Fast Head Pose Estimation for Human-Computer Interaction

Mario García-Montero[1], Carolina Redondo-Cabrera[1], Roberto López-Sastre[1], and Tinne Tuytelaars[2]

[1] GRAM, University of Alcalá, Alcalá de Henares, Spain
[2] KU Leuven, ESAT - PSI, iMinds, Leuven, Belgium

Abstract. This paper describes a Hough Forest based approach for fast head pose estimation in RGB images. The system has been designed for Human-Computer Interaction (HCI), in a way that with just a simple web-cam, our solution is able to detect the head and simultaneously estimate its pose. We leverage the Hough Forest with Probabilistic Locally Enhanced Voting model, and integrate it into a system with a skin detection step and a tracking filter for the head orientation. Our implementation drastically speeds up the head pose estimations, improving their accuracy with respect to the original model. We present extensive experiments on a publicly available and challenging dataset, where our approach outperforms the state-of-the-art.

Keywords: Head pose estimation, Hough Forest, Tracking, Detection

1 Introduction

Human-Computer interaction (HCI) is gradually getting more and more attention. As people interact by means of many different channels, including body posture and head pose, an important step towards more natural interfaces is the visual analysis of the user movements by the machine. Besides the interpretation of full body movements, as done by systems like the Kinect for gaming, new interfaces would highly benefit from an automatic and fast estimation of the head pose, as the one presented in this work.

Recent state-of-the-art methods based on the principle of Hough Forests (HFs) [14] have proved to be very successful detecting and estimating the pose of the head (*e.g.* [11,12,18,19]). All these methods train a Random Forest (RF) [13] for a joint classification and regression. For inference, patches are densely sampled from the image and the model determines for each patch whether it belongs to the face or the background. Additionally, for head patches, the model performs a regression to localize the center of the head and to estimate its pose. However, while the methods presented in [11,18,19] require depth images, the HF with Probabilistic Locally Enhanced Voting (PLEV) described in [12] has achieved state-of-the-art results using only 2D imagery.

Here we leverage the HF with PLEV model [12], and design an approach to get a fast head pose estimation (see Figure 1). Our solution is able to perform

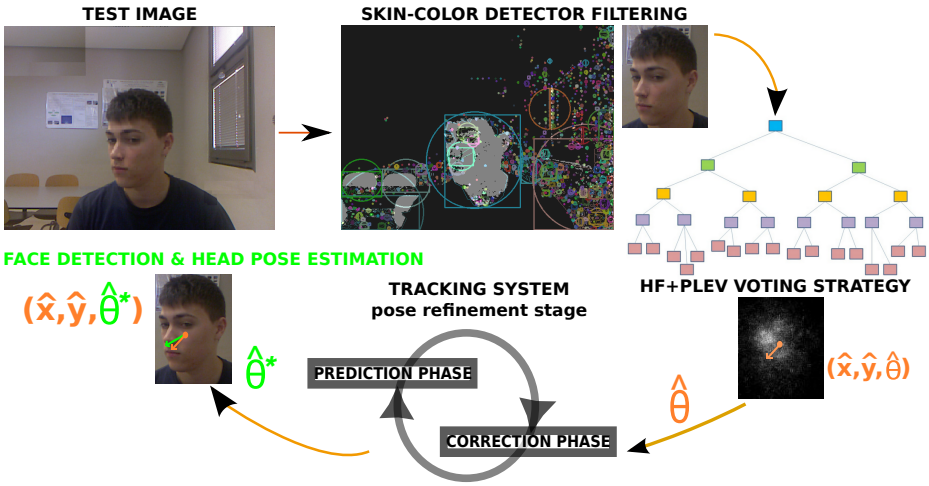

Fig. 1: Our approach is able to jointly estimate the localization and the continuous pose of the head. We start performing a skin-color filtering, and then we follow a HF + PLEV regression voting [12], in conjunction with a tracking step to consolidate the final head orientation estimations.

the detection and estimation using a simple web-cam. We incorporate into the system a skin-color filtering and a tracking stage to refine the pose estimations. Our implementation has been completely developed in C++, which makes our model dramatically faster than the original approach of [12]. Furthermore, our experimental validation shows that the proposed solution is able to outperform the state-of-the-art results in the *Biwi Kinect Head Pose Database* (Biwi) [11] using only RGB images.

The rest of the paper is organized as follows. In the next section, a review of related work is given. Section 3 introduces the proposed system. Section 4 presents the experimental evaluation. We conclude in Section 5.

2 Related Work

The literature contains several works on head pose estimation [20], which can be conveniently divided depending on whether they use RGB images, depth data or both.

In general, approaches that use RGB images are sensitive to illumination and the lack of distinctive features. Therefore, some of the recent works use depth as the primary cue. Breitenstein *et al.* [8], first compute hypotheses of nose locations from high-resolution depth images, and then minimize an error function between these hypotheses and the reference pose images. Lately, some works propose to use a RF combined with a Hough voting strategy to solve the problem [11,12,18,19]. Fanelli *et al.* [19] introduce the use of RF for real time head pose estimation from high quality range scans. The approach was

later adapted to depth data from consumer depth cameras [11,18]. In [12], the authors propose to extend the HF voting with the concept of *Probabilistic Locally Enhanced Voting* (PLEV), a regression strategy which consists in modulating the regression with a kernel density estimation to consolidate the votes in a local region near the maxima detected in the Hough space. Schulter *et al.* [1] have improved the RF proposing the novel Alternating Regression Forest, which are able to minimize a global loss to obtain better generalization, in contrast to the local minimization process employed during the learning of traditional RF. Finally, Riegler *et al.* [16] describe a method that combines convolutional neural networks with the idea of Hough Forests for a continuous head pose estimation.

Within the 2D image-based algorithms, we focus here on appearance-based methods. In contrast to our model, a common appearance-based approach is to discretize the head poses and learn a *separate* detector for each of them, *e.g.* [2,3,9,17]. Several works rely on statistical models of the face shape and appearance, *e.g.*, Active Appearance Models [4] and their extensions [5,6,7], but their focus is usually on detection and tracking of facial features. These methods rely on coarse quantizations of the poses for multi-view face detection, instead of considering that the pose estimation is ultimately a continuous problem. Recently, Demirkus *et al.* [21] have proposed a hierarchical temporal graphical model to estimate a continuous head pose angle from real-world videos. The introduced methodology provides a probability density function (PDF) of head poses for each video frame, rather than a single decision. However, only the yaw angle is considered, while our solution performs estimations simultaneously for the pitch, yaw and roll angles.

We build our approach on the work described in [12]. So, ours is also a HF based strategy. However, unlike [1,11,18,19], we present a model which does not need depth information but RGB images. Specifically, we have developed a compact implementation of the HF+PLEV [12] using C++. In order to get a fast system for HCI, we have extended the HF+PLEV [12] to work with a preprocessing filter for selecting face candidate regions, implementing a skin-color detection step. This preprocessing speeds up the pose estimations. Finally, we feed a tracking model *only* with the pose estimations of the face candidate regions to improve the predictions. We evaluate the use of two tracking solutions: a Kalman Filter [22] and the Condensation algorithm [23]. All these extensions let us build a final model which is faster than the original HF+PLEV [12], and also more precise – our results show that our implementation improves the results reported in [12].

3 The model

3.1 Learning a HF for Head Pose Estimation

A typical RF [13] is an ensemble classifier consisting of a set of randomized decision trees. During training, a binary weak classifier is learned for each non-leaf node. At runtime, test samples are passed through the trees, and the output is computed by averaging the distributions learned at the reached leaf-nodes.

HF [14] are a generalization of the Hough transform within the RF framework. The randomized trees are trained to learn a mapping from sampled d-dimensional features to their corresponding votes in a Hough space $\mathcal{H} \in \mathbb{R}^H$.

We build on the approach of Redondo-Cabrera *et al.* [12]. As it is described in [12], in the HF $\mathcal{F}$, we aggregate a set of T binary decision trees $\mathcal{T}_t(\mathcal{P}) : \mathcal{P} \rightarrow \mathcal{H}$, where $\mathcal{P} \subseteq \mathbb{R}^d$ is the d-dimensional feature space and $\mathcal{H} \subseteq \mathbb{R}^h$ describes the Hough space where the hypotheses are encoded. This Hough space lets us recover hypotheses for the location and the continuous pose of the head at multiple scales. Each head hypothesis $\mathbf{h} \in \mathcal{H}$ can be defined as $\mathbf{h} = (x_h, y_h, \theta_h, s_h)$, where x_h and y_h encode the face center (in our case, the tip of the nose), $\theta_h \in \mathbb{R}^p$ represents the continuous pose, and s_h identifies the scale.

Each of the decision trees $\mathcal{T}_t$ is built using a set of sampled patches $P_i = \{(\mathcal{I}_i, c_i, d_i, \theta_i)\}$, where $\mathcal{I}_i = \left\{I_i^1, I_i^2, ..., I_i^N\right\}$ defines the appearance of the training image, I_i^j is the j^{th} appearance channel, $c_i \in \mathcal{C} : \{0, 1\}$ is a class label (0 for a background sample and 1 for a head sample), $d_i = (x_i, y_i)$ encodes the relative 2D location of the face center to the sampled patch. And θ_i defines the continuous pose of the head. Training a single decision tree involves recursively splitting each node such that the training data in newly created child nodes is pure according to class label, relative 2D location and pose.

In our case, as in [12], the split function $f(\mathcal{I}_i; \tau_1, \tau_2, \{R_i\}_{i=1}^4)$ is characterized by the following parameters: the appearance channel specified by $\tau_1 \in \{1, 2, ..., N\}$, four asymmetric rectangles defined within the patch $\{R_i\}_{i=1}^4$, and a threshold $\tau_2 \in \mathbb{R}$ for the difference of average values of the rectangular areas. We then define

$$f(\mathcal{I}_i; \tau_1, \tau_2, \{R_r\}_{r=1}^4) = \begin{cases} 0 \text{ if } f_a(\mathcal{I}_i; \tau_1, \{R_r\}_{r=1}^4) < \tau_2, \\ 1 \text{ otherwise.} \end{cases} \tag{1}$$

with $f_a(\mathcal{I}_i; \tau_1, \{R_r\}_{r=1}^4) = |R_1|^{-1} \sum\limits_{q \in R_1} I_i^{\tau_1}(q) - \sum\limits_{r=2}^{4} \left(|R_r|^{-1} \sum\limits_{q \in R_r} I_i^{\tau_1}(q) \right)$.

Each node chooses the best splitting function by optimizing one of the following three impurity measures, $\mathcal{M}_*(\mathcal{S})$, which are chosen randomly during training. The class label impurity is measured as in [15] by

$$\mathcal{M}_c(S) = H(S) - \sum_{child \in (left, right)} \frac{S^{child}}{S} H(S^{child}), \tag{2}$$

where $H(S)$ is the entropy given by $H(S) = -\sum\limits_{c=0}^{1} p(c|S) \log(p(c|S))$, and $p(c|S)$ indicates the empirical distribution over classes within the set S.

The impurity of the relative 2D patch location, as in [14], is defined by

$$\mathcal{M}_d(S) = \sum_{child \in (left, right)} \sum_{j: c_j = 1} ||d_j - \tfrac{1}{|S^{child}|} \sum_{k: c_k = 1} d_k||^2. \tag{3}$$

And the impurity of the head pose is computed as in [12]:

$$\mathcal{M}_p(S) = \sum_{child \in (left, right)} \sum_{j: c_j = 1} \left(\tfrac{min\{(||\theta_j - \boldsymbol{\theta}_A||), 360° - (||\theta_j - \boldsymbol{\theta}_A||)\}}{180°} \right)^2, \tag{4}$$

where $\boldsymbol{\theta_A}$ is the viewpoint angle average over all foreground patches in the set S^{child} and it is computed taking the cyclic nature of pose angles into account.

3.2 Fast Head Detection and Pose Estimation for HCI

Given a test image, see Figure 1, we start performing a *pixel-based* skin detection method. This can be considered a preprocessing step to find candidate face regions.

Our pixel-based skin detector uses a thresholding method to classify skin and non-skin pixels [10]. This kind of skin detector defines the boundaries of the skin cluster in certain color spaces using a set of fixed skin thresholds.

After the skin detection step, image patches are sampled only from the candidate regions identified. We also incorporate a stride parameter, which controls how densely patches are extracted, thus easily steering speed and accuracy of the regression.

These patches traverse the learned trees and cast votes to the multidimensional Hough voting space $\mathcal{H} \subset \mathbb{R}^{2+p}$ based on the location and pose distributions stored in the leaves. Note that p is the number of angles that defined the continuous pose of the face. The forest-based estimate is then computed by aggregating votes from different patches at different scales $\{s_1, s_2, \ldots, s_S\}$. Following a standard HF regression approach [14], votes are accumulated in an additive way into the corresponding Hough voting spaces $\{\mathcal{H}^1, \mathcal{H}^2, \ldots, \mathcal{H}^S\}$, where $\mathcal{H}^i \in \mathbb{R}^{2+p}$. Then, these Hough spaces are stacked and scaled, so the maxima can be jointly localized at multiple scales. For finding these maxima, we use the procedure described in [12] named PLEV, where a local Hough region ($H_r^{\hat{\mathbf{h}}} \subset \mathcal{H}^i$), rather than a single Hough maximum, is considered for the regression of the pose. PLEV aggregates all *pose* votes received in $H_r^{\hat{\mathbf{h}}}$, obtaining a global distribution $g_r^{\hat{\mathbf{h}}}$ in the Hough region, which can be computed as,

$$g_r^{\hat{\mathbf{h}}} = \sum_{v_i \in H_r^{\hat{\mathbf{h}}}} \left(\sum_{L_j \rightarrow v_i} \frac{p(c=1|L_j)}{|L_j|} p(\theta|L_j, v_i) \right), \tag{5}$$

where $p(c=1|L_j)$ and $|L_j|$ encode the foreground likelihood and the number of patches in leaf L_j, respectively. $p(\theta|L_j, v_i)$ is the distribution of poses associated to the patches in leaf L_j which cast a vote in $H_r^{\hat{\mathbf{h}}}$, which we denote as $L_j \rightarrow v_i$.

We finally perform a Gaussian KDE on $g_r^{\hat{\mathbf{h}}}$ distribution in order to obtain a smooth probability density function for the pose estimation,

$$f_{g_r^{\hat{\mathbf{h}}}}(\theta) = \frac{1}{|g_r^{\hat{\mathbf{h}}}|} \sum_{\forall g_r^{\hat{\mathbf{h}}}(i)} \frac{1}{\sqrt{2\pi}} \exp\left(\frac{(\theta - g_r^{\hat{\mathbf{h}}}(i))^2}{2h^2}\right), \tag{6}$$

with h the bandwidth and $|g_r^{\hat{\mathbf{h}}}|$ the total number of voting positions considered. At last, our final hypothesis for the pose $\hat{\theta}$ is obtained as $\arg\max_{\hat{\theta}} = f_{g_r^{\hat{\mathbf{h}}}}(\theta)$.

3.3 Head Pose Estimation Tracking

The approach in [12] works on each frame independently. Here we propose to *refine the pose estimation* with a tracking solution (see Figure 1). Our aim is to improve the accuracy of the predictions for head pose values $\hat{\theta}$, avoiding noisy results. For doing so, we integrate into the system a tracking approach to obtain the refined estimations $\hat{\theta}^*$. In our model, the tracking is done for each of the angles of the face (pitch, yaw and roll) independently. We experimentally validate the use of two off-the-shelf trackers: a Kalman filter (KF) [22] and a Particle filter (PF) [23].

We perform a validation process which affects to the initialization and configuration parameters of each of the tracking filters. This validation phase is based on a triple tuning process, one for each angle to estimate in $\hat{\theta}$, as they are tracked with separated filters.

We first take the trained HF+PLEV $\mathcal{F}$, and choose a set of training images to tune the tracking parameters. With the PF, we have performed the validation to tune the following parameters: a) the number of samples generated by the filter (N), b) the number of iterations, and c) the characteristic parameters of the dynamical model $(A, \bar{x}, B)$, which is of the following form:

$$x_t - \bar{x} = A(x_{t-1} - \bar{x}) + Bw_t, \tag{7}$$

where w_t are independent vectors of independent standard normal variables, x_t is the state-vector, $\bar{x}$ is the mean value of the state, and A and B area matrices representing the deterministic and stochastic components of the dynamical model, respectively.

For the KF, the validation step consists in adjusting the initialization for the following matrices: a) process noise covariance (Q), b) measurement noise covariance (R), c) posteriori error estimate covariance (P) and d) measurement (H) [22]. We initialize the first predicted state of the KF according to $\hat{\theta}$.

4 Experiments

4.1 Experimental Setup

Data set. We report the performance of our model using only the RGB images provided in the dataset *Biwi Kinect Head Pose Database* (Biwi) [11]. It contains over 15K images of 20 people. The head pose range covers about ± 75 degrees yaw and ± 60 degrees pitch. Ground truth is provided in the form of the 3D location of the head and its rotation. Following [11], we split the database into two sets: a testing and training set of respectively 2 (subjects 1 and 12) and 18 subjects. We use the training subject 24 for validation.

Implementation details. We have developed the complete system in C++, porting all the Matlab functions of the original HF+PLEV [12].

During training, the positive examples are cropped and rescaled to the same size, chosen so that the largest bounding box dimension is equal to 100 pixels.

20 positive and 20 negative patches, with size of 32×32 pixels, are randomly extracted from each training image. Our forests have 15 trees with a maximum depth of 20. In each node, 20.000 binary tests are considered during learning. For the PLEV we consider a neighborhood size of 11×11 pixels. We use the 32 feature channels used in [14].

For the skin-color detector, we apply an image conversion to the YCrCb color space. The color filter restrictions allow to recognize as skin pixels those complying: Y$\in [15, 235]$,Cb$\in [60, 122]$,Cr$\in [135, 170]$. A filtering of all the candidate regions is performed, discarding regions whose area in pixels is out of the interval $[4000, 40000]$, or whose aspect ratio ($\frac{height}{width}$) does not belong to the interval $[0.5, 3]$. Following the validation process described for the tracking filter parameters, these are values chosen. For the PF: $N = 1300$, $N_{iterations} = 500$, $\bar{x}_{yaw} = 0.1$, $\bar{x}_{pitch} = 0.15$, $\bar{x}_{roll} = 0.15$, $A_{yaw} = 0.5$, $A_{pitch} = 7$, $A_{roll} = 0.5$, $B_{yaw} = 0.079$, $B_{pitch} = 0.3$, $B_{roll} = 0.079$. For the KF: $H = I$, $Q = I10^{-4}$, $R_{yaw} = I10^{-1}$, $R_{pitch} = I10^{4}$, $R_{roll} = I10^{3}$, $P_{yaw} = I10^{1}$, $P_{pitch} = I10^{8}$, $P_{roll} = I10^{5}$. I denotes the identity matrix.

4.2 Face detection and head pose estimation results

Results are reported in Table 1, offering a comparison between the state-of-the-art using depth and RGB images.

First, it is worth mentioning that our estimations for the pitch, yaw and roll reduce the error with respect to [12], even for a lower missed frames ratio (1.4%). Moreover, our model, using the same trained forest than in [12], and considering that the tracking is only done for the pose estimations (not for the face localizations), achieves a face detection performance slightly higher than in [12]. Results show that the KF is able to mainly reduce the pitch and roll errors. The PF lets us reduce all the pose errors, and especially those associated to the roll. We experimentally observe that the PF casts a slightly superior accuracy than the KF. In conclusion, our approach outperforms both the face detection and head pose estimation state-of-the-art results reported in [12] when only RGB images are used.

Note that our model, using simply RGB data, outperforms the results of methods which need depth images, such as [18,1,16]. It is relevant that our error for the roll estimation is the lowest reported using this dataset. Regarding to the yaw and pitch errors, the increment of our errors is $1.5°$ and $1.7°$, respectively, compared to the winning methods using depth. For the nose direction error, our method is far from [18], but note that their missed frames ratio is higher. For a comparable missed frames ratio, *i.e.* [16], our method offers a slightly better nose direction estimation. As a conclusion, we can claim that our model is able to improve the state-of-the-art-results for the problem of head pose estimation from RGB images, which is an important contribution. Qualitative results are provided in Fig. 3.

Our model has been designed to perform a fast estimation for HCI, while the accuracy does not degrade. In the following experiment, we evaluate the performance of our approach as a function of the stride parameter. Recall this

Table 1: Results using the Biwi Kinect Head Pose Database.

Images	Model	Position Error (mm)	Direction Error (°)	Yaw (°)	Pitch (°)	Roll (°)	Missed (%)
Depth	[16]	8.1 ± 5.3	9.8 ± 8	$\mathbf{3.8 \pm 3.7}$	6.7 ± 6.6	4.3 ± 4.9	**1**
	[1]	10.8 ± 6.1	12.2 ± 9	$5.5° \pm 5.8°$	$7.8° \pm 7.9°$	$5° \pm 4.4°$	3
	[12]	$\mathbf{7.2 \pm 12.1}$	7.3 ± 5.9	$4.1° \pm 6.9°$	$3.9° \pm 4°$	$3.2° \pm 3°$	5
	[11]	12.2 ± 22.8	$\mathbf{5.9 \pm 8.1}$	$3.8 \pm 6.5°$	$\mathbf{3.5 \pm 5.8°}$	$5.4 \pm 6.0°$	6.6
	[18]	14.7 ± 22.5	–	$9.2 \pm 13.7°$	$8.5 \pm 10.1°$	$8 \pm 8.3°$	**1**
Images	Model	Position Error (pixels)	Direction Error (°)	Yaw (°)	Pitch (°)	Roll (°)	Missed (%)
RGB	[12]	3.2 ± 1.4	9.8 ± 6.8	5.8 ± 5.9	5.8 ± 4.8	3.5 ± 3.4	2.4
	Ours + KF	$\mathbf{3 \pm 1.4}$	9.4 ± 7.5	5.8 ± 6.4	5.1 ± 4.6	3.1 ± 3.3	1.4
	Ours + PF	$\mathbf{3 \pm 1.4}$	9.1 ± 6.9	5.3 ± 6.3	5.2 ± 4.4	$\mathbf{2.8 \pm 2.9}$	1.4

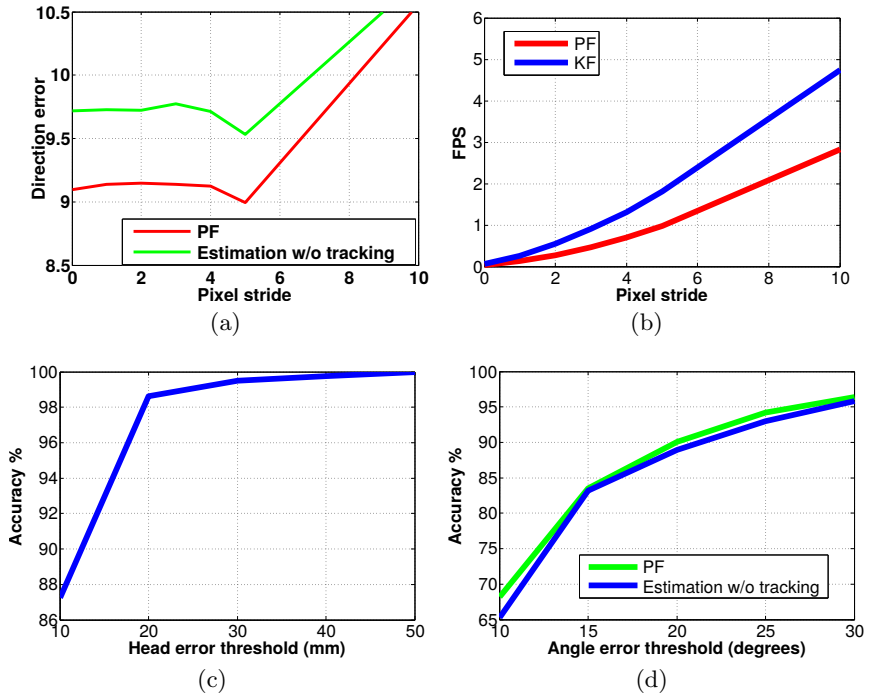

Fig. 2: (a) and (b) quantitative results as a function of the stride pixel parameter. (a) Errors for nose direction. (b) Frames per second (FPS). Frame accuracy of our method for different success thresholds of (c) the head position in mm and (d) the head pose in degrees.

parameter controls the spatial sampling of the patches to be extracted within an input candidate region identified by the skin detection step. It can be tuned to find the desired trade-off between accuracy and runtime of the process. Figure 2(a) shows how the direction error varies with respect to the stride. We can observe that our model with PF systematically reports better results than our implementation without the tracking. This reveals that the tracking solution implemented really improves the performance of the system. Figure 2(b) shows the frames per second of our implementation for different stride values. It is relevant to note that our approach is able to process up to 3 frames per second, using the PF, reporting a direction error lower than 11°.

Finally, we report the accuracy as a function of the success threshold for the head localization error and for the pose estimation error in Figures 2(c) and 2(d), respectively. We can observe (see Figure 2(c)) that our model gives a good performance for all success thresholds in the head position regression. It achieves a 87.24% for the most restrictive threshold of 10 mm. With respect to the angle error threshold used to evaluate the pose estimation, Figure 2(d) shows that the tracking with PF improves the accuracy of the model. It is worth to mention that for a restrictive threshold of 10°, our implementation with tracking reports a 68.2% of accuracy, compared to the 65% of our approach without the tracking stage.

Fig. 3: Qualitative results. Ground truth in blue, good estimations in green and wrong estimations in red.

5 Conclusion

We have proposed a fast HCI solution to estimate the location and continuous pose of the head. We combine the HF+PLEV model with a preprocessing step for the skin detection. Then the pose estimations are refined employing a tracking methodology. Our model simply needs RGB images, and it is able to report state-of-the-art results, while not relying on the availability of depth information.

Acknowledgements. This work is supported by projects CCG2013/EXP-047, CCG2014/EXP-054, TEC2013-45183-R, SPIP2014-1468, ERC Starting Grant COGNIMUND and the MECD Collaboration Grants 2014/15.

References

1. Schulter, S. and Leistner, C. and Wohlhart, P. and Roth, P.M. and Bischof, H.: Alternating regression forests for object detection and pose estimation. CVPR (2013)
2. Jones, M. and Viola, P.: Fast multi-view face detection. Technical Report (2003)
3. Morency, L.P. and Sundberg, P. and Darrell, T.: Pose estimation using 3D view-based eigenspaces. AMFG (2003)
4. Cootes, T.F. and Edwards, G. J. and Taylor, C. J.: Active appearance models. PAMI (2001)
5. Cootes, T.F. and Wheeler, G. V. and Walker, K. N. and Taylor, C. J.: View-based active appearance models. AMFG (2000)
6. Ramnath, K. and Koterba, S. and Xiao, J. and Hu, C. and Matthews, I. and Baker, S. and Cohn, J. and Kanade, T.: Multi-view aam fitting and construction. IJCV (2008)
7. Storer, M. and Urschler, M. and Bischof, H.: 3d-mam: 3d morphable appearance model for efficient fine head pose estimation from still images. Workshop on Subspace Methods (2009)
8. Breitenstein, M.D. and Kuettel, D. and Weise, T. and Van Gool, L.J. and Pfister, H.: Real-time face pose estimation from single range images. CVPR (2008)
9. Ding, H.X. and Fang, C.: Head pose estimation based on random forests for multiclass classification. ICPR (2010)
10. Vezhnevets, V. and Sazonov, V. and Andreeva, A.: A survey on Pixel-Based skin color detection techniques. GraphiCon (2003)
11. Fanelli, G. and Dantone, M. and Gall, J. and Fossati, A. and Van Gool, L.: Random Forests for Real Time 3D Face Analysis. IJCV (2013)
12. Redondo-Cabrera, C. and Lopez-Sastre, R. and Tuytelaars, T.: All together now: Simultaneous Object Detection and Continuous Pose Estimation using a Hough Forest with Probabilistic Locally Enhanced Voting. BMVC (2014)
13. Breiman, L.: Random Forests. Machine Learning (2001)
14. Gall, J. and Yao, A. and Razavi, N. and van Gool, L. and Lempitsky, V.: Hough Forests for Object Detection, Tracking, and Action Recognition. PAMI (2011)
15. Criminisi,A. and Shotton, J. and Konukoglu, E.: Decision Forests: A Unified Framework for Classification, Regression, Density Estimation, Manifold Learning and Semi-Supervised Learning. FTCGV (2012)
16. Riegler, G. and Ruther, M. and Bischof, B.: Hough Networks for Head Pose Estimation and Facial Feature Localization. BMVC (2014)
17. Ghodrati, A. and Pedersoli, M. and Tuytelaars, T.: Is 2D Information Enough For Viewpoint Estimation?. BMVC (2014)
18. Fanelli, G. and Weise, T. and Gall, J. and Van Gool, L.J.: Real time head pose estimation from consumer depth cameras. GAPR (2011)
19. Fanelli, G. and Gall, J. and Van Gool, L.J.: Real time head pose estimation with random regression forests. CVPR (2011)
20. Murphy-Chutorian, E. and Trivedi, M.M.: Head pose estimation in computer vision: A survey. PAMI (2009)
21. Demirkus, M. and Precup, D. and Clark, J.J. and Arbel, T.: Probabilistic Temporal Head Pose Estimation Using a Hierarchical Graphical Model. ECCV (2014)
22. Welch, G. and Bishop, G.: An Introduction to the Kalman Filter. (2006)
23. Isard, M. and Blake, A.: CONDENSATION – conditional density propagation for visual tracking. IJCV (1998)