SAND2000-2777C

Real-Time Tracking of Articulated Human Models Using a 3D Shape-from-Silhouette Method

Jason P. Luck

Colorado School of Mines, Dept. of Systems Engineering

Daniel E. Small

Sandia National Labs, Intelligent Systems and Robotics Center & University of New Mexico, Dept. of Computer Science

Abstract

This paper describes a system which acquires 3D data and uses the data to track an eleven degree of freedom human model in real-time. Using four cameras we create a time-varying volumetric image (a visual hull) of whatever object is moving in the space observed by all four cameras. We are currently operating the sensor in a volume of approximately 500,000 voxels (1.5 inch cubes) at a rate of 25 Hz. We are able to track the upper body dynamics of a human within the workspace. The system is able to track the x, y position of the body, a torso rotation about the z axis, along with four rotations in each arm. Tracking occurs using a method we developed to be extremely fast, which allows both data acquisition and tracking to occur on one computer at a rate of 16 Hz. We also developed a calibration procedure, which allows the system to be reconfigured or even moved and to quickly be recalibrated. Furthermore the system utilizes another computer to visualize either the voxel data overlaid with the joint locations or to view a human avatar, both of which are driven in real-time. Lastly our system has been implemented to perform crane gesture recognition, and has been linked with a large robotic arm to simulate crane movements.

Corresponding author
Jason Luck
Dept. Of Engineering Systems
Colorado School of Mines
1500 Illinios St.
Golden, CO 80401
USA

phone
Office (303) 273-3666
Home (303) 384-3557

Email
jluck@mines.edu

1 Introduction

Due to the enormous number of applications involving human-computer interaction, real-time 3D human motion-tracking has become a highly valued goal. Applications such as virtual reality, telepresence, smart rooms, human robot interaction, surveillance, gesture analysis, movement analysis for sports, and many others all have a need for real-time human motion-tracking. Accordingly there has been a lot of work done in this field. Most of the work has been done off-line, where images are acquired in real-time and analyzed at a later time [1,2]. However, this does not allow for any human-computer interaction. Therefore we must move to a real-time system. The approaches that work in real-time can be divided into two categories. The first category works in the 2D domain and attempts to get 3D information either from a single view or through combining 2D information from multiple cameras [3,4,5,6]. The problem with this type of approach is that modeling 3D articulated objects from 2D data is often an ill-posed problem due to occlusion/self-occlusion. The second category works directly in the 3D domain. At the time of this paper only one other team has attempted to work directly in the 3D domain [7]. In their work ellipsoids are fit to body segments that provide a coarse model of the body. Our system uses a similar data acquisition approach, but instead fits a stick model of the human so that joint angles can be recovered. In this way a detailed analysis of the human motion can be performed for a wide variety of applications. Specifically, our system is recovering the shoulder and elbow joint angles, which allows us to use a much richer set of pose-analysis functions.

The system developed in this paper acquires data of an area composed of 500,000 3D volume elements (voxels) with dimensions of 1.5 inches on a side at a rate of 25 Hz. The system then uses the data to track the upper body dynamics of a eleven degree-of-freedom (DOF) humanoid stick figure model at 16 Hz, which is fast enough to track very rapid movements. Currently our model incorporates the (x, y) horizontal location of the torso, a rotation of the torso about the vertical (z) axis, and 4 rotations for each arm (3 in the shoulder and 1 at the elbow). However, testing to expand this model to allow 6 DOF in the torso and include leg dynamics has already begun. When the user is in the workspace real-time visual feedback is provided to the user in terms of viewing the voxels overlaid with the joint locations or viewing a human avatar driven from the tracking results. Our system has also been implemented for a crane gesture analysis system and achieved excellent results.

This paper will first provide an overview of our approach and system architecture in section 2. Given that in section 3 the paper will explain our data acquisition system and present our new calibration algorithm, which allows for rapid setup of our system. Following that our new tracking algorithms are explained in section 4. Next the results of our system are discussed in section 5. A brief overview of an application follows in section 6. Lastly, conclusions and future work are discussed in section 7.

2 The System

Our system can be divided into three main components. The first acquires our 3D data using 4 cameras and a technique called shape-from-silhouette. The second component uses the data to fit an eleven DOF stick figure model to the data. The algorithm has to perform quickly in order to ensure real-time tracking. Therefore, we require the user to initialize the system by standing in a particular pose in which body parameters are measured. Once the system detects the initialization has been completed, it switches into a tracking algorithm. The tracking algorithm uses an approach where the voxels create forces to pull the model into alignment with the data. The last component uses the tracked model to provide feedback to the user and to perform analysis of the motion. First a real-time visualization system can either display the current voxels and the computed joint locations, or it can display a human avatar with the joint angles computed from the tracking algorithm. A second system reads in the joint positions from the tracking algorithm computes human-joint angles and performs gesture recognition to control a crane.

The hardware and software architecture of the system is robust and straightforward to implement. Figure 1 is a diagram of the hardware that is used. The first system is a 866 MHz Pentium3 computer with 4 image nation video cards connected to 4 cameras. This system is used for the image processing and volumetric calculations as well as tracking the humanoid model. The second system is a 450 MHz Pentium2 with a 3D accelerator and is used for visualization of the data. The systems are connected by a 100-megabit ethernet switch that allows for a direct, uninterrupted connection between the two computers. The software architecture is based around a client/server framework. It makes use of the Common Object

Request Broker Architecture (CORBA) for client/server communications, and multithreaded process partitioning.

3 The 3D Video Motion Detector

The system we are using to acquire 3D data is a real-time shape-from-silhouette sensor that we call the 3D Video Motion Detector, or 3DVMD. This sensor uses a combination of industry standard components including a high-end PC, four RS170 monochrome video cameras, and four PCI-bus frame grabber cards. Using this hardware we create a time-varying volumetric image of the visual hull of whatever object is moving in the space observed by all four cameras. We are currently operating the sensor in a volume of approximately 500,000 voxels (1.5 inches cubed) at a rate of 25 Hz.

3.1 Algorithm Description

The algorithm for performing the shape from silhouette involves extracting silhouettes from the four images using an adaptive background subtraction and thresholding technique. This algorithm indicates which pixels have changed from the background in each of the cameras, as seen in figure 1.

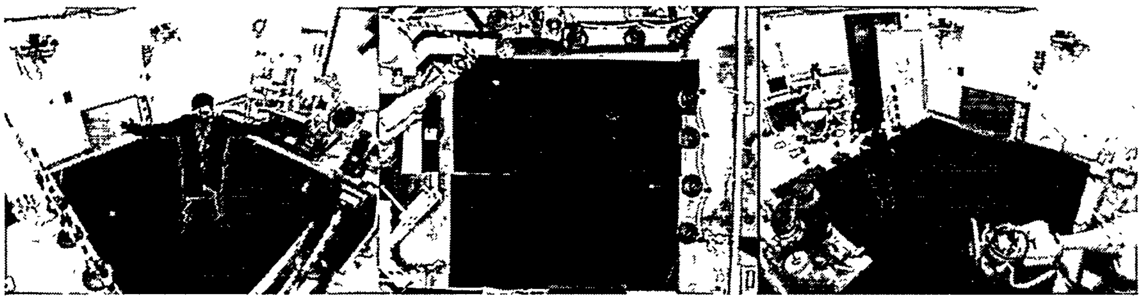

Figure 1 The 3 difference images used by the 3DVMD. The blue pixels represent new motion.

The calibration procedure creates look up tables that store voxel-pixel associations, which relate each voxel to a pixel in each camera. By traversing the voxels and examining the appropriate image-pixels we can tell which voxels are occupied. Voxels that are occupied will have the appropriate pixel in all of the cameras active. To speed the process of traversing through the voxels we are defining a subvolume around the region in which the person is moving, thereby minimizing the search. This results in a very fast, low-latency system that is appropriate for our tracking work, as well as for development of uninstrumented 3D user interfaces. Figure 2 shows an example of our data.

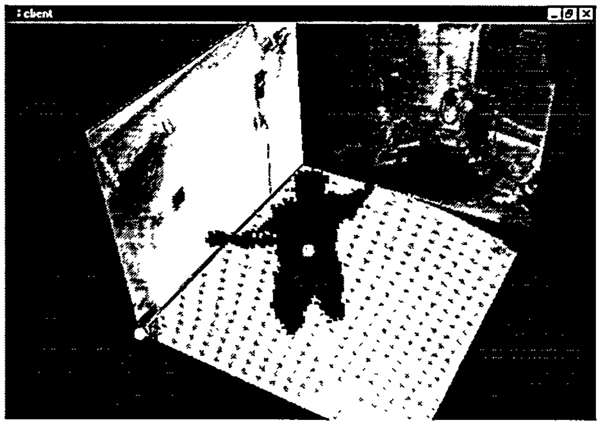

Figure 2 Example of 3DVMD data

3.2 *Calibration of the 3DVMD Sensor:*

The 3DVMD system relies on an accurate calibration of each camera's intrinsic and extrinsic parameters. We estimate parameters for each camera's intrinsic characteristics by holding an 8 by 10 inch checkerboard pattern by hand at a distance that fills most of the camera field of view. The paper target pattern is glued to a glass plate. A series of six images of this target are recorded from each camera, with the angles between the target plane and the camera at a variety of values. From this information we can determine the focal length, center pixel location, and the coefficients of radial distortion.[8] The next step is to estimate the camera's world-frame position. We place targets on the floor of the space we are observing and determine their image-plane locations with subpixel accuracy, as seen in figure 3. We then use the same planar calibration methods, but instead of solving for the intrinsic parameters, we now solve for the extrinsic parameters. Because all of the cameras observe the targets in the same positions, we are able to obtain the relative positions of each camera in a global coordinate frame. By using this analytical calibration method, we can now set these systems up much more rapidly than previous manual calibration methods.

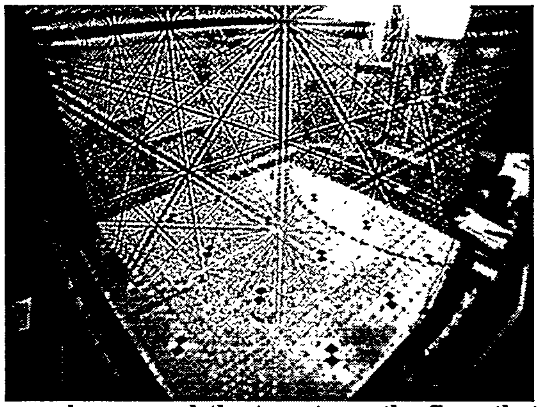

Figure 3 The image shows the workspace and the targets on the floor that are used to derive the extrinsic calibration of the cameras. From this camera calibration data, the volumetric points can be easily computed and projected back into the image plane.

4 Tracking the humanoid model

4.1 *Initialization*

To simplify tracking we require that the user performs an initialization pose upon entering the workspace. The pose is a simple cross formation with the user facing along the y axis with his arms extending parallel to the floor and straight out to the sides of the body (along the x axis). Once in this pose, the system measures the body parameters required for tracking: body radius, shoulder height, and arm length. Body radius is measured by gradually increasing the radius of a infinite cylinder (along z) placed at the center (median) of the voxels, until the number of new voxels within the cylinder drops off significantly. Arms are found by starting at the expected shoulder location (calculated from the height of the user, the direction the user is facing, and his body radius) and repeatedly growing to neighboring voxels outside of the body radius until no new voxels are found. We then check to ensure that the vectors from the last voxel found on each arm (the fingertip) to the shoulder locations are pointed in the correct direction for the initialization pose. Once in the initialization pose, the height of the shoulders and the lengths of the arms are recorded. The system then indicates that the initialization phase has ended and tracking can begin.

Currently the user must always begin by initializing the system. However, the model parameters can easily be saved to a file and read in before the user enters the workspace. In this way the user would only have to tell the system who he or she is and tracking can commence immediately. Our system already uses voice communication, therefore it would be easy to incorporate an option to simply tell the system your name while entering the workspace and skip the initialization phase.

4.2 Tracking

In order to speed tracking, the tracking procedure it broken into two phases. The current algorithm first finds the (x,y) location and heading (the direction the person is pointed) of the torso. We can then predict the location of the joints (currently just the shoulders), and assume that the locations remain constant for a particular torso orientation. These locations are then used to anchor models extending from the torso (currently just the arms). In this way we only need to solve for the angles at which the upper arms extend from the shoulders, and then for which the lower arms extend from the elbows.

Currently we only compute the x, y location of the torso and a rotation about z; accordingly we are assuming the user is standing straight up. The x, y location can easily be computed using the median value of all the voxels. The heading is computed by fitting an ellipsoid to all the data within the body radius of the x, y location. This is done by performing an eigen-decomposition on the moment matrix M. The eigenvectors of this matrix correspond to the principal axes of the ellipsoid. The principal axis closest in orientation to the last known heading is taken as the new heading of the person.

$$M = 4\begin{bmatrix} M_{xx} & M_{xy} & M_{xz} \\ M_{yx} & M_{yy} & M_{yz} \\ M_{zx} & M_{zy} & M_{zz} \end{bmatrix} \text{ where, } M_{ab} = \tfrac{1}{n}\sum_{i=1,n} a_i b_i \,.$$

To compute the angles for a particular arm segment, we simply compute the angles from each voxel that is not within the body radius to the anchor point. The result is a pulling force from the current orientation to a new orientation that passes through this voxel, as shown in figure 4. In this way each voxel can have an effect on all of the arm segments, which overcomes the problem of having to decide which arm segment a voxel belongs to.

However, we weight each pull by the distance, *d*, to each arm segment, so that a voxel exerts a stronger pull on closer arm segments than on those further away. For each voxel, let the minimum distance to any arm segment be denoted as d_{min}. The weight for each segment is given by $(d_{min} / d)^3$. Accordingly as the model is pulled into the correct orientation, the forces exerted from a particular voxel should be almost entirely on the segment closest to the voxel. This weighting strategy works extremely well for small adjustments, which is all it should have to make since our tracking rate is extremely high.

In addition we employ zero weighting to any arm segment further than a certain distance from the voxel (d_{max},computed from a maximum arm velocity divided by our tracking rate). Also, if a voxel is within a very small distance of an arm segment (less than d_{seg}), then we assume that it belongs to only that segment, and assign a zero weight to all other segments. The weighting assignment is shown below.

$$Weight = \left\{ \begin{array}{ccc} 1 & if & d = d_{\min} \\ (d_{\min}/d)^3 & if & d_{\min} < d < d_{\max} \\ 0 & if & d > d_{\max} \\ 0 & if & d > d_{\min} \,\&\, d_{\min} < d_{seg} \end{array} \right\}$$

In addition, for those points very close to the body radius we lessen the weights, because they could be noise or body voxels. Also, the weights of points are scaled by their distance to the anchor so that close points have a small weight. This is done because the calculated adjustment angle is larger for points with the same offset from the segment but closer to the anchor.

Once the adjustment is computed for all voxels outside of the body radius, an adjustment is made to each of the arm segment angles and the loop repeats. The process stops for each arm segment when either the adjustment is too small or the move was bad (fewer points are close to new orientation of the segment).

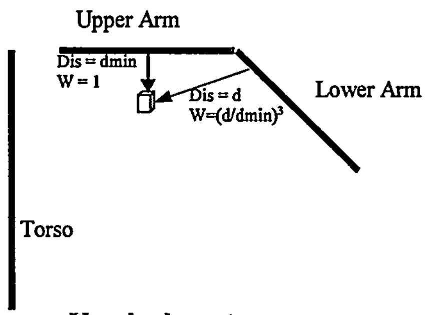

Figure 4 Example of alignment forces. Voxels closer to an arm segment exert a larger force on that segment than to arm segments further away (the weight to the upper arm is 1 while the weight to the lower arm, which is further away, is scaled down).

Currently our system stops tracking an arm when the arm lies along the body. If we hope to track the arms when they are extremely close to the body we must move to a much more precise torso model (work is in progress). In addition we will need data which has far less noise from shadows. Employing color cameras and using hue to distinguish shadows as done by Cheung and Kanade [7] might solve this problem.

The process also has a recovery algorithm in case it "loses" an arm. If too few voxels are close to one of the arm segments, the process assumes that the optimization has failed and attempts to grow the arm instead. Since we know the shoulder position, we start a growing algorithm from this point. Growing can include any neighboring point outside of the body radius, and continues until no new points are found. In this way the last point found should again be the hand and the elbow should be somewhere in the middle. Accordingly we compute the elbow to be at the voxel that was 2/3 of the way down the arm. Angles are calculated using these locations. The growing algorithm may not be able to find the arm if we have missing data (data acquisition has failed to find the arm voxels) or if the arm lies within the body radius. In this case our algorithm will leave the arm where it was last located and wait for the next set of data.

5 Results

The current system is able to collect data and track our model at 16 frames per second. Both data collection and model tracking occur on the same computer. Currently we are not able to indicate precisely how accurate our tracking is because we do not know the ground truth for the users movement within the workspace (see future work). However, the fit looks good when comparing the movement of the user and the avatar side by side *(a movie will be available on the web soon).* In addition we were able to use the joint angles to distinguish several gestures to control a crane as will be discussed in the application section.

There are two options for visual feedback while the user is within the workspace. He or she can either see the voxel data set with colored spheres to indicate the current joint estimations, or a human avatar that uses the joint angles computed from the current joint estimations. Both output displays shown in Figure 5 run in real-time.

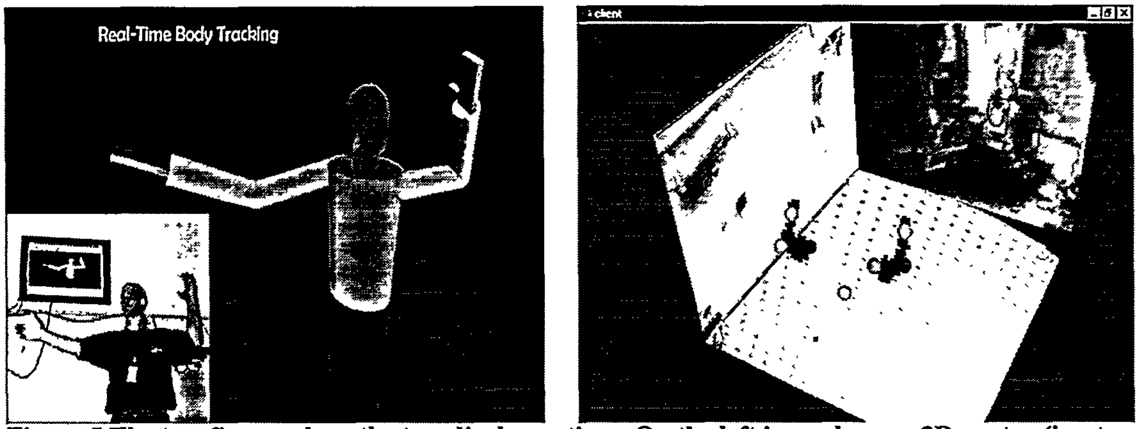

Figure 5 The two figures show the two display options. On the left is our human 3D avatar (inset shows our colleague Brian Rigdon in the workspace). On the right is a display of the voxels with large colored spheres representing joint positions (the body voxels have been removed).

6 Application

We incorporated our tracking algorithm into a system for crane gesture analysis. The system utilizes four cameras to form a workspace in front of a large robotic arm, which simulates a crane. A separate computer, reads in the same data as the visualization computer (the joint locations) and uses it to interpret gestures and control the crane. Also implemented on this computer is a voice recognition routine that allows verbal communication between the system and the user.

The process follows this pattern. When enough voxels appear to indicate a user within the workspace the computer says "Hello" and asks the user to verbally identify himself (eventually this will be used to skip the initialization step). The user is then asked to stand in the initialization pose and the system verbally communicates when initialization is done. Once initialization is complete, the user lowers his arms and the system indicates that gesturing can now begin. Once in this state, the system continuously inspects the joint angles to see if a gesture is occurring. Because joint angles are being used for gesture interpretation the user can perform a gesture facing any direction and even while moving. Additionally, the size of the user has no effect on the joint angles, and thereby no effect on gesture recognition. Currently the system is able to reliably interpret all of the gestures shown in Figure 6. *(Again a movie will be available on the web soon).*

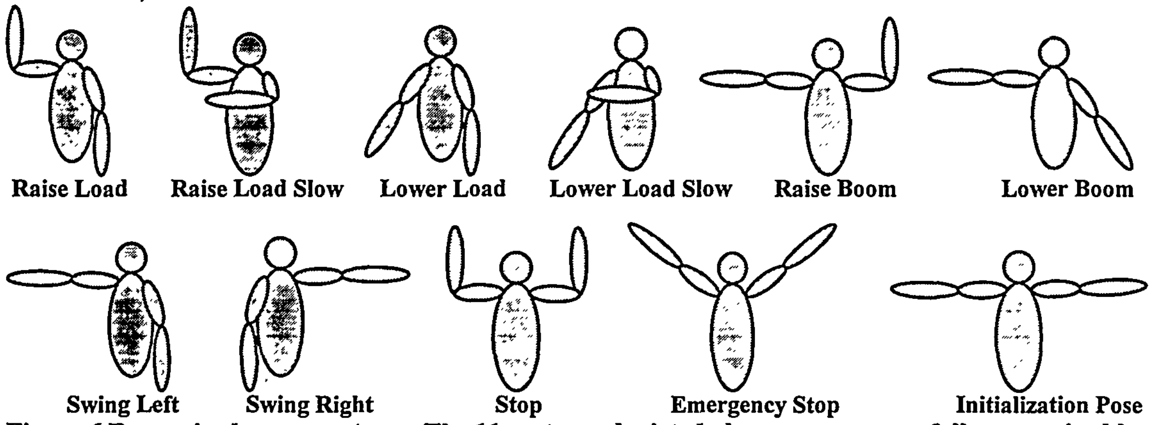

Figure 6 Recognized crane gestures. The 11 gestures depicted above were successfully recognized by our system using human joint angles. We also allowed the gestures on the top row to be performed with the arms switched.

7 Conclusions and Future Work

The system we have developed is able to collect high-resolution 3D voxel data at 25 Hz. We have also developed a calibration procedure that allows the system to be rearranged or even moved and to be quickly recalibrated. We are currently able to use the hardware and software that we have developed to track a low-dimensional human avatar figure at 16 Hz. The avatar we are using has a total of eleven degrees of freedom: four in each arm, and XY-theta for the position of the body. The system provides visual feedback in real-time with either human avatar display or through displaying the voxels overlaid with the joint positions. Lastly we were able to employ our system to perform crane gesture interpretation and successfully interacted with a system using a robotic arm to simulate crane movement.

By working directly in the 3D domain we eliminate problems caused by occlusion/self-occlusion, thereby allowing complicated motions to be accurately captured and tracked. By tracking a human stick figure model we are able to directly calculate human joint angles, which greatly simplifies motion analysis. In addition, because our system works at a high cycle rate, we are able to track extremely fast motions.

This project is currently undergoing many improvements. First we plan to implement a tracking algorithm that will fit an exact torso model to our data, which incorporates x, y, and z translation and accounts for not only rotation about its main axis but for tilting to the side and forward. Within this new torso model it will be easy to obtain both shoulder and hip locations. Hence our current tracking algorithm can still be used for tracking the arms and can also be implemented to track the legs. We are currently in the testing phase for this algorithm. Once this is complete we plan to further extend our algorithm to track the head and possibly even some hand orientations. We also plan to test the accuracy of our system by obtaining a ground truth to compare to the joint angles obtained through tracking. This will be accomplished by using a traditional electromagnetic tracking system (Ascension's Flock-of-Birds). The markers can be used to calculate a ground truth while the results of our tracking are written to a file for comparison. In addition, we plan to implement our system in several applications. We believe the system will allow us to interact in real-time with another human avatar driven at a remote location. This will enable a "3D conference-call" between users. Lastly would like to extend our algorithms to allow for multiple people to interact within the workspace of a single sensor. Currently, we are limited to one user for one sensor.

8 Bibliography

[1] C. Bregler and J. Malik, "Tracking People with Twists and Exponential Maps," *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition,* 1998.
[2] D. M. Gravilla, "The Visual Analysis of Human Movement: A Survey," *Computer Vision and Image Understanding*, vol. 73, no. 1, pp. 82-98, 1999.
[3] C. Wren, A. Azarbayejani, T. Darrel, and A. Pentland, "Pfinder: Real-Time Tracking of the Human Body," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 19, no. 7, pp. 780-5, July 1997.
[4] Q. Chai, and J.K. Aggarwal, "Tracking Human Motion in Structured Environments Using a Distributed-Camera System," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 2, no. 11, pp. 1241-7, Nov. 1999.
[5] I.A. Kakadiaris and D. Metaxas, "3D Human Body Model Acquistion from Multiple Views," *Proceedings of the Fifth International Conference on Computer Vision,* pp. 618-23, Boston, MA, June 1995.
[6] D. Gravrila and L. Davis, "3-D Model-Based Tracking of Humans in Action: A Multi-view Approach," *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 73-80, San Francisco, 1996.
[7] K.M. Cheung, T. Kanade, J.Y. Bouguet, M. Holler, "A Real Time System for Robust 3D Voxel Reconstruction of Human Motions", *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition,* 2000.
[8] Open Source Computer Vision Library,
http://www.intel.com/research/mrl/research/cvlib/overview.html.

Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company, for the United States Department of Energy under contract DE-AC04-94AL85000.