The Interactive Specification of Human Animation

G. Ridsdale, S. Hewitt and T. W. Calvert
Laboratory for Computer and Communications Research
Simon Fraser University, Burnaby, B.C. V5A 1S6

Abstract

This paper describes the Figure Animation Project in progress at Simon Fraser University. The project has two goals for the specification of figure animation: the first is to implement an interactive Figure Animation Test Bed for specifying movement at the *detail* level and the second is to develop a mechanism for describing figure animation at the *scene* level. These goals — and approaches to solving them— are discussed. In particular, the application of knowledge-based inference to the problem of scene description is examined.

Introduction

The long-term goal of this project is to develop scene-level motion descriptions for articulated, humanoid figures. In general, three-dimensional animation of the human body — or of any other vertebrate — is based on an underlying framework of articulated elements. For instance, a reasonable approximation of the human skeleton can be achieved with about 24 segments if the fingers and toes are ignored. Most of our efforts over the past few years have been directed towards developing interactive techniques for specifying detailed figure movements, since a system embodying such techniques is a necessary component for developing and testing scene descriptions. Such an interactive test bed must be capable of displaying the full range of interesting scene actions; this range includes both the actions of individual joints, and the movements of the body as a whole.

Above the detail level are the motivations of the characters and their interactions with the environment. This is animation at the *scene* level. After about ten years of continuous research in this area we are still not sure whether truly convincing animation of the full range of human movement is feasible, but there is no question that progress has been made in certain specialized areas of figure animation. What is needed now is the guidance of an intelligent supervisor to tie together these many pieces of the animation problem.

To coordinate the activities of a human animation system we require a supervisory program that has knowledge of the overall goals of the characters in the scene. In particular, such a program needs to deal with the issues of path planning and with the physical constraints on jointed figures. Ideally, it should only be necessary to specify the character's motivation in the scene and its initial position. The figure would then progress automatically and *characteristically* to its most likely destination from that starting point. In reality, the problem of determining an appropriate path from knowledge of the character's intentions is difficult, as is the problem of having the character navigate around both the fixed objects and the other characters in the scene. This latter problem has occupied robotics researchers for many years. The problem of sophisticated route planning is particularly difficult when the characters are jointed walking figures. Not only does the figure have to move about unencumbered, but the feet must also pick their way around and over any objects that may be found on the floor. While performing this, the figure must maintain its balance and a reasonable posture. This is a particular problem when shifting weight smoothly from one limb to another.

There are many physical constraints on what a character can and cannot do in a scene. For instance, a real figure's anatomy and physiology impose limitations on how it can move and interact. An intelligent supervisory program for human figure animation must understand these limitations; in this way, only physically realizable scenes will result.

Who Uses the System

The Figure Animation Test Bed is used mostly by choreographers who work in the disciplines of skating and dance. This test bed system has been designed to evaluate the various interactive techniques — buttons, menus, pick-and-drag, rendering speed traded off against image quality — that can be applied in a figure animation system. Since the people best qualified to judge the effectiveness of a system's user interface are themselves the future users, we select the techniques with which these choreographers feel most comfortable.

The major users of the scene-level animation system are expected to be film and theater directors. The intelligent supervisory program requires *expert knowledge* on which to base its decisions. This knowledge is developed through cooperation between the computer scientists who developed the system and the directors who possess the expert knowledge.

Project Background

Over the years, many techniques have been applied to the problem of specifying the complexity of human movement.

Rotoscoping

One basic approach is to capture real movement patterns from a live subject. This can be done by "rotoscoping" — digitizing by hand the joint co-ordinates of all body segments from at least two orthogonal views recorded on film or video. This approach, while accurate, is tedious. It is important in biomechanics research, and there is continuing interest in automating it, but the pattern recognition problems involved are difficult.

Live movement can also be captured in real time with special instrumentation. Goniometers provide a cumbersome but relatively inexpensive method [Calvert 80]. Expensive video scanning systems such as WATSMART and SELSPOT [Ginsberg 82], on the other hand, allow subjects to move freely in space; their actions are tracked by time-multiplexed light-emitting diodes attached to their joints. The movement patterns digitized with any of these methods can be normalized for speed and body size, and can be stored to create a library of fundamental movement patterns.

Notation

Another way movement patterns can be specified is with notation. While human movement can be described by a number of *dance* notation systems, such as as Labanotation [Smoliar 77] [Calvert 78] [Ryman 83], Eshkol-Wachman notation [Eshkol 79] and Benesh notation [Singh 83], none of these deal directly with the problem of describing human movement in an unambiguous way. This is because all of these systems are intended for use by trained dancers and, as a result, they normally leave out numerous details that these dance experts consider obvious. The knowledge these artists bring to bear on the interpretation of a Laban or Benesh score is an example of the sort of *expert* knowledge needed by any functioning movement interpretor, be it human or machine.

Our own experience with Labanotation has shown that it is a viable way to specify animation. Labanotation has the definite advantage that it relies on the animators' conceptualization of the movements required, and it certainly lends itself to the development of complicated scores. However, the basic commands are at too low a level, and users have trouble predicting the outcome of commands. Even with the addition of a macro capability, it is still tedious (some might compare it to programming in assembly language). Not even dancers find it easy to learn.

These systems are capable of describing a movement in arbitrary detail. But since they lack the grammatical structure needed for the construction of higher-level primitives from simpler components, this capability is not enough. Thus the important characteristic of *extensibility* is missing (in any really useful form) from existing movement-notation systems.

Interactive Positioning

A third approach (after rotoscoping and notation), is interactive positioning. This is the basis of the Figure Animation Test Bed, and involves the interactive specification of body positions in a 3-D graphics environment. The user is presented with a space-filling vector representation of the human body on the screen of a graphics workstation. The body can be viewed from any angle — in perspective — under the control of a mouse or equivalent device. The mouse selects body segments and orients each segment in three-dimensional space. The end result is directly equivalent to the output from notation, but the user has direct visual feedback and finds the adjustments to be natural and intuitive.

Several attempts have been made to produce integrated systems for animating human movement [Calvert 80] [Calvert 82] [Calvert 83], [Badler 82] [Badler 79a], [Zeltzer 82] [Nichol 83]. Most of these have not directly addressed the problem of specifying the movement involved in a high-level, extensible way. Instead, the movement is described at what may be termed an "assembly-language" level, where it is difficult to collect (or "abstract") detailed movements together into complex actions. Although a simple form of *macro-expansion* is available in the system developed at SFU [Calvert 78], this still does not provide enough power to develop a complex hierarchy of movement concepts.

More fruitful work has been done in the area of the interface between an animation system and its users. Foley and Van Damm [Foley 82], describe the fundamental elements of good interactive design in terms of the *conceptual*, *semantic*, *syntactic* and *lexical* design levels.

The *conceptual* design level describes the *user model* of the system; that is, the key concepts that the user must understand in order to make full use of the proposed system. The names and relationships of the *objects* in the system make up this level. In the context of our scene-level animation system, the conceptual level includes such things as characters, stage directions and props.

The *semantic* design level specifies the set of *functions* that the system is expected to perform. In our system, these functions include the ability to display a particular scene, the ability to learn new stage directions and character names, and so on.

At the *syntactic* design level, the rules that specify the acceptable sequences of input tokens (and the functions that they trigger), are set out. In the scene-level system, these values include the detailed protocol of menus, windows, valuators and text that will control the scene display and the management of the expert system.

Finally, the *lexical* level specifies the input tokens that the system recognizes. These include *text tokens* (such as "walk", "run" and "stage left"), *graphical tokens* (sketches and selected menu-items) and *gestural tokens* (movements of the user's body).

There is considerable interest in automating the development of a user interface, given its specification in some formal language. Such an automatic system is described by Olsen [Olsen 84]. Here a formal specification of a user interface can be used to generate a Pascal program that implements the functions of that interface. The input, in this case, is in the form of a *grammatical* description of the interface specification, which is entered by the system designer.

System Configuration

There are two prime requirements for an interactive environment in which the body positions are to be specified. The first is for realistic, three dimensional visualization of the spatial orientation of the figure; the second is for fast motion checking. To meet these requirements, we employ an IRIS 2400 Workstation as the heart of our hardware system configuration. A machine of this power, while expensive, is essential for smooth interactive positioning; a very large number of transformations is needed to represent two fully articulated figures. Hardware graphics power is also very important for fast and smooth motion checking. For these two needs, vector-based machines — such as those produced by IMI or Evans and Sutherland — could also have been used. However, in addition to fast line drawing and matrix computation capabilities, the IRIS contains a 32-bit frame buffer with both smooth shading and Z-buffer hidden surface elimination available in hardware. These capabilities are useful while rendering the final animation.

Using the Figure Animation Test Bed

When the user develops a piece of choreography on the computer she performs four main steps. These steps are:

- to design the sequence of phrases,
- to interactively generate the keyframes,
- to interpolate the intermediate frames and finally,
- to motion test the result.

When the choreographer is satisfied, the resulting animated sequence may be rendered onto film.

Sequences

In our terminology, a piece of choreography is referred to as a *sequence*. A sequence in turn is composed of any number of *phrases*. A particular sequence is defined by a list of phrase names that are saved in a text file along with other global information that pertains to the the sequence as a whole. This includes the number of phrases in the sequence and the number of frames in each phrase. Note that the same phrase may be re-used in any number of different sequences.

A phrase in turn is composed of a group of *keyframes*. Three keyframes are currently used to define each phrase since an approximating third-order spline is used to generate the intermediate positions. Each keyframe position is generated interactively on the screen of the IRIS 2400. At the start of this interactive process the user is presented with an image of two figures in a standard initial position. Each figure is represented by a vector image, where each body segment is modeled by a four-sided prism. It is important to use at least a crude space filling model like this, in order to give the user feedback on limb rotation and on the contact between body segments. Hidden lines are not removed. However, adjacent body segments are drawn with different colours to aid discrimination. The body parts themselves are positioned individually by a pick and drag procedure.

Interaction

To begin the interactive process, the user initially selects the foot which will act as the support point for the entire figure. Then the body position is built up by orienting each limb segment in turn, moving away from the support point. Using the mouse, a body segment is picked and its three angles of orientation are adjusted in turn; the user can change the angle of view at will. A digital readout of the angles is given at the bottom of the display. A mouse-based valuator generates these analog values for the figure's joint angle orientations, which in turn determine the orientation of the body parts presented in real time on the screen of the system console. This provides immediate feedback to the system user. After the approximate body position has been achieved, the user will iteratively refine the inter-segment angles until the desired result is obtained.

To specify a second frame, the user can either start with a standard position, as she did with the first frame, or the first frame can be copied and used as the starting point for the second. Similarly a third frame is specified. These three frames form a phrase which is given a name and stored. Multiple phrases are built up in turn, typically using the last frame of one phrase as the first of the next. Very little typing need be done by the choreographer while interacting with the system. Instead, software buttons guide the user through the sequence of steps that result in the final animation.

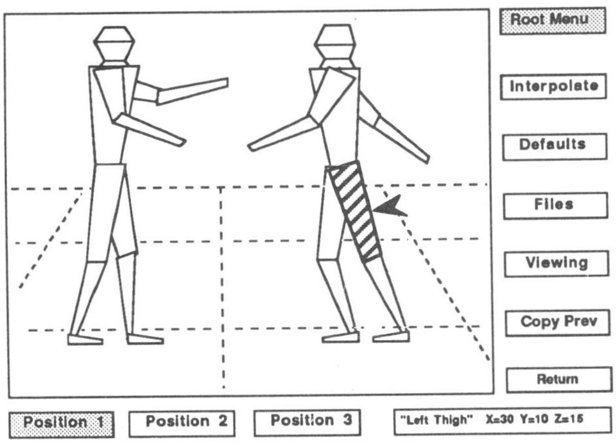

Figure 1 : Figure Animation Test Bed Screen

In this screen, the user has picked the left thigh of the right figure, and is about to change its orientation.

Currently, the user can specify the positions of two figures at a time. At this point the information available is equivalent to that which can be obtained by interpreting Labanotation commands or from instrumentation; thus data from different methods of movement specification can be combined.

Interpolation

Once the keyframes have been collected into phrases and the phrases assembled into a sequence, the "inbetweening" stage is performed. In this step a smooth series of frames connecting the keys is produced. A form of curve fitting based on third-order splines — one for each joint of the body — is used to determine the appropriate angle in space between the body parts articulated by that joint. Joint angle interpolation is very compute-intensive, requiring approximately 2000 floating point operations per frame. By formulating the cubic curve-fit in terms of matrix multiplication, the array-processing capabilities of the IRIS "Geometry Engines" can be used to solve the interpolation problem. Using these pipelined matrix processors, 1500 intermediate frames can be computed per minute.

Viewing

Having completed the interpolation, the final step is to view the resulting action. This involves selecting one of the display options that trade off rendering speed v.s. image quality. These rendering options include line drawings, filled polygons, and smoothly shaded solids.

Line Drawings — This results in the fastest rendering. By drawing each body part as a rectangular prism, frame rates of about 4/sec can be achieved. Faster frames rates (as high as 15/sec) can also be achieved using a simpler stick figure.

Filled Polygons — An intermediate level of rendering quality is obtained by using filled polygons to represent the body parts. This requires hidden surface elimination, increasing the rendering time to two seconds per frame.

Smooth Shading — The highest quality rendering requires the use of smooth shading and an accurate lighting model. However, flexible, jointed figures produce special problems for any solid modeling technique; joint coverage is particularly difficult.

We have experimented with using spheres as building blocks for constructing the figures [Badler 79b]. The figures are built up using a Constructive Solid Geometry (CSG) approach where the primitives are shaded coloured spheres of varying diameter. There are over 800 spheres in each body and each sphere is rendered with a polygon approximation which takes account of the lighting for each sphere. The shading is obtained with the Gouraud method and the hardware z-buffer in the IRIS provides hidden surface removal.

This resulting image contains over 400,000 smooth-shaded polygons per frame and requires 2-3 minutes to render on the IRIS 2400. Obviously, this is nuch too slow for previewing the motion of the figures, and so this technique is used only for frame by frame reproduction onto 16mm film.

Work In Progress

The facilities of the Figure Animation Test Bed system are being extended and the quality of figure rendering is being improved. Also, the design and programming of the scene-level animation system is under way.

Animation Test Bed

The test bed system has been used by a figure-skater and by a dancer as an experimental tool. This has already resulted in some significant segments of animation. Both users both have expressed a very definite preference for interactive specification over the use of Laban-style notation. As a result of their experience, two significant needs have been identified:

- The movement patterns in our animated films result from the smooth interpretation of keyframe body positions. One problem is that the result is *too* smooth to be credible as human movement. As a result, we are investigating methods of adding small oscillations to the interpolated data into the Test Bed system. Each new movement generates an oscillatory transient and a small "wobble" is present at all times. In this way the movements achieve a more natural quality.

- At present, the path traced out by a figure results from accumulating the individual actions (stepping, jumping, gliding) of the character. This makes higher-level path planning difficult. A higher-level route planning facility is being developed as part of the scene-level animation system.

Director's Apprentice

There is a long term interest in developing systems that can interpret very high level descriptions of animated *scenes*. A good real-world model of such a description is the *film script*. The knowledge needed to make sense of such a description requires the use of a *knowledge base* composed of *rules*. These rules will come from the knowledge and experience of an expert director. We have named this project *The Director's Apprentice*, as it will "learn its craft" by studying the rules and practises of a human director. This learning process will take place as it aids him in the task of developing an interactive story-board.

The knowledge base of directing principles — used to interpret the script — will contain a collection of "if-then-else" rules. These rules map scene attributes — such as character motivations, script text and classic directing rules — into scene action. Of course, this mapping is neither unique nor well-defined, and it will result in numerous conflicting interpretations for each combination of attributes. A rule-interpreting inference engine must then arbitrate these conflicting conclusions, and decide on some reasonable resulting action for the scene. There are many categories of rules and concepts that such a system will need to address. These concepts include the inter-character feelings that dominate motivation, and the appropriate shifting of audience focus from one character to the next as the plot progresses. Knowledge of directing terminology — stage right, up stage, down stage — and standard set compositions will be needed to support the script interpretation.

Since considerable knowledge of the scene constraints is needed, the process of *abstracting* the detailed description of movement is not a trivial matter. For example, the phrase *"John walks across the room and stops at the other side of the desk"* may be used to describe many different scenes; the exact scene would depend on the arrangement of the furniture, on the other people in the room, on John's starting position and on his particular gait. To achieve the level of descriptive power needed for an effective directing language, an *expert* or *rule-based* system is being developed to provide the knowledge of scenes, physiology, habits, and so on, that are needed to abstract. the directing concepts.

Expert systems in AI have been developed largely to solve problems involving the deduction of answers from a set of facts stored in a *knowledge base*. Such a knowledge base contains the accumulated knowledge of one or more experts in that particular field. But how does this differ from a conventional data-base management system? Perhaps, the most important capabilities that distinguish an expert system are:

1. **Inference:** the ability to form a long, complex conclusion using the information present in the knowledge base [vanMelle 81]. This knowledge base contains both *relations* — as in a conventional relational data base — and the *rules of inference* that permit the system to infer new relations from those that were initially given.

2 **Self-knowledge:** the ability of the system to understand and explain interactively the structure of its own data base and its own inferential mechanism [Davis 82] [Davis 80].

3. **Flexibility:** the ability of the knowledge base to grow and adapt to new knowledge as a result of correcting comments make by an expert consultant [Winston 81].

Expert-system projects have been developed for many problem domains, but only a few of these were ever taken to the point of being really useful [Nau 82]. Of these, probably the most successful implementation was the MYCIN [Shortcliffe 76] [Davis 82] project, a question-answering system intended for the problem of medical diagnosis. MYCIN was based on the use of *production-system* methodology, where the knowledge base consists of *productions* (or *rules*) of the form

"*if* A_1 *is true,*

and if A_2 *is true,*

and if A_n *is true,*

then C_i *is true.*"

The applicability of production systems to the accumulation of expert knowledge has been discussed by Langely [Langely 83]. This work points out that the inherently *modular* nature of information that has been described in terms of productions makes the the task of incorporating new information into the structure much easier. The considerable success of the MYCIN project in answering complex diagnostic questions — measured against the performance of real "expert" physicians — led to the development of EMYCIN [Davis 82]. This consists of the pure "expert system" of MYCIN, but stripped of the medical-diagnosis data base. It led to further successful tests of the production-system mechanism in other subject domains (such as civil engineering and geology [vanMelle 81]).

The success of production-based expert systems largely derives from the ability of these systems to *encapsulate* specific facts in the knowledge base in the form of individual productions [Davis 82], [Langely 83]. The advantages of this are that *local*, specific changes can be made to the knowledge base, (adding, modifying or deleting individual rules) and the changes' effect on the other rules in the system can generally be predicted in a straight-forward manner. In fact, the same mechanism can also be applied to the strategies used by the reasoning system itself, as described in [Davis 80]. This opens the possibility of having the search-strategy mechanism *itself* defined in terms of productions — which can be added, modified, and deleted easily.

The production system in MYCIN was augmented by its ability to assign a *certainty factor* to the conclusion of a rule. These "confidence measures" range from 100% ("is certain to") down to -100% ("is certain not to"). This involves having the rule interpreter combine the certainty factors of the subordinate rules to form the certainty of the conclusion. The use of continuous-valued (or *fuzzy*) logic in the inferential mechanism derives from the observation that few conclusions in any real domain can be made with absolute certainty [Zadeh 83]. Statements such as "**If** *the character is John* **then** *there is an* 80% *chance that he will walk quickly across the room,* **and** *there is a* 20% *chance that he will walk slowly across the room,*" may be important when acquiring empirical judgements from human directors.

The function of an expert system is to answer questions using expert knowledge and reasoning. The "questions" that the Director's Apprentice will try to answer will arise during the interpretation of the script. For example the sentence "*John walks quickly across the room, and stands behind the desk*" will generate the following questions for the expert system to answer:

1. Is John presently in the room?

2. Are there any obstacles between John's present position and the desired final position?

3. If so, what is the best path for him to follow around, over, or under those obstacles? What sort of motion, if any, is meant by "walks quickly", or by "stands still"?

Issues of route planning, relating to the solution of problems (2) and (3), have been discussed by Lozano-Perez [Lozano-Perez 80]. More difficult questions relating to the *meaning* of the film, such as "Is it *in character* for John to kick over a chair on his way to the desk?" require the presence in the database of detailed information about the psychological and emotional aspects of the action. These considerations have been discussed by Fleischer [Fleischer 84].

Many recent expert systems have been concerned with the problem of using expert knowledge to make sense of sentences expressed in a natural language. Since we believe that free-form natural language is an inappropriate interface for an interactive graphics system, the linguistic issues that these systems address (pronoun references, multiple-clause sentences, and others) are not directly relevant to this research. Instead of an unconstrained natural language interface, we are are implementing a *system-directed* conversation, implemented by menu-picking. The advantages of this approach have been described by Rich [Rich 84].

Interacting with the Director's Apprentice

Interaction with the Director's Apprentice involves two steps. The first step is to design a set of rules that capture the facts and relations inherent in the key directing concepts — this forms the knowledge base. The second step is to query the knowledge base using the control panel of the Director's Apprentice inference engine.

Most stage action in a play consists of explicit commands to the actors, such as "exit stage left", "approach the upstage character" and so on. This is termed *inherent* movement. However, stage actions may often be *inferred* from a script even if that script contains only dialogue. This is termed *imposed* movement. Many excellent texts (for example Allensworth [Allensworth 82]) have been written for theater students, detailing how certain types of action may be generated from a knowledge of the flow of dialogue from one speaker to the next.

The initial set of rules that have been tried in the knowledge base of the Director's Apprentice deal with the theatrical concept of *focus*. Focus concerns the use of subtle character action to shift audience attention from one character to the next, generally one step ahead of the next change of speaker. This is done so that the audience will have time to settle its "focus" on a character *before* he begin to speak; otherwise, his first few words or gestures may be lost to those of the audience who are watching someone else. The following are some ways that may be employed to achieve shift of focus, based on the known flow of dialogue in a script.

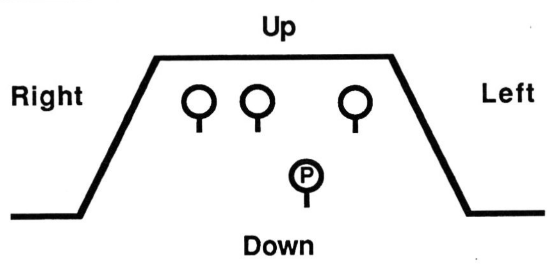

Figure 2: Focus by Position
Here, the principal actor (the next to speak) has been given focus by having the other actors stand further upstage.

In *focus by position*, the principal actor — that is, the actor who is just about to begin speaking — receives focus by walking downstage, or by having the other actors walk upstage. This puts him in an attention-grabbing downstage position relative to the others. Encoded as a knowledge-base, this action may include such rules as:

IF next-to-speak is $actor_i$,

AND $actor_j$ is-downstage-of $actor_i$,

THEN j moves-upstage-of i.

Many other rules, defining the concepts "is-downstage-of" and "moves-upstage-of" would also be required.

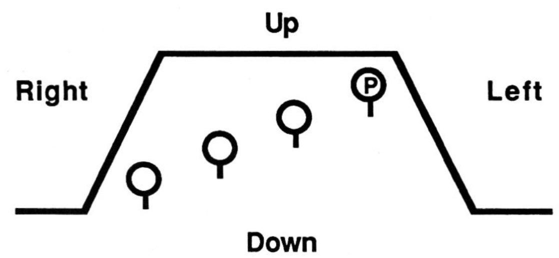

Figure 3: Actual Line Focus
Here, the principal actor (the next to speak) has been given focus by having the other actors form a line which points toward him (After Allensworth [82])

In *actual line focus*, the principal actor receives focus by having the other actors align themselves in a virtual "arrow" that points at him. To handle actual line focus, the knowledge base would need to contain rules such as:

IF next-to-speak is $actor_i$,

AND $actor_j$ is-not-aligned-to $actor_i$,

AND $actor_j$ is-closest-to $actor_i$,

THEN align-actor j,i,

AND next-to-be-aligned-is j

IF next-to-be-aligned is $actor_k$,

AND $actor_l$ is-closest-to $actor_k$,

THEN align-actor l,k,

AND next-to-be-aligned-is l

and so on.

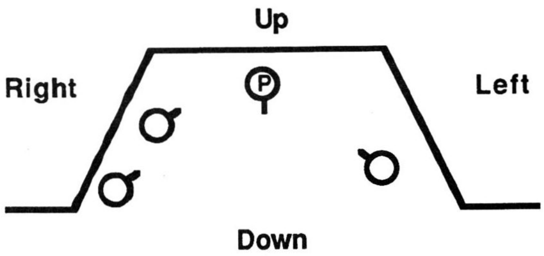

Figure 4: Visual Line Focus

Here, the principal actor (the next to speak) has been given focus by having the other actors turn to face him. (After Allensworth [82])

In *visual line focus*, the principal actor receives focus by having the other actors turn to face him. To handle this focusing rule, the knowledge base would need to contain rules such as:

IF next-to-speak is actor$_i$,

AND actor$_j$ is-not-facing actor$_i$,

AND actor$_j$ is-closest-to actor$_i$,

THEN turn-to-face-actor j,i,

AND next-to-face-is j

IF next-to-face is actor$_k$,

AND next-to-speak is actor$_i$,

AND actor$_l$ is-not-facing actor$_i$,

AND actor$_l$ is-closest-to actor$_k$,

THEN turn-to-face-actor l,i,

AND next-to-face-is l

At present, an ordinary text editor is used to enter these rules into the knowledge base. A structured rule editor is planned.

The inference engine for the Director's Apprentice runs on a SUN Workstation, with a high-resolution (1150 X 890) bit-mapped screen. The various functions of the rule interpretor and query system are invoked by menu picks on a series of adjustable panels (windows) that pop up under control of the inference mechanism. This way, it is hoped that directors using the system will adapt quickly to the interactive dialogue.

Improved Rendering

The current rendering technique, based on a body built up from spheres, leaves much to be desired. The use of a skin approximated with a polygon mesh promises both the improved application of modern lighting models and smoother surfaces. However, when human figures are rendered by existing polygon modelers, the results are generally stilted and cartoon like. Closeups of bending joints are particularly difficult to render smoothly. A robust method to move the control points of a skin derived by splines is being developed. The goal is to achieve a skin which stretches naturally as the body segments move relative to each other.

Conclusions

Many areas of research need to be explored further, at both the detail level and scene level. At the detail level, more study needs to be done on the best ways to interactively specify figure movement. The present system is continually being revised in response to user comments and suggestions.

At the scene level, a great many more rules will be needed to effectively implement change of focus. Other forms of imposed action are being studied as well. In order to put the knowledge base on a firm theoretical foundation, we are also looking into developing a formal semantic model (as described in DelGrande [Delgrande 86]) of directing concepts.

Acknowledgements

We wish to thank Catherine Lee, our principal choreographer, for her comments in developing the Animation Test Bed. This research was supported, in part, by grants from NSERC and from the B.C. Science Council.

References

[Allensworth 82] Carl Allensworth.
The Complete Play Production Handbook(Rev. Ed.).
Harper and Row, Inc., 1982.

[Badler 79a] N.I. Badler and S.W. Smoliar.
Digital representation of human movement.
Computing Surveys 11:19- 38, March, 1979.

[Badler 79b] N.I. Badler, J. O'Rourke and H. Toltzis.
A spherical representation of a human body for visualizing movement.
Proc. IEEE 67:1397- 1402, 1979.

[Badler 82] N.I. Badler.
Modelling the Human Body for Animation.
IEEE Computer Graphics and Applications 2, November, 1982.
Special Issue.

[Calvert 78] T.W. Calvert and J. Chapman.
Notation of movement with computer assistance.
In *Proceedings Annual Conference*, pages 731- 736. ACM, 1978.

[Calvert 80] T.W. Calvert, J. Chapman and A. Patla.
The integration of subjective and objective data in animation of human movement.
Computer Graphics 14:198- 203, 1980.

[Calvert 82] T.W. Calvert, J. Chapman and A. Patla.
Aspects of the Kinematic Simulation of Human Movement.
IEEE Computer Graphics and Applications 2:41- 50, November, 1982.

[Calvert 83] T.W. Calvert.
Computer assisted filmmaking: A review.
In *Proceedings of Graphics Interface 83 Conference.* Graphics Interface, Edmonton, May 1983.

[Davis 80] Randall Davis.
Meta-Rules: Reasoning About Control.
Technical Report AI Memo No.576, MIT AI Lab, 1980.

[Davis 82] Randall Davis and Douglas Lenat.
Knowledge-Based Systems in Artificial Intelligence.
McGraw-Hill, 1982.

[Delgrande 86] James P. Delgrande and John Mylopoulos.
Knowledge Representation: Features of Knowledge.
Fundamentals in Man-Machine Communication:Speech, Vision and Natural Language.
Cambridge University Press, 1986.

[Eshkol 79] Noa Eshkol.
Movement Notations.
Movement Notation Society, Tel Aviv University, Tel Aviv, Isreal, 1979.

[Fleischer 84] Kurt Fleischer, Mark Vickers, Ann R. Marion, James R. Davis.
Towards Expressive Animation for Interactive Characters.
In *Proceedings of Graphics Interface 84.* Graphics Interface, Ottawa, Ontario, May 1984.

[Foley 82] James D. Foley and Andries Van Dam.
Fundamentals of Interactive Computer Graphics.
Addison-Wesley, 1982.

[Ginsberg 82] Carol M. Ginsberg and Delle Maxwell.
Graphical Marionette: A Modern-Day Pinocchio.
Technical Report, Architecture Machine Group, MIT, 1982.

[Langely 83] Pat Langley.
Representational Issues in Learning Systems.
IEEE Computer , October, 1983.

[Lozano-Perez 80] Tomas Lozano-Perez.
Automatic Planning of Manipulator Transfer Movements.
Technical Report AI Memo No.606, MIT AI Lab, 1980.

[Nau 82] Dana S. Nau.
Expert Computer Systems: A Tutorial.
Technical Report TR-1201, Computer Science Dept., University of Maryland, College Park, Maryland 20742, August, 1982.

[Nichol 83] C.J. Nichol.
Animation Systems for Rigid Bodies.
Technical Report, Computing Science Department, Glasgow University, Glasgow Scotland, 1983.

[Olsen 84] Dan R. Olsen and Elizabeth P. Dempsey. SYNGRAPH: A Graphical User Interface Generator. In *SIGGRAPH-83*. ACM, Detroit, Michigan, July, 1984.

[Rich 84] Elaine Rich. Natural-Language Interfaces. *IEEE Computer* , September, 1984.

[Ryman 83] R. Ryman, A. Patla and T. Calvert. Use of Labanotation for clinical analysis of movement. In *Conference*. International Congress Kinetography Laban, August, 1983.

[Shortcliffe 76] Edward Shortcliffe. *Computer-Based Medical Consultations: MYCIN*. American Elsevier, New York, 1976.

[Singh 83] A. Singh. A computerized editor for Benesh movement notation. Master's thesis, University of Waterloo, 1983.

[Smoliar 77] S.W. Smoliar and L. Weber. Using the computer for a semantic representation of Labanotation. *Computing and the Humanities*. University of Waterloo Press, Waterloo, Ontario, 1977, pages 253-261.

[vanMelle 81] William J. van Melle. *System aids in constructing consultation programs*. University Microfilms International, 1981.

[Winston 81] Patrick H. Winston. *Learning New Principles from Precedents and Exercises: The Details*. Technical Report AI Memo No.632, MIT AI Lab, 1981.

[Zadeh 83] Lotfi A. Zadeh. Commonsense Knowledge Representation Based on Fuzzy Logic. *IEEE Computer* , October, 1983.

[Zeltzer 82] D. Zeltzer. Motor control techniques for figure animation. *IEEE Computer Graphics and Applications* 2:53-59, November, 1982.