# The Process of Motion Capture: Dealing with the Data

Bobby Bodenheimer Chuck Rose
Microsoft Research

Seth Rosenthal John Pella
Interactive Media Production
Microsoft

**Abstract**

This paper presents a detailed description of the process of motion capture, whereby sensor information from a performer is transformed into an articulated, hierarchical rigid-body object. We describe the gathering of the data, the real-time construction of a virtual skeleton which a director can use for immediate feedback, and the offline processing which produces the articulated object. This offline process involves a robust statistical estimation of the size of the skeleton and an inverse kinematic optimization to produce the desired joint angle trajectories. Additionally, we discuss a variation on the inverse kinematic optimization which can be used when the standard approach does not yield satisfactory results for the special cases when joint angle consistency is desired between a group of motions. These procedures work well and have been used to produce motions for a number of commercial games.

## 1 Introduction

Motion capture is a popular process for generating human animation. In this paper we describe the process whereby magnetic sensor information is transformed into an animated human figure. Our emphasis is on the data collection and processing that goes into determining an animation of a hierarchical 3D rigid-body skeleton.

Human motion capture techniques may be categorized according to the intended degree of abstraction imposed between the human actor and the virtual counterpart. Highly abstracted applications of motion capture data, analogous to puppetry, are primarily concerned with motion character, and only secondarily concerned with fidelity or accuracy. Beyond an initial calibration to insure that the 'puppet' or animated figure can be adequately manipulated, the most significant calibration takes place in the minds of the animators or puppeteers who learn to manipulate the figure indirectly as a puppet, rather than as a direct representation of themselves. Such applications commonly require the development of a unique capture procedure to take into account the characteristics of the puppet and its range of motion, and often rely on a combination of multiple actors, multiple input devices and procedural effects. Furthermore they often depend on real-time electromagnetic or electro-mechanical motion capture devices.

At the other end of the spectrum, efforts to accurately represent human motion depend on limiting the degree of abstraction to a feasible minimum. These projects typically attempt to approximate human motion on a rigid-body model with a limited number of rotational degrees of freedom. This work

Authors' address: One Microsoft Way, Redmond, WA 98052, USA. Email: {bobbyb, chuckr, sethr, johnpe}@microsoft.com

is not restricted to real-time systems, and is often conducted with non-real-time techniques such as optical tracking as well as with electromagnetic and electro-mechanical systems. It requires that close attention be paid to actual limb lengths, offsets from sensors on the surface of the body to the skeleton, error introduced by surface deformation relative to the skeleton, and careful calibration of translational and rotational offsets to a known reference posture.

Motion capture systems are commercially available, and the two main types of systems are optical and magnetic. At the present time neither has a clear advantage over the other, but magnetic systems are significantly cheaper. Considerable literature exists on using and editing motion capture data in animation (e.g., [1, 15, 3, 13, 14]). Motion capture for use in animation has been surveyed in [10], and various descriptions of the end product of its use have appeared (see, for example, [9, 6]) but beyond descriptions of the various algorithms for inverse kinematics such as [2, 16], there has been little attention given to processing the data for use by inverse kinematics routines. The work by Molet *et al.* [12] gives an alternative technique to inverse kinematics for going from sensors on an actor to an animated articulated figure. The goal of producing an articulated rigid body is critical if additional motions which depend on dynamics are to be added, either from dynamical simulation (e.g., [7]) or spacetime constraints (e.g., [14]); additionally, accurate motion analysis is important to the biomechanics community.

The present work deals with the data processing discussed in [12] but gives a detailed presentation of the processing techniques needed for a system which uses inverse kinematics as the base routine for producing the articulated figure. Inverse kinematic techniques are used because they have potential to avoid rotational error propagation which may result in unacceptable positions of end effectors when, for example, there is interaction with props. In the first part of our paper, we present the basic motion capture process, including sensor attachment and derivation and inference of rotational degrees of freedom (DOF). Note that this phase is accomplished in real time, so that a director can view the basic quality of the captured motion. Next we determine the skeleton lengths from the recorded data, and generate an inverse kinematic solution. In particular, we discuss our approach to the problems of noisy sensor data caused by a limited number of sensors, sensor slip, and sensor noise. We note that an advantage of our technique over many commercial systems is that it generates data which can be easily sub-sampled. Finally, we present additional steps which can be taken when our inverse kinematics optimization step fails to find a realistic or consistent solution between similar motions in a motion capture session.

## 2 Basic Motion Capture Process

Our motion capture data is generated from an Ascension MotionStar system and is input directly into a 3D modelling and animation program, Softimage, at capture time. Data is sampled at up to 144Hz. This high sampling rate is advisable when fast motions, such as sports motions, are captured; using slower sampling rates for such motions can often produce problems in the inverse kinematics phase, since there is less frame-to-frame coherence. Actors are suited using from 13 to 18 six DOF sensors. The typical location for the sensors of an actor is shown in Fig. 1. Our motion capture method is designed to take advantage of (a) the real-time capability of the electromagnetic capture system which allows for careful initial calibration and periodic re-calibration of sensor offsets to the virtual skeleton

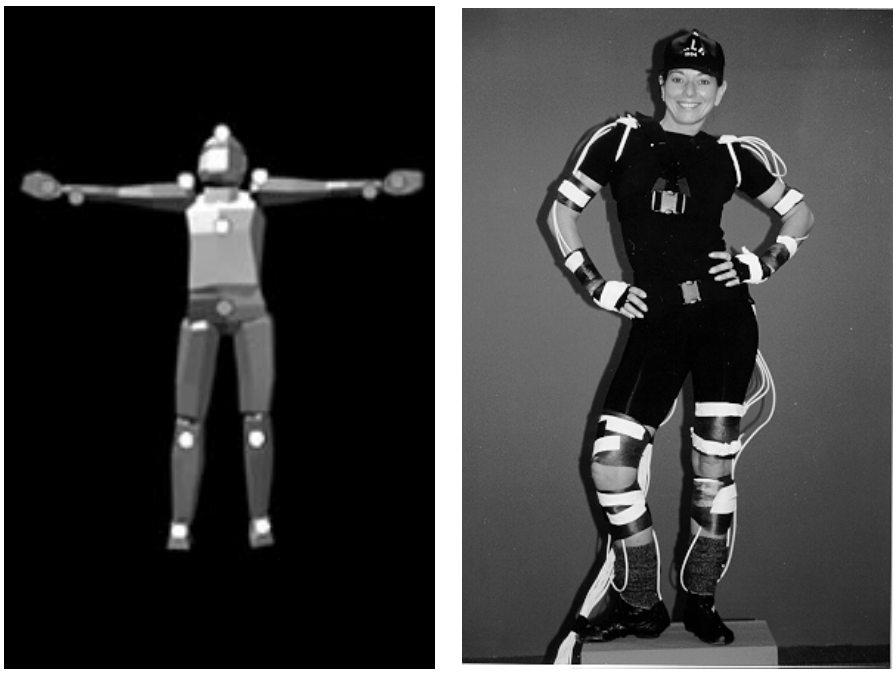

Figure 1: Typical placement of sensors on an actor. On the left is a 13 sensor configuration; the gray dots show sensors on the back of the body. On the right is an 18 sensor configuration.

(i.e., the non-kinematically constrained rotation and translation data), (b) animation tools in Softimage which allow fine control over secondary structures used as a source of derived or inferred data, and (c) the ability of statistical analysis and inverse kinematics to discard gross errors and outlying data, and to fit a hierarchical rigid body with a reduced set of DOFs to the data.

## 2.1 Sensor Placement

Our typical capture configuration relies primarily on the pelvis, forearms, head, and lower legs; for each of these, six DOFs are captured. These body segments are chosen for the degree to which they define the position and posture of the figure and for their comparative advantages as anchor points for the sensors. The data sampled for these segments are considered *primary data*, and are not processed beyond translating their six DOFs to their respective rotation points. Data for additional body segments are considered secondary, and are inferred from the primary data. In particular, a 3D virtual skeleton is constructed that provides translational and rotational constraints enabling us to conveniently infer such things as the rotation of a virtual limb about its longitudinal axis, based on the orientation of a dependent limb.

### 2.1.1 Primary Data

The forearms and lower legs provide superior surfaces for immobilizing sensors relative to the skeleton. For the lower legs, the flat surface of the tibia just below the patella is used. For the forearms, the top surface of the forearm, directly behind the wrist and spanning the radius and ulna is used. The forearm is assumed to behave as a rigid body with its longitudinal axis of rotation slightly offset from the center towards the radius.

The position of the hip joints and the base of the spine are measured as offsets from a single sensor on the pelvis. The pelvic sensor is typically placed on the back of the pelvis near the top of the sacrum. The head is represented by data from a single sensor secured to a tight-fitting hat or helmet.

Jointed kinematic chains for the arms and legs are avoided at this stage for two reasons. First, capturing the global position and orientation of the lower legs and forearms localizes error and preserves primary data for the analysis and global optimization. This precludes any alteration of the orientation of the lower limbs due to migration of other sensors, as would occur if jointed kinematic structures were used in the initial skeleton (note that if this were done, the inverse kinematics would be enforced frame by frame by the modelling program). Second, at capture-time, visual cues that indicate deteriorating sensor calibration, such as excessive separation at the knee, are more valuable than real-time inverse kinematic solutions.

### 2.1.2 Secondary Data

Secondary data is inferred by exploiting Softimage's animation capabilities to enforce translational, rotational, directional, and up vector constraints. The up vector is commonly used to orient a virtual camera about the view direction. An up vector constraint provides a convenient method for constraining the rotation of a virtual limb about its longitudinal axis based on the orientation of a dependent limb. The up vector constraint makes use of a user-specified point to determine the resolution plane of a kinematic chain. These constraints are used to infer the rotational DOFs for the upper arms and legs, the torso, and in some cases, the neck.

The thorax is a logical candidate for treatment as a source of primary data. However, at present, the main application for our data is for real-time rendered 3D games featuring animated low-polygon humanoid characters. These characters typically represent the torso as a single rigid body with fixed shoulders. Our approach to providing data for this type of application is to infer the torso DOFs from a single segment constrained to the virtual pelvis and to the virtual nape of the neck (which is measured as an offset from a sensor on the upper back). For applications which can use more DOFs, a collarbone is added to aid in such things as shoulder shrug. The longitudinal rotation of the torso is inferred from an up vector constraint applied to the sensor on the upper back.

The upper legs and arms are difficult to capture directly with electromagnetic sensors, due to the absence of stable anchor locations. With adequate care, sensors on these body segments can be initially calibrated, but are particularly susceptible to sensor migration as a result of strenuous motions. The virtual femur is constrained by its root to the virtual hip joint and by its effector to the proximal end of the virtual tibia. The rotation of the virtual femur about its longitudinal axis is inferred from an up vector constraint applied to a contrived offset from the sensor on the lower leg. The small degree of elasticity in the virtual knee that results from the natural elongation of the actual knee in the course of

flexion and any accumulated error from sources such as imperfect calibration of offsets or uncontrolled sensor migration can be removed with conventional inverse kinematics techniques, or in the global optimization phase.

The upper arm or humerus poses a more difficult challenge. Accurately representing the complex motion of the shoulder requires an impractically large number of DOFs and is complicated by the degree to which motion of the skeletal structure of the many components of the shoulder bears little relation to the motion of the surface of the body [11]. Two techniques can be used for estimating the position of the shoulder with electromagnetic sensors (other techniques are possible with optical tracking methods). First a sensor can be imperfectly immobilized to the top or back of the scapula and the position of the virtual shoulder joint can be estimated as an offset from this sensor's data. This method can be improved marginally by assuming that the direction of this estimated point from the virtual elbow is more accurate than the distance from the virtual elbow to this point. This assumption follows from observations that the direction of translational error of the shoulder sensor tends to be parallel to the longitudinal axis of the humerus. Given this assumption, the humerus can be represented by a single bone constrained at its root to the virtual elbow joint and constrained by its effector to the virtual shoulder. The proximal end of this bone is then assumed to represent the shoulder joint.

Second, the position of the shoulder joint can be estimated as an offset from a sensor on the upper arm. This method is difficult in practice due to the difficulty of immobilizing a sensor relative to the humerus. Even mildly strenuous motions can generate unacceptable variations between the DOFs recorded at the surface of the upper arm, and the actual position and orientation of the humerus. This arrangement is often acceptable for real-time puppetry-type applications but is inadequate for more exacting motion tracking applications.

As an example, the model for the arm is shown in Fig. 2. In this figure, (A) is the chain representing the upper arm, and (B) is the chain representing the lower arm. The position of the distal end of (A) is determined by the proximal end of (B). The longitudinal rotation of the upper arm needs to be inferred; (C) is the axis that controls the longitudinal rotation. (D) is an arbitrary point defined as an offset from the lower arm on a line projected back along the longitudinal axis of the lower arm. (K) is the plane defined by the root and effector of (A) and the point (D). As the effector of (B) moves through the trajectory (H), the point (D) moves through the trajectory (G). The up vector constraint applied to (A) forces (C) to lie in the resolution plane (K), thus determining the longitudinal rotation of (A). As (D) approaches the longitudinal axis of (A) the inferred rotation becomes unstable. This instability is addressed by carefully choosing the offset (D), by substituting estimated positions for (D) based on previous or subsequent frames, and, in some cases, by manually animating (D).

Finally, hand and foot motion is represented by rotational degrees of freedom from their respective sensors. Hands and feet are projected from wrists and ankles in a forward kinematic fashion. For example, the hands are not articulated but are mittens attached to the wrist.

## 2.2 Measuring and Building the Skeleton

Our production concerns dictate that the process of preparing an actor for a capture session, and building and calibrating a virtual skeleton be as convenient as possible. Our method largely automates the process of measuring rotational offsets, and requires a comparatively small set of manual adjustments for the final calibration. It does, however, require systematic hand measurements of translational off-

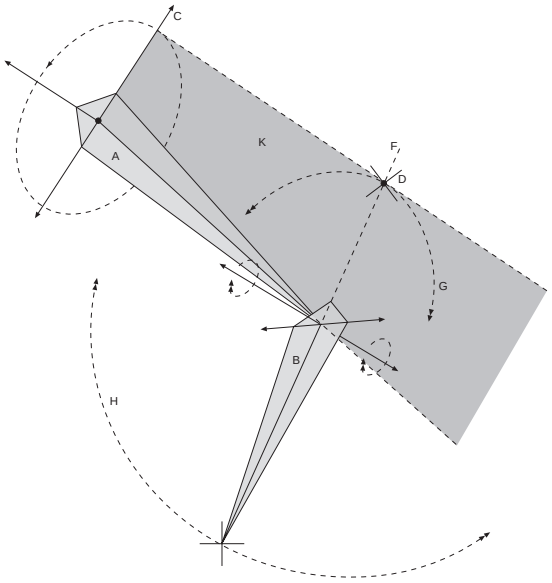


Figure 2: Upper arm model.

sets for all sensors for which translation data is used. The need for such measurements can be reduced by relying on methods based solely on the rotations of sensors secured to each body segment, and a single global translation [12]. However, the tendency of rotation-based techniques to propagate error can make them unwieldy for tracking motions that rely on the precise placement of the hands and feet, such as self-referential motions and motions that depend on extensive interaction with props.

Prior to securing the sensors, the actor's limbs are carefully measured. After the sensors are in place, their translational offsets are measured according to a coordinate system based on an arbitrary reference posture or "zero position." All measurements are rounded to the nearest 0.25 inches but are assumed to be somewhat less accurate. A skeletal model based on the measured limb lengths and offsets, and posed in the zero position is then generated programmatically.

This model maps the data from each sensor to that sensor's corresponding "null model." A null model is a node in the virtual skeleton which has no geometry. Null models are used to introduce an offset between the motion of two linked objects. For each capture sensor there is a null-model that holds its rotational and translational offset to the virtual skeleton. The translational offsets are assumed to be approximately correct. The rotational offsets are arbitrary and are assumed to be wrong, as the sensors are oriented on the body according to practical concerns such as cable management and sensor stability. Before a hierarchical relationship is established between the captured input and the joint centers, a single keyframe of rotation data is recorded with the actor standing in the zero position thus setting the offset to the frame of captured data.

Fine calibration is necessary to account for any error in the measurements of the limbs and the translational offsets, and to correct for the degree to which the actor cannot perfectly assume the theoretical zero position of the skeleton. This fine calibration is accomplished by manually adjusting the translations and rotations of the null model in an interactive calibration mode, that allows manipulation

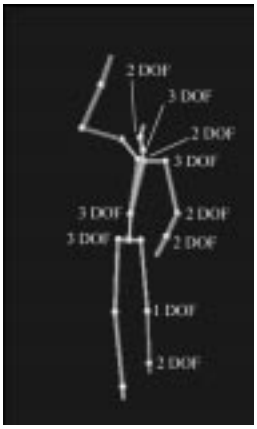


Figure 3: Articulated body model illustrating degrees of freedom.

of scene elements while the capture device and drivers are running. A simple set of routine motions is generally sufficient to identify any necessary refinements to the calibration; the arms may not line up, the hands may not come together, the legs may not be parallel, and so on. In practice, the fine calibration is primarily confined to the adjustments to the offsets for six sensors–those on the lower legs, the lower arms, the pelvis, and the chest. The resulting skeleton closely approximates the actor's motions and tends to localize error caused by sensor migration.

At capture time, data is recorded for the sensors and used to drive the virtual skeleton. The next stage is the optimization step. The data used in this step is not the sensor data, but data which represents the translation and rotation for each joint.

# 3 Optimization

Given a set of data based on the virtual skeleton described above, our goal is to construct an articulated, hierarchical rigid-body model. The model to which we will fit length and rotational data is shown in Fig. 3 and contains 38 joint degrees of freedom and six degrees of freedom at the root, located in the center of the pelvis, for positioning and orienting the entire figure. Our first task is to extract the best limb lengths from the motion capture data. Once the scale of the segments is determined, an inverse kinematics solution is calculated to determine the joint angles for the figure. Our inverse kinematics routine uses penalty functions to constrain the joint angles to approximate a human's range of motion.

## 3.1 Optimizing the skeleton

As mentioned previously, motion capture data is noisy and often contains gross errors. The source of the noise is primarily the magnetic sensors themselves, although we note that in our experience optical data is as noisy. We determine the size of the skeleton by determining the distances between the translated joint locations over a motion or repertoire of motions. Using the simple arithmetic mean to compute these distances results in answers unduly distorted by the gross errors, and editing the data

by hand to remove outliers is impractical. As an example, gross errors in fast motions such as throwing may, for a frame or two, give a distance between the elbow and wrist of over 3 meters. Thus a robust statistical procedure which can minimize or reject the influence of these outliers is employed. We have found that a one-step $M$-estimator [5] works well. This estimator has the form

$$T_n = T_n^{(0)} + S_n^{(0)} \frac{\sum_{i=1}^{n} \psi\left((x_i - T_n^{(0)})/S_n^{(0)}\right)}{\sum_{i=1}^{n} \psi'\left((x_i - T_n^{(0)})/S_n^{(0)}\right)}$$

where $n$ is the size of the data set, $x_i$ is a data point, $T_n$ is the estimated statistic of location, $T_n^{(0)}$ and $S_n^{(0)}$ are initial estimates of the location and scale, and $\psi$ is estimator function. In our work we use the Huber estimator

$$\psi_b = x \cdot \min\left(1, \frac{b}{|x|}\right)$$

where $b$ is the cutoff point and is determined by the median deviation of the data. Our initial estimates of location and scale are the recommended ones for this type of estimator (see [5]), and are given by

$$T_n^{(0)} = \text{median}(x_i)$$
$$S_n^{(0)} = 1.483\text{median}_i\left\{|x_i - \text{median}_j(x_j)|\right\}.$$

This estimator insures that any error, no matter how gross, has only a fixed impact on the data set.

In this phase, an outlier is defined as any data point beyond the cutoff point $b$, which is typically twice the median deviation (defined above for $S_n^{(0)}$). When an outlier is found, it is tagged so that during the inverse kinematics phase, data from that frame and for those joints is automatically ignored. As an example illustrating the importance of this, a short motion of a walk consisting of 508 frames of motion has 6 frames of clearly gross error. Editing these frames out by hand and computing the collarbone to shoulder length gives 13.3 cm. Our $M$-estimator gives 13.2 cm, but using the arithmetic means gives a length of 14.1 cm, a 6% difference. Errors of this magnitude will frequently cause difficulties in the inverse kinematics procedure, thus marking them is a valuable tool. Since the statistical estimator we are using is of the "one-step" variety, the limb length calculation is not computationally expensive.

## 3.2 Inverse Kinematics

Once the hierarchical model has been determined using robust statistical analysis of the data, each frame of data must be analyzed to produce a set of joint angles. Additionally, these joint angles should form reasonable piecewise-linear DOF curves which can be sampled at any time, not just the original frame times. Piecewise-linearity is extremely useful if the data is to be sub-sampled. In contrast, many commercial data sets often contain discontinuities in the rotational data from frame to frame, which make sub-sampling impossible. Our primary and secondary data yield information about many areas of the body, giving us a highly constrained kinematic problem. This problem can be solved using a non-linear optimization technique, which seeks to minimize the deviation between the recorded data and the hierarchical model. We use a modification to the technique presented in Zhao [16].

The fitness function we are minimizing is defined as

$$\begin{aligned} F(\Theta) = \sum_{j \in J} & w_p \left( P_j - \tilde{P}_j \right)^2 + \\ & w_{O_0} \left( O_{0,j} - \tilde{O}_{0,j} \right)^2 + \\ & w_{O_1} \left( O_{1,j} - \tilde{O}_{1,j} \right)^2 + \\ & w_c C_j^2 \end{aligned}$$

where $\Theta$ is the set of joint angles for the set of joints $J$. $P_j$ is the position of the $j$th joint given $\Theta$ and $\tilde{P}_j$ is the recorded joint position from the capture phase. $O_{0,j}$ and $O_{1,j}$ are two vectors defining the orientation of the joint with $\tilde{O}_{0,j}$ and $\tilde{O}_{1,j}$ being the recorded orientations. Two vectors are used because, together with their cross product, they will form a right-handed coordinate system. The quantities $w_p$, $w_{O_0}$, and $w_{O_1}$ are scalar weights that can be tuned to achieve better results. Additionally, we employ a joint angle constraint term, $C_j$, in the form of a penalty function. Joint angle constraints for humans have been measured and can be found in the biomechanical literature [8].

The quasi-Newton BFGS optimization technique [4] is used to solve the system and uses the gradient of the fitness function, given by

$$\begin{aligned} \frac{\partial F_j}{\partial \Theta_I} = & 2 w_p (P_j - \tilde{P}_j)(u_j \times d_{ji}) + \\ & 2 w_{O_0} (O_{0,j} - \tilde{O}_{0,j})(u_j \times O_{0,j}) + \\ & 2 w_{O_1} (O_{1,j} - \tilde{O}_{1,j})(u_j \times O_{1,j}) + \\ & 2 w_c C_j \end{aligned}$$

to produce our DOF curves in the form of piecewise-linear functions. If the data is relatively non-noisy and the skeleton is well formed, this technique will work well. It can produce poor results if these conditions are not present. Robust statistics helps to insure these conditions by making the best skeleton and by marking data points which are considered outliers. Since a hierarchical description of a skeleton is a biological simplification and since non-linear optimization is hard, the analysis can still fall into an insufficient local minima if the starting guess for the optimization is far from the desired solution.

Our internal representation of rotations is as $XYZ$ Euler angles. This representation was originally used because it provided simplicity in our code. However, Euler angles provide only a local parameterization of the group of rotations in $\mathbb{R}^3$, and thus have singularities. While our technique works well for many motions, it cannot be denied that the use of a global parameterization, such as quaternions, would be better.

We mitigate this problem in two ways. The first technique is simple: use the result of frame $i$ for the starting guess of frame $(i + 1)$. For many motions, this technique is perfectly valid. It suffers if the data and the skeleton are mismatched near where the skeleton goes through a singularity or where the data points are too far apart in time for a given motion's velocity. Additionally, it will suffer if it never finds a good solution for an initial frame. Over the shoulder reaching, fast motions, and motions

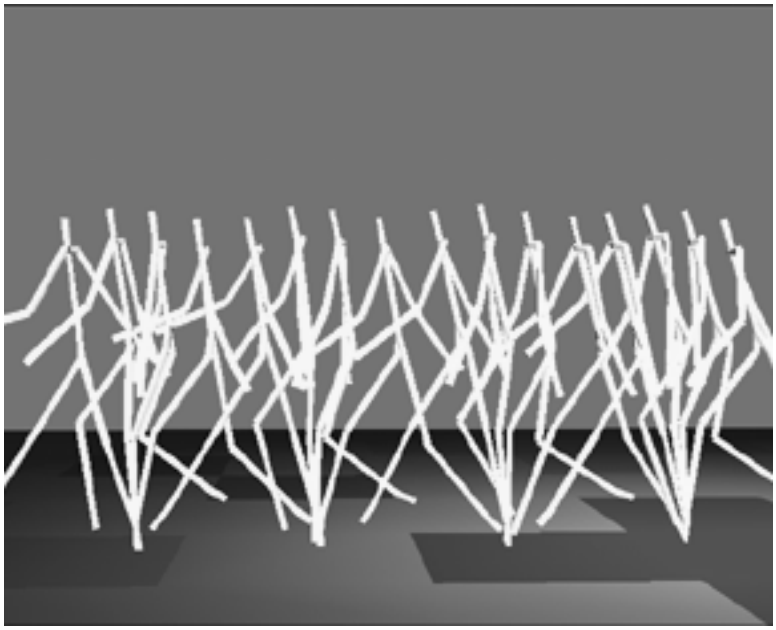

Figure 4: Walk motion.

where the arm is extended to its limit are examples. If this happens, the solution can jump over to another local minima and stay there. This behavior is not desirable.

Analyzing a walk motion of $6.7$ seconds duration at 30 frames per second required 306 seconds on a Pentium 133 machine with 4389 BFGS iterations for satisfactory convergence of the solution. A selected frame showing the fit of the skeleton (yellow) to the data (black) is shown in Fig. 10 (see Appendix). Notice that the fit is extremely good and shows only a slight discrepancy in the left arm. The resulting walk motion is shown in Fig. 4.

A further refinement of the motion capture analysis presented here is to use motions to bootstrap one another by providing good starting guesses to the BFGS optimization. The assumption for this technique is that many motions of similar structure are to be analyzed, and that motions of a similar structure will have similar joint angle curves. Such a data set might include reaches, runs, walks, etc. These sets are likely to be a part of any motion capture session.

Assume there is a motion $M$, a set of DOF curves, for a motion of type $T$. If we have a motion capture dataset for another motion of this type, the joint angles for this motion will be similar to those for motion $M$. The main difference between the solution for the new desired motion $\tilde{M}$ and $M$ will be a time warping to account for differences in the relative timing between $M$ and the captured data. Thus a scaling in time on the data sets is needed. We mark a set of correspondence times, key times, in $M$ and in the data set. We time-warp $M$ and then use that as the starting guess for the inverse kinematics optimization described earlier.

Thus, this technique will not propagate errors, whereas in the previous technique a bad starting guess may result in a bad solution, which can propagate from frame to frame. As a result, similar motions will make similar use of their joints when analyzed. This similar joint use is important when

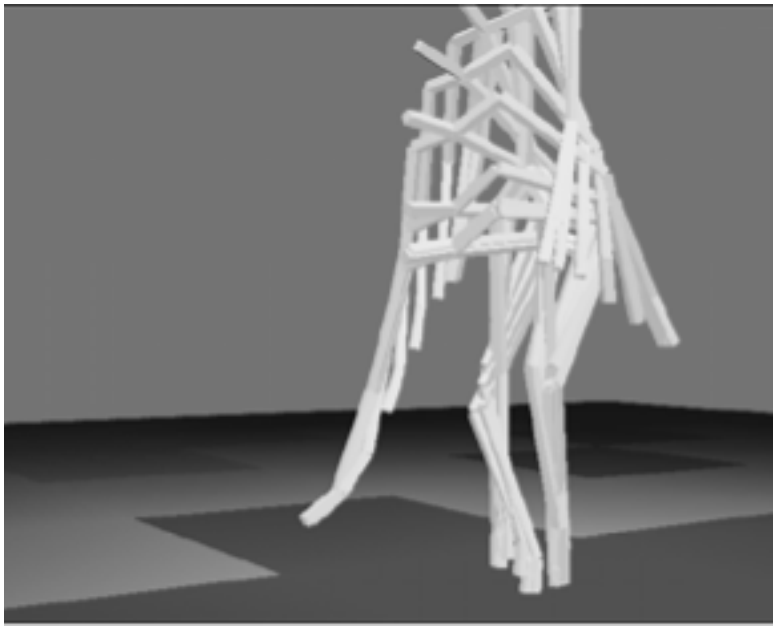

Figure 5: A low reaching motion.

these motions are later used in techniques like those presented in, for example, [15]. Note that this technique requires operator intervention to mark the key times, and thus it is only employed for groups of datasets where the aforementioned method did not work.

The reach motion shown in Fig. 5 was analyzed using the same technique as the walk. Its duration is 3.8 seconds and required 1278 BFGS iterations and a total of 37.08 seconds to analyze. Unfortunately, due to noise, inadequacies in the skeletal model, and the joint angle constraints model, this method is not powerful enough to insure that it makes similar use of the joint angles as another reaching motion previously analyzed and shown in Fig. 6. Compare the shoulder DOFs shown in Fig. 7 versus those of Fig. 8. These differences would represent a fundamental obstacle if, for example, we tried to interpolate between these motions to obtain a reach motion of a different height.

Using the medium reach as a reference motion, and a time warp to align it as the starting guess for the low reach motion capture data, we can obtain a more consistent use of shoulder angles, as shown in Fig. 9. This analysis took 3295 iterations and 232 seconds. Notice that, up to a time warp, this set of shoulder angles is very similar to those for the medium reach shown in Fig. 8.

# 4 Conclusion

We have presented a detailed account for taking human performance sensor data and producing animations of articulated rigid bodies. Our technique involves using geometric modelling to translate rotation data to joint centers, a robust statistical procedure to determine the optimal skeleton size, and an inverse kinematics optimization to produce desired joint angle trajectories. We presented a variant

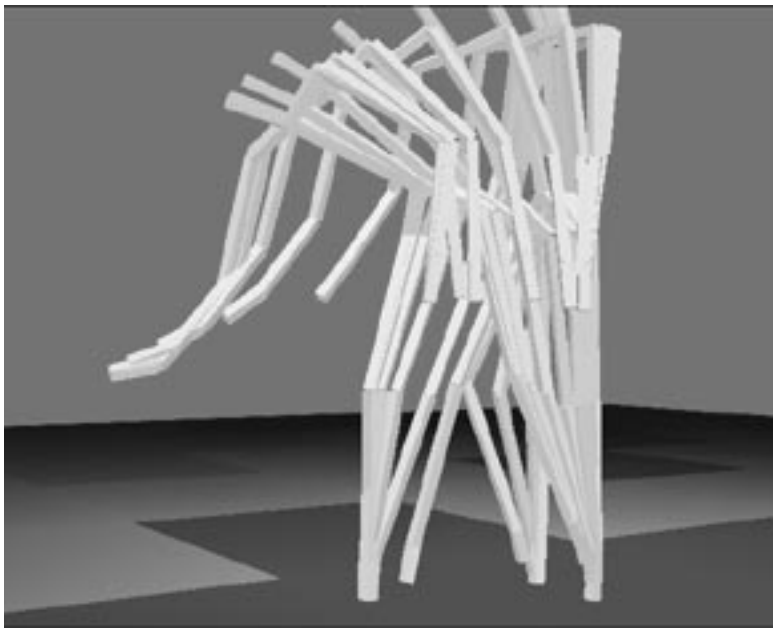

Figure 6: A previously analyzed medium reaching motion.

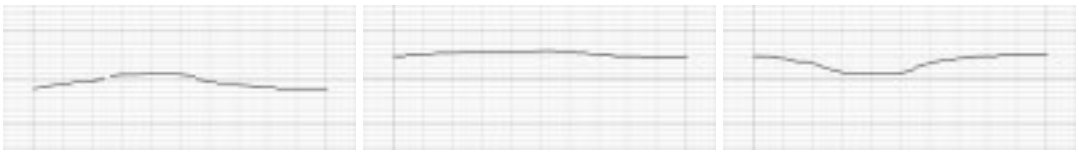

Figure 7: Shoulder DOFs for the low reaching motion.

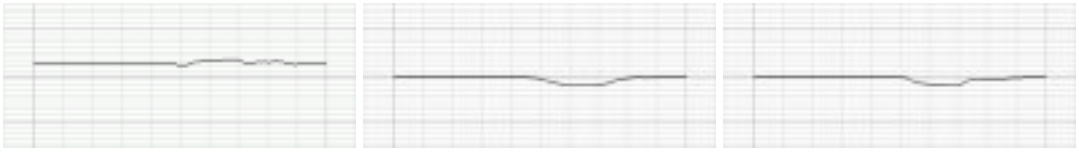

Figure 8: Shoulder DOFs for the medium reaching motion.

.

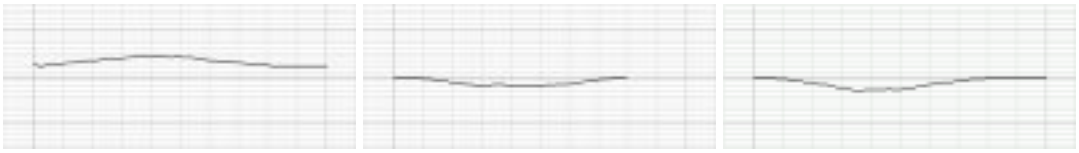

Figure 9: Shoulder DOFs for the low reaching motion, using the medium reach as a reference guess.

of the inverse kinematics optimization to be used when an initial approach has not yielded satisfactory results, for the special cases when joint angle consistency is desired between several motions. This procedure has been used to produce a number of motions for various commercial games and has been found to work well. Fig. 11 (see Appendix) shows the motion capture process at the various stages of processing: the first figure shows the capture phase, where an actor is interacting with a prop (a model of an Indy car); the second shows the articulated rigid body skeleton obtained after inverse kinematics optimization; the third shows this skeleton again repositioned with the prop; the fourth shows a full rendering of the character and the prop. Notice that the inverse kinematics optimization has not changed the location of the end effectors significantly, since they are still able to interact with the prop. Finally, we remark that if any animator intervention is required, this intervention will occur in a step between that of the third figure and the fourth.

The least satisfactory aspect of the motion capture method described here is the gross over-simplification of the motion of the spine. This aspect is not an inherent limitation of the method and can be improved by capturing or inferring data for the abdomen, thorax and neck as distinct segments. The need to obtain reasonably accurate translational offsets for sensors and to carefully calibrate the virtual skeleton has required the development of an efficient systematic approach to measuring the skeleton and the sensor positions. This approach allows us to successfully capture complex motions involving good registration between the virtual actor and the virtual representations of props in the capture space. However, it requires a fairly high degree of production preparedness for its efficient execution. This process is a likely candidate for a more general and robust solution.

Changing our internal representation of rotations from Euler angles to quaternions would likely help our inverse kinematics processing, particularly for fast sports motions such as throwing. Better models of the human body, including more accurate skeletons and more realistic joint angle constraints will yield better analysis of motion capture data. In the statistical analysis of the data, it is likely that a "redescending" estimator, which has the ability to reject outliers outright, would produce better results, and this will be explored further. Optimization techniques such as active set optimization [4], rather than penalty-based ones, may give better adherence to joint angle constraints. Additionally, methods for self-calibration and other avenues which reduce the operator workload and shorten the time to produce an animation will be a major avenue of research.

## Acknowledgements

We are greatly indebted to Michael Cohen and Jana Wilcoxen for assistance and suggestions throughout this project. Thanks also to David Thiel and Agnieszka Morgan for help with the video and image production, respectively. Larry Frame, who designed and built Microsoft's motion capture facility, made his staff and resources readily available for our work. Hank Meuret also made numerous contributions. The IMP 3D graphics group was always eager to help, and the Simulation Games group generously allowed us to use their data. Finally, the second author thanks Princeton University, where he was a graduate student for much of this project. MotionStar is a trademark of Ascension Technology Corporation, and Softimage is a trademark of Softimage, inc., a wholly owned subsidiary of Microsoft.

## References

[1] BADLER, N. I., HOLLICK, M. J., AND GRANIERI, J. P. Real-time control of a virtual human using minimal sensors. *Presence 2*, 1 (1993), 82–86.

[2] BADLER, N. I., PHILLIPS, C. B., AND WEBBER, B. L. *Simulating Humans: Computer Graphics Animation and Control*. Oxford University Press, Oxford, UK, 1993.

[3] BRUDERLIN, A., AND WILLIAMS, L. Motion signal processing. In *Computer Graphics* (Aug. 1995), pp. 97–104. Proceedings of SIGGRAPH 95.

[4] GILL, P. E., MURRAY, W., AND WRIGHT, M. H. *Practical Optimization*. Academic Press, 1981.

[5] HAMPEL, F. R., RONCHETTI, E. M., ROUSSEEUW, P. J., AND STAHEL, W. A. *Robust Statistics: The Approach Based on Influence Functions*. John H. Wiley, New York, 1986.

[6] HARS, A. Masters of motion. *Computer Graphics World* (Oct. 1996), 26–34.

[7] HODGINS, J. K., WOOTEN, W. L., BROGAN, D. C., AND O'BRIEN, J. F. Animating human athletics. In *Computer Graphics* (Aug. 1995), pp. 71–78. Proceedings of SIGGRAPH 95.

[8] HOUY, D. R. Range of motion in college males. Presented at the Conference of the Human Factors Society, Santa Monica, CA, 1983.

[9] MAESTRI, G. Capturing motion. *Computer Graphics World* (1995), 47–51.

[10] MAIOCCHI, R. 3-D character animation using motion capture. In *Interactive Computer Animation*, N. Magnetat-Thalmann and D. Thalmann, Eds. Prentice-Hall, London, 1996, pp. 10–39.

[11] MAUREL, W., THALMANN, D., HOFFMEYER, P., BEYLOT, P., GINGINS, P., KALRA, P., AND THALMANN, N. M. A biomechanical musculoskeletal model of human upper limb for dynamic simulation. In *Computer Animation and Simulation '96* (Aug. 1996), R. Boulic and G. Hégron, Eds., pp. 121–136.

[12] MOLET, T., BOULIC, R., AND THALMANN, D. A real time anatomical converter for human motion capture. In *Computer Animation and Simulation '96* (Aug. 1996), R. Boulic and G. Hégron, Eds., pp. 79–94.

[13] PERLIN, K. Real time responsive animation with personality. *IEEE Transactions on Visualization and Computer Graphics 1*, 1 (Mar. 1995), 5–15.

[14] ROSE, C. F., GUENTER, B., BODENHEIMER, B., AND COHEN, M. Efficient generation of motion transitions using spacetime constraints. In *Computer Graphics* (Aug. 1996), pp. 147–154. Proceedings of SIGGRAPH 96.

[15] WITKIN, A., AND POPOVIĆ, Z. Motion warping. In *Computer Graphics* (Aug. 1995), pp. 105–108. Proceedings of SIGGRAPH 95.

[16] ZHAO, J., AND BADLER, N. I. Inverse kinematics positioning using non-linear programming for highly articulated figures. *ACM Trans. Gr. 13*, 4 (Oct. 1994), 313–336.

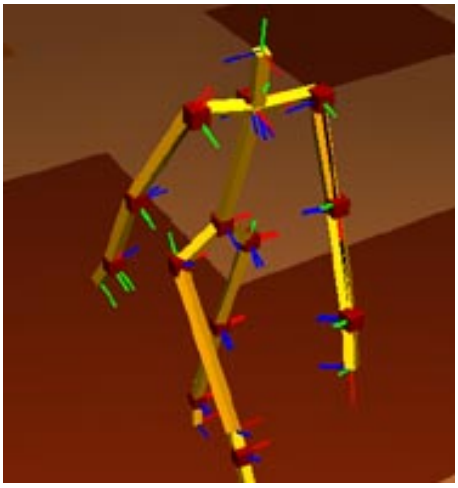

Figure 10: Fit between skeleton (yellow) and data (black).

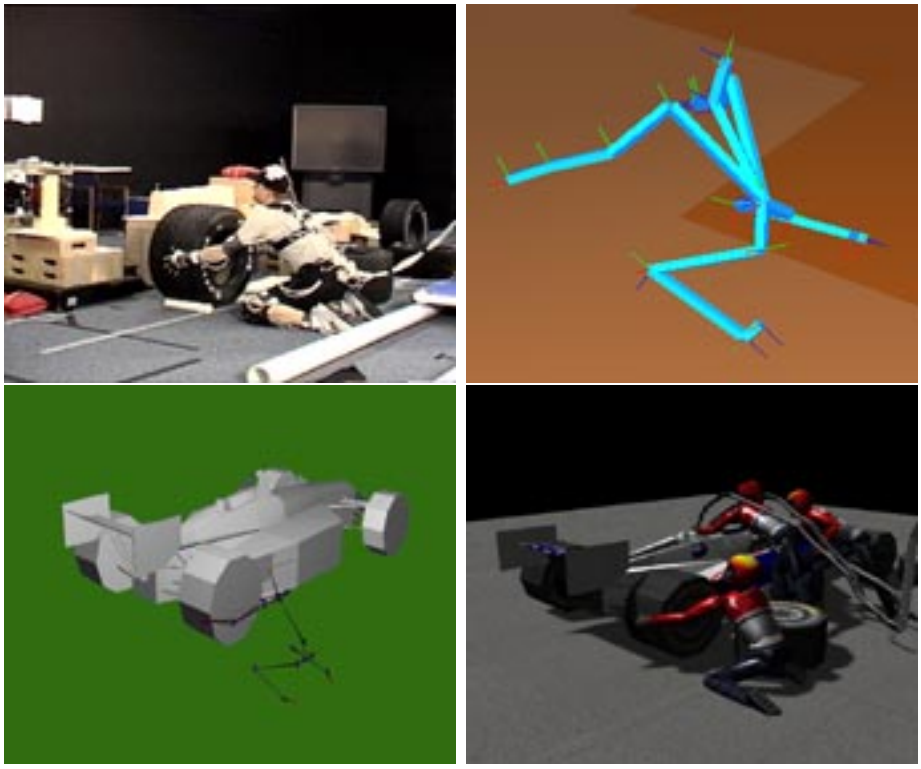

Figure 11: Various phases of the motion capture process.