Motion capture-driven simulations that hit and react

Victor B. Zordan
College of Computing and GVU Center, Georgia Institute of Technology
Jessica K. Hodgins
School of Computer Science, Carnegie Mellon University

Abstract

Controllable, reactive human motion is essential in many video games and training environments. Characters in these applications often perform tasks based on modified motion data, but response to unpredicted events is also important in order to maintain realism. We approach the problem of motion synthesis for interactive, humanlike characters by combining dynamic simulation and human motion capture data. Our control systems use trajectory tracking to follow motion capture data and a balance controller to keep the character upright while modifying sequences from a small motion library to accomplish specified tasks, such as throwing punches or swinging a racket. The system reacts to forces computed from a physical collision model by changing stiffness and damping terms. The freestanding, simulated humans respond automatically to impacts and smoothly return to tracking. We compare the resulting motion with video and recorded human data.

CR Categories: I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism—Animation

Keywords: motion capture and human body simulation, physically based animation, virtual environments, computer games

1 Introduction

Motion capture-driven characters are often used to populate synthetic environments in electronic games, entertainment, and educational applications. Character interaction affects the believability of these environments. For example, football players should form a convincing pile-up and a boxer must throw punches that land realistically. Modifying motion capture data automatically for all such general interactions is not yet a solved problem. We present motion capture-driven simulations that respond to a variety of unexpected impacts in the upper body. The physically simulated, freestanding humans accomplish tasks such as boxing and table tennis based on examples from motion libraries. Through this research, we make progress towards more general interaction for motion capture-driven characters.

Physical models allow interaction through forces and impacts but controllers for such physically based humans are difficult to construct. Further, the resulting motions often lack important features of natural human motion. In this work, we avoid these problems by creating simulations that match human data while they interact with each other and their environment using a collision model. The simulations perform actions from a motion library of examples without a pre-existing control system for the specific behaviors. The system includes a reaction controller that allows the simulated humans to respond to impacts based on their dynamics.

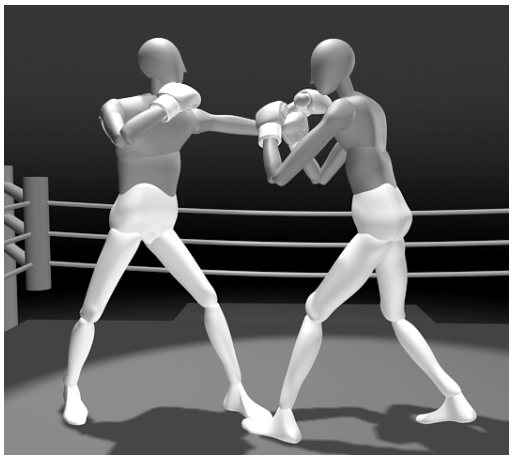

Figure 1: **Motion capture-driven simulations.**

Recently, a number of researchers have explored combining motion capture and dynamic simulation in order to retain the advantages of each while avoiding the disadvantages. Hodgins and Popović organized a SIGGRAPH course on this topic in 2000. The approaches covered there and elsewhere fall along a continuum between the two techniques with some methods relying more on the information embedded in the data and some relying more on the physical realism of the simulation. In this paper, we also combine motion capture and simulation, with a technique that lies closer to the simulation of the end of the spectrum.

By controlling a simulation with motion capture data, we hope to capture subtle, humanlike details from the data while adding responsiveness and interactivity with the dynamic simulation. Therefore motion capture-driven simulations must meet two, sometimes conflicting, goals: remaining close enough to the motion capture data to keep the important characteristics of the original motion while deviating sufficiently to accomplish the given task. These important characteristics include the style such as a jab versus a hook in boxing. And the deviations allow for hitting to a particular location, reacting to collisions, and balancing.

We demonstrate the power of our approach with characters that hit and react under a variety of conditions including simulations that play table tennis and box. Full-body simulations are driven by motion data that has been modified with time-scaling, blending, and inverse kinematics to achieve specified tasks. A gain scheduler modulates the stiffness parameters of the controller between tracking and reacting so that the characters respond to collision forces

from impacts in the upper body. While the hits and reactions take place, balance is maintained through control torques computed according to feedback for the simulation's center of mass. We assess our technique by comparing synthesized motion with human motion both graphically and in side-by-side comparison.

2 Background

In this work, we build on research both in the editing of motion capture data and in constructing control systems for the simulation of human figures. Recently, researchers have also begun to explore the potential advantages of combining these two sets of techniques as we do.

2.1 Motion capture editing

Most of the research in motion capture has explored techniques for modifying data for a given scenario and for generalizing it for effective reuse. Straightforward interpolation with keyframes was suggested initially [Bruderlin and Williams 1995; Witkin and Popović 1995]. Gleicher improved edits and adapted motion to new characters by maintaining desired constraints such as contact with the environment while optimizing over an entire sequence [Gleicher 1998]. Lee and Shin describe a general editing approach using multi-level B-splines [Lee and Shin 1999]. Choi, Park, and Ko propose modifying data by minimizing velocity disturbances caused by edits [Choi et al. 1999]. Our system performs simple edits on motion capture data automatically based on a technique similar to Witkin and Popović [1995] and uses this edited motion as input to the controller.

Combining data sequences to create a new movement or parameterized behavior is another central problem in using motion capture data. Bruderlin and Williams combine two dissimilar motions by decomposing the motions into frequency bands and allowing user control over blending gains for the bands [Bruderlin and Williams 1995]. Unuma, Anjyo, and Takeuchi use a Fourier series expansion to interpolate and transition between motion sequences for walking, jumping, and running [Unuma et al. 1995]. Similarly, Rose, Cohen, and Bodenheimer interpolate between like data sets, such as happy and sad walking, but also allow non-periodic motions such as reaching [Rose et al. 1998]. They use radial basis functions and timewarping to align key events in their blended motion. Wiley and Hahn suggest resampling a number of motion examples through linear interpolation and time-scaling [Wiley and Hahn 1997]. We use this approach in table tennis to create a large pool of example swings in our motion library.

2.2 Physically based humans

Fully simulated dynamic characters offer the potential for physically realistic motion and interactions with other characters and objects via impact and response. Hodgins and her colleagues present hand-tuned, state-machine driven controllers for motions like running, gymnastics, and diving [Hodgins et al. 1995]. Laszlo, van de Panne, and Fiume used limit cycle control for physically realistic, periodic walking and running [Laszlo et al. 1996]. They began from an unstable open-loop controller for walking and perturbed it to find a stable cyclic movement. van de Panne and Lamouret use external forces to maintain the attitude of an unstable walking human while guiding the optimization process towards a stable solution [van de Panne and Lamouret 1995]. We found external forces a helpful mechanism as well in the preliminary stages of designing our balance control system.

Although most work has focused on the design of controllers for individual behaviors, two efforts have combined controllers to create more complex motions [Wooten and Hodgins 2000; Faloutsos et al. 2001a]. Wooten and Hodgins implemented four parameterized controllers that were concatenated to create gymnastic behaviors such as diving and flipping. Faloutsos, van de Panne, and Terzopoulos took this approach a step further by automatically combining primitive controllers based on each controller's pre-conditions for success. They also added recovery and reactions to falling to the repertoire of motor skills [Faloutsos et al. 2001b]. Our system strings together primitive motions with high-level state machines to create movements such as a boxer throwing a series of punches although combining behaviors is not the main focus of our work.

2.3 Combining motion capture and simulation

A number of techniques combine simulation and motion capture by using physical models to modify motion data. For example, Popović and Witkin produced a variety of modifications to running and jumping motion data using a low resolution physical model to constrain the search space and trajectory optimization to solve for modifications [Popović and Witkin 1999]. Pollard's work on "simple machines" uses a similar approach with a low degree-of-freedom model for editing running motions, foregoing constrained optimization for a faster solution [Pollard 1999].

Full-resolution forward and inverse dynamic models have been used to achieve both physical realism in motion editing and human detail in simulation. Rose and his colleagues find a minimum energy solution for transitions between motion sequences with an inverse dynamics model [Rose et al. 1996]. In previous work, we filter motion using a simulation to create physically plausible motion, including transitions, for upper-body behaviors [Zordan and Hodgins 1999]. Playter combines motion capture with dynamic simulation and a controller for running by using motion capture to drive joint angles during underconstrained sections of the behavior such as the flight phase [Playter 2000].

Two works are particularly similar to ours in that they also modify motion to include physically based reactions. Kokkevis, Metaxas, and Badler use a model reference controller and simulation to modify keyframed data based on gravity or external forces such as a large impulse that knocks over a soldier [Kokkevis et al. 1996]. Oshita and Makinouchi use a tracking controller and simulation to show a character responding to a mass being dropped on his back [Oshita and Makinouchi 2001]. Their approach actively controls a subset of the character's joints based on heuristics, actuating the shoulders for arm motion, the back for upper-body motion and the legs to control the pelvis acceleration for balance. Unlike these techniques, our work includes motion capture data and requires that the simulations actively follow the data in each joint as well as react to contact.

3 Motion capture-driven control

To control the motion capture-driven simulations, we combine a trajectory tracking controller with a process for selecting among and interpolating between sequences of human motion data. At each simulation timestep, the system determines a desired state for the simulation from the motion capture data and the tracking controller computes torques based on those desired values. The simulation is integrated forward in time, the state variables are updated, and the process repeats.

The simulations are rigid body models of human figures with the equations of motion computed using SDFast [Symbolic Dynamics Inc. 1990]. The degrees of freedom are shown in figure 2 and the simulation's root node (at the pelvis) is free to move and rotate in space. For collisions, a hierarchical detection and position/velocity penalty method identify and resolve contact between objects in the scene and the simulations' body parts. Impact forces are applied to the body of the contacted simulation in order to create reactions.

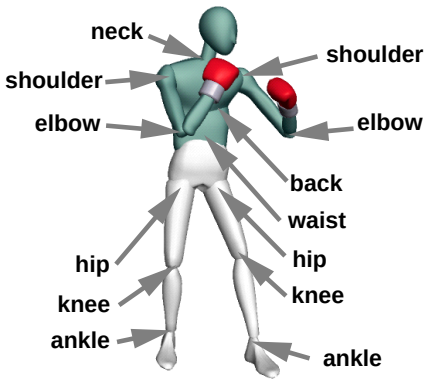

Figure 2: **Each of the joints indicated have three degrees of freedom. The table tennis character also includes wrist joints. The mass and inertial parameters of the simulations are computed based on their geometric models and human density measurements.**

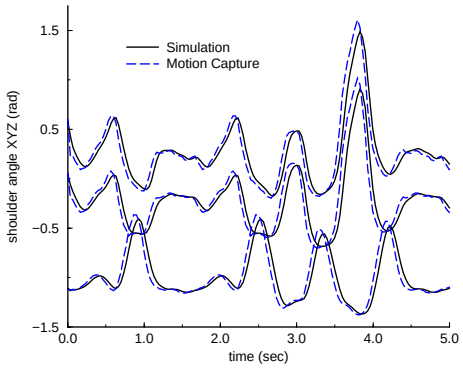

Figure 3: **A human was recorded performing a bob and weave. That data was input as the desired value of the tracking controller. The graph shows the resulting shoulder joint angles. By accounting for the inertial mass being moved and keeping the stiffnesses high, a trajectory-tracking controller follows the motion capture data closely introducing only a small time delay.**

The torque controller for each joint is a proportional-derivative servo:

$$\tau_t = k_t\,(\theta_d - \theta) - b_t\,(\dot{\theta}) \tag{1}$$

where θ and θ_d correspond to the actual and desired joint angles, $\dot{\theta}$ is the joint velocity, and k_t and b_t are the position and velocity gains. To make the tracking more precise, the system scales the gains according to the moment of inertia of the chain of bodies affected by that joint. For example, the shoulder gains are scaled by the moment of inertia of the upper arm, lower arm, and hand about the shoulder joint during each timestep. Similarly, the torque at the ankle, which affects the whole body when the foot is on the ground, is scaled by the inertia of the leg, pelvis, and upper body. This process also acts to reduce the number of tuned parameters so that stiffness over the entire body can be controlled with a single value, speeding up the tuning process tremendously. Figure 3 shows the performance of the tracking controller.

With this tracking control system in place, the simulation is able to follow motion capture data closely, however, deviations from the raw motion data are required in order for the characters to perform desired tasks and to react to contact (sections 4 and 5). In addition, without feedback control, the characters would eventually fall over. Combining tracking with balance control is described in section 6. Further implementation details and examples for each of these topics are included in the thesis correponding to this research [Zordan 2002].

4 Control for Hitting

When the motion capture-driven simulation performs a hitting action, such as throwing a punch in boxing or swinging a racket at a ball in table tennis, our approach is to modify the motion capture segment to control the important features of the action automatically and then track the resulting motion as described in the previous section. To complete tasks successfully, we control the end effector (the glove or racket) in two ways: changing the joint angles from the original motion capture data to create the desired position and orientation and time-scaling the motion capture sequence to control the speed. For continuous play, we also use a state machine to schedule hitting actions.

An inverse kinematics (IK) solver adjusts raw or synthetic motion segments to control the position and orientation of a hit. Our IK solver searches for the arm joint angles with a hybrid approach similar to that discussed by Lee and Shin [1999]. An adjusted offset computed from the IK solution is smoothly interpolated over a time interval of fixed length. This IK solver allows the character to perform a wider range of motions than exist in the raw recorded library. When the original motion capture library is sparse, like the dozen forehand and backhand swings we used in table tennis, synthetic examples fill in gaps and help the IK solver by keeping the required adjustments small. Table tennis swings were time-scaled to align the beginning, end, and hit time and interpolated to generate a denser library with 550 synthetic swings.

We make small adjustments in the speed of the end effector like those required for the precise control of a ball in table tennis by smoothly scaling the timestep of the motion capture data and driving the simulation with the *sped-up* or *slowed-down* data. This approach requires a pre-processing step that executes the simulation once for each motion sample in the library and records the speed of the end effector at the time of impact. The timestep used to index into the motion capture data is adjusted based on the ratio of the desired speed and the pre-recorded speed. The velocity of the end effector relative to the root, v_r, is

$$v_r(t) = \sum_{i=0}^{n} r_i \times \omega_i(t) \tag{2}$$

where r_i is the vector describing the limb, $\omega_i(t)$ is the angular velocity and n is the number of joints. In this case, a scale in each ω_i yields the same scale in the velocity v_r. If the simulation's root node is moving slowly, we can use this relationship to approximate the global speed of the end effector. A sinusoidal weighting smoothly adjusts the timestep so that it moves from the original, to the scaled timestep at the time of contact, and back again. Figure 4 shows a plot of the predicted and resulting speed for a table tennis swing as well as the performance of the hit controller as it aims for specific targets and figure 5 shows a plot for boxing punches being thrown to different locations.

The hit controller is managed automatically by a state machine to create behaviors that continuously box and play table tennis. The

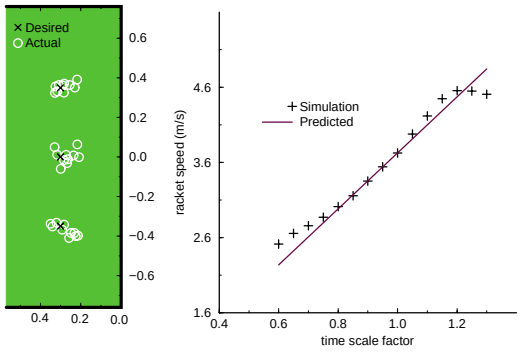

Figure 4: **The left image shows a top-down view of a regulation-size table tennis table with the net to the left (not shown) and results for three desired hit locations. The plot on the right shows a series of racket speeds at the time of contact for a timewarped swing and compares them with the predicted speed.**

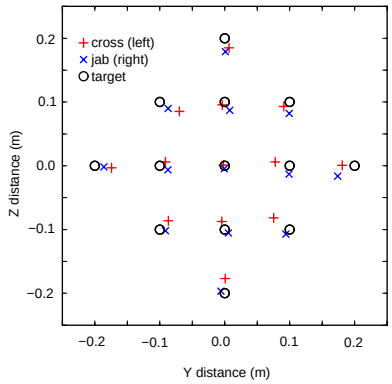

Figure 5: **Contact points for two styles of punches. The center represents the unmodified point of contact for the simulation. The remaining target locations appear on a 0.1 m grid relative to this point.**

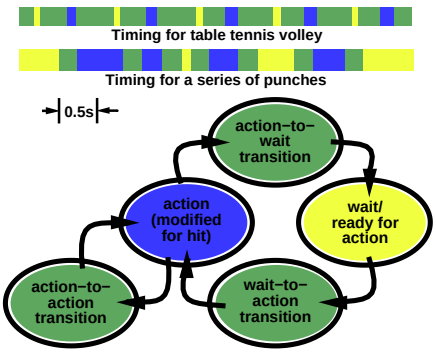

Figure 6: **State machines for punching and swinging a racket drive motion libraries consisting of 4 distinct punches and 2 blocks in the boxing library and 12 raw swings (converted to 560 synthesized swings) in table tennis. Timing color bars show the length of time the simulation stays in each state.**

state machine, shown in figure 6, queues the desired motion capture segments from the library and creates transitions when the end of a segment approaches or a user initiates a new activity. Transitions interpolate the end and beginning of the adjoining segments with a simple ease in/ease out ramp. When a hit is not taking place, a wait or ready action is scheduled. In continuous play, the state machine cycles by performing an action like a specific swing or punch, transitioning, waiting, and transitioning again to the next action as seen in the timing bars in the figure.

5 Reacting to Contact

During contact, the gains of the affected joints are reduced to allow the dynamics of the impact to influence the motion in a natural manner. While high stiffness parameters in the tracking controller follow the motion capture data effectively, these parameters make the simulations appear overly strong and inflexible when contact is made (and can lead to numerical instability). By dropping the gains during contact, the simulation becomes more pliable and, if tuned correctly, more natural. This approach modifies the control gains and, therefore only models the passive effects of a contact. It does not capture active human responses such as protecting the head during a fall. Those responses would require additional motion capture examples or hand-programmed control systems such as the ones shown by Faloutsos and his colleagues [2001a].

A gain-scheduling controller swaps the original tracking gains with a new set of stiffness parameters, k_r and b_r, during contact. The gains that are adjusted depend on the region of the body being contacted. If the contact occurs along the trunk or head, the neck, back, and waist gains are adjusted. If contact occurs on an arm, only the gains of that arm are modified. To find these values, we tune the simulations offline by creating collisions and selecting stiffness and damping based on the reaction. Because only a few parameters are tuned at a time, this process can be done quickly.

After tuning, when contact is detected, the scheduler updates the stiffness parameters according to the following scheme:

$$k_t' = \begin{cases} k_t & t \le t_c \\ k_r & t > t_c, t \le t_c + t_e \\ k_r(1-\gamma(t)) + k_t\gamma(t) & t > t_c + t_e, t \le t_c + t_e + t_f \\ k_t & t > t_c + t_e + t_f \end{cases}$$

where k_t' replaces the original tracking gain, k_t in equation 1. The variables t_c, t_e, t_f correspond to the time of initial collision, a timed delay, and a ramp-up time to return to the original tracking gain. The weighting parameter $\gamma(t)$ uses a sinusoidal ramp to return to the original tracking gain. This scheme allows the reaction to happen as soon as the contact occurs and smoothes the return to tracking once the reaction is complete. A corresponding method is used to update the damping parameter, b_t.

Figure 7 shows a plot from a number of reactions created under the same external conditions but using different gains and times for t_f. Though the change in gains affects the tracking controller's ability to follow the data, we do not explicitly change the sequence of motion data being tracked. In this manner, the tracking and reacting control systems do not compete, but work together to produce the resulting motions.

6 Balancing while tracking

Many researchers in both robotics and graphics have constructed control systems for balance that maintain the position and velocity of the center of mass projected onto the ground plane. The effect of this control is to reduce the acceleration of the center of mass which is a characteristic of human balance [Pai and Patton 1997]. Our

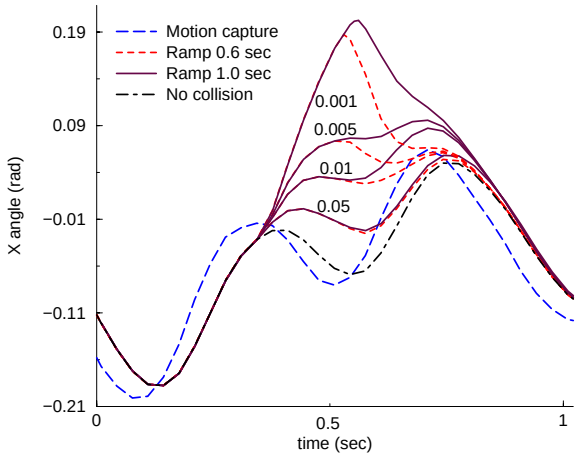

Figure 7: **A controlled hit to the side of the head with a variety of reactions in the neck. The different reactions are generated for four ratios of the reacting and tracking gains described in section 5. These ratios, labeled above, include two trials each with different delay times for the ramp of the scheduler.**

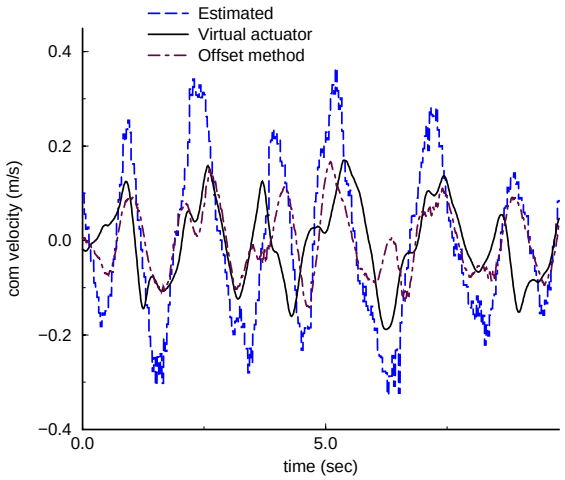

Figure 8: **This plot shows the center of mass velocity for a dancing behavior. We approximate the center of mass for the motion capture data and our offset method for balance follows this estimate. Conversely, the virtual actuator follows the center of support while the lower body tracks the motion capture data. The larger extent of the estimated center of mass velocity likely reflects errors in the estimate and kinematic differences between the actor and simulated character.**

problem differs in that the lower body motion is driven by motion capture data while we simultaneously control the center of mass for balance. We explored two approaches for controlling balance with the lower body: a *virtual actuator* computes joint torques for balance and combines them with the tracking torques in the lower extremities; a second controller offsets the desired angle in the hips and ankles before computing torques.

We maintain the center of mass by selecting a reasonable desired position and reducing the error between the simulated and desired values. The center of support, computed from the polygon created by the feet, is a simple value to compute and, when used as a desired position, yields stable balance. To add high-level information about balance from the human motion, we also experimented with estimating the projected center of mass from the data using a weighted sum of the recorded marker positions with estimated masses for the body parts.

In the *virtual actuator* approach, several actuators are coordinated to accomplish a single "virtual" actuation. Pratt used this technique for a variety of models and behaviors [Pratt 1995]. In our case, a balance torque in each of the leg joints (ankles, knees, hips) is computed based on a virtual horizontal force that "pushes" the center of mass toward the center of support:

$$^{O}F_v = k_v\,(x_d - x) - b_v\,(\dot{x}) \tag{3}$$

where $^{O}F_v$ indicates the force in the global coordinate system and k_v and b_v are the gain and damping terms. x and $\dot{x}$ are the center of mass and center of mass velocity projected on the ground plane while x_d is the desired center of mass. This force is not applied to the simulation directly but converted to joint torques.

Unlike the virtual actuator algorithm described by Pratt that computes a Jacobian relating the torques to the desired forces, we compute the torques directly from:

$$\tau_v = {}^{J}T_O\left({}^{O}F_v \times r_v\right) \tag{4}$$

where $^{J}T_O$ transforms the torque from the global coordinate system to the individual joint's coordinate system and r_v is the global vector from the center of mass to the joint center. This torque is then added to the tracking torque computed from equation 1, as $\tau = \tau_t + \tau_v$, and applied to the simulation.

A second approach offsets the desired joint angles following the algorithm suggested by Wooten and Hodgins [2000]. The controller computes an offset in the hips and ankles (h,a) as:

$$\lambda_{(h,a)} = k_{(h,a)}\,(x_d - x) - b_{(h,a)}\,(\dot{x}) \tag{5}$$

where $k_{(h,a)}$ and $b_{(h,a)}$ are the offset gain and damping terms, respectively. Once computed, the λ values are added to θ_d in equation 1. Figure 8 shows a comparison for the two torque-based balance approaches applied to a dancing behavior along with a plot of the approximate center of mass for the human data.

7 Results

We implemented a variety of examples to show different aspects of our system. Several filmstrips of the behaviors are shown in figure 10. Two sports, table tennis and boxing, were investigated in depth while other examples, such as fencing strikes, add breadth to the spectrum of hitting behaviors. In table tennis, our goal is to show precision control over the hitting action. With boxing, we show the simulation's ability to react to various punches. We also include a dancing example that reacts to user-controlled impacts to the head, waist, and arm in order to show a range of reactions.

7.1 Table tennis

Previous work on virtual tennis using motion capture has been presented by Molet and colleagues [1999], while their efforts focused on system architecture for an interactive application and ours on controlling a simulation with motion capture. Ping pong itself is a challenging behavior to control that has attracted the interest of roboticists as well [Andersson 1988]. While a motion

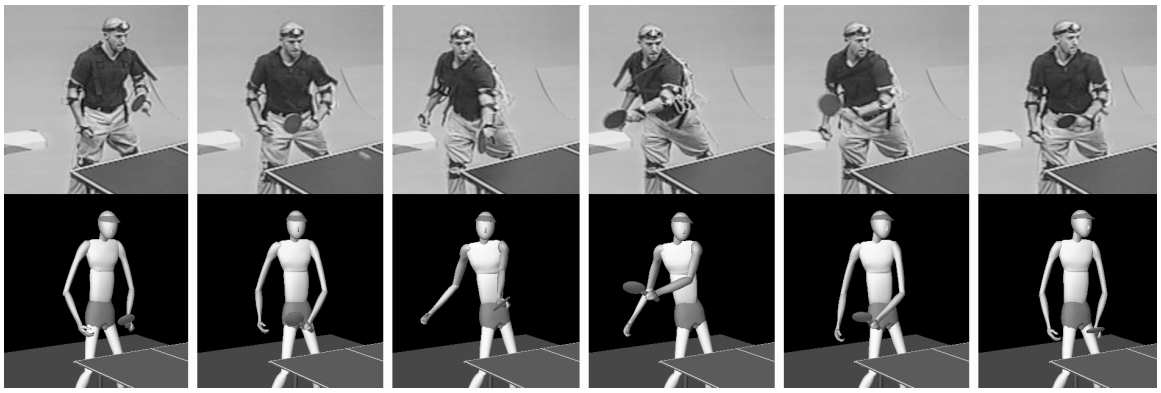

Figure 9: **Table tennis swing comparison. The specific data recorded with this video footage is driving the upper body of the simulated table tennis player, frames sampled at 0.17 sec. The character portrays visual features of the subject's movement even with the differences in their kinematics.**

capture-driven character could accomplish much the same performance kinematically as our dynamic simulation, we use this example to show that our physically based human can move by following motion capture while adapting automatically to perform controlled tasks. These two competing goals present an interesting control problem for a dynamic system.

To play table tennis, the position, orientation, and velocity of the racket must be controlled to hit an incoming ball to a particular location with a desired speed. Only the upper body joints track human data in the table tennis player, leaving the lower body to control balance using the offset method. A constant desired angle is fed into the lower body controller, providing continuity from swing to swing. The ball is implemented without spin and using a point-and-plane collision for contact with the racket. The ball's impact is assumed not to affect the motion of the racket. A figure of the players hitting a ball back and forth for a medium-speed volley along with results showing control over the ball and racket appear in the color images of the appendix. A comparison in figure 9 contrasts video footage of a table tennis swing with the simulated motion produced by tracking the upper-body motion recorded for that swing.

7.2 Boxing

The most important contribution of our work, creating motion capture-driven characters that can interact in a general way, is seen in the boxing examples. We show the integration of hits and reactions in this behavior. Punches are modified, like in table tennis, using inverse kinematics so that the boxing simulations are able to hit specified locations. As the simulations box each other, punches are exchanged back and forth, combining the tracking and reaction controllers without noticeable discontinuities in the generated motion. The simulations track data in each joint, combining balance with the torque-based virtual actuator.

In addition to the reactions found in boxing, we experimented in a more controlled setting where a "hand" moving under a user's specification interacts with a dancing simulation. The user disturbs the dancer's motion with "playful" jabs in the animation shown in figure 10. While gains are modified only in the local region of contact, under this more controlled setting the physical reaction of the collisions can be seen to extend through the entire body. The simulation even takes involuntary steps if the forces are large enough.

To evaluate the resulting animations, we show comparisons with live footage. Side-by-side comparisons of the person being captured and the simulation driven by the recorded motion illustrate characteristics that are retained in the simulated motion. Because our motion library does not include sample reactions, we rely on comparison with real boxing footage to assess the naturalness of the response of our simulated boxer reacting as in figure 10.

8 Discussion

In this paper, we present simulations that hit based on human data and react based on their dynamics. The underlying technology is new in that it explores whole body tracking with balance, data-driven control for hitting, and simulated reactions to hits for which there was no available data. These simulations are too slow to be used in games today (running about 10 times slower than real-time, without graphics, on a 400 Mhz R12000 SGI) but this research develops techniques for controlling simulations that may eventually replace kinematic models in appropriate applications. We also make suggestions about how to lower the computational load while keeping many of the benefits of the dynamics.

Whole body tracking means that the balance control must reconcile the sometimes conflicting goals of tracking the lower body motion and balancing. Our intuition had been that following a high-level parameter estimated from the motion capture data, such as the location of the center of mass, would be more effective than tracking joint angles in order to reproduce characteristics of the balancing behavior found in the tracked motion. While this was true in the slower free-style dancing behavior, the lag produced in faster motions, like boxing, was large enough to be detrimental. Until we better understand how to control the location of the center of mass, tracking the joint angles seems more promising.

Our balance controller occasionally takes a small, undeliberate step but is currently not able to lift and plant a foot purposefully to restore balance, much less to perform the quick footwork often seen in boxing. If a contact is too hard, the simulation will fall over, ungracefully. We do believe that our upper body control for hitting would work with more aggressive moves in lower body. However, reconciling the difference between stepping in the lower body and a tracked motion segment presents a potentially difficult control problem. One straightforward improvement to the controller would be to add intentional use of the arms for balance.

Our controllers for hitting are based on a small number of motion capture sequences: a dozen for table tennis and four for boxing. An open question is whether the level of control would have been improved by a larger motion capture library. Boxing would certainly be improved visually by including more variety in the styles of punch sequences. Also, table tennis may be adequately spanned by a small database and interpolation but, a larger set of swings would be useful in creating an interpolated library based on slow,

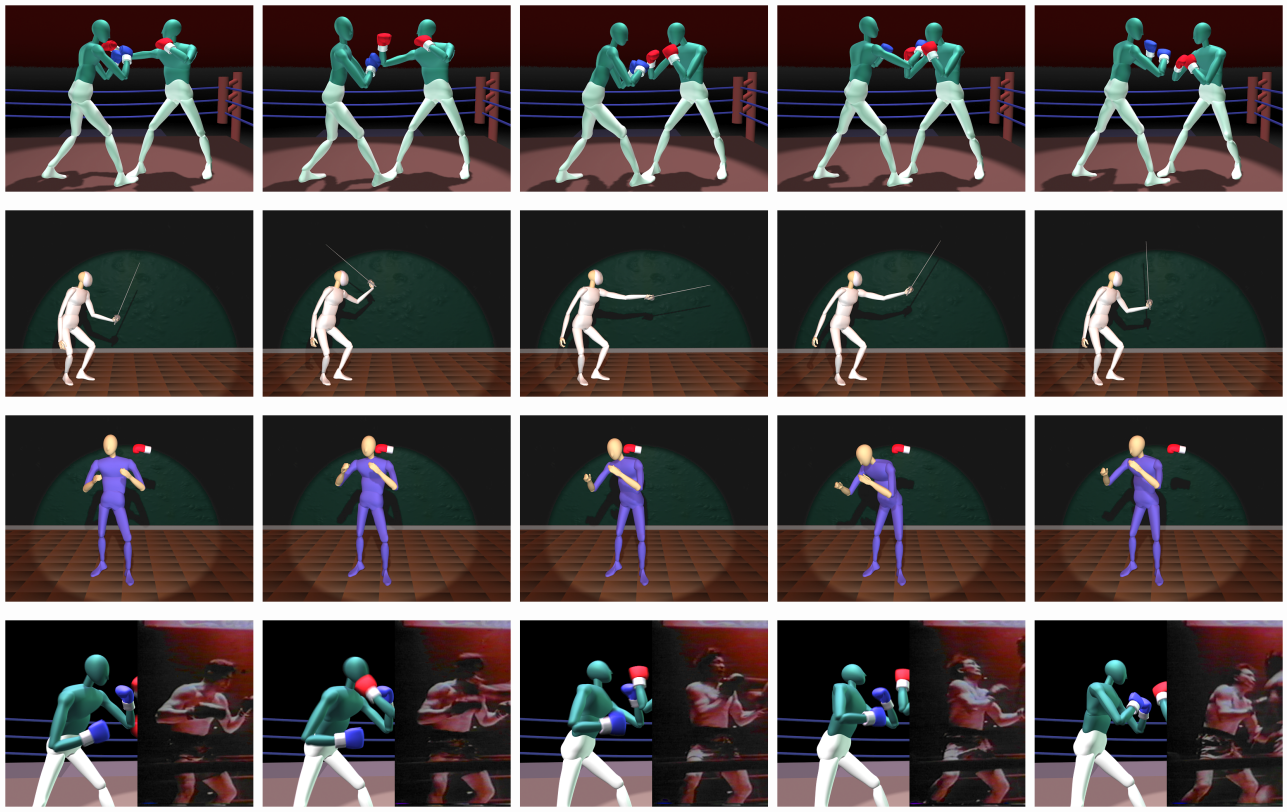

Figure 10: **Images include two boxers exchanging blows in the ring, a fencer performing a strike, a dancing character being hit in the head, and a comparison between a simulated and human boxer reacting to a thrown punch. The spacing of the images in time is uniform with the boxers in the ring shown at intervals of 0.33 s, the fencer at 0.33 s, the dancer at 0.25 s, and the composite of the simulated and human boxers at 0.08 s. (These images are repeated in color in the appendix.)**

medium, and fast hits. The number of synthetic swings in the current implementation was chosen by hand to be dense enough to keep the required adjustments to the racket position small. As more example swings with different speeds are considered, modifications for speed control could be minimized as well.

Passive reactions are implemented with a gain scheduler that reduces the stiffnesses at the instant of contact to allow the dynamics of the collision to influence the motion. Human motor control does not include such a drastic drop for reactions because the gains (stiffnesses) used during human arm movements, even strong arm movements such as boxing, are probably much lower. The high gains in our system are an artifact of the performance that we demand from the trajectory tracking controller. Including gravity compensation or other elements of inverse dynamics as feedforward terms would allow much lower feedback gains. Trajectory learning as proposed by Kawato would also provide a reduction in the gains [Kawato et al. 1987; Kawato 1990]. These approaches would likely more closely mimic the human control system and might obviate the need for lower gains during reactions.

Although we include physical effects of contact to provide realistic interactions in the resulting animations, we make certain approximations such as ignoring impact forces when the disturbances are small. In table tennis, the effect of the ball on the racket is unnoticable and can be ignored without visually changing the final animation. In boxing, the system only applies impact forces to the punched simulation because the response of the impact in the punching boxer is very small compared to that of the hit simulation. While our approximation shows the disturbance of the reaction in the simulation being hit, the impression from viewers is that the more subtle reactions of the hitter are important and seem missing, especially when the simulated impact is fairly large. In future versions, as this approximation become the limiting factor in the simulations' realism, a more complete solution will be required so that the hitting simulation also reacts to disturbances instead of following through stiffly as it does now.

8.1 Practical implementation

Physical approximations have value as debugging tools, devices for creating control systems, and speed-ups for faster motion generation. We use a number of tools in the process of developing the simulation controllers. Initial debugging and tuning is done on "pedestal" simulations that are anchored to ground at the waist. These systems run more quickly and eliminate difficulties related to maintaining balance. After tuning, porting the control systems to a full-body balancing simulation was relatively straightforward.

External forces applied to the simulation for control are physically unrealistic but maintain balance in a much more stable manner at the expense of some physical realism. The force described in equation 3 can be applied to the simulation at the center of mass rather than being converted to joint torques as in the presented examples. This control approach is equivalent to letting the character stand on its own legs, while an unseen spring and damper adjust the simulation's center of mass to keep it balanced. This character is able to perform more aggressive hits and withstand a wider range of blows than one supported by joint torques alone. For entertain-

ment applications where robustness is sometimes more important than naturalness, this type of supernatural control is likely to be quite helpful.

Simpler, hybrid dynamic/kinematic models can be used to reduce the overall computation. For example a simulation with many fewer joints could approximate the gross body motion reasonably while kinematic "add–ons" could improve the visual appearance. In addition, removing parts of the body with low mass allows the simulation to run with a larger timestep, significantly cutting computation cost, because the numerical stiffness of the overall system is lowered. Another, less compute-intensive approach for including simulated reactions is to drive characters with motion capture kinematically and turn on the simulation only during the time of impact and response. In this manner, the expense of the simulation can be avoided until it is required.

In closing, the combination of motion capture data and simulation proved to be quite powerful for the applications presented here. We believe the information about human motion contained in motion capture data coupled with the physical realism provided by simulation will be of great benefit in other applications where controllable and reactive human motion is required.

ACKNOWLEDGMENTS

The authors would like to thank Joel Heires for his help modeling and Georgia Tech's IMTC for use of their labspace. This research and the motion capture equipment used was funded in part by NSF Instrumentation Grant 9818287 and NSF 9900333.

References

ANDERSSON, R. L. 1988. *A Robot Ping-Pong Player*. The MIT Press.

BRUDERLIN, A., AND WILLIAMS, L. 1995. Motion signal processing. In *Proceedings of SIGGRAPH 95*, ACM SIGGRAPH, 97–104.

CHOI, K.-J., PARK, S.-H., AND KO, H.-S. 1999. Processing motion capture data to achieve positional accuracy. *Graphical models and image processing: GMIP 61*, 5, 260–273.

FALOUTSOS, P., VAN DE PANNE, M., AND TERZOPOULOS, D. 2001. Composable controllers for physics-based character animation. In *Proceedings of SIGGRAPH 2001*, ACM SIGGRAPH, 251–260.

FALOUTSOS, P., VAN DE PANNE, M., AND TERZOPOULOS, D. 2001. The virtual stuntman: dynamic characters with a repertoire of autonomous motor skills. *Computers & Graphics 25*, 6, 933–953.

GLEICHER, M. 1998. Retargeting motion to new characters. In *Proceedings of SIGGRAPH '98*, ACM SIGGRAPH, 33–42.

HODGINS, J. K., WOOTEN, W. L., BROGAN, D. C., AND O'BRIEN, J. F. 1995. Animating human athletics. In *Proceedings of SIGGRAPH '95*, ACM SIGGRAPH, 71–78.

KAWATO, M., FURUKAWA, K., AND SUZUKI, R. 1987. A hierarchical neural-network model for control and learning of voluntary movement. *Biological Cybernetics 57*, 169–185.

KAWATO, M. 1990. Feedback-error-learning neural network for supervised motor learning. In *Advanced Neural Computers*, R. Eckmiller, Ed. Elsevier Science Publishers, 365–472.

KOKKEVIS, E., METAXAS, D., AND BADLER, N. 1996. User-controlled physics-based animation for articulated figures. In *Proceedings of Computer Animation 1996 Conference*, 16–26.

LASZLO, J. F., VAN DE PANNE, M., AND FIUME, E. 1996. Limit cycle control and its application to the animation of balancing and walking. In *Proceedings of SIGGRAPH '96*, ACM SIGGRAPH, 155–162.

LEE, J., AND SHIN, S. Y. 1999. A hierarchical approach to interactive motion editing for humanlike figures. In *Proceedings of SIGGRAPH '99*, ACM SIGGRAPH, 39–48.

MOLET, T., AUBEL, A., CAPIN, T., CARION, S., LEE, E., THALMANN, N. M., NOSER, H., PANDZIC, I., SANNIER, G., AND THALMANN, D. 1999. Anyone for tennis? *Presence: Teleoperators and Virtual Environments 8*, 2, 140–156.

OSHITA, M., AND MAKINOUCHI, A. 2001. A dynamic motion control technique for human-like articulated figures. *Computer Graphics Forum 20*, 3.

PAI, Y.-C., AND PATTON, J. 1997. Center of mass velocity-position prediction for balance control. *Journal of Biomechanics 30*, 4, 347–354.

PLAYTER, R. 2000. Physics-based simulation of running using motion capture. In *Course notes for SIGGRAPH 2000*, ACM SIGGRAPH.

POLLARD, N. S. 1999. Simple machines for scaling human motion. In *Computer Animation and Simulation '99, Eurographics Animation Workshop*, 3–11.

POPOVIĆ, Z., AND WITKIN, A. 1999. Physically based motion transformation. In *Proceedings of SIGGRAPH 99*, ACM SIGGRAPH, 11–20.

PRATT, J. E. 1995. *Virtual Model Control of a Biped Walking Robot*. Masters thesis, Massachusetts Institute of Technology.

ROSE, C., GUENTER, B., BODENHEIMER, B., AND COHEN, M. F. 1996. Efficient generation of motion transitions using spacetime constraints. In *Proceedings of SIGGRAPH '96*, ACM SIGGRAPH, 147–154.

ROSE, C., COHEN, M., AND BODENHEIMER, B. 1998. Verbs and adverbs: Multidimensional motion interpolation. *IEEE Computer Graphics and Applications 18*, 5, 32–40.

SYMBOLIC DYNAMICS INC. 1990. *SD/Fast User's Manual*.

UNUMA, M., ANJYO, K., AND TAKEUCHI, R. 1995. Fourier principles for emotion-based human figure animation. In *Proceedings of SIGGRAPH '95*, ACM SIGGRAPH, 91–96.

VAN DE PANNE, M., AND LAMOURET, A. 1995. Guided optimization for balanced locomotion. In *Computer Animation and Simulation '95*, Eurographics, 165–177.

WILEY, D. J., AND HAHN, J. K. 1997. Interpolation synthesis of articulated figure motion. *IEEE Computer Graphics & Applications 17*, 6, 39–45.

WITKIN, A., AND POPOVIĆ, Z. 1995. Motion warping. In *Proceedings of SIGGRAPH 95*, ACM SIGGRAPH, 105–108.

WOOTEN, W. L., AND HODGINS, J. K. 2000. Simulation of leaping, tumbling, landing, and balancing humans. *IEEE International Conference on Robotics and Automation*.

ZORDAN, V. B., AND HODGINS, J. K. 1999. Tracking and modifying upper-body human motion data with dynamic simulation. In *Computer Animation and Simulation '99*, Eurographics, 13–22.

ZORDAN, V. B. 2002. *Motion capture-driven simulations that hit and react*. Ph.D. Thesis, Georgia Institute of Technology.

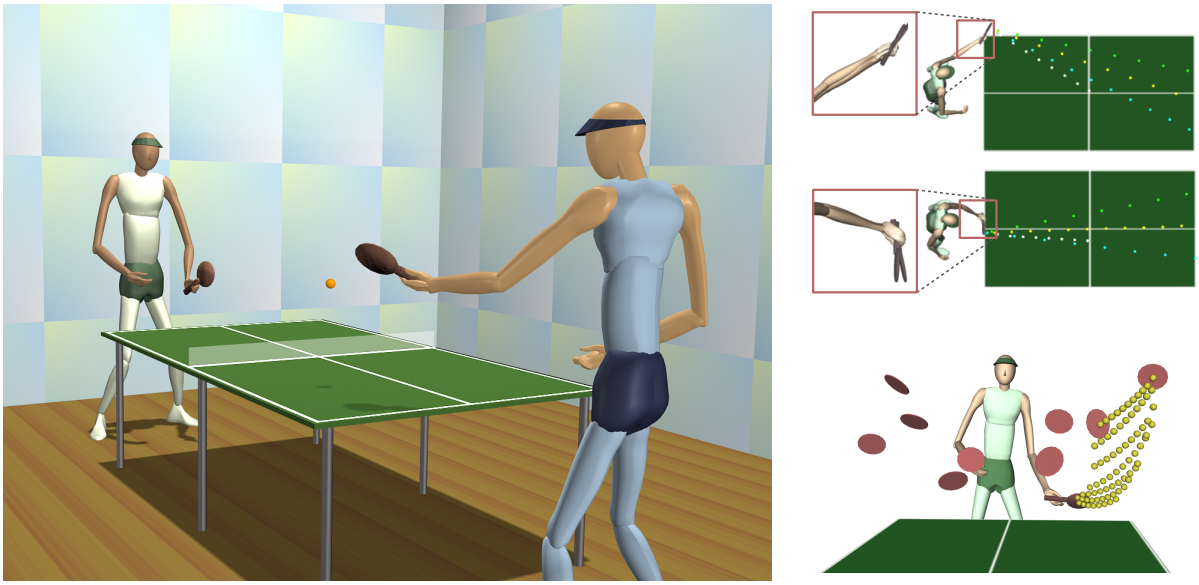

Figure 11: **Image 1 (clockwise from left): Two simulated table tennis players hit a ball back and forth in a controlled volley. Image 2: Two swings with the player shown at the time of contact. The incoming ball, starting from the net moving to the left, is returned in different directions (sampled every frame). The inlays show the racket angle for the furthest extents. Image 3: Racket locations at the time of contact for tracked raw swings combined in a continuous motion with each disk representing the paddle's position and orientation at the contact point. The spheres show hit locations for synthetic swings generated between three raw swings.**

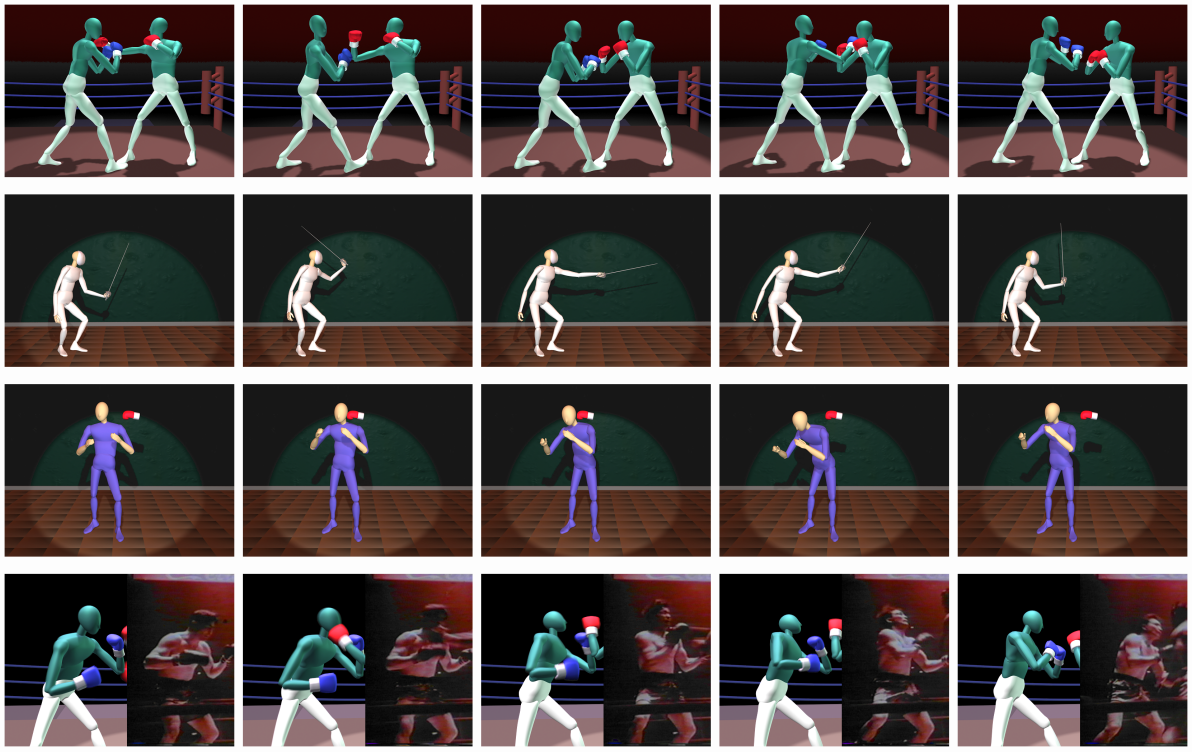

Figure 12: **Images include two boxers exchanging blows in the ring, a fencer performing a strike, a dancing character being hit in the head, and a comparison between a simulated and human boxer reacting to a thrown punch. The spacing of the images in time is uniform with the boxers in the ring shown at intervals of 0.33 s, the fencer at 0.33 s, the dancer at 0.25 s, and the composite of the simulated and human boxers at 0.08 s.**