Development and Analysis of a Real-Time Human Motion Tracking System

Jason P. Luck[1,2] Christian Debrunner[1] William Hoff[1] Qiang He[1] Daniel E. Small[2]

[1]*Colorado School of Mines*
Engineering Division
{jluck, cdebrunn, whoff, qhe}@mines.edu

[2]*Sandia National Labs*
Intelligent Systems and Robotics Center
desmall@sandia.gov

Abstract

This paper describes a method for tracking human body motion from multiple views in real-time. The method extracts silhouettes in each view using background subtraction, and then intersects the visual hulls generated by the silhouettes to create a set of voxels. The voxels then exert attractive forces on a kinematic model of the human body to align the model with the voxels. The linked nature of the model allows tracking of partially occluded limbs. The size parameters of the kinematic model are determined automatically during an initialization phase. The kinematic model also incorporates velocity, joint angle, and self collision limits. The entire system with four cameras runs on a single PC in real-time at 20 frames per second. Experiments are presented comparing the performance of the system on real and synthetic imagery to ground truth data.

1 Introduction

Due to the enormous number of applications involving human-computer interaction, real-time markerless 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 medicine, advanced user interfaces, 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 [1-5,8-10]. However in his review of work done in human tracking, Gavrila states that results of markerless vision based 3D tracking are still limited. In conclusion he lists several challenges that must be resolved before visual tracking systems can be widely deployed [1]:

1) *Model acquisition*. The majority of previous work assumes the 3D model is known *a priori*.
2) *Occlusion*. Most systems cannot handle significant occlusion and do not have mechanisms to stop and restart tracking of individual body parts.
3) *Modeling*. Few body models have incorporated articulation constraints or collision constraints.
4) *Ground truth*. No systems have compared their results to ground truth.
5) *3D data*. Few systems have used 3D data as direct input. Using 3D data relieves the problems associated with retrieving 3D information from a 2D-view [1].

In addition to Gavrila's challenges we believe that there are two other requirements for a tracking system to be readily deployed:

6) *Real time*. A system must also perform tracking in real-time to be useful for most applications.
7) *Calibration*. Calibration of the data acquisition device must be simple and fast.

The method proposed in this paper not only expands upon the previous work but will also attempt to meet these challenges. We propose a method whereby 3D data is collected with a real-time shape from silhouette sensor [4] developed at Sandia National Laboratories that is similar to that of Cheung and Kanade [2] and Borovikov and Davis [3]. A physics based method is then used to align our model with the data in real-time.

2 The RTS^3 Sensor

The system we are using to acquire 3D data is a **R**eal-**T**ime **S**hape-from-**S**ilhouette **S**ensor that we call RTS^3 [4]. This sensor uses a combination of industry standard components including a high-end PC, four analog color video cameras, and four PCI-bus color frame grabber cards. Using this hardware we create a time-varying volumetric estimate of the visual hull of whatever object is moving in the space observed by all four cameras.

The algorithm for performing the shape from silhouette involves extracting silhouettes from the four images using an adaptive background subtraction and thresholding technique, described in detail in [9]. This algorithm indicates which pixels have changed from the background in each of the cameras. The calibration procedure described in [4] is used to determine the internal (including radial distortion) and external parameters. These parameters are then used generate look up tables, which relate every voxel to a pixel in each of the cameras. By traversing the voxels and examining the appropriate image-pixels, one can tell which voxels are occupied. Voxels are considered occupied if the appropriate pixels in each of the cameras have changed from the original background, as depicted in Figure 1. This results in a very fast, low-latency 3d modeling system that is suitable for tracking work.

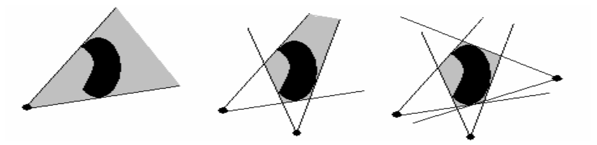


Figure 1 Visual hull. This shows the progression of a 2D visual hull as more cameras are added to the system. The visual hull includes not only the object (shown in black), but also other regions (shown in light gray) such as the concavities and tails.

Creating a background subtraction technique that performs robustly in different situations is difficult because the problems due to noisy cameras and shadows complicate the separation of silhouette pixels from background pixels. In monochrome systems, intensity differencing is used to distinguish silhouette from background pixels. However, ambient noise levels vary widely between cameras making it impossible to use the same threshold values across a set of cameras. Our approach was to account for differences in both the camera and across the image by taking a standard deviation of a series of images and creating a per-pixel threshold for each camera. To avoid including shadows in the silhouette hue differencing was also added, as a shadow will only produce slight changes in hue. One last problem of filling of small holes in the silhouettes was also addressed. Typically morphological operators are used for this purpose; however these operators introduce errors to the silhouette. Instead a routine that searches the image for contours and then fills small interior regions and removes small exterior regions was added. Accordingly, the routine not only fills holes in the silhouette but also removes speckle from the background. All of these routines were then combined into an efficient algorithm, (see [4] for details), that is able to run on a single computer using four cameras at near frame-rate.

The advantage of this system over previous systems is the speed of the algorithm. With the use of look up tables and efficient background subtraction routines, we are able to run in real time on a single computer, unlike other systems that require multiple computers to achieve real time performance [2] [3]. In addition, the system does not require the user to wear any special clothing, nor does it require a specialized background. However, if clothing has a similar color to the background, tracking can be affected (this is discussed in Section 4).

3 Tracking the Humanoid Model

The ultimate goal of human tracking is be able to autonomously track in real-time a reduced kinematic model of a human undergoing unconstrained movement in the workspace. We use the RTS^3 system to acquire the volumetric data in real time. The next component is a method to acquire and track the human model. The model is a series of linked segments, which articulate at joints between the segments, as depicted in Figure 2. The model contains four degrees of freedom (DOF) in each arm and leg (rotations only), three DOF for the head (rotations), and six DOF in the back (3 translations and 3 rotations). The segments model the body as closely as possible and do not change shape or size during tracking, which could cause errors in subsequent tracking.

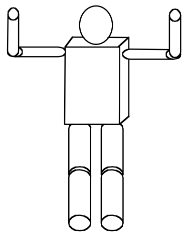


Figure 2 Humanoid model. The reduced body model consists of a six degree of freedom torso ($X,Y,Z,\alpha,\beta,\omega$) with four degree of freedom articulated limbs (arms, legs) and a three degree of freedom head.

3.1 Initialization

Currently, to acquire the model the user must perform an initialization pose upon entering the workspace. The pose is a simple stance in which body segments are in clear view (standing erect and facing an approximate direction with arms extended to the sides and legs separated from one another), so that it is easy to measure parameters of each body segment. Once in this pose the system segments out voxels associated with each body part using a growing algorithm that intelligently grows out from the center to each body part [5]. Body parameters are then estimated for each segment by fitting an ellipsoid to all the voxels associated with that segment (explained in [2] and [5]).

3.2 Tracking

Once the model has been acquired, tracking of the model can begin. The tracking scheme is a physics-based approach in which the data points (voxels) exert forces on the model as described in detail in [9]. With our model one only needs to solve for the joint angles of succeeding segments, because they are anchored by the previous segment. This principle has been widely used for controlling the movement of robotic arms [6]. Along these lines each voxel exerts a force upon the model, which act like springs to pull the model into alignment with the data as shown in Figure 3. Accordingly the force exerted increases with the voxel's distance from the model, until greater than a maximum distance at which voxels are assumed to be erroneous and their pull is set to zero.

To calculate the pull each voxel exerts on the model, the routine must project each voxel onto the model. This process is carried out by computing the nearest point on each body part to the voxel. The minimum of these distances is taken to be the proper association between the voxel and the model. This assumption is valid as long as the adjustment between the voxels and the model is small, which is the case for us since we are operating at approximately 20 Hz. (a person does not move far in 1/20 of a second). Hence each voxel will pull on the model as a spring would at the point on the model that it is closest to, thereby pulling that point on the model towards the voxel.

This spring-like method is used in an iterative scheme in which several small adjustments to the model's position and orientation are made for each set of data. Since data is acquired at extremely fast rates (~20 frames per second) the adjustment to the model will be small for each data set. The adjustments are calculated using the forces to calculate accelerations, which are then transformed into adjustments via the principle that *distance = $at^2/2$* (time is arbitrary for this system since the forces and hence the acceleration are virtual).

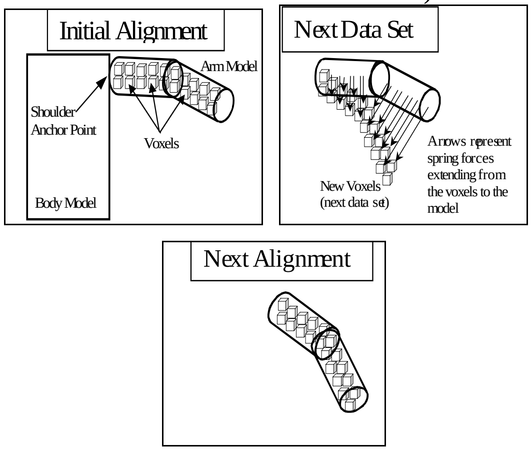

Initial Alignment
Arm Model
Shoulder Anchor Point
Voxels
Body Model
Next Data Set
Arrows represent spring forces extending from the voxels to the model
New Voxels (next data set)
Next Alignment

Figure 3 Model alignment. Voxels exert spring-like forces, which pull the model into alignment with the data.

3.2.1 Torso Tracking

The torso model used has six degrees of freedom, translation in three directions and three rotations. Therefore, forces exerted on the torso must create both translational and rotational adjustments. Translation is calculated according to $\vec{a} = \vec{f}/m$, where $\vec{f}$ is a force vector of the sum of all the forces exerted on the torso, and m is the mass of the torso (mass is set to the number of voxels that project onto the torso). The rotational adjustment is calculated according to $\vec{\alpha} = I^{-1}\vec{\tau}$, where $\vec{\tau}$ is a vector of the sum of the torques created by each force about the centroid, and I is a diagonal matrix of the inertia of the torso model. These equations yield accelerations that are used to calculate adjustments for the iterative alignment using $\vec{d} = \vec{a}t^2/2$ and $\vec{\theta} = \vec{\alpha}t^2/2$.

3.2.2 Limb Tracking

The legs and arms are anchored at their respective hip and shoulder locations on the torso, which are assumed to remain constant for a particular torso orientation. Therefore, the forces acting on the limbs translate to pure rotations in the form of torques. For a system of connected segments the Jacobian can be used to simplify torque calculations, $\vec{\tau} = J^T \cdot \vec{f}$ [6]. Newton-Euler dynamics relates joint torques to the velocities and accelerations of a system, $\vec{\tau} = M(\theta)\vec{\theta}'' + \vec{v}(\theta,\theta') + \vec{g}(\theta)$, where M is an inertia matrix of the segments, $\vec{v}$ is a vector of the Coriolis and centrifugal terms, and $\vec{g}$ is a vector of gravity terms [6]. In this virtual environment the effects of Coriolis, centrifugal, and gravity forces can be thrown out because they do not exist. Accordingly the equations now relate the joint torques to the angular acceleration at each of the joints through the inertia matrix, $\vec{\alpha} = M^{-1}\vec{\tau}$. Again using the simple equation, $\vec{\theta} = \vec{\alpha}t^2/2$, one can compute an adjustment to the joint angles due to each force.

Since many voxels exert pulls on each segment, it is far more efficient to combine all of the forces for each body segment into a single force before applying the matrix equations. In order to preserve the torques at not only the current joint but also at preceding joints these two equations must be observed: $\sum \vec{f}_i l_l = \vec{F}L$ and $\sum \vec{f}_i (l_i + l_o \cos(\theta)) = \vec{F}(L + l_0 \cos(\theta))$. It can easily be shown that if the resultant force vector is equal to $\vec{F} = \sum \vec{f}_i$, and if it acts on the axis of the arm at a distance equal to $\vec{L} = \sum \vec{f}_i l_i / \sum \vec{f}_i$ from the joint, the torques at each preceding joint will be equivalent.

Additional invariance to missing data/occlusions is built into our algorithm by dynamically adjusting the model parameters to comply with the data. Since our data is extremely noisy and data is often missing or occluded from view, the tracking algorithm must work with more or fewer data points. Hence as the number of points projecting on the model varies, the mass of each segment is dynamically set to the number of voxels currently projecting onto the segment. In this way if only 1 voxel projects onto the segment the pull will be appropriate to align the segment with the data, and later if hundreds of voxels project onto the same segment the mass of the segment will be proportional to the forces applied to it. To account for occlusion or missing data in a particular region of a segment, the center of mass of each segment is positioned at the center of all of the projected voxels. Accordingly if points only project onto the tip of the segment the center of mass of the segment will be in the center of these pulls and the resulting torques on the segment and any proceeding segments will be appropriate to align the model with the data.

Robustness to occlusion is also inherent to our algorithm because information from all four cameras is used to generate our data; hence self-occlusion will rarely happen in all four-camera views. Also, because the entire model is adjusted if data is completely missing from a segment, the preceding and following segments will still align with their data and pull the segment into the correct orientation.

3.2.3 Head Tracking

The head model is again anchored at the neck, which is positioned according to the torso orientation. Once more the rotations can easily be calculated using normal dynamics: $\vec{\alpha} = I^{-1}\vec{\tau}$, where torques are calculated about the base of the head. As before adjustments are calculated according to $\vec{\theta} = \vec{\alpha}t^2/2$.

4 Results

The current system is able to collect data, track our model, and provide visual feedback to the user at 20 frames per second on a single computer at a volumetric resolution of two-inch voxels. Increasing the resolution to one-inch voxels slows the system to about 9 Hz.

We have also set up an Optotrak system [7] to record ground truth pose information while our tracking algorithm is running. The Optotrak system records the position of markers placed upon our user. From these markers joint angles are calculated and compared to the joint angles obtained with our tracking algorithm. In this way we are able to compare our tracking results to accurate ground truth data. The accuracy of our ground truth is not exact because the markers are attached to the skin; however it is accurate enough to assess the accuracy of our system.

Given the Optotrak ground truth data, we were able to construct synthetic 3D models of the ground truth and render them using Open Inventor graphics software in views nearly identical to those of the real data[1]. We then used these synthetic images as input to our method and compared its performance on the synthetic imagery to the performance on real imagery. These synthetic images are rendered with flat shading and a uniform background so that we can extract perfect silhouettes (within the pixel resolution limits of the images). Figure 4 shows an example of one of the real images, the corresponding simulated image, and the silhouettes extracted from each. Figure 5 shows an example of the rendered model superimposed with the voxels extracted from the synthetic imagery of that model.

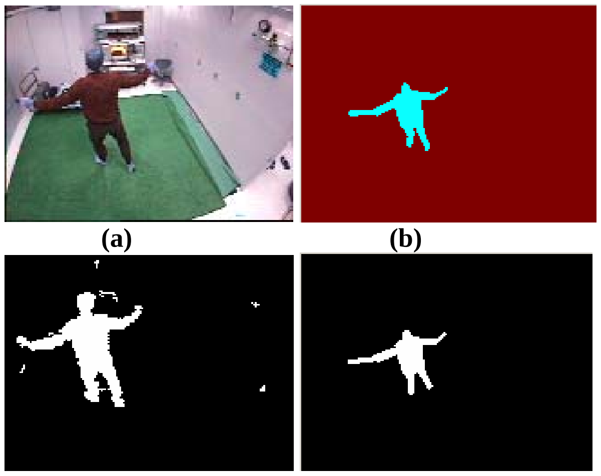


Figure 4 Typical real (a) and synthetic (b) images used in the ground truth experiments. (c) and (d) show the silhouettes extracted from the images of (a) and (b) respectively.

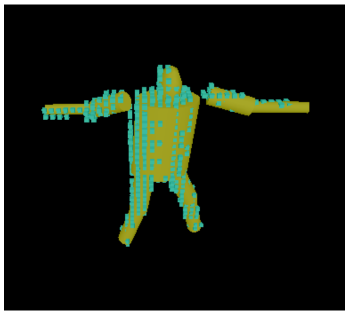
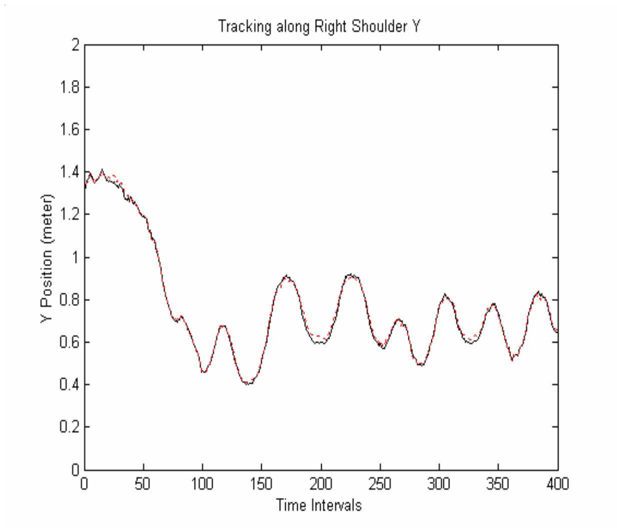


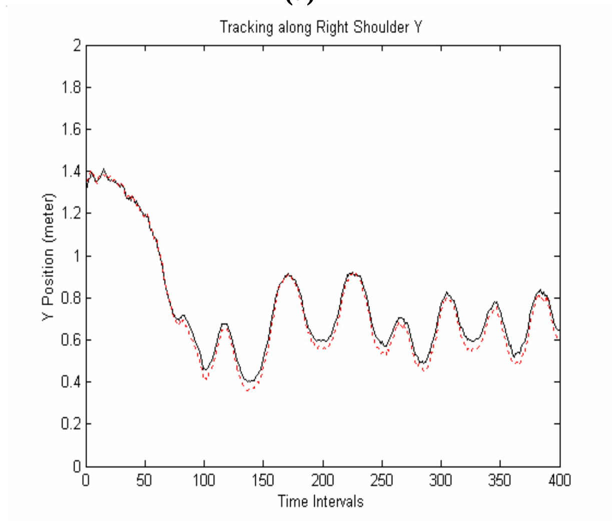
Figure 5 Example of a 3D model created from the ground truth data superimposed with the voxels extracted from the synthetic images of the model.

Our analysis of the results shows that tracking in synthetic imagery is substantially more accurate that tracking in real imagery. We measured the error in tracking results in terms of the difference between the ground truth and the tracked joint positions. Figure 6 shows one example (the y-coordinate of the right shoulder position) typical of the synthetic and real imagery tracking results.

[1] Since OpenInventor camera models do not include radial lens distortion, we were not able to model this aspect of the real cameras.



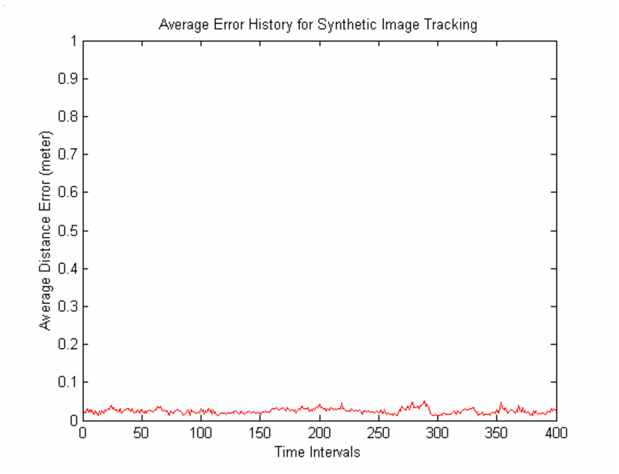
(a)



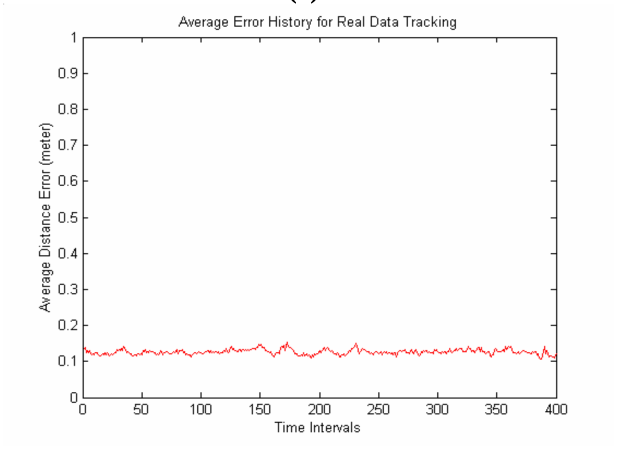
(b)

Figure 6 An example of point position errors in tracking in synthetic imagery (a) and in real imagery (b). The solid line is the ground truth, and the dotted line is the tracking result.

Figure 7 shows the average joint position error (averaged over all joints) at each time instant for both synthetic and real data. Note that the 2 cm error of the synthetic tracking result is about 40% of the two-inch voxel size used to represent the volumetric data. Future experiments will determine whether this relationship is consistent across a range of voxel sizes and image resolutions.



(a)



(b)

Figure 7 Average joint tracking position error (averaged over all joints) at each time instant for the synthetic imagery (a) and for the real imagery.

Table 1 shows the errors for each joint averaged over all times for both the synthetic image data and the real data. Note that every joint is tracked more accurately in the synthetic imagery and in the real imagery.

The ground truth sequence was recorded to disk so that it could be processed at different voxel sizes. The sequence was processed using both two-inch and one-inch voxel sizes and the results show no correlation between voxel size and accuracy, as can be seen in Figure 8. From this we conclude that at our current image resolution, the overall accuracy of our system is not improved by processing at a higher voxel resolution (which would greatly slow the system). The same conclusion was reached by Laurentini and Bottino [8]. In their work shape-from-silhouette data is used in a more precise and slower tracking approach, and then synthetic images are used to assess the accuracy of the system.

Table 1 Average tracking error (averaged over time) of each joint for synthetic and real data.

Joint	Average Error (mean, stdev) (m)	
	Synthetic	Real
Right Shoulder	(0.025, 0.010)	(0.070, 0.015)
Left Shoulder	(0.034, 0.013)	(0.070, 0.012)
Right Elbow	(0.023, 0.013)	(0.146, 0.014)
Left Elbow	(0.020, 0.011)	(0.132, 0.015)
Right Hand	(0.026, 0.015)	(0.176, 0.033)
Left Hand	(0.020, 0.012)	(0.160, 0.034)
Right Hip	(0.032, 0.012)	(0.055, 0.014)
Left Hip	(0.026, 0.012)	(0.055, 0.016)
Right Knee	(0.024, 0.011)	(0.127, 0.020)
Left Knee	(0.021, 0.010)	(0.129, 0.017)
Right Foot	(0.023, 0.012)	(0.197, 0.030)
Left Foot	(0.019, 0.009)	(0.201, 0.022)

A movie of our ground truth sequence and several other tracking sequences are available at http://egweb.mines.edu/cardi/3dvmd/3dvmd.htm. The movies show the system both acquiring and then aligning the model with the data in real-time while the user undergoes complex motions.

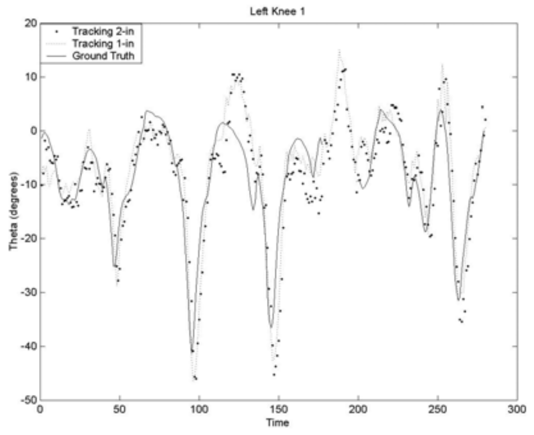


Figure 8 Comparison to ground truth. The figure shows the comparison of our tracking results to ground truth taken with an Optotrak sensor. The results for both two-inch and one-inch voxel resolutions are shown.

Tracking errors can arise from several problems associated with the data. Problems occur when full length of the arm touches the body because visual hull is not exact enough to distinguish the arm voxels from the body voxels, i.e. the voxels of the arm and torso are joined. However the routine will recover once the arm moves away from the body and the problem will not occur when just the hand comes in contact with the body because there is still enough separation to distinguish arm and body voxels. Tracking can also fail when too much data is missing along a limb. In the case shown in Figure 9 (a) and (b) the arm was next to the body, and as the subject moved his arm away from the body too few voxels were detected due to the similarity in hue of the subject's clothes and the background. Hence when the subject's arm was moved up there were not enough voxels pulling the arm of the kinematic model up to overcome the force of the erroneous voxels around the body which held the arm in place. Errors can occur due to tailing from the cameras (see Figure 1). In the example shown in Figure 9 (c) and (d) the subject is squatting in a position in which cameras cannot make a clean cut along the right side of the body, hence tailing occurs and many erroneous voxels appear along the right front of the body. These points twist the body and hence affect the arm and leg orientations. In both of the cases shown in Figure 9 the algorithm recovered in a few iterations and tracking continued.

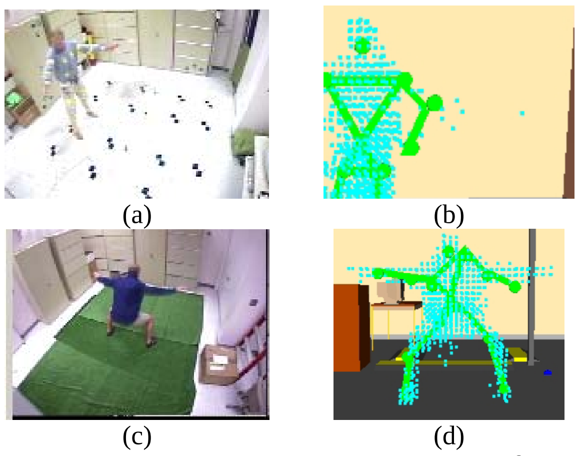


Figure 9 Figures (a) and (b) show the tracking after the left arm has just been raised after being adjacent to the torso. The model does not properly pull away from the torso. Fig (c) and (d) show the tracking errors produced by excessive tailing on the right front of the body.

5 Conclusions and Future Work

We have built a shape-from-silhouette data acquisition system, which works in real-time on a single-processor commercially available computer and can robustly acquire data from complicated backgrounds while dealing with both camera noise and shadows. The combination of a single computer and relatively simple calibration makes this system applicable for implementation in a large variety of applications.

The tracking system created for this sensor addresses all of the limitations of previous systems as listed by Gavrila [1] and the additional limitations presented by the authors:

1) The model is automatically acquired when the user enters the workspace.

2) The method is able to deal with occlusion because the forces exerted by voxel data transfer through the system. For instance if no voxels project onto the upper arm, points on the lower arm will still pull the upper arm into the correct pose.
3) Velocity, self-collision, and joint limits constraints have all been incorporated.
4) We have measured the accuracy of our tracking system against ground truth data.
5) Our RTS^3 sensor obtains 3D data, thus avoiding the problems of recovering 3D pose from 2D data.
6) We are able to work in real-time using only a single commercially available computer.
7) We have developed a method to quickly and easily calibrate the system.

The tracking algorithm was designed using sound dynamic and mathematical properties, and in doing so we have also made several valuable contributions to visual tracking. First of all, this will be the only human tracking paper that we know of which compares its results to actual ground truth data acquired during tracking. Secondly, we have managed to eliminate the known singularities in Jacobian based tracking of our humanoid model. In doing so, we have also created a general tracking framework under which tracking can be extended to any rigid articulated model and remove known singularities. The system can also acquire data, automatically acquire our model from a simple initialization pose, track the model, and provide visual feedback all in real-time on the same computer.

6 Acknowledgements

This research was supported by a grant from the Colorado Advanced Software Institute (CASI). CASI is sponsored in part by the Colorado Commission on Higher Education (CCHE), an agency of the State of Colorado. This work was also funded by Sandia National Laboratories. 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.

Special thanks go to the Intelligent Systems and Robotics Center, in particular: Dan Schmidt, Brian Rigdon, Jeff Carlson, and Carl Diegert.

7 Bibliography

1. Gavrila, D.M., *The Visual Analysis of Human Movement: A Survey*. Computer Vision and Image Understanding, Jan. 1999. 73(1): p. 82-98.
2. Cheung, G.K.M., *et al.* A Real Time System for Robust 3D Voxel Reconstruction of Human Motions. *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2000.
3. E. Borovikov and L.S. Davis. A Distributed System for Real-Time Volume Reconstruction. *Proceedings International Workshop on Computer Architectures for Machine Perception*, Padova, Italy, Sept. 2000.
4. D.E. Small, *Real-Time Shape-from-Silhouette*. Masters thesis, University of New Mexico, 2001.
5. J.P. Luck and D.E. Small. Real-Time Tracking of Articulated Human Models using a 3D Shape-from-Silhouette Method. *Proceedings Robot Vision 2001*, Auckland, New Zealand, 2001.
6. John J. Craig, “Introduction to Robotics Mechanics and Control.” 2nd Edition. Addison-Wesley Pub. Inc. Reading, MA, 1989.
7. Web Site. Northern Digital Optotrak 3D Tracking Sensor, http://www.ndigital.com/optotrak.html, accessed August 2001.
8. A. Bottino and A. Laurentini, *A Silhouette Based Technique for the Reconstruction of Human Movement*. Computer Vision and Image Understanding, 2001, 83: p. 79-95.
9. J. P. Luck and D. E. Small. Real-Time Markerless Motion Tracking using Linked Kinematic Chains. *Proceedings CVPRIP 2002*, Durham, NC, Mar. 2002.
10. Y. Song, X. Feng, P. Perona, Towards Detection of Human Motion, *Proceedings* CVPR00, Hilton Head Island, SC, June 2000.