Proceedings of the 2000 Winter Simulation Conference
J. A. Joines, R. R. Barton, K. Kang, and P. A. Fishwick, eds.

AVATAR KINEMATICS MODELING FOR TELECOLLABORATIVE VIRTUAL ENVIRONMENTS

Cristian Luciano
Pat Banerjee

Industrial Virtual Reality Institute
Department of Mechanical Engineering (M/C 251)
2039 Engineering Research Facility (ERF)
University of Illinois at Chicago
842 West Taylor Street
Chicago, IL 60607-7022, U.S.A.

ABSTRACT

This paper introduced the application of a more efficient mathematical representation of the kinematics of avatars, or digital human beings, in telecollaborative virtual reality environments (VRE). The human head, torso and arms were modeled as a redundant eight-degree-of-freedom kinematics structure using an excellent alternative tool to transformation matrices, called dual quaternions. This approach achieves an extremely fast and accurate iterative algorithm that converges to one possible solution of this inverse kinematics problem. The method was implemented and tested in a CAVE Automatic Virtual Environment to demonstrate its performance in real time.

1 INTRODUCTION

The goal of this project is to allow real time telecollaborative activities between two or more people geographically distant, manipulating virtual parts, products or facilities, through avatars that perform human movements in interconnected VRE. This would enable specialized engineers and designer to analyze in detail the performance of complex activities such as collaborative operation of machinery and equipment, part assembly or virtual manufacturing, incorporating human factors in their simulations (Brown A. S. 1999).

The idea is to use the minimum number of sensors per participant. Only three sensors are sufficient to track the movements of the user in a simplified model of the upper portion of a human being. One of the sensors is attached to the head and the other two, to each hand. These sensors continuously give information about its position and orientation in each VRE. The inverse kinematics algorithm locally creates the connection between the tracked human head (connected to torso) and hands, allowing the avatar to be shown in the remote VRE.

In order to successfully execute real-time interactivity between the participants, this algorithm should be fast enough to find a solution each time they move their head and hands. Unfortunately it is impossible to get an analytical close-form solution to more than six-degree-of-freedom kinematics structures as the human body (Craig 1986). Therefore, a numerical iterative algorithm based on the manipulator Jacobian and the Newton's method to find roots can be used (Chin 1996).

Traditionally homogeneous matrices were used because of their simple and well-known representations of rotations and translations. However, this method presents many inconveniences (Dam, Koch and Lillholm 1998), especially the cost of processing when several matrix products must be calculated in each iteration. This makes it difficult to achieve real time performance. The use of a different and more sophisticated concept derived from quaternions (Hamilton 1853, 1899), called "dual quaternions", allows you to model and solve the kinematics problem without compromising real-time user interactivity.

A methodology and its application to solve the redundant kinematic structure of avatars are described in this paper. This is followed by its implementation in telecollaborative VRE.

2 QUATERNIONS AND DUAL QUATERNIONS

A quaternion Q, which represents a rotation by an angle θ about the axis given by the unit vector $\mathbf{R} = (r_1, r_2, r_3)$, is a four-component number consisting of a scalar part and three orthogonal parts, defined as:

$$Q = Q_1 i + Q_2 j + Q_3 k + Q_4$$

where:

$$i^2 = j^2 = k^2 = ijk = -1$$

$$Q_1 = r_1 \; s \qquad (1)$$

$$Q_2 = r_2 \; s$$

$$Q_3 = r_3 \; s$$

$$Q_4 = c$$

where

$$s = \text{Sin}\,[\theta/2]$$
$$c = \text{Cos}\,[\theta/2]$$

While quaternions are very useful to represent orientations of a rigid body, they do not contain any information about its position in the 3D space. However there is a way to combine both rotation and translation in a unified notation. That way is using dual quaternions, whose elements are dual numbers.

A dual number **N** (Study 1903) is defined as the addition of a real and a dual part:

$$N = n_0 + \varepsilon \, n_1$$

where $\mathbf{n_0}$, $\mathbf{n_1}$ are real numbers and $\boldsymbol{\varepsilon}$ is an imaginary number such that $\boldsymbol{\varepsilon}^2=0$.

Extending the concept of dual numbers to quaternions, it is possible to define the dual number quaternion, or dual quaternion, **Q** as:

$$Q = q_0 + \varepsilon \, q_1$$

Given a normalized rotation axis $\mathbf{R}=\{r_1, r_2, r_3\}$, an angle $\boldsymbol{\theta}$ and a 3D translational vector $\mathbf{T}=\{dx, dy, dz\}$, the real part $\mathbf{q_0}$

$$q_0 = \{Q_1, Q_2, Q_3, Q_4\}$$

is defined by the formulae (1), and the dual part $\boldsymbol{\varepsilon}\,\mathbf{q_1}$, by:

$$\begin{aligned} q_1 = \{ \quad & \tfrac{1}{2} (dx \;\; c + (dy \;\; z - dz \; y) \; s), \\ & \tfrac{1}{2} (dy \;\; c + (dz \;\; x - dx \;\; z) \; s), \\ & \tfrac{1}{2} (dz \;\; c + (dx \;\; y - dy \;\; x) \; s), \\ & -\tfrac{1}{2} (x \;\; dx + y \; dy + z \; dz) \; s \qquad \} \end{aligned}$$

where

$$s = \text{Sin}\,[\theta/2]$$
$$c = \text{Cos}\,[\theta/2]$$

Dual quaternions allow you to replace the 4x4 homogeneous matrices by an 8-dimensional vector that corresponds directly to the screw 3D-transformation representation shown in Figure 1. (Goddard and Abidi 1998)

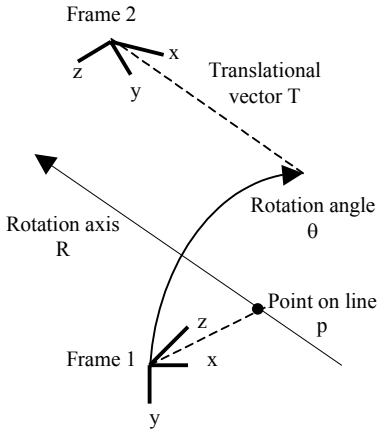

Figure 1: Diagram Showing the 3D Transformation Between Two Reference Frames Given by a Dual quaternion

Product between dual quaternions and conversions from and to transformation matrices can be defined (Walker and Shao 1991).

For simplicity, in this paper the dual quaternion Q will be expressed by Q(R, θ, T).

3 CREATION OF THE AVATAR

In order to create a well-dimensioned anthropomorphic avatar that correctly matches the body of the person being represented in the VRE, it is necessary to measure his/her body. A possible way to do this is comparing the location of the sensors attached to the body and calculating their distances about each axis (x, y and z). Of course this procedure must be performed only at the very beginning of the runtime of the application.

To succeed a fast, simple and automatic measurement the person has to stand up in a natural pose with the right arm straight down the side, close to the body, and then

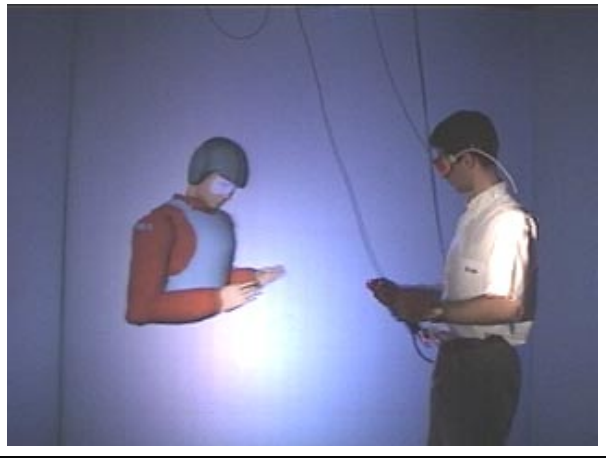

Figure 2: Initial Joint Configuration

flexing the elbow 90 degrees to the front (as shown in Figure 2). This will be the initial position for the kinematics algorithm, where all joint angles are zero.

This procedure takes into account only the right arm, assuming that both arms have the same length.

Given:

S1 = position of the head sensor
S2 = position of the right hand sensor

Shoulder = Abs (S2[x] – S1[x])
Neck = Abs (S2[z] – S1[z]) * 0.5
Arm = Abs (S2[z] – S1[z]) * 0.5
Forearm = Abs (S2[y] – S1[y]) * 0.7
Hand = Abs (S2[y] – S1[y]) * 0.3

The proportionality factors used here are given by heuristic ergonomic measurements.

4 AVATAR FORWARD KINEMATICS MODELING

The human arm and torso can be modeled as an eight-DOF structure, based on the previous work by Tolani and Badler (1996). Placing a fixed coordinated system at the chest and moving coordinate systems at the joints of the torso, shoulder, elbow and wrist, eight dual quaternions (McCarthy 1990) are defined as follows:

$$Q_i = Q\ (R_i, \theta_i, \varnothing) * Q\ (\varnothing, 1, T_i)$$

where:

R_i = rotation axis of the joint i
θ_i = rotation angle of the joint i
$\varnothing = \{0,0,0\}$
T_i = translation to the joint i+1

Then the transformations that describe the avatar forward kinematics are: (2)

$Q_1 = Q\ (\{0,0,1\},\theta_1+Head,\varnothing) * Q\ (\varnothing, 1,\{Shoulder,0, 0\})$
$Q_2 = Q\ (\{0,1,0\},\theta_2,\varnothing)$
$Q_3 = Q\ (\{1,0,0\},\theta_3,\varnothing)$
$Q_4 = Q\ (\{0,0,1\},\theta_4,\varnothing) * Q\ (\varnothing, 1,\{0,0, -Arm\})$
$Q_5 = Q\ (\{1,0,0\},\theta_5,\varnothing) * Q\ (\varnothing, 1,\{0, Forearm, 0\})$
$Q_6 = Q\ (\{0,1,0\},\theta_6,\varnothing)$
$Q_7 = Q\ (\{1,0,0\},\theta_7,\varnothing)$
$Q_8 = Q\ (\{0,0,1\},\theta_8,\varnothing) * Q\ (\varnothing, 1,\{0, Hand, 0\})$

where *Head* is its rotation angle about the axis z, given by the sensor located on the stereo shutter glasses.

5 ITERATIVE ALGORITHM TO SOLVE THE INVERSE KINEMATICS

The algorithm used to solve the avatar inverse kinematics is based on the traditional Newton's method for nonlinear systems. This method generally gives quadratic convergence and a possible solution is found in a few iterations (Burden and Faires 1989).

In order to avoid the computational cost involved in the processing of transformation matrices, dual quaternion algebra is introduced to calculate, in each iteration, the manipulator Jacobian and estimate the direction vector or end-effector velocity.

Basically this method involves the following steps:

1. Calculate the current end-effector position and orientation, given by the product of the eight dual quaternions $\mathbf{Q_i}$ obtained during the forward kinematics.
   $K_8 = Q_8 * Q_7 * Q_6 * Q_5 * Q_4 * Q_3 * Q_2 * Q_1$
2. Compute the Jacobian **J** for the current joint configuration $\mathbf{q_k}$
3. Estimate the end-effector velocity (differential change) V_k
4. Invert the Jacobian **J** and solve the system of linear equations for the joint rates:
   $q_k' = J^{-1}(q_k)\ V_k$
5. Compute the new joint angles by $q_{k+1} = q_k + q_k'$
6. Set k = k+1
7. Repeat until $\Sigma\ V_k <=$ tolerance factor

5.1 Manipulator Jacobian Calculation

The manipulator Jacobian **J** is a matrix that describes the relationship between joint velocities **q'** and the end-effector velocity **V**, taking into account the current joint angles **q** (Paul, 1981).

$$V = J(q)\ q'$$

where

$$J(q_i) = [\psi x_i, \psi y_i, \psi z_i, dx_i, dy_i, dz_i]^T$$

defines the angular $\boldsymbol{\psi}$ and linear **d** velocity.

There are numerous traditional methods for computing the Jacobian matrix using transformation matrices, for example the formulation presented by Zomaya (1992). However, using dual quaternions the procedure is computationally faster and extremely simple to implement.

In quaternion algebra, any point **p** could be rotated to **p'** by an angle **θ** about the axis defined by certain unit vector.

$$p' = Q * p * Q^{-1}$$

where **Q** and $\mathbf{Q^{-1}}$ represent the quaternion and its conjugate respectively (Bobick, 1999).

Similarly, the 3D line S could be rotated and translated to S' by the dual quaternion Q.

$$S' = Q * S * Q^{-1}$$

Since the end-effector is rotated and translated consecutively about its eight joints, the product between the dual quaternions $\mathbf{Q_i}$ obtained by the formulae (2), will give the position and orientation of each joint.

Finally, defined the rotation axis **S** of each joint, the column **i** of the Jacobian **J** can be computed simply by the following product:

$$J_i = K_i * S_i * K_i^{-1}$$

where:

$$K_i = Q_i * Q_{i-1} * \ldots * Q_1$$

$$S_i = Q\,(R_i, 0, \varnothing)$$

$$R_i = \begin{cases} \{1,0,0\} & \text{if joint } i \text{ rotates about axis x} \\ \{0,1,0\} & \text{if joint } i \text{ rotates about axis y} \\ \{0,0,1\} & \text{if joint } i \text{ rotates about axis z} \end{cases}$$

$$\varnothing = \{0,0,0\}$$

5.2 End-Effector Twist Estimation

In order to push the arm to the desired position and orientation, it is necessary to estimate the twist of the end-effector, guaranteeing the convergence of the algorithm.

Given the dual quaternion **C** that represents the current end-effector position and orientation, and **D** that denotes the desired end-effector position and orientation, it is possible to find the transformation **Q** as follows:

$$Q = D * C^{-1}$$

Q is associated with a continuous screw motion with fixed axis, which maps **C** to **D**. **Q** can be converted back to rotation axis **R**, angle **θ** and translational vector **T**.

R holds the information about the direction of the screw axis with a rotation angle **θ**. Then, the end-effector angular velocity **av** is:

$$av = \theta * R \qquad (3)$$

On the other hand, since the product **R.T** gives the amount of the translation **t** in the direction of **R,** the moment vector M of the screw axis is given by:

$$M = \tfrac{1}{2}\,(T \times R + \text{cotangent}\,[\theta/2]\,(R \times T) \times R)$$

where the cross product is denoted by x (Wagner 1998).

It is possible to calculate the end-effector linear velocity lv as:

$$lv = t*R + \theta*M \qquad (4)$$

Finally combining the formulae (3) and (4), the estimated end-effector velocity **V** will be:

$$V = \{av, lv\}^T$$

5.3 Solving of the System of Linear Equations

Given the end-effector velocity **V**, the inverse of the Jacobian would allow you to calculate, in each iteration, the corresponding joint velocities simply solving a system of linear equations. This could be used to deduce a gradient-based iterative technique for solving the inverse kinematics, pushing the arm to the desired position and orientation.

$$q' = J^{-1}(q)\,V$$

Due to the fact that **J** is a non-square 8x6 matrix, there are fewer equations than unknowns. In this case, there are an infinite number of possible solutions. To solve this underdetermined system of linear equations, the least square factorization can be applied using the QR decomposition method (Burden and Faires 1989).

Finally, **q'** are added to the current joint angles **q**, becoming the new value of **q**. The iterative process continues until the end-effector velocity is less than certain tolerance factor.

6 IMPLEMENTATION

In order to demonstrate the performance of this new approach to solve the inverse kinematics of 3D avatars in real time, the method was developed and implemented using classes in C++. Manipulation and computation of dual quaternions was optimized in order to succeed high user-interactivity. The commercial library LAPACK.C++, by Rogue Wave, was used to solve the system of linear equations by the QR factorization method.

The avatar geometry, as well as the pivot points corresponding to its joints, was modeled in 3D Studio Max. The 3ds format models of each part of the avatar (head,

torso, arms, forearms and hands) are imported and converted to OpenGL by the application.

The resulting interactive virtual reality simulation was tested using the VR environment available at the University of Illinois at Chicago. Figures 3 and 4 show the real-time application running in the CAVE.

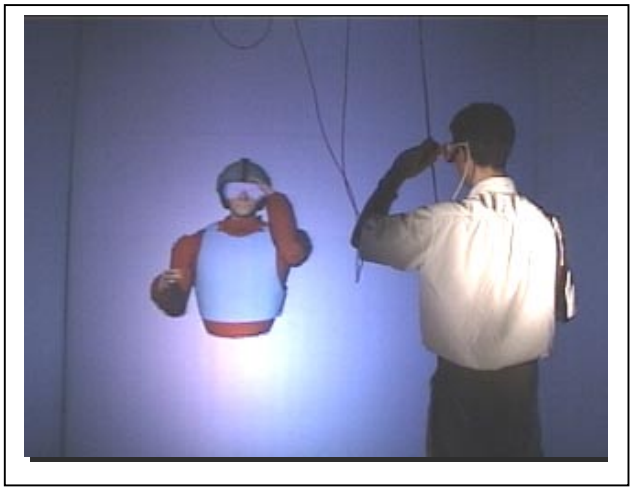

Figure 3: User and his 3D Avatar in the CAVE

7 CONCLUSIONS AND FUTURE RESEARCH

We have presented a fast, robust and accurate alternative approach to solve the inverse kinematics of avatars in virtual reality environments using dual quaternions. Although the method consists of an iterative algorithm, experimental results indicate that the performance is highly adequate for simulations in real time, guaranteeing maximum interactivity between many participants carrying out telecollaborative activities in remote interconnected CAVEs.

Analyses of repeatability, with circular and repetitive movements of the end-effector, have shown that it is convenient to reset the joint angles, before starting the iterative process, each time the desired end-effector position and orientation has changed. This allows the algorithm to find a possible solution that is closest to the initial joint configuration, thereby the joint limit constraints are automatically satisfied.

In theory, that strategy would attempt against real-time performance, since it would take fewer iterations to find the next joint configuration from the previous solution, without resetting the joint angles. However, laboratory tests have demonstrated the algorithm is extremely efficient at overcoming this limitation. It is not noticeable by the users and realistic looking postures are obtained throughout.

Extending the possibilities of this application, it is straightforward to incorporate recording capabilities of telecollaborative activities between participants located in remote CAVEs. In the recording phase, it would be necessary to store only the position and orientation of the three tracked sensors involved by participant. Then, during the re-playing phase the algorithm could be applied taking into account this pre-recorded information. In order to minimize the quantity of data to be stored, the recording process should be completed at a certain time interval, for example one second, and then interpolate subsequent sensor positions and orientations to recreate the complete smooth movement of the avatar. This could be successfully executed applying spline for the positions and squad quaternion interpolation for the orientations (Dam, Koch and Lillholm 1998).

A mathematical performance comparison between the algorithm introduced in this paper and the traditional method based on transformation matrices, as well as the recording of telecollaborative activities using avatars, will be carried out in future.

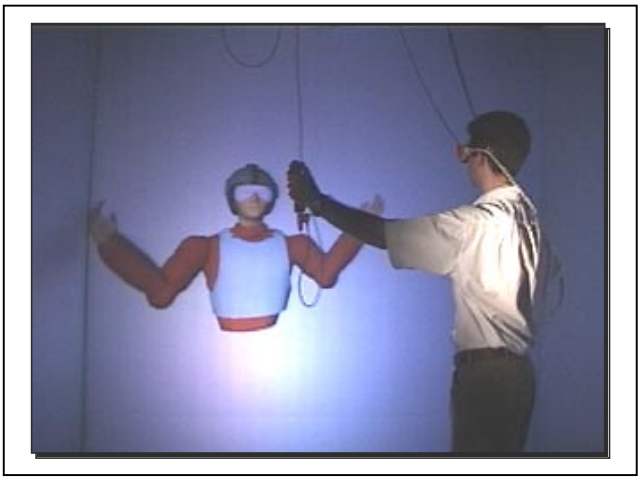

Figure 4: Real-Time Simulation Running in the CAVE

ACKNOWLEDGMENTS

The support granted by the Fulbright/CONICOR scholarship program (Argentina) and the Department of Mechanical Engineering at the University of Illinois at Chicago (UIC) is gratefully acknowledged.

The authors would like to thank Prof. Krishna Gupta, from the Mechanical Engineering Department at the UIC, for providing valuable and interesting suggestions during this research.

REFERENCES

Brown A. S. 1999. Role models: Virtual people take on the job of testing complex design. *Mechanical Engineering*: 44-49.

Bobick N. 1999. Rotating objects using quaternions. <http://www.gamasutra.com/features/programming/19980703/quaternions_01.htm>

Burden R. and Faires D. 1989. *Numerical analysis.* Fourth edition. PWS-KENT Publishing Co., Boston.

CAVE Automatic Virtual Environment <http://www.evl.uic.edu/EVL/VR/systems.shtml#CAVE>, University of Illinois at Chicago.

Chin, K.W. 1996. Closed-form and generalized inverse kinematic solutions for animated the human articulated structure. Bachelor's Thesis in Computer Science. Curtin University of Technology.

Craig J. 1986. *Introduction to robotics, mechanics and control*. Addison-Wesley

Dam E. B., Koch M. and Lillholm M. 1998. Quaternions, interpolation and animation. Technical Report DIKU-TR-98/5. University of Copenhagen, Denmark

Hamilton W. R. 1853. *Lectures on quaternions*. Hodges Smith & Co., Dublin

Hamilton W. R. 1899. *Elements of quaternions*, volumn 1-2 Longmans, Green and Co.

Goddard J.S. and Abidi M.A. 1998. Pose and motion estimation using dual quaternion-based extended Kalman filtering. *Proceedings of SPIE: Three-Dimensional Image Capture and Applications*, 3313.

McCarthy J. M. 1990. *An introduction to theoretical kinematics*. MIT Press, Cambridge, Massachusetts

Paul R.P. 1981. *Robot manipulators: mathematics, programming and control*. The MIT Press.

Study E. 1903. *Geometrie der dynamen*. Leipzig.

Tolani D. and Badler N. 1996. Real-time inverse kinematics of the human arm. *Presence*, vol. 5(4).

Wagner M. 1998. Advanced Animation Techniques in VRML 97. VRML '98 Symposium in Monterrey, California.

Walker M. W. and Shao L. Estimating 3-d location parameters uisng dual number quaternions. *CVGIP: Image Understanding*. vol. 54(3):358-367.

Zomaya, A. Y. 1992. *Modelling and simulation of robot manipulators: A parallel approach*. World Scientific.

AUTHOR BIOGRAPHIES

CRISTIAN J. LUCIANO is a Research Assistant in the Industrial Virtual Reality Institute at the University of Illinois at Chicago (UIC). He received his B. S. in Computer Science from Universidad Tecnológica Nacional, Córdoba, Argentina. He is currently pursuing his M.S. in Industrial Engineering at UIC. His interests include virtual reality, simulation, inverse kinematics and avatars. His email and web addresses are <clucia1@uic.edu> and <www_ivri.me.uic.edu/cristian>.

PAT BANERJEE is a Professor and Director of the Industrial Virtual Reality Institute at the University of Illinois (UIC). He received his B. S. in Mechanical Engineering from Indian Institute of Technology, and his M. S. and Ph.D. in Industrial Engineering from Purdue University. His email and web addresses are <banerjee@uic.edu> and <www.me.uic.edu/faculty/banerjee.html>.