Human Motion Capture using Data Fusion of Multiple Skeleton Data

Jean-Thomas Masse[1,2], Frédéric Lerasle[1,3], Michel Devy[1], André Monin[1], Olivier Lefebvre[2] and Stéphane Mas[2],

[1] CNRS, Laboratoire d'Analyse et d'Architecture des Systèmes
7 avenue du colonel Roche, F-31400 Toulouse, France.
{jean-thomas.masse, frederic.lerasle, michel.devy, andre.monin}@laas.fr
[2] Magellium SAS,
F-31520 Ramonville Saint-Agne, France.
{olivier.lefebvre, stephane.mas}@magellium.fr
[3] Université de Toulouse, UPS, LAAS
F-31400 Toulouse, France.

Abstract. Joint advent of affordable color and depth sensors and super-realtime skeleton detection, has produced a surge of research on Human Motion Capture. They provide a very important key to communication between Man and Machine. But the design was willing and closed-loop interaction, which allowed approximations and mandates a particular sensor setup. In this paper, we present a multiple sensor-based approach, designed to augment the robustness and precision of human joint positioning, based on delayed logic and filtering, of skeleton detected on each sensor.

Keywords: Human Posture Reconstruction, Motion Capture, Data Fusion, Delayed Logic, Kalman Filter, Kinect.

1 Introduction

Interest in Human Motion Capture (HMC for short, MoCap in general) is rising. Each video game entertainment systems companies introduced devices to provide this input. One of them, Microsoft Kinect, proved a commercial success [1], and introduced a reliable budget alternative to expansive time-of-flight cameras and texture-dependent stereo benches. The device is powered by PrimeSense's patented technology of structured light sensing [2]. The chip also powers the equivalent device Asus Xtion Pro Live [3] used in this paper.

While HMC is interesting for User Interaction in software design, it is also of paramount importance for robotics and surveillance, where it allows communication and interpretation of intents. In other fields there is also a surge of papers using the Kinect, for instance in education, sociology, and health.

Multiple SDKs exist, such as the Kinect-exclusive Microsoft Kinect SDK and the open-framework OpenNI [4]. PrimeSense developed the OpenNI Middleware NiTE [5] for OpenNI. It is what produces the user segmentation and skeleton. The algorithms exploit machine learning on foreground-segmented depth maps to detect the body parts. Since it only needs a depth map generator, we could have used any such maps from any sensor, such as a Kinect, or even a time-of-flight or stereo camera. That is why, in this paper, we will call *sensors* the Xtions we used to acquire our data.

However, this approach is mono-sensor, and prone to occlusion and perspective deformation. Literature focuses mainly on investigating the precision of the sensor's depth sensing [6] [7], or HMC from low-level data fusion [8]. But few, if any, tackle the problem of Motion Capture and skeleton precision, by fusing skeletons and comparing with a commercial MoCap system. That is why our goal is two-fold: (1) implement a high-level framework working atop monocular techniques and leveraging the multiple points of view; and (2) use a commercial Motion Capture system to get access to the ground truth necessary to measure our improvement over monocular.

The structure of the paper is as follows: first, we position ourselves against related works. Section 3 formalizes our approach. Before the results section, we describe the ground truth acquisition. The last section contains our conclusions and perspectives.

2 Related Work

NiTE/OpenNI, if similar to Microsoft Research Team's solution [9], [10], relies on the learning algorithm of Random Forests. It is very fast, and relatively accurate. Although it tries to cope up with self-occlusion, it is bound by monocular limitations. Also, filtering is very basic. It is only to smooth, and not predict.

Literature on Motion Capture by classical camera and computer vision techniques is very rich, as the survey from Moeslund et al. shows [11]. Better self-occultation resistance is beginning at three or four cameras [12]. These kinds of setup do require additional care to maintain data spatial and temporal coherence. Relying on color cues and silhouette [13], video MoCap is successful. However, it can not work in some circumstances such as absence of texture, where active sensors prevail [14].

HMC is the sensing of the physical quantity of the Human body joints position. As such, temporal reasoning is natural. There is a great amount of works in the video-based HMC literature [15]. In pedestrian tracking, tracking-by-detection approaches are more often combined with Delayed Logic [16], [17], [18]. To our best knowledge, such delayed logic principle has never been extended to depth sensor based HMC, where purely track-by-detection approaches [19] are privileged for the moment. Also, while multi depth sensor-based MoCap research exists [8], it focuses on low level sensor data fusion instead of high level skeleton reconstruction. Therefore, it constitutes natural and interesting to try to extend and rely on the OpenNI framework.

At any rate, ground truth is the base on which to measure results. Some work, such as HumanEva [20], described their eponym public multi-view video datasets for human motion estimation with ground truth. Few have investigated this since a large-scale, public RGB-D database with ground truth is yet to be found published.

Based on our own ground truth, our multi-sensor strategy leverages both: temporal information is exploited using Filtering techniques, and is fed at the same level than the skeleton measurements to a delayed-logic type of trajectory smoother that ensures the trajectory is consistent with observations and physical constraints.

3 Description of our Approach

In this section, we will first describe the formalism of our approach and then detail each step. An overview of the formalism-derived framework is located mid-way in subsection 3.3.

3.1 Formalism

As in much of current HMC software, the goal is to estimate $X = [X^j]_{j\in J}$, the Euclidian positions of a restricted but representative set J of joints from the human skeleton. They are: the hands, elbow, shoulders, feet, knees, hips, torso, neck and head. At each time step t, each sensor $k \in K$ produces a foreground segmentation bitmap S_t^k over the depth bitmap, and detect a skeleton measurement $Y_t^k = \left[Y_t^{j,k}\right]_{j\in J}$. We additionally choose not to limit ourselves to the measurements produced by NiTE so we consider instead $L \supset K$ the superset of hypotheses. For sets like J, K and L, we may use K instead of $|K|$ or $card(K)$ for ease of notations where applicable.

Each measurement is of the form $Y_t^l = X_t + V_t^l, \forall l \in L$, where V_t^l is the error of the skeleton reconstruction. This error is certainly neither white nor Gaussian.

Due to speed constraints, we chose to limit the search to those hypotheses, looking for the reconstruction with the smallest error. For increased stability, we also optimize over a time lapse of $M \in \mathbb{N}$ consecutive frames.

For ease of notation, we simply write Y_t^l the event $\hat{X}_t = Y_t^l$. The goal is then to find:

$$Y_{t-M:t}^* = \underset{\left\{Y_s^{l_s}\right\}_{s=t-M:t}}{\operatorname{argmax}} \mathbb{P}\left(\left\{Y_s^{l_s}\right\}_{s=t-M:t} \middle| \{S_{t-M:t}^k\}_{k\in K}\right) \tag{1}$$

This type of problem, called delayed logic, or Modal Trajectory Estimation, has already been solved by dynamic programming. Its formalism is described in the following subsection.

3.2 Delayed-logic

According to [21], the final position of the optimized trajectory can be found as:

$$Y_t^* = \underset{Y_t^{l_t}}{\operatorname{argmax}} \mathcal{J}_t^*\left(Y_t^{l_t}, \{S_{t-M:t}^k\}_{k\in K}\right) \tag{2}$$

where $\mathcal{J}_t^*$ is the marginal Bayesian likelihood:

$$\mathcal{J}_t^*\left(Y_t^{l_t}, \{S_{t-M:t}^k\}_{k\in K}\right) \triangleq \max_{Y_{t-M:t-1}} \mathbb{P}\left(\{S_{t-M:t}^k\}_{k\in K} \middle| \left\{Y_s^{l_s}\right\}_{s=t-M:t}\right) \mathbb{P}\left(\left\{Y_s^{l_s}\right\}_{s=t-M:t}\right) \tag{3}$$

Beginning with the knowledge of its initial value $\mathcal{J}_{t-M}^*\left(Y_{t-M}^{l_{t-M}}, \{S_{t-M}^k\}_{k\in K}\right) = \mathbb{P}\left(\{S_{t-M}^k\}_{k\in K} \middle| Y_{t-M}^{l_{t-M}}\right) \times \mathbb{P}\left(Y_{t-M}^{l_{t-M}}\right)$, it can be recursively computed by

$$\begin{aligned}&\mathcal{J}_t^*\left(Y_t^{l_t},\{S_{t-M:t}^k\}_{k\in K}\right)\\&=\max_{Y_{t-1}^{l_{t-1}}}\left(\mathbb{P}\left(\{S_t^k\}_{k\in K}\middle|Y_t^{l_t}\right)\times\mathbb{P}\left(Y_t^{l_t}\middle|Y_{t-1}^{l_{t-1}}\right)\times\mathcal{J}_{t-1}^*\left(Y_{t-}^{l_t},\{S_{t-M:t-1}^k\}_{k\in K}\right)\right)\end{aligned}\tag{4}$$

where $\mathbb{P}(\{S_t^k\}_{k\in K}|Y_t^{l_t})$ can be considered as an observation probability for which we have to find an observation model, and $\mathbb{P}(Y_t^{l_t}|Y_{t-1}^{l_{t-1}})$ a transition probability for which we have to find a dynamic model.

Finally, one implementation exploiting this finding is the Viterbi [22] algorithm.

Before detailing the rest of those, let us summarize the whole framework as it stands.

3.3 Overview

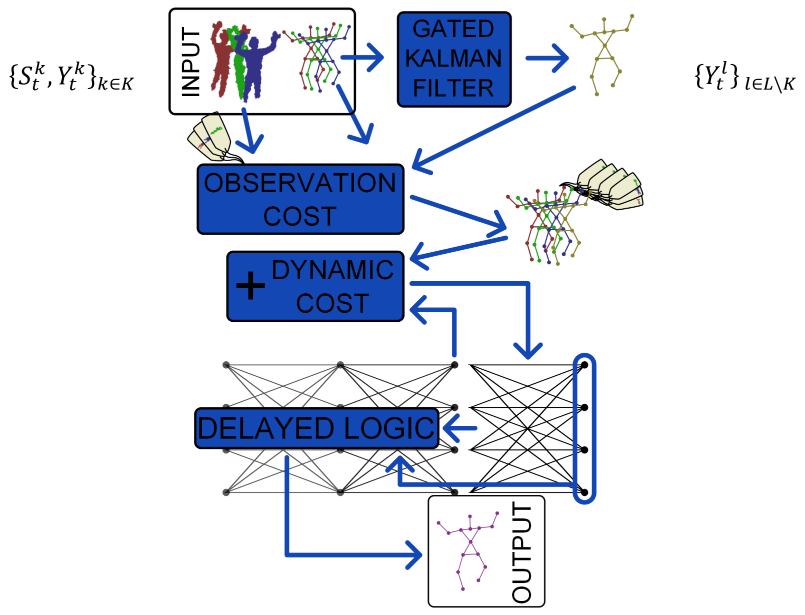

Fig. 1. Block-diagram of our approach.

An overview of the process is given by **Fig. 1**. The framework receives from each sensor $k\in K$, the user segmentations S_t^k and derived skeleton Y_t^k. We consider the segmentation to be reliable so we use it as an observation to measure the probability of a skeleton $\mathbb{P}(\{S_t^k\}_{k\in K}|Y_t^{l_t})$. The measured skeletons are fed to a Kalman Filter to introduce prediction in the process. All the skeletons $Y_t^l, l\in L\supset K$ are introduced at the end of the trellis of possible trajectories of length M. Transition probabilities $\mathbb{P}(Y_t^{l_t}|Y_{t-1}^{l_{t-1}})$, based on dynamics are combined to observation probability computed earlier. Finally, a Viterbi-like algorithm computes the most likely trajectory

$$\{\hat{X}_{t-M},\dots,\hat{X}_t\}=\underset{Y_{t-M}^{l_{t-M}},\dots,Y_t^{l_t}}{\operatorname{argmax}}\ p\left(Y_{t-M}^{l_{t-M}},\dots,Y_t^{l_t}\middle|S_{t-M}^K,\dots,S_t^K\right)$$

through the trellis. The whole process' estimation of the joints positions is $\hat{X}_t$.

We first wanted to complement the K skeleton hypothesis with dynamics-based randomly generated hypothesis, in a fashion similar to Markovian approaches. For instance, Particle Filters have already been applied successfully in similar situations [23], but was first discarded in favor of trying a dynamic programming approach, and second, out of concern for the heavy computation cost of the observation. Using a Kalman Filter to produce a supplementary skeleton hypothesis appeared to be an intermediate, cheaper and more sophisticated alternative.

In the following subsections, we describe the formalism and implementation of each step in more detail: the Observation and Dynamics Models, and the Gated Kalman Filter.

3.4 Observation Model

In order to assess likelihood of a skeleton estimate, the readily available data we could compare against was the foreground segmentation produced in the detection of the skeletons by NiTE, as illustrated in **Fig. 2**. As far as we could figure, the middleware seems to use a background substraction, as it is lost if the sensor is moved. New objects entering the field of view are labeled and kept independent afterward, and depth sensing is insensible to texture and color traps contrary to video segmentation.

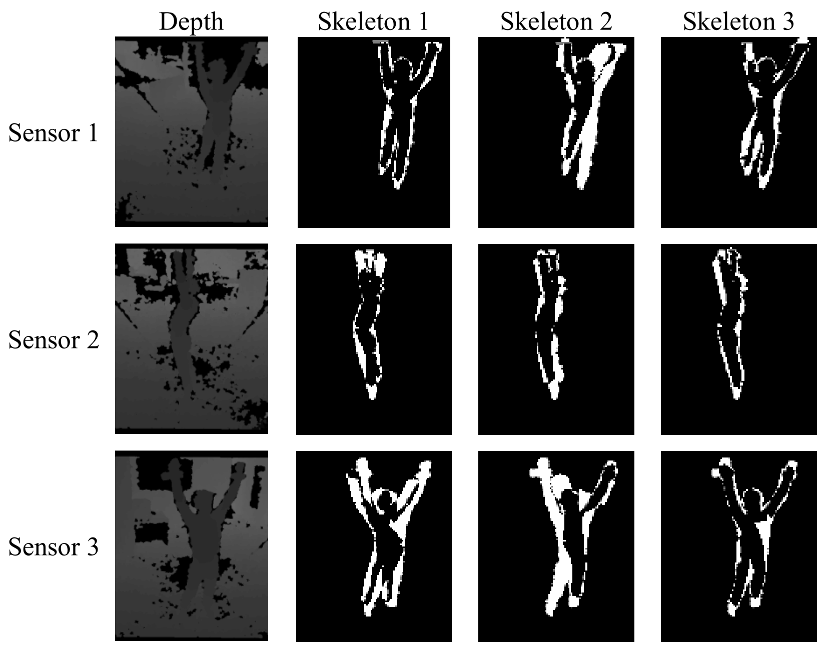

Fig. 2. Example of "exclusive or" of two segmentation images, taken from our dataset acquired with a system of 3 sensors fully presented in Section 4. A white pixel means only one segmentation encompassed this pixel. One can see that reprojected in another viewpoint, the seemingly correct skeleton (in diagonal frames, where the skeleton is generated from the same image) can produce varying results. Images are horizontally truncated here for presentation purposes.

As the user segmentation $S_t^k(i,j)$ is the foreground segmentation over i,j of the user seen from the sensor k, it can be compared to an artificial segmentation $\mathbb{S}_t^l(i,j)$ based on projecting a skeleton Y_t^l made of white cylinders on the image.

To compute a distance between foreground segmentation maps, we simply count the number of pixels exclusively in one or the other image, that is, the exclusive or, or sum of the absolute difference in each pixel $d_t(k,l) = \sum_{i,j}\left|S_t^k(i,j) - \mathbb{S}_t^l(i,j)\right|$.

As-is, d_t varies with the total coverage of the body on the image, that is, the distance of the user to the sensor. Therefore, we have to normalize. We do it over the total area of the body in the sensor image, giving a mostly distance-insensitive measure, if we ignore sampling effects of working with bitmaps. The result is:

$$\mathrm{D_t}(k,l) = \frac{d_t(k,l)}{\sum_{i,j} S_t^k(i,j)} \tag{5}$$

We then model the probability density $\mathbb{P}\left(\{S_t^k\}_{k\in K}\middle|Y_t^{l_t}\right)$ of having a correct observation, as the normal distribution around a D_t of an average $\mu = 0.1$ due to noise and approximating the limbs to cylinders and of standard deviation $\sigma = 0.1$, meaning

$$\mathbb{P}\left(\{S_t^k\}_{k\in K}\middle|Y_t^{l_t}\right) \triangleq \prod_{k\in K} \frac{1}{\sqrt{2\cdot\pi\cdot\sigma^2}} e^{-\frac{(\mathrm{D_t}(k,l)-\mu)^2}{2\cdot\sigma^2}} \tag{6}$$

It can be noted that the full size of 640×480 pixels has not been used in implementation. We use a smaller resolution of 160×120 to speed up calculation and remain super-realtime. Also, instead of maximizing the probability, the Viterbi algorithm is applied as-is, minimizing $\sum_{k\in K}\frac{(\mathrm{D_t}(k,l)-\mu)^2}{\sigma^2}$ instead for the same result, but faster computation, and no floating-point underflow. Indeed, with 45 degrees of freedom, the (especially dynamics) probability density sometimes valued below $2.22507e-308$, rounding to 0. We employed the same conversion for the Dynamics model, as it also uses a Gaussian probability model.

3.5 Dynamics model

Considering the Cartesian nature of our joint model, a correct dynamic model is difficult to obtain. We believe that using an articulatory model would make more sense but would be too long to compute, incompatible with our current speed objective.

For now, the Dynamics model goal is to remove jitter and promote temporal continuity. It is useful, for instance, if the skeleton suddenly flips left and right, as this is not possible for a human.

Therefore, we use a centered Gaussian density for each joint, with standard deviation attuned for each joint based on ground truth standard deviation at capture rate.

$$\mathbb{P}\left(Y_t^{l_t}\middle|Y_{t-1}^{l_{t-1}}\right) \triangleq \prod_{j\in J} \frac{1}{\sqrt{2\cdot\pi\cdot{\sigma_j}^2}} e^{-\frac{\left(Y_t^{j,l_t}-Y_{t-1}^{j,l_{t-1}}\right)^2}{2\cdot{\sigma_j}^2}} \tag{7}$$

3.6 Measurement-gated Kalman Filter

The Kalman Filter we employed uses a very simple skeleton model to maintain speed. Each j joint's state X^j is tracked separately as follows (units in mm, timestep is $20^{-1}s$):

$$X^j = (x \quad y \quad z \quad \dot{x} \quad \dot{y} \quad \dot{z})^T \quad (S)$$
$$X_t^j = A \cdot X_{t-1}^j + W_{t-1} \quad (E), \quad Z_t^j = H \cdot X_{t-1}^j + V_{t-1} \quad (O)$$
$$W \sim \mathcal{N}(0, Q) \quad and \quad V \sim \mathcal{N}(0, R) \quad (N)$$
$$H = (I_3 \quad 0_3) \quad (O_M), \quad A = \begin{pmatrix} I_3 & I_3 \\ 0_3 & 0.8 \cdot I_3 \end{pmatrix} \quad (E_M)$$
$$Q = \begin{pmatrix} 100^2 \cdot I_3 & 0 \\ 0 & 100^2 \cdot I_3 \end{pmatrix} \quad (N_{M1}), \quad R = 200^2 \cdot I_3 \quad (N_{M2})$$

For ease of discourse and establishing variable names, we remind the main equations of processing a Kalman Filter:

$$\begin{aligned} \hat{X}_{t+1}^{j-} &= A \cdot \hat{X}_t^j \\ P_{t+1}^{j-} &= A \cdot P_t^j \cdot A^{-1} + Q \end{aligned} \tag{8.1}$$

$$\begin{aligned} S_{t+1} &= H \cdot P_{t+1}^{j-} \cdot H^T + R \\ \boldsymbol{r}_{t+1}^j &= Y_{t+1}^j - H \cdot \hat{X}_{t+1}^{j-} \end{aligned} \tag{8.2}$$

$$\begin{aligned} K_{t+1} &= P_{t+1}^{j-} \cdot H^T \cdot S_{t+1}^{-1} \\ \hat{X}_{t+1}^j &= \hat{X}_{t+1}^{j-} + K_{t+1} \cdot \boldsymbol{r}_{t+1}^j \\ P_{t+1}^j &= P_{t+1}^{j-} - K_{t+1} \cdot H \cdot P_{t+1}^{j-} \end{aligned} \tag{8.3}$$

$$\left(\hat{X}_{t+1}^{j-}, P_{t+1}^{j-}\right) = \left(\hat{X}_{t+1}^{j}, P_{t+1}^{j}\right) \tag{8.1'}$$

Step (8.1) gets done once at every time step. (8.2) is used to test the Measurement Gate explained below. If the measurement passes the gate (i.e. it is to be processed) then step (8.3) is done. For each additional measurement, step (8.1') is done before resuming at step (8.2). If no measurement is fitting through the gate, then the reverse assignation from (8.1') is made (the estimate is only the prediction).

This model has but one drawback, non-compliance to an articulated body. But in practice, since the measurements are from an articulatory-body compliant system, the results are approximately so. However, no conversion is needed from input format and parameters are straightforward.

Unfortunately, all the joints do not follow such a model even remotely. For instance, if the user is sideways, both the feet may be measured at the same place. To prevent such measure from being integrated (which heavily degrades the performance), a measurement gate (or residual monitoring) is implemented.

This technique first published in [24] consists in rejecting Y_t^j from processing if $(\boldsymbol{r}_t^j)^T \cdot S_t^{-1} \cdot \boldsymbol{r}_t^j > g_t^2$, where $\boldsymbol{r}_t^j$ is the innovation, S_t the residual covariance matrix, and g_t is the gating threshold. g_t is generally set between 1 and 3, expressing "innovation's L2-norm is beyond g_t residual sigmas." This generally indicates a faulty sensor; as such phenomenon has 32%, 5% or 0.2% probability of happening for g_t=1, 2 or 3. Since we have redundancy in measurements, a single correct measurement is sufficient. If no measurements are taken into account, the Error variance P_t^j eventually gets big enough to allow the measurement through the gate (as S_t depends on P_t^j, and also repeatedly gets incremented by R).

4 Our Multi-Sensor Platform Setup and Data Acquisition

Our objective is to appreciate the precision in the one-sensor and our multi-sensor system's estimation of the human body joints' positions. Therefore, we decided to use the Motion Capture system from Motion Analysis [25].

This acquisition was meant to serve three purposes:

- First, evaluate the performance of NiTE from several angles: frontal (as it was designed), and also sideways and behind;
- Second, create a dataset to exploit in a learning algorithm;
- Third, create a dataset to test fusion and other learning algorithms on.

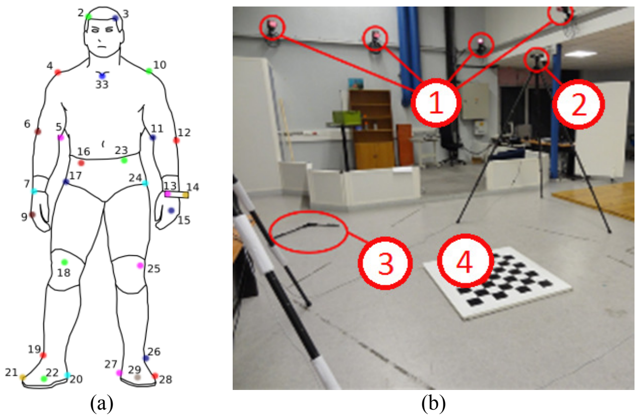

Fig. 3. Precision measurement experiment. (a) Motion Capture markers setup [25], and (b) Motion Capture (① are 4 of the 10 IR or NIR cameras) and Xtion (②) setup, with MoCap reference frame (③) and 1 m² camera calibration chessboard (④) also present.

We placed and oriented the MoCap reference frame on the RGB calibration chessboard, respectively ③ and ④ in **Fig. 3**(b), in order to localize all the sensors in MoCap reference frame. Localizing the device's RGB sensor was sufficient because the Depth-to-RGB registration is factory-determined and readily available in and done by OpenNI. Works show that significant improvement can be made, but only in the close range of the sensor [26].

Time synchronization was subsequently achieved by MoCap marker projection (**Fig. 3**(a)) in RGB frames. On such a frame, a fast marker would leave a blurry trail. Since the MoCap frames were 10 times as numerous, we fine-tuned the time synchronization so as to project the marker on the middle of the blur. Then, all the depth streams were synchronized to MoCap since the frames of each were all time-stamped.

We also considered sequences short enough (90 seconds on average) for time drift to be insignificant compared to the end result precision.

Acquired data consist of two sets, labeled IRSS35, with a full set of MoCap markers (35), allowing for skeleton building; and NS-CAP13, with a reduced set of markers (13), for quick testing. Both feature several sequences covering exercise and sport moves, and regular moves and poses from regular life, and extremely varying random poses for completion. IRSS35 has nine sequences, sometimes rerecorded, totaling 16 minutes or upward of 21569 workable depth frames.

It must be noted that each sequence is not a simple movement, but a mix and match of actions such as weight lifting, running, squats, as well as more mundane movement such as dancing, broom handling (broom not detected), and moving furniture.

5 Evaluation and Results

The presented results of our approach are achieved real-time at 20Hz with a mono-thread CPU application on an Intel Core i7 2760QM (2.40 Ghz). As it was offline filtering, the PNG decompression of the user images occupied 80% of the computations. This means NiTE may run its 3 own threads on the same CPU core and the framework will still perform the same online.

Fig. 4 shows the proportion of skeleton frames whose joints were closer than a distance from ground truth (50, 100, 150 and 200 mm. Those are cumulative). Each skeleton proposed by the OpenNI single-sensor detector has an entry (*Sensor 1* to *Sensor 3*) for comparison. Additionally, we made the Oracle *Sensor Best5* pseudo-result whose output is the skeleton of the *Sensor* which had at that time the lowest cumulated distance to the reference positions of the joints.

The three tested setups were:

- *Our approach*, based on Viterbi algorithm fed with the three skeletons and first-order gated Kalman filter:
- A single, 0-order (no speed in state) ungated *Kalman*, for reference;
- The described implementation with no additional skeleton provided other than the single sensors'. It is the "*Delayed Logic Only*" entry.

The figures allow us to come to three conclusions.

First, a Viterbi-type implementation, without any additional skeleton, then acting as a sensor selector, can closely match the expected result of being precise based on joints.

Second and third, adding additional skeletons choices brings two advantages: slightly increased reliability, as fewer frames are outside the outstanding distance of 200 mm from ground truth; and much increased precision, as the amount of positions lower than the 50 mm mark increases.

Besides, *Sensor 3* was primarily faced during the experiment (except during rotation movements). The statistics indicate that while this Kinect will be the most reliable over all joints, it is not necessarily the most precise everywhere. Indeed, the other two sensors have better statistics over one side of the body.

Effect differs on the type of joint: from the always-visible such as the head, the increase in precision is higher, while the increase in reliability is better seen at occluded joints, such as the feet. Hands exhibit both since they are alternatively occluded or visible during manipulations.

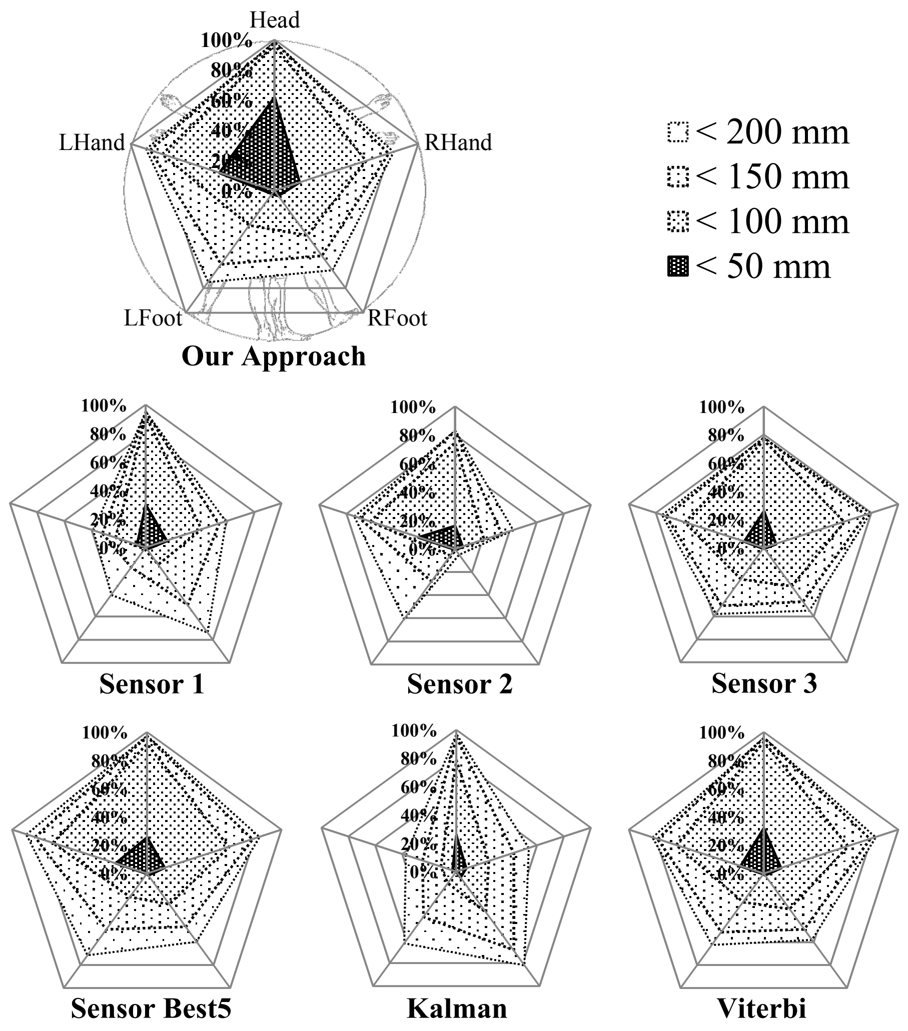

Fig. 4. Results.

6 Conclusion and Perspectives

In this paper, we described a complex framework applying concepts from Dynamic Programming, Filtering and Vision. The result was positively compared to other classic and "perfect" algorithms using ground truth acquired from a commercial Motion Capture System. The framework original contribution runs in super-realtime fashion, that is, achieving real-time speed with a budget of a fraction of a CPU core. The result is a single, more accurate skeleton from the results of separate monocular skeleton reconstructions.

The results were attained using only simple techniques and making numerous simplifications, such as working only in Cartesian space, using only a single prediction model, and using 2D reprojection. We would like to improve upon these results by going over these assumptions, trying several models in parallel, adaptive models, going into articulatory space, or using a voxel-based observation. Also, we consider leveraging the power of GPU for the latter part.

Acknowledgements
The research leading to these results has received funding from the European Community's Seventh Framework Programme FP7/2007-2013 FoF.NMP.2011-2 - under grant agreement No 285380 - PRACE.

References

1. Alfonso, D.: Microsoft Investor Relations - Press Release. (Accessed January 27, 2011) Available at: http://www.microsoft.com/investor/EarningsAndFinancials/Earnings/PressReleaseAndWebcast/fy11/q2/default.aspx
2. Freedman, B., Shpunt, A., Arieli, Y.: Distance-Varying Illumination and Imaging Techniques for Depth Mapping. United States Patent US 20100290698 A1 , 18 November 2010
3. ASUS: ASUS Xtion PRO LIVE. Available at: http://www.asus.com/Multimedia/Xtion_PRO_LIVE/
4. WordPress: OpenNI | The standard framework for 3D sensing. Available at: http://www.openni.org/
5. PrimeSense: NiTE Middleware - PrimeSense. Available at: http://www.primesense.com/solutions/nite-middleware/
6. Binney, D., Boehm, J.: Performance Evaluation of the PrimeSense IR Projected Pattern Depth Sensor. University College London, London, United Kingdom (September 2011)
7. Andersen, M., Jensen, T., Lisouski, P., Mortensen, A., Hansen, M., Gregsersen, T., Ahrendt, P.: Kinect Depth Sensor Evaluation for Computer Vision Applications., Aarhus (2012)
8. Zhang, L., Sturm, J., Cremers, D., Lee, D.: Real-Time Human Motion Tracking using Multiple Depth Cameras. Proc. of the International Conference on Intelligent Robot Systems (IROS) (2012)
9. Shotton, J., Fitzgibbon, A., Cook, M., Sharp, T., Finocchio, M., Moore, R., Kipman, A., Blake, A.: Real-time human pose recognition in parts from single depth images. Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on, 1297 -1304 (2011)
10. Taylor, J., Shotton, J., Sharp, T., Fitzgibbon, A.: The Vitruvian manifold: Inferring dense correspondences for one-shot human pose estimation. In : IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp.103-110 (2012)

11. Moeslund, T., Hilton, A., Krüger, V.: A survey of advances in vision-based human motion capture and analysis. Computer Vision and Image Understanding 104(2-3), 90 - 126 (2006)
12. Deutscher, J., Reid, I.: Articulated Body Motion Capture by Stochastic Search. International Journal of Computer Vision 61(2), 185-205 (2005)
13. Hofmann, M., Gavrila, D.: Multi-view 3D Human Pose Estimation combining Single-frame Recovery, Temporal Integration and Model Adaptation. In : Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on, pp.2214-2221 (2009)
14. Abramov, A., Pauwels, K., Papon, J., Worgotter, F., Dellen, B.: Depth-supported real-time video segmentation with the Kinect. In : Applications of Computer Vision (WACV), 2012 IEEE Workshop on, pp.457-464 (2012)
15. Forsyth, D., Arikan, O., Ikemoto, L., O'Brien, J., Ramanan, D.: Computational Studies of Human Motion: Part 1, Tracking and Motion Synthesis. In : Foundations and Trends in Computer Graphics and Vision (2006)
16. Wojek, C., Walk, S., Schiele, B.: Multi-cue onboard pedestrian detection. In : Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on, Miami, FL, pp.794-801 (2009)
17. Benfold, B., Reid, I.: Stable Multi-Target Tracking in Real-Time Surveillance Video. In : CVPR, pp.3457-3464 (2011)
18. Berclaz, J., Fleuret, F., Fua, P.: Robust People Tracking with Global Trajectory Optimization. In Society, I., ed. : Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, vol. 1, pp.744-750 (2006)
19. Andriluka, M., Roth, S., Schiele, B.: People-Tracking-by-Detection and People-Detection-by-Tracking. In : Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on, Anchorage (2008)
20. Sigal, L., Balan, A., Black, M.: HumanEva: Synchronized Video and Motion Capture Dataset for Evaluation of Articulated Human Motion. International Journal of Computer Vision 87(1-2), 4-27 (March 2010)
21. Larson, R. E., Peschon, J.: A dynamic programming approach to trajectory estimation. Automatic Control, IEEE Transactions on 11(3), 537-540 (1966)
22. Viterbi, A. J.: A personal history of the Viterbi algorithm. Signal Processing Magazine, IEEE 23(4), 120-142 (2006)
23. Mekonnen, A. A., Lerasle, F., Herbulot, A.: Cooperative passers-by tracking with a mobile robot and external cameras. Computer Vision and Image Understanding(0), - (2012)
24. Maybeck, P.: Stochastic models, estimation, and control. Academic Press (1979)
25. Maloney, R.: Movement Analysis Products. (Accessed January 4, 2013) Available at: http://www.motionanalysis.com/html/movement/products.html
26. Herrera C., D., Kannala, J., Heikkila, J.: Joint Depth and Color Camera Calibration with Distortion Correction. IEEE Transactions on Pattern Analysis and Machine Intelligence 34(10) (2012)