Human Model Reaching, Grasping, Looking and Sitting using Smart Objects

D. Slonneger*†, M. Croop†, J. Cytryn†, J. T. Kider Jr.†, R. Rabbitz‡, E. Halpern‡, and N. I. Badler†

† *SIG Center for Computer Graphics, University of Pennsylvania*

‡ *Lockheed Martin Maritime Systems Corporation*

Abstract

Manually creating convincing animated human motion in a 3D ergonomic test environment is tedious and time consuming. However, procedural motion generators help animators efficiently produce complex and realistic motions. Using the concept of a Human Modeling Software Testbed (HMST), we created novel procedural methods for animating reaching, grasping, looking, and sitting using the environmental context of 'smart' objects that parametrically guide human model ergonomic motions. This approach enabled complicated procedures such as collision-free leg reach and contextual sitting motion generation. By procedurally adding small secondary details to the animation, such as head/eye vision constraints and prehensile grasps, the animated motions look more natural with minimal animator input. A 'smart' object in the scene graph provides specific parameters to produce proper motions and final positions. These parameters are applied to the desired figure procedurally to create any secondary motions, and further generalize to any environment. Our system allows users to proceed with any required ergonomic analyses with confidence in the visual validity of the automated motions.

Keywords: Posture and Motion, Human Performance, Motor Behavior.

1. Introduction

Operators of human factors software tools manipulate graphical user interfaces to define human tasks to be performed and analyzed in 3D modeled environments. Although the historic purposes of such analyses were human fit, safety, and comfort, motions were mostly limited to arm and leg reaches in a seated posture, lift or placement actions using arms or whole body motions, or locomotion from one workplace to another. Such movements precipitated great interest in algorithms for reach, visible space, strength, collision avoidance and path navigation.

Over the years the job of the human factors analyst has been to reproduce desired tasks in digital manikin modeling systems in order to analyze necessary feasibility, fit, accommodation, comfort and health factors. Reproducing human tasks has thus become rather more aligned with human motion animation as used in the movie and game industry, without giving the human factors engineer the benefit of the extensive tools (and concomitant background and training) developed for those well-funded industries.

Figure 1: This diagram shows an image from the Human Modeling Software Testbed (HMST). This is an ergonomic test platform tool based on the Jack and Process Simulate Human toolkit. The tool allows users to test designs of new environments with virtual people and test many human factors, such as timing, efficiency, reachability, lines of sight, and user comfort.

This paper describes the processes and methodologies we engaged to add new functionality to allow procedural support for secondary animations. This section outlines in detail the general approach to procedural motion generation across several body systems including view control, hand grasps, arm reach, leg reach, and a sit-down

generator based on a “smart” chair object. We present and discuss our results and a discussion of an informal assessment of these improvements’ utility for the user.

2. Materials and Methods

When humans reach for objects they typically do not stare lifelessly ahead but instead track what they do. A view motion procedure provides a tracking system that drives the figure’s head and eyes to follow the hand, the reach target, or the midpoint between these two. The eyes of the figure follow the chosen point until the eye limits are reached, at which time the head begins to track, allowing for greater tracking coverage. While the angles are constrained by the figure’s physical limits, the speed of motion is based upon Fitts’ Law. The eyes are allowed to move independently or in parallel, depending if visual vergence is desired, e.g. depending on target proximity.

Proper hand grasps also add realism to the animation. We incorporated motions for thirteen grasps (MacKenzie and Iberall 1994). To couple the grasp action with arm reach, we conducted a simple study to measure the time and distance away from various objects where people begin to change their hand shape for the final grasp. From these empirical tests we found that the transition began about 50 cm from the object.

While the HMST already had a collision-free arm reach (Zhao et al. 2005), leg reaches had to be animated by hand. We designed a collision-free leg reach allowing for more automatic animations, e.g., by allowing the feet to reach for pedals or move around chair legs. We also modified the arm/leg strength curves (used to create natural joint positions) by making each smooth and continuous. This creates smoother more pleasing animated motions by avoiding discontinuities. Using (Baerlocher 2001) the function used to calculate the swivel angle for the arms and legs has been modified so that only one orientation singularity may occur. This point is located behind the figure in an area that cannot be reached.

Finally, the sit generation procedure is a significant aid to the animator. This tool allows any sized figure to sit in a variety of chairs. Drawing from the idea of a “smart object” (Kallman and Thalmann 1998), a given chair object in the HMST scene can provide parameters to the figure so that a proper sitting motion and final resting position can be reached. The parameters are a sit point and a desired knee position; if the chair has arms then points for each are needed to animate the figure’s reach and its final rest position. These parameters are read from a file before animation and are used to drive the figure from standing, to reaching back for balance, and finally to the rest position in the chair.

2.1. View Constraints

To create more realistic looking reach animations, head and eye tracking have been added. Originally the figure would look straight ahead while reaching for a point, but by making the figure’s view track the target point during the reach, the animation looks more lifelike and natural.

Adding the view tracking was complicated by the fact that the order in which the three degrees of freedom of head rotation are applied is different than the logical way of specifying the desired pitch and yaw. Thus, to prevent looking at the target with the head rolled, we perform rotation matrix manipulations to get the rotations applied in the correct order. Head Tilting is prevented in the sense that we always keep the local left vector of the head (which determines tilt) perpendicular to the (locally) constant up vector of the base of the neck. Thus, if the user rotates the figure and then aims at a target, there will be no roll relative to the upper torso.

There are various options now worked into HMST for different ways of doing head-eye motion during a reach. The user can track the target, the hand(s) that are moving toward the target, or the midpoint between the two. Also, the user can have the eyes independently track the target or force the eyes to be parallel (useful for two-handed reaches).

The main issue with head motion at this juncture is how the system deals with natural angle limits of the joints. For example, conflicts arise if we tell the model to look past the model's naturally comfortable range of rotation. Right now, we clamp each Euler angle to the natural joint limits, but with this method the model will end up looking at some arbitrary point. A better solution would be to find the arc between the unclamped target and the original view direction, and then clamp to the last point on this arc that is still within limits.

Figure 2: This image shows automatic head and eye tracking to the target of the reach.

2.2. Hand Grasping Model

In order to create a larger variety of animations with the hands, more grasp configurations were needed. At first we considered using motion capture to decide what kinds of grasps people used for everyday objects, but a lot of work has already been done on the subject of hand grasps. We decided to create a database of general usage grasps from the types enumerated in *The Grasping Hand* (MacKenzie and Iberall 1994) as well as in (Feix et al. 2009). We integrated these new hand grasps into HMST to allow the figure more generalized hand shapes while reaching. These grasps may be used in any part of the HMST but when used with the collision free reach, allows the hand to animate as the reach is occurring.

To better understand how humans reach change their hand shape while reaching for objects, we conducted a study. This study measured when people tend to start opening their hands in order to reach their target hand shape. We calculated this based upon distance from target and type of object being reached for. We ran these tests with 20 men and women to get a broader range of test subjects. These tests show that at about 50.0 cm from the object people begin to transition from their original hand shape to the grasp needed to grip the object. Below is a chart of what we found based upon two different starting distances from the object.

Table 1: Averaged finding from group study on reaching for variously sized objects.

	Short Reach (~137 cm)	Long Reach (~275 cm.)
Start of grasping position / total distance (%)	63.6%	81.6%
Start of grasping time / total time (%)	63%	78.6%
Distance from target when starting grasp (cm.)	49.96	50.35

While figuring out when humans start to change grasps adds realism to reach animations we decided more could be done. In order to do this we looked at how reach worked with vision tracking via Fitts' law. By using Fitts' law we can determine the speed for the reach and by replacing distance to target with the angle between the current view and target view vectors, the figure's head-eye tracking motion speed as well. Fitts' law depends on the width of the object; in the future this size would be read in from the smart object that the figure is reaching for. Currently, we are using a value of 10.0 cm as the width of the target objects but would like to incorporate different sizes for future work.

Using what we learned from the studies above, we determined the timing for the hand grasp. The hands start to open up towards the final hand grasp at a certain distance away from the target (about 50 cm) and does 80% of the interpolation to final hand pose by the time it gets to within 5 cm of the target. The last 20% of the motion is animated within these final centimeters so that the hand "clenches" more quickly at the end of the reach. Fitts' law is used to determine the speeds of both reach and head-eye motion. Finally, since the user would presumably start looking at a target before reaching for it, when tracking the target with the eye-head system we put in a time offset so that the eye-head tracking begins before reach does.

2.3. Collision-free Leg Reach

Expanding upon the work done in Zhao et al. (Zhao et al. 2005), we added collision free reach to the figure's legs in the HMST. This was done to accommodate reaching for pedals, avoiding chair

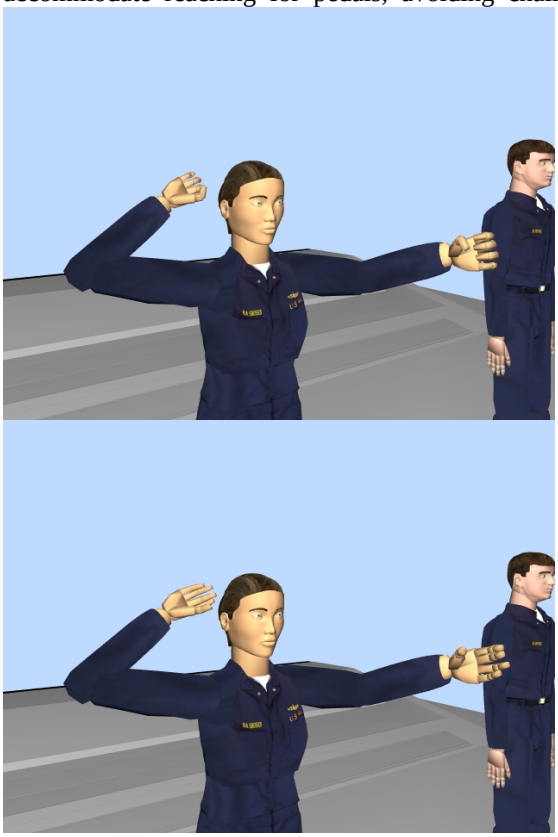

Figure 3: Two of the newly added grasps while reaching.

legs, and other animations that require leg adjustment within an enclosed work space. Modifying the arm reach for the leg was straight forward by replacing the shoulder, elbow, and wrist with the hip, knee, and ankle, respectively, and taking into account knee rotation. We also changed

some aspects of the general collision-free reach while incorporating the collision-free reach into the leg.

To start, we noticed that the strength curves from Zhao's reach (Zhao et al. 2005) were not always continuous or provide consistently smooth motions. We fixed these discontinuities by tweaking the control points and smoothed the curves by weighting low-twist angles to be more comfortable. An issue of having two singularities across possible swivel angles also existed and our solution is below.

2.4.1Discontinuity changes to IKAN

The HMST uses IKAN inverse kinematics for actions like walking and reaching (Tolani and Badler 1996). While performing leg reaches, the user will often want to move the end effector from a position in front of the model to a position behind the model. This required us to extend the functionality of reach so as to allow for a larger range of motion. IKAN models the leg with four degrees of freedom: three for the hip and one for the knee. The desired end effector position constrains three dimensions; however, for a given end effector position, there exists a circle of possible knee positions that are available for the IK solver. IKAN uses swivel angle as a fourth parameter to disambiguate among the knee positions. One point on this circle of positions is designated the zero swivel angle, and all other knee positions on the circle are determined by their angle away from this zero point. The function IKAN used to determine the reference elbow position from a given wrist position had two discontinuities. However, Zhao's (Zhao et. al 2005) reach depends on this function being continuous across adjacent voxel cells, as it uses nearby swivel angles to determine which positions are close to each other. The collision-free reach algorithm searches a graph of (position, swivel angle) pairs called pose cells, weighting edges between adjacent pose cells based on their spatial distance and swivel angle difference. Therefore, the function that maps end effector positions to zero swivel angle should be as stable as possible. If nearby positions have vastly different reference swivel angles, then similar poses may have large swivel angle differences, making this a poor metric near any discontinuities in the function. Although there is no completely continuous function mapping each vector to a vector orthogonal to the input, we changed the function so that it only has one discontinuity which is easily avoided, resulting in smoother animations during playback of the motions.

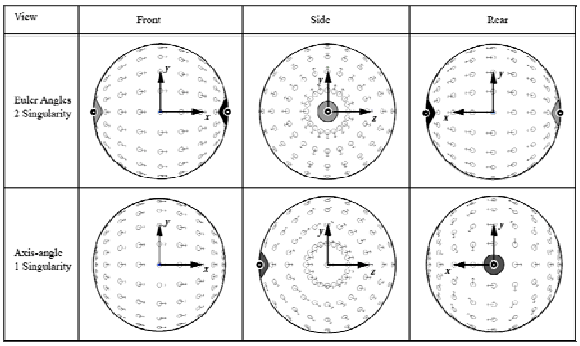

Figure 4: This diagram shows the direction of zero swivel angle for every possible unit end effector position for an Euler angle representation of rotation (equivalent to IKAN's previous method) and for an axis-angle representation of rotation (equivalent to our method). (Image source: Baerlocher 2001)

To determine the reference swivel angle, IKAN projects the end effector position onto a fixed plane, and returns the angle between the projected vector and a fixed reference vector in this plane. This function has two discontinuities, one at each unit normal vector of this plane, resulting in sporadic motions when the end effector approaches these discontinuities. We would like to choose the fixed plane so that typical reach motions do not move the end effector near these discontinuities; however, since the discontinuities must be in opposite directions, this is unfeasible. To rectify this, we instead find the minimum-angle rotation from a fixed vector to the end effector, and apply this rotation to a second fixed vector orthogonal to the first (see Figure 4). This results in a function with only one discontinuity, in the direction opposite the first fixed vector. The discontinuity can easily be positioned in a location that the end effector cannot reach, specifically the up vector for leg reach. Applying this change resulted in improved motion for leg reach. Similar modifications to arm reach were successful in increasing range of smooth motions.

Figure 5: This figure has just completed reaching for the target by using the collision-free leg reach.

2.4. Smart Objects

In order to create the sit generator we had to develop the smart object specification and design a set of classes to incorporate into the HMST. We referred to the works of (Kallman M, Thalmann D, 1998) and (Peters C et al. 2003) to design our smart objects. Below is the description for each class as well as an image of the class in UML (Unified Modeling Language).

Smart_Object:
The Smart_Object class holds the intrinsic properties of the object that will be interacted with, which are used for the physical simulation. The behavior variable is of type Object_Behavior and will control how a figure interacts with the object at a given moment in time.

Object_Behavior:
This Object_Behavior class is a finite state machine containing each possible interaction of the figure with the object. For example, a door could be open or closed and each of these options allows for different ways the figure could act. States is a container for each of the interaction possibilities, containing objects of type FSM_Entry. A state_controller is needed so the object can know the state variables, current state, next states that are possible, and when to change to a new state.

FSM_Entry:
The FSM_Entry class acts as an individual entry for the finite state machine contained in Object_Behavior. It holds the name of the state and possible animations for the object and figure. The animations would be dependent on what state this entry represents. For example, a door opening would animate the object but reaching for the knob or walking through would be an animation for the figure. Finally, there could also be a list of constraints to aid in the figure interaction of type Interaction_Information.

Interaction_Information:
The Interaction_Information class holds interaction values relating a figure's segment to the object. The agent_part is the specific segment of the figure's skeleton being constrained. The interaction_target is where on the object the figure will interact. Approach_vector describes how the figure's segment needs to be oriented during the interaction. If the segment needs to be in a specific configuration at the end of interaction an IK target is stored in the shape variable. Finally, this interaction constraint can be declared as required or not depending on the desired outcome.

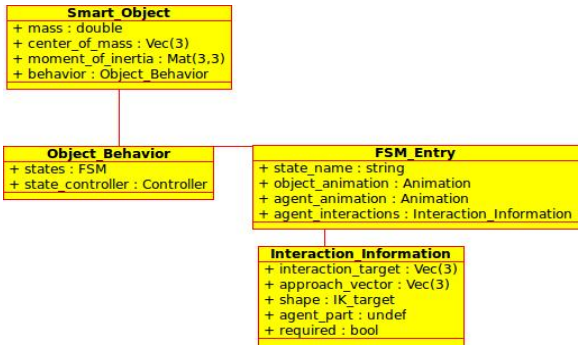

Figure 6: Image of the *Smart Object* class in UML format.

2.5. Sit-down Generator

In order to further help the engineers using the HMST, a way was desired to incorporate automatically allowing any sized figure to sit in any chair. Drawing from the idea of a smart object, an external file linked to a given chair object in the HMST scene provides parameters for the figure. These parameters describe how to animate a sitting motion and achieve a final rest position. The chair files contain two required sets of points and a combination of up to three other point sets. The required points tell the character where to sit and position its knees. The optional points allow the figure to reach for the chair's right and left arms for balance, rest its forearms on the chair's right and left arms, or position its wrists on a console in front of the chair. If no chair arm rest or console positions are given, the figure will place its hands on its lap midway between hip and knee. These points are given in the chair's coordinate system so that there is consistency in the coordinates and to ensure the figure is always facing forward no matter where in the HMST space the chair is.

Once these values are input and the sit animation begins, there are two separate components of the sit generation. First the character's sit point is driven

from standing to the chair's sit point by interpolating across the vector between the two. While this is happening, the figure is animated to make it appear to be balancing. An angle for its waist is calculated to cause the torso to lean forward, balancing itself from falling backwards. A knee height (position) is needed due to the fact that smaller figures cannot reach the floor while seated. If the knee height was not provided, the smaller figure's legs would unrealistically penetrate the chair. Using this height, the hip angles are changed each frame so that the knees positions are driven towards the desired final height. Finally, if arms exist on the chair, the character hands reach for the given points. This is accomplished by using the reach from Zhao's reach (Zhao et al. 2005) but without collision avoidance with the world. This was done because currently collision free reach works when only the figure is still; removing this constraint is work for the future.

After the figure is seated, the second part of the process begins. The torso, which is leaning forward, is driven to an upright position in the chair by changing the angle of the waist. The arms are moved depending on whether the chair has arms, no arms, or is at a console. If the chair has arms, the figure's left and right arms move to the rest position. The shoulder angle is driven so that the wrist can be moved into position within the Y-Z plane. At the same time the elbow angle is driven so that the wrist is in the proper position in the X-Z plane. When no chair arms are present the midpoint of the figure's lap is used as the rest position for each wrist. Finally, when a console exists, the figure reaches for the given console points in the same manner as when the figure reaches for the chair's arms.

Figure 7: This crew member is currently bracing herself on the chair arms as she settles into the chair.

3. Results

Using these new methods the HMST human factors engineers are able to quickly produce more realistic looking animations with less effort. As seen in Figure 8 adding view tracking to the reach quickly adds a lifelike appearance to the figure. The same is true of the new hand grasps. Figure 3 shows how the model's final hand configuration is animated during the reach.

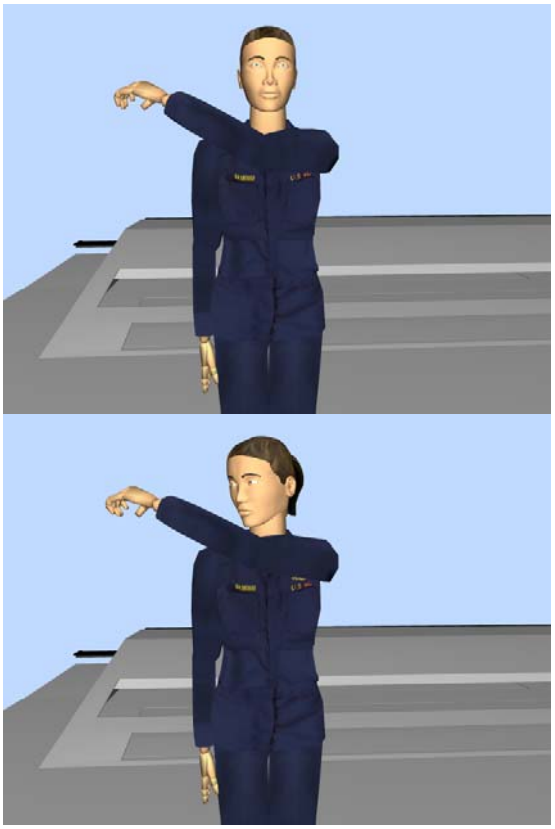

Figure 8: Reaching without and with view tracking can be seen. Notice how the bottom image looks more natural.

Using our collision-free leg reach algorithm, the figure comfortably reaches for a target end effector without unnecessary twist as in Figure 4. As with arm reach, the motion of the leg during reach no longer suffers from sudden changes in orientation, resulting in smoother, more natural-looking motion.

Our procedural sit generation method generates plausible sitting down animations for figures of arbitrary size on chairs with smart object data. Figure 7 shows a model sitting down using our technique. Note that the model uses the arms of the chair for balance. Figure 1 shows a group of crew members all seated using our sit generation algorithm.

4. Discussion

With the addition of view tracking, leg reach, grasp animations, and sit generation, crew member ergonomic evaluation can be expedited since we have decoupled the animation generation from tedious key framing methods. This allows the user to work on other aspects of the animation without needing to spend excessive time working on simple tasks and implied behaviors.

For example, objects can be quickly grasped without exhaustively key framing every joint. Even if the final grasp is not exactly what the animator desires, minimal key framing can be added once the reach is complete. While either vision tracking or final grasp could be done by key framing, our method does this automatically during the collision-free reach. Thus the user only needs to select the final hand configuration and where to reach before running the simulation.

As with the arm reach, the collision-free leg reach speeds up user workflow by animating the model in tight or hard to reach areas. This simulation allows the user to work on other parts of the animation instead of time-consuming key framing. Previously, users had to fix animations from the arm and leg reach due to irregularities caused by the two singularities in the function mapping elbow and knee positions to reference swivel angles. By limiting this to one singularity the simulation always creates realistic motions.

Finally, informal evaluation shows that it takes about an hour to animate a model sitting down in the HMST. Obviously this time grows linearly when 20 crew members must be seated; animating 100 is out of the question. The sit generator we implemented can do any number of crew members sitting in chairs. There is set up time for a specific chair type's smart object, but in these simulations it is more likely that the same chair type would be used repeatedly. With our system the user only needs to select the chairs and models they want to sit and then run the simulation. The simulation time does increase with the number of figures sitting but the animator can work on other tasks while this occurs.

5. Conclusion

We describe a suite of tools for procedurally simulating important secondary movement effects for human factors animations. Our system provides users with tools to perform sitting, collision free leg reaching, eye tracking and grasping animations. Our extensions save the HMST operator's time, and these effects provide added animated realism without having to manually key-frame the activities. These tools improve indoor environment design by speeding up placement, fit and motion tests using virtual people and objects.

Acknowledgements

The authors wish to acknowledge donations from NVIDIA for the graphics hardware, Siemens for the Jack Toolkit, and Autodesk for Maya.

References

Baerlocher P, 2001. Inverse kinematics techniques of the interactive posture control of articulated figures. Ph.D. Thesis, École Polytechnique Fédérale de Lausanne.

Feix T, Pawlik R, Schmiedmayer H.B., Romero J, Kragic D, 2009. The generation of a comprehensive grasp taxonomy. Technische Universität Wien. Technical report. <http://web.student.tuwien.ac.at/~e0227312/>

Kallman M, Thalmann D, 1998. Modeling Objects for Interaction Tasks. In: Proceedings of the 9th Eurographics Workshop on Animation and Simulation, Lisbon, Portugal.

MacKenzie C, Iberall T, 1994. The Grasping Hand. In *Advances in Psychology* 104, Stelmach G, Vroon P, eds., North Holland, Amsterdam,15-30.

Peters C, Dobbyn S, MacNamee B, O'Sullivan C, 2003. Smart Objects for Attentive Agents. Proceedings of the International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision, Plzen, Czech Republic.

Tolani D and Badler N, 1996. Real time human arm inverse kinematics. Presence 5(4), 393-401.

Zhao L, Liu Y, Badler N, 2005. Applying empirical data on upper torso movement to real-time collision-free reach tasks. In: Proceedings of SAE International Digital Human Modeling for Design and Engineering; paper 2005-01-2685, *SAE Transactions Journal of Passenger Cars - Mechanical Systems*.