TASK-ORIENTED COMPUTER ANIMATION OF HUMAN FIGURES

Norman I. Badler

MS-CIS-88-34
GRAPHICS LAB 21

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

June 1988

NATO AC/243, Panel 8, Research Group 9 Workshop on Application of Human Performance Models to System Design

---

Acknowledgements: This research is partially supported by Lockheed Engineering and Management Services, Pacific Northwest Laboratories B-U0072-A-N, the Pennsylvania Benjamin Franklin Partnership, NASA Grant NAG-2-426, NSF CER Grant MCS-82-19196, NSF Grants IST-86-12984 and DMC-85-16114, and ARO Grant DAAG29-84-K-0061 including participation by the U.S. Army Human Engineering Laboratory.

TASK-ORIENTED COMPUTER ANIMATION OF HUMAN FIGURES

Norman I. Badler

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

ABSTRACT

The effective computer animation of human figures is an endeavor with a relatively short history. The earliest attempts involved simple geometries and simple animation techniques which failed to yield convincing motions. Within the last decade, both modeling and animation tools have evolved more realistic figures and motions. A large software project has been under development in the University of Pennsylvania Computer Graphics Research Facility since 1982 to create an interactive system which assists an animator or human factors engineer to graphically simulate the task-oriented activities of several human agents. An interactive system called TEMPUS and its high performance successor is outlined which is intended to graphically simulate the task-oriented activities of several human agents. Besides an anthropometric database, TEMPUS offers multiple constraint-based joint positioning, dynamic simulation, real-time motion playback, a flexible three-dimensional user interface, and hooks for artificial intelligence motion control methods including hierarchical simulation, and natural language specification of movements. The overall organization of this project and some specific components will be discussed.

HUMAN TASK ANIMATION

With the widespread acceptance of three-dimensional modeling techniques, high-speed hardware, and relatively low-cost computation, modeling and animating one or more human figures for the purposes of design assessment, human factors, task simulation, and human movement understanding has become quite feasible. Though not recent, the demand for creating, modeling, and controlling one or more human figures in a 3-D world is expanding and the application base is growing. Human figure models have long been used in cockpit and automobile occupant studies (Dooley, 1982); now they are finding application in vehicle and space station design, maintainence assessment, product safety studies, and computer animation for its own sake (Badler, 1987). When motion information is measured directly off human subjects the result is natural motion but little theory of how such motion can be synthesized.

The scope of the task animation process is much broader than usually realized: to produce convincing animation without an expert animator requires a computational understanding of motion and its "semantics"; in other words, a synthetic "expert." Our intention is to extend the

capabilities of the design engineer, the human factors analyst, or even the casual user to create, animate, and evaluate human performances. Especially in an engineering rather than artistic environment, users will need an effective motion design and analysis tool without feeling pressed to become overly involved in the mechanism of producing animations.

In actuality we must be careful that reducing the inherent complexity of human animation by simplifying one dimension does not squeeze the difficulty into another. We counter this in two ways: first by providing motion specification tools that move closer to verbal descriptions of tasks and motion characteristics; and second by providing both graphical and textual interfaces to a multiplicity of expressive systems. The consequence of the former is that the more common skill of verbal rather than artistic expression may become a vehicle for task control. The consequence of the latter is that the sheer variety of human movement probably precludes any single simple method or interface. Thus it is rather pointless to argue the *general superiority* of dynamics, kinematics, key parameters, local motor control, etc.: each method has its individual strengths but all are necessary. Instead of seeming clumsy and inelegant, the diversity of methods can in fact be nicely embedded in a formal framework (Badler, 1986, Badler and Dadamo, 1988).

THE UNIVERSITY OF PENNSYLVANIA *TEMPUS* PROJECT

The human figure must become just another object to the design system, albeit one with very special capabilities, requirements, and size variability. We have designed, developed, and built a human figure modeling system which acts as an adjunct to a computer-aided design (CAD) system for human figure modeling, animation, and task performance assessment. Over the last six years, this effort has produced a program, called TEMPUS (Badler et al., 1985), and more recently a high performance workstation version, called JACK (Phillips, 1988), with greatly enhanced features. The principal functions of this system

- Provide a high performance graphics workstation for human figure manipulations.
- Provide a consistent, effective, powerful, and extensible graphics interface to human figure models and human factors tools.
- Create and select individual or statistical human figure models and body sizes.
- Provide interfaces to CAD object information for workplace descriptions.
- Position body segments by direct manipulation, workplace point reach goals, multiple goal positioning, constraint processing, and dynamics control.
- Offer a multiple window environment for easy study of body, camera, light, and scene interaction.
- Provide fast and high quality graphics output for both bodies and objects.

We are currently extending this system into a *task analysis* tool for assessing the actions of one or more individuals in a given environment. For example, the tasks to be performed are enumerated and decomposed into simple, primitive tasks such as reach, view, grasp, transport, etc., each of which has an instantiation as a sequence of movements. Given an environment (3D workplace), agent(s) (human or robotic figures to carry out tasks), and the task description, the system can animate the tasks. In addition, the system provides quantitative and qualitative information about the performance of the agents doing the tasks in that environment. By performance we mean

- Reach assessment. For an individual or a population, specify end effector(s) and fixed ends or restraints. Figure must reach a point in space or a workplace point. Show failure distance, reachable objects, and reachable space. Reaches should respect joint and environment limits and be specifiable for multiple reach goals and arbitrary restraints.
- View assessment. For an individual or a population, specify one or both eyes and the

viewed point. Show the corresponding view and show or list visible objects.

- Collision and interference detection. Adjacent body segment collisions are checked by joint limits. Non-adjacent segment collisions depend on the particular geometric representation of the body. A real-time display may be used for simple visual assessment without explicit computation.
- Strength or reaction force assessment. Determine the nominal or maximum force or torque achieved at a body part or end effector. Forces must be resisted, maintained, or reacted through restraints.
- Task load. Determine whether or not a task can be executed in some specific circumstances (e.g., time or strength constrained), whether two or more agents can work in parallel, whether fewer agents can get the jobs done, how much motor or psychomotor workload is imposed on each agent, and so on.

There are many components required to realize this task performance analysis system. The TEMPUS system and its evolving suite of programs is directly addressing large scale questions of effective, general purpose, flexible, and usable human factors analysis tools. The original TEMPUS system runs on a DEC VAX system under VMS. It is essentially a stable, frozen software system. The latest generation of software runs under Unix on a Silicon Graphics Iris 4D-GT (or lower capability) workstation. The computer graphics interface software JACK on the Iris provides the development structure for most of the new features and additions to the design, animation, and evaluation environment.

There are many sources of support for this project, each with its own emphasis and application:

- NASA Johnson Space Center and Lockheed Engineering and Management Services: primarily Space Shuttle and Space Station applications, with major interest in animation, strength models, zero-gravity simulation, and language-based task (command) processing.
- NASA Ames Research Center: the A^3I project to simulation all aspects of a helicopter mission is the application, with primary interest in the pilot model, task load, and task simulation from (separate) mission simulators.
- Army Research Office, the Human Engineering Laboratory at Aberdeen Proving Grounds: application to multi-operator vehicles, with a primary interest in evaluation of reach, strength, workload, and cooperative behavior.
- Pacific Northwest Laboratories, Battelle Memorial Institute: application to control a mobile robot mannequin used to test suit designs for permeability to chemical and biological agents, with a primary interest in animation control, safe path determination, collision avoidance, and motion feasibility.
- State of Pennsylvania Benjamin Franklin Partnership: technology development in Artificial Intelligence methods to aid human factors evaluation.
- National Science Foundation: representations and systems to assist in the interactive and automatic generation of natural, animated human motion.

In addition, this project greatly benefits from its home in a Computer Science Department because we feel that usable computational tools are essential for such a broad spectrum of human performance problems and applications. Rather than solve individual analysis problems, we can focus our efforts on longer-term systems design issues.

SYSTEM COMPONENTS

Figure 1 is a block diagram of the structure of the entire task analysis system. In general, boxes denote processes, ovals denote data storage or knowledge bases, and arrows denote data flow (structures or files) or access. Interaction pervades the whole structure. Below we give a summary

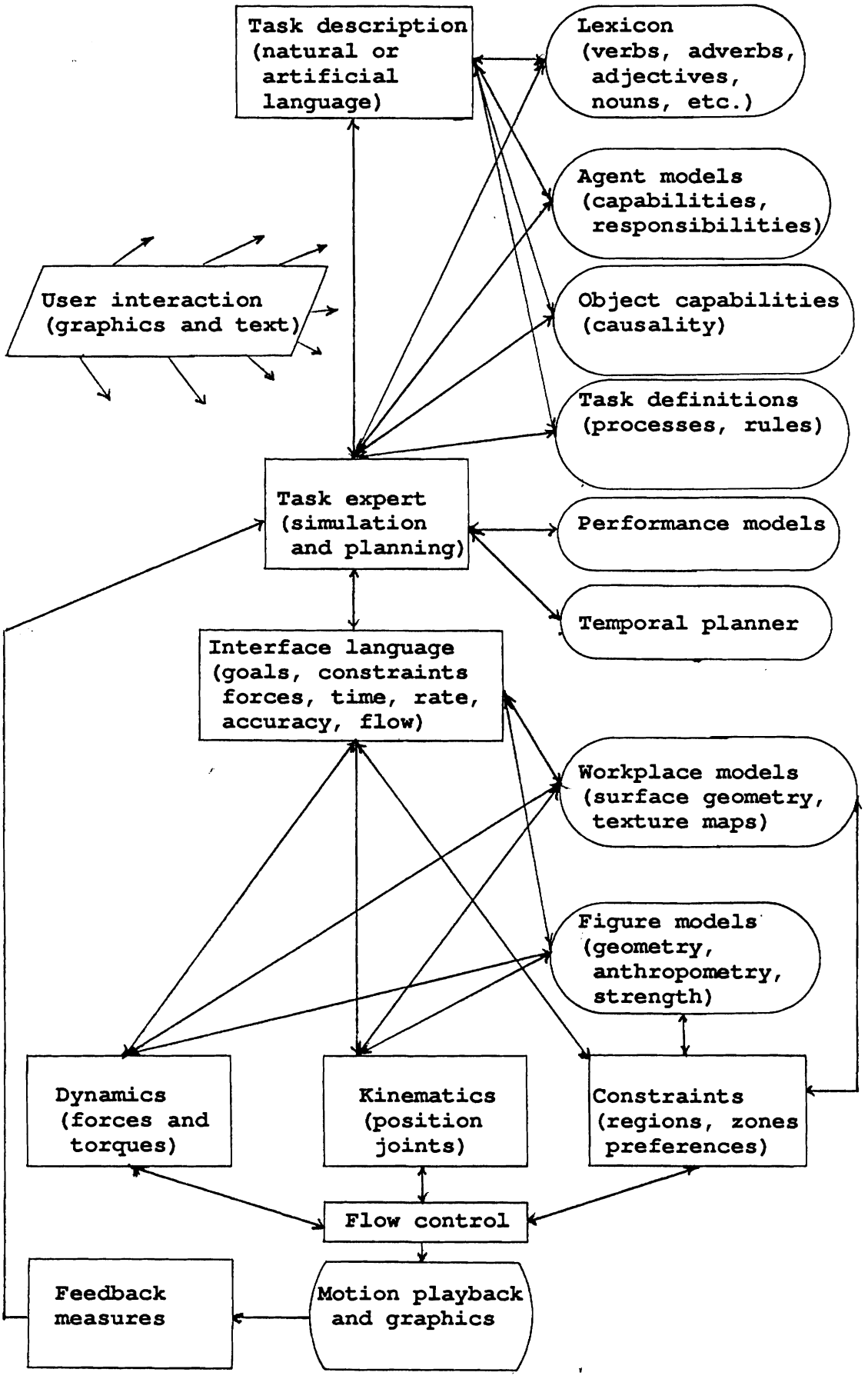

Figure 1. Block diagram of University of Pennsylvania human task animation system.

of the characteristics of each component.

Workplace Models

Workplace geometry is obtained from an existing internal or external CAD system. By separating object design from human figure modeling, independence from a specific CAD system (and its computer) is assured. Interfaces to CAD systems providing either boundary polygons or constructive solid geometry are available. Internally objects are stored as planar-faced boundary models. Additional surface attributes such as color, transmittance, specular coefficient, and texture may be specified. All workplace models may be displayed in either wire-frame or solid renderings.

The surface models are organized in a database structure, called PEABODY, which represents objects and their relationships in a network of figures, segments, sites, joints, and constraints. Any object may be formed by defining a figure which consists of segments. Segments contain polygon, curved surface, superquadric, etc. geometry models. Joints or constraints at sites (coordinate reference points) are used to connect segments. There is no restriction to hierarchical structures only; arbitrary connections are supported and encouraged giving the designer great freedom in creating the body and environment database. The representation of attached or closed-loop structures is easy: picking up an object or wearing a suit is accomplished by simply attaching the objects through a constraint, while closed loop structures or devices are created with the required joints or constraints. When needed during graphical display, a spanning tree is computed to define a traversal path. The tree is extended through joints before crossing constraints thereby insuring the integrity of the human figure models.

Texture maps are used for a novel function in workplace simulation. Although they can be used simply for visual richness and realism, a more important function is to save geometric storage space for panel-like arrangements of devices. By defining a texture map to be an image of an existing or proposed panel, the tedious and costly modeling of many or all of the contained objects is eliminated. Objects on the texture map are positioned and identified, then become reachable sites on some target polygon in the geometric workplace. During real-time motion display the reachable sites may be indicated by small squares on the polygon (Figure 2); on rendered images the texture map itself appears for accurate visual feedback. We have found that the use of texture maps can reduce the designed model complexity by hundreds of polygons without sacrificing any task animation capability. Moreover, panel texture maps are easily edited on the graphical screen, encouraging panel redesign for improved human performance.

Figure Models

Computer graphics figures with reasonable human-like appearance are provided in TEMPUS. There are at least four different levels of detail that can be used: BUBBLEpeople (Badler et al., 1979), polyhedral figures, and a stick figure. (The stick figure is rather useless.) The most detailed models are BUBBLEpeople: they look surprisingly lifelike and yet are neither expensive nor difficult to move and display. Constructed entirely from overlapping spheres specially rendered to appear smooth and visually continuous across sphere boundaries, the BUBBLEpeople are nonetheless an effective visualization aid in all but the most demanding visual image requirements. There are both detailed and low resolution versions of BUBBLEpeople.

The polyhedral figures come in at least two levels of detail. The lowest resolution polyhedral figure is shown in Figure 2. They are used for fast wireframe positioning, display, and motion playback. The polyhedral figures are used exclusively on the Iris workstation to gain display speed. The models may be customized with additional polygons or spheres to model suits, gear, life-support systems, helmets, etc. All figure models may be solidly rendered to aid visualization of their spatial configuration and workplace fit.

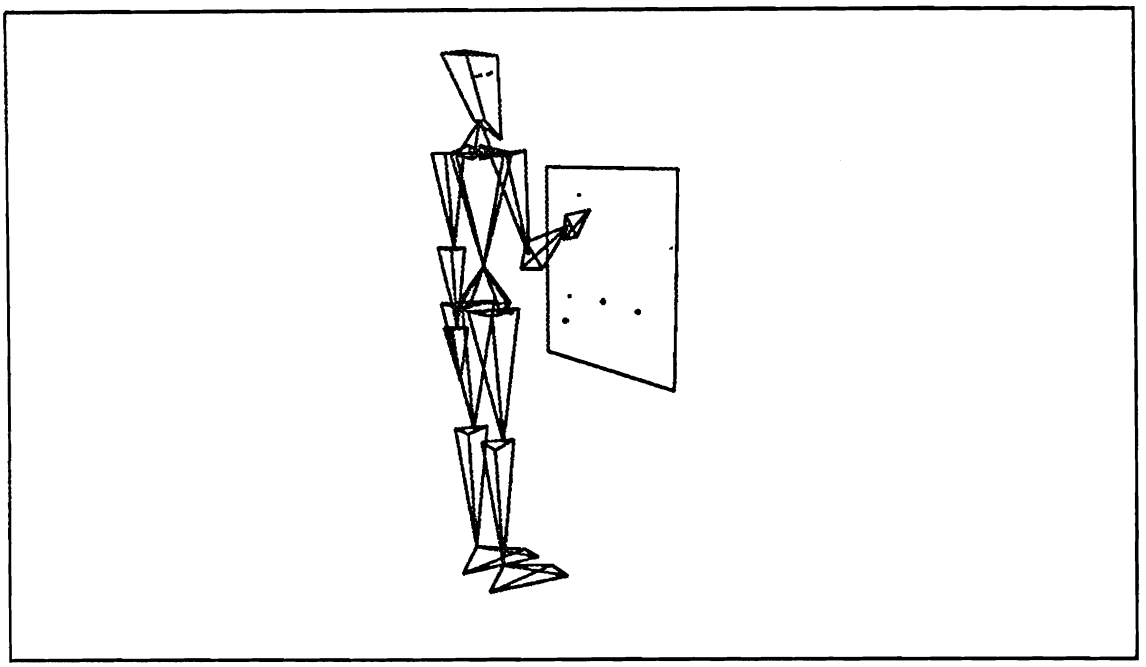

Figure 2. Simple polyhedral figure reaching a site on a texture mapped polygon in the workplace. Other sites on the polygon are indicated by the small dots. The hardcopy output does not show the texture map itself. Other objects and the ground plane have been removed for clarity.

Anthropometry

The models are sized from available anthropometric data. For example, we have been using statistical data from the NASA Manned Systems Integration Standards Handbook. Among the figure data fields are sex, segment lengths, girth values, joint spring and damper values (for dynamics), landmark points, and an indicator telling whether the body represents a real person or a statistical or otherwise specifically constructed generic body. The visualization geometry is not intimately associated with the figure characteristics in the database, but rather is sized when a particular individual is instantiated. Thus body feature locations (sites) are independent of the visualization. If more elaborate and detailed figure models are required, they may be defined in a normalized coordinate system especially designed for body segments and scaled by a set of anthropometric data-defined functions. All bodies may be selected, sized by explicit segment lengths or percentiles, and stored interactively. As many figures as needed may be manipulated concurrently.

Strength Model

A strength model is being constructed which will be used determine reasonable joint torques and forces in a given body position. Based on a degree-of-freedom decomposition of joint torques (whenever possible), this data is used to compute maximum forces at any end-effector. Strength data and forces may be used to determine reaction forces or active forces exertable through the body linkage. In the former, strength data is translated to spring and damper functions for dynamic simulation; in the latter, strength data may be used to assess estimates of task completion times. The strength model will be used in various places in the system; we will return to it later.

Motion Playback

Key postures created by TEMPUS or other animation systems may be interpolated by B-spline curves (Steketee and Badler, 1985). Object file information, key postures, and interpolation

parameters are processed to produce an animation. The resulting object, camera, and articulated figure motions are displayed in real-time wireframes on the Silicon Graphics Iris Workstation so that motions may be assessed and tasks validated. The playback software in JACK permits single frame viewing, speed control, camera view control, and single frame rendering. A fully featured graphics display system is included for realistic solid shaded renderings of each frame. This system can shade polygon environments with anti-aliasing, translucency, multiple light sources, and object surface attributes such as texture, glossiness and specular reflection. As part of the JACK interface, image parameters such as light positions, light concentration cones, and the camera position can be interactively set and viewed.

Position Control

An articulated figure is manipulated in several ways. In TEMPUS, positions can be specified as body joint orientations (angles) or by end effector (limb) goals. In either case, joint angles are subject to known joint limits (Korein, 1985). The joint limits are stored in a file and can be adjusted to different situations, such as suits or special capabilities. The limb reach permits positioning the hand tip, grip, or wrist at a point in space while the shoulder is fixed. The remaining degree of freedom permits the elbow to move in an arc while the reach point is held fixed. Similar criteria hold for the legs.

In the JACK interface, any figure segment can be manipulated in translation or rotation independently, including segments representing lights and cameras. The camera view may also be identified with a figure's eye position. There are a variety of user interface tools designed to make this positioning task as straightforward as possible, including on-screen segment picking, real-time feedback, and two-dimensional inputs transformed to three-dimensional rotations around selected axes. Whole figures may be positioned relative to any other object or figure surface, edge, or vertex.

The figure (or object) positioning may also be accomplished by less direct manipulation. Below we discuss some of the alternatives: kinematics, dynamics, constraints, flow, and higher-level task control.

Kinematics

While the TEMPUS reach positioning capabilities are an improvement over joint angle changes alone, single goals and fixed proximal joints are still too limited for general human capabilities. A human or robot figure model must also be kinematically-controlled so that goals and constraints may be used to position and orient the parts and end-effectors (Badler et al., 1987). We developed an algorithm that permits specification of a spatial goal for each body joint. The joint goals are satisfied by a recursive tree balancing algorithm which is iterated until there are essentially no further joint position changes. Goals are described as springs of variable tension connected from selected joints to points in space. The springs move the body joints in such a way as to attempt to minimize the spring energy by simple heuristics.

Though the body is a tree, this algorithm is able to easily handle closed loop situations such as two hands holding the same object. Multiple simultaneous goals are naturally accomodated: for example, a seat belt restraint while the figure is seated and reaching for different objects with each limb, a foot restraint while reaching with the whole body, or a free-floating body reaching with one hand while holding a fixed grip. Figure 3 shows two alternative reaches executed with a figure restrained by a lower torso goal simulating a lap belt. In (a) the figure is given the reach goal for the right hand. In (b) the reach is achieved; notice how the entire torso as well as the arm joints participate in the reach. In (c), the reach is attempted under an additinal constraining goal for the left shoulder (simulating a shoulder belt). The hand reaches toward the goal, but fails; the failure distance would be displayed to the user.

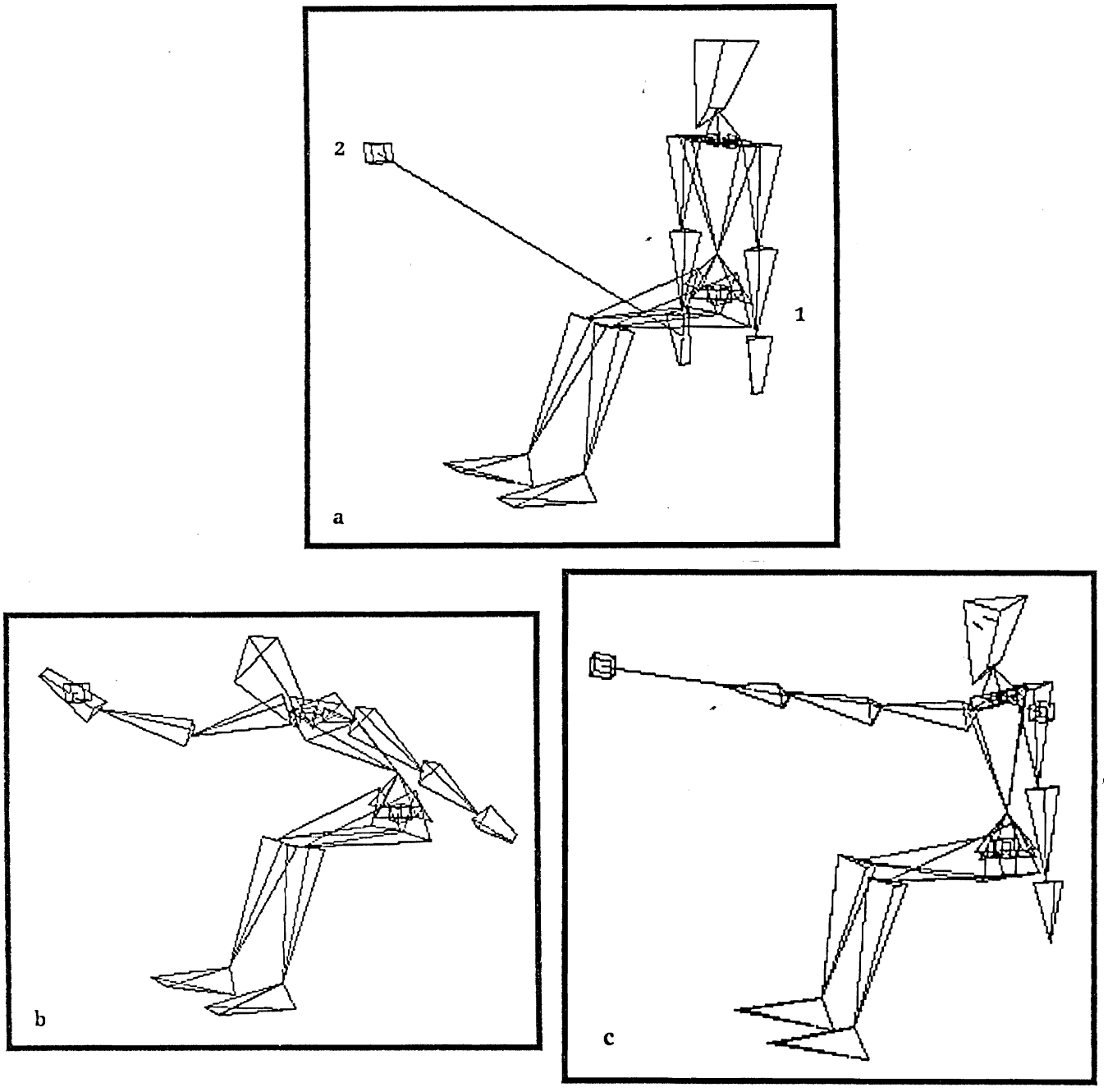

Figure 3. Simple polyhedral figure reaching a goal point in space while restrained by a lower torso goal simulating a lap belt. Other objects have been removed for clarity. In (a), the two goals are shown: (1) is the lower torso goal that tends to keep it in place; (2) is the desired reach position for the right hand. The relative weights of the two spring goals are 100 for (1) and 10 for (2). In (b), the reach is accomplished; all body segments from the right hand through the lower torso are involved in the reach. In (c), the reach is attempted with an additional constraining goal for the left shoulder joint. The reach fails. Notice that the shoulder joint has actually been displaced from its original position, demonstrating the interpretation of the goals as springs.

This feature is being extended to include joint angle limits during the positional goal achievement process. Orientation goals are also being added. The more general algorithm of Witkin *et al.* (Witkin et al., 1987) is being implemented for this and other applications (such as obstacle avoidance).

Dynamics

External or internal forces or torques may be specified through JACK and applied to an articulated figure to produce motion. Dynamic control is most useful for fast motions, for response to external forces (such as gravity), and for incorporating strength models. Our system incorporates a general mechanism simulation system called DYSPAM (Paul and Schaffa, 1985).

As in Wilhelms work (Wilhelms, 1986), we also expect to use kinematics and interpolation to create approximate motions, derive forces and torques, and then adjust the resulting forces and torques to modify the animation. Direct dynamic control (with the exception of restraining forces, environmental obstacles, and joint limits) appears to be much more difficult to specify (Armstrong et al, 1987). We differ though, in the interaction between kinematics and dynamics, preferring to run both in parallel and mix the results according to the requirements of the motion. This animation control method, called Flow, forms the basis of our new animation system TAKE_ONE (Badler and Dadamo, 1988). We expect that the Flow concept will provide a consistent and controllable mechanism for animating complex actions where individual movement styles may vary.

Task Expert

An expert system shell called HIRES (Fishwick, 1986, Fishwick, 1988) transforms task descriptions into kinematics, constraints, and dynamics for execution by the appropriate animation processors. HIRES is a production rule engine with a frame-like (Artificial Intelligence) knowledge base DC-RL. Multiple agents may be utilized. HIRES handles task simulation, agent interaction, and (eventually) motion planning. Its major strength is the general process representation which can be used to animate most any deterministic, stochastic, or rule-based process description. Under revision now, HIRES will be extended to provide more consistent rule syntax, incorporate a recent temporal planner (Kushnier et al., 1988), fully utilize the DC-RL knowledge base for rule storage and application, and provide a task priority, interrupt, and restart facility.

HIRES includes a facility to model the same process at different levels of abstraction. Thus the task does not always require simulation at the most detailed level, but rather at a level which is compatible with user goals. For example, detailed dynamics can be included in one level of a process model, but if that process is being executed "off-stage" then the work need not be actually performed as long as the future state of the system is known or predictable. This is a feature most advantageously exploited in conventional as well as computer animation where complex activities are frequently handled by inference rather than by explicit visualization (Thomas and Johnson, 1981).

Agent Models

Agent capabilities and responsibilities are modeled explicitly. This includes physical attributes such as handedness, strength, and handicaps, and behavioral preferences or characteristics, duties, areas of responsibility (in the workplace), role in a group, etc. Also, general properties of agents may be expressed here, such as the hands being used for most grips, the relationship between the size of the object gripped and the capacity of the gripper, the preferred (normal gravity) support on the feet, the inability to occupy space concurrently with another object, the visual observation of something requiring a gaze or head orientation, etc.

Agent models (other than their anthropometric, strength, and visualization geometry data) are stored in a frame-based knowledge base (DC-RL) accessible to HIRES. Many agent features (hands, view, etc.) are considered as "resources" which may be allocated and freed by HIRES. Conflicts between multiple tasks may therefore by resolved by resource constraints similar to those modeled in computer operating systems.

Task Definitions

Tasks are defined by rules or procedures which are decomposed into simpler acts the system can interpret as goals, constraints, affected objects, paths, directions, etc. Task definitions are built from process models (scripts, Petri nets, data flow diagrams, production rules, or discrete or continuous simulation models) (Fishwick, 1986). The expectation is that a suitable process model will make the specification of a task animation much simpler by capturing the relationships between all the participants (agents as well as objects) and executing the process in a simulation-type (but rule-based) environment.

An important aspect of task description and its simulation by HIRES is the interface language between HIRES and the animation processors. We view this as the "missing link" between Artificial Intelligence knowledge representation systems and the actual animation of the human figure. Additional evidence for this view is also offered by Wilhelms (Wilhelms, 1987) in describing path planning, collision avoidance, and stimulus-response control.

Our YAPS extension of HIRES to better task animation interfaces will include task interrupt control, temporal planning, and task time estimation based on the human strength model and Fitts' law. Task time specification is crucial to the viability and accuracy of a task simulation. Arbitrary time estimates will not do, primarily because the temporal and spatial context of a task is critical to the time duration needed for task completion. For example, a simple button push will be accomplished in rather different durations depending on how close to the button the designated finger is positioned by the previous command. It is unrealistic to expect every action to be accompanied by a departure from and return to some neutral posture.

Task completion times will be specified in one of three ways: by temporal specification, by performance rate, and by target accuracy. In the first case, the time specification (duration or end time) is given and the event can be scheduled to begin immediately and proceed at a rate commensurate with goal achievement at the desired time. In the second case, the performance rate (as a percentage, say) is used as a multiplier of the maximum strength performance of this agent in achieving the goal. The strength model provides as estimate of maximum torques which can be used to compute the duration of the task. The performance rate modifies this duration for the required simulation time. In the third case, the accuracy value is used in a Fitts' Law formula for the generic task type to compute an expected task duration.

Feedback

Critical to the interpretation of the simulation as a task animation is the provision for direct feedback from the figure and the environment models to inform and control the simulation. The information returned includes any desired position, velocity, acceleration, torque, force, or collision. Thus the simulation can take appropriate (rule-based) actions when a collision occurs, when a strength limit would be exceeded, etc. This ability to react to a changing (external) environment outside its high-level knowledge base is not normally associated with Artificial Intelligence systems, though the concept has been developed and is essential for robotics and sensory control applications.

Task Description

Task, action, or process descriptions are provided by programming languages, scripts, or commands in a subset of a natural or artificial language. Certain primitive actions are represented by semantics meaningful to the HIRES simulation, such as move, turn, grasp, look at, etc. More complex actions are expanded to request or determine necessary information such as object referents, to resolve ambiguities such as choosing the proper agent or instrument used by the agent, to supply a default sequence of subtasks, and to establish approximate temporal relationships and

timings.

Our first attempt at task description used a subset of natural language or an artifical language (syntactically stylized checklists) to describe tasks in a generic control panel setting (Badler and Gangel, 1986). This system, MVP, uses a parser and a knowledge base of agent and object capabilities to understand the task command and provide a first cut at the subtasks required to execute it. Our initial applications of this task input method focused on panel-type objects: switches, indicator lights, meters, valves, etc. (Gangel, 1985). Recently, the incorporation of more complex tasks and movable objects has been studied (Karlin, 1987). Both systems will produce assertions in the DC-RL representation system which are meant to be interpreted by HIRES.

This natural and artificial language input system is being extended to include additional control constructs with the ultimate intention of processing complete task descriptions with inherent contingencies, repetitions, and alternatives. There is significant human factors material in this form (for example, the NASA Flight Data File cue cards). The ability to use this command data directly to run purely computational human factors and performance data experiments is a realistic goal.

An alternative source of task descriptions is an (external) task simulation. For example, in the A^3I effort, a helicopter mission is simulated by a planner; the tasks required of the helicopter pilot are output in a conventionalized format and transferred to the pilot model in JACK. The tasks are presently a simplified list of reach and view tasks with geometric targets. The timing for each action is determined by the mission simulator's progress. Constraint-based positioning achieves the reach goals as expeditiously as possible in real-time on the Iris. One interesting aspect of this attempt at real-time graphical task simulation is a consequence of driving the graphical simulation too fast. If a task cannot be completed, it is interrupted to begin execution of the next task (since tasks arrive in real-time and in temporal order). The pilot's hands return to a neutral position between tasks only if there is time for that action to occur; otherwise the hands move as fast as the graphical simulation will allow from reach goal to reach goal. Since the tasks are also saved, the task sequence can be replayed after the mission simulation to allow all tasks to complete. At this point various measures of workload could be computed.

Knowledge Bases

Knowledge bases store information shared across system components, such as the geometry data, the anthropometric database, the agent models, the task descriptions, and object capabilities. Object capabilities are used to determine the meaningfulness of a task command and the results of the action on the workplace environment. Sample interaction with control panel objects and their interrelationships have been investigated. For example, turning a dial may change an indicator.

On the Silicon Graphics Iris, all databases are actually in Unix files. Dependence on any specific database system is thereby eliminated. In contrast, our attempts to standardize on a relational database in the TEMPUS VAX system were well intentioned but ultimately failed. In general, our systems are built on the premise that no additional software systems besides the standard language processors and Unix file systems are available. The JACK interface and the accompanying computer graphics is therefore portable to any Silicon Graphics Iris without additional cost or investment in third party software. Likewise, the higher-level functions (HIRES, MVP, and DC-RL) are all written in Commonlisp and run on a VAX, a Symbolics, or even the Silicon Graphics (being tested). We are not dependent on any third-party systems for the AI component. The knowledge base DC-RL, in particular, is quite powerful as knowledge-based systems go. In fact, DC-RL will even allow back-end interfaces through Commonlisp directly to any other existing database, provided that its data schemas and suitable conversion functions are written.

User Interaction

The user may interact with the task animation system at any level. It is expected that different tasks will require utilization of various parts (and maybe all) of the whole system. All interaction is through effective computer graphics interfaces or flexible language understanding processors. We have already reviewed the JACK interface for direct computer graphics manipulation on the Iris; likewise MVP and DC-RL exist for user expression of task commands and world knowledge.

Some programs do not fit so well into these two major interfaces. In particular, the selection of figures and their anthropometry is a separate textually interactive system, and the creation of texture maps uses a different graphical interface. The latter is used to define flat panels of objects as a two-dimensional image with certain named sites identified as panel-type objects (switches, etc.). The panel with its objects is developed interactively through a paint system with generic object icons, or simply read in as a digitized image from a photograph, drawing, or the real thing. Objects may be moved, deleted, or added in either case. Object characteristics are associated with the various image features. When satisfactory, the texture map is stored, the high-level device information is sent to DC-RL, and the geometry of the object locations (as sites) are inserted on a given polygon inside the PEABODY geometric database.

CONCLUSION

All of the system components in Figure 1 are functioning in some form. Though significant efforts remain to broaden the scope of some of the components and build task vocabulary, feasibility has been demonstrated. Moreover, any approach to human performance animation that fails to include all these processes can be shown to have significant weaknesses for certain animation, analysis, and assessment tasks.

There are several ongoing efforts to use our software for actual human performance visualization and assessment tasks. In general, the software is available on a research basis from the University of Pennsylvania Computer Graphics Research Laboratory. While not claiming its universal applicability to all human performance issues, it does offer a substantial, broad, and extensible framework for the investigation and solution of many real problems.

ACKNOWLEDGMENTS

This research is partially supported by Lockheed Engineering and Management Services, Pacific Northwest Laboratories B-U0072-A-N, the Pennsylvania Benjamin Franklin Partnership, NASA Grant NAG-2-4026, NSF CER Grant MCS-82-19196, NSF Grants IST-86-12984 and DMC-85-16114, and ARO Grant DAAG29-84-K-0061 including participation by the U.S. Army Human Engineering Laboratory. This work would not be possible without the assistance of the numerous participants in the Computer Graphics Research Laboratory at the University of Pennsylvania.

REFERENCES

Armstrong, William, Mark Green, and R. Lake. (June 1987). Near-real-time control of human figure models. *IEEE Computer Graphics and Applications, 7*(6), 52-61.

Badler, Norman I. (1986). *A representation for natural human movement* (Tech. Rep.). Philadelphia, PA: Dept. of Computer and Information Science, Univ. of Pennsylvania.

Badler, Norman I. (June 1987). Articulated figure animation. *IEEE Computer Graphics and*

Applications, 7(6), 10-11.

Badler, Norman I. and Diana Dadamo. (1988). *The Flow approach to animation control* (Tech. Rep.). Philadelphia, PA: Dept. of Computer and Information Science, Univ. of Pennsylvania. (submitted to The Visual Computer).

Badler, Norman I. and Jeffrey S. Gangel. (June 1986). Natural language input for human task description. *Proc. ROBEXS '86: The Second International Workshop on Robotics and Expert Systems.* Instrument Society of America.

Badler, Norman I., Joseph O'Rourke, and Hasida Toltzis. (Oct. 1979). A spherical representation of a human body for visualizing movement. *IEEE Proceedings, 67*(10), 1397-1403.

Badler, Norman I., Jonathan D. Korein, James U. Korein, Gerald Radack, and Lynne S. Brotman. (1985). Positioning and animating human figures in a task-oriented environment. *The Visual Computer: The International Journal of Computer Graphics, 1*(4), 212-220.

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

Dooley, Marianne. (Nov. 1982). Anthropometric modeling programs -- A survey. *IEEE Computer Graphics and Applications, 2*(9), 17-25.

Fishwick, Paul A. (1986). *Hierarchical Reasoning: Simulating Complex Processes over Multiple Levels of Abstraction.* Philadelphia, PA: Doctoral dissertation, Dept. of Computer and Information Science, Univ. of Pennsylvania.

Fishwick, Paul A. (Jan/Feb. 1988). The role of process abstraction in simulation. *IEEE Trans. Systems, Man, and Cybernetics, 18*(1), 18-39.

Gangel, Jeffrey S. (August 1985). *A motion verb interface to a task animation system.* Philadelphia, PA: Master's thesis, Dept. of Computer and Information Science, Univ. of Pennsylvania.

Karlin, Robin. (December 1987). *SEAFACT: A semantic analysis system for task animation of cooking operations.* Philadelphia, PA: Master's thesis, Dept. of Computer and Information Science, Univ. of Pennsylvania.

Korein, James U. (1985). *A Geometric Investigation of Reach.* Cambridge, MA: MIT Press.

Kushnier, Scott, Jugal Kalita, and Norman I. Badler. (1988). *Constraint-based temporal planning* (Tech. Rep.). Philadelphia, PA: Dept. of Computer and Information Science, Univ. of Pennsylvania. (submitted to AAAI-88 Conference).

Paul, Burton and Ronald Schaffa. (1985). DYSPAM User's Manual. Dept. of Mechanical Engineering and Applied Mechanics, Univ. of Pennsylvania.

Phillips, Cary and Norman I. Badler. (1988). *Jack: A toolkit for manipulating articulated figures* (Tech. Rep.). Philadelphia, PA: Dept. of Computer and Information Science, Univ. of Pennsylvania. (to appear, ACM/SIGGRAPH Symposium on User Interface Software, Banff, Canada).

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

Thomas, Frank and Ollie Johnston. (1981). *Disney Animation: The Illusion of Life.* New York: Abbeville Press.

Wilhelms, Jane. (1986). Virya - A motion editor for kinematic and dynamic animation. *Proceedings.* Vancouver: Graphics Interface '86.

Wilhelms, Jane. (April 1987). Toward automatic motion control. *IEEE Computer Graphics and Applications, 7*(4), 11-22.

Witkin, Andrew, Kurt Fleisher and Alan Barr. (1987). Energy constraints on parameterized models. *Computer Graphics, 21*(3), 225-232.