Model based Full Body Human Motion Reconstruction from Video Data

Hashim Yasin
Department of Computer Science II
University of Bonn
Bonn, Germany
yasin@cs.uni-bonn.de

Björn Krüger
Department of Computer Science II
University of Bonn
Bonn, Germany
kruegerb@cs.uni-bonn.de

Andreas Weber
Department of Computer Science II
University of Bonn
Bonn, Germany
weber@cs.uni-bonn.de

ABSTRACT

This paper introduces a novel framework for full body human motion reconstruction from 2D video data using a motion capture database as knowledge base containing information on how people move. By extracting suitable two-dimensional features from both, the input video sequence and the motion capture database, we are able to employ an efficient retrieval technique to run a data-driven optimization. Only little preprocessing is needed by our method, the reconstruction process runs close to real time. We evaluate the proposed techniques on synthetic two-dimensional input data obtained from motion capture data and on real video data.

Categories and Subject Descriptors

I.3.7 [**Computer Graphics**]: Three-Dimensional Graphics and Realism—*animation*; H.3 [**Information Storage and Retrieval**]: Information Search and Retrieval

Keywords

Motion reconstruction, motion retrieval, data-driven optimization

1. INTRODUCTION

Human motion reconstruction and analysis from video data is a current field of research in the scope of computer vision, computer animation and computer graphics. Over last few decades, an increasing interest has been appeared in the areas of human motion understanding, reconstruction and analysis. Thus, the demand of high quality motion capture is increasing and new applications for everywhere motion capture based on various consumer electronic devices are emerging.

On one hand, marker-based optical motion capturing has become a standard technique to record human motions for movie industry, computer games, sports and medical sciences. Therefore, there is a growing pool of high-quality motion capture data which can be used for scientific studies [10, 2, 5]. On the other hand, the reconstruction of human motion from a single video stream is still a current strand of research.

In this paper, we propose a method to reconstruct full body human motion on the basis of a video stream and pre-existing knowledge available in a motion capture (MoCap) database. Our work is inspired by the work of Chai and Hodgins [3] where human motions are reconstructed on the basis of few optical motion capture markers and the work of Tautges et al. [17] where the control signal is replaced by only four accelerometers. We adapt their techniques to work with even more sparse input signals. We only use two dimensional information of five specific joints to reconstruct human full body motion sequences.

To access relevant information from the database, a kd-tree based retrieval technique is employed. Here, in this paper, two-dimensional feature sets are derived from three-dimensional motion capture data at different viewing directions and are compared against two-dimensional feature sets obtained from the input video. One strength of our approach is that only the positions of hands, feet and the head are needed to search the database. These features are detected and tracked from input video with standard feature detection techniques like Maximally Stable Extremal Regions (MSER) and Speeded Up Robust Features (SURF). Based on the information retrieved from the database and the control signal obtained from the video, three dimensional poses can be reconstructed by solving an energy minimization problem.

2. RELATED WORK

Chai and Hodgins [3] classify motion reconstruction research into three types; constructing human motion model, reconstruction with utilization of motion graph and motion interpolation. They utilize the neighborhood graph to find similar poses and motion interpolation for motion reconstruction. Park et al. [12] describe a novel method for human motion reconstruction from the inter-frame feature correspondence in case of video streaming by employing some motion capture library. They reconstruct the human motion using time-warping, joint orientations and root trajectories. Rocha et al. in [14] present the motion reconstruction using the invariant moments with a set of ellipses and matching is

Mirage '13 June 06 - 07 2013 Berlin, Germany

performed on the basis of these ellipses. They exercise the *bsp*-tree as data structure. Wu et al. in [19] describe the combination of adaptive cluster method and sparse approximation in order to extract the character pose from a large motion database. Krüger et al. [8] elaborate pose-by-pose matching and then global matching for motion using the lazy neighborhood graph for similarity search. The authors compare feature sets of different dimensions and found that a 15-dimensional feature set can describe human poses accurately. We take into account this significant fact and build up our system on the basis of similar feature sets. Tautges et al. [17] enhance the Chai and Hodgins [3] technique and reconstruct human motion using the sparse accelerometer data with the help of online lazy neighborhood graph.

3D pose retrieval from 2D video streaming is ill-posed problem and has been tried to resolve by using some prior knowledge. Chen and Chai in [4] reconstruct the 3D human motion as well as skeleton model from uncalibrated monocular video by nonlinear optimization technique with the help of generative models. They solve the motion, skeleton and camera parameters with gradient based optimization. They employ a small set of 2D image features tracked from a monocular video sequence in order to reconstruct the 3D motion. Wei and Chai [18] model human motion from monocular video using full perspective model. First, they estimate camera parameters, human skeleton size and 3D pose; calculate in-between poses from 2D images; and then interpolate them in reconstruction process. Baak et al. in [1] develop the real time 3D full body pose estimation from 2.5D depth image with the help of some pose database. Hornung in [6] reconstruct the 3D model from the single uncalibrated video sequence and presents the synchronized multi-view setting by employing the actor's pose synchronization. Park and Sheikh [11] make the 3D articulated trajectory reconstruction from the collection of 2D image sequences by taking 2D projections of trajectory, 3D trajectory pose and captured camera time as prior information. In [16] the novel method for 2D-3D matching problem has been described using the *k*d-tree based approach and 3D points are represented by taking mean of SIFT (Scale-Invariant Feature Transform) feature descriptors. Visual dictionary of the visual words is developed, and from this dictionary they have searched for correspondence of the 2D-3D points. Ramakrishna et al. present in [13] an activity-independent approach for 3D pose reconstruction from 2D positions of anatomical landmarks in an image. They estimate the weak perspective camera parameters by considering it as Orthogonal Procrustes problem. Roodsarabi and Behrad [15] describe 3D human motion reconstruction by employing Taylor method. They utilize Discrete Cosine Transform (DCT) as descriptors in the process of matching. Jain et al. [7] develop three level 3D proxies (single point proxy, the cylindrical shape model, and the joint hierarchical model) from 2D hand drawn characters with the help of user input hand annotation.

3. OVERVIEW

The first step towards the full body human motion reconstruction is the selection and extraction of feature sets which are not only of low-dimensions, but can also represent the high dimensional motion without losing significant information. 3D positional information of the hands, feet and head are used in order to extract 15-dimensional feature sets. These feature sets are projected into 10-dimensional feature sets containing two-dimensional points at different elevation and azimuth angles, using an orthographic transformation. The extracted 10-dimensional feature sets are used further in order to build a spatial data structure, in our case we use a *k*d-tree .

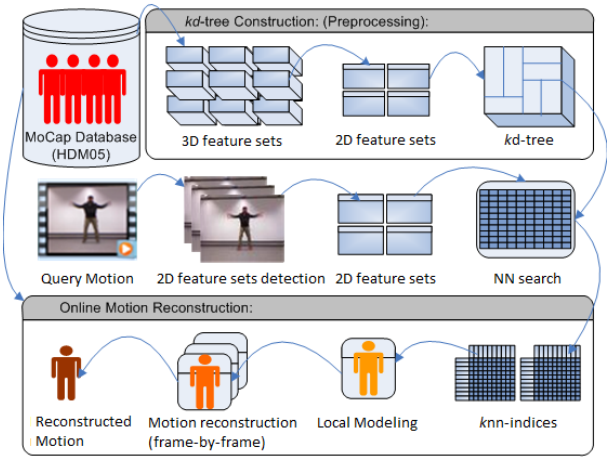


Figure 1: System overview diagram

As query motion sequences we consider two scenarios in this paper: First, synthetic examples, where the feature sets are computed from an input motion capture sequence from predefined viewing directions. Second, video motion clips, where the recorded motion has to be reconstructed. In this case, the relevant two dimensional feature sets have to be detected and tracked before they can be used as query for the similarity search.

With the feature sets, extracted from the query motion sequences, a *k*-nearest-neighbor (*k*nn) search is performed. These nearest neighbors are used as prior information to synthesize poses, that are known from the database and are close to the input signal. The motion synthesis or on line motion reconstruction is implemented as energy minimization, considering different energy units like control unit, prior knowledge unit, smoothness unit and pose adjustment unit. The overall system flow has been expressed in Figure 1.

The remainder of this work is organized as follows: In the following sections 4 and 5, we describe the details of the steps mentioned above. In section 6, we present the results of our evaluations and we conclude this work in section 7.

4. MOTION RETRIEVAL

For our data-driven motion reconstruction scheme, we have to search the motion capture database for motion sequences that are similar to the input motion. To this end, we adapt the motion retrieval technique from Krüger et al. [8] to work with feature sets based on two dimensional input data. The authors conclude in their work, that the feature set $\mathcal{F}_{\mathrm{E}}^{15}$ is the one for choice especially for real-time applications. Thus, we develop feature sets $\mathcal{F}_{\mathrm{2D}}^{10}$ which is derived from feature sets $\mathcal{F}_{\mathrm{E}}^{15}$ and feature sets $\mathcal{F}_{\mathrm{video}}^{10}$obtained from video data for our scenarios.

4.1 Feature Set Extraction from MoCap-Data

To compute the feature sets $\mathcal{F}_{\mathrm{2D}}^{10}$, first step is the extraction of the feature sets $\mathcal{F}_{\mathrm{E}}^{15}$ along the lines of Krüger

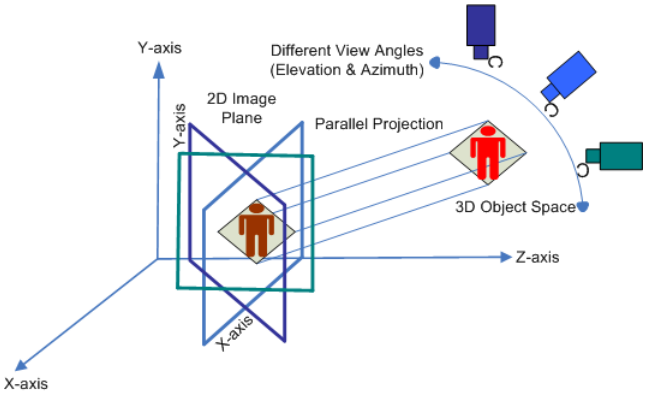


Figure 2: 2D feature set extraction model

et al. [8] for all frames of the motion capture data. This feature sets include the three dimensional positions of the hands, feet and the head in a normalized pose space. As normalized pose space, we consider the joints' positions in the root nodes coordinate system. In this representation, we discard information of the orientation and position in the global system—poses might be similar independent from the actual place where they are performed at.

The second step is the projection of the points included in $\mathcal{F}_{\mathrm{E}}^{15}$ to a plane, that is parameterized with elevation and azimuth angles. Similar to the pinhole camera model, we make use of an orthographic projection and ignore all intrinsic camera parameters as sketched in Figure 2. As a result from this projection step, we obtain temporary feature sets depending on the viewing directions that are specified by the angles the plane is parameterized with.

Finally, in the third step, the feature sets $\mathcal{F}_{\mathrm{2D}}^{10}$ are computed by an additional normalization step. We translate the two dimensional feature points to have their center of mass in the origin of the 2D coordinate system. This step is needed to get the feature sets comparable to the later described feature sets $\mathcal{F}_{\mathrm{video}}^{10}$ from video data where no articulated skeleton exists. An illustration of multiple poses under various viewing directions and the resulting feature sets $\mathcal{F}_{\mathrm{E}}^{15}$ and $\mathcal{F}_{\mathrm{2D}}^{10}$ is given in Figure 3.

4.2 Feature Set Extraction from Video Data

In order to retrieve poses based on video data, we developed a feature sets $\mathcal{F}_{\mathrm{video}}^{10}$ that are comparable to the feature sets $\mathcal{F}_{\mathrm{2D}}^{10}$ extracted from motion capture data.

Camera Parameter Estimation.

We have recorded our video sequences for input query using a Kinect RGB camera and have used Kinect 3D skeleton information of a first couple of frames for camera calibration only. In the process of calibration, the variables involved are categorized into *intrinsic camera parameters* and *extrinsic camera parameters*. In case of kd-tree construction, we only consider the extrinsic camera parameter as mentioned in Subsection 4.1, while in case of video data as query input, we need *intrinsic* as well as *extrinsic* camera parameters. The transformation between 3D feature sets $[X_w Y_w Z_w 1]^T \in \mathbb{R}^4$ and 2D image feature sets $[u_i v_i 1]^T \in \mathbb{R}^3$ in homogeneous coordinate system has been done by the projective Equation 1

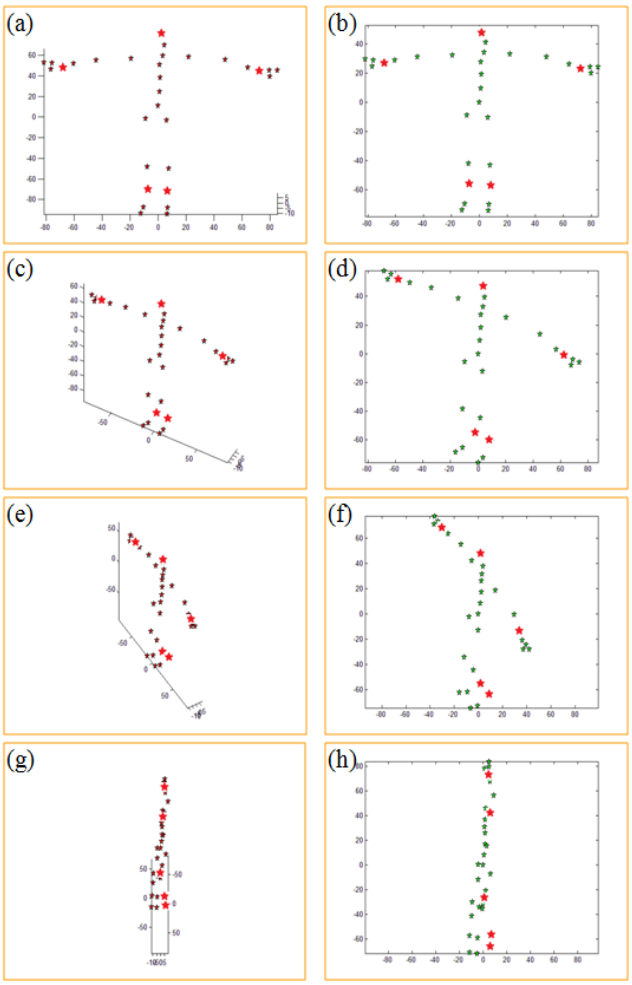


Figure 3: 3D and 2D feature sets when elevation angle is fixed to 45 degree and azimuth angle are 0 degree for(a)-(b), 30 degree for (c)-(d), 60 degree for (e)-(f) and 90 degree for (g)-(h).

given as follows;

$$\begin{bmatrix} u_i \\ v_i \\ 1 \end{bmatrix} = C_m \left[R_{(\alpha,\beta,\gamma)} \mid t_{x,y,z} \right] \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix} \tag{1}$$

Where $\left[R_{(\alpha,\beta,\gamma)} \mid t_{x,y,z}\right]$, expressed as *extrinsic camera parameters*, involves the 3 rotational parameters (α, β and γ) and 3 translational parameters (t_x, t_y and t_z) and C_m is the *camera matrix* which represents the *intrinsic* or *internal* camera parameters and is explained in Equation 2.

$$C_m = \begin{bmatrix} s_x & \mu & i_x \\ 0 & s_y & i_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} f_x & 0 & 0 \\ 0 & f_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2}$$

The notations f_x and f_y are the focal lengths in pixel size units, μ is the skew coefficient between x-axis and y-axis and its value is set to be zero, s_x and s_y are the scaling factors in x and y-directions respectively, i_x and i_y are the principal points which are ideally considered as image center. In this paper, we are dealing with the single static camera and the

performing actor perform actions at his place, so we only consider the *intrinsic camera parameters* and need not to find out the *extrinsic camera parameters*. In this way, only the focal lengths (f_x and f_y) and scaling factors (s_x and s_y) are the unknown parameters which can be computed by the already known 2D and 3D information of a first few frames.

Feature Detection and Tracking.

A video .avi file is given as input and the first task towards the feature tracking is the detection of the features of the hands, feet and the head. For that purpose, MSER, colorMSER and SURF feature detection techniques have been utilized. At start, the positions of the hands, feet and head in first frame are annotated manually and draw boxes around the hands, feet and head. 2D image features are detected and extracted by using MSER, colorMSER and SURF techniques. The extracted features are tracked in next frames by matching them with the already extracted features of the previous frames. In case the features are not matched with previously detected features, the box moves around (left, right, up and down) to find new features until features are matched with previously extracted features. After getting the features matched, the box shifts to the new position and updates its position. This process is carried out for all frames in the video. Like bags of words model, a dictionary of features (DOF) has been developed which maintains all the extracted features of the previous frames. The detected features of the all new frames are added into this dictionary of features. In this way, DOF has complete record of features of the hands, feet and head at different positions and orientations and we can deal properly with the problem of matching in case of different positions and orientations of hands, feet and head in different frames. Otherwise, in case of mistracking, positions of boxes are corrected manually.

Normalization Step.

Normalization step is necessary here in order to match the 2D image coordinate system of the feature sets extracted from the video data with the 2D motion capture coordinate system of the feature sets extracted from the motion capture database. We consider center of the mass as origin of 2D coordinate system and translate 2D image features by computing mean value as described earlier in Subsection 4.1.

4.3 Nearest Neighbors Search

With the previously described feature sets at hand, we are now able to search for similar poses in the database.

We want to make no assumptions about the direction at which our input motion sequences are recorded during the reconstruction process. For this reason we sample the whole database from different viewing directions and obtain multiple feature sets $\mathcal{F}_{\text{2D}}^{10}$ for each pose stored in the database. Based on all these feature sets, we construct a kd-tree that is used later for k-nearest neighbor search.

Depending on the considered scenario, we extract the feature sets $\mathcal{F}_{\text{2D}}^{10}$ and $\mathcal{F}_{\text{video}}^{10}$ for the input sequences respectively and search for the k-nearest-neighbors for every single frame. Due to the sampling of the database from different directions, the same frame of the database might be included to the neighborhood of a query frame multiple times. This doesn't mean a disadvantage—these frames contribute stronger in the later reconstruction process. If one wants to avoid such a stronger influence on the result, duplicates can be easily removed from the neighborhood. In our experiments, we have not found this additional step necessary. In the result section, we report on some experiments, concerning the parameters (size of k and sampling of the database) for the $k$nn-search. We use ANN (Approximate Nearest Neighbor searching) C++ library [9] in order to search for nearest neighbors.

The time complexity for k-nearest neighbor search using kd-tree is represented as $O(km\log(p \times n))$, where k is the fixed value for k-nearest neighbors, n is the size (total number of frames) of the query, p is the number of 2D projections and m is the size (total number of frames) of the database.

5. ONLINE MOTION RECONSTRUCTION

In this section, we describe in detail how the resulting motion sequences are synthesized. The motion is reconstructed frame by frame by computing joint angle configurations $Q = \{\vec{q}_t, \ldots, \vec{q}_T\}$ for all frames of input signal. The goal is to reconstruct the human motion as close as possible to the original motion independently the used two-dimensional input, and to have the motion similar to the examples stored in the motion capture database. We formulate the process of reconstruction as energy minimization problem where different units in the optimization ensure that the result fits the sometimes contradictory requirements. The optimization process itself is implemented using the gradient decent method. The process of optimization for reconstruction is the bottleneck in the performance of the system.

5.1 Local Model for Pose Synthesis

According to Chai and Hodgins [3], low dimensional local models are adequate in order to develop the global model of high dimension. The key idea behind the local model is to synthesize and reconstruct human motion pose by pose. The poses' information are accumulated for all frames of 2D input query in order to reconstruct the complete human motion. The local model is based on mean vector $\hat{\mathcal{M}}_t$ of k_q-examples (the joint angle configuration of k-nearest-neighbors obtained from database) at current frame t, principal component coefficients Ω_t of k_q-examples and low dimensional vector Υ_t of current synthesized pose $\hat{\mathcal{P}}_t^T$. Principal component coefficients are the eigenvectors relevant to largest eigenvalues of covariance matrix of k_q-examples and are calculated with the help of Singular Value Decomposition (SVD).

$$\hat{\mathcal{P}}_t^T = \Omega_t \Upsilon_t + \hat{\mathcal{M}}_t \tag{3}$$

5.2 Energy Minimization Function

We formulate the energy minimization function in the same direction as Chai and Hodgins have done in [3]. We optimize pose by pose reconstruction by using a set of four energy units; *control unit*, *prior knowledge unit*, *smoothness unit* and *pose adjustment unit*. These energy units are combined to generate the energy minimization function for motion synthesis,

$$E_{\text{rec}} = argmin[w_c E_c + w_{pk} E_{pk} + w_s E_s + w_{pa} E_{pa}] \tag{4}$$

Where, the terms w_c, w_{pk}, w_s and w_{pa} are the weights for control unit, prior knowledge unit, smoothness unit and pose adjustment unit respectively. These weights are considered as user defined constants. Moreover, each energy unit is

normalized with normalization factor N_t at frame t which represents the number of elements in the energy unit as described in energy Equations 5 to 8.

Control Unit.

Control unit computes the distance or deviation between 2D projections of *reconstructed pose* $P_t^{r,2D}$ and 2D feature sets of *estimated pose* $P_t^{e,2D}$ at current frame t. The reconstructed pose is the normalized 2D projected locations of the current pose obtained from synthesize pose $\hat{\mathcal{P}}_t^r$ of the local model after the process of forward kinematics. Whereas the estimated pose is the 2D feature sets obtained from the query motion directly. Mathematically,

$$E_c = [\frac{1}{\sqrt{N_t}}(P_t^{r,2D} - P_t^{e,2D})] \tag{5}$$

Prior Knowledge Unit.

This unit compels the system to produce acceptable results according to database, with the help of prior knowledge. It measures a-priori likelihood of the current synthesized pose into the knowledge base developed from motion capture database and restricts the results according to the pre-existing knowledge in database. The prior knowledge unit is calculated by employing Mahalanobis distance as,

$$E_{pk} = \| \frac{1}{\sqrt{N_t}}(\hat{\mathcal{P}}_t^r - \hat{\mathcal{M}}_t)^T C^{-1} (\hat{\mathcal{P}}_t^r - \hat{\mathcal{M}}_t) \|^2 \tag{6}$$

Where $\hat{\mathcal{P}}_t^r$ is the synthesized pose, $\hat{\mathcal{M}}_t$ is the mean vector of k_q-examples at frame t and $(\hat{\mathcal{P}}_t^r - \hat{\mathcal{M}}_t)^T$ is the transpose of the difference between them. The term C^{-1} is the inverse of the covariance matrix which is calculated with the help of SVD as mentioned earlier in Subsection 5.1.

Smoothness Unit.

Smoothness unit is necessary in order to impose smoothness on reconstructed pose otherwise high frequency jittering and jerkiness effects may arise. To avoid these jerkiness effects, previously two or three reconstructed poses can be utilized in a way that newly reconstructed pose have an impact from the already reconstructed poses,

$$E_s = [\frac{1}{\sqrt{N_t}}(P_t^r - 2P_{t-1}^r + P_{t-2}^r)] \tag{7}$$

Where P_t^r, P_{t-1}^r and P_{t-2}^r are the reconstructed poses at frames t, $t-1$ and $t-2$ respectively.

Pose Adjustment Unit.

This unit is entertained only when the video signal is given as input query. It minimizes the distances between the 3D reconstructed pose and 3D pose information obtained from nearest neighbors through database. We assume that in case of 2D image feature sets $\mathcal{F}_{\text{video}}^{10}$ extraction and normalization process, we may get some pose information which causes back and forth unnecessary movement. To avoid this situation, we introduce pose adjustment unit which compels the 3D reconstructed pose according to the k-nearest neighbors in Principal Component Analysis (PCA) space,

$$E_{pa} = \| \frac{1}{\sqrt{N_t}}(P_t^r - M_t)^T C^{-1} (P_t^r - M_t) \|^2 \tag{8}$$

Where P_t^r is the reconstructed pose, M_t is the mean vector of $k$nn-examples at frame t, C^{-1} is the inverse of the covariance matrix and $(P_t^r - M)^T$ is the transpose of the difference between extracted pose and the mean vector.

Table 1: Databases for experimental scenarios.

Databases	Details
DB_{comp}	It contains HDM05 with elevation angles (0-30-90) and azimuth angles (0-30-360).
$\overline{DB}_{comp}$	It includes HDM05 while elevation angles are (0-15-90) and azimuth angles are (0-20-360).
$\overline{\overline{DB}}_{comp}$	It contains HDM05 with elevation angles (0-10-90) and azimuth angles (0-10-360).
DB_{actor}	This database contains all motions of just one performing actor. e.g. $\overline{DB}_{mm}$
$DB_{actorMin}$	It contains all motions of HDM05 excluding motions of one actor. e.g. $\overline{DB}_{mmMin}$
$DB_{actorMirr}$	It contains only one actor's motions with mirroring copies as well. e.g. $\overline{DB}_{mmMirr}$

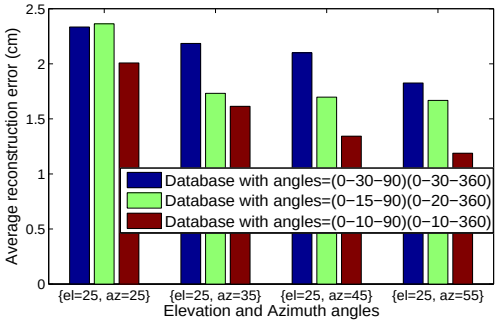


Figure 4: Average reconstruction error for databases with different viewing angle step sizes, when walking motion is given as query motion.

6. PERFORMANCE EVALUATION

We elaborate the performance of our proposed approach by modeling a variety of databases as described in Table 1 with respect to different experimental scenarios, using motion capture database HDM05 [10]. This is heterogeneous database which is consisting of 70 different motion classes performed by five different actors and thus resulting into more or less 1500 motion clips, 381,157 frames at 30 Hz and 50 minutes motion capture data. In order to evaluate performance of the method, the Euclidean distance in centimeters between each frame of original motion and reconstructed motion has been calculated and then accumulated by taking average, referred as *average reconstruction error*.

Before going into detail for evaluation, we have performed some pre-experiments in order to set the suitable value for parameter k. After setting the various values of k like (64, 128, 256, 512) at different combination of viewing angles, we observe that when the value of k is kept 256, best results in terms of reconstruction error have been executed. The value of k may vary depending on the size of the database. In our case, the value of k is set to be 256 for all other experiments.

6.1 Evaluation based on Synthetic Data

We have tested effectiveness of our approach on 2D synthetic data. We refer the 2D information that were obtained from motion capture data by projection as synthetic input

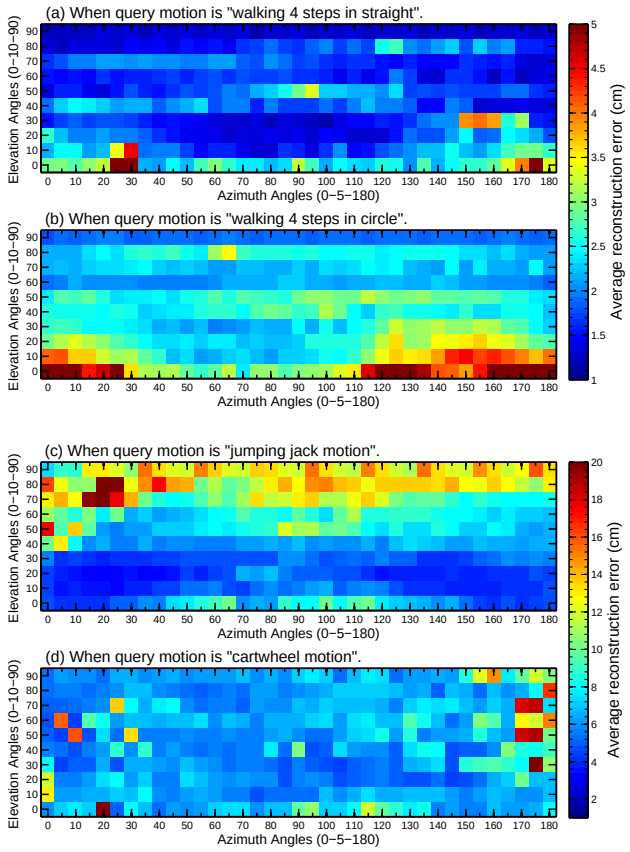


Figure 5: Average reconstruction error graphs for different types of motions: (a) Walking in straight; (b) Walking in circle; (c) Jumping jack motion; (d) Cartwheel motion.

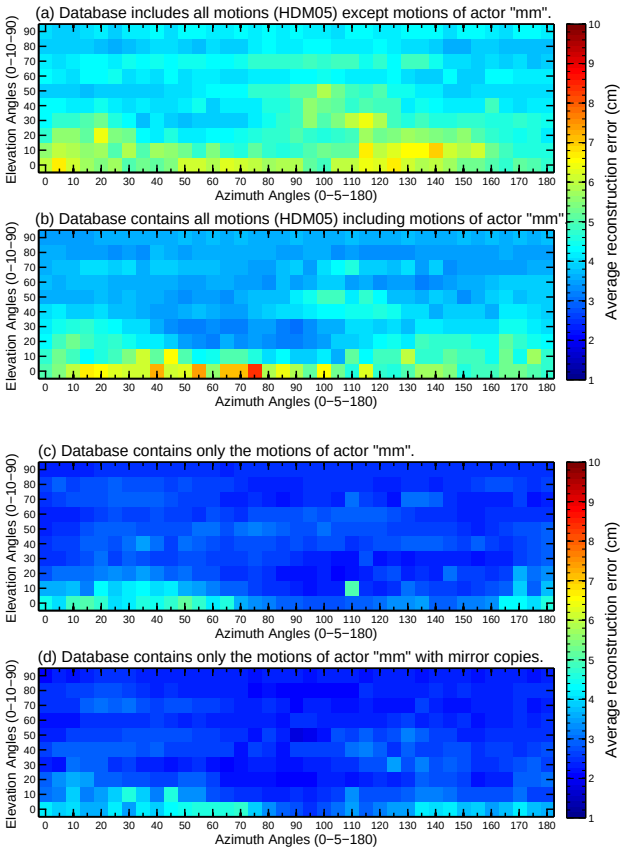


Figure 6: Average reconstruction error graphs for walking in straight motion, when databases are (a) $\overline{DB}_{mmMin}$ (b) $\overline{DB}_{comp}$ (c) $\overline{DB}_{mm}$ (d) $\overline{DB}_{mmMirr}$.

data. The experimental scenarios used for evaluation based on synthetic data, have been decomposed into two categories: First, diversity of elevation and azimuth angles and second, diversity of database in terms of performing actor.

Diversity of Elevation and Azimuth Angles.

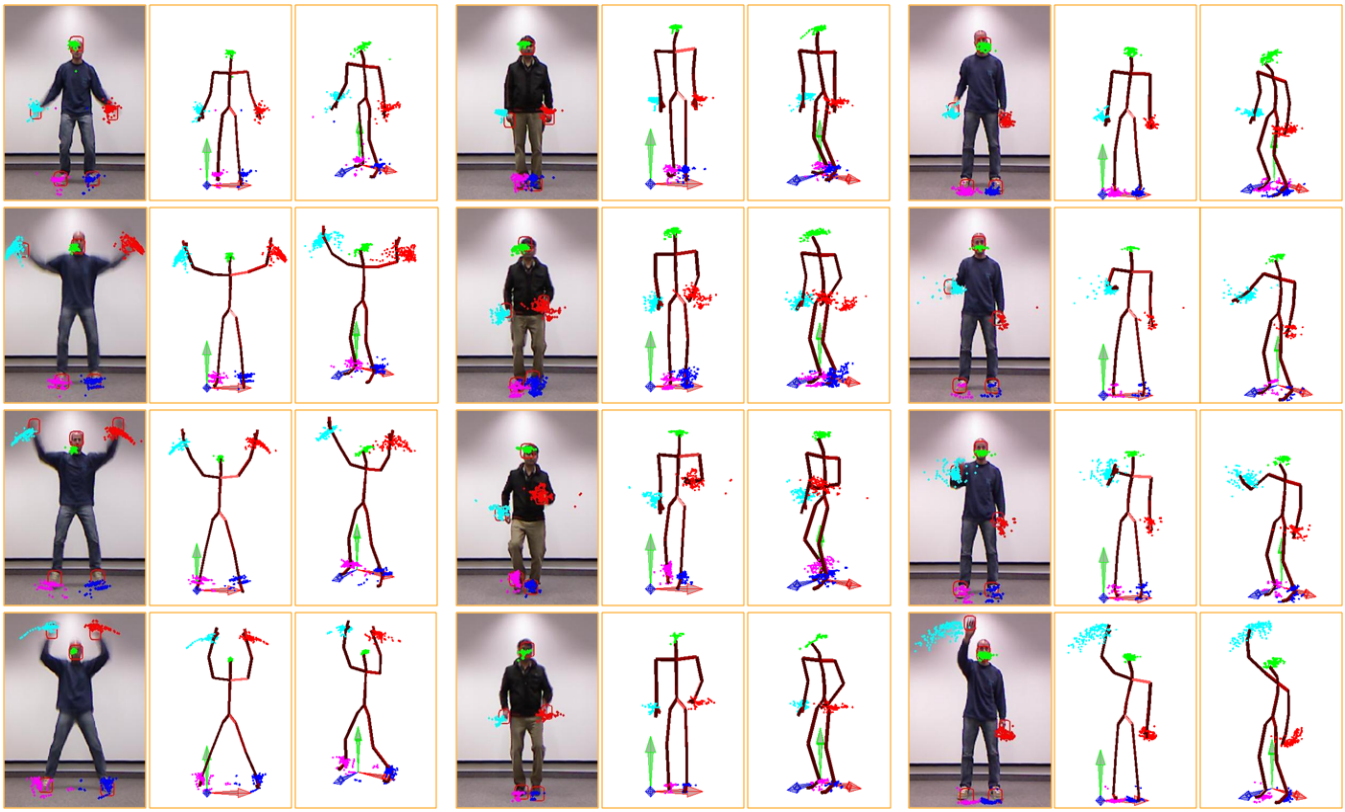
In this experimental scenario, we demonstrate that how the databases with diversity of elevation and azimuth angles impact on system's performance and also elaborate the facts that how the elevation and azimuth angles affect the results in case of different types of motions like walking, jumping jack, and cartwheel motions.

In the first part of the experiment, different databases with various step sizes like with step sizes 30, 20, 15 and 10 degree for viewing angles have been developed as described in Table 1 to check the performance of the developed algorithm. From the experiments, we discover that whenever the database with reduced step sizes has been used, system performs more efficiently as shown in Figure 4. The database with reduced viewing angle step sizes, no doubt, gives better results but also cover more memory space. So, considering both, performance and memory space, the database with elevation angles (0-15-90) and azimuth angles (0-20-360) has been selected in this paper for further experiments.

In the second part of the experiment, we have testified our approach on various kinds of motions like; walking straight, walking in circle, jumping jack and cartwheel motions, and have found some interesting and significant facts which are described in detail as follows.

For **walking motions**, it has been discovered from experiments that at *top view*, best results have been obtained in terms of reconstruction error due to the reason that for walking motion, *top view* executes more clean and clear viewing information as compared to other viewing angles. Similarly, *side view* also shows some optimum results but on the other hand, when there is *front view*, reconstruction error seems to be highest due to the fact that at front side it is difficult to capture detailed information of the hands and feet's movement precisely. As a conclusive remarks, the best suitable view in case of walking motion of all types is the combination of top and side views and the worst is when it is viewed at front side at lower elevation angles. All these significant conclusions are quite obvious in *average reconstruction error graph* in Figure 5 (a) and (b). In this graph, we represent azimuth angles from 0 to 180 degree with step size 5 degree along x-axis and elevation angles from 0 to 90 degree with step size 10 degree along y-axis. The error in the corresponding reconstructions is color coded.

In case of **jumping jack motions**, an opposite behavior to walking motions has been observed because movement of hands and feet are just opposite to walking motion. From



(a) Jumping Jack motion (b) Jogging on Spot motion (c) Grabbing Top motion

Figure 7: Reconstruction results of different types of motions with extracted k-nearest neighbors from motion capture database in case of video input query: (first column) shows video input with 2D feature set detection and extracted k-nearest neighbors; (second column) represents relevant reconstructed motions with extracted k-nearest neighbors; (third column) demonstrates reconstructed motions at some other viewpoint.

the *top view* and *side view*, poor results have been executed due to the deficient captured information of hands and feet but when there is *front view*, best results in relation to reconstruction error are obtained as obvious in Figure 5 (c).

For **cartwheel motions**, no such type of behavior like in walking motions or jumping jack motions has been observed. For all viewing elevation and azimuth angles, approximately similar results in the context of average reconstruction error are found as shown in Figure 5 (d).

Diversity of Databases in terms of Actors.

We have also evaluated our proposed system for different sorts of databases with reference to performing actor as explained in Table 1. To carry out the experiment, we deploy databases with different sizes like $\overline{DB}_{comp}$, $\overline{DB}_{mmMin}$, $\overline{DB}_{mm}$ and $\overline{DB}_{mmMirr}$. The database $\overline{DB}_{comp}$ consists of complete motion capture database HDM05 while $\overline{DB}_{mmMin}$ includes motion capture database HDM05 excluding all motions of performing actor mm. The database $\overline{DB}_{mm}$ contains only the motions of the actor mm and the database $\overline{DB}_{mmMirr}$ also has the mirror copies of the motions of the actor mm. From various experiments, it has been noticed that our system performs well even in a situation when the performing actor is not the part of database as shown in Figure 6. From results, it is quite apparent that when the database $\overline{DB}_{mmMin}$ is exercised, the worst results in terms of reconstruction error have been obtained because the relevant actor mm is not the part of database. For database $\overline{DB}_{comp}$, the results are quite better due to the reason that the motions of the concerned actor mm are now included in database. When the database $\overline{DB}_{mm}$ is employed, even better results are observed because now the database has only the motions of the relevant actor mm. For the database $\overline{DB}_{mmMirr}$, the best results are obtained due to the reason that the database has not only the motions of the concerned actor mm but also the mirror copies of the motions in database. Similar behavior has been observed for all types of motions and other performing actors too.

6.2 Evaluation based on Video Data

The proposed algorithm's performance is tested on variety of uncaliberated video streaming too, like jumping jack motions, grabbing motions and jogging motions etc. The $\overline{DB}_{comp}$ database is deployed as knowledge base in case of video data as input. The video data is first pre-processed as mentioned above in order to get relevant information required for input query. Some results of reconstructions based on video data are presented in Figure 7 and in detail in the supplemental video. The results are quite acceptable even in case of noisy input data. The noisy data is because of some missing and deficient information in detection and tracking of 2D image feature sets $\mathcal{F}^{10}_{\text{video}}$. The deficient information

in detection and tracking may be due to the reasons: in a few frames, the features cannot be detected at all as a result of some illumination, occlusion or blurring effects; sometime hands and feet's movements are inconsistent e.g. hands or feet move very fast in a frame as compared to the movement in previous frames; hands' and feet positions and especially orientations vary continuously. All these factors may lead to problem in feature detection and tracking and as a solution, user annotations are employed where needed to acquire accurate 2D image feature sets from video data signals.

7. CONCLUSION & FUTURE WORK

In this paper, we present an efficient model based approach to reconstruct human motion from different types of 2D input data signals. Our developed system can reconstruct full body human motion efficiently in a real time even when a low-dimensional 2D feature sets are given as input query either in the form of 2D synthetic data signals or 2D uncalibrated monocular video sequences. We have testified the effectiveness of our system on wide variety of databases and various types of human motions like walking, cartwheel, jumping jack or jogging motions. Our system performs reconstruction approximately 5-8 frames per second.

In future work, the feature detection and tracking technique can be made more robust using previous frame information and 3D $k$nn obtained from the database. The visual cues like silhouette extracted from input video might be incorporated in energy function in order to improve the pose by pose motion reconstruction process. The weak perspective camera model can be extended to full perspective model with all intrinsic and extrinsic camera parameters with 11 degree of freedom. 3D $k$nn obtained from the database might be utilized in the process of estimation of orientation and translation (extrinsic camera parameters). Temporal information may be helpful to make the system more robust and fast by employing online lazy neighborhood graph presented in [17]. Moreover, instead of single static camera moving cameras might be deployed as future work.

8. REFERENCES

[1] A. Baak, M. Müller, G. Bharaj, H.-P. Seidel, and C. Theobalt. A data-driven approach for real-time full body pose reconstruction from a depth camera. In *IEEE 13th International Conference on Computer Vision (ICCV)*, pages 1092–1099. IEEE, Nov. 2011.

[2] Carnegie Mellon University Graphics Lab. CMU Motion Capture Database, 2013. `mocap.cs.cmu.edu`.

[3] J. Chai and J. K. Hodgins. Performance animation from low-dimensional control signals. *ACM Trans. Graph.*, 24(3):686–696, July 2005.

[4] Y.-L. Chen and J. Chai. 3d reconstruction of human motion and skeleton from uncalibrated monocular video. In *Computer Vision - ACCV 2009*, volume 5994 of *Lecture Notes in Computer Science*, pages 71–82. Springer Berlin Heidelberg, 2010.

[5] G. Guerra-Filho and A. Biswas. The human motion database: A cognitive and parametric sampling of human motion. *Image and Vision Computing*, 30(3):251 – 261, 2012.

[6] A. Hornung. *Shape Representations for Image-based Applications*. PhD Dissertation, RWTH Aachen, 2009.

[7] E. Jain, Y. Sheikh, M. Mahler, and J. Hodgins. Three-dimensional proxies for hand-drawn characters. *ACM Trans. Graph.*, 31(1):8:1–8:16, Feb. 2012.

[8] B. Krüger, J. Tautges, A. Weber, and A. Zinke. Fast local and global similarity searches in large motion capture databases. In *2010 ACM SIGGRAPH / Eurographics Symposium on Computer Animation*, SCA '10, pages 1–10, Aire-la-Ville, Switzerland, Switzerland, July 2010. Eurographics Association.

[9] D. M. Mount and S. Arya. ANN: a library for approximate nearest neighbor searching. Programming manual, Department of Computer Science, University of Maryland, College Park, Maryland, U.S.A., 2006.

[10] M. Müller, T. Röder, M. Clausen, B. Eberhardt, B. Krüger, and A. Weber. Documentation Mocap Database HDM05. Technical Report CG-2007-2, Universität Bonn, June 2007.

[11] H. S. Park and Y. Sheikh. 3d reconstruction of a smooth articulated trajectory from a monocular image sequence. In *Proceedings of the 2011 International Conference on Computer Vision*, ICCV '11, pages 201–208, Washington, DC, USA, 2011. IEEE Computer Society.

[12] M. J. Park, M. G. Choi, and S. Y. Shin. Human motion reconstruction from inter-frame feature correspondences of a single video stream using a motion library. In *Proceedings of the 2002 ACM SIGGRAPH/Eurographics symposium on Computer animation*, SCA '02, pages 113–120, New York, NY, USA, 2002.

[13] V. Ramakrishna, T. Kanade, and Y. Sheikh. Reconstructing 3d human pose from 2d image landmarks. *Computer Vision–ECCV 2012*, pages 573–586, 2012.

[14] L. Rocha, L. Velho, and P. Carvalho. Motion reconstruction using moments analysis. In *Computer Graphics and Image Processing, 2004. Proceedings. 17th Brazilian Symposium on*, pages 354–361, Oct 2004.

[15] N. Roodsarabi and A. Behrad. 3d human motion reconstruction using video processing. In *Image and Signal Processing*, volume 5099 of *Lecture Notes in Computer Science*, pages 386–395. Springer Berlin Heidelberg, 2008.

[16] T. Sattler, B. Leibe, and L. Kobbelt. Fast image-based localization using direct 2d-to-3d matching. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 667–674, Nov 2011.

[17] J. Tautges, A. Zinke, B. Krüger, J. Baumann, A. Weber, T. Helten, M. Müller, H.-P. Seidel, and B. Eberhardt. Motion reconstruction using sparse accelerometer data. *ACM Trans. Graph.*, 30(3):18:1–18:12, May 2011.

[18] X. Wei and J. Chai. Videomocap: modeling physically realistic human motion from monocular video sequences. *ACM Trans. Graph.*, 29(4):42:1–42:10, July 2010.

[19] X. Wu, M. Tournier, and L. Reveret. Natural character posing from a large motion database. *IEEE Comput. Graph. Appl.*, 31(3):69–77, May 2011.