Fast Skeleton Estimation from Motion Capture Data using Generalized Delogne-Kåsa method

Jonathan Kipling Knight and SK Semwal
Department of Computer Science
University of Colorado
Colorado Springs, CO. 80933, USA
kip.knight@itt.com | semwal@cs.uccs.edu

ABSTRACT

This paper presents a fast closed-form solution estimating the rotation points of joints relative to the motion capture data. The proposed solution estimates the physical location of joints inside the body of the person wearing the trackers. The Generalized Delogne-Kåsa method used for our implementation fits spheres, cylinders, circles and planes to the motion capture data without the need for initial guessing. This non-iterative, closed-form solution is fast as it calculates the rotation point with O(N) averaging along with one inversion of a 3x3 positive semi-definite matrix for each joint. The error in the joint location is on average $3\sigma/\sqrt{N}$ which is low. In addition, sample points for every joint can be from different time sequence allowing flexibility in recovering the joint locations. Once the joint location relative to the tracker position is determined, it could be used for the remainder of the data set. Publicly available CMU motion capture data was used for this study. Two animation sequences, showing our method, are included with this paper. These results can be compared to that available at the CMU site for the same animation. Since the pose is found relative to the given data, our pose estimation provide better fit to the given data, revealing subtle, individual nuances of the person used for the motion capture. Because of the closed form solution, our technique is ideally suited for the use of motion captured data to create skeletal motion in 3D games or applications where real time performance is essential.

Key words: sphere curve fit, joint location, articulated motion, Delogne-Kåsa Method, motion-capture data

1. Introduction

Motion capture animation has been continuously improved by many authors [2, 5, 6, 12-21, 25, 27, 28, 32]. Their studies can be divided into two methods of analysis: kinematic and kinetic [1]. In kinematic methods, scientists study the mechanical displacements of the limbs during motion. In kinetic (or dynamics) methods, the energies and forces on the limbs are studied during the motion of the articulated figure.

Kinematic methods are used in animation by determining the joint angles from space-time constraints. Holt et al. [8] estimated the 3D motion of an articulated object from a sequence of 2D perspective views. They used a decomposition approach to break down the motion of each segment. This was a good use of video motion capture to estimate the animation of a figure. Choi [5] improved standard forward kinematic techniques to provide more accurate end-effector positions. Inverse kinematics is another technique employed by many authors [19]. Kinematics tends to be a faster technique than kinetics.

Kinetic methods consider the changes in energy, inertia, and forces to create animation. These changes determine the way the joint angles change in time. Semwal et al. [24] visualize the leg rotations of a cyclist showing forces during the pedal-movement. A straightforward approach is to use Newtonian or Lagrangian mechanics, and involves solving simultaneous second order partial differential equations. Recursive methods

 ISBN 978-80-86943-98-5

can be more efficient [30]. Liu and Popović [15] infuse physical reality in to sparsely keyed motion data. An articulated figure realistically played hopscotch using a minimal set of pre-determined positions. Liu and Popović [15] relied on off-line calculations. Popović and Witkin [21] discuss a novel application by transforming standard models of motion in to a diverse assortment of similar motions. For instance, a standard run sequence can transform into a run with a limp-sequence, while retaining realism.

Recently, motion captured data has become a rich source for providing realistic human motion. Content based retrieval [17] and using lower dimensional data [5] are recent examples of use of the motion captured data in recent studies. Kinetics is of interest on the motion captured data in Zordan [32]. Most of the existing methods involve an initial guess of the underlying skeleton and iterating through until a stable solution evolves. Various forms of least squares fitting are the most popular, most of which use the Levenberg-Marquart method for solving the minimum. O'Brien [19] calculates a skeleton and then uses inverse kinematics to produce motion from motion capture data. Our method also belongs to the kinematics group by analyzing motion-captured data to find a skeleton within the data similar to O'Brien [19]. The proposed method improves on the work of O'Brien [19] by providing a faster replacement for the least squares fit of joints. Our technique is more computationally efficient than O'Brien because once the offsets from the sensors are calculated during pre-processing, they are reused during skeletal animation. This direct approach captures the subtle deformations that the human body is capable of undergoing during complex movements and is inherent in the sensor (motion captured) data. These subtle body deformations are also present in our skeletal animation because of our algorithm, thus improving the quality of animation. Although not a focus of this paper, an incremental improvement technique, being implemented, would provide yet another order of magnitude improvement in future.

An articulated figure can be divided into rigid *segments* that are connected to each other by joints. A tree structure is a perfect way to organize these segments [19]. Root to leaf processing of segment properties allow for a dependency on the parent segment's position and orientation. Such a dependency is exploited in this paper. The tree structure of segments can be known ahead of time (a-priori). When the tree structure is not known, e.g. motion data from an unknown figure), much more off-line analysis is needed to produce the hierarchy. The solution is the equivalent of a minimal spanning tree [19]. Ko and Cremer [11] used 34 DOF for their a-priori system. Bolt [4] used 17 DOF in his model of the human figure. The proposed research in this paper extends to arbitrary number of DOF, and motion captured from other animal forms. Although implemented for a human skeleton that conforms to a tree hierarchy, our technique easily extends to the case when tree structure is not apparent, and animation must directly work from the sensor data.

2. Closed Form Rotation Point Determination

In our research, it was found that a closed-form solution existed when the variance of the square of the distance from the measurement to the point of rotation is minimized. Zelniker [31] shows that the 2D equivalent had been studied and rediscovered since the 1970's. Zelniker generalized the method to any number of dimensions and proved that the bias in the estimation is not significant enough. Our Monte Carlo experiments comparing the Least Squares Linear Regression (LS), and the Generalized Delogne-Kåsa Estimator (GDKE) show them to have approximately the same error in answer. The speed is improved when analyzing the same count as shown in Figure 1.

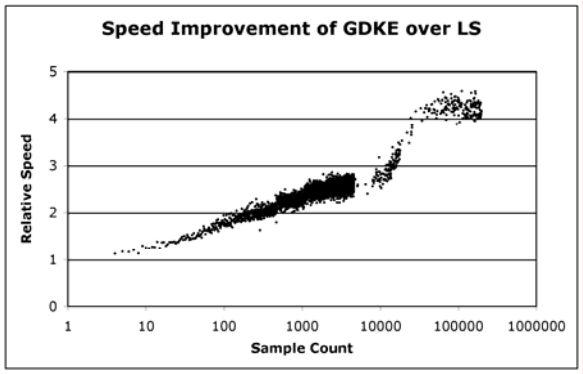

Figure 1: Relative speed improvement of GDKE versus linear least-squares.

Previous solutions [3, 7] and recently O'Brien et.al. [19] determine the rotation point by iterating on a least-squares equation or M-estimators starting from an initial guess. This approach involves either linear or non-linear fitting of the data and has a chance of not converging to a solution. The initial guess must be close enough to the truth or the iterations may diverge away from the point. The GDKE solution to the center of rotation involves no guessing, no iterations, and the inversion of a 3x3 matrix. The GDKE is robust with noise and also can distinguish cylindrical joint motion, with a little extra work. The speedup shows that our method is computationally more efficient than the least-

square method of O'Brien's [19], and is a suitable replacement for both linear and non-linear least-squares fitting of a sphere, cylinder, circle, and a plane.

The requirements in order to determine the point of rotation for this method are: (a) Known fixed axes on a parent segment. (b) No translational movement. (c) Enough rotational motion. (d) At least 4 positions of one marker. The above requirements are satisfied for the problem in hand which amounts to finding the best-fit sphere for a 2 or 3 degrees-of-freedom (DOF) joint or the best-fit cylinder for a 1 DOF joint. According to the National Institute of Standards and Technology [26] the best approach to this problem is non-linear Least-Squares fitting but that method could be problematic for the general case. O'Brien [19] uses linear least-squares fitting. Our research, consistent with recent results in Zelniker's [31], shows that the GDKE is a viable and robust replacement for speeding up the calculations as explained below.

2.1. Theory

Presented here is a derivation of the Generalized Delogne-Kåsa Estimator (GDKE). The estimator is a closed form solution for a hypersphere that has O(N) averaging and involves the inversion of a 3x3 positive-semidefinite matrix. Stated simply -- the solution involves solving for the absolute minimum of the variance of the square of the lengths from the point of rotation to each position for a single marker. The ith position for the jth joint (x_{ji}) around the joint's rotation point (c_j) can be expressed as

$$x_{ji} = c_j + R_{ji}\rho_{ji}$$

where

$$\left|\rho_{ji}\right| = 1$$

This position (x_{ji}) is not the absolute position that comes directly from motion capture. Instead it is relative to the parent segment in its fixed coordinate system. The square of the radius of the measurement (R_{ji}) has a variance of

$$s_j^2 = \tfrac{1}{N_j - 1}\sum_{i=1}^{N_j}\left(R_{ji}^2 - \hat{R}_j^2\right)^2 \qquad (1)$$

where the radius can be written as

$$R_{ji} \equiv \left|x_{ji} - c_j\right|$$

and the average square of the radius is

$$\hat{R}_j^2 = \tfrac{1}{N_j}\sum_{i=1}^{N_j} R_{ji}^2 .$$

Equation 1 is then minimized to solve for the GDKE rotation point (c_j). Setting the gradient of the variance to zero thus

$$\nabla s_j^2 = 0$$

will provide an equation to find c_j. The gradient can be written, after much algebraic manipulation not presented here due to space-limitation, as

$$\nabla s_j^2 = 4\left(2\mathbf{C}_j\left(c_j - \bar{x}_j\right) - \mathbf{S}_j\right)$$

where $\mathbf{C}_j$ is the standard definition of the sample variance-covariance matrix for a vector quantity given by

$$\mathbf{C}_j = \tfrac{1}{N_j - 1}\sum_{i=1}^{N_j}\left(x_{ji} - \bar{x}_j\right)\left(x_{ji} - \bar{x}_j\right)^T \qquad (2)$$

The average is given by

$$\bar{x}_j = \tfrac{1}{N_j}\sum_{i=1}^{N_j} x_{ji} . \qquad (3)$$

The variance-covariance matrix (2) is a biased estimator for the covariance of the vector measurement. $\mathbf{C}_j$ has been used in many situations [9] for curve fitting and has proven useful in identifying the best-fit plane for data.

The other vector quantity $\mathbf{S}_j$ can be considered the vector equivalent of the third central moment and is

$$\mathbf{S}_j = \tfrac{1}{N_j - 1}\sum_{i=1}^{N_j}\left(x_{ji} - \bar{x}_j\right)\left(x_{ji} - \bar{x}_j\right)^T\left(x_{ji} - \bar{x}_j\right) \qquad (4)$$

Equation 4 is an unbiased mix of multi-dimensional moments, and is a new quantity, not available in [29, 23]. The final solution for the center of the hypersphere (i.e. the relative joint location) is

$$c_j = \bar{x}_j + \tfrac{1}{2}\mathbf{C}_j^{-1}\mathbf{S}_j . \qquad (5)$$

This formula looks different than Zelniker [31] but is algebraically equivalent.

The variance-covariance matrix $\mathbf{C}_j$ is positive-semidefinite so Cholesky decomposition provides a more efficient solution to the equation. There are two cases that Cholesky decomposition produces an undesirable answer or even fails. The trivial case is if all points coincide, then $\mathbf{C}_j$ is singular. This case occurs

when the joint does not move. The non-trivial case is during planar motion. If there exists a vector n such that

$$x_{ji}^T n = k$$

where k and n are constants for all positions of the marker, then $\mathbf{C}_j$ is singular. All is not lost though with these singularities or even near singularities. The planar motion can still be solved for as explained below.

The Null Space of a matrix is a set of vectors that solve the equation $\mathbf{C}_j n=0$. This set of vectors is inherently extracted during the Singular Value Decomposition (SVD) [22] of a matrix based on some threshold. The condition number of $\mathbf{C}_j$ will provide the threshold and the measure for which to check if the motion is planar. The condition number is the ratio of the largest eigenvalue to the smallest eigenvalue. If the threshold is equal to the inverse of the condition number then a single vector exists in the Null Space of $\mathbf{C}_j$ and that vector happens to be the normal to the plane of motion. That vector is also the eigenvector that corresponds to the smallest eigenvalue. This is proven by solving for the minimum of the variance of the distance from the plane. The distance (z_{ji}) from the plane for each point is

$$z_{ji} = x_{ji}^T n.$$

The gradient of the variance is then determined to be

$$\nabla Var\left(z_j\right) = 2\mathbf{C}_j n$$

which clearly shows the minimum occurs when $\mathbf{C}_j n=0$. The only solution to this equation is the Null Space of the variance-covariance matrix $\mathbf{C}_j$. If the motion is determined to be planar the null vector can be used to determine the center of the circle on the plane. The equation for the best-fit circle is then

$$c'_j = c_j + nn^T\left(\bar{x}_j - c_j\right) \tag{6}$$

which basically removes one dimension along the null vector for the solution to the hypersphere.

So we have a solution for the equation when it is both singular and positive. How do we know it is unique or worse yet, maybe it is a maximum and not a minimum? The answer can be achieved by finding the double derivative of s^2_j (i.e. the Hessian). The Hessian is determined as

$$\tfrac{1}{2}\nabla\nabla^T s_j^2 = 4\mathbf{C}_j.$$

Since the Hessian is positive-semidefinite, the answer in Equation 1 is proven to be an absolute minimum.

2.2 Monte Carlo Simulations

The GDKE method estimator has been shown to be biased [31]. Zelniker's analysis showed that the bias is of order of the standard deviation of the measurements. This bias is actually not significant enough to warrant dismissal of the method. We conducted Monte Carlo simulations of spherical data to confirm our claim. Data points were produced uniformly on a sphere with a known error introduced to the positions. The number of points were uniformly chosen between 4 and 1000 inclusive. The known error was picked with a LogNormal(0,10) distribution. The radius of the sphere was chosen with a LogNormal(0.18,1). The center of the sphere was chosen with Gaussian(0,0.5) uniformly around the point (0.6,-0.2,0.9). The simulation was run 5000 times and GDK estimators were calculated. The bias from the true center was analyzed using the Ordered Statistics method [10] of determining the probability distribution. The data showed that the bias magnitude is proportional to $\sigma/\sqrt{N}$. The ordered statistics method shows a best fit probability of Weibull(2.378, 3.302) as can be seen in Figure 2.

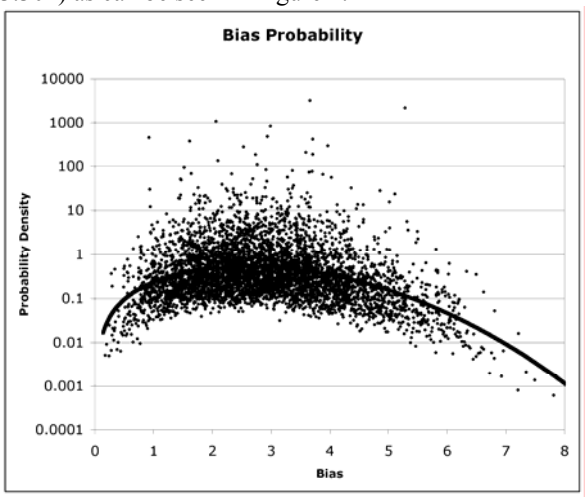

Figure 2: Probability Density for the bias for the GDKE center.

The resultant distribution has an expected value of 2.93 and a standard deviation of 1.31. So the final center estimator for the sphere can be expressed empirically as

$$\left|c - c_0\right| \approx 2.93\frac{\sigma}{\sqrt{N}} \tag{7}$$

which is $O(\sigma/\sqrt{N})$ where N is typically 100 samples during preprocessing. Here, c_0 is the true center, σ is the true standard deviation of the positions, N is the

sample count. The radius was similarly analyzed to find that the value is unbiased when

$$\sigma < 0.1R_0$$

as shown in Figure 3. R_0 is the true radius. This situation occurs for most measurements and makes physical sense since one does not want to have an inaccuracy anywhere near the radius of the sphere (i.e. $\sigma = R_0$). The empirical formula for this situation is

$$\left|R - R_0\right| \approx 0.$$

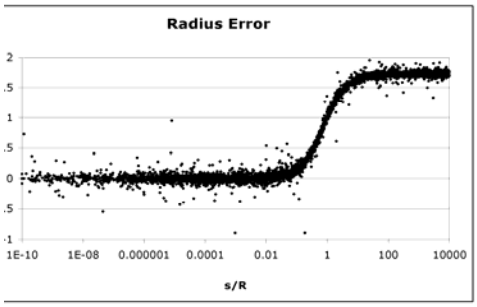

Figure 3: Error in GDKE radius estimate

These experiments and the analysis from Zelniker show that the GDKE method is a better replacement for the slower linear and non-linear least-squares fitting used by most authors, including OBrien et al. [19].

2.3. Procedure for Determining Center

The procedure to determine point of rotation of a joint is as follows. This method automatically determines the type of joint, either spherical, cylindrical or not moving at all.

1) Choose one marker with absolute positions x'_{ji} on segment.
2) Make points relative to parent.
$\mathbf{M}_{j-1}$ = column matrix of parent axes
p_{j-1} = center of parent's coordinate frame

$$x_{ji} = \mathbf{M}_{j-1}^{T}\left(x'_{ji} - p_{j-1}\right)$$

3) Calculate mean, covariance and third central moment of points using Equations 2,3,4.
4) Using singular-value decomposition, determine the condition number, the null-space vector, and the solution for the best-fit sphere. Use Equation 5.
5) If condition number is small then use spherical solution otherwise find the rotation center for the planar motion. Use Equation 6.
6) Store the center of rotation and the null-space vector for later use of displaying skeleton at each frame.

2.4. Applying the GDKE to Motion captured Data

The motion capture data used in this paper comes mainly from the very large database (>2GB) of motions captured by Carnegie Mellon University Graphics Lab (CMU Labs). The data is freely downloadable at http://mocap.cs.cmu.edu/. The database was created with funding from the NSF grant EIA-0196217. There are 1576 trials in 6 categories and 23 subcategories. We also analyzed the data from Advanced Computing Center for the Arts and Design (ACCAD) at http://accad.osu.edu/.

Motion capture information is acquired on up to forty or more markers on the body depending on the motion being studied. The CMU data comes in a few file formats. The raw Cartesian coordinates of the data are stored in the C3D file format [18]. Each time frame is stored, with each frame consisting of X,Y,Z for each marker on the body. If any data is missing from a frame, that point is zeroed and marked. The captured motion also comes in the form of ASF and AMC file formats. These are created after the VICON program has done analysis of the data. The ASF format stores the skeleton and joint information to create an articulated figure on the screen. The AMC format contains every time frame's translation and rotation for each bone. Our method works with the raw XYZ data and therefore does not consider the post-analysis data in the AMC files. A motion capture file (e.g. C3D file) contains the data, as well as a name associated with each marker. For example, a marker is placed on the right thigh and is called "JOE::RTHI". Usually, the animator doesn't have the same designation so a cross-correlation must be achieved to identify which segment this marker belongs. Our implementation uses a two-file process to cross-correlate a C3D data-set with the segments on the ATM model.

2.3. Extracting Joint Information in Our Implementation

Most information in motion capture data comes in the form of Cartesian coordinates of markers placed on the segments of an articulated figure. Each joint between segments has a point of rotation (if purely rotational) and joint axes. Magnetic tracking sometimes comes with orientation axes so much of this section can be skipped if that is the case. Usually, and in our case, the frames of data must be analyzed during joint motion to determine the axes and rotation point. The center of rotation can be calculated for each joint if: 1) The orientation of the parent segment can be evaluated. 2) The

child segment has at least one marker. 3) The child segment is in motion relative to the parent segment.

The orientation of a segment can be determined if there are at least three non-linear data points fixed to that segment. One of the data points can be the center of rotation. There is a hierarchical dependency for determining the center of rotation and the orientation. Thus, the orientation can be determined if there: 1) is one data point, rotation point, and a rotation axis; 2) are two data points and one rotation point, all non-collinear; 3) are three non-collinear data points. The information can be retrieved if calculated hierarchically from root to leaf. First, define the tree. Then, assign the markers to their appropriate segments. The root must have at least three markers. No center of rotation for the root can be determined. The children of the root can determine their center of rotation relative to their parent by the above mentioned method of GDKE.

The rotation point c_{jk} for the current time frame k is determined from

$$c_{jk} = x_{jk} + \mathbf{M}_{jk} v_j$$

where v_j is the local relative vector as determined from GDKE, x_{jk} is the center of the coordinate system, usually one of the marker points, and $\mathbf{M}_{jk}$ is the 3x3 matrix of column vectors that represent the axes of the segment's coordinate system.
The coordinate matrix is composed of three column vectors that make the orthogonal coordinate system. It is determined differently for each of the three cases.
1) One data point is available, use null-vector:

$$\hat{x} = \hat{n}$$

2) Two data points are available, use second point and center of rotation:

$$\hat{x} = \frac{p_1 \times c}{|p_1 \times c|}$$

3) Three data points are available, use second and third point:

$$\hat{x} = \frac{p_2 \times p_1}{|p_2 \times p_1|}$$

Then the other two axes can be calculated from the first point by

$$\hat{z} = \frac{p_0 \times \hat{x}}{|p_0 \times \hat{x}|}$$

and

$$\hat{y} = \hat{z} \times \hat{x}$$

These produce the rotation matrix **M** by placing them in the columns.

2.6. Generated Motion

The root segment (hips) motion can be considered freely moving and freely rotating. In order to find the motion of such an object (6 DOF), the segment needs at least three unique points on the surface tracked. The motion capture data can be used to create a set of orientation axes and produce the center for any particular frame of the data. The axes are fixed relative to the root segment and are a 3x3 matrix determined by the above equations. Next we use the motion data set and recursively traverse the tree to traverse down the segments. Each segment in the tree contains its own time-dependent position relative to its parent. To position the figure at a specific time-frame involves traversing the tree from the root segment to the endpoints. During the analysis phase, data will be collected giving each segment a chance to calculate its motion constants. Once these constants of the data-set are determined (i.e. relative rotation points and null-vectors), skeletons for the remainder of a motion capture data are easily drawn using these determined constants. This is computationally far efficient than applying the inverse-kinematics solution in existing techniques.

3. Results

We conducted several experiments comparing GDKE method with linear least square method and found that the GDKE produces similar answers to the Least-Squares Method of O'Brien et al. [19] but is computationally more efficient as discussed in the paper. In addition, this new method can also produce a better answer in the cylindrical case. The error in the calculation is only of $O(\sigma/\sqrt{N})$ [31] and corresponds to the bias as explained in the empirical relationship in Equation 7.

An example of a single time-frame for a karate pose is given in Figure 4. A more complex data-set was analyzed involving two figures during a salsa dance. A time-frame from that data is given in Figure 5.

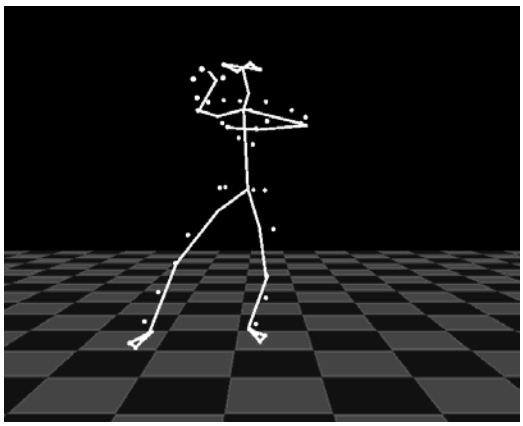

Figure 4: Karate Pose from ACCAD data "Break Dancer 1"

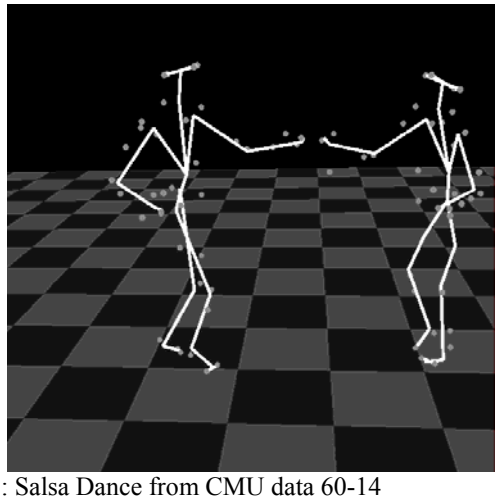

Figure 5: Salsa Dance from CMU data 60-14

Noise can also contribute to the propagation of errors. Noise comes in two flavors, measurement noise and process errors. A look at the problem at hand, i.e. measured markers on a body, shows how errors are introduced. The accuracy of capturing the coordinates of the marker in time is considered measurement noise. This is usually a simple Gaussian noise for each measurement. Process errors are produced when, for example, the marker is placed on loose clothing and subsequently moves slightly on the body during motion. All these errors contribute to the accuracy of the rotation point calculation whichever method is chosen.

4. Conclusions and Future Research

We have presented a GDKE implementation that is computationally superior to the method of O'Brien et al. [19]. In addition, our method is faster that inverse kinematic solutions as the actual joint locations are directly calculated from the sensor data. These estimates are found by using a fast closed form solution. As our method derives the animation from sensor data, animation sequences preserve the subtle variations of the person used for the motion capture study. Qualitatively, this can be observed while playing the CMU data and our animation side by side. Our algorithm is well suited for on-the-fly capture and display of the motion capture data, and would allow merging of animation sequences with ease. In the future, we plan to utilize an incremental improvement technique, which could further provide an order of magnitude speed improvement.

Acknowledgements

CMU motion capture data sets were essential for our study. We gratefully acknowledge all who were involved in collecting this invaluable resource that was publicly available to us. Our "break-dance" and "salsa" movies correspond to the CMU data-set 85-14 and 60-14 respectively. Other data-sets analyzed were from ACCAD.

References:

[1] J.K. Aggarwal, Q. Cai, and W. Liao, "Nonrigid motion analysis: articulated and elastic motioin" *Computer Vision and Image Understanding* 70, no 2 (May 1998) : 142-56.

[2] J. Assa, Y. Caspi, D. Cohen-Or, "Action Synopsis : Pose selection and illustration" *ACM Transaction on Graphics (SigGraph 2005)* 24(3) (July 2005) : 667-676.

[3] B. Bodenheimer, C. Rose, S. Rosenthal, J. Pella, "The Process of Motion Capture: Dealing with the Data" *Eurographics* CAS (Sept. 1997): 3-18

[4] D.A. Bolt, "Two Stage Control for High Degree of Freedom Articulated Figures" Ph.D. thesis for Computer Science, University of Colorado in Colorado Springs, (2000).

[5] J. Chai, J. Hodgins "Performance Animation from Low Dimensional Control-Signals" *ACM Transaction on Graphics (SigGraph 2005)* 24(3) (July 1005) : 686-696.

[6] K. Choi, S. Park, H. Ko. Processing motion capture data to achieve positional accuracy. *Graphical Models and Image Processing*, 61(5):260–273, September 1999.

[7] L. Herda, P. Fua, R. Plankers, R Boulic, D. Thalmann, "Skeleton-based motion capture for robust reconstruction of Human Motion", Computer Graphics Lab (LIG) EPFL:Lausane, Switzerland (2000).

[8] R.J. Holt, T.S. Huang, and A.N. Netravali, R.J. Qian, "Determining articulated motion from perspective views: a decomposition approach" *Pattern Recognition*: 30, no. 9,

(1997): 1435.

[9] K. Kanatani. “Cramer-rao lower bounds for curve fitting.” *Graphical Models and Image Processing*, 60(2):93–99, March 1998.

[10] J. K. Knight. Probability Density Function Determination from Ordered Statistics. Masters Thesis for Applied Mathematics, California State University, Fullerton, August 1995.

[11] H. Ko, J. Cremer, “VRLOCO: Real-Time Human Locomotion from Positional Input Stream” *Presence* 5(4) (1996): 367-380.

[12] L. Kovar, M. Gleicher, F. Pighin, “Motion Graphs” *Proceedings of SIGGRAPH 2002*: 473-482.

[13] J. Lee, S.Y. Shin, “A Hierarchical Approach to Interactive Motion Editing for Human-like Figures” *Proceedings of SIGGRAPH 1999.*

[14] J. Lee, J. Chai, P.S.A Reitsma, “Interactive Control of Avatars Animated with Human Motion Data” *Proceedings of SIGGRAPH 2002*: 491-500.

[15] C.K. Liu, Z. Popović, “Synpaper of Complex Dynamic Character Motion from Simple Animation” *Proceedings of SIGGRAPH 2002*: 408-416.

[16] Z. Lui, S. Gortler, M. Cohen, “Hierarchical Spacetime Control” Computer Graphics (*SIGGRAPH 1994 Proceedings*).

[17] M. Muller, T. Roder, M. Clausen, “Efficient Content-Based Retrieval of motion capture Data” *ACM Transaction on Graphics (SigGraph 2005)* 24(3) (July 2005) : 677-685.

[18] Motion Lab Systems, Inc. “C3D Format User Guide”, PDF document at http://www.motion-labs.com, Motion Lab Systems, Baton Rouge, LA, (2003) (98 pages)

[19] J. F. O’Brien, R. E. Bodenheimer Jr., G. J. Brostow, and J. K. Hodgins. *Automatic Joint Parameter Estimation from Magnetic Motion Capture Data*, pages 53–60, Montreal, Quebec, Canada, May 15-17 2000. Graphics Interface.

[20] N.S. Pollard, P.S.A. Reitsma, “Animation of Humanlike Characters: Dynamic Motion Filtering with a Physically Plausible Contact Model” *Yale Workshop on Adaptive and Learning Systems* (2001).

[21] Z. Popović, A. Witkin, “Physically Based Motion Transformation” *Computer Graphics Proceedings*, Annual Conference Series, 1999.

[22] W.H. Press, S.A. Teukolsky, W.T. Vetterling, B.P. Flannery, “Numerical Recipes in C: The Art of Scientific Computing” 2nd.ed., Cambridge University Press: 1992.

[23] P. L. Rosin. Further five point ellipse fitting. *Graphics Models and Image Processing*, 61:245–259, 1999.

[24] S. Semwal, M.J. Parker, “An Animation System for Biomechanical Analysis of Leg Motion and Predicting Injuries during Cycling” *Real-Time Imaging*: 5 (1999): 109-123.

[25] S.K. Semwal, R. Hightower, S. Stansfield, “Closed Form and Geometric Algorithms for Real-Time Control of an Avatar” *IEEE Proceedings of VRAIS 1996*.

[26] C.M. Shakarji “Least-Squares Fitting Algorithms of the NIST Algorithm Testing System” *J. of Research of the National Institute of Standards and Technology*: 103, No. 6 (1998): 633.

[27] D. Tolani, A. Goswami, N.I. Badler, “Real-time inverse kinematics techniques for anthropomorphic limbs” *Graphical Models*: 62 (2000): 353-388.

[28] M. van de Panne, “From Footprints to Animation” *Computer Graphics Forum*: 16, no. 4 (1997): 211-223.

[29] L. Yang, F. Albregtsen, and T. Taxt. “Fast computation of three-dimensional geometric moments using a discrete divergence theorem and a generalization to higher dimensions.” *Graphical Models and Image Processing*, 59(2):97–108, March 1997.

[30] J. Znamenáček and M. Valášek. An efficient implementation of the recursive approach to flexible multibody dynamics. *Multibody System Dynamics*, 2(3):227–251, 1998.

[31] E. E. Zelniker and I. V. L. Clarkson. A Generalisation of the Delogne-Kåsa Method for Fitting Hyperspheres. Pacific Grove, California, November 2004. Thirty-Eighth Asiomar Conference on Signals, Systems and Computers.

[32] V.B. Zordan, A. Majkowaska, B. Chiu, M. Fast, “Dynamic Response for the Motion Capture Animation” *ACM Transaction on Graphics (SigGraph 2005)* 24(3) (July 2005) : 686-701.

[33] A Kirk, J O Brien, and D Forsythe. “Skeletal parameter estimation from optical motion capture data,” Proceedings of IEEE conference on Computer Vision and Pattern Recognification (CVPR), 2005, pp. 782-788.