for a 1280×720 resolution. This range covers the typical error for most standard tracking methods, such as LK, IF, ASM or AAM. Using the ideal 3D correspondences of the noisy instances of 2D facial points, POSIT and wPOSIT were applied and compared in terms of HPE error. Ideal weights could be applied to wPOSIT, since the 2D error of each point in each image frame was known and the weight was determined inversely proportional to this error. It was observed that, on the one hand, the number of weight levels had little influence in the result and 50 is a right value for this parameter and, on the other hand, wPOSIT gives an approximately stable 10% accuracy improvement with respect to POSIT for any tracking noise level (except for perfect tracking, were both algorithms perform equally as expected).

2.4. Tracking Accuracy Index (TAI)

This section presents the method that we have developed to obtain a TAI, an index that describes how well a point is being tracked in each frame of a sequence. It is based on the use of two invariant shape metrics that we propose, and their associated tolerance models. In order to guarantee the feasibility of the method, techniques that ensure metric independency against the object's specific shape and pose are also proposed. The method is described for any generic object, and specific adaptations to HPE are described when necessary.

2.4.1. Invariant shape metrics

The proposed method is based on using interlandmark relationships to analyze the 2D geometrical configuration of the tracked points for each videoframe. This idea was introduced by Lekadir et al. (2007), where the ratio of interlandmark distances in triplets of points was defined as an invariant shape metric. We propose two complementary shape metrics (r and s) to characterize the geometrical configuration of

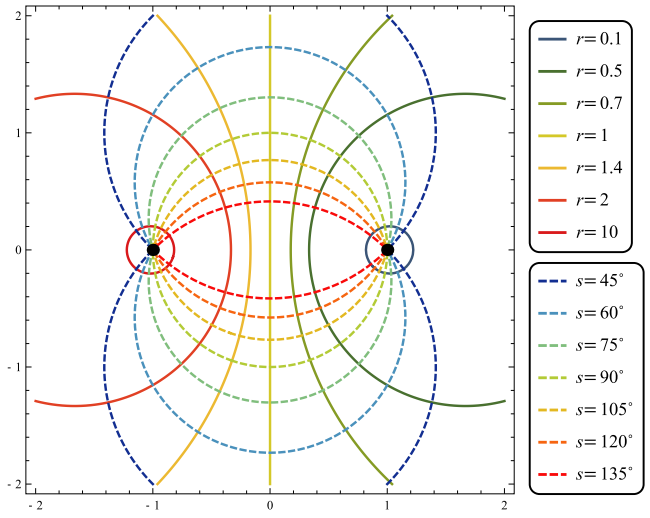

Fig. 2. Complementarity of the shape metrics r and s. Two fixed points, $(-1, 0)$ and $(1,0)$ are represented, and around them a set of contours where the third point of the triplet may lie for each of the metrics to keep a constant value.

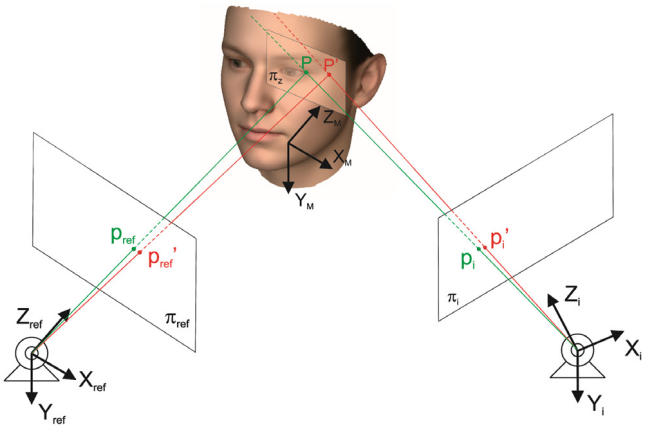

Fig. 3. Illustration of the pose normalization process. Original camera (X_i, Y_i, Z_i) and pose-normalized camera $(X_{ref}, Y_{ref}, Z_{ref})$ are represented, together with the image projections of a correctly-tracked point P and a wrongly-tracked point P′ from the 3D head model coordinate system (X_M, Y_M, Z_M).

the 2D features tracked in the object. These metrics are defined for each triplet of 2D points (p^j, p^k, p^l) as

$$r^{jkl} = \frac{d^{jk}}{d^{kl}}, \qquad s^{jkl} = \theta(\vec{jk}, \vec{kl}) \tag{1}$$

where d^{jk} and d^{kl} represent the Euclidean distance from p^j to p^k and from p^k to p^l respectively, and $\theta(\vec{jk}, \vec{kl})$ is the angle formed by two vectors, one going from p^j to p^k and the other one from p^k to p^l.

The advantage of using these two shape metrics is that they are complementary, in the sense that any 2D point that drifts from its original position in the image produces a change in at least one of the two metrics. Fig. 2 shows this complementarity: fixing two points of a certain triplet, $(-1, 0)$ and (1, 0) in the figure, for each value of r there is a contour along which the third point of the triplet may drift while keeping r constant. This means that there is a possibility of a point drifting along a specific contour in the object during the tracking so that this drifting would go undetected for the first shape metric. An analogous behavior is observed for the metric s. However, the contours that keep each metric constant are different and intersect only in two specific points in the image. Therefore, there does not exist any contour along which a point may drift and go undetected for both shape metrics.

These metrics show the advantage of being invariant to 2D scaling, rotation or translation, but they are not invariant to 3D pose changes in the object and neither to differences in appearance of objects of the same kind. The following sections describe the solutions proposed to overcome these issues and achieve a full invariability of the shape metrics, which is one of the main challenges of the proposed method.

2.4.2. Pose normalization

In order to make the shape metrics invariant with respect to the 3D pose of the object, the tracked 2D points are pose-normalized to a reference pose, e.g. a frontal view of the object. The process of pose normalization can be observed in Fig. 3, illustrated for the specific example of face tracking and head pose normalization. It consists in simulating a camera rotation and translation in order to get a frontal view of the face. First, the 3D pose of the object is estimated using POSIT and all the tracked points that define the object shape. In the example in the figure, the estimated pose consists in the transformation between the head (X_M, Y_M, Z_M) and the camera (X_i, Y_i, Z_i) coordinate systems, being the camera in its actual position (before the pose-normalization). Using that information, each of the 2D image points is back-projected to the 3D model in the calculated pose, and the intersection between the back-projection line and the Z-plane corresponding to the depth of the original model point (π_Z plane in Fig. 3) is obtained as the current 3D object point being tracked in the image. Finally, this new 3D point is reprojected to a virtual camera located just in front of the object, according to the calculated pose (image plane π_{ref} in Fig. 3). In the figure, an inaccurately tracked (p_i') and ideally tracked (p_i) corner of the eye is shown, thus two image representations (one correct and one incorrect) of the same anatomical point. Both points give place to two different pose-normalized reprojections that keep being representative of the tracking error committed, since the difference between both points will only be due to the 2D tracking error and not to the instantaneous 3D pose of the head, if we assume ideal pose estimation. In reality, this is not possible and the error in the 3D pose estimation will inevitably affect to some length the pose normalization process and the resulting 2D points, making impossible to achieve a full pose independency. Nonetheless, it will be later shown in the results section that this error is acceptable and overall the method performs well, overcoming the 3D pose dependency problem.

2.4.3. Object independency

In order to achieve object independency of the metrics, the initial frame of the sequence to track is chosen as a reference template. The assumption made here is that our initial 2D feature point detection is correct and can therefore be used as a reference for comparison. This applies to the points given by ASM in our HPE system. The initial configuration of the points is then taken as 'ideal' for that object, and r and s metric values in every frame can be compared with that reference template, using the following comparison metrics:

$$cr_i = \begin{cases} \dfrac{r_i}{r_{ref}} & if\ r_i \geq r_{ref} \\ -\dfrac{r_{ref}}{r_i} & if\ r_i < r_{ref} \end{cases}, \qquad cs_i = s_i - s_{ref} \tag{2}$$

being r_i and s_i the shape metrics calculated for the ith frame as defined in Eq. (1), and r_{ref} and s_{ref} the same metrics calculated for the reference frame (1st frame). The superscript that refers to the point number shown in Eq. (1) is not shown in Eq. (2) for simplicity in notation. The comparison metric cr_i is defined as a ratio between r shape metrics, since these in turn are defined as distance ratios between landmark triplets. It is defined in two sections in order to get two symmetric ranges for the values of the metric, i.e. $(-\infty, -1)$ and $[1, \infty)$. The comparison metric cs_i, on the other hand, is defined as a difference between s shape metrics, since these in turn are defined as angles between interlandmark vectors and thus cs_i measures the differential angle.

The shape metrics for every frame in the sequence, r_i and s_i, are obtained using the previously pose-normalized images. This is also done

with the metrics for the first frame, r_{ref} and s_{ref}. Working with the comparison metrics defined in Eq. (2) instead of the absolute metrics defined in Eq. (1) allows us to compare the interlandmark configuration of the current frame with that of the initial frame that is taken as reference. This comparison can be carried out whatever the 3D pose difference between both frames is, since the 2D shape points have previously been pose-normalized. And measuring the relative differences between both interlandmark configurations assures that appearance differences between different objects of the same kind are not affecting the measure. Therefore, we can claim that, through the described process, we obtain two invariant shape metrics (i.e. cr_i and cs_i) that only depend on the accuracy with which the feature points in the object are being tracked. Note that, in perfect tracking conditions, $cr_i = 1$ and $cs_i = 0$ for any triplet of points.

2.4.4. Tolerance model

Each landmark is now associated with two sets of invariant shape metrics for each frame. The main idea behind the calculation of a TAI is that an inaccurately tracked point will cause metrics calculated from triplets that include that point to be invalid. In order to tell whether a certain metric value is accepted or not, tolerance intervals (T) that characterize a valid model have to be calculated in what is usually called tolerance analysis (Guttman, 1970). Extreme values of the metrics can then be detected if they fall outside the statistically determined tolerance intervals, usually calculated from training samples. When a Gaussian distribution of the samples cannot be assumed, as in our LK tracker, non-parametric tolerance intervals should be used, calculated from the smallest and largest observation in the training set. Note that each triplet of points in the shape (j,k,l) gives room to two different shape metrics (cr^{jkl} and cs^{jkl}), and these will in turn have their own tolerance intervals associated (T_{cr}^{jkl} and T_{cs}^{jkl}).

The tolerance model for HPE has been built with the aid of the UPNA Head Pose Database that contains a large set of automatically annotated faces (Ariz et al., 2016). Since the labeled faces represent the ideal tracking, we can obtain the training samples by following the full procedure of pose normalization and comparison-shape-metrics' calculation using the automatically annotated 2D points in the database. Being the tracking ideal, the 2D error in the pose-normalized points will just be due to inaccuracies in the 3D model fitting, which in turn leads to inaccuracies in the pose normalization and in the resulting 2D observations that correspond to the frontal view of the face. Therefore, we are able to build the tolerance model in perfect tracking conditions and we can assume that any sample that lies outside these intervals corresponds to inaccurate 2D tracking.

2.4.5. TAI calculation

Once the tolerance model has been built, two steps are followed in order to measure the dissimilarity of a point with the model. First, a likelihood measure f_c is obtained for each metric. According to how the comparison metrics have been defined in Eq. (2), we build the likelihood measure for each metric as a linear function with values going from 1 to 0 inside the tolerance interval, being 1 when the metric corresponds to the perfect tracking, and 0 outside the interval:

$$f_{cr}(cr) = \begin{cases} \dfrac{cr_H - cr}{cr_H - 1}, & if\ 1 \le cr \le cr_H \\ \dfrac{cr_L - cr}{cr_L + 1}, & if\ -1 > cr \ge cr_L \\ 0, & elsewhere \end{cases} \tag{3}$$

$$f_{cs}(cs) = \begin{cases} 1 - \dfrac{cs}{cs_H}, & if\ 0 \le cs \le cs_H \\ 1 - \dfrac{cs}{cs_L}, & if\ 0 > cs \ge cs_L \\ 0, & elsewhere \end{cases} \tag{4}$$

where tolerance intervals corresponding to both metrics are defined for each triplet as $T_{cr} = [cr_L, cr_H]$ and $T_{cs} = [cs_L, cs_H]$. Note that the superscript 'jkl' corresponding to each triplet is not shown in the equations for clarity of notation.

When a metric falls outside of the tolerance, it is not straightforward to identify which point in the triplet is the cause, but we can calculate the likelihood f^p of each point to be an inlier by adding up the contributions of all the metrics associated to the point:

$$f^p(p^j) = \frac{1}{N^j} \sum_{k,l} f_c(c^{jkl}) f_c(c^{klj}) f_c(c^{ljk}) \tag{5}$$

where N^j is the number of triplets in which the point p^j is present. Here again we use a general notation for both metrics, but we will actually obtain two likelihood measures (fr^p and fs^p) applying equation (5) to the two metrics. The two likelihood measures are averaged to get the final f^p for every landmark. This process is carried out for each frame in the video sequence, and hence the likelihood measure can be written as f_i^p, where $i = 1 \dots N$ frames and $p = 1 \dots P$ points defining the shape of the object. As the reader may already have noticed, f_i^p is the TAI we were looking for from the beginning, ranging from 1, when the point is perfectly tracked, to 0, when all the metrics in which the point is involved fall outside the tolerance model.

2.4.6. Weight calculation

The calculated TAI can easily be transformed into a point weight for wPOSIT by applying weights proportional to the TAI value, knowing that the minimum weight (i.e. 1) must correspond to a TAI of 0, and the maximum weight w_{max} (i.e. the number of weight levels we want to use) to a TAI of 1. Weights are thus obtained from TAIs through the following equation:

$$w^p = round\left(1 + (w_{max} - 1) \cdot f^p\right) \tag{6}$$

2.5. Outlier detection

Using the calculated TAI, outlier detection (OD) can be carried out in order to eliminate outliers in the 2D shape tracked in each frame. We need to set a threshold λ for the TAI below which a point will be considered a potential outlier. This condition would then be defined by the following equation:

$$f^p(p^j) < \lambda \tag{7}$$

If Eq. (7) is satisfied, the point is considered a potential outlier and undergoes the final checking process that will confirm or discard it as an actual outlier. The pose is re-estimated without this landmark and the new pose-normalized face is obtained, where new metrics are calculated for the remaining points, leading to new f_p's and new weights. These new TAIs are compared with the original ones (excluding the outlier) and, if the average value of the new TAIs exceeds the average value of the old ones, the outlier is confirmed. If this final condition is met, it means that the metrics in which the outlier was involved were contributing negatively to the TAIs of the other landmarks, which is a very reasonable indicative of a point being an outlier. This OD loop is iteratively repeated until no more outliers are detected.

2.6. Outlier correction

As a final contribution, we present a method for outlier correction (OC). Once an outlier has been detected in a certain frame, an alternative to just removing it is to try to readjust it to its correct image position. OC is laid out as an iterative process of 2D position correction, in which the estimated pose, initially obtained by wPOSIT without the outlier (OD), and calculated using the corrected point in subsequent iterations (OC), is used to calculate the ideal 2D image position of the detected outlier through a pose denormalization process. Let us explain this in detail: we have initially obtained a reference template of the 2D shape in a frontal view by pose-normalizing the 2D points detected in the initial frame. This reference is supposed to be the ideal tracking, and thus the calculated TAIs in a certain frame measure the 2D tracking

Algorithm 1 wPOSIT+OD+OC for generic 3D object pose estimation

Input: $\boldsymbol{X}^p: (X^p, Y^p, Z^p)$, $\boldsymbol{x}_i^p: \left(x_i^p, y_i^p\right)$; $p = 1 \dots P$ points, $i = 1 \dots N$ frames. We denote them as $\boldsymbol{X}$ ($P \times 3$ matrix) and $\boldsymbol{x}_i$ ($P \times 2$ matrix) for simplicity.
Process initial frame (we denote it as *ref* for 'reference'):
    Apply POSIT: $\left(\boldsymbol{R}_{ref}, \boldsymbol{T}_{ref}\right) = POSIT\left(\boldsymbol{X}, \boldsymbol{x}_{ref}\right)$
    Pose-normalization: $\tilde{\boldsymbol{x}}_{ref} = PoseNorm\left(\boldsymbol{R}_{ref}, \boldsymbol{T}_{ref}, \boldsymbol{x}_{ref}\right)$
    Calculate r_{ref} and s_{ref} invariant shape metrics from $\tilde{\boldsymbol{x}}_{ref}$
for $i = 2 \rightarrow N$ **do** (process rest of frames)
    Apply POSIT: $(\boldsymbol{R}_i, \boldsymbol{T}_i) = POSIT(\boldsymbol{X}, \boldsymbol{x}_i)$
    Pose-normalization: $\tilde{\boldsymbol{x}}_i = PoseNorm(\boldsymbol{R}_i, \boldsymbol{T}_i, \boldsymbol{x}_i)$
    Calculate r_i and s_i invariant shape metrics from $\tilde{\boldsymbol{x}}_i$
    Comparison metrics: $cr_i = f\left(r_{ref}, r_i\right)$ and $cs_i = f\left(s_{ref}, s_i\right)$
    Tolerance analysis to obtain TAI: $f_i = TolAnalysis(cr_i, cs_i)$
    while $(min(f_i) < \lambda)$ **do**
        Repeat steps 9-11 without outlier and obtain f_i^{OD}
        Calculate weights without outlier: $w_i^{OD} = WeightCalc(f_i^{OD})$
        wPOSIT without outlier: $(\boldsymbol{R}_i^{OD}, \boldsymbol{T}_i^{OD}) = wPOSIT(\boldsymbol{X}^{OD}, \boldsymbol{x}_i^{OD}, w_i^{OD})$
        Initialize OC: $(\boldsymbol{R}_{iter}^{OC}, \boldsymbol{T}_{iter}^{OC}) = (\boldsymbol{R}_i^{OD}, \boldsymbol{T}_i^{OD})$, $d_{iter}^{OC} = \infty$
        while $(d_{iter}^{OC} < \varepsilon)$ **and** $(iter \leq M_{iter})$ **do**
            Outlier correction: $\boldsymbol{x}_{iter}^{OC} = PoseDenorm\left(\boldsymbol{R}_{iter}^{OC}, \boldsymbol{T}_{iter}^{OC}, \boldsymbol{x}_i^{OD}, \tilde{\boldsymbol{x}}_{ref}\right)$
            Pose-normalization: $\tilde{\boldsymbol{x}}_{iter}^{OC} = PoseNorm(\boldsymbol{R}_{iter}^{OC}, \boldsymbol{T}_{iter}^{OC}, \boldsymbol{x}_{iter}^{OC})$
            Repeat steps 9-11 with $\tilde{\boldsymbol{x}}_{iter}^{OC}$ and obtain f_{iter}^{OC}
            Calculate weights for OC: $w_{iter}^{OC} = WeightCalc(f_{iter}^{OC})$
            Update pose with OC: $(\boldsymbol{R}_{iter}^{OC}, \boldsymbol{T}_{iter}^{OC}) = wPOSIT(\boldsymbol{X}, \boldsymbol{x}_{iter}^{OC}, w_{iter}^{OC})$
            Update convergence metric: $d_{iter}^{OC} = EuclDist(\boldsymbol{x}_{iter}^{OC}, \boldsymbol{x}_{iter-1}^{OC})$
        end while
        if $\left(mean(f_i^{OC}) > mean(f_i)\right)$ **then**
            OC confirmed. Update $f_i = f_i^{OC}$, $\boldsymbol{x}_i = \boldsymbol{x}_i^{OC}$
        else
            Break **while loop**
        end if
    end while
    Calculate final weights: $w_i = WeightCalc(f_i)$
    Final pose: $(\boldsymbol{R}_i', \boldsymbol{T}_i') = wPOSIT(\boldsymbol{X}, \boldsymbol{x}_i, w_i)$
end for
Output: $(\boldsymbol{R}_i', \boldsymbol{T}_i')$; $i = 1 \dots N$ frames

performance with respect to that reference. Therefore, we can apply an inverse process of pose denormalization to that template and obtain the corresponding 2D projection of the reference points in the desired pose, i.e. the pose estimated for the current frame.

The projection obtained for the point detected as an outlier is further refined based on image appearance. A small window (i.e. 7×7) around the obtained projection is checked against the corresponding point 10 frames before, assuming a small image variation between frames, and assuming the loss of tracking is recent – the tracking was correct 10 frames before –.Patches of 21×21 pixels are compared by measuring the sum of absolute differences (SAD). The point in the 7×7 window that matches best the previous appearance is kept as the final correction.

This template-back-projection plus image-based-correction process is iteratively repeated until a convergence is reached. This convergence is assessed by measuring the 2D Euclidean distance between corrections in consecutive iterations, setting a minimum distance below which the algorithm stops.

The final image points are pose-normalized with the last estimated 3D pose, and the process to obtain the final TAIs is followed. Similarly to the OD method described in the previous section, if the new TAIs, calculated with the corrected outlier, exceed in average the old ones (i.e. without the correction), the correction is accepted, updating the tracker. This OC loop is iteratively repeated until no more outliers are detected for the current frame, checking the lowest TAI among the new ones against the threshold at each iteration.

Once the outlier-corrected 2D–3D correspondences have been determined and the final TAIs and weights have been calculated, wPOSIT is applied in order to obtain the definitive pose for the current frame. The complete pose estimation system, including wPOSIT and OD + OC, is described in detail in Algorithm 1.

3. Experimental results

The methods proposed in this work give room to four different HPE approaches when we incorporate just wPOSIT, wPOSIT plus the OD module, or wPOSIT with both OD and OC modules, to the baseline method. For the evaluation of the methods, two head pose video databases have been employed: the BU Headtracking Database (La Cascia et al., 2000) and the UPNA Head Pose Database (Ariz et al., 2016). The former contains 45 videos of 200 frames of 5 users performing free head movements, whereas the latter contains 120 videos of 300 frames of 10 users performing both guided and free movement sequences. HPE

Table 1
2D head tracking accuracy results for the UPNA database (1280 × 720 resolution). Errors for all frames (AF) and only for frames in which at least one outlier is detected (OF) are shown. The accuracy gain given by each method with respect to the baseline method is also shown.

Method	Tracking error (px)		Gain obtained (%)	
	AF	OF	AF	OF
Baseline method	4,03	6,73	–	–
wPOSIT	4,03	6,73	–	–
wPOSIT+OD	3,56	4,82	11,68	29,26
wPOSIT+OD+OC	3,59	5,20	10,75	23,78

results have been evaluated in both databases. Since the head pose ground truth of the BU database is notably noisy compared to the one of the more modern UPNA database, the former was smoothed for this evaluation by applying a moving average with a window of 5 frames. It was observed that this smoothing provided a consistent accuracy increase of ∼0.07° independently of the HPE method used.

In addition to that, the UPNA database also contains a 2D ground truth, which allows us to evaluate 2D tracking performance.

The 2D facial point tracking accuracy is measured as the average Euclidean distance between the tracked points and the 2D ground truth for a 1280 × 720 pixel resolution. This error is presented in Table 1, measured for all the frames (AF) in the database, on the one hand, and just for frames in which at least one outlier has been detected (OF), on the other hand. The proportion of OF depends on the parameters of the algorithm, especially the threshold for outlier detection, which has been set to 0.6 based on preliminary experimentation. We have found that 23.57% of the frames in the database give place to at least one outlier according to our 2D tracking system and OD module. The tracking accuracy gain given by each of the proposed enhancements with respect to the baseline method is also shown in the table. Note that the wPOSIT approach gives the same result as the baseline method in terms of point tracking, since it only affects the 3D pose estimation leaving the tracked 2D points unaltered.

Average HPE errors for the two databases are summarized in Table 2, again for both AF and OF. Following the classical approach, only rotation errors are shown. Table 3 shows an extensive comparison of our methods against other state-of-the-art methods on the two databases.

Visual examples of the effect of wPOSIT and OD and OC modules in 2D tracking and HPE accuracy are shown in Fig. 4, in which two example frames from the UPNA database are shown. wPOSIT achieves to compensate for tracking inaccuracies and improve HPE, whereas OD and OC are able to detect and correct outliers, eventually improving both 2D tracking and 3D HPE.

4. Discussion

Regarding the 2D tracking results, it is observed that OD and OC methods perform well and improve the baseline method considerably, achieving an average error of around 3.5 pixels for a 1280 × 720 resolution. The wPOSIT approach does not give any tracking improvement since it only affects the estimation of the 3D pose without altering the 2D points. The fact that the OD and OD + OC approaches achieve similar tracking results means that the 2D error of the corrected points is in the same range as the inliers' error, and thus it is performing successfully; OD eliminates outliers and they are not included in the 2D error calculation, whereas OC includes those points by calculating their corrected position. The main advantage of OC against OD is that we do not lose any tracked points, which is a key point for HPE accuracy and, besides, allows us to track much longer sequences.

In terms of the accuracy gain given by the enhanced approaches, it is observed that the OD and OD + OC methods achieve a 11.68% and 10.75% tracking accuracy improvement respectively if the 36.000 frames of the database are taken into account, and a 29.26% and 23.78% improvement if we compare the tracking only for frames where at least one outlier has been detected. All in all, we can expect a tracking accuracy improvement of over 10% using the OD module in normal 2D tracking conditions, and an accuracy improvement of up to 30% in noisy tracking conditions.

These results show that we have consistently addressed the problem of drifting of points that lose track, inherent to most point tracking systems. The capacity of our system to detect and correct drifting points makes it robust, and thus suitable to track long sequences. The tracking accuracy improvement achieved has a direct effect over the subsequent HPE, as stated in the lines below.

Regarding HPE results presented in Table 2, we obtain a top accuracy of 1.09° for the UPNA database, and 2.58° for the BU database. The gain given by the enhancements proposed is overall in the range of the one obtained for the 2D tracking, which shows that the tracking accuracy is directly related with the HPE accuracy, and therefore the enhancements proposed are being effective on both. In this case, we can observe that wPOSIT has a beneficial effect over HPE, reducing considerably the error with respect to the baseline method (1.21° and 2.71° vs. 1.30° and 2.78° for the UPNA and BU database respectively). When wPOSIT is combined with the OD module, this error is further reduced (1.18° and 2.62°) and, when the OC module is also implemented, the highest accuracy is achieved for both databases (1.09° and 2.58°). In frames containing at least one outlier this effect is highly magnified, in accordance with what we have observed for the 2D tracking accuracy.

For the UPNA database, which shows the expected performance of the system in current technological conditions, the HPE gain given by the final approach (wPOSIT + OD + OC) with respect to the baseline method is over 16% in normal tracking conditions and over 30% in noisy

Table 2
Mean absolute HPE errors for UPNA (Ariz et al., 2016) and BU (La Cascia et al., 2000) databases. Errors of the proposed approaches for all the frames in the database and only for frames in which at least one outlier is detected are shown.

HPE error – UPNA database								
Method	All frames				Frames with outliers			
	Roll (°)	Yaw (°)	Pitch (°)	Avg (°)	Roll (°)	Yaw (°)	Pitch (°)	Avg (°)
Baseline method	0,70	1,57	1,64	1,30	1,53	3,69	2,78	2,67
wPOSIT	0,67	1,42	1,53	1,21	1,37	3,08	2,45	2,30
wPOSIT+OD	0,63	1,33	1,58	1,18	1,20	2,71	2,62	2,18
wPOSIT+OD+OC	0,58	1,16	1,51	1,09	1,02	2,15	2,36	1,84
HPE error – BU database								
Method	All frames				Frames with outliers			
	Roll (°)	Yaw (°)	Pitch (°)	Avg (°)	Roll (°)	Yaw (°)	Pitch (°)	Avg (°)
Baseline method	1,90	3,54	2,90	2,78	3,01	10,34	5,18	6,18
wPOSIT	1,87	3,42	2,85	2,71	2,81	9,39	4,72	5,64
wPOSIT+OD	1,85	3,23	2,80	2,62	2,56	7,35	4,05	4,65
wPOSIT+OD+OC	1,86	3,09	2,78	2,58	2,71	5,94	4,00	4,22

Table 3
HPE accuracy comparison for the UPNA and BU databases between our methods and state-of-the-art algorithms.

UPNA database

Method	HPE error (°)			
	Roll	Yaw	Pitch	Avg
wPOSIT+OD+OC	0,58	1,16	1,51	1,09
wPOSIT+OD	0,63	1,33	1,58	1,18
wPOSIT	0,67	1,42	1,53	1,21
PnP tplus RANSAC	0,62	1,57	1,56	1,25
Baseline method	0,70	1,57	1,64	1,30
Iterative PnP	0,65	1,68	1,73	1,36
AAM+POSIT (Ariz et al., 2016)	1,04	1,63	2,19	1,62
Jeni 2017 (Jeni et al., 2017)	0,98	2,71	1,53	1,74
ASM+POSIT (Ariz et al., 2016)	0,88	2,50	2,77	2,05
Efficient PnP (Lepetit et al., 2009)	0,75	2,66	3,02	2,14
Xiong 2013 (Xiong and De Torre, 2013)	0,92	3,12	2,98	2,34
Ondras 2017 (Ondras et al., 2017)	–	4,10	2,50	3,30

BU database

Method	HPE error (°)			
	Roll	Yaw	Pitch	Avg
wPOSIT+OD+OC	1,86	3,09	2,78	2,58
wPOSIT+OD	1,85	3,23	2,80	2,62
wPOSIT	1,87	3,42	2,85	2,71
Wang 2012 (Wang et al., 2012)	1,86	3,75	2,69	2,77
Baseline Method	1,90	3,54	2,90	2,78
Xiao 2002 (Xiao et al., 2002)	1,40	3,80	3,20	2,80
Baltrusaitis 2012 (Baltrusaitis et al., 2012)	2,08	3,00	3,81	2,96
Jeni 2017 (Jeni et al., 2017)	2,41	3,93	2,66	3,00
Lefèvre 2009 (Lefèvre and Odobez, 2009)	2,00	4,40	3,30	3,23
Jang 2010 (Jang and Kanade, 2010)	2,07	4,22	3,44	3,24
Diaz Barros 2018 (Diaz Barros et al., 2018)	2,54	4,07	3,27	3,29
Prasad 2010 (Prasad and Aravind, 2010)	3,60	3,80	2,50	3,30
Xiong 2013 (Xiong and De Torre, 2013)	2,02	3,85	4,06	3,31
Diaz Barros 2017 (Diaz Barros et al., 2017)	2,56	3,99	3,39	3,31
PnP +RANSAC	2,06	4,61	3,47	3,38
Iterative PnP	2,57	3,64	4,03	3,42
Jang 2008 (Jang and Kanade, 2008)	2,10	4,60	3,70	3,47
Asteriadis 2014 (Asteriadis et al., 2014)	2,61	4,29	3,74	3,55
An 2008 (An and Chung, 2008)	2,83	3,95	3,96	3,58
Choi 2008 (Choi and Kim, 2008)	2,82	4,04	3,92	3,59
Morency 2008 (Morency et al.)	2,91	4,97	3,67	3,85
Saragih 2011 (Saragih et al., 2011)	2,60	4,30	4,80	3,90
Tran 2013 (Tran et al., 2013)	2,40	5,40	3,90	3,90
Tran 2015 (Tran et al., 2015)	2,20	5,00	4,50	3,90
Efficient PnP (Lepetit et al., 2009)	2,50	5,07	4,81	4,13
Mbouna 2013 (Mbouna et al., 2013)	3,78	3,94	4,83	4,18
Cheung 2015 (Cheung et al., 2015)	2,69	4,53	5,48	4,23
Vicente 2015 (Vicente et al., 2015)	3,20	4,30	6,20	4,57
Sung 2008 (Kanade et al., 2008)	3,10	5,40	5,60	4,70
Kumano 2009 (Kumano et al., 2009)	2,90	7,10	4,20	4,73
Guo 2012 (Guo et al., 2012)	5,30	4,90	4,80	5,00
Valenti 2009 (Valenti et al., 2009)	4,20	6,60	6,40	5,73
La Cascia 2000 (La Cascia et al., 2000)	9,80	3,30	6,10	6,40

tracking conditions. The accuracy of the 3D head tracker is of 1.09°, and it is incremented to only 1.84° when the tracking is very noisy, in those 23% of the frames that contain at least one outlier.

In order to get a better idea of the accuracy of the proposed approaches, most important state of the art HPE methods have been included in the HPE comparison in Table 3. The BU database is a benchmark of reference in the literature and has allowed us to compare our methods with 29 other state of the art algorithms, many of them recently published. The UPNA database is quite recent and not yet widely referenced. Ariz et al. (2016) and Ondras et al. (2017) published their results on it and we have included them in the table. Besides, we have tested three state of the art PnP approaches (i.e. Iterative PnP, EPnP (Lepetit et al., 2009) and Iterative PnP followed by RANSAC) available as functions of the OpenCV libraries. We have tested them by passing them the 2D points given by our baseline tracker in each frame, and we have included their results on both databases. We have also had access to the code of two algorithms recently published in articles of great quality in our opinion, i.e. IntraFace by Xiong and De Torre (2013) and the work by Jeni et al. (2017). We have tested IntraFace on both databases, and the algorithm by Jeni et al. on the UPNA database, since they already reported their results on the BU database. The comparison shows that our method is the most accurate one among all. The wPOSIT + OD + OC implementation provides the best results in both databases (1.09° of average error in the UPNA database and 2.58° in the BU database), followed by the wPOSIT+OD implementation (1.18° and 2.62°) in second position, and the wPOSIT implementation (1.21° and 2.71°) in third position. In the UPNA database, the fourth position is for PnP+RANSAC (1.25°), followed by our baseline method (1.30°), Iterative PnP (1.36°), AAM+POSIT (Ariz et al., 2016) (1.62°) and Jeni et al. (2017) (1.74°). The rest of the methods show HPE errors over 2°. In the BU database, the fourth and sixth best results are obtained by Wang et al. (2012) (2.77°), and Xiao et al. (2002) (2.80°). Our baseline method is fifth in the ranking, with 2.78° in average. The rest of the methods from the literature obtain average errors above 3°. All these results show to a greater extent the validity and accuracy of the methods presented in this work.

The results given by the three alternative PnP approaches (i.e. Iterative PnP, EPnP (Lepetit et al., 2009) and Iterative PnP followed by RANSAC) confirm the correct choice of POSIT as our baseline method for HPE. Only PnP+RANSAC achieves a slightly better accuracy than POSIT alone on the UPNA database (1.25° vs. 1.30°), but our outlier detection and correction methods clearly outperform RANSAC when combined with POSIT (1.09° vs. 1.25°). Furthermore, on the noisier BU database RANSAC is not able to perform that well: PnP+RANSAC gives 3.38° whereas Iterative PnP alone gives 3.42°, and both are far from POSIT alone (2.78°) and from our full OC approach (2.58°).

We have also measured the computational cost introduced by the methods proposed, measured as the average time employed to process each frame of the UPNA database. The processing is performed using Matlab and the Computer Vision System Toolbox on an IntelCore i5 processor with 6GB of RAM. The baseline method works at 23.89FPS (frames per second), whereas the wPOSIT approach works at 22.50FPS, the wPOSIT+OD approach at 21.22FPS, and the wPOSIT + OD + OC at 20.47FPS. The computational cost of the enhancements proposed is thus very small, and we have observed that the 2D feature tracking Lucas–Kanade algorithm takes most of the processing time (41.60 ms per frame in average). The complete HPE system works almost real-time in Matlab, and thus it should be straightforward to obtain a C++ version of the algorithm for real-time video processing. Therefore, these results show the feasibility of the methods proposed in terms of computational cost.

5. Conclusion

We have presented different contributions for improving the accuracy of 3D positioning systems based on 2D point tracking, showing that our methods lead to an enhancement of both 2D tracking and 3D pose estimation. All the methods presented can be applied to the tracking and pose estimation of any kind of object for which an approximate 3D model is available, which helps us achieving a wider impact in terms of application and gives an important added value to this work.

These global methods have been evaluated for the specific task of HPE using two different head pose video databases; the recently published UPNA database that reflects the expected performance of the system in current technological conditions, and the older BU database that allows an extensive comparison with state-of-the-art HPE methods. Results show that the proposed enhancements improve the accuracy of both 2D facial point tracking and 3D HPE, with respect to the implemented baseline method, by over 15% in normal tracking conditions and over 30% in noisy tracking conditions, with a minimal additional computational cost. Moreover, the proposed final HPE system outperforms the state of the art on the two databases, which shows to a greater extent the value of the methods presented in this work.

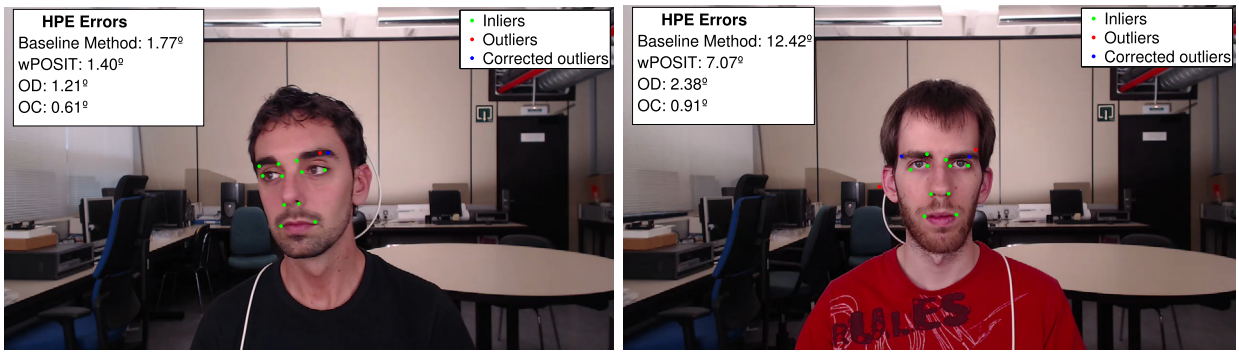

Fig. 4. Visual example of the effect of the OC module on two different frames from the UPNA database. Inliers are represented in green, outliers detected by the OD module in red, and corrected points estimated by the OC module in blue. Instantaneous HPE errors of the four approaches are shown on the top left side, showing the improvement achieved by each enhancement.. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Acknowledgments

We would like to acknowledge the Spanish Ministry of Economy, Industry and Competitiveness for their support under Grant "Formación de Profesorado Universitario AP2010-5191" and Contract "TIN2014-52897-R". We would like also to acknowledge the Spanish Ministry of Science, Innovation and Universities, contract TIN2017-84388-R.

References

An, K.H., Chung, M.J., 2008. 3D head tracking and pose-robust 2D texture map-based face recognition using a simple ellipsoid model. In: IEEE/RSJ Int. Conf. Intell. Robot. Syst. IROS, pp. 307–312.

Ariz, M., Bengoechea, J.J., Villanueva, A., Cabeza, R., 2016. A novel 2D/3D database with automatic face annotation for head tracking and pose estimation. Comput. Vis. Image Underst. (Special Issue on Assistive Computer Vision and Robotics).

Asteriadis, S., Karpouzis, K., Kollias, S., 2014. Visual focus of attention in non-calibrated environments using gaze estimation. Int. J. Comput. Vis. 107 (3), 293–316.

Ba, S.O., Odobez, J.-M., 2011. Multiperson visual focus of attention from head pose and meeting contextual cues. IEEE Trans. Pattern Anal. Mach. Intell. 33 (1), 101–116.

Baltrusaitis, T., Robinson, P., Morency, L.P., 2012. 3D constrained local model for rigid and non-rigid facial tracking. In: IEEE Conf. Comput. Vis. Pattern Recognit., pp. 2610–2617.

Belhumeur, P.N., Jacobs, D.W., Kriegman, D.J., Kumar, N., 2013. Localizing parts of faces using a consensus of exemplars. IEEE Trans. Pattern Anal. Mach. Intell. 35 (12), 2930–2940.

Cazzato, D., Leo, M., Distante, C., 2014. An investigation on the feasibility of uncalibrated and unconstrained gaze tracking for human assistive applications by using head pose estimation. Sensors 14 (5), 8363–8379.

Cerrolaza, J.J., Villanueva, A., Cabeza, R., 2012. Hierarchical statistical shape models of multiobject anatomical structures: Application to brain MRI. IEEE Trans. Med. Imaging 31 (3), 713–724.

Cheung, Y., Member, S., Peng, Q., 2015. Eye gaze tracking with a web camera in a desktop environment. IEEE Trans. Human-Machine Syst. 45 (4), 419–430.

Choi, S., Kim, D., 2008. Robust head tracking using 3D ellipsoidal head model in particle filter. Pattern Recognit. 41 (9), 2901–2915.

Cootes, T.F., Edwards, G.J., Taylor, C.J., 2001. Active appearance models. IEEE Trans. Pattern Anal. Mach. Intell. 23 (6), 681–685.

Cootes, T.F., Taylor, C.J., Cooper, D.H., Graham, J., 1995. Active shape models-their training and application. Comput. Vis. Image Underst. 61 (1), 38–59.

DeMenthon, D.F., Davis, L.S., 1995. Model-Based object pose in 25 lines of code. Int. J. Comput. Vis. 15 (1), 123–141.

Diaz Barros, J.M., Garcia, F., Mirbach, B., Stricker, D., 2017. Real-time monocular 6-dof head pose estimation from salient 2D points. In: IEEE Int. Conf. Image Process.

Diaz Barros, J.M., Garcia, F., Mirbach, B., Varanasi, K., Stricker, D., 2018. Combined framework for real-time head pose estimation using facial landmark detection and salient feature tracking. In: Proc. 13th Int. Jt. Conf. Comput. Vision, Imaging Comput. Graph. Theory Appl., VISIGRAPP 2018, vol. 5, VISAPP, pp. 123–133.

Guo, W., Kotsia, I., Patras, I., 2012. Tensor learning for regression. IEEE Trans. Image Process. 21 (2), 816–827.

Guttman, I., 1970. Statistical Tolerance Regions: Classical and Bayesian. Griffin, London, UK.

Jang, J.-S., Kanade, T., 2008. Robust 3D head tracking by online feature registration. In: IEEE Int. Conf. Autom. Face Gesture Recognit., pp. 1–6.

Jang, J.S., Kanade, T., 2010. Robust 3D Head Tracking by View-Based Feature Point Registration. Tech. report, People Image Anal. Consortium, Carnegie Mellon Univ.

Jeni, L.A., Cohn, J.F., Kanade, T., 2017. Dense 3D face alignment from 2D video for real-time use. Image Vis. Comput. 58, 13–24.

Kanade, T., Sung, J., Kim, D., 2008. Pose robust face tracking by combining active appearance models and cylinder head models. Int. J. Comput. Vis. 80 (2), 260–274.

Kumano, S., Otsuka, K., Yamato, J., Maeda, E., Sato, Y., 2009. Pose-invariant facial expression recognition using variable-intensity templates. Int. J. Comput. Vis. 83 (2), 178–194.

La Cascia, M., Sclaroff, S., Athitsos, V., 2000. Fast, reliable head tracking under varying illumination: An approach based on registration of texture-mapped 3D models. IEEE Trans. Pattern Anal. Mach. Intell. 22 (4), 322–336.

Langton, S.R.H., Honeyman, H., Tessler, E., 2004. The influence of head contour and nose angle on the perception of eye-gaze direction. Percept. Psychophys. 66 (5), 752–771.

Larumbe, A., Ariz, M., Bengoechea, J.J., Segura, R., Cabeza, R., Villanueva, A., 2017. Improved strategies for HPE employing learning-by-synthesis approaches. In: 2017 IEEE Int. Conf. Comput. Vis. Work., pp. 1545–1554.

Lefèvre, S., Odobez, J.M., 2009. Structure and appearance features for robust 3D facial actions tracking. In: Proc. IEEE Int. Conf. Multimed. Expo, ICME, pp. 298–301.

Lekadir, K., Merrifield, R., Yang, G., 2007. Outlier detection and handling for robust 3-D active shape models search. IEEE Trans. Med. Imaging 26 (2), 212–222.

Leo, M., Medioni, G., Trivedi, M., Kanade, T., Farinella, G.M., 2017. Computer vision for assistive technologies. Comput. Vis. Image Underst. 154, 1–15.

Lepetit, V., Moreno-Noguer, F., Fua, P., 2009. EPnP: An accurate O(n) solution to the PnP problem. Int. J. Comput. Vis. 81, 155–166.

Lu, C.P., Hager, G.D., Mjolsness, E., 2000. Fast and globally convergent pose estimation from video images. IEEE Trans. Pattern Anal. Mach. Intell. 22 (6), 610–622.

Lucas, B.D., Kanade, T., 1981. An iterative image registration technique with an application to stereo vision. In: Proc. 7th Int'l Jt. Conf. Artif. Intell, vol. 130, pp. 674–679.

Mbouna, R.O., Kong, S.G., Chun, M.G., 2013. Visual analysis of eye state and head pose for driver alertness monitoring. IEEE Trans. Intell. Transp. Syst. 14 (3), 1462–1469.

Morency, L.P., Whitehill, J., Movellan, J., 2008. Generalized adaptive view-based appearance model: Integrated framework for monocular head pose estimation. In: IEEE Int. Conf. Autom. Face Gesture Recognit. pp, 1–8.

Murphy-Chutorian, E., Member, S., Trivedi, M.M., 2009. Head pose estimation in computer vision. IEEE Trans. Pattern Anal. Mach. Intell. 31 (4), 607–626.

Navallas, J., Ariz, M., Villanueva, A., San Agustín, J., Cabeza, R., 2011. Optimizing interoperability between video-oculographic and electromyographic systems.. J. Rehabil. Res. Dev. 48 (3), 253–265.

Ondras, J., Celiktutan, O., Sariyanidi, E., Gunes, H., 2017. Automatic replication of teleoperator head movements and facial expressionas on a humanoid robot. In: 26th IEEE Int. Symp. Robot Hum. Interact. Commun.

Paysan, P., Knothe, R., Amberg, B., Romdhani, S., Vetter, T., 2009. A 3D face model for pose and illumination invariant face recognition. In: Sixth IEEE Int. Conf. Adv. Video Signal Based Surveill., pp. 296–301.

Prasad, B.H.P., Aravind, R., 2010. A robust head pose estimation system for uncalibrated monocular videos. In: Proc. 7th Indian Conf. Comput. Vision, Graph. Image Process., pp. 162–169.

Qiao, Y., Xie, X., Sun, T., Li, Y., 2008. A design of human-computer interaction based on head tracker. In: IEEE Pacific-Asia Work. Comput. Intell. Ind. Appl., pp. 718–721.

Saragih, J.M., Lucey, S., Cohn, J.F., 2011. Deformable model fitting by regularized landmark mean-shift. Int. J. Comput. Vis. 91 (2), 200–215.

Shi, J., Tomasi, C., 1994. Good features to track. In: IEEE Conf. Comput. Vis. Pattern Recognit., pp. 593–600.

Subramanian, R., Yan, Y., Staiano, J., Lanz, O., Sebe, N., 2013. On the relationship between head pose, social attention and personality prediction for unstructured and dynamic group interactions. In: Proc. 15th ACM Int. Conf. Multimodal Interact., pp. 3–10.

Tawari, A., Martin, S., Trivedi, M.M., 2014. Continuous head movement estimator for driver assistance: Issues, algorithms, and on-road evaluations. IEEE Trans. Intell. Transp. Syst. 15 (2), 818–830.

Tran, N.-T., Ababsa, F., Charbit, M., 2015. A robust framework for tracking simultaneously rigid and non-rigid face using synthesized data. Pattern Recognit. Lett. 65, 75–80.

Tran, N., Ababsa, F., Charbit, M., Petrovska-delacr, D., 2013. 3D face pose and animation tracking via eigen-decomposition based Bayesian approach. Adv. Vis. Comput. Lect. Notes Comput. Sci. 8033, 562–571.

Triggs, B., McLauchlan, P., Hartley, R., Fitzgibbon, A., 2000. Bundle adjustment - A modern synthesis. Vis. Algorithms Theory Pract. 1883, 298–372.

Tu, J., Huang, T., Tao, H., 2006. Accurate head pose tracking in low resolution video. In: Proc. IEEE Int'l Conf. Autom. Face Gesture Recognit., pp. 573–578.

Valenti, R., Sebe, N., Gevers, T., 2012. Combining head pose and eye location information for gaze estimation. IEEE Trans. Image Process. 21 (2), 802–815.

Valenti, R., Yucel, Z., Gevers, T., 2009. Robustifying eye center localization by head pose cues. In: IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. Work. CVPR Work. pp. 612–618.

Valstar, M.F., et al., 2017. FERA 2017 – Addressing head pose in the third facial expression recognition and analysis challenge. In: 12th IEEE Int. Conf. Autom. Face Gesture Recognit., FG 2017, Washington, DC, pp. 839–847.

Vicente, F., Huang, Z., Xiong, X., De Torre, F., Zhang, W., Levi, D., 2015. Driver gaze tracking and eyes off the road detection system. IEEE Trans. Intell. Transp. Syst. 16 (4), 2014–2027.

Wang, H., Davoine, F., Lepetit, V., Chaillou, C., Pan, C., 2012. 3-D head tracking via invariant keypoint learning. IEEE Trans. Circuits Syst. Video Technol. 22 (8), 1113–1126.

Wang, K., Xu, L., Fang, Y., Li, J., 2013. One-against-all frame differences based hand detection for human and mobile interaction. Neurocomputing 120, 185–191.

Xiao, J., Moriyama, T., Kanade, T., Cohn, J.F., 2002. Robust full-motion recovery of head by dynamic templates and re-registration techniques. In: Proc. Fifth IEEE Int. Conf. Autom. Face Gesture Recognit., pp. 593–600.

Xiong, X., De Torre, F., 2013. Supervised descent method and its applications to face alignment. In: IEEE Conf. Comput. Vis. Pattern Recognit., pp. 532–39.

Yu, W., Gang, L., 2011. Head pose estimation based on head tracking and the Kalman filter. In: Int'l Conf. Phys. Sci. Technol., vol. 22, pp. 420–427.

Yuan, X., Zhao, Q., Tu, D., Shao, H., 2013. A novel approach to estimate gaze direction in eye gaze HCI system. In: 5th Int. Conf. Intell. Human-Machine Syst. Cybern., pp. 588–591.