September 1990

Human Factors Simulation Research at the University of Pennsylvania

Norman I. Badler
University of Pennsylvania, badler@seas.upenn.edu

Recommended Citation

Norman I. Badler, "Human Factors Simulation Research at the University of Pennsylvania", . September 1990.

University of Pennsylvania Department of Computer and Information Science Technical Report No. MS-CIS-90-67.

Human Factors Simulation Research at the University of Pennsylvania

Abstract

Jack is a Silicon Graphics Iris 4D workstation-based system for the definition, manipulation, animation, and human factors performance analysis of simulated human figures. Built on a powerful representation for articulated figures, *Jack* offers the interactive user a simple, intuitive, and yet extremely capable interface into any 3-D articulated world. *Jack* incorporates sophisticated systems for anthropometric human figure generation, multiple limb positioning under constraints, view assessment, and strength model-based performance simulation of human figures. Geometric workplace models may be easily imported into *Jack*. Various body geometries may be used, from simple polyhedral volumes to contour-scanned real figures. High quality graphics of environments and clothed figures are easily obtained. Descriptions of some work in progress are also included.

Comments

University of Pennsylvania Department of Computer and Information Science Technical Report No. MS-CIS-90-67.

Human Factors Simulation Research
At The University of Pennsylvania

MS-CIS-90-67
GRAPHICS LAB 34

Norman I. Badler

Department of Computer and Information Science
School of Engineering and Applied Science
University of Pennsylvania
Philadelphia, PA 19104-6389

September 1990

Human Factors Simulation Research
At the University of Pennsylvania

Norman I. Badler

Computer and Information Science Department
University of Pennsylvania
Philadelphia, PA 19104-6389

Abstract

Jack is a Silicon Graphics Iris 4D workstation-based system for the definition, manipulation, animation, and human factors performance analysis of simulated human figures. Built on a powerful representation for articulated figures, *Jack* offers the interactive user a simple, intuitive, and yet extremely capable interface into any 3-D articulated world. *Jack* incorporates sophisticated systems for anthropometric human figure generation, multiple limb positioning under constraints, view assessment, and strength model-based performance simulation of human figures. Geometric workplace models may be easily imported into *Jack.* Various body geometries may be used, from simple polyhedral volumes to contour-scanned real figures. High quality graphics of environments and clothed figures are easily obtained. Descriptions of some work in progress are also included.

1. Introduction

The Computer Graphics Research Lab at the University of Pennsylvania has been involved in the research, design, and implementation of computer graphics human figure manipulation software since the late 1970's. The history of this effort is too lengthy to detail here; rather, we wish to describe the current state of our system, called *Jack*, as of mid-1990.

The *Jack* sofware is built on Silicon Graphics Iris 4D workstations because those systems have the 3-D graphics features that greatly aid the process of interacting with highly articulated figures such as the human body. Of course, graphics capabilities themselves do not make a usable system. Our research has therefore focused on software to make the manipulation of a simulated human figure possible and even easy for a rather specific user population: human factors design engineers or ergonomics analysts involved in assessing human motor performance, fit, reach, view, and other physical tasks in a workplace environment. The software also happens to be quite usable by others, including graduate students and animators. The point, however, is that program design has tried to take into account a wide variety of physical problem-oriented tasks, rather than just offer a computer graphics and animation tool for the already computer-sophisticated or skilled animator.

This orientation toward *tasks* gives *Jack* its particular flavor. As we are Computer Scientists, we seek computationally general yet efficient solutions to problems. Human factors engineers often analyze a succession of specific tasks or situations. The role we play is transforming the specific needs of the engineer or analyst into the general case so that

- at least some large percentage of situations may be successfully analyzed;
- there is sufficient research required to justify doing the software in the Computer Science environment; and
- conversely, the general problems are difficult enough to expect that a specific problem-oriented approach will be economically or technologically infeasible for a particular human factors engineer.

As we continue to interact with human factors specialists, and particularly our research sponsors, we have come to appreciate the broad range of problems they must address. The challenge to embed a reasonable set of capabilities in an integrated system has provided dramatic incentives to study issues and solutions in 3-D interaction methodologies, multiple goal positioning, visual field assessment, and strength guided motion, to name a few. Our Lab effort has involved full-time staff and dozens of students over the past few years. Many of them are mentioned in this paper. Their efforts will be noted here because such a large project must clearly involve a large number of contributors[1]. They have worked cooperatively and collaboratively on the common *Jack* framework. It must be noted, however, that the principal architect of the *Jack* system is Cary Phillips. To a great extent the "look and feel" of *Jack* (as well as the name) is due to him.

The remainder of this paper discusses the major *Jack* features, organized around the topics of body and other geometric object structures, anthropometry, user interface, positioning, animation, analyses, rendering, and external interfaces. The final section briefly outlines some of the relevant work in progress in our Lab.

2. Summary of *Jack* Features

Jack is a Silicon Graphics Iris 4D Workstation-based system for the definition, manipulation, animation, and human factors performance analysis of simulated human figures. Built on a powerful representation for articulated figures composed of joints and segments with boundary geometry, *Jack* offers the interactive user a simple, intuitive, and yet extremely powerful interface into any 3-D articulated world using only the three-button mouse, keyboard, and pop-up menus. All *Jack* software has been written in *C* at the University of Pennsylvania; it does not depend on any third-party software (or hardware) outside the usual Silicon Graphics Iris utilities. In this section we discuss the major features currently available in *Jack*.

2.1. Body and other geometric object structure

Bodies as well as all other geometric objects, called *figures*, are represented externally to *Jack* in a language (*Peabody*) which describes their attributes and topological connections [PHIL88]. Figures consist of segments connected by joints, each with various degrees of freedom and joint limits. Important points on each segment are termed *sites* and are used, for example, to describe the attachment locations of joints or the positions of notable landmarks. Geometric transformations called *constraints* are used to position figures in the world coordinate reference frame.

The surface geometry associated with a segment has its own local coordinate system and is typically described as a network of polygons called *psurfs*. Geometry and topology editing facilities written by Osman Niazi are supplied in *Jack* though it is not intended to be or substitute for a "real" Computer-Aided Design system. *Jack* is very comfortable obtaining its geometric data from other systems (Section 2.9).

2.1.1. Default body model

The default human figure in *Jack* consists of 34 segments and 52 degrees of freedom. The segments are:

TORSO PART(8): body root, lower torso, lumbar1, lumbar2, lumbar3, thorax1, thorax2, upper torso.

ARM PART(6 * 2): sternum, clavicle, upper arm, lower arm, hand, finger mass

LEG PART(5 * 2): hip, upper leg, lower leg, foot, toe mass

[1] And, of course, sponsors. Please see the list in the Acknowledgments.

HEAD PART(4): neck, bottom head, eyeball(2)

There are 8 * 2 degrees of freedom from upper torso to fingers, 14 degrees of freedom from lower torso to upper torso, 10 * 2 degrees of freedom from hips to toes, and 2 degrees of freedom for the neck.

Normally the hands and feet are only minimally articulated (but see Section 2.1.6). There are five segments in the torso, yielding reasonable flexibility and appearance without sacrificing shaded drawing speed and interactive response. A full spine and torso model is available (Section 2.1.8).

2.1.2. User-specifiable topological structures

The body structure is not built into *Jack*; rather, the default body is there for user convenience. Any topological structure can be defined through *Peabody* and manipulated in *Jack*. In particular, this allows the use of figure models with greater or lesser articulation, as well as mechanisms, robots, insects, and so on. We frequently use a simpler human figure model or even a Puma robot model for testing purposes. More detailed spine, hand and finger models can be substituted for the simpler segments and used in any combination desired.

The joints that connect figure segments typically have up to three rotational degrees of freedom (translational degrees of freedom are allowed but are not used in the human models). Joint centers are described in terms of sites on the connected segments. While real human joints are not so simple, this model suffices for most ergonomic analyses. For a brief discussion of our efforts addressing more flexibility in joint action, see Section 3.6.

2.1.3. Independent surface geometry per segment

For interactive manipulation, detailed human figure surface geometry is usually unnecessary, however, the *psurfs* associated with each segment may be as simple or complex as desired. The default human model has a rather polyhedral appearance to keep the number of polygons low for graphical display update efficiency. The more accurate figures (the contour bodies, Section 2.1.7) may have hundreds of polygons per segment to give a smoother and more rounded appearance. The selections can be mixed from segment to segment: for example, a smoother head model with simple arms.

2.1.4. Other body models

Since the topology and geometry of figures are completely accessible, building any other existing body in *Peabody* and *psurfs* should be a rather simple matter. Transforming the segment topology is straightforward, and most geometry formats are readily converted into *psurfs*. Perhaps the most effort would be involved in establishing commensurate sites on each segment for the joint connections. For example, we are presently converting the Crew Chief model [EAST90] into a *Jack*-compatible figure.

2.1.5. Surface "clothing"

Clothing a figure is important for ergonomic analyses since clothing often affects mobility and joint limits. *Jack* presently contains three types of clothing:

1. A rather simple kind, implemented by Jiahe Lu, which is simply a color differentiation for various segments (e.g. brown legs and lower torso yield "pants," blue upper torso and arms, a long-sleeved "shirt," etc.);
2. A more realistic "thick" clothing, implemented by Eunyoung Koh, which is the actual expansion of the segment geometry (hence its diameter) relative to the segment axis while still preserving the overall shape;
3. Additional equipment (such as helmets, tool belts, pockets, air supplies, etc.) attached or worn by simply adding appropriate geometric models to segments.

All three improve graphics appearance. The second and third approach are the more serious since thick clothing and equipment should affect joint limits. This contextual modification of body capabilities is currently under development (Section 3.6). The attachment of loose fitting or draped clothing is another matter entirely and is not addressed here.

2.1.6. Hand model

Jack contains a simple geometric hand model, constructed by Wallace Ching, with fully articulated and joint limited fingers and thumb. The more interesting feature, however, is an automatic grip. Given a geometric object that is to be grasped, the user can specify one of three types of grips -- power, precision, or disk [IBER87] -- and *Jack* will move the hand to the object then move the fingers and hand into a reasonable grip position. The actual grip is completed by using real-time collision detection on the object's geometry to determine when finger motion should cease.

2.1.7. Biostereometric contour bodies

One of the most interesting body databases in *Jack* is derived from biostereometric (photographically) scanned body surface data of 76 subjects. Originally supplied by Kathleen Robinette of the U.S. Air Force Armstrong Aerospace Medical Research Laboratory, the data consists of approximately 6000 data points for each subject, organized by body segment and arranged in parallel slices. Marc Grosso, Jeff Weinberg, and Pei-Hwa Ho determined reasonable joint centers from the segment contours and surface landmark data, converted the segment topology into *Peabody* structures, and tiled the contours into polyhedral meshes [FUCHS77].

The major difficulty with the contour data is that it looks very realistic in the standard posture, but immediately developes annoying gaps when the joints are moved. Eunyoung Koh recently remedied this in a general fashion that extends as well to the clothing defined by the segment expansion method. Given two adjacent segment geometries, a procedure generates a curved surface to fill in a plausible solid connection between them depending on the joint angle and the respective tangets to the segments. This procedure fills the "gaps" between the scanned segment geometries and, by extension, the clothing defined over them.

The single torso segment in the original scanned body data was rigid. In Section 2.1.8 we describe how we dramatically improved that situation.

2.1.8. 17 segment flexible torso with vertebral limits

The lack of accurate flexibility in the torso is a notable weakness of most anthropometric models. Indeed, the default *Jack* figure and even the biostereometric bodies suffered from torso rigidity. Recently, Gary Monheit has constructed a 17 segment vertebral column (from lumbar to thoracic) whose movements are dictated by kinematic limits and some simple parameters [MONH90][2]. The torso, in turn, is broken into 17 corresponding "fat" slices, one for each vertebra. With this arrangement it is easy to have the contour body "breathe" and bend in a very realistic fashion. Movements of the torso are basically described by lateral, saggital, and axial rotations of the neck. The flexible spine shape is history-dependent, that is, the motion of each vertebra in space is not determined solely by these rotations: other parameters such as motion-resisting joints, and the motion-originating joint affect its actual path.

[2] Previous modeling and simulation by Willmert [WILL82] apparently failed to adequately account for the joint relationships between vertebrae, especially in the neck. Numerical error also appeared to cause trouble. These problems do not plague the *Jack* torso model.

2.1.9. Facial model

Humans have faces, and *Jack* provides two mechanisms for presenting a face on a human figure.

- A photograph of a [real] face may be texture mapped onto a contour body head. The figure bears a close resemblance to a real person and the resulting image looks reasonable even when rotated. The disadvantages are the rather delicate (for correct) positioning of the texture on the head and the requirement of ray-tracing to see the face[3]. Welton Becket and Dawn Vigliotti have provided this feature.
- A polyhedral model of a generic face may be used on a special head *psurf*. The advantages are that the polygons are displayed directly and, most importantly, the facial features are *animated* [PELA90]. While not important (perhaps) for human factors work, the expressions certainly enliven finished animations. The original facial data was supplied by Steve Platt [PLAT85] and extensively modified by Catherine Pelachaud, Soetjianto, and Khairol Yussof.

2.2. Anthropometry

Having a body model is one thing; being able to easily make it correspond to human size variation is another. Anthropometric scaling of body models is an important component of *Jack* [GROS89a].

2.2.1. Segment and joint attributes

The human figures used in *Jack* have various attributes associated with them that are used during manipulation and task analysis. The current set includes segment dimensions, joint limits, moment of inertia, mass, center of mass, and joint strength [GROS89b]. Segment dimensions are used to scale the segment geometry for proper sizing. Joint limits are used to restrict motion. Mass, center of mass and moments of inertia are used during dynamic simulations. The strength data is used for certain reach and lifting tasks. Raw anthropometric measurements (e.g. for specific landmarks or composite measurements such as "sitting height") can also be associated with an individual in the database.

The strength data may be based on tabular (empirical) data or strength prediction formulas [WEI90]. Strength parameters may be either scaling (e.g. gender, handedness) or non-scaling (e.g. depending on the population). In any case the user may alter the stored data or formulas to conform to whatever model is desired.

2.2.2. Population percentiles or individuals

Either population statistics may be used to provide percentile data, or else an actual database of [real] individuals may be used. The former, e.g., is common in U.S. Army analyses, while the latter is often used by NASA for the specific individuals in the astronaut trainee pool.

2.2.3. Spreadsheet interface for selection, changes, or database query

The interface to the anthropometry database is through *SASS*: the Spreadsheet Anthropometric Scaling System [GROS89b]. As part of *Jack* it offers flexible access to all the body attributes and a simple mechanism for changes. Specific body models may be selected or customized as needed. Queries about the contents of the anthropometric database are constructed entirely from pop-up menus without requiring user knowledge of a particular database query language. For example, the query

> "Find all females under the 25th percentile in stature who have left elbow strength greater than 15 ft-lbs."

is constructed by direct menu selection of each field, relation, and value. Individuals satisfying arbitrary requirements may be listed and selected for creation and display. *Peabody* model files are created by

[3] Although more expensive display hardware can do the texture mapping in real-time.

SASS and made available to *Jack*. Alternatively, one can interactively manipulate the current body in *SASS* while displaying it in *Jack* to rapidly try out the effect of varying the individual, population percentile, gender, joint limits, etc.

2.2.4. Concurrent display of interactively selected dimensions

As noted above, a human figure may be modified by *SASS* while it is being displayed in *Jack*. In fact, the process is much more powerful than just updating a display. In Section 2.5.4 we will see that a figure may be subject to arbitrary goals for one or more of its joints. These goals are maintained (subject to joint limits and body integrity) during interactive manipulation. The process also applies to segment attribute changes done interactively in *SASS*: as the segment lengths change, e.g., the body will move to maintain the required position, orientation, or viewing constraints. It is therefore very easy to assess posture and viewing changes (as well as success or failure) across population percentiles or gender.

2.2.5. On-screen interactive strength data display

Besides the numeric listing of strength data, a graphical display feature is available. Interactive displays of joint torque or end-effector forces may be shown in *Jack* as the user manipulates the figure directly [WEI90]. Current as well as cumulative maximum forces or torques are displayed as moving bars in a ***strength box*** whose axes correspond to the joint's degrees of freedom. Individual, gender differentiated, and population percentile (95th, 50th, and 5th) strengths may be compactly and comparatively displayed.

Torques along a joint chain may be shown, too. Given a force on an end-effector, *Jack* can compute the reaction forces generated anywhere else in the body. Since the body is an active mechanism, forces may be resisted in differing amounts by activating different muscle groups. Phil Lee and Susanna Wei have implemented displays that show, given a weight held by an end-effector, the reaction forces (torques) at each joint degree of freedom along a given chain [WEI90]. In addition, a trace of the "safe" and "unsafe" regions (relative to the current strength model) is left in the display as the end-effector is moved about, producing a direct and real-time visualization of the accessible space.

2.3. Body somatotypes

Pei-Hwa Ho has examined the original biostereometric contour body dataset to select specimens covering the approximate midpoint and extremes of body somatotype for each of the 5th, 50th, and 95th percentile males and females: 18 body "styles" in all. This set is integrated into *SASS* with a new attribute for somatotype. The user can select a gender, somatotype, and percentile, causing one of the 18 prototype bodies to be scaled to the individual segment dimensions. The figures retain significant realism in form while providing infinite variability across all shape dimensions.

2.3.1. Multiple figures

Jack allows the manipulation and display of as many figures as desired up to the memory limits of the hardware. There are no restrictions whatsoever on the geometry, topology, or anthropometry used across the several figures.

2.4. User Interface

One of the the most attractive features of *Jack* is the natural user interface into the three-dimensional world [PHIL88]. A significant part of the interface is offered by the hardware capabilities of the Silicon Graphics Iris 4D workstation platform upon which *Jack* is built. The software, however, makes this hardware power controllable.

2.4.1. Three button mouse and keyboard

Jack relies solely on the standard Iris 4D three button mouse and keyboard for interaction. The mouse is used to perform direct manipulation on the 3-D scene, e.g. selecting objects by picking their images, translating objects by holding down one or two mouse buttons corresponding to spatial coordinates, etc. The mouse is also actively used to negotiate through the pop-up command menus.

The keyboard is used for occasional command entry. The escape and control keys are used as meta-mouse buttons, e.g. to change the button interpretations from translation to rotation, or the affected coordinate frame from global to local.

2.4.2. Menu-driven commands

The command menus are built from the standard Iris menu library. There is a tradeoff in making all the commands accessed this way: simplicity and an uncluttered screen are advantages, while on the other hand frequently used commands are treated the same as infrequently used ones.

2.4.3. Command completion

To alleviate the menu bottleneck, any *Jack* command maybe entered via the keyboard. To save typing, and to act as a simple help system, command completion shows the choices for any partially-entered command.

2.4.4. Natural 3-D interactive interface

The naturalness of the interface arises from the coherence of hand motions with the mouse and the correspondence between mouse cursor motion on the 2-D screen, a 3-D cursor (looking like a "jack") in the world, and 3-D objects displayed there. In particular, translations and rotations are selected with the mouse buttons (and perhaps a key), and the mouse motion is transformed into an appropriate 3-D cursor movement. Rotations display a wheel perpendicular to the selected axis; motion of the mouse cursor about the wheel display invokes a 3-D rotation about the actual axis. Any joint limits are respected.

Object selection is done by simply placing the mouse cursor over the desired part. If more than one object lies under the cursor, a button push cycles among the possibilities which are highlighted in turn.

Other motions that are easy to perform in *Jack* include real-time end-effector dragging (Section 2.5.5). The position and orientation of the end-effector is controlled by the same mouse and button interpretation method.

2.4.5. Multiple windows

Jack supports multiple independent windows into the current environment. Thus, e.g. one could be a global view, one could be a view from a figure's eye, another could be a view from a certain light source (to see what is being illuminated), etc. The camera and lights are represented as *psurfs* so that they may be positioned and observed just as any other object in the scene. Of course, as the camera is moved in one view, the corresponding camera view window shows the changing image. The same result obtains if a window view is attached to a figure's eye (or hand, etc.).

2.4.6. Perspective or orthographic views

The view in a window may be either perspective or orthographic. The latter is most useful when dimensions are important. In perspective, the three orthographic projections may be optionally displayed within the same window as wireframe "shadow" images on the imaginary walls and floor. These greatly assist in object and goal positioning.

2.4.7. Feature on/off toggles

There are a number of display features which may be turned on or off at will. These include the orthographic projections, a ground plane, shaded or wireframe mode, background star field, motion traces, and so on.

2.4.8. Command language files

Any *Jack* scene can be written out as an environment file; when read in it restores the exact situation for continued manipulation. Even more useful is the *Jack command language* or *jcl* file. This is a record of the *Jack* commands issued during a selected portion of an interactive (or program-controlled) session. The *jcl* files may be recursive in the sense that they may contain commands to read and execute other *jcl* files. Such files also provide a command format for external (non-interactive) control of *Jack*, e.g. from an animation procedure.

2.5. Positioning

The manipulation power in *Jack* comes from novel real-time articulated figure positioning algorithms. These embue the jointed figure with "behavioral intelligence"; that is, the ability to respond to varied positioning *goals* as well as to direct joint rotation.

2.5.1. Joint degrees of freedom

Joint angles may be manipulated directly to position a figure. During rotation, a rotation wheel will appear only for allowed degrees of freedom.

2.5.2. Rotations subject to joint limits

During rotation, the displayed wheel will follow the cursor, but the joint will only be allowed to rotate to the joint limits. For two and three degree of freedom joints, the joint limits are tested in the individual rotation directions. This is not totally correct, especially for a complex joint such as the human shoulder, but it suffices for most purposes. Adding more accurate joint limits is possible in the future (Section 3.6).

2.5.3. End-effector position and orientation goals

One of the most powerful features of *Jack* is the positioning of a joint by specifying the other end of the kinematic chain (e.g. the shoulder or the waist for a hand movement), and giving the end-effector an arbitrary position or orientation goal (or both) in space. Using a real-time inverse kinematics procedure based on nonlinear optimization (with linear constraints) written by Jianmin Zhao, a joint space solution (subject to joint limits) is computed for the intermediate joints along the chain [PHIL90, ZHAO89]. The solution moves the selected joint (end-effector) to the goal if it is reachable, otherwise it moves as close as feasible given the figure posture, the joint chain, and the joint limits. Any failure distance is reported numerically as well. This movement does not represent how a person would actually move, nor does it attempt to find the "best" or most "natural" position. It merely achieves goals. For better postures, additional goals can be created and maintained (Section 2.5.4).

There are several goal types available:

- position (a point in space)
- orientation (e.g., a particular orientation of the proximal segment coordinate space)
- position and orientation (weighted to arbitrate conflicts: e.g. a position may be achievable only by violating the orientation goal or *vice versa*, so the weight determines which to favor)
- aim (a specified direction on the proximal segment should point at the desired point; this is frequently used for eye and camera positioning)

- view (a specified direction; like "aim" except that twist is not allowed so the camera or eye view will not rotate (roll) along its sighting axis)
- line (a position anywhere along the line is acceptable)
- plane (a position anywhere in the plane is acceptable)
- half-space (a position anywhere in the half-space volume is acceptable)

In order to distribute intermediate joint motions more realistically, a stiffness parameter can be set for each joint degree of freedom if desired, or for the chain as a whole. The stiffness forces motion to be favored or resisted more in a degree of freedom, e.g. to encourage torso bending rather than twisting. Over a chain, the stiffness may favor motion at the proximal or at the distal end.

Inverse kinematics executes in "real-time," meaning that most positioning actions are accomplished in time that is not much different than that required by a real person.

2.5.4. Multiple simultaneous position and orientation goals

The goal satisfaction procedure has the additional advantage of operating on multiple simultaneous goals of any of the above types. For example, a figure can be seated by supplying goals for the feet (to stay on the floor), the center hip (to be near and just above the chair seat), the knees (to stay in front of the hips) and the neck (to stay above the waist). Some of these goals may be plane goals (such as for the feet) or half-plane goals (to keep the waist above the seat and in front of the chair back). As usual, joint limits are respected and the best solution (though it may be a local rather than global minimum) satisfying the goals is displayed. If the goals are not entirely satisfiable, some minimum distance solution will be offered. If the results are not acceptable, more goals may be added. This algorithm still runs in real time for modest numbers of goals; it is superlinear convergent with each iteration of complexity of just $O(nm)$ where n is the number of degrees of freedom and m is the number of goals. A sample posture to move the figure's head over the end of a large upright tube, aim the view to see the bottom of the tube, and grasp the tube with two hands at opposite sides while keeping the elbows out in a plane parallel to the tube axis took only 23 seconds to solve on a Personal Iris workstation.

2.5.5. Real-time end-effector dragging

Since inverse kinematics is available, and since multiple goals may be active, *Jack* allows a joint to be moved interactively by attaching a position or orientation goal to the 3-D cursor controlled by the mouse [PHIL90]. The solution time is actually reduced because the current posture is likely to be close to the solution at the next input position, so the algorithm converges quickly. To avoid waiting for the solution, however, *Jack* updates the joint angles at every graphics window update by taking the solution obtained thus far. As the goal is moved or rotated, the posture changes as quickly as possible and "catches up" with the user whenever there is a significant pause in cursor motion.

2.5.6. Rotation propagation when joint limits are exceeded

A consequence of joint limits and inverse kinematics is an apparent "behavioral intellegence" in the manipulated figure. If the wrist is twisted, the rotations propagate along the arm toward the fixed end as joint limits are encountered. Thus the user can freely move the joints about and the remainder of the body will act in a reasonable fashion.

2.5.7. Constrain center of mass of entire figure

Cary Phillips developed an interesting application of the multiple goal solution algorithm by constraining the center of mass of a figure. The center of mass is not a specific joint or point of the body, rather it is a computed quantity. Nonetheless, *Jack* permits it to be a participant in a goal. By constraining the center of mass to lie along a line goal above the figure's support polygon, a *balanced* *reach* may be effected. The motion is most dramatic when only one foot is constrained to the floor; moving a hand causes the other leg to lift off the floor for counterbalance when it is needed![4]

2.6. Animation

The manipulations in *Jack* discussed so far are not really "animations" as we have already mentioned. For an animation we expect some coherence and smoothness to the figure's motion; it is not enough to merely animate a numerical search so matter how clever or effective it is. *Jack* incorporates a number of mechanisms to produce human-like motion.

2.6.1. Key parameter specification and rational spline interpolation

The simplest method of animation is based on the specification of a series of postures and the subsequent interpolation of the joint angles to create a smooth sequence of "in-between" postures. The important postures are called "keys"; the joint angles from key to key are parameterized by time (given for each key) and interpolated to compute values at any other time. A textual interface written by Jean Griffin helps in the creation and editing of the script keys and times. Jianmin Zhao implemented a method of interpolation and motion control using rational spline curves in a fashion similar to that described in [STEK85].

While capable of creating effective motion sequences, key parameter approaches put the burden of motion design on the user/animator, requiring skill and patience to define the key postures. There are alternatives, but each with advantages and disadvantages. *Jack* includes a useful selection, as we explain below.

2.6.2. Forward dynamics

One method of creating accurate and realistic motion is to use the physics of forces and torques to drive a figure (e.g. [ARMS87, WILH87]). The results are physically correct, but the problem is in determining the proper directions, magnitudes, and timings of the forces. Most people (or animators) cannot do that. Moreover, the motions tend to work best on passive figures (when they are under the control of external forces, e.g. falling, dangling, crashing) rather than on active ones (when the person is trying to perform some task, e.g. reaching, lifting, throwing).

For completeness, *Jack* offers motion control by force and torque specification using forward dynamics to compute joint positions and orientations. This procedure is being built by Mike Edwards.

2.6.3. Strength guided motion

A rather more useful, but more restricted, animation method developed by Phil Lee uses the inherent strength model stored for a human figure as the basis for computing certain types of motion. If the task involves moving a weight rather slowly to some goal position, then a *strength guided motion* algorithm computes a motion path based on the strength model and two additional parameters [LEE90]. The parameters are the comfort level at which the motion should be performed and the allowed deviation from a straight-line path to the goal. Using a number of strategies based on the available torque at each joint in an arm (plus upper torso) joint chain, the algorithm computes an acceptable posture at every instant (say, 15 times a second) of the action. Strategies include moment reduction, pull back, adding joints, and recoil to bring comfort to acceptable levels. Present limitations are two-dimensional paths, upper body chains, and inverse kinematics incremental positioning rather than a more realistic dynamical

[4] I am personally very fond of the "worm on a fishhook" example: A linear chain of segments (the worm) is attached at one end to a point in space (the hook) and allowed to dangle below. As the free end of the worm is dragged about, the segments wiggle and contort to maintain the center or mass along a line goal below the hook. Very dramatic.

rate-control process (Sections 3.3 and 3.4). A useful range of lifting and reaching motions may be produced already, however, including weight lifting, rising from a chair, pulling the body upwards in a chin-up, and two-person coordinated lifting.

2.6.4. Hand grip

Using the real-time collision detection capabilities in *Jack* (Section 2.7.5) a hand may be made to grip any object. (Section 2.1.6 already mentioned this.) The motion is an animation in the sense that the fingers are moved until they contact the object. The grip is fake, though, as the object is merely attached via a constraint (in the *Peabody* sense) so that it moves in concert with the hand.

2.6.5. External control

Cary Phillips and Jeff Esakov coded the beginnings of a very powerful animation mechanism in *Jack*. By attaching a joint angle or some other parameter such as arm length or object size to a Unix *socket*, external processes can be used to control their values. The process can reside anywhere on the Ethernet attached to the host workstation, including the host itself. The process can be another user, an autonomous procedure, a physical sensor, or a simulation. Thus a gauge needle's rotation can be controlled by an external simulation, a joint angle could be read from a goniometer on an actual subject, or an object's location could be controlled by another user interacting in the same space[5].

2.7. Analyses

All the *Jack* features are available to compute certain aspects of some of the most commonly performed task analyses.

2.7.1. Reach space

Jack can display a trace of any site; in particular, it can show the path of an end-effector as it is manipulated. The resulting trace gives a good idea of the reachable space as the end-effector is dragged about. Any joint chain can be used due to the general inverse kinematics solution. Other algorithms being studied by Tarek Alameldin can compute the reachable space boundary or volume off-line [ALAM90].

2.7.2. Eye view

We have already seen that *Jack* can show the view from any object, in particular, a figure's eye. Besides the normal perspective view, a simplified retinal projection window may be drawn. Objects in front of the eye are mapped into a (radius, angle) polar plot. When features such as foveal or peripheral areas are drawn in the retinal window, the relative visibility of scene features may be assessed[6].

2.7.3. Translucent view cones

In addition to the retinal window, translucent view "cones" may be displayed from the eyes of a human figure. With the apex at the eye lens center, the shape of the cones follows any desired polygonal path, e.g. foveal area. By aiming the eyes with an interactive goal, the view cones follow the point of interest, converging or diverging as needed (subject to eye "joint" limits). Since the cones are translucent, workplace objects show though, giving the user a good impression of what can and cannot be seen by the subject.

[5] Sometimes called *Virtual Environments* or *Virtual Reality*, e.g. [BLAN90].

[6] Much of the useful effort in this analysis mode was accomplished by a collaboration between Cary Phillips of our lab, Aries Arditi of The Lighthouse in New York, and Mike Prevost of the NASA Ames A^{3}I project.

2.7.4. Torque load and comfort during reach

Since the strength guided motion computes the instantaneous joint torques in the current (changing) body posture, this information is available for display. During such actions, moving bar charts can show the level of comfort, physical work, or fatigue.

2.7.5. Real-time object-object collision detection

In collaboration with the GRASP (General Robotics and Active Sensory Perception) Lab in our department, a real-time collision detection facility[7] was added to *Jack*. For efficiency, only a pair of selected objects are checked in real-time. The general problem is very costly (time-consuming) in complex, changing environments. Given that the user is normally in control of the simulated figure's motion, the limitation to checking, say, a lower arm against another object is useful but clearly sub-optimal. As we have mentioned, it is used to accomplish the hand grip in *Jack*.

2.7.6. Interactive body sizing under active constraints

We have already mentioned in Section 2.2.4 that changes to bodies made in *SASS* would maintain (as well as possible) any active constraints. Thus testing workplace reaches over any population range is nearly trivial: e.g. constrain the feet or lower body, set the reach goal for the desired end-effector, and alter the percentile field of the appropriate *SASS* spreadsheet display. In another situation, suppose the eye is constrained to the design eye point of a cockpit, the hands and feet are positioned to appropriate goals, and the shoulders and hips are restrained by point goals representing a suitable restraint system. Then running through the percentiles with reach goals for the hands, feet, and hips will show how well or how poorly the population can carry out that task.

2.7.7. Hooks to AI-based simulation system and Knowledge Base

The ultimate analysis tool is a simulation which executes some task and drives the human figure with a set of goals and timings. We are actively working in this area. Jeff Esakov is building a system, called *YAPS*, which is basically an object-oriented discrete event simulator running over a Knowledge Base [ESAK89, ESAK90]. Jugal Kalita has constructed verb semantics describing generic methods for achieving certain goals [KALI90]. Presently the lexicon of executable tasks includes computational definitions for *open, close, push, pull, put, place, slide, reach*, and *look-at*, as well as a few spatial prepositions and adverbial modifiers. A temporal planner organizes the goals in a reasonable order [BADL88, ESAK90]. A human performance rate predictor based on Fitts' Law (if appropriate) [FITT54] is used to postulate reasonable task durations for reach and viewing actions [ESAK89]. At the highest level, simple natural language task commands are accepted and animated [KALI90, BADL90]. The *YAPS* sysem supports some simple task planning and task interruption capabilities.

The *YAPS* simulation and Knowledge Base are written in CommonLisp. *YAPS* drives *Jack* figures through the UNIX socket interface. Our *YAPS* simulation is migrating from a Hewlett-Packard workstation implementation onto the Iris. At NASA Ames, the *MIDAS* simulator performs a similar function, communicating parameters over the network and driving the *Jack* figure as a helicopter pilot mannequin.

2.8. Rendering

Besides the hardware rendering available for polyhedral models on the Iris workstation, the *Jack* system includes a sophisticated ray-tracer written by Welton Becket. Its capabilities include anti-aliasing, textures, specularity, translucency, reflections, shadows, multiple light sources, material properties, and

[7] Thanks to Janez Funda, who needed it for a telerobotic application.

chromatic aberrations. It successfully rendered hundreds of images for a movie containing over 45,000 polygons at an average rate of about 4 images per 45 minutes an Iris 4D/240.

2.9. External Interfaces

Jack can obtain geometric information from several commercial systems. This list grows as sponsors require *Jack* to handle data from diverse CAD systems.

- Wavefront Technologies (Preview and Model format; interface written by John Granieri)
- Pixar Renderman (For image output)[8]
- SDRC I-DEAS (Universal file format; interface written by Cary Phillips)
- MultiGen (A polygon modeler; interface written by Cary Phillips)
- BRL-CAD (The Constructive Solid Geometry objects are polygonalized through code written by Osman Niazi; the Boolean operations are applied to these polygonalized objects in *Jack* based on an algorithm from Brown University [LAID86])
- Utah Raster toolkit RLE image files (Used for image manipulation)

There are some animation hardware and hardcopy capabilities supported in the *Jack* environment:

- Interface to Abekas A60 digital image store (written by Joe Procopio)
- Interface to Lyon-Lamb animation controller
- Hardcopy image output via Tektronix 4693 (RGB format) and Apple laserwriter (via Postscript)

3. Work in Progress

Jack is an evolving system with continual enhancements motivated by our desire to achieve certain graphic and animation goals as well as provide ever more powerful and usable human performance understanding and modeling. The following sections outline some of the enhancements in progress or scheduled for the near future.

3.1. Additional strength data

The present strength data for the arms must be augmented by similar data for the upper torso. Hand (grip) strength would also be a useful addition. The strength data we use is for isometric exertion and does not necessarily reflect proper values for strength during motion. There are many issues surrounding the validity of strength data. We prefer that the user supply an acceptable strength model simply because ours is probably not very good. *SASS*, however, makes changing the strength prediction functions or adding new tabular empirical data rather straightforward.

3.2. Fatigue model

During strength guided motion, *Jack* can compute a measure of the work or energy expenditure per unit time. This should be expressible as a muscle group load and hence generate some specific strength loss due to fatigue. Phil Lee is incorporating a reasonable fatigue model into *Jack* so that strength changes can dynamically affect movement (or the mere holding) of a weight.

[8] In progress.

3.3. General force trajectory

The limits to strength guided motion must be relaxed. One is to extend the algorithm to 3-D motions. Fortunately, most movements are planar (at least over short distances [MORA86], so this does not appear to be a major difficulty. Somewhat more important is having a satisfactory strength model. The ability of *SASS* to interpolate strength values is critical to success here.

3.4. Dynamics rate-control during strength reach

Strength guided motion uses inverse kinematics to do the incremental positioning of the end-effector. A more accurate model is being developed by Phil Lee and Wallace Ching that uses a dynamics approach to insure that end-effector motion does not exceed realistic and consistent accelerations. There will be interesting interactions between the concurrent needs to reach the goal, sustain coherent muscle group [strength] activity, monitor comfort levels, and manage fatigue.

3.5. Walk procedure

The motions of the figure often appear stilted as it is unable to locomote other than by floating or sliding. Bill Kriebel is implementing a walk procedure based on Bruderlin and Calvert's model [BRUD89]. A reach task involving the entire body will then use locomotion to bring the end-effector within a suitable distance of the goal. (A definition of "suitable" must be determined.) Concomitant problems include path planning and collision avoidance if obstacles are present. A preliminary *Jack*-compatible spatial path planner written by Chris Yu based on the algorithm by Lozano-Perez [LOZA79] is available for experiments.

3.6. Dependent joints

The original *Peabody* and *psurf* structure, while robust, must be enhanced to permit groups of joints to work together as a unit. The idea is that these joint dependencies provide for more natural motion and easier control. The 17 segment spine and torso is a good example of the kind of dependency that is required. Other examples include clavicle motion as a function of shoulder position [OTAN89, BADL89] and head motion dictated by eye direction [SPAR89]. Jianmin Zhao and Cary Phillips are working out the changes in *Jack* needed to incorporate such structures.

Related problems include complex shoulder joint limits based on the shoulder position rather than just the geometrically required three [independent] degrees of freedom. This assumes even greater importance when dealing with the computation of joint limits based on clothing. It may not be possible to pre-compute the limits; rather, they may have to be detected as a certain tolerable level of intersection (collision) between adjacent segments and their attached geometry. In this case, joint motion is determined by segment compressability[9].

3.7. Anthropometry updates

Jiahe Lu is considering a significant set of changes to *SASS*. These include a corrected implementation of segment percentiles within a population, more appropriate segment scaling relative to the given population, stature adjustments when certain individual segment dimensions are changed, and global (cross-attribute) effects such as mass changes when sizes are changed.

Additional populations are also being examined for conversion into *SASS* format, especially the

[9] The converse problem is somewhat easier; e.g. see [GOUR89, THAL90, CHAD89] for segment deformation given joint angles. The finite element approach may be viable for our version of the problem as well.

recent U. S. Army soldier data.

An attractive idea for training applications is to read the user's own anthropometry from a login file, and use his or her body description as the default scaling for the *Jack* figure. Tasks being performed would then be sympathetic to the user's own capabilities.

3.8. Clothing experiments

Joe Procopio is trying some simplified methods for defining clothing. One "trick" is to represent the garment as an articulated figure with "joints" at various seams. The garment is then brought to the proper shape by applying multiple simultaneous goals to bring the various parts into proper alignment or contact (e.g. [point] buttons to [point] button holes, zippers to line goals, etc.). Additional goals position the garment on the figure by identifying major points of contact (e.g. shoulders, front of chest, elbow, etc.). Some of this work is being done for the Army Natick Labs with Steve Paquette.

3.9. Three-dimensional input devices

In a previous system we experimented with direct 3-D input through a Polhemus 6-degree of freedom digitizer [BADL87]. We were limited by the rather slow speed of the inverse kinematics algorithm then available to us. Moreover, that algorithm suffered by providing a solution that was too local. With the new inverse kinematics procedure in *Jack* it should be easy to connect a spatial input device to drag the selected end-effector around in direct mimicry of the user's hand motion.

3.10. Passive position sensing

Since the early 1970's we have tried to understand how a computer could be programmed to observe human activity and describe or at least mimic the motions in a computer graphics model [BADL76, OROU80]. The ability to control a realistically shaped and behaved human figure with *Jack* opens the possibility of real-time monitoring activities. The inverse kinematics procedures may be robust enough to work from a few *two*-dimensional (e.g. image plane) joint positions and known anthropometric dimensions to establish 3-D locations for all the joints. Thus, given a [real] person performing some task in a remote location and passive monitoring from one or more video cameras, a simulated figure of the same size could be fit in *real-time* to the acquired positions. This real-time automated modeling will permit the indirect and low cost monitoring of EVA or other novel work activities where physical mock-ups are currently the only option. The computer models can be used for task planning, safety testing, task load predictions, and -- by making measurements on the simulated model -- indirect assessment of physiological states such as fatigue or comfort without direct sensing or verbal communication.

3.11. High level task control

Controlling human motion tasks specified by language commands or instructions is a long-term goal of our research. Analysis of the form and content of instructions has begun in collaboration with Computer and Information Science department faculty members Bonnie Webber and Mark Steedman [BADL90].

3.12. Task planner

One of the principal issues involved in understanding and executing instructions is the form of the action planner. Classical planning strategies do not seem to suffice for human motion because people are highly redundant mechanisms and use flexible, incremental, and interruptable plan execution. A reactive and incremental planning scheme for executing conditional and temporal instructions is being investigated by Moon Jung.

3.13. Task performance time database

The Fitts' Law formulation for task time performance is adequate for very simple reach and view tasks. For more generality the strength model can be referenced to obtain estimates of minimum trajectory times. This approach, however, is limited to knowing the strength model and, moreover, does not adequately compute timings for more complex task units (e.g. inserting a bolt into a hole). Libby Levison is examining several task time databases to see how they might be incorporated into the planner. These databases will be extremely useful for task analyses in the maintenance domain where nominal time-motion studies for common tasks have been extensively measured.

3.14. Language and speech interfaces

Once the natural language instructions can be used to generate a plan for execution by the simulated figure in *Jack*, a next step is to try speech input for the same set of understood commands. This work is presently underway in collaboration with Christoph Rumpf of Siemens Corporate Research in Munich using their speech understanding system.

4. Conclusion

Even though *Jack* is under continual development, it has nonetheless already proved to be a substantial computational tool in analyzing human abilities in physical workplaces. It is being applied to actual problems involving space vehicle inhabitants, helicopter pilots, maintenance technicians, foot soldiers, and tractor drivers. This broad range of applications is precisely the target we intended to reach. The general capabilities embedded in *Jack* attempt to mirror certain aspects of human performance, rather than the specific requirements of the corresponding workplace. There is only one "version" of *Jack*; though its features are sometimes motivated by a particular application, the solutions are shared by all who support the research effort. Of course, there are some general problems we wanted to solve that have contributed much to *Jack* from our own research perspective. We have enough on this queue to keep us busy for a long time.

Acknowledgments

Thanks to all the Computer Graphics Research Lab members who read this paper to keep it honest.

This research is partially supported by Lockheed Engineering and Management Services (NASA Johnson Space Center), NASA Ames Grant NAG-2-426, FMC Corporation, Martin-Marietta Denver Aerospace, Deere and Company, NSF CISE Grant CDA88-22719, and ARO Grant DAAL03-89-C-0031 including participation by the U.S. Army Human Engineering Laboratory and the U.S. Army Natick Laboratory.

References

[ALAM90] Tarek Alameldin, Tarek Sobh, and Norman I. Badler. An adaptive and efficient system for computing the 3-D reachable workspace. *IEEE International Conf. on Systems Engineering*: 503-506, Pittsburgh, PA, Aug. 1990.

[ARMS87] W. Armstrong, M. Green, and R. Lake. Near-real-time control of human figure models. *IEEE Computer Graphics and Applications*, 7(6):52-61, June 1987.

[BADL76] Norman I. Badler. *Temporal scene analysis: Conceptual descriptions of object movements.* University of Pennsylvania, Computer and Information Science, Technical Report MS-CIS-76-4. (Also PhD dissertation, Department of Computer Science, University of Toronto, 1975).

[BADL88] Norman I. Badler, Scott Kushner, and Jugal Kalita. *Constraint-based temporal planning*. Technical Report, Dept. of Computer and Information Science, Univ. of Pennsylvania, Philadelphia, PA, 1988.

[BADL89] Norman I. Badler, Philip Lee, Cary Phillips, and Ernest Otani. The *Jack* interactive human model. *Concurrent Engineering of Mechanical Systems*, Vol. 1:179-198. (First Annual Symposium on Mechanical Design in a Concurrent Engineering Environment, Univ. of Iowa, Iowa City, IA) October 1989.

[BADL87] Norman I. Badler, Kamran Manoochehri, and Graham Walters. Articulated figure positioning by multiple constraints. *IEEE Computer Graphics and Applications*, 7(6):28-38, June 1987.

[BADL90] Norman I. Badler, Bonnie L. Webber, Jugal Kalita, and Jeffrey Esakov. Animation from instructions. In *Making Them Move: Mechanics, Control, and Animation of Articulated Figures*. N. Badler, B. Barsky, and D. Zeltzer (eds.):51-93, Morgan-Kaufmann, 1990.

[BRUD89] Bruderlin, A. and T. W. Calvert. Goal-directed, dynamic animation of human walking. *Computer Graphics* 23(3):233-242, 1989.

[BLAN90] Chuck Blanchard, Scott Burgess, Young Harvill, Jaron Lanier, Ann Lasko, Mark Oberman, and Michael Teitel. Reality built for two: A virtual reality tool. *Computer Graphics* 24(2):35-36, 1990.

[CHAD89] John E. Chadwick, David R. Haumann, and Richard E. Parent. Layered construction for deformable animated characters. *Computer Graphics*, 23(3):243-252, 1989.

[EAST90] Jill Easterly. Crew-Chief: A model of a maintenance technician. AFHRL Technical Paper 90-19, 1990.

[ESAK90] Jeffrey Esakov and Norman I. Badler. An Architecture for Human Task Animation Control. In *Knowledge-Based Simulation: Methodology and Applications*. P.A. Fishwick and R.S. Modjeski (eds.), Springer Verlag, New York, 1990.

[ESAK89] Jeffrey Esakov, Norman I. Badler, and Moon Jung. An investigation of language input and performance timing for task animation. *Proc. Graphics Interface '89*: 86-93, Waterloo, Canada, 1989.

[FITT54] P. Fitts. The information capacity of the human motor system in controlling the amplitude of movement. *Journal of Experimental Psychology*, 47:381-391, 1954.

[FUCH77] H. Fuchs, Z. M. Kedem, and S. P. Uselton. Optimal surface reconstruction from planar contours. *Comm. of the ACM*, 20(10):693-702, 1977.

[GOUR89] Jean-Paul Gourret, Nadia Magnenat-Thalmann, and Daniel Thalmann. Simulation of object and human skin deformations in a grasping task. *Computer Graphics*, 23(3):21-30, 1989.

[GROS89a] Marc R. Grosso, Richard D. Quach, Ernest Otani, Jianmin Zhao, Susanna Wei, Pei-Hwa Ho, Jiahe Lu and Norman I. Badler. *Anthropometry for computer graphics human figures*. Technical Report, Dept. of Computer and Information Science, University of Pennsylvania, Philadelphia, PA, 1989.

[GROS89b] Marc Grosso, Richard Quach, and Norman I. Badler. Anthropometry for computer animated human figures. In *State-of-the Art in Computer Animation*, N. Magnenat-Thalmann and D. Thalmann (eds.): 83-96, Springer-Verlag, 1989.

[IBER87] T. Iberall. The nature of human prehension: Three dextrous hands in one. *IEEE Conference on Robotics and Automation*, 396-401, 1987.

[KALI90] Jugal K. Kalita and Norman I. Badler. Semantic analysis of action verbs based on animatable primitives. *Proc. 12th Annual Conference of the Cognitive Science Society*: 412-419, July 1990.

[LAID86] David H. Laidlaw, W. Bemjamin Trumbore, and John F. Hughes. Constructive solid geometry for polyhedral objects. *Computer Graphics* 20(4):161-170, 1986.

[LEE90] Philip Lee, Susanna Wei, Jianmin Zhao and Norman I. Badler. Strength guided motion. *Computer Graphics*, 24(4):253-262, 1990.

[LOZA79] Tomás Lozano-Pérez and Michael Wesley. An algorithm for planning collision-free paths among polyhedral obstacles. *Comm. of the ACM*, 22(10):560-570, October 1979.

[MONH90] Gary Monheit and Norman I. Badler. A kinematic model of the human spine and torso. In preparation. To be submitted to *IEEE Computer Graphics and Applications*, 1990.

[MORA86] P. Morasso and V. Tagliasco (eds.). *Human Movement Understanding*. North-Holland, New York, 1986.

[OROU80] Joseph O'Rourke and Norman I. Badler. Model-based image analysis of human motion using constraint propagation. *IEEE Trans. PAMI*, 2(6):522-536, November 1980.

[OTAN89] Otani, Ernest. *Software tools for dynamic and kinematic modeling of human motion*. Master's thesis, Department of Mechanical Engineering, University of Pennsylvania, Technical Report, MS-CIS-89-43, Philadephia, PA, 1989.

[PELA90] Catherine Pelachaud. *A 3D animation system for facial expression, emotion, and intonation*. Forthcoming PhD Dissertation, Department of Computer and Information Science, University of Pennsylvania, 1990.

[PHIL88] Cary Phillips and Norman I. Badler. Jack: A toolkit for manipulating articulated figures. *ACM/SIGGRAPH Symposium on User Interface Software*: 221-229, Banff, Canada, 1988.

[PHIL90] Cary Phillips, Jianmin Zhao and Norman I. Badler. Interactive real-time articulated figure manipulation using multiple kinematic constraints. *Computer Graphics* 24(2):245-250, 1990.

[PLAT85] Stephen Platt. *A structural model of the human face*. PhD Dissertation, Department of Computer and Information Science, University of Pennsylvania, 1985.

[SPAR89] D. L. Sparks. The neural control of orienting eye and head movements. *Proc. of the Dahlem Conference on Motor Control*, Berlin, 1989.

[STEK85] Scott Steketee and Norman I. Badler. Parametric keyframe interpolation incorporating kinetic adjustment and phrasing control. *Computer Graphics* 19(3):255-262, 1985.

[THAL90] Nadia Magnenat-Thalmann and Daniel Thalmann. Human body deformations using joint-dependent local operators and finite-element theory. In *Making Them Move: Mechanics, Control, and Animation of Articulated Figures*. N. Badler, B. Barsky, and D. Zeltzer (eds.):243-262, Morgan-Kaufmann, 1990.

[WEI90] Susanna Wei. *Human strength database and multidimensional data display*. PhD Dissertation, Department of Computer and Information Science, University of Pennsylvania, 1990.

[WILH87] Jane Wilhelms. Using dynamic analysis for realistic animation of articulated bodies. *IEEE Computer Graphics and Applications*, 7(6):12-27, June 1987.

[WILL82] K. D. Willmert. Visualizing human body motion simulations. *IEEE Computer Graphics and Applications*, 2(9):35-38, November 1982.

[ZHAO89] Jianmin Zhao and Norman I. Badler. *Real time inverse kinematics with joint limits and spatial constraints*. Technical Report MS-CIS-89-09, Department of Computer and Information Science, University of Pennsylvania, Philadelphia, PA, 1989.