Closed Form and Geometric Algorithms for Real-Time Control of an Avatar

Sudhanshu K Semwal[1] Ron Hightower Sharon Stansfield
Sandia National Laboratories
Albuquerque, NM, USA
semwal@redcloud.uccs.edu|rrhight@isrc.sandia.gov|sastans@sandia.gov
[1]On sabbatical from University of Colorado, Colorado Springs

ABSTRACT

In a virtual environment with multiple participants, it is necessary that the user's actions be replicated by synthetic human forms. Whole body digitizers would be the most realistic solution for capturing the individual participant's human form, however the best of the digitizers available are not interactive and are therefore not suitable for real-time interaction. Usually, a limited number of sensors are used as constraints on the synthetic human form. Inverse kinematics algorithms are applied to satisfy these sensor constraints. These algorithms result in slower interaction because of their iterative nature, especially when there are a large number of participants. To support real-time interaction in a virtual environment, there is a need to generate closed form solutions and fast searching algorithms.

In this paper, a new closed form solution for the arms (and legs) is developed using two magnetic sensors. In developing this solution, we use the biomechanical relationship between the lower arm and the upper arm to provide an analytical, non-iterative solution. We have also outlined a solution for the whole human body by using up to ten magnetic sensors to break the human skeleton into smaller kinematic chains. In developing our algorithms, we use the knowledge of natural body postures to generate faster solutions for real-time interaction.

MOTIVATION

The aim of Virtual Reality (VR) research is to provide real-time interaction and immersion in a Virtual Environment (VE). The existence of (at least) seventeen senses has been described in the book by Rivlin and Gravelle [1]. However, the focus of virtual reality research at this time is to provide human-computer interaction [2] using mainly visual (graphics), force and tactile feedback[3, 4], and sound cues [2]. Interaction using smell, taste, and direct stimulation of the brain [1] are some other active and open areas of research at this time.

In distributed virtual reality applications with multiple participants [5, 6, 7, 8, 9, 10], users are embodied in the virtual world as complete human forms. This embodied synthetic human form, also called an *Avatar*, is enslaved to the participant as it replicates every move of the participant [6, 11]. Constant tracking and *real-time* mapping of the participant's postures to the corresponding avatar are two main issues.

MODELING HUMAN FORM AND MULTIPLE SENSORS

Consider the complexity of modeling the human form. The human body has some 206 bones acted upon by over 600 pairs of skeletal muscles [12]. All of these muscles and bones have complex 3D shapes. In addition, there is a certain amount of variation in the structure of the human body from person to person. Some of the smaller muscles are known to not even exist for certain percentages of the population. For this reason, the avatar issue, that of enslaving a synthetic human form to the participant's action, is an enormous computation and technical challenge.

To meet this challenge, a four-layered approach [13] has been used by several researchers [14, 15, 16]. These four layers are: skeletal, muscular, skin, and clothing layers.

The skeletal layer defines the human body by a set of joints. These joints are connected by line segments resulting in an articulated (or stick)

figure (See Figure 1). Motion algorithms specify the position and orientation of the joints for the synthetic human-figure based on the length of the limbs, the degrees of freedom of the skeletal system, and the underlying muscle models. There are several motion techniques available to determine the position of joints. A recent survey of these algorithms is in [17]. The following are some examples: 3D-Keyframing [18, 19, 20], forward and inverse kinematics [20], forward and inverse dynamics [21, 22, 23], goal directed motion [24], rotoscopy using video analysis [22, 25], and Artificial Neural Networks [26].

Once the position and orientation of the human-form has been estimated, skin and clothes [19, 24] are wrapped around the skeleton to provide a realistic appearance for the synthetic human-form. Once the joint positions are specified, the shape is generated by surrounding the skeleton with polygons or free form parametric surfaces. Usually, a polygonal mesh covering the human body is used for this purpose [14, 17, 27]. Bezier surfaces [28], S-patches [15, 29], and generalized cylinders [15] are some examples of parametric surfaces used for human animation [30]. Range data [31], and *metaballs* [32] have also been used to render human forms.

Accurate representation of the musculoskeletal geometry is important for a biomechanical model of the human form, and for visualizing the three-dimensional geometric relationship between the muscles and bones [33, 34]. Chen and Zeltzer describe a robust and general biomechanical model for muscles using the finite element method [33]. Line of action is the focus for biomechanical human animation in the works of [34].

As the synthetic human form moves, surface deformations are usually simulated by changing the vertices of the polygons or the control points that define the surface [15, 27]. The works of Carignan and Thalmann et. al. [35] are excellent examples of modeling the clothing layer and associated deformations [27].

Sometimes it is necessary that a large number of sensors on the human body be tracked. Commercial software (for example, available from *PeakPerformance technology*) essentially allows the user to digitize key body landmarks from video tapes, and then perform kinematic analysis from the resulting stick figures. At times the whole human body is digitized (e.g. using the *Cyberware* digitizer) to provide life-like synthetic forms, e.g. in the movies *Terminator 2* and *Jurassic Park*. On the other hand, the work by Badler et. al. [10]

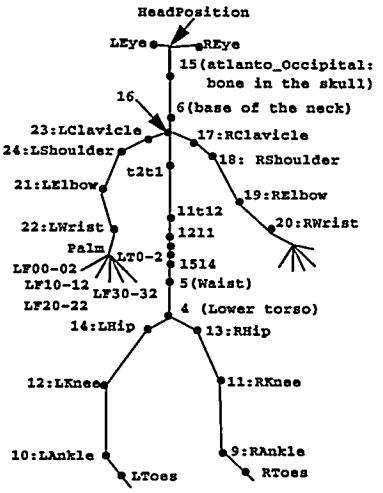

Figure 1: The human Skeleton

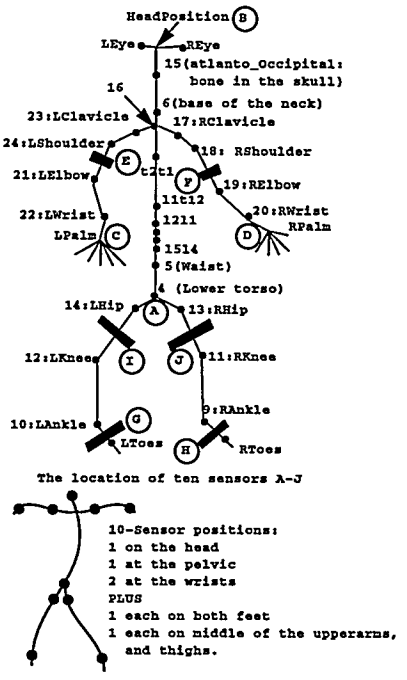

Figure 2: Placement of ten sensors

uses four sensors to map the user's actions.

AVATARS IN A DISTRIBUTED ENVIRONMENT

A shared virtual environment is being developed at Sandia National Laboratories for situational and close quarters training. The system is a distributed and shared VE [6, 7, 8]. We have been using a system of four magnetic trackers to map the participant's position to that of the avatar. In Figure 2, these sensors are marked A–D. These position trackers are placed on the participant's palms, head, and lower back (lower torso). Similar to the solution in [10], the position and orientation of these trackers is used as a constraint to the Jack®'s motion algorithm which uses the inverse kinematics algorithm to satisfy these constraints.

The distributed system runs on a variety of SGI platforms across the local area network using multi-casting. A detailed explanation of VR Station and the VR/IS[1] network I/O support can be found in [6, 8, 9]. Here we briefly explain the overall functioning of the system. (See Figure 3).

The VR Station process is responsible for displaying the avatar. The *VR/IS model format* (also called the *.hi format*) defines the avatar's skeleton-tree hierarchy, and the associated body parts. This *.hi* format for an avatar is known to all the processes in the distributed system. Therefore only joint orientations are needed to define the posture of an avatar. These joint positions (a set of transformations) are sent over the network in a packet called the *VR/IS packet.*

By using multi-casting, each avatar can have its own own Jack®server for applying inverse kinematic algorithms to satisfy the sensor constraints. The conversion from the Peabody hierarchy of Jack®[24] to VR/IS model format is by means of a separate conversion utility.

Our focus now is on using eight to ten sensors for a full body implementation of the avatar. See Figure 2 for the placement of the sensors (A to J). The main idea is that up to ten sensors are enough to break the human skeleton into smaller, manageable portions; yet they are only slightly more encumbering than the four sensor solution developed in [10]. However, a major advantage is that we are not using the iterative, slower inverse kinematics algorithms as in [10]. Instead we have developed a closed form, analytic solution

[1] Virtual Reality and Intelligent Simulation Laboratory

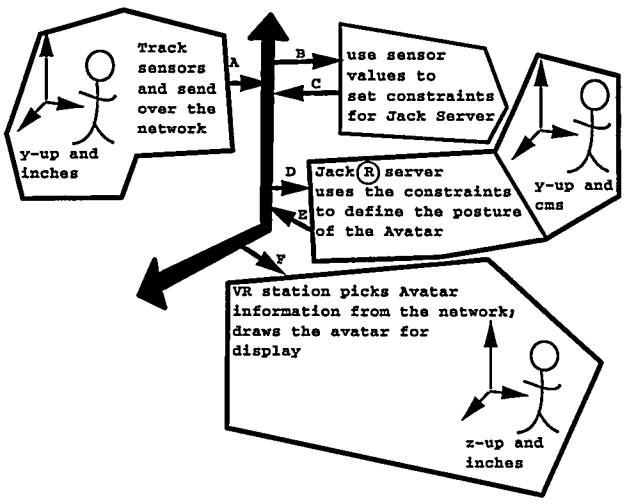

Figure 3: Interaction in the Distributed Environment

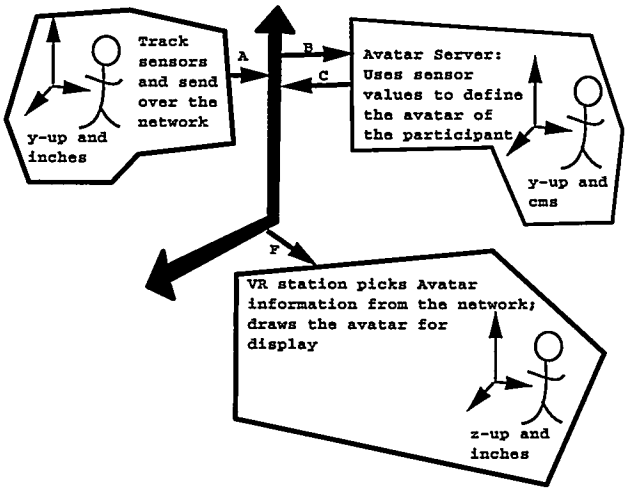

Figure 4: Interaction in the Distributed Environment using the Avatar Server

for arms and legs of the skeleton. For example, we use two sensors for the human arm to provide an analytical solution.

In the following sections, we first explain the mapping from one coordinate system to another in our distributed environment. We then present a closed form solution for an arm using two sensors. This solution is easily extensible to both the arms and legs. Later we propose a solution for the whole body positioning, which is still being implemented.

Mapping of the Coordinate Systems

Working in such a distributed environment, the first task is to understand the issue of moving the information from one coordinate system to another. Although the transformations are simple, one has to determine the proper sequence (see also Figure 4):

1. The position and orientation of the FastTrak magnetic sensors is available in the coordinate system for the physical space (room). This physical space is defined in inches in a right handed coordinate system (RHCS) whose y-axis is perpendicular to the floor of the room. The orientation of a magnetic sensor is specified by its own local coordinate system (LCS). Figure 5(a) shows the three axes of the LCS for the two sensors. These sensors are strapped to the user's upper arm and palm. The z-axis of these sensors lies along the length of the limb, from one joint to another. The y axis is perpendicular and points away from the center of the limb.

2. The Avatar Server (See Figure 4) also uses a RHCS with its y-axis perpendicular to the floor in the virtual world, and works in cms.

3. The VR Station process displays the avatar's present posture. The VR/IS model format (also called .hi format) is a RHCS, with the z-axis perpendicular to the floor and works in inches.

In the Avatar Server, we use a routine (called the traverseTree routine) in our implementation to compute the coordinates and orientation of the joints in the VR/IS model format. This routine goes through the skeleton-tree, uses the lower torso value as the root, and can determine the position of any joint on any part of the tree. It can build the information for every joint, depending upon the respective local coordinate system orientation. Thus, this routine provides a convenient way to determine the joint positions and orientation for a posture at any time.

The matrices, obtained by using the traverseTree routines, are in VR/IS model format, i.e. z-axis is pointing upwards from the floor, and is in inches. When we want to convert values from Avatar Server format to the VR/IS model format, we first transform the z-axis to point up (using the zUp transform), and then scale from cms to inches. To convert the traverseTree matrices from VR/IS model format to Avatar Server format, we scale from inches to cms, and then use the inverse of the zUp transformation. These transformations ensure that the values are in the proper coordinate system in which we are working, and provide mapping of the physical room and the virtual space.

USING TWO SENSORS TO SOLVE FOR THE ARM

For clarity, we revisit the VR/IS model format. It defines the human-skeleton by creating a tree, with the root as the lower torso (joint 4 in Figure 1). The kinematic chains for the lower and upper body emanate from the lower torso (root). The dimension and geometry of the human-body parts are already known as they are defined in the VR/IS model format. The only thing one has to determine to specify a posture is the orientation of the local coordinate system for line segments (limbs) in the tree. This orientation is captured by *animate_transform* for a joint, which is a 4X4 homogeneous, orthonormal matrix with zero translation. The convention is that the z-axis of this RHCS lies along the line-segment from one joint to another.

We work in the Avatar Server coordinate system which is in cms with the y-axis pointing upwards perpendicular to the floor in the virtual world. Once we find the orientation of the local coordinate system for the joints of the arm, then these animate transforms can be plugged into the VR/IS model format, and the posture can be displayed (See Figure 4). This is done by sending the animate transforms over the network in the VR/IS packet. These values are in turn picked up by the VR Station process for displaying the avatar. Once the animate-transforms are found, the VR Station process uses the VR/IS model format to draw the posture of the avatar at any given

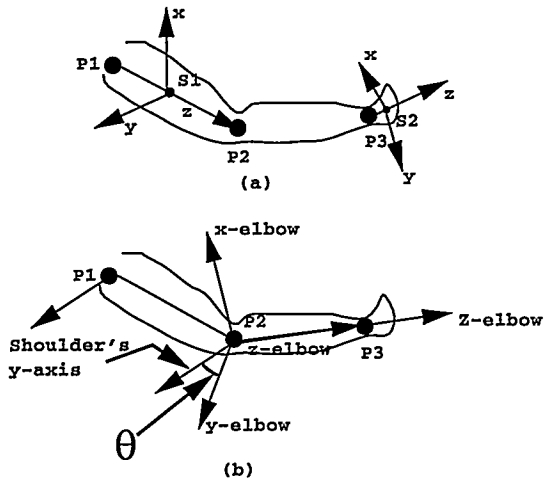

Figure 5: Solution using two sensors

moment.

The method for finding the orientation of the local coordinate system for the joints of the arm is as follows (See Figure 5):

- Translate the sensor frame (s1) on the upper arm, along P1P2 towards P1 by moving a defined distance *distanceToShoulder*. This gives the position of the shoulder, and the animate transform (i.e. the orientation of the local coordinate system) for the shoulder.

- Translate the sensor frame (s1) along P1P2 towards P2 by moving a defined distance *distanceToElbow*. This defines the position of the elbow; but we still need to find the orientation (animate transform) for the elbow, which in turn is based on the wrist position.

- Use defined distance *distanceToWrist* to translate the sensor frame (s2) on the palm along the negative z-axis of the sensor frame s2, and find the position and the orientation of the wrist's local coordinate system (i.e. the animate transform for the wrist).

- Next find the animate transform for the elbow. We know the elbow position, wrist-position, and the shoulder position. First, define the z-axis as the unit vector from elbow to the wrist, call it z_{elbow}. Use the shoulder's y axis to be another vector (y_1). These two vectors are sufficient to define a right handed coordinate system or the animate transform of the elbow, by using two cross products.

  We observe that the movement of the joint at the elbow is such that the Shoulder's y axis (or y_1 vector) and the elbow's y-axis should make ninety degrees in the worst case. This angle is denoted by θ in Figure 5(b). When the arms are comfortably on the side near the thighs in a natural standing pose, this angle θ is zero in our convention and these two y-axes are parallel to each other. When the arm is moved, the flexion/extension about the elbow, the rotation of the lower arm about the elbow with respect to the upper arm, or the supination/pronation of the hand – all of these actions change the relationship between these two y-axes (and so the angle θ). However the angle θ between these two y-axes remains less than ninety degrees in all cases (unless the person is unusually flexible). If the angle is more than ninety degrees by the above calculation, we swap the two vectors before taking the cross product. This ensures that the angle between the two y-axes is not more that ninety degrees. This musculoskeletal constraint provides the animate transform for the elbow joint (lower arm). This completes our closed form solution for the arm calculation using two sensors. We note that this solution does not restrict any degree of freedom for the elbow or the wrist joint (as in Tolani's closed form solution [36]).

- The calculated position of the shoulder, elbow and the wrist also provide the scaling information for the VR Station. The scaling is performed along the z-axis. The limbs could be scaled along the z-axis of the local coordinate system for the joint, based upon these estimated values. Thus the difference between the length of the limbs of the individual participants could be reflected in their avatar, by comparing with the corresponding default limb-lengths, and scaling that limb along the z-direction appropriately.

While running the experiments, we found that one could also make minor adjustments, and fine tune the physical appearance of the avatar by moving the position of the sensor on the upper-arm, physically along the sensor's z-axis (which is by our design parallel to the z-axis of the limb's local coordinate system). Any change in the values of these sensors can have an effect on the estimated position of the shoulder, elbow and the wrist joints for the avatar using the above algorithm. In other words, the user could fine tune the avatar's dimensions interactively, if desired. Figure 6 shows various postures for our solution.

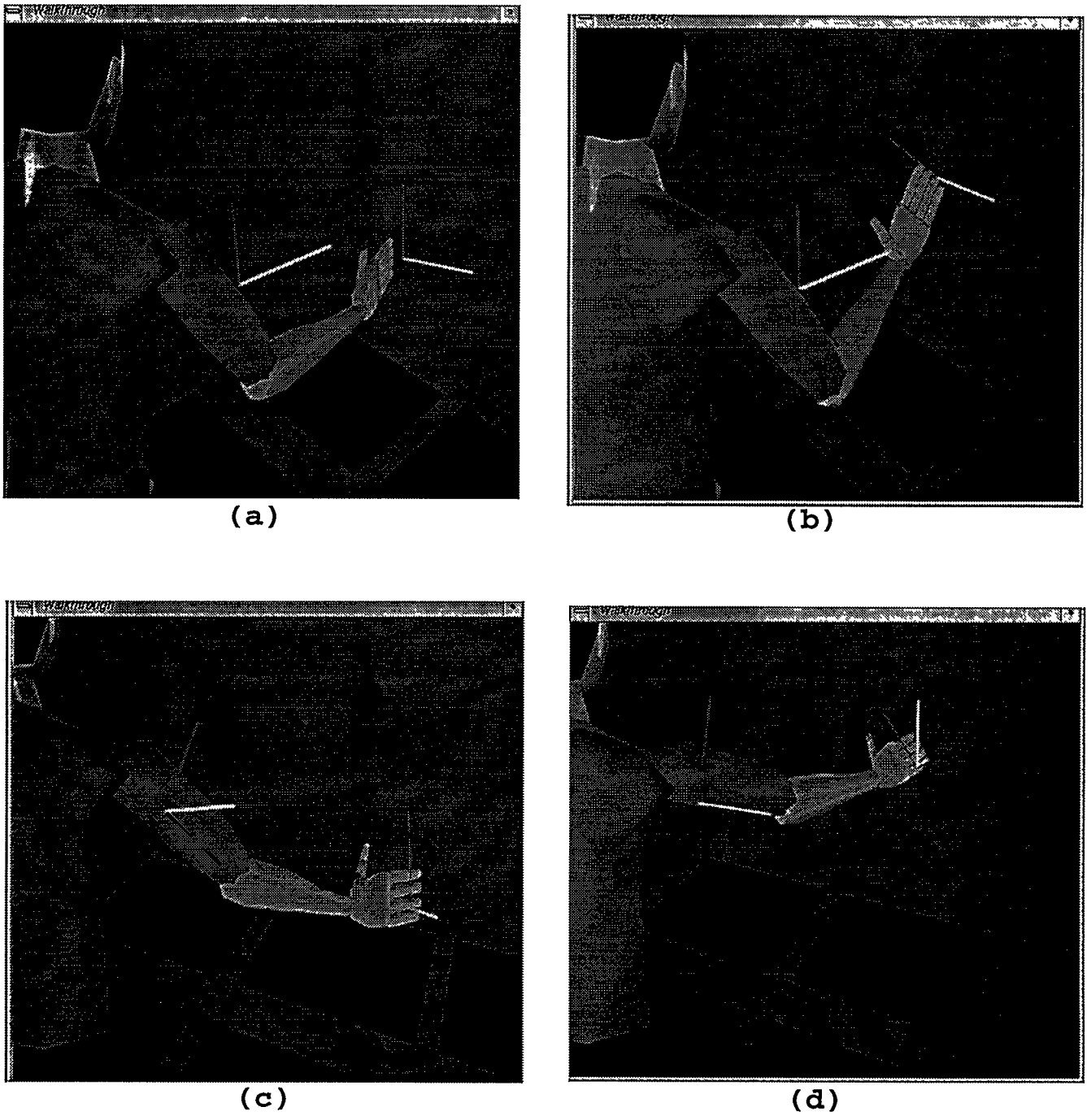

Figure 6: Various positions of the avatar hand.

We note that the solution presented above, is directly applicable to the four kinematic chains for arms and legs which are shown in Figure 2. These chains are – from the left shoulder to the left palm, from the right shoulder to the right palm, from the left hip to the left toe, and from the right hip to the right toe. We could easily extend the above solution for both the legs for our ten sensors orientation shown in Figure 2.

Solution of the complete hierarchy: Geometric Searching

We are now developing algorithms to find the location and orientation of the human joints in the upper-body, surrounded by the head position sensor, the lower torso sensor, and the two shoulder joints, in the kinematic chain hierarchy (see Figure 7). We plan to use a geometric and localized searching algorithm. This algorithm would eliminate several orientations of the limbs which are not possible, but are geometrically feasible. For example, note that joint 16 would always be inside at least one of the three 2D-convex hulls created by projecting the following four points onto x-y, y-z, and z-x planes: the position of left and right shoulders, the waist, and the head position. Therefore search outside this area could be limited. Since the length of the limbs is known, as well as the head position (sensor B in Figure 2), and the positions of the left and right shoulders have been estimated, we now have three kinematic chains. All three chains end at joint number 16 starting from the headPosition (sensor B), the left, and the right shoulders respectively. This algorithm is currently under development.

We are also looking into working with eight sensors by eliminating the two thigh sensors, using the lower torso position, and applying a geometric searching algorithm similar to that mentioned above. We could also use our two sensor closed form solution for the eight sensor case by moving the sensor on the feet to just above the ankles to obtain a chain similar to that for the arm.

Lastly, the position of the joints of the spine need to be estimated. Intricate and realistic postures of avatar mimicking the participant are only possible by determining the exact simulation of the spine-joints. However, real-time interaction is the driving focus for our research, and therefore, as in Badler's work [10], a reasonable linear interpolation algorithm is planned. A linear interpolation algorithm could start from joint 16 and end at the lower torso, in turn varying the corresponding axes linearly between the two end positions. We could also use a hermite curve [37], by defining three curves (one for each of the 3 axes). The two end points could still be the joint number 16 and the lower torso, and the two end-vectors are the respective vectors at the two end points. Similarly two hermite curves would be defined for the other two axes. The parameter (t) values could be based upon the values of the avatar's dimensions of the joints in the spine, scaled by the user's limbs dimensions.

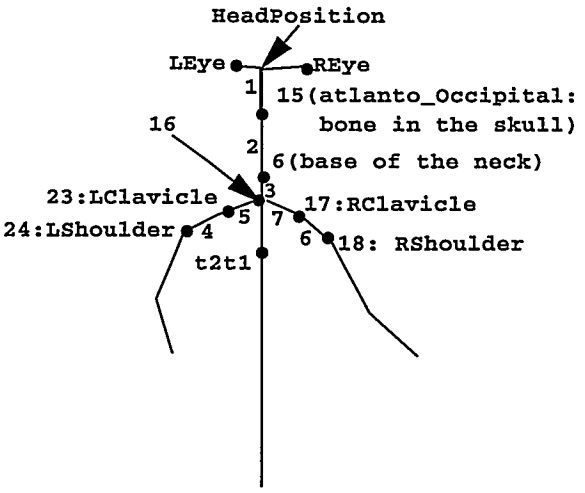

Figure 7: Kinematic chains of the neck area

CONCLUSIONS AND FUTURE RESEARCH

In this paper, we have provided a new closed form solution for the avatar's arms and legs. This solution does not restrict any degree of freedom for the human-arm as in [36]. The novelty of this method is that it uses musculoskeletal constraints to solve for the arms (or legs) with two sensors. We have also laid the groundwork for providing a solution for the whole human skeleton using eight or ten sensors. We believe that our work will lead to a new class of motion algorithms where only a limited number of sensors need to be used for real-time representation of the human form in a virtual environment with multiple participants. We have also proposed a new notion of geometric searching algorithms with culling based on natural constraints of the human posture. By breaking the human skeleton in smaller sub-chains, we

have opened the possibility of finding solutions in parallel for these sub-chains in our distributed environment. We believe that this will lead to a real-time solution.

ACKNOWLEDGMENTS

Special thanks are due to Dan Shawver for explaining the workings of the distributed environment. We would also like to thank Dave Rogers, Deepak Tolani, Xinmin Zhao, Meisha Collins, Denise Carlson and James Singer for their invaluable discussions. This work was performed at Sandia National Laboratories and was supported by the US Department of Energy under Contract DE-AC04-94AL85000.

References

[1] R Rivlin and K Gravelle. Deciphering the Senses, The expanding world of human perception. Chapter 1, pp. 9-28. Simon & Schuster, Inc. New York, 1984.

[2] MW Krueger. Artificial Reality II, Addison Wesley Publishing Company, Reading, MA. 1991.

[3] GJ Monkman. An Electroheological Tactile Display *PRESENCE* 1(2):219-228, Spring 1992.

[4] K Meyer, HL Applewhite, and FA Biocca. A Survey of Position Trackers. *PRESENCE*, 1(2):173-200, Spring 1992.

[5] MR Macedonia, MJ Zyda, DR Pratt, DP Brutzman, and PT Barham. Exploiting Reality with Multicast Groups, *IEEE CG&A*, 15(5):38-47, September 1995.

[6] S Stansfield, N Miner, D Shawver, and D Rogers. An Application of Shared Virtual Reality to Situational Training, *VRAIS 1995*, 156-161, 1995.

[7] S Stansfiled, D Shawver, D Rogers, and R. Hightower. Mission Visualization for Planning and Training, *IEEE CG&A*, 15(5):12-14, September 1995.

[8] S Stansfiled. A Distributed Virtual Reality System for Situational Training, *PRESENCE*, 3(4):360-366, Fall 1994.

[9] D Shawver and S. Stansfiled. VR/IS Lab Virtual Actor Review, *Proc first Workshop on Simulation and Interaction in Virtual Environment*, Univ of Iowa, 1995.

[10] NI Badler, MJ Hollick, and JP Granieri. Real-Time Control of a Virtual Human using Minimal Sensors, *PRESENCE*, 2(1): 82-86, 1993.

[11] JP Granieri, J Crabtree, and NI Badler. Production and Playback of Human Figure Motion for 3D Virtual Environments, *VRAIS 1995*, 127-135 1995.

[12] CD Clemente. Anatomy A Regional Atlas of the Human Body. Urban & Schwarzenberg, Baltimore-Munich, 1987.

[13] J Chadwick, D Haumann, and R Parent. Layered construction for deformable animated characters, *Computer Graphics*, 23(3):234-243, SIGGRAPH 1989.

[14] NI Badler, KH Manoochehri, and G Walters. Articulated Figure Positioning by Multiple Constraints, *IEEE CG&A*, 7(6):28-38, 1987.

[15] SK Semwal, JK Armstrong, DE Dow, FE Maehara. MultiMouth Surfaces for Synthetic Actor Animation, *The Visual Computer*, 10(7):399-406 1994.

[16] N Magnenat-Thalmann and D. Thalmann. Complex Models for Animating Synthetic Actors, *IEEE CG&A*, 11(5), 32-44 (1991).

[17] TD Pueyox. Human Body Animation: A Survey, *The Visual Computer*, 3(5), 254-264, 1988.

[18] D Zeltzer. Representation of Complex Animated Figures, *Graphics Interface 1982*, 205-211, 1982.

[19] D Zeltzer. Task Level Graphical Simulation: Abstraction, Representation, and Control. In *Making them Move: Mechanics, Control, and Animation of Articulated Figures*, Morgan Kaufmann, San Mateo, California, pp. 3-33 (1991).

[20] T Calvert. Composition of Realistic Animation Sequences for Multiple Human Figures, *Making them Move: Mechanics, Control, and Animation of Articulated Figures*, Morgan Kaufmann, CA, 35-50, 1991.

[21] WW Armstrong and MW Green. The dynamics of articulated rigid bodies for purpose of animation, *The Visual Computer*, 1(4):231-240, 1985.

[22] J Williams. Dynamic Experiences. *In Making them Move: Mechanics, Control, and Animation of Articulated Figures*, Morgan Kaufmann, San Mateo, California, pp. 265-280, 1991.

[23] PM Isaacs and MF Cohen. Mixed Methods for complex Kinematic Constraints in Dynamic Figure Animation. *The Visual Computer*, 4(6):296-305, 1988.

[24] NI Badler, CB Phillips, and BL Webber. Simulating Humans Computer Graphics Animation and Control, Oxford University Press, 1993.

[25] MW Lee and TL Kunii. Animation Design - A Database Oriented Animation Design Method with a Video Image Analysis Capability. In: Magnenat-Thalmann N, Thalmann D (eds) State of the Art in Computer Animation, Springer, Tokyo, pp. 97-112, 1989.

[26] SK Semwal. A Proposal for using ANNs for CG Animation. *The Journal for the Integrated Study of Artificial Intelligence, Cognitive Science and Applied Epistemology*, 10(1-2):93-106, 1993.

[27] N Magnenat-Thalmann, and D Thalmann. Human Body deformations using Joint Local Dependent Operators and Finite-Element Theory. In *Making them Move: Mechanics, Control, and Animation of Articulated Figures*, Morgan Kaufmann, San Mateo, California, pp. 243-264 (1991).

[28] K Komatsu. Human Skin Model capable of Natural Shape Variation, *The Visual Computer*, 3:265-271, 1988.

[29] C Loop and T DeRose. Generalized B-spline Surfaces of Arbitrary Topology, *Computer Graphics*, 24(4):347-356, 1990.

[30] FI Parke. Parametric Models for Facial Animation. *IEEE CG&A*, 2(6):61-68, 1982.

[31] S Muraki. Volumetric Shape Description of Range Data using Blobby Model, *Computer Graphics*, 24(4):227-235, 1991.

[32] GL Graves. The magic of metaballs, *Computer Graphics World*, 26-32, May 1993.

[33] DT Chen and D Zeltzer. Pump it Up: Computer Animation of a Biomechanically Based Model of Muscle Using the Finite Element Method, *Computer Graphics*, 26(2):89-98, 1982.

[34] SK Semwal and J Hallauer. Biomechanical Modeling: Implementing Line-of-Action Algorithm for Human Muscles and Bones using Generalized Cylinders, *Computers and Graphics*, 18(1) 1994.

[35] M Carignan, Y Yang, N Magnenat-Thalmann, and D Thalmann. Dressing Animated Synthetic Actors with Complex Deformable Clothes, *Computer Graphics*, 26(2):99-104, 1992.

[36] D Tolani. Inverse Kinematics of the Human Arm, draft of an internal report from University of Pennsylvania, PA, 1995.

[37] ME Mortenson. Geometric Modeling, *John Wiley & Sons Inc.*, 1985.

DISCLAIMER