Real-Time Head and Hand Tracking Based on 2.5D Data

Xavier Suau, *Student Member, IEEE*, Javier Ruiz-Hidalgo, *Member, IEEE*, and Josep R. Casas, *Member, IEEE*

***Abstract*—A novel real-time algorithm for head and hand tracking is proposed in this paper. This approach is based on data from a range camera, which is exploited to resolve ambiguities and overlaps. The position of the head is estimated with a depth-based template matching, its robustness being reinforced with an adaptive search zone. Hands are detected in a bounding box attached to the head estimate, so that the user may move freely in the scene. A simple method to decide whether the hands are open or closed is also included in the proposal. Experimental results show high robustness against partial occlusions and fast movements. Accurate hand trajectories may be extracted from the estimated hand positions, and may be used for interactive applications as well as for gesture classification purposes.**

***Index Terms*—Gesture recognition, hand tracking, head tracking, interactivity, range camera.**

I. Introduction

GESTURE recognition technologies are being widely applied to countless applications. One may notice a global tendency to replace traditional control devices with vision-based Natural Human Interaction (NHI) solutions. In a first step towards this goal, tactile interfaces are already in the market and allow the suppression of traditional input devices such as keypads and mouse. Indeed, the mid-term objective is to obtain marker-less, gesture recognition systems which allow users to interact as naturally as possible, providing a truly immersive experience.

Tracking of body parts is a key aspect for many recognition systems. For example, being able to track the head and hands of a person may help navigating through menus on a TV screen or to select regions of interest on a broadcasted football match. Indeed, a visual feed-back may be provided to the user showing how gestures are being interpreted by the system, as in the case of any standard remote control. Furthermore, body tracking provides spatio-temporal information, such as trajectories of points of interest or joint angles of a body model. Such features may be exploited in a gesture classification step, enlarging the classifier input data, which may lead to a more robust classification and recognition.

Classically, image-based tracking solutions are separated into single-camera and and multicamera. Single-camera proposals obtain impressive results given the lack of available information. Very interesting works have studied how to extract human pose from single color cameras. With this purpose, Yan and Pollefeys [1] recover the articulated structure of a body from single images with no prior information. In their work, trajectories of segmented body parts are mapped on linear subspaces to model the global body movement. Guan *et al.* [2] obtain a synthetized shaded body. Body pose is estimated by searching into the learned poses, reflectance and scene lighting which most likely produced the observed pose. Brubaker *et al.* [3] use a simple model of the lower-body, based on physical walking movement called *Antropomorphic Walker*, proposed by Kuo [4]. Hasler *et al.* [5] propose a pose estimation algorithm which performs on mono and multiple uncalibrated cameras. Unfortunately, single color cameras inherently provide poor information, due to information loss originated from perspective projection to a single view point. Single-camera based methods are usually very specific and hardly generalize to different kinds of movement, scenes and view points.

On the other hand, multicamera based systems offer a more precise tracking, but nonportable and costly setups are required. Results in this area are excellent since complete 3-D movement information is available. Gall *et al.* [6] go beyond pose estimation and cover possible nonrigid deformations of the body, such as moving clothes. Sundaresan and Chellappa [7] predict pose estimation from silhouettes and combined 2-D/3-D motion queues. Neumann *et al.* [8] propose a full-body tracking in a special environment called SmartRoom. Alcoverro *et al.* [9] adjust a body model to the shape and size of a given user, incorporating a meshed flesh to calculate cost functions over silhouettes. A successful pose inference method according to these cost functions is performed by means of a particle filter. Corazza *et al.* [10] generate a person-wise model which is updated through ICP (iterative closest point) measures on visual-hull data. Pons-Moll *et al.* [11] combine video images with a small number of inertial sensors to improve smoothness and precision of the human body pose estimation problem. Nevertheless, these 3-D capture environments are very expensive and cumbersome to setup, since they require precise calibration and, usually, controlled illumination conditions. In addition, the computational cost of multicamera methods is prohibitive and real-time is hardly achieved.

Manuscript received September 29, 2011; revised February 27, 2012; accepted February 29, 2012. Date of publication March 05, 2012; date of current version May 11, 2012. This work was supported in part by the European Union's Seventh Framework Programme (FP7/2007–2013) under Grant agreement no 248138 and in part by the Spanish Ministerio de Ciencia e Innovación, under project TEC2010–18094. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Sethuraman Panchanathan.

The authors are with the Universitat Politècnica de Catalunya, 08034 Barcelona, Spain (e-mail: xavier.suau@upc.edu; josep.ramon.casas@upc.edu; j.ruiz@upc.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TMM.2012.2189853

A. Rise of Range Cameras

Recently, new families of sensors have appeared as suitable tradeoff solutions between 2-D and 3-D approaches. As an example, cameras which are based on the time-of-flight (TOF) principle, as explained by Kolb *et al.* [12], are able to deliver a depth estimation of the recorded scene. TOF cameras currently provide low-resolution images, resolutions of 176×144 (SR4000) [13] and 204×204 (PMD CamCube) [14] being the most commonly used nowadays.

New camera models based on structured light technology, such as the Kinect sensor, have appeared recently, opening a whole new field of research given the increasing resolution (up to VGA), frame-rate (about 30 fps) and depth estimation quality. Since the information extracted from these cameras is essentially the pixel depth, but restricted to a given viewpoint, we call it 2.5D information. Thus, 2.5D is a simplified and sampled 3-D (x, y, z) surface representation that contains at most one depth value (z direction) for every point in the (x, y) plane. The main advantage of range cameras is that complex multicamera setups are no longer needed to obtain depth information.

Both the Kinect and TOF cameras have an average depth estimation error of 1 cm. However, Kinect provides a more stable depth estimation, with fewer outliers and artifacts at depth edges (sharp depth changes), and also the advantage of providing a higher resolution.

In the literature, Bevilacqua *et al.* [15] track multiple persons in a crowded scene with a TOF zenital camera. Knoop *et al.* [16] propose a fitting of the 2.5D data with a 3-D model by means of ICP. Grest *et al.* [17] use a nonlinear least squares estimation based on silhouette edges, which is able to track extremities in adverse background conditions. Zhu *et al.* [18] propose a tracking algorithm which exploits temporal consistency over frames to estimate the pose of a constrained human model. Lehment *et al.* [19] propose a model-based annealing particle filter approach on data coming from a TOF camera. More recently, Plagemann *et al.* [20] present a fast method which localizes body parts on 2.5D data at about 15 frames per second. Ganapathi *et al.* [21] extend the work in [20] and extract full body pose by filtering the 2.5D data, using the body parts locations. Shotton *et al.* [22] presented recently the pose recognition algorithm running in the Microsoft Kinect depth sensor, which exploits an enormous dataset to perform a pixel-wise classification into different body parts.

Focusing on head tracking, Haker *et al.* [23] compute principal curvatures on 2.5D data as features to estimate the position of the head. Bohme *et al.* [24] improve Haker's proposal. Nichau and Blanz [25] present an algorithm to detect the tip of the nose in 2.5D data by comparing the extracted silhouette to an average head profile template. Malassiotis *et al.* [26] estimate head pose by fitting an ellipse to the 2.5D data and detecting the nose tip, which helps determining head orientation.

In this paper we propose a novel approach based on 2.5D data, which concatenates head tracking and hand tracking (see a final result in Fig. 1). Head position is estimated depending on the distance between the user and the camera. Such estimation is robust against partial occlusions and fast movements, and helps in defining a region where hands are likely to be found. Hands are detected and tracked in such region. The presented experiments are carried out with an SR4000 TOF camera and also with a Kinect camera, showing the flexibility of the proposed method.

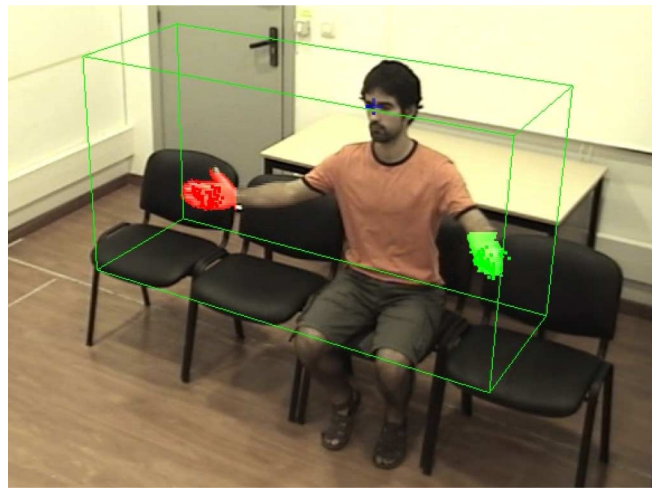

Fig. 1. Snapshot of the proposed $\mathrm{head} + \mathrm{hand}$ tracking system output. In this sequence, the head (blue cross) and both hands (green and red blobs) are being tracked. Movement is restricted to the hand workspace (green box), which is attached to the estimated head position. Results are presented on a lateral view for visualization purposes.

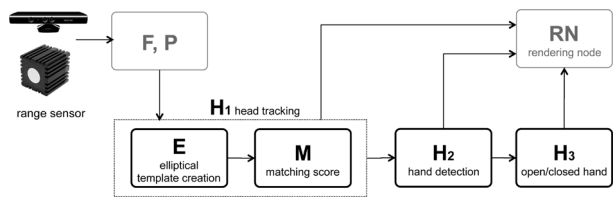

Fig. 2. Summarized block diagram of the proposed $\mathrm{head}+\mathrm{hand}$ tracking system, from the capture with a range camera to the final feed-back visualization on the rendering node (i.e., TV set). Both a Kinect sensor or a custom TOF camera have been used in this paper, highlighting the flexibility of the proposed approach.

The remaining of this paper explains the main parts of the tracking algorithm, summarized in the block diagram in Fig. 2. The head tracking algorithm is explained in Section III, followed by the hand tracking proposal in Section IV and the algorithm to detect open hands in Section IV. Experimental results on accuracy and speed are shown in Section VI, as well as a comparative summary with other recent works. Conclusions are drawn in Section VII, including ideas to improve the proposed approach in the future.

II. Preliminary Steps

Head and hands tracking depends on two previous processing steps to extract foreground and detect persons (blocks F and P in Fig. 2).

A. Foreground Extraction (F)

A foreground mask $\mathcal{F}$ is extracted from the 2.5D depth estimation images. In this paper, a simple and fast thresholding on

a previously learned background depth is used. The results may be appreciated in Fig. 4 (third column).

In order to suppress noisy samples, some consecutive background frames are considered, the background depth being the pixel-wise average depth of these frames. The experiments presented in this work use 10 background frames to calculate the scene background.

B. Person Detection (P)

A threshold on the number of $\mathcal{F}$ pixels is set, so that a first person is detected when the overall $\mathcal{F}$ pixels is larger than the mentioned threshold. An area filtering step is performed on the $\mathcal{F}$ to filter small noisy zones. The centroid of the main remaining area is computed and used for tracking initialization, as explained in Section III.

Once the head of the first person is tracked, a second strong increase of the number of pixels is assumed to be a second person in the scene, and so on. The heads of these persons are searched taking into account the already existing persons.

III. Head Tracking

Many persons at different distances from the camera may coincide in the same scene. Such persons may enter, exit and freely move around the camera field-of-view. Therefore, some aspects have to be taken into account when undertaking the head tracking problem.

- **Occlusions**: Partial and total occlusions are a common problem of (single viewpoint) visual analysis systems, since moving objects may overlap.
- **Apparent head size**: User's heads may be placed at different depth levels, resulting in different head sizes when projected onto the image plane.

In order to overcome such problems, we propose to firstly estimate the size of the head on the image (E), then estimate its position (M), which corresponds to the head tracking step H_1 in Fig. 2.

A. (E) *Head Size Estimation*

Human heads may present many different sizes on the recorded images depending on the depth level where they are placed. Nevertheless, most of people's heads are likely to have an elliptical shape, no matter the distance they are away from the camera. Furthermore, the elliptical shape of heads is invariant to rotation of the human body around the vertical axis. Such invariant properties related to the elliptical shape of heads is exploited thanks to depth estimations from the range camera. We remark that hair could strongly change the shape of the head. However, the effect of hair is reduced, given its low reflectivity and high scattering of IR light.

Indeed, we assume that people will stand-up or be seated, laying down and other poses are not considered. Therefore, the depth of the whole body is likely to be similar to the depth of the head. After the block (P) explained in Section II-B to the foreground mask of the image, we are able to detect when persons appear in the scene, as well as their centroids C_i. Thus, we assume that head's depth $d_{H_i} \approx d_{C_i}$.

An elliptical mask of the size of a regular adult head is placed at the calculated d_{H_i} depth level and projected onto the camera image plane, obtaining an ellipse of the apparent head size (H_x, H_y) in pixels (see Fig. 6). Such ellipse, which is called *template* or $(\mathcal{E})$ is then used to find a head position estimate.

(a) (b)

Fig. 3. On the right, a graphical example of the elliptical mask $\mathcal{E}$ used in the algorithm. On the left, a representation of a human head and its foreground mask $\mathcal{F}$, onto which a matching score is calculated.

In order to better distinguish between the head ellipse and other elliptical shapes in the scene, some margins are added to $\mathcal{E}$. More precisely, the upper, right and left margins are extended with background pixels; while the lower margin is not modified, as shown in Fig. 3.

B. (M) *Head Position Estimation*

With the aim of finding the image zone which better matches the elliptical shape, a matching score between the template ellipse $(\mathcal{E})$ and the global foreground mask $(\mathcal{F})$ is calculated at every pixel position (m, n) of the image. Such matching is performed within a rectangular search zone (see Section III-C), by shifting the template ellipse across the image plane. The matching score is calculated according to conditions C^k presented in (1), where $bg = background$ and $fg = foreground$. Conditions are checked at every pixel position $(u, v) \in \mathcal{E}$ of the template, which is itself centered at (m, n). When a condition C^k is satisfied, $C^k = 1$, otherwise $C^k = 0$. The final matching score for the pixel (m, n) is calculated as the sum of all the scores obtained on the template pixels, as shown in (1)

$$\begin{aligned} C^1_{u,v}: &\quad (\mathcal{E}_{u,v} = bg) \quad \wedge \quad (\mathcal{F}_{u,v} = bg) \\ C^2_{u,v}: &\quad (\mathcal{E}_{u,v} = fg) \quad \wedge \quad (\mathcal{F}_{u,v} = fg) \\ &\wedge \quad (|d_{u,v} - d_{H_i}| < d_{\max}) \\ C^3_{u,v}: &\quad (\mathcal{E}_{u,v} = bg) \\ &\wedge \quad (\mathcal{F}_{u,v} = fg) \quad \wedge \quad (|d_{u,v} - d_{H_i}| > d_{\max}) \\ \mathcal{M}_{m,n} = &\sum_{\forall\,(u,v)\in\mathcal{E}} \left(C^1_{u,v} + C^2_{u,v} + C^3_{u,v}\right). \end{aligned} \tag{1}$$

The pixel (m, n) with a higher score $\mathcal{M}^H = \max\{\mathcal{M}_{m,n}\}$ is selected, by simple max-pulling, as the best head position estimation $\hat{p}_H$ in a given search zone. Conditions C^2 and C^3 provide robustness against partial occlusions. Indeed, the elliptical shape of the head would be very polluted by occlusions, since we are working on a foreground mask $\mathcal{F}$ which does not take depth into account. By means of conditions C^2 and C^3, depth is incorporated to the matching score, not taking into account those pixels which are not consistent with the calculated head depth d_{H_i}. Remark that a depth threshold $d_{\max}$ is used to decide whether a depth value is consistent or not.

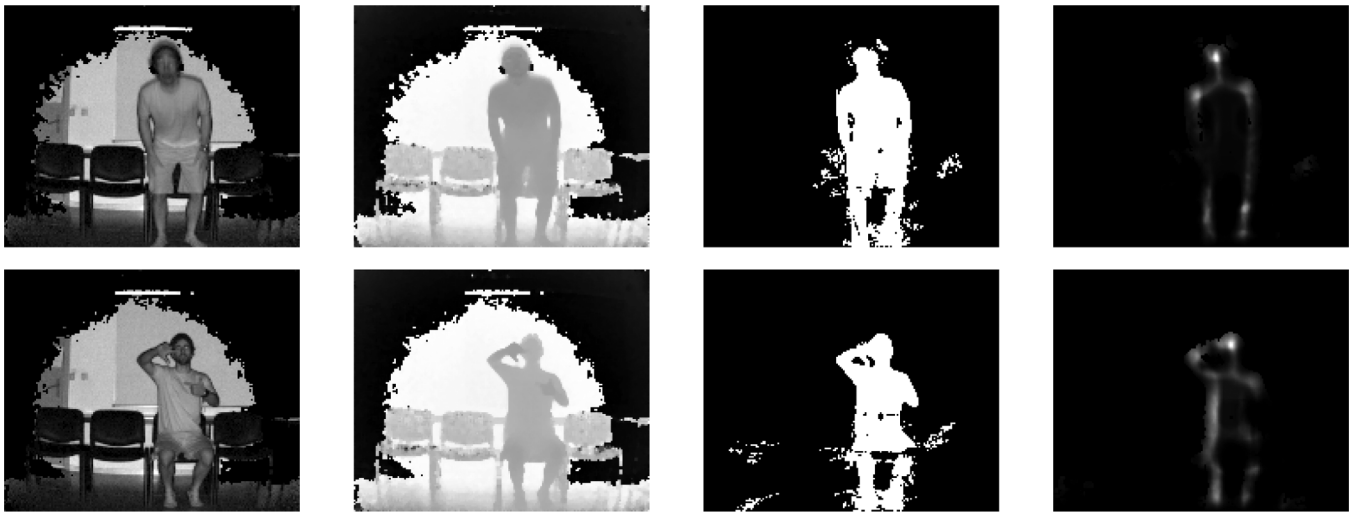

Fig. 4. Head position estimation in two video frames (each row) ontained with a SR4000 TOF camera. From left column to right: IR amplitude, 2.5D depth estimation, raw foreground mask and the obtained head matching score. The whitest zone is chosen as the most likely head position.

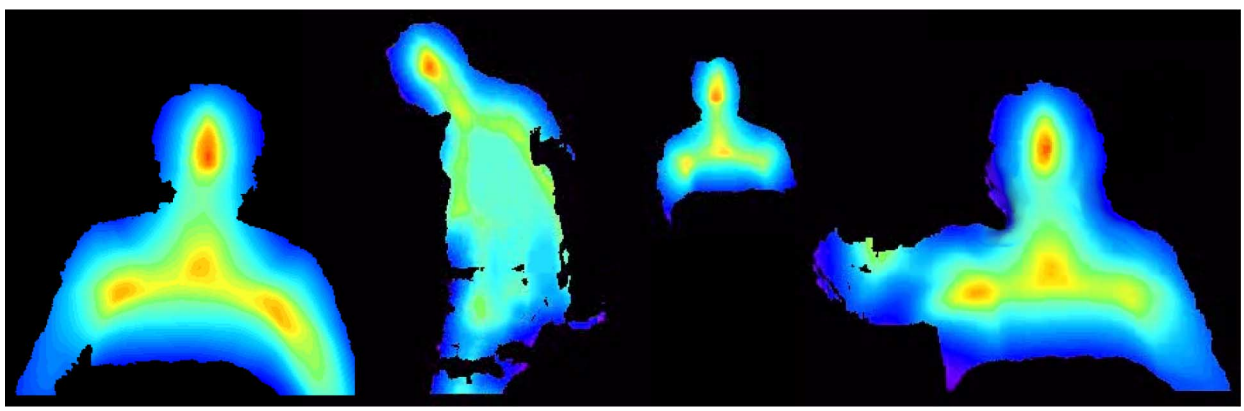

Fig. 5. Head matching score in various situations ontained with a Kinect camera, including (from left to right): back view, side view (with slight head tilt), far person and long-haired person. In all these cases, the matching score presents a maximum in the head zone. Indeed, the user viewpoint does not strongly affect our algorithm, since the elliptical shape of heads does not substantially vary with vertical rotation.

Two examples of the proposed solution are presented in Fig. 4. It may be seen how, even with a noisy $\mathcal{F}$ mask, the algorithm manages to find the best estimate in the center of the head. Such inherent robustness is still increased with the search zone resizing step, presented in Section III-C.

Furthermore, such approach is robust against horizontal head rotation and slight lateral head tilt, since the elliptical shape is still recognizable. As shown in Fig. 5, the proposed algorithm succeeds when dealing with side and back views of the head (rotation along the vertical axis), as well as with long-haired heads, even if a slightly lower score is obtained in such case.

Using either a SR4000 TOF camera or a Kinect camera has insignificant impact on the proposed head estimation. Of course, the higher resolution provided by Kinect makes our algorithm slower. For real-time experiments, the Kinect frames are downsampled by 4, obtaining a 160×120 px images, which are similar in resolution to TOF images.

C. Search Zone Resizing

In order to increase the robustness of the presented head estimation algorithm we propose to resize the search zone where the matching score $\mathcal{M}_{m,n}$ is calculated. The fact of stretching the search zone to the previous estimate helps ignoring other possible elliptical shapes in the scene. For example, a second person entering the scene could lead to two similar maxima of $\mathcal{M}_{m,n}$. However, by limiting the search zone size, such second head is not taken into account in the matching score $\mathcal{M}_{m,n}$. A second thread could be run on the remaining pixels to find and track the second head, as shown in Fig. 6.

Moreover, reducing the search zone drastically reduces the computational load of the algorithm, which processes the complete image only during the initialization frames.

The position and size of the head search zone is adapted to the head position variance, and also to the confidence on the estimation. More precisely, the new search zone size is calculated as a function of last matching score $\mathcal{M}^H$, the head size estimation (H_x, H_y), and the spatial variance of the estimation σ. The latter is calculated over the previous L frames, obtaining σ_x and σ_y for each axis. As for the matching score, it is normalized by the best achievable score for the current template $\mathcal{M}^{\max}$ such that $\bar{\mathcal{M}} = \mathcal{M}^H / \mathcal{M}^{\max} \in [0, 1]$.

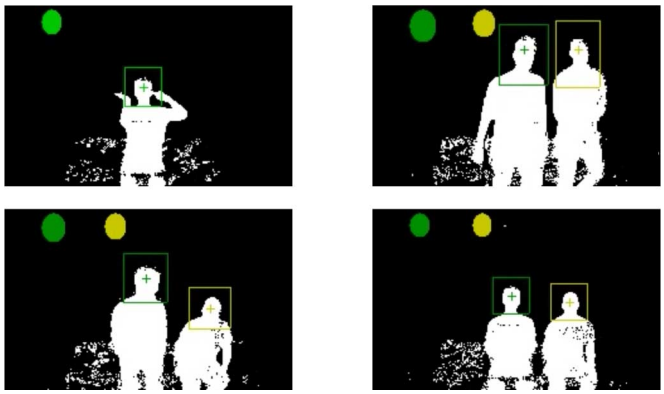

Fig. 6. Head tracking snapshots from our experiments obtained with the SR4000 TOF camera. Head position is estimated by shape matching with an ellipse which is continuously resized depending on the distance between the camera and the person. Ellipses being currently used are presented at the upper-left corner of the image. The rectangles in the image correspond to the current search zones.

The new rectangular search zone is centered at the last head position estimation, while the rectangle sides R_x and R_y are resized according to (2)

$$R_x = \sigma_x + (1+\mu)\cdot H_x$$
$$R_y = \sigma_y + (1+\mu)\cdot H_y \quad \text{with} \quad \mu = e^{-\bar{\mathcal{M}}/1-\bar{\mathcal{M}}}. \qquad (2)$$

Such resizing is effective against fast head movements as the search zone is adapted to the variance of the estimations. For example, horizontal movements will enlarge the search zone along the horizontal axis, as shown in Fig. 10(c).

Furthermore, including the matching score in (2) makes the system robust against bad estimations, making the search zone slightly larger in case of bad matching. The objective is to include some more pixels to the matching score computation in case some better matches appear close to the previous processed zone.

Note that when the head estimation is stable ($\sigma \approx 0$) and confident ($\bar{\mathcal{M}} \approx 1$), the search zone is about the size of a human head $(R_x, R_y) \approx (H_x, H_y)$.

IV. Hand Tracking

Hands are probably one of the most difficult parts of the body to track, given their mobility and size, but also one of the most important targets for many applications. Gesture recognition is tightly related to hand tracking, most of the information being obtained from hand movement. An accurate and robust hand tracking system is desirable to face complex gesture recognition, as well as to achieve interactive and immersive multimedia systems.

The hand tracking system proposed in this section (block H_2 in Fig. 2) relies on the robust head estimation presented in Section III. Hands are supposed to be *active* (performing gestures) in a zone placed in front of the body. Following this basic assumption, a hand workspace Ω is defined as a 3-D box of size $140 \times 70 \times 60$ cm, as shown in Fig. 1. Hands are supposed to lay within it when moving. Ω is attached to the head position $\hat{p}_H$ so that Ω follows the user's head at every time instant.

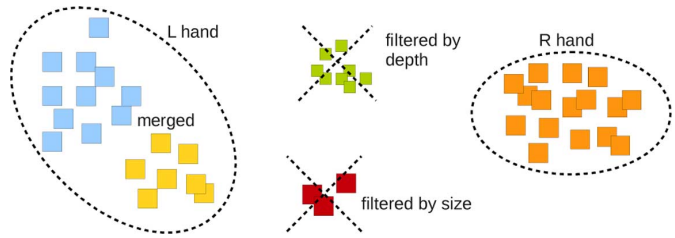

Fig. 7. Example of cluster merging and filtering for hand detection. Color represents candidate clusters obtained with a kd-tree structure. The red cluster is too small. The blue and yellow clusters are merged since the Hausdorff distance between them is small enough. Three clusters remain, but the green one is filtered since it is placed farther in depth (represented with smaller squares). Thus, the remaining two clusters are labeled as R and L hands.

Hands are to be detected among the 2.5D points in Ω. Dense clusters are searched by means of a kd-tree structure, which allows fast neighbor queries among 3-D point clouds [27], [28]. A list of candidate clusters is obtained and filtered according to the following sequential criteria.

1) **Merging**: Two clusters are merged as a single cluster if the Hausdorff distance between them is smaller than a given distance threshold $\delta_{\min}$ (typically $\delta_{\min} \approx 10$ cm).
2) **Size filtering**: The resulting merged clusters are filtered by size (number of points in cluster), keeping the largest ones. A size threshold $s_{\min}$ (typically $s_{\min} \approx 15$ cm^2) is set to determine what clusters are accepted as hand candidates.
3) **Depth filtering**: Clusters that fulfill the previous criteria are sorted by depth. Those clusters placed closer to the camera are selected, knowing that a maximum of two clusters may be chosen.

Thresholds $\delta_{\min}$ and $s_{\min}$ should be tuned depending on the type of camera and scene. A graphical illustration of these criteria is shown in Fig. 7. The number of detected hands depends on the number of clusters that pass the merging and filtering steps, resulting in two, one or no hands being detected in Ω. For example, two hands are being detected in the example of Fig. 1.

Following the proposed criteria, one could mislead the system by introducing, for example, an elbow in Ω. Such issue may be overcome by placing the Ω box at a convenient distance of $\hat{p}_H$. A distance of 25–30 cm has proven to be robust enough in our experiments with nontrained users.

In addition, spatio-temporal coherence is included in the hand tracking scheme, which increases its robustness. Hand estimates at time $t+1$ are compared to those in time t by means of Hausdorff distance measurements, in order to provide coherence to tracking and avoid right-left hand shifting. Furthermore, when only one hand is being detected, it is labeled (right/left) depending on the previous estimates and its relative position in Ω. For example, a cluster placed further right in Ω probably corresponds to the right hand.

No significant differences have been appreciated between Kinect and TOF input data regarding hand detection.

V. Open/Closed Hand Detection

In some applications, knowing where the user hands are located may be enough to provide an interactive experience. How-

ever, being able to determine whether the user opens or closes hands may substantially improve the potential of that application, increasing the interactivity. This way, commands like grabbing and releasing objects in a virtual environment, selecting buttons or navigating through a panoramic scene, may be easily implemented.

Before describing how such open/closed hand decision is carried out, some concepts about the relation between apparent and physical area are discussed in Section V-A. The explanation focuses on the range camera case.

A. Apparent and Physical Area on 2.5D Data

Generally speaking, a real world object is seen as a 2-D area on the image plane when recorded with a camera. Such area, called apparent area $\tilde{A}$, depends on the distance between the object and the camera.

Since 2.5D data is obtained from a single viewpoint, the concept of apparent area arises. Depth estimates are obtained in pixels, organized as images. If a surface S with a physical area A is captured by the depth sensor, its area A has to be translated into an area on the image plane, or apparent area $\tilde{A}$, which is expressed in pixels. Therefore, the size of $\tilde{A}$ will vary depending on the distance z between the camera and the recorded surface, which makes it impossible to recover A from the image without knowing the magnitude z.

Such inconsistency may be straightforwardly resolved in the case of 2.5D data, since z is the estimated depth z_i for every pixel. A physical pixel-wise area A_i^{pix} may be assigned to a given pixel p_i with depth z_i. Such assignment depends on the estimated depth z_i and the optical behavior of the camera.

An empirical law Γ^C is experimentally obtained for a given camera by means of recording a surface of a known physical area at different depth positions (Fig. 8). The relationship between physical and apparent area, for a given pixel, which increases quadratically with depth, is formulated in (3), providing a valid approximation for reasonable depth levels $z \in (80,\ 430)\ \mathrm{cm}$. Equation (3) is the quadratic approximation of the samples in Fig. 8. Given the empirical nature of the approach, the camera parameters are not needed, so the approximation may be done without calibration

$$A_i^{pix} = \Gamma^C(z_i) \approx 1.12 \cdot 10^{-6} \cdot z_i^2 + 8.41 \cdot 10^{-5} \cdot z_i - 4.64 \cdot 10^{-3}. \tag{3}$$

Let S be a surface with an apparent area $\tilde{A}^S$. If S is sufficiently perpendicular to the camera, its physical area A^S may be approximated as shown in

$$A^S = \sum_{\forall\ p_i \in S} \Gamma^C(z_i). \tag{4}$$

The depth of each pixel z_i may be replaced by the mean depth z^S of the observed region, simplifying the physical area calculation using (5). The result finally depends on the number of points (or pixels) N^S in the area

$$A^S \approx \sum_{\forall\ p_i \in S} \Gamma^C(z^S) = N^S \cdot z^S. \tag{5}$$

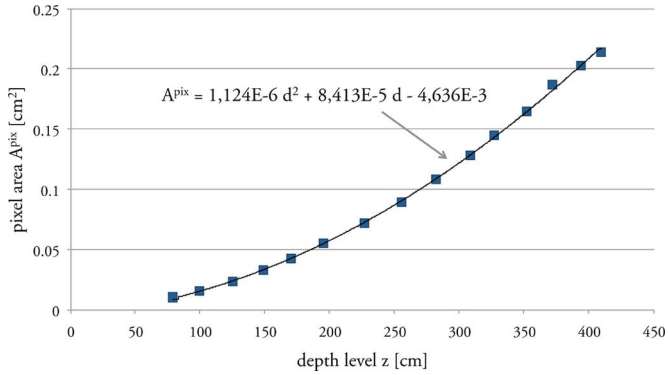

Fig. 8. Empirical estimation of the law Γ^C which gives the actual size of a pixel at a given depth level. Measurements have been carried out with a known flat surface (din A2 paper sheet) at various depth levels. The blue points are the division of the physical area of the paper sheet by the number of pixels it occupies on the image, which is equal to the physical area per pixel at that depth level. A quadratic approximation is shown in the figure.

B. Detecting an Open or Closed Hand

We propose to compute the area of the detected hands (Section IV) and threshold it to quickly decide whether it is closed or not. Such strategy assumes that the observed area is reasonably perpendicular to the camera. The latter condition is indeed not a very restrictive condition for interactive applications; when a user wants to show the system its open hand, instinctively places it as perpendicular as possible with respect to the optical axis of the camera.

In practice, such perpendicularity assumption enables the use of the approximation in (5). The law Γ^C giving a relationship between the physical area A^S, expressed in cm^2, and the apparent area N^S expressed in pixels.

The physical area of an open human hand perpendicular to the camera is about $A^S = 70 - 90\ \mathrm{cm}^2$, whilst that of a closed hand is about $A^S = 20 - 30\ \mathrm{cm}^2$. It seems reasonable to set an open-closed hand threshold of $A_{oc} = 50\ \mathrm{cm}^2$ to determine the hand status.

Indeed, many applications require robust open hand detection, while closed hands are not mandatory fists. Some hand poses with small physical area (i.e., pointing with one finger towards de camera) are also detected as closed hand. Such side-effect is not relevant for application purposes, since open hands are still robustly detected.

Remark that a threshold based on the physical area makes the approach independent of the position of the user. Indeed, the physical area does not vary and the threshold A_{oc} is always valid no matter the location of the person in the working range (depth between (80, 430) cm).

VI. Experimental Results

A. Head Tracking Results

Given the recent interest in 2.5D cameras and the variety of environments and scenes where they may be used, up to the author's knowledge, it does not exist a common dataset to which compare our results.

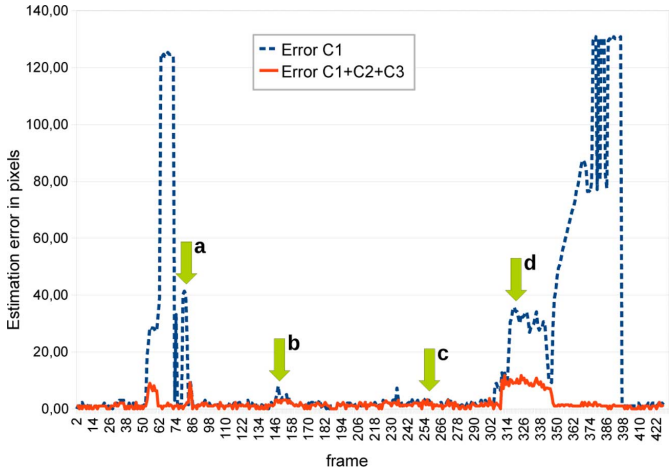

Fig. 9. Error between the obtained head estimations and the ground-truth. $C^1+C^2+C^3$ error does not go above 10 pixels, which is about the head radius. The C^1 version loses target twice, a reset of the algorithm being needed (frames 74 and 398). The labeled arrows in Fig. 9 correspond to the frames shown in Fig. 10.

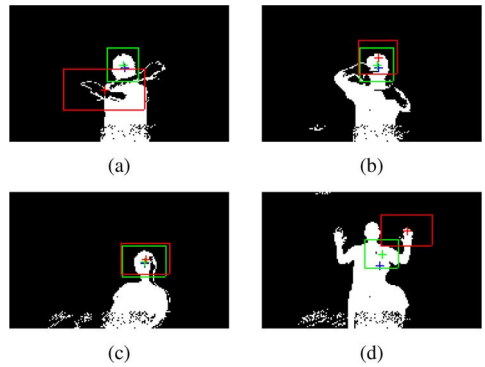

Fig. 10. Head tracking snapshots (SR4000 TOF camera) of the sequence presented in Fig. 9. In red, the C^1 version, in green the $C^1+C^2+C^3$ and in blue the ground-truth position. Note how the $C^1+C^2+C^3$ is more robust in these adverse situations (occlusions and clutter).

Therefore, in order to evaluate the performance of the proposed head tracking, we analyze the convenience of conditions C^2 and C^3 in (1), and how robustness is increased by exploiting 2.5D data. We compare the proposed algorithm with a reduced version which only verifies condition C^1. This way, the contribution of C^2 and C^3 is shown. The estimation error between a ground-truth (manually marked) head position g^H and the estimated head position is calculated as $\varepsilon = |g^H - \hat{p}^H|$, and presented in Fig. 9 for the two versions of the algorithm. Note that, even if C^1 plays the role of 2-D method, 2.5D data is used for the initial head size estimation. Both methods run on the same foreground extraction, obtained from the 2.5D depth images as explained in Section II-A.

Fig. 9 shows the estimation error of a sequence which contains two persons and three main events. Around frame 50, a single person waves hands before his head, hiding it. At frame 135, he performs a gesture with both hands which partially cover the head zone. At frame 300 the second person enters the scene and walks behind the first person. These events are illustrated in Fig. 10, showing the successful behavior against occlusions and clutter.

The proposed $C^1+C^2+C^3$ algorithm is able to track the first user's head despite these three polluting events, while the C^1 versions loses the target when the shape of the head is changed (hands moving, persons walking, etc.).

B. Head+hands *Tracking Results*

As stated in Section IV, hand tracking depends on the robustness of head tracking. A set of gestures have been considered to analyze the performance of hand tracking, including gestures with one or two hands such as waving, pointing or separating hands apart (see Table I). Hand centroids are manually marked on the 2.5D video sequences. Thanks to the depth information, 3-D ground-truth trajectories may be extracted from the 2-D manually marked points. Such 3-D ground-truth trajectories are used for comparison hereafter.

TABLE I
HAND DETECTION 3-D ACCURACY ON DIFFERENT GESTURES

Gesture	# frames	error R hand	error L hand
push	30	2.62 *cm*	-
circle	30	6.61 *cm*	-
replay	35	2.86 *cm*	-
hand up-down	115	5.87 *cm*	-
separate hands	75	2.36 *cm*	3.80 *cm*

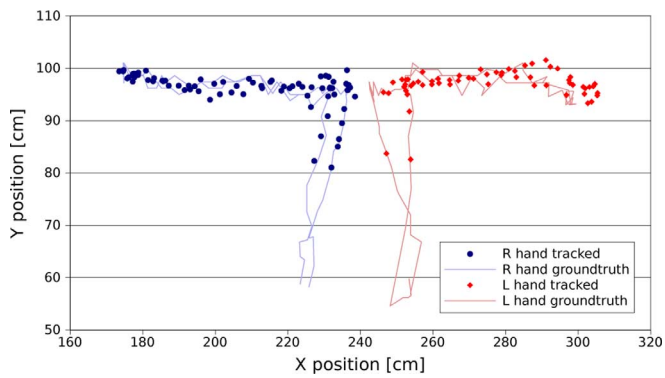

Fig. 11. Ground-truth and estimated trajectories of the (R)ight and (L)eft hands. The estimated hand positions are fairly close to the reference ground-truth positions. Only the XY projection of these 3-D trajectories is shown.

In Fig. 11 the marked and estimated trajectories of both hands are presented corresponding to a gesture where each hand moves horizontally, like a slow hand clap (only the frontal projection of these 3-D trajectories is shown for visual clarity). The error between the estimated and ground-truth trajectories is shown in Fig. 12.

Hands are detected from the moment that they enter Ω (frame 7). During the first 10 frames, both hands are touching each other, misleading the tracker which only *sees* one hand. However, once they are separated for a few cm, the tracker is able to detect and track both hands. It should be emphasized that both ground-truth and estimated trajectories are polluted by noise from the depth estimate. In addition, ground-truth trajectories have been extracted by hand, selecting a reasonable hand center which may vary in some cm along frames. Despite these adverse

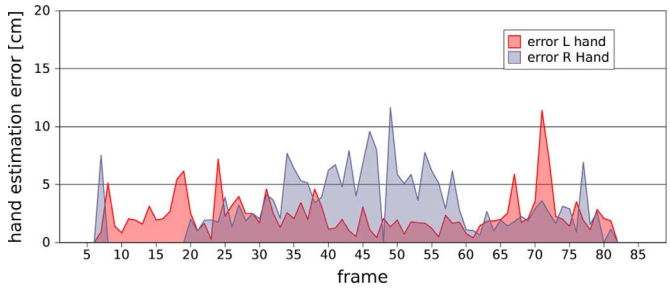

Fig. 12. Hand estimation error, calculated as the Euclidean distance between the estimated and ground-truth 3-D positions. The maximum error is about 10 cm for this sequence.

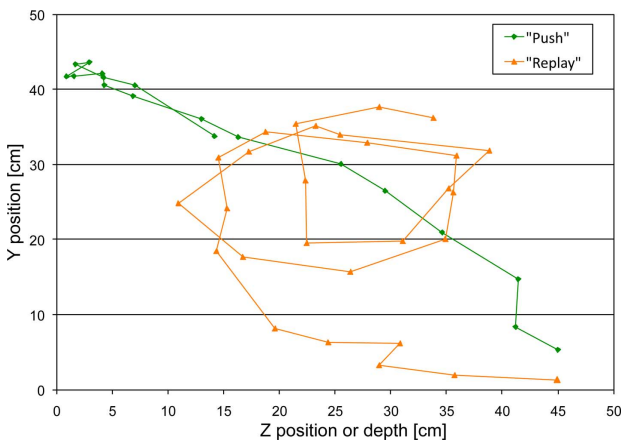

Fig. 13. Trajectories obtained in a real-time experiment for the *push* and *replay* gestures. These results could be an intersting input for a classification step. Only the YZ plane is presented as movement is mainly contained in such plane.

noisy conditions, the hand estimation error rarely goes above 10 cm.

Table I summarizes the average error for different one-handed and two-handed gestures gestures, computed as the average 3-D error (with respect to the ground-truth hand positions) along the duration of the gestures. For example, the errors for *separate hands* gesture are calculated from the errors in Fig. 12. Even if accuracy along the depth axis depends on the type of sensor, we found it interesting to include it in the overall error, since both the Kinect and SR4000 used sensors provide a similar ~ 1 cm depth precision.

The average error is higher for fast movements such as the *circle* and the *up-down* gestures, resulting in about 6 cm of error. For the other gestures, the error is of about 3 cm, which is fairly adequate given the size of a human hand.

For the sake of illustration, the trajectories corresponding to the *push* and *replay* gestures are presented in Fig. 13. The *push* gesture consists in extending one arm from the body towards the camera and backwards. The *replay* gesture consists in describing circles with one hand in the YZ plane. Since both gestures are representative on the YZ plane, only these coordinates are presented, relative to the hand workspace Ω origin. The nature of the gesture may be easily derived from the trajectory, hence, such results may be interesting for further classification purposes.

Fig. 14. Snapshot of a real-time demonstration of the proposed head+hand tracking system using the Kinect camera. In the scene, a user is able to navigate through a panoramic recording of a football game. Pan and tilt movements are done by grabbing, dragging and releasing the viewport with one hand. Zooming is performed in a similar way but with both hands, the distance between them being the current zoom factor.

C. Public Demonstrations

The proposed method has been set up for many public demonstrations (Fig. 14). Satisfactory qualitative results and feed-back of the users have been retained up to the moment. The reliability of the algorithm has specially been assessed in two main shows.

- The 2011 *International Conference on Multimedia and Expo* (ICME) in Barcelona, as a demonstration of the work in [29] (of which this paper is an extension). Over 70 persons where able to test the demonstration at ICME.
- The 2011 *International Broadcasting Convention* (IBC) in Amsterdam, as a part of the FascinatE project [30]. At IBC, the hand tracking positions where used to control a panoramic video stream, being able to navigate within it (pan, tilt, zoom) and to grab the screen with the open/closed hand detector. Over 150 persons where able to interact with the demonstration with very encouraging results and feed-back.

After these public experiments, some conclusions where drawn. The head detection algorithm performed with extreme robustness, only missing the head position at initialization for very few users (less than 1%). The hand tracker also proved to be robust with many different users (in height, shape, hand size, arm length, etc.). Open and closed hand distinction was somehow hard for users with very big or small hands. However, after some tries, users got used to the system operation, all of them managing to properly interact.

D. Execution Speed

The main applications of the proposed head+hand tracking algorithm require real-time processing, otherwise natural interactivity would not be possible. In order to evaluate the real-time performance of our proposal, execution speed experiments have been carried out on a Intel Xeon 3 GHz CPU.

Real-time experiments are performed with a Kinect depth camera, which is down-sampled by a factor of 4, resulting in depth images of $R_4 = 160 \times 120$ pixels, which is the resolution used for the accuracy experiments presented in Sections Sections VI-A and VI-B. Experiments are run in sequences

TABLE II
COMPARATIVE SUMMARY

Authors	Camera	Resolution	Full body?	Speed	GPU?	Accuracy
Knoop *et. al* [16]	SR4000	176×144	yes	$10 - 14\,fps$	no	-
Grest *et. al* [17]	PMD	64×48	yes	$5\,fps$	no	-
Zhu *et. al* [18]	SR3000	176×144	no	$10\,fps$	no	$3 - 10\,cm$
Lehment *et. al* [19]	XB3	1280×960	yes	$2.5\,fps$	yes	-
Ganapathi *et. al* [21]	SR4000	128×128	yes	$4 - 6\,fps$	yes	$10 - 20\,cm$
Proposed	Kinect, SR4000	160×120	no	$68\,fps$	no	$3 - 6\,cm$

where the head and both hands are detected, which is the worst case in terms of computational load.

After hundreds of experiments with different two-handed gestures, we obtain a processing frame rate $f^P \approx 68$ fps, which is largely enough for real-time applications. Moreover, it should be remarked that no GPU computing power is used in this proposal.

However, some experiments are also performed at other resolutions, specially $R_2 = 320 \times 240$ (down-sampling by 2) and $R_1 = 640 \times 480$ (original Kinect resolution). Real-time is slightly achieved with R_2, with a frame-rate of about 9 fps, while R_1 is too slow for any real-time purpose.

When dividing the overall processing time into tasks, it may be noticed that the head tracking task takes 92.4% of the total, while hand tracking is much faster (7.6%) of CPU time.

E. Comparison With Other Proposals

As explained in Section I-A, many works focused on 2.5D data have been presented recently. Their achievements and results are summarized in Table II.

The proposed method largely overcomes the referred State of the Art methods in terms of speed of execution. The fastest method is the work of Knoop *et al.* [16], with a frame-rate up to 14 fps using a similar resolution. Our proposal achieves 68 fps on a regular CPU, which is about $4 - 5\times$ faster than the cited method. Moreover, some of the proposals in Table II use GPU implementations [19], [21], which should speed their performance up.

As for the accuracy of the hand detection and tracking, the proposed method performs with an average error of about 3–6 cm while gestures are performed. Such error is similar to that claimed by Zhu *et al.* [18]. It should be remarked, in favor of the latter work, that hands are tracked at every time instant, no matter whether a gesture is being performed or not. This detail makes tracking more difficult, since hands may suffer of more occlusions or they may be located very near the body. Keeping an average error between 3–10 cm is impressive.

Ganapathi *et al.* [21] state that a 10 cm error may be considered as a perfect match with the ground-truth, and that an error smaller than 30 cm is good extremity match. In their work, they address the problem of a global full-body pose estimation and they detect all the visible extremities of a human body with an average error between 10 cm and 20 cm. Our proposal takes advantage of a local and focused approach to overcome such results, as far as hand and head are concerned. The tradeoff between accuracy and amount of information (full-body vs. only hands) is clearly observed after these results.

Recently, a Kinect SDK [31] has been released, providing a complete human body pose estimation at about 20 fps. Head and hands may be extracted from the complete body, providing a comparable usage to the proposed method with a similar accuracy. However, the tracking in [31] is slower than the proposed method, and it is partly performed in the Kinect hardware, reducing the flexibility of the approach.

VII. CONCLUSION

Real-time head and hands tracking is a crucial issue for many gesture recognition systems. In this paper we have proposed a fast algorithm for head and hands tracking based on 2.5D data obtained from a range camera.

The proposed algorithm performs head tracking in a first step. With this aim, an elliptical head template is adapted to the depth level where the person is placed. Such template is projected onto the camera image plane where a matching score is calculated, in order to find the best head position estimation by simple max-pulling. Tracking robustness is increased with an adaptive resizing of the search zone where the matching score is computed, depending on the confidence and variance of the estimation.

The hand tracking algorithm fully relies on the head position estimation. A box-shaped 3-D zone where hands are likely to be placed is attached to this head estimation, allowing the user to freely move in the scene. Hands are detected and tracked as 2.5D data blobs, which are filtered and merged in order to increase robustness and trajectory accuracy.

An empirical law relating physical and apparent area is obtained. We propose a very simple and efficient way to detect whether a hand is open or closed, quickly calculating its physical area from the apparent one, and thresholding it.

Experimental results show that the contribution of 2.5D data makes head tracking more robust than using only 2-D data. Indeed, situations with partial occlusions and clutter are resolved with an error smaller than the head radius. In our experiments, the head tracking does not lose target thanks to the inclusion of 2.5D data.

As for hand tracking, results show that hands are tracked with an average 3-D error of between 3 cm and 6 cm depending on the gesture being performed. Fast gestures are tracked worse than slow and smooth ones. Two-handed gestures are also tracked, with an average error smaller than 4 cm per hand. Such results are slightly better than the best state-of-the-art referred method, proposed by Zhu *et al.* [18]. In their work, they achieve an average error of 3–10 cm. However, their proposal is about $7\times$ slower. Other recent works such as [21], achieve errors between

10 cm and 40 cm, at the expense of providing full-body information.

The presented system executes at 68 fps on an Intel Xeon 3 GHz CPU (most of the experiments performed during 2011), which is fast enough for real-time applications. Such execution speed largely overcomes the frame-rates available in the literature, which achieve speeds of, at most, 14 fps in [16] (Pentium 4 at 3.2 GHz), 10 fps in [18] (3 GHz PC) or 4–6 fps in [21] (specific GPU implementation). However, works in the literature focus on full body tracking and pose estimation, while our proposal strongly focuses on hand and head, reducing the computational load at the expense of losing information (i.e., feet, elbows, etc.).

Many improvements to this proposal are foreseen. Exploiting hand estimation to improve head estimation and vice-versa (cross feedback) is currently under study. Other setup proposals are considered, such as including a color camera, which will contribute with color and disparity information. Finally, fitting an upper-body model to these estimates is also considered. Such approach will help increasing the temporal consistency of the tracking, and also including inverse kinematics techniques to be able to perform a more complete and accurate tracking, as well as to recognize more complex gestures.

REFERENCES

[1] J. Yan and M. Pollefeys, "A factorization-based approach for articulated nonrigid shape, motion and kinematic chain recovery from video," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 30, no. 5, pp. 865–877, May 2008.

[2] P. Guan, A. Weiss, A. Balan, and M. Black, "Estimating human shape and pose from a single image," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2009, pp. 1381–1388.

[3] M. A. Brubaker, D. J. Fleet, and A. Hertzmann, "Physics-based person tracking using the anthropomorphic Walker," *Int. J. Comput. Vis.*, vol. 87, no. 1–2, pp. 140–155, Aug. 2009.

[4] A. D. Kuo, "Energetics of actively powered locomotion using the simplest walking model," *J. Biomech. Eng.*, vol. 124, no. 1, pp. 113–113, 2002.

[5] N. Hasler, H. Ackermann, B. Rosenhahn, T. Thormahlen, and H. Seidel, "Multilinear pose and body shape estimation of dressed subjects from image sets," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2010, pp. 1823–1830.

[6] J. Gall, C. Stoll, E. De Aguiar, C. Theobalt, B. Rosenhahn, and H.-P. Seidel, "Motion capture using joint skeleton tracking and surface estimation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2009, pp. 1746–1753.

[7] A. Sundaresan and R. Chellappa, "Multicamera tracking of articulated human motion using shape and motion cues," *IEEE Trans. Image Process.*, vol. 18, no. 9, pp. 2114–2126, Sep. 2009.

[8] J. Neumann, J. R. Casas, D. Macho, and J. R. Hidalgo, "Integration of audiovisual sensors and technologies in a smart room," *Personal and Ubiquitous Computing*, vol. 13, no. 1, pp. 15–23, 2007.

[9] M. Alcoverro, J. R. Casas, and M. Pardàs, "Skeleton and shape adjustment and tracking in multicamera environments," in *AMDO*, ser. Lecture Notes in Computer Science, F. J. P. López and R. B. Fisher, Eds. : Springer, 2010, vol. 6169.

[10] S. Corazza, L. Mündermann, E. Gambaretto, G. Ferrigno, and T. P. Andriacchi, "Markerless motion capture through visual hull, articulated ICP and subject specific model generation," *Int. J. Comput. Vis.*, vol. 87, no. 1–2, pp. 156–169, Sep. 2009.

[11] G. Pons-Moll, A. Baak, T. Helten, M. Muller, H. Seidel, and B. Rosenhahn, "Multisensor-fusion for 3d full-body human motion capture," *Elements*, pp. 2–9, 2010.

[12] A. Kolb, E. Barth, R. Koch, and R. Larsen, "Time-of-flight cameras in computer graphics," *Comput. Graph. Forum*, vol. 29, no. 1, pp. 141–159, 2010.

[13] MESA SR4000 MESA Imaging, 2011 [Online]. Available: http://www.mesa-imaging.ch/prodview4k.php

[14] PMD[vision] CamCube PMDTechnologies, 2011 [Online]. Available: http://www.pmdtec.com/products-services/pmdvisionr-cameras/pmd-visionr-camcube-30/

[15] A. Bevilacqua, L. Stefano, and P. Azzari, "People tracking using a time-of-flight depth sensor," in *Proc. IEEE Int. Conf. Video and Signal Based Surveillance*, 2006, pp. 89–89.

[16] S. Knoop, S. Vacek, and R. Dillmann, "Sensor fusion for 3D human body tracking with an articulated 3D body model," in *Proc. Int. Conf. Robot. Autom.*, May 2006, pp. 1686–1691.

[17] D. Grest, V. Krüger, and R. Koch, "Single view motion tracking by depth and silhouette information," *Lecture Notes in Computer Science*, vol. 4522, pp. 719–729, 2007.

[18] Y. Zhu, B. Dariush, and K. Fujimura, "Controlled human pose estimation from depth image streams," in *Proc. Int. Conf. Comput. Vis. Pattern Recognit. Workshops.*, Jun. 2008, pp. 1–8.

[19] N. H. Lehment, M. Kaiser, D. Arsic, and G. Rigoll, "Cue-independent extending inverse kinematics for robust pose estimation in 3D point clouds," in *Proc. Int. Conf. Image Process.*, Hong Kong, 2010, pp. 2465–2468.

[20] C. Plagemann, V. Ganapathi, D. Koller, and S. Thrun, "Real-time identification and localization of body parts from depth images," in *Proc. Int. Conf. Robot. Autom.*, 2010, pp. 3108–3113.

[21] V. Ganapathi, C. Plagemann, D. Koller, and S. Thrun, "Real time motion capture using a single time-of-flight camera," in *Proc. Conf. Comput. Vis. Pattern Recognit.*, San Francisco, CA, 2010, pp. 755–762.

[22] J. Shotton, A. Fitzgibbon, M. Cook, T. Sharp, M. Finocchio, R. Moore, A. Kipman, and A. Blake, "Real-time human pose recognition in parts from single depth images," in *Proc. Conf. Comput. Vis. Pattern Recognit.*, Colorado Springs, CO, 2011, pp. 1297–1304.

[23] M. Haker, M. Bohme, T. Martinetz, and E. Barth, "Geometric invariants for facial feature tracking with 3D TOF cameras," in *Proc. ISSCS*, Jul. 2007, pp. 3–6.

[24] M. Bohme, M. Haker, T. Martinetz, and E. Barth, "Head tracking with combined face and nose detection," in *Proc. Int. Symp. Signals Circuits Syst.*, Jul. 2009, pp. 1–4.

[25] M. Nichau and V. Blanz, "Pose-insensitive nose detection in TOF-scans," *Comput. Vis. Image Understand.*, vol. 114, no. 12, pp. 1346–1352, Dec. 2010.

[26] S. Malassiotis and M. G. Strintzis, "Robust real-time 3D head pose estimation from range data," *Pattern Recognit.*, vol. 38, no. 8, pp. 1153–1165, Aug. 2005.

[27] J. H. Friedman, J. L. Bentley, and R. A. Finkel, "An algorithm for finding best matches in logarithmic expected time," *ACM Trans. Math. Software*, vol. 3, no. 3, pp. 209–226, 1977.

[28] S. Maneewongvatana and D. M. Mount, "It's okay to be skinny, if your friends are fat," in *Proc. 4th Annu. CGC Workshop Comptutat. Geometry*, Oct. 1999, pp. 1–8.

[29] X. Suau, J. R. Casas, and J. Ruiz-hidalgo, "Real-time head and hand tracking based on 2.5D data," in *Proc. Int. Conf. Multimedia Expo*, Barcelona, 2011, pp. 1–6.

[30] Format-Agnostic Script-Based Interactive Experience FascinatE [Online]. Available: http://www.fascinate-project.eu/

[31] OpenNI PrimeSense, 2011 [Online]. Available: http://openni.org

Xavier Suau (S'09) received the degree in telecommunications engineering from the Universitat Politècnica de Catalunya (UPC), Barcelona, Spain, in 2007. In the same year, he received the degree in Aeronautics from the École Nationale Supérieure de l'Aéronautique et de l'Espace (SupAéro). Also in 2007, he received the M.Sc. degree rom SupAéro. In 2008, he joined the Image Processing Group at UPC, and since 2009 he holds a Ph.D. grant at UPC.

His current work is focused on exploiting range information for feature extraction and body pose estimation, in the framework of his Ph.D. and also in the FP7 FascinatE project.

Javier Ruiz-Hidalgo (S'00–A'06–M'10) received the degree in telecommunications engineering from the Universitat Politècnica de Catalunya (UPC), Barcelona, Spain, in 1997, and the M.Sc. and Ph.D. degrees from UPC in 1999 and 2006, respectively.

During 1999 and 2000 he worked in the ACTS(AC308) DICEMAN project developing new descriptors and representations for image and video sequences. From 2001 to 2003 he was also involved in the IST/FET(2000–26467) project MASCOT developing an efficient compression scheme exploiting metadata information. Since 2006 he is the principal investigator in the HESPERIA (CENIT-2006) project involved in developing new image algorithms in security applications. Since 2001 he has been an associate professor at UPC. He is currently lecturing on the area of digital signal and systems and image processing. His current research interests include image segmentation, still image and sequence coding, compression, and indexing.

Josep R. Casas (M'96–A'96–M'04) received the M.S, and Ph.D. degrees in telecommunications engineering from the Technical University of Catalonia (UPC), Barcelona, Spain, in 1990 and 1996, respectively.

He is currently an Associate Professor in the Department of Signal Theory and Communication, UPC. He was visiting researcher at CSIRO Mathematics & Information Sciences in Canberra, Australia from 2000 to 2001. He is a principal investigator of the project PROVEC ("Video Processing for Controlled Environments") of the Spanish R&D&I Plan started in 2007, and has led or contributed to a number of industry-sponsored projects, projects of the Spanish Science and Technology System (VISION, HESPERIA) and European FP projects (ACTIBIO, SCHEMA, ADVISOR). In particular, he coordinated UPC contribution to CHIL ("Computers in the Human Interaction Loop"), an IP of the IST/EU 6th Framework Program in the strategic objective of Multimodal Interfaces, involving video, audio and natural language technologies. He has authored or co-authored over 10 papers in international journals, 12 papers in LNCS, 50 contributions to conferences and nine book chapters and a teaching book in the areas of video coding, analysis, indexing and image processing.