January 1993

Real-Time Control of a Virtual Human Using Minimal Sensors

Norman I. Badler
University of Pennsylvania

Michael J. Hollick
University of Pennsylvania

John P. Granieri
University of Pennsylvania

Postprint version. Published in *Presence*, Volume 2, Issue 1, 1993, pages 82-86. Publisher URL: http://www.mitpressjournals.org/loi/pres

Real-Time Control of a Virtual Human Using Minimal Sensors

Abstract
We track, in real-time, the position and posture of a human body, using a minimal number of 6 DOF sensors to capture full body standing postures. We use 4 sensors to create a good approximation of a human operator's position and posture, and map it on to our articulated computer graphics human model. The unsensed joints are positioned by a fast inverse kinematics algorithm. Our goal is to realistically recreate human postures while minimally encumbering the operator.

Keywords
computer graphics, computer animation, human model, human posture

Comments
Postprint version. Published in *Presence*, Volume 2, Issue 1, 1993, pages 82-86. Publisher URL: http://www.mitpressjournals.org/loi/pres

Next Up Previous
Next: Abstract

Real-Time Control of a Virtual Human Using Minimal Sensors

Norman I. Badler
Michael J. Hollick
John P. Granieri

Computer Graphics Research Laboratory
Department of Computer and Information Science
University of Pennsylvania
Philadelphia, Pennsylvania 19104-6389

Next Up Previous

Next: Background **Up:** Real-Time Control of a **Previous:** Real-Time Control of a

Abstract

We track, in real-time, the position and posture of a human body, using a minimal number of 6 DOF sensors to capture full body standing postures. We use 4 sensors to create a good approximation of a human operator's position and posture, and map it on to our articulated computer graphics human model. The unsensed joints are positioned by a fast inverse kinematics algorithm. Our goal is to realistically recreate human postures while minimally encumbering the operator.

Next Up Previous

Next: Sensor Interface **Up:** Real-Time Control of a **Previous:** Abstract

Background

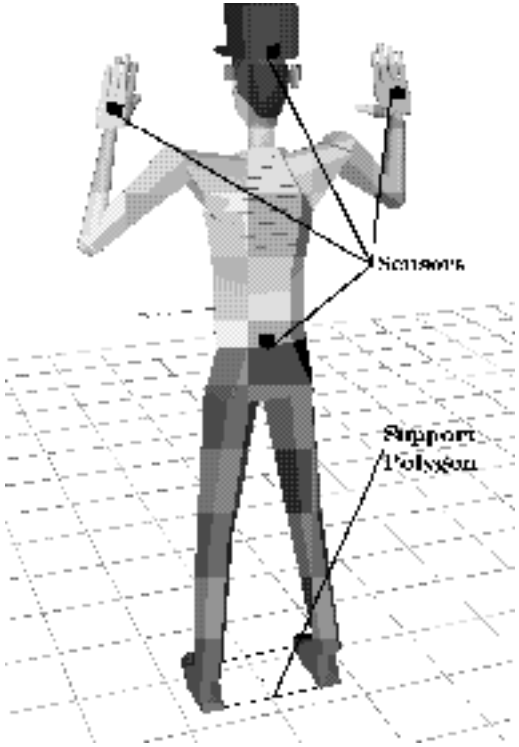

Figure 1: Sensor Placement and Support Polygon

Ideally, a virtual environment interface should be able to measure and recreate a participant's posture exactly. Rather than the traditional ``disembodied hand'' approach, we would like to generate a complete, realistically postured human image. However, the equipment needed to accurately track every body segment (or joint angle) of a human is costly and cumbersome. We face several questions: how closely must the virtual human's posture match the operator's, and how

much information do we need to achieve this degree of realism?

As in other areas of VR, the degree of realism necessary depends greatly on the tasks we would like to perform [11]. We have concentrated on creating an interface that will allow a human participant to perform basic tasks, using a minimal number of sensors to derive feasible, reasonably accurate postures. Three pieces of information are essential for our posture reconstruction algorithm: the participant's view vector, center of mass, and the location of the end-effectors the participant will use to interact with the virtual environment.

It is fairly obvious that the view vector needs to be accurate. If the synthesized view does not correspond to the participant's head motions, disorientation is inevitable [4][8]. Similarly, end-effectors must be placed accurately in the environment. User interaction will be difficult if the virtual representations of end-effectors do not match reality closely. The body posture can then be computed, based on these inputs, the participant's center of mass, and our human body model. While this posture may not match that of the participant's at every joint, the vital aspects will be preserved. For many applications, this degree of accuracy is sufficient.

In our simulation, we are considering the hands to be the only end-effectors available to the participant. Therefore, we need four sensors to get the information required to drive the simulation. These are placed on each palm, the head, and the waist (Fig. 1).

This posture information can then be used in a variety of ways:

- A participant's motion can be used to directly drive a simulated human in a virtual environment. This can occur in real-time, and additionally allows for interaction with other virtual humans in a networked environment.
- This information can also be used for motion/task recording and analysis. A movement can be recorded and used to drive human figures of different sizes, in different environments. For example, a 5^{th} percentile operator's sequence of motions can be mapped onto a 95^{th} percentile figure.
- It can be used as an intuitive way for controlling or positioning figures in a traditional, non-immersive environment. Even if the figure being controlled is not a direct representation of the operator in a virtual environment, this technique provides a simple way of inputting a human posture for a keyframed animation.

All our work is in the context of *Jack*™, a multifaceted system for interactively modeling, manipulating, and animating articulated figures, principally human figures [7]. *Jack* represents figures as collections of rigid segments connected by joints that may have arbitrary rotational or translational degrees of freedom. The model of the human figure that we use for the example here has 36 joints with a total of 88 degrees of freedom, excluding the hands and fingers. It has a torso consisting of 17 segments and 18 vertebral joints.

Jack also provides a set of general constraints (e.g. point-to-point, point-to-plane) and the methods to solve them [12]. *Jack* provides a set of behavior functions for human models [9]. The behavior functions manage a set of constraints and sub-models (e.g. the spine, active stepping) of the human model. We use both of these features, constraints and the behavior functions, to map sensor values into human postures.

Next Up Previous

Next: Sensor Interface **Up:** Real-Time Control of a **Previous:** Abstract

---

Next Up Previous

Next: Posturing the Jack **Up:** Real-Time Control of a **Previous:** Background

Sensor Interface

We are using the **Flock of Birds** from Ascension Technology, Inc. for position/orientation tracking. With an Extended Range Transmitter the operator can move about in an 8-10 foot hemisphere. Each Bird sensor is connected to a Silicon Graphics 310VGX via a direct RS232 connection running at 38,400 baud.

One of the initial problems with this system was slowdown of the simulation due to the sensors. The IRIX operating system introduces a substantial delay between when data arrives at a port and when it can be accessed. This problem was circumvented by delegating control of the Flock to a separate server process. This server will configure the Flock to suit a client's needs, then provide the client with updates when requested. The server takes updates from the Birds at the maximum possible rate, and responds to client requests by sending the most recent update from the appropriate Bird. This implementation allows us to access the Flock from any machine on our network and allows the client to run with minimal performance degradation due to the overhead of managing the sensors. We now get about 25-30 updates per second on each of the sensors, of which we use only about 8 to 10. We achieve a frame rate of about 8-10 frames per second on an SGI 310/VGX, with a shaded environment of about 2000 polygons. The bulk of the computation lies in the inverse kinematics routine, which runs in the interframe update to maximize position accuracy without sacrificing refresh rate.

Next Up Previous

Next: Conclusion and Future **Up:** Real-Time Control of a **Previous:** Sensor Interface

Posturing the Jack Figure

Jack uses an inverse kinematics algorithm to solve several types of constraints that can be placed on articulated figures. These constraints generally require several parameters to be specified: a type (e.g. point-to-point, point to plane), an end effector site, the joint chain that will be involved, and a weight, among others. We control the figure through the behavior functions associated with human figures in *Jack*. The various human behaviors operate by managing a number of constraints that are placed on the figure.

The system must first be calibrated to account for the operator's size. This can be done in two ways - the sensor data can be offset to match the model's size, or the model can be scaled to match the operator. Either approach may be taken, depending on the requirements of the particular situation being simulated. For motion recording, it is usually preferable to have the simulated figure scaled to match the operator. An ergonomic study of a vehicle, however, would require an arbitrary figure size; in this case, the sensor data would be offset as required.

Each frame of the simulation requires the following steps:

- **Step 1:** The pelvis segment is moved as the first step of the simulation. The absolute position/orientation of this segment is given by the waist sensor after adding the appropriate offsets. The figure is rooted through the pelvis, so this sensor determines the overall location of the figure.

- **Step 2:** The spine is now adjusted, using the location of the waist sensor and pelvis as its base. The *Jack* human figure includes an articulated spine that models the human spine as a chain of 18 dependent joints. These joints represent the 12 thoracic and 5 lumbar vertebrae. Four parameters are required to manipulate the spine: the initiator joint, resistor joint, resistance, and spine target position [6]. In our simulation the first three parameters are fixed, and the spine target position

is extracted from the relationship between the waist and head sensors. The waist sensor gives us the absolute position of the pelvis and base of the spine, while the rest of the upper torso is placed algorithmically by the model.

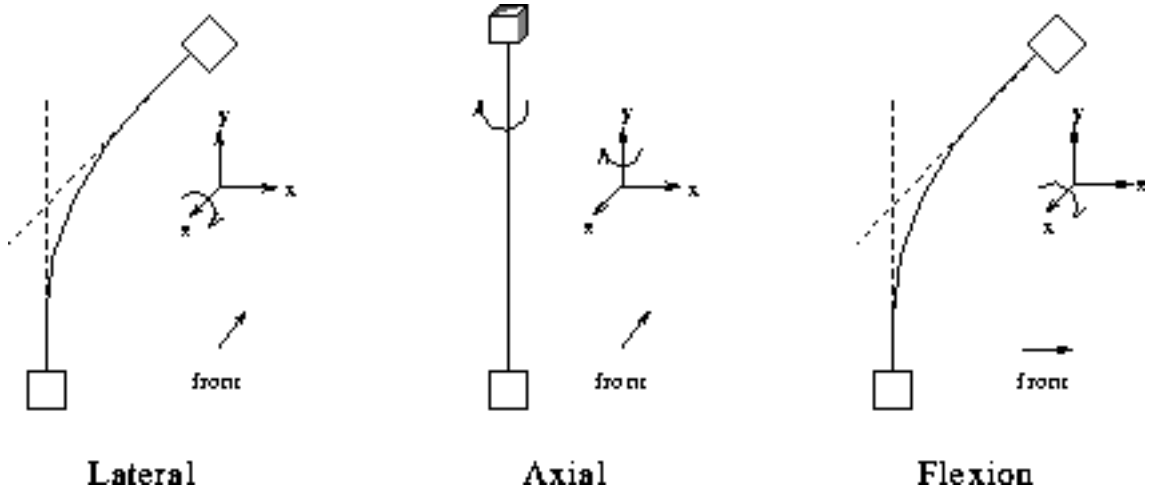

Figure 2: Extracting the Spine Target Vector

The spine target position is a 3 vector that can be thought of as the sum of the three types of bending the spine undergoes - front/back bending or flexion, twisting or axial bending, and side or lateral bending.

The waist sensor gives us orientation information for the base of the spine; information on the top of the spine is extracted from the head sensor via a heuristic. The target vector is computed from the differences between these two orientations. Lateral bending is found from the difference in orientation along the **z** axis, axial twisting is found from the difference in **y** orientation, and flexion is determined from the difference in **x** orientation (Fig. 2). Note that the ``front'' vectors in this figure indicate the front of the human. The target vector is sent directly to the model to simulate the approximate bending of the operator's spine.

- **Step 3:** Now that the torso has been positioned, the arms can be set. Each arm of the figure is controlled by a sensor placed on the operator's palm. This sensor is used directly as the goal of a position and orientation constraint. The end effector of this constraint is a site on the palm that matches the placement of the sensor, and the joint chain involved is the wrist, elbow, and shoulder joint.

  We can optionally employ a collision avoidance technique at this point [13]. This system manages constraints on the human body to prevent body segments from intersecting each other. This process slows the simulation down somewhat,

but it is a useful tool when maximum realism is required.

- **Step 4:** The figure's upper body is now completely postured so we can compute the center of mass. This site is projected onto the ground plane, and compared against the figure's support polygon (Fig. 1). If the projection of this site is outside the support polygon, it is no longer balanced. The active stepping behaviors are used to compute new foot locations that will balance the figure. Leg motions are then executed to place the feet in these new locations [9][5].

Next Up Previous

Next: Conclusion and Future **Up:** Real-Time Control of a **Previous:** Sensor Interface

---

Next Up Previous

Next: Acknowledgements **Up:** Real-Time Control of a **Previous:** Posturing the Jack

Conclusion and Future Work

For a given task and work environment, end-effector positions will be similar for a wide range of body sizes. Our system is designed to take advantage of this similarity, by measuring the absolute 3D cartesian space coordinates of the points of interest on the body, and using them directly as goals for end-effector constraints, rather than measuring or deriving the operator's joint angles. Thus, while the model's posture may not precisely match the operator's, the end effectors of the constraints are always correct. This is very important, especially in situations where the operator is controlling a human model of different size in a simulated environment. Additionally, by obtaining our data in a relatively unobstructive manner, we insure that the operator moves in a natural way, unencumbered by measuring equipment.

We have not run experiments to determine the absolute accuracy or fidelity of our tracking system between the recreated posture and the actual posture of the operator. With the collision avoidance enabled, the generated postures are generally accurate, if not exact, representations of the actual posture. For high-fidelity motion recording, this is not good enough, but for most interactive applications it is an extremely useful technique.

Some future enhancements:

- We will use our anthropometric scaling system (SASS) to better choose a correct body size to match that of the operator. That is a matter of interactively anthropometrically scaling (as opposed to regular scaling) the human figure from 5th percentile up to 95th percentile, stopping where the sensor locations fit best [3][1].

- Another problem present is sensor noise. We need better noise filtering algorithms, both for noise from the actual equipment, and noise from the human operator. Humans tend to twitch and oscillate at certain frequencies, and these could be filtered out.

- Our simulation will be augmented to allow the participant to interact more directly with the environment. Actions can be initiated with direct actions (e.g. press a button to grasp), or indirectly; for example, placing both hands in intersection with an object can be a cue for grasping [2]. Sound rendering will be used to add realism and to signal events, such as collisions or warnings [10].

Even at this stage of development, the virtual *Jack* model is proving to be a useful tool in fast, intuitive input of whole body posture and location with minimal sensing hardware. Our system provides a method of interacting with a virtual environment without encumbering the participant.

Next Up Previous

Next: Acknowledgements **Up:** Real-Time Control of a **Previous:** Posturing the Jack

---

Next Up Previous

Next: References **Up:** Real-Time Control of a **Previous:** Conclusion and Future

Acknowledgements

This research is partially supported by ARO Grant DAAL03-89-C-0031 including participation by the U.S. Army Research Laboratory (Aberdeen), Natick Laboratory, the Army Research Institute, and NASA Ames Research Center; U.S. Air Force DEPTH contract through Hughes Missile Systems F33615-91-C-0001; DARPA through General Electric Government Electronic Systems Division; and NSF CISE Grant CDA88-22719.

Next Up Previous
Up: Real-Time Control of a **Previous:** Acknowledgements

References

1

F. Azoula. Sass user's guide. Technical report, Computer and Information Science, Univ. of Pennsylvania, Philadelphia, PA, 1993.

2

D. Zeltzer D. Sturman and S. Pieper. Hands on interactions with virtual environments. *UIST '89: ACM SIGGRAPH/ SIGCHI Symposium on User Interface Software and Technology*, 1989.

3

M. Grosso, R. Quach, and N.I. Badler. Anthropometry for computer animated human figures. In N. Magnenat-Thalmann and D. Thalmann, editors, *State-of-the Art in Computer Animation*, pages 83--96. Springer-Verlag, New York, NY, 1989.

4

R. Held. Correlation and decorrelation between visual displays and motor output. In *Motion sickness, visual displays, and armored vehicle design*, pages 64--75. Ballistic Research Laboratory, Aberdeen Proving Ground, MD, 1990.

5

H. Ko and N.I. Badler. Straight line walking animation based on kinematic generalization that preserves the original characteristics. *Graphics Interface '93*, 1993.

6

G. Monheit and N. Badler. A kinematic model of the human spine and torso. *IEEE Computer Graphics and Applications*, 11(2):29--38, 1991.

7

C. Phillips N.I. Badler and B.L. Webber. *Simulated Humans: Computer Graphics, Animation, and Control*. Oxford University Press, to appear 1993.

8

C.M. Oman. Motion sickness: A synthesis and evaluation of the sensory conflict theory. *Canadian Journal of Physiology and Pharmacology*, 68:264--303, 1990.

9

C. Phillips and N.I. Badler. Interactive behaviors for bipedal articulated figures. *Computer Graphics*, 25(4):359--362, 1991.

10

T. Takala and J. Hahn. Sound rendering. *Computer Graphics*, 26(2):211--219, 1992.

11

D. Zeltzer. Autonomy, interaction, and presence. *Presence*, 1(1):127--132, 1992.

12

J. Zhao and N.I. Badler. Real time inverse kinematics with joint limits and spatial constraints. Technical Report MS-CIS-89-09, Computer and Information Science, Univ. of Pennsylvania, Philadelphia, PA, 1989.

13

X. Zhao and N.I. Badler. Near real-time body awareness. *submitted to First Bilkent Computer Graphics Conference*, 1993.
