Issues in the Automated Generation of Animated Presentations

Peter Karp
Steven Feiner
Department of Computer Science
Columbia University
New York, NY 10027

Abstract

Much research in computer animation has concentrated on motion planning, ranging from high-level script design to low-level specification of how each character moves. Generating an animated presentation, however, also requires decisions about camera planning, the selection of what material should be shown and how it should be ordered, the maintenance of visual continuity, and viewport selection. If the presentation is to communicate information coherently, these decisions must be based in part on knowledge about the content domain and about filmmaking techniques.

In this paper, we describe some of the principal techniques for generating good animated presentations, and point out some of the issues in developing systems that could apply these techniques automatically. Some of these issues are illustrated with examples made using ESPLANADE (Expert System for PLANning Animation Design and Editing), a rule-based testbed that we are developing for exploring the automated generation of animated presentations.

Résumé

La plupart des efforts de recherche en animation assistée par ordinateur ont porté sur le problème de planification de mouvement, depuis, au plus haut niveau, la conception du script jusqu'à une spécification détaillée de comment chaque personnage est animé. Pour génèrer une présentation animée, cependant, il faut aussi décider du positionement de la caméra, choisir quelle information doit être montrée, dans quel ordre, maintenir la continuité visuelle et choisir le viewport. La présentation ne pourra communiquer l'information de façon cohérente que si ces décisions sont basées en partie sur une connaissance du domaine et des techniques cinématographiques.

Dans cet article, nous présentons les principales techniques de génération d'animation, et soulignons certains problèmes qui se posent lors du dévelopement de systèmes utilisant ces techniques automatiquement. Nous illustrons ces problèmes sur des exemples tirés du système ESPLANADE (Systeme Expert de PLANification d'Animation Conception et Edition), un système à base de règles que nous dévelopons pour étudier la génération automatique d'animations.

Keywords: Animation, Knowledge-based graphics, Motion picture editing

1 Introduction

Animated presentations can be particularly effective in communicating information, especially when it involves (or can be represented metaphorically as) complex spatial interactions. Designing effective animated presentations is currently a time-consuming task that requires much skilled human involvement. Therefore, automating the design of these presentations would decrease the cost of producing them. In addition, automation could ultimately make it possible to generate presentations on the fly that are customized for a particular viewer and situation, adaptively presenting information whose content cannot be fully anticipated. Under these conditions it would not be possible to have a human animator designing the presentation, and a user of the system, unfamiliar with effective presentation techniques, would be unable to create a better presentation then the one designed automatically.

The process of generating an animated presentation can be divided into two major tasks: planning the objects that reside in the world being animated and the actions in which they participate, and planning how these objects and actions will be presented. The first task, that of planning objects and actions, can involve activities ranging from high-level scripting to low-level object modeling and determination of the frame-to-frame motion of interacting objects in a Newtonian world. The results of this task may be thought of as defining an animated virtual world. It is the second task, however, that of planning how the objects and actions will be presented to a viewer, that distinguishes an animated *presentation* from an animated virtual world. Presentation decisions include determining what information to present and when to present it, and defining the virtual cameras through which the world is viewed, by selecting viewing specifications, viewports, and transitions between cameras within viewports. In some situations, it makes sense to allow the viewer to roam the world freely, determining on their own what it is they will see and when and how they will see it; in the extreme, all the presentation decisions are made by the viewer. There are many cases, however, in which a presentation is intended to fulfill a set of specific goals, such as communicating particular information to a viewer. In these cases, the decisions made in designing how the information is to be presented will determine the presentation's effectiveness.

We believe that presentation decisions for animation should rely on knowledge of filmmaking techniques that have been developed to design visually coherent presentations. In this paper we discuss previous work on automating the generation of animation, introduce some of the relevant techniques that are used in filmmaking, and discuss some of the research issues that we have identified in designing and implementing a testbed system for generating animated presentations automatically.

2 Previous Work

A number of researchers are doing work in motion planning with the goal of automating the generation of animation, representative of which are [Zel83, Rey87, Wil87, EBJ89, BC89]. In contrast, relatively little research has been directed toward planning the presentation component of animations. One of the earliest examples is Kahn's pioneering work on ANI [Kah79], which generates 2D symbolic animations from a story description. ANI determines the speed at which characters move and how they are positioned, based on an input description of the dramatic action in the story. For example, a character will be positioned between two others if the story specifies that this character helps to protect one of these two characters from the other. A later system by Ridsdale [Rid87] determines motion paths for characters based on the characters' feelings, the actions they are to perform, and the geometry of the environment.

Neiman's GAK [Nei82] produces 2D animations that explain how to use a CADCAM system. GAK does simple motion planning to generate explicit examples of general actions that are specified by an associated knowledge-based help system. The presentation, however, is actually a side effect of these actions—all motion occurs in a single, visible 2D viewport.

Rubin [Rub89] describes a system that builds a movie of specified duration automatically from a set of prerecorded videotaped shots. The shots are generated conventionally by human camera operators and their image contents are carefully logged by hand. Rubin's automated editing system assembles a selection of shots, applying continuity rules to form a coherent visual narrative from a subset of the shot database. The system is restricted to static camera shots, does not make use of camera position information, provides cuts as the only form of transition between shots, and provides no facility for presenting the output of multiple cameras in the same frame.

In the work described here, we are concerned with the problems of integrating camera planning and editing, constrained by the requirements of visual continuity. In doing so, we also rely in part on the results of our previous research in automating the design of static images, including the design of sequences of pictures of 3D actions [Fei85], and the generation of composite pictures that include multiple viewports [SF89].

3 Background

In this section we provide a brief overview of some of the basic techniques used in shooting and editing films. More detail can be found in [RM68, Zet73, Che74, Bur81]. Reisz and Millar [RM68] give a good introduction to a variety of different editing styles for use in different motion pictures. Zettl [Zet73] offers a well structured presentation of film and video aesthetics.

3.1 Frames, Shots, Scenes, Sequences

The fundamental visual element of a film is a *frame*. Each frame is a graphic representation of the system state at some moment of time (1/24 of a second for film, 1/30 of a second for NTSC video). The frame rate is sufficiently high to give the illusion of continuous motion or change. A stream of frames, recorded from the moment the camera starts to when it stops, is referred to as a *shot*. A shot is the fundamental unit for development of the cinematic narrative. A set of shots that represents the same event and that uses different camera specifications and fields of view is called a *scene*. Scenes are joined together to form a *sequence*, which may be thought of as an act in a play or a chapter of a book.

Research by cognitive psychologists on the perception of motion pictures shows how a viewer can integrate the successive shots that are the basis for all motion picture presentations [HB78]. The viewer's attention can be directed to specific events through the use of shots made with different viewpoints and fields of view. By showing successive shots, a presentation makes it possible for the viewer to develop a 3D mental model of the subject environment that has been displayed on the 2D screen. Establishing this mental model is an important step towards building an understanding of the material being depicted.

Maintaining the visual interest of the viewer is another important reason that different camera positions are used. The eye scans the image, hunting for new information to add to the mental model. *Visual momentum* [HB78] is the term given to the active pursuit of visual content. It is proportional to the *cutting rate* (the number of shot changes per second) and the information content. Shortly after a shot is shown, the eye has completed its scan and momentum is reduced; therefore, shots are rarely longer then several seconds. A shot of a complex environment can be shown for a longer time, however, as can an image on a large screen. Television news studios will typically use three or more cameras to provide a greater variety of camera views, with shots of longer duration.

3.2 Transitions

Shots are joined together in time by a *transition.* The simplest transition, called a *cut*, is accomplished by seamlessly following one shot by another. A cut joins two distinct shots together to form a single visual unit. Other transitions may be used to serve specific purposes in different parts of a film. A *fadein* usually starts with a blackened screen, from which the image fades in, becoming brighter over a period of two or three seconds. A fadein is often used to start a sequence. A *fadeout* is the reverse process, and is often used to end a sequence. A *dissolve* composites the fadeout of one shot with the fadein of the next shot. The two shots are usually part of two different sequences. A dissolve can also be used to indicate that redundant or irrelevant information has been removed and that a large change in time or location has occurred between two scenes. A *wipe* is a dramatic transition that emphasizes that a transition is in progress, in contrast to the cut, which conceals the transition. There are many different kinds of wipes, but each usually has the effect of pushing the old shot off the screen with a new viewport that expands from the sides or center.

3.3 Triple-Take Filming

Action is recorded first, and edited later. Triple-take filming is an important technique, used when action is being recorded, that makes it possible to develop a coherent 3D screen space and visual presentation when the presentation is edited. A scene's action is recorded using three camera specifications, each associated with a different field of view. A *long shot* is used to record and establish the positions or interrelationships of objects in the shot. An *insert* or *close-up* is used to direct the viewer's attention to a detail in the long shot. It is sometimes difficult for the viewer to understand where the detail is located in the long shot. In this situation, an *intermediate* or *mid shot* is used to bridge the difference in scale. For example, a long shot may show a busy office filled with workers at their desks. The next shot may show someone's hands at a terminal. This abrupt change to an close-up could leave the viewer confused about the location of the terminal and its user. By using a mid shot, however, the user can be shown first with surrounding context before switching to the close-up. Triple-take filming thus assures that the film editor will have a suitable set of shots to work from later.

3.4 Multiple Viewports

There are some situations in which it is necessary to show the output of multiple cameras simultaneously through the use of multiple viewports. An *inset* is often used when it is important to emphasize the close relationship between two simultaneous events, one shown in the main viewport and the other in a nested inset. The inset viewport can depict a magnified detail from the main viewport, a parallel event of related but less significance, or a parallel event that is either the cause or effect of that shown in the surrounding viewport. A *split screen* may be used to show the same event from two or more perspectives or to allow independent events to be compared.

3.5 Continuity

After the shooting stops, the editing begins. *Continuity editing* is performed by the film editor, using a set of techniques intended to minimize jarring transitions between shots and to maintain consistency throughout the film. There are certain obvious continuity problems that can be avoided at the time the film is shot. For example, an actor may wear a red carnation one day, but on another day may wear a white one or may wear it on the wrong side. Precise records are kept to avoid such problems.

Other continuity problems are avoided during both shooting and editing. For example, there are a number of rules for joining shots together. *Motion vectors* specify the screen direction of motion and *index vectors* specify the direction of a gaze or gesture (as in, "They went that a way") [Zet73]. A *line* is established by an object's motion vector or index vector. If the camera starts on one side of the line in one shot and moves to the opposite side in the next shot, a moving object will appear to be moving in the opposite direction from that in the first shot and a pair of objects with converging index vectors will appear to be reversed in position. Both of these disturbing effects are caused by camera placement that is said to *cross the line.* If the *line* must be crossed, an intermediate shot is often used, and is taken when the camera is positioned on the line with its view vector parallel to the line, to prepare the viewer for the change to follow in the subsequent shots. For example, if an object is moving across the screen from right to left, an intermediate shot may be used to show the object moving into or away from the camera, before following it with a shot of the object moving from left to right.

If camera position or field of view is changed between shots, the change should be substantial enough that it appears to have a purpose. It is generally suggested that a 30° rotation about the object of interest is sufficient camera movement [RM68].

A visual problem results when an object appears on the left half of the screen in one shot and on the right half of the screen in the next: The object will seem to have jumped inexplicably to the new position. This can be avoided by making the cut so that the object is shown in the same relative screen position at the end of the first shot and at the start of the second.

Film editors will adjust the length of each shot so that an entire movement is visible. A cut should not interrupt a steady, flowing action. Jarring transitions between shots can occur when the field of view changes. Often, when cutting between two shots of the same action, several frames at the start of the second shot will overlap or repeat the action in the first shot. There is a moment of confusion that can be compensated for by duplicating the action [Met74]. Alternatively, when cutting from a long shot to a close-up, some editors will delete several frames of movement at the start of the second shot because the "shock value" of the abrupt move closer carries with it the illusion of movement.

4 A Testbed for Automated Generation of Animation

We are currently developing a research testbed that we are using to experiment with the automated generation of animated presentations, building on some of the concepts that we have discussed. ESPLANADE (Expert System for PLANning Animation Design and Editing) is a rule-based system that is provided with three kinds of input: a detailed plan of the actions occurring in a world and their causal relationships; information about the geometry of the objects in the world, their properties, and relations; and a description of constraints on the presentation that will guide the information to be presented. ESPLANADE produces as output a plan for a finished presentation.

The input action plan is the sort of information that can be generated by an automated planning system; thus far, we have built several simple planners that generate input for ESPLANADE. The input plan expresses all movement, and all changes in object properties and in the relationship among objects. The input plan must be sufficiently detailed to allow ESPLANADE to determine the location and properties of any object at any time. It contains a start state and the goal hierarchy that was generated in accomplishing the plan. The goal hierarchy references at its leaves the primitive operations that transform objects and their properties. The action planner creates a set of event information (`eInfo`) records that specify when and where each event occurs, the surface normal of the face of interest, and event-specific arguments. These records are of the form:

```
;Event information
(eInfo name gen4 type pressButtons
begTick 0 endTick 26 duration 26
begPos -25 -21 -95 endPos -25 -21 -95
center -20 -21 -88 normal 1 0 0
args buttons off on off on off on)
```

Other action planner information provided to ESPLANADE indicates the relationships among objects. For example, A is attached to B and, therefore, will move if B moves.

ESPLANADE translates this input into a table that allows it to determine efficiently for each point in time in the input world, the set of events in progress, and the current state of each object.

ESPLANADE also requires as input polyhedral solid models of the objects that are used for camera planning, as discussed below. (For reasons of efficiency two separate representations are maintained: the actual object geometry that is used during rendering and a simplified version with objects represented with fewer polygons that is used for camera planning. For example, a many-sided pyramidal "cone" in the actual geometry is represented by a four-sided pyramid.)

ESPLANADE's rules create a presentation plan for a completely specified animation. The presentation planner output includes specifications for cameras, viewports, and transitions. The camera specification has information about the time the camera is active, the camera position, the center of attention, and the field of view. The camera specification also has a list of event names that are visible in its field of view. For example, the camera specification for a 27 tick shot:

```
;Camera information
(cInfo name gen1 begTick 0 endTick 26
duration 27 scope long field wide
angle centerMiddle move no
orient vertical offset NULL
eRecTime 0 eList gen5 vpName gen4
startpos 70 15 -120
stoppos 70 15 -120
startref 20 -25 -65
stopref 20 -25 -65)
```

Viewport information is indexed by the camera specification's `vpName` field. The viewport specification indicates the position and size of the viewport and indices to transition effects. For example, the viewport record indexed by the camera information just shown specifies a full viewport:

```
;Viewport information
(vInfo name gen4 type full begTick 0
endTick 26 duration 27 begTrans gen2
endTrans gen3 origin 0 0 offset 1 1
camera gen1)
```

The transition information describes an effect to use when a viewport is added to or removed from the display. The viewport shown above uses a fadein during the first six ticks and a cut at completion. These transition records are:

```
;Transition information
(tInfo name gen2 type fadein
begTick 0 endTick 5 duration 6
vpName gen4)
(tInfo name gen3 type cutout
begTick 26 endTick 26 duration 1
vpName gen4)
```

An animation is played back by traversing in real-time, the output information from ESPLANADE. For each camera that is active at a tick, a viewport is sized and positioned on the display. A transition effect is performed if it is active at the current tick. The primitive action indexed by the camera event is executed and the environment state is modified. The frame is then displayed using the solid model for rendering. A graphic backend is responsible for executing and rendering the animation.

In the course of designing and implementing ESPLANADE, we have encountered a number of difficult issues, some of which we discuss in the following section. Several of these issues are illustrated with pictures generated by our current version of ESPLANADE, based on an input plan created by a planner that operates in a simple world—a room containing a crane and a collection of boxes (see Figure 1). A button box on the wall controls the crane, driving it to pick up, move, and release the boxes to move them from a start configuration to a goal configuration.

5 Research Issues

5.1 Selecting and Ordering Events

Given a topic for an animation and the event plan as input, the presentation planner must select the events from the plan that are necessary to produce an animation that fulfills the input constraints. For example, if the input plan is a complete plan for building a house, and the presentation is to describe how to install windows, the system should select those events related to window installation.

Although the input plan may be highly parallel, animation is an inherently sequential process in which one shot follows the next. A limited amount of true parallelism is available, however, through the use of multiple simultaneous viewports. Therefore, a presentation planner will often have to linearize parallel events, and in some cases could even elect to change the order of sequential events (e.g., to show a later important event before an earlier less important one). It is often necessary to establish for the viewer that a set of events are in fact occurring in parallel. If multiple viewports are not used, this can be achieved by intercutting short sequential segments of the parallel events.

5.2 Planning Camera Placement, Movement, Framing, and Field of View

Camera placement is one of the most critical decisions made by a presentation planner. It affects nearly every phase that follows, including how additional cameras are placed.

The viewing specification must satisfy several constraints, determined in part by one or more objects being viewed:

- *The object must be visible.* This involves choosing a view volume that contains the object and ensuring that the object is not obstructed by intervening objects. If we think of the object as an area light source, it will be at least partially obstructed when seen from any viewpoint that lies in the shadow cast by other objects illuminated by that light source. Regions in umbra cannot see any point on the object, and, thus, are completely obscured from it. Regions in penumbra can see some, but not all, points on the object, and thus, are partially obscured from it. Nishita and Nakamae [NN85] describe an approach to determining these regions analytically. We are implementing this approach using the solid modeling component developed for [CF89], based on the BSP-tree CSG algorithm [TN87]. A similar approach to viewpoint planning is described in [CK87].

- *The object's projection must fill a specified amount of the viewport.* In a close-up shot (narrow field of view), the object will generally occupy more of the viewport than in a long shot (wide field of view). Figure 1 shows an establishing long shot of the crane environment, and the following mid shot for attaching the crane's cable to a box, as generated by ESPLANADE.

- *The projection should provide as much geometric information about the object as possible.* Kamada and Kawai [KK88] call a viewpoint that fulfills this requirement a "general position," and present a simple algorithm for determining a general position for a wireframe object. A loss of generality results, for example, when a cube is viewed along the normal to a face and projects to a square. Arnheim [Arn69] refers to projections that are especially informative with regard to the 3D structure of their objects as possessing *renvois.* This is an extremely difficult problem. ESPLANADE currently selects a projection based on maximizing the visibility of one of a set of "key" faces specified in advance for each object of interest.

- *The viewing specification should not violate the visual continuity rules.* This means that a view for one shot must be specified in context of the views chosen for shots in the sequence and scene of which it is part. Especially important is consistency in *framing,* the information about the part of the screen onto which the object is projected. Maintaining consistency may require lookahead, which we discuss below.

- *The viewing specification should be selected to minimize the well documented distortions of extreme linear projections* [HE76, Kub86]. For example, if the viewpoint is too close to the object, extreme perspective foreshortening occurs.

Sometimes a single static viewing specification cannot be found that will keep the objects participating in an event visible, while meeting these other constraints throughout the duration of the event. There are several options available:

- Use a set of discrete viewing specifications, each of which is suited to part of the entire event.

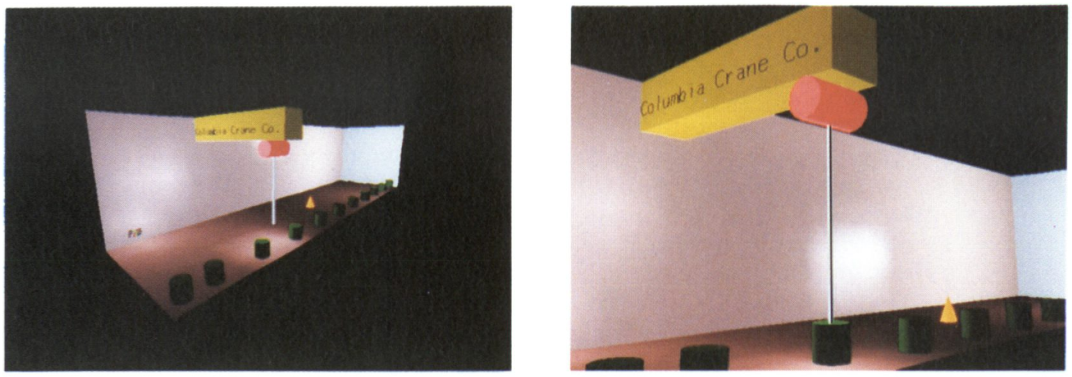

Figure 1: Long shot and mid shot of crane world.

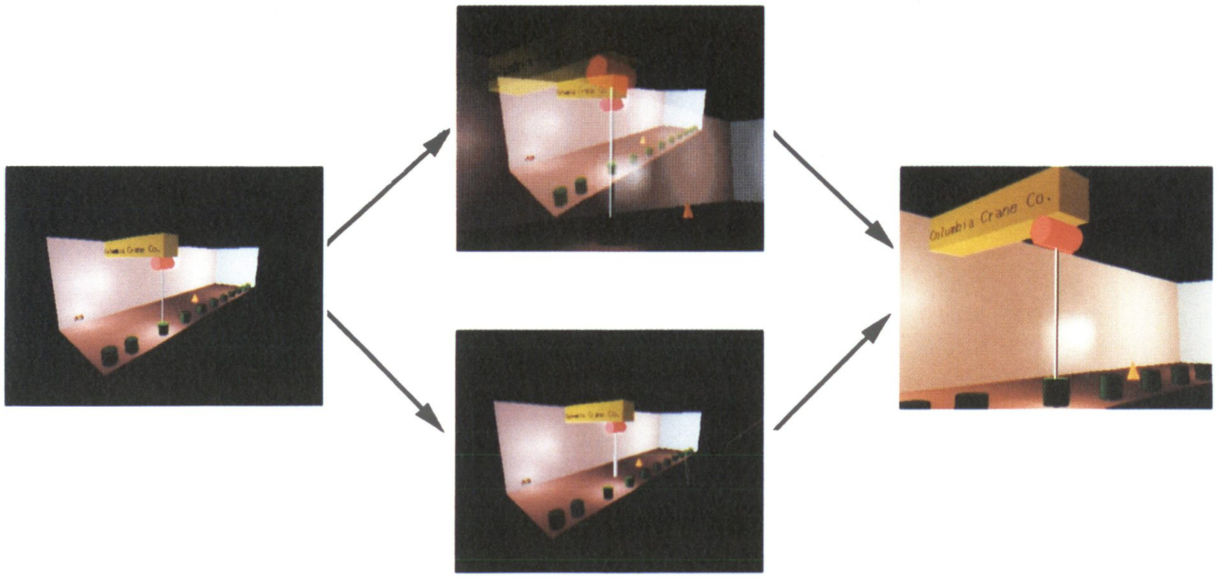

Figure 2: Long shot and mid shot joined by a dissolve vs. a cut.

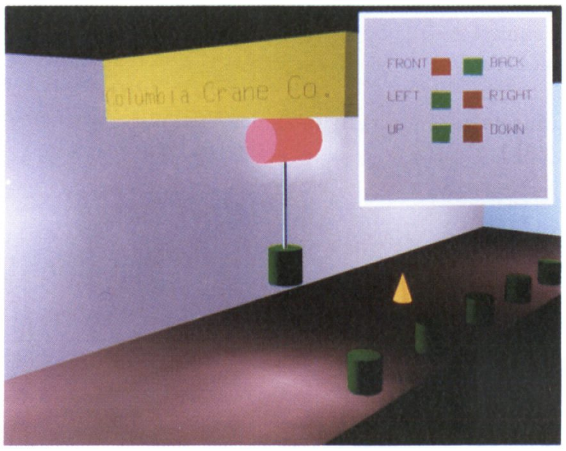

Figure 3: Inset viewport inside of a main viewport.

- Modify one or more parameters continuously. For example, the camera may be moved along a continuous path.
- Remove blocking objects from the scene or render them as partially transparent or in cutaway view [SF89]. In contrast to the use of real cameras in a world of real objects, we can easily remove an obstructing object or make it partially transparent, allowing the use of an otherwise satisfactory viewing specification.

5.3 Maintaining Continuity

An automated presentation should attempt to maintain visual continuity when possible. Line-crossing errors introduce the most confusion. In ESPLANADE we compute the cross product between the motion or index vectors of an object and the vector from the center of projection to the object. The sign of the cross product can be compared to that computed previously for the same object to avoid placing the camera on the opposite side of the *line*. If the sign of the cross products differ then the camera has been placed across the *line*.

5.4 Selecting Transitions

The transition between shots must be selected to conform to the current section of animation and the function of the transition depends on the two shots that are being joined. A dissolve can sometimes be used to cover line-crossing errors. However using dissolves too often can destroy event continuity [Zet73]. Figure 2 shows two choices for a transition—a dissolve and a cut—used for the two shots of Fig. 1. (Section 7 explains how ESPLANADE'S real-time renderer performs dissolves.) Note how the dissolve appears to be inappropriate in this situation.

5.5 Selecting Viewport Layout

A decision must be made about which cameras' views will be visible, and the size and placement of the viewport in which each will be displayed. ESPLANADE's rules allow inset viewports to be used to show causality. Figure 3 shows an inset viewport that is created to depict an event that caused the event depicted in the main viewport. (An approach to generating insets automatically in static pictures is described in [SF89].)

5.6 Eliminating Unnecessary Details and Redundant Events

Eliminating unnecessary details requires knowledge of the content domain and is related to the problem of selecting relevant events. Events may be selected as relevant because they satisfy some of the input constraints on the material to be presented, but may then be eliminated because they are at a level of detail that exceeds the requirements of the animation. For example, in the case of detaching two objects that are bolted together, events that involve removal of the bolts would be selected for inclusion, since they are required for detaching the objects. If all the bolts are similar, however, loosening the second bolt may be shown with less detail than loosening the first. (In fact, the details of removing a bolt do not need to be shown at all, if the user is known to understand how to do it.) Furthermore, in context of an animation, removing the set of bolts might be depicted by showing only the first and last, with a dissolve used as the transition between the shots to indicate that some intervening events have been deleted.

5.7 Determining Shot Duration, Ordering, and Total Screen Time

A scene's shot structure is intended to show a complete event. Not only is the event shown in its entirety, but it is usually depicted using a variety of viewing specifications. Selecting an appropriate duration for a shot depends on the usefulness of the image content. If an event can no longer be viewed with the current viewing specification, then a new one must be selected. Sometimes this can be accomplished by modifying the viewing specification smoothly during a single shot; a discontinuous change (impossible in real-time with a physical camera) would correspond to a transition between shots. Shot duration is the inverse of cutting rate. As mentioned earlier, a high cutting rate is important for maintaining visual momentum. On the other hand, there may be a conflict between the excitement of visual momentum, and the need to communicate information clearly that may be better served by a low cutting rate. ESPLANADE currently sidesteps this issue by assigning a camera to an event for its entire duration.

5.8 Allowing Lookahead

ESPLANADE is provided as input a completed plan for a superset of the actions to be depicted. If ESPLANADE is allowed to look ahead at the entirety of the plan before generating a single frame, it can plan continuity and animation length based on a global view of all actions to be depicted and of the entire animation produced thus far. On the other hand, if a system were called upon to create a live presentation of ongoing actions that are being planned as they are presented (or whose plans are unknown before they are executed), lookahead would not be possible. Given predetermined actions, we can simulate this restriction by eliminating lookahead completely. In this case, continuity planning can only be based on previously generated frames. Between these two extremes is the notion of near real-time presentation. For example, a live television presentation may actually be delayed by several seconds to allow coarse, real-time editing of material. This effect may be simulated by allowing ESPLANADE a limited number of frames of lookahead. Note that in all cases ESPLANADE does not actually generate the presentation plan in real-time, but does generate a plan that would be identical to one that it would create, were it given real-time input with the stated amount of lookahead.

Consider, for example, one kind of continuity problem that can occur when the presentation system has severely limited lookahead. As shown in Fig. 4, ESPLANADE can open an inset viewport to show an event being performed in parallel with another and can close it after the event has completed. Several frames later, however, if a similar pairing of events occurs, the inset viewport may be reopened in the same position. Without lookahead, this can produce the disconcerting effect of a viewport repeatedly appearing and disappearing from the display. With sufficient lookahead, ESPLANADE's rules keep the viewport open across a set of events that are presented within a predetermined time period. We are particularly interested in experimenting with varying amounts of lookahead to determine how the tradeoffs between animation quality and immediacy are affected by the kind of actions being presented.

6 An Example Animation

Figure 5 shows example frames from part of a scene generated by ESPLANADE. The animation demonstrates some of the capabilities discussed in the paper, including the use of different camera specifications, transition effects, overlapping action on a transition, and multiple viewports for displaying parallel actions. The example, starts with a fadein (a) to a long shot (b) to show the entire environment. In (b), the crane is shown moving in a long shot and in (c) it stops at one of the boxes. In the cut to the mid shot in frame (d), we see the slight repetition of action used when changing between camera specifications. In frame (e), the mid shot is used to show the box being attached to the crane. In frame (f), an inset viewport is opened to show that buttons are being pressed, causing the crane and box to move to another location. Frame (g) shows the box being detached from the crane. Instead of removing the inset (because no action is being shown in it), ESPLANADE uses lookahead to discover that the inset will be used several frames later. Frame (h) shows the buttons being pressed again in the inset viewport and the crane moving to get the next box.

7 Implementation

ESPLANADE is implemented under UNIX using the CLIPS production system language [Cul88] and C++, and was used to make the figures in this paper. Its renderer runs in real-time on an HP 9000 370 TurboSRX, using 3D polygon-based object geometry, and interprets a plan created during (non-real-time) execution of ESPLANADE's presentation planner.

ESPLANADE's dissolves and fades are rendered using a frame buffer mask provided by the graphics hardware to support "screendoor transparency." Each bit in a 4x4 mask replicated over the frame buffer may be set to enable or disable writing at that pixel. To perform a dissolve, we enable $n/16$ bits for writing when rendering shot A and the remaining $(16-n)/16$ bits when rendering shot B. If n is interpolated between 0 and 16, over a series of frames, A is faded in and B is faded out. (The moiré pattern in the first frame of Figure 5 is caused by the "beating" between the screendoor mask and the halftone mask used in printing.)

8 Conclusions and Future Work

We have discussed the desirability of automating the generation of animated presentations, and have provided a brief introduction to some of the cinematic techniques used to create effective films. Some of the issues that must be addressed to automate the use of these techniques were surveyed, and several of these were illustrated by frames generated by the current version of a research testbed that we are using to explore these issues.

One major direction that we are pursuing is the choice of viewing specification, based in part on the criteria discussed above, and on the use of varying amounts of lookahead to influence the amount of forward continuity. Ultimately, we are interested in incorporating animations designed by ESPLANADE into the COMET system for generating multimedia explanations being developed at Columbia [FM90].

9 Acknowledgements

This work is supported in part by the Defense Advanced Research Projects Agency under Contract N00039-84-C-0165 and the Hewlett-Packard Company under its AI University Grants Program. Kevin Ramsey provided advice on filmmaking techniques and assisted in prototyping ESPLANADE's rules. Sam Fenster helped design and implement parts of ESPLANADE. Norman Chin wrote the CSG package that is being incorporated into ESPLANADE to help plan its viewing specifications.

References

[Arn69] Rudolph Arnheim. *Visual Thinking.* Faber and Faber, London, 1969.

[BC89] Armin Bruderlin and Thomas Calvert. Goal-directed, dynamic animation of human walking. In *Proc. ACM SIGGRAPH 89*, pages 233–241, Boston, MA, July 1989.

[Bur81] Noel Burch. *Theory of Film Practice.* Princeton University Press, New York, 1981.

[CF89] Norman Chin and Steven Feiner. Near real-time shadow generation using bsp trees. In *Proc. ACM SIGGRAPH 89*, pages 99–106, Boston, MA, July 1989.

[Che74] David Cheshire. *The Book of Movie Photography.* Alfred A Knopf, New York, 1974.

[CK87] C. Cowan and P. Kovesi. Automatic sensor placment from vision task requirments. Technical report, SRI, Menlo Park, Ca, June 1987.

[Cul88] C. Culbert. Clips reference manual. Technical report, NASA/Johnson Space Center, TX, April 1988.

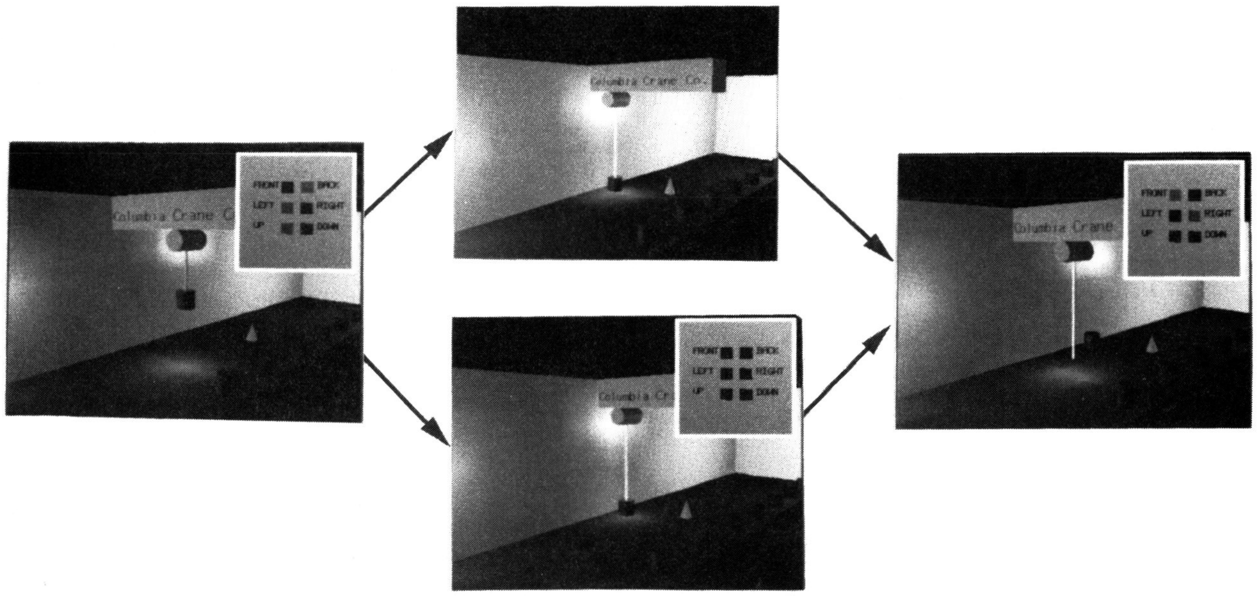

Figure 4: Continuity is sacrificed if inset viewport is eliminated between shots, and is maintained if inset is retained.

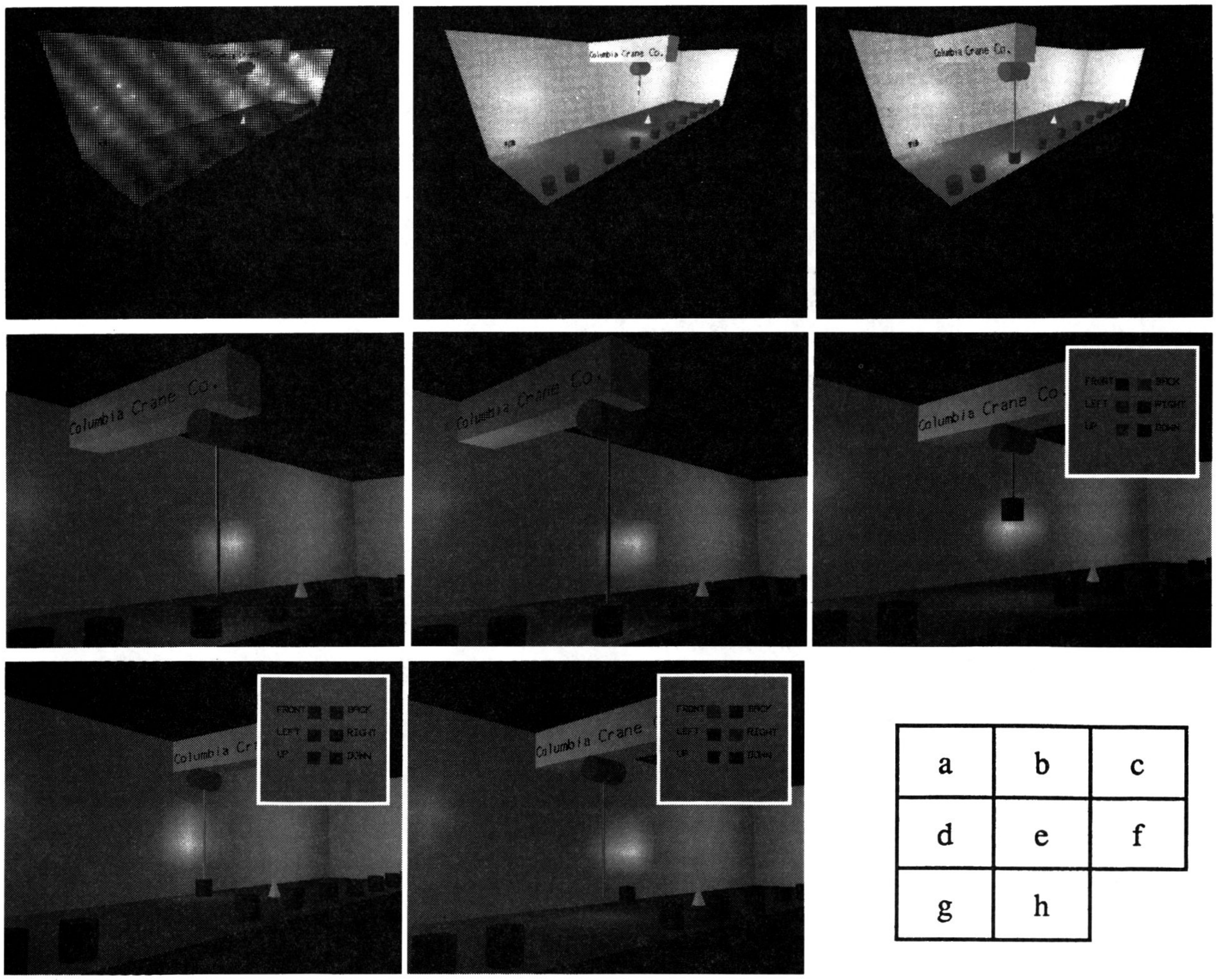

Figure 5: A segment of animation generated by ESPLANADE

[EBJ89] Jeffrey Esakov, Norman Badler, and Moon Jung. An investigation of language input and performance timing for task animation. In *Proc. Graphics Interface 89*, London, Ontario, June 19-23 1989.

[Fei85] Steven Feiner. Apex: An experiment in the automated creatation of pictorial explanations. *IEEE Computer Graphics and applications*, 5(11):29–39, 1985.

[FM90] Steven Feiner and Kathleen McKeown. Generating coordinated multimedia explanations. In *Proc. CAIA90 (6th IEEE Conf. on Artificial Intelligence Applications)*, Santa Barbara, CA, March 5-9 1990.

[HB78] Julian Hochberg and Virginia Brooks. The perception of motion pictures. In E. Carterette and M. Friedman, editors, *Handbook of Perception X: Perceptual Ecology*, chapter 11. Academic Press, New York, 1978.

[HE76] Margaret A. Hagen and Harry B. Elliot. An investigation of the relationship between viewing condition and preference for true and modified linear perspective with adults. *Journal of Experimental Psychology: Human Perception and Performance*, 2(4):479–490, 1976.

[Kah79] Kenneth M. Kahn. *Creation of Computer Animation from Story Descriptions*. PhD thesis, AI Lab, Massachusetts Institute of Technology, Cambridge, MA, January 1979.

[KK88] Tomihisa Kamada and Satoru Kawai. A simple method for computing general position in displaying three-dimensional objects. *Computer Vision, Graphics and Image Processing*, 41(1):43–56, January 1988.

[Kub86] Michael Kubovy. *The Psychology of Perspective and Renaissance Art*. Cambridge University Press, New York, 1986.

[Met74] Christian Metz. *Film Language: A Semiotics of the Cinema*. Oxford University Press, New York, 1974.

[Nei82] Daniel Neiman. Graphical animation from knowledge. In *Proc. AAAI 82*, pages 373–376, Pittsburgh, PA, August 18-20 1982.

[NN85] Tomoyuki Nishita and Eihachiro Nakamae. Continuous representation of three-dimensional objects taking account of shadows and interreflection. In *Proc. ACM SIGGRAPH 85*, pages 23–30, San Francisco, CA, July 1985.

[Rey87] Craig Reynolds. Flocks, herds and schools: A distributed behaviorial model. In *Proc. ACM SIGGRAPH 87*, pages 25–34, July 1987.

[Rid87] Gary Ridsdale. *The Director's Apprentice: Animating Figures in a Constrained Environment*. PhD thesis, School of Computing Science, Simon Fraser University, Burnaby, BC, 1987.

[RM68] Karel Reisz and Gavin Millar. *The Technique of Film Editing*. Focal Press, London, 1968.

[Rub89] Benjamin Rubin. Constraint-based cinematic editing. Master's thesis, MIT Media Arts and Science Section, Cambridge, MA, June 1989.

[SF89] Dorée Seligmann and Steven Feiner. Specifying composite illustrations with communicative goals. In *Proc. UIST 89 (ACM SIGGRAPH Symp. on User Interface Software and Technology)*, pages 1–9, Williamsburg, VA, November 13-15 1989.

[TN87] William Thibault and Bruce Naylor. Set operations on polyhedra using binary space partitioning trees. In *Proc. ACM SIGGRAPH 87*, pages 153–162, Anaheim, CA, July 27-31 1987.

[Wil87] Jane Wilhelms. Towards automatic motion control. *IEEE Computer Graphics and Animation*, 7(4):11–22, April 1987.

[Zel83] David Zeltzer. Knowledge-based animation. In *Motion: Representation and Perception (Proc. ACM SIGGRAPH/SIGART Workshop on Motion)*, pages 187–192, Toronto, Canada, April 4-6 1983.

[Zet73] Herbert Zettl. *Sight, Sound, Motion—Applied Media Aesthetics*. Wadsworth Publishing Co., Belmont, CA, 1973.