# Automating Spielberg

Doron Friedman
Department of Computer Science
University College London
Gower Street, London WC1E 6BT, UK
http://www.cs.ucl.ac.uk/staff/d.friedman
d.friedman@cs.ucl.ac.uk

**Abstract**

**Synthetic 3D animation is becoming increasingly popular, and with the advance of software tools the authoring process is gradually being automated. It is now possible to conceive of a fully automated process that would convert textual screenplays into animated movies. This could allow non-professionals to easily create 3D visualizations, as well as assist professional animators in rapidly producing animation prototypes. We have implemented a system that automates camera decision-making, which is arguably the heart of filmmaking. In this paper we try to draw lessons from what we have already done, and describe a hypothetic complete framework for converting screenplays into movies.**

## Introduction

The famous British producer Sir David Putnam once told of a conversation he overheard in a conference. One person was asking the other: "Do you think computers will ever make movies?" "Sure," the other replied, "and other computers will watch them." In this paper we try to take the first part of this conversation seriously, and, based on our research, try to see what it takes for computers to automatically create movies. We acknowledge that creating a movie generally requires human-level intelligence, far beyond today's capabilities in computer science. Thus, our goal is to simulate the conversion of a screenplay into a movie as it is done within well-defined processes and genres (hence Spielberg, rather than, say, Fellini).

As synthetic computer animation becomes more popular, the computer is used to make the authoring process easier, faster and more efficient. In the most popular method, animators only need to describe the scene at a discrete set of time points, called key frames, and computation is used to interpolate between these scene descriptions and create a complete movie, with 24 or 30 frames per second. Additional automation seems to gradually become more acceptable: inverse kinematics is used for skeleton animation, lighting and shadowing are often computed automatically, physical models are used to simulate dynamic systems, landscape is generated automatically, and more [16].

Automated movie making has applications for both off-line generation of linear animation and real-time interactive virtual environments. Such a tool may be used by non-professionals to easily visualize scenarios, and by animators to rapidly produce a first sketch, which they can later refine. For interactive environments, the need is even more

obvious: since a real-time director is not available, all the cinematic decisions need to be carried out automatically.

Studying automated cinematography is also an exciting exercise in automated art. Computers are very successful at producing abstract art forms, such as abstract paintings and music, but are notoriously bad at creating art pieces that relate to the real world, such as generating stories. Movies share both aspects: cinematic language includes many formal principles, yet movies are typically concrete and describe real-world events.

In this paper we provide an overview of what we have implemented already, which covers subsets of the movie-making paradigm, and then sketch a possible framework for the whole paradigm.

## Background

Computer graphics has made significant progress since the term was introduced in 1970; it is now possible to produce sophisticated photo-realistic animation sequences. In order to automatically generate movies, the bottleneck is not with the state-of-the-art in computer animation, but rather with artificial intelligence (AI). Despite over 50 years of research, we are still very far from computers that possess human-level intelligence; we will call this "the AI problem." The main challenge in automating movie making is to bypass this problem.

What would be the input to such a system? One alternative is to expect the system to be able to process a screenplay in a natural language. Natural-language understanding is a notoriously difficult problem; it has been addressed by the AI community from the earliest days of computing, and is still considered to be an unsolved problem in the general case. We note that a typical screenplay may be easier to understand than a typical story; screenplays are more concrete and provide more information. In a story, you may find a sentence like: "Alice and Bob got married." In a screenplay, if such an event took place, it will include a detailed description of the scene, and the specific actions carried out by the actors.

Regardless of how the story is input into the system, some level of automatic story understanding is required. Story understanding was studied by Schank for the last few decades [19,20]. Among other contributions, one may want to build on his theory of primitive actions, which suggests a classification of all actions into eleven generic actions. Story understanding is probably "AI-hard", i.e., it cannot be solved unless we have human-level AI. One way to override it is by allowing the input screenplay to contain annotations, or hints, which will compensate for the tool's lack of understanding.

Automated camera control has received some research attention, including efforts at automating some cinematic principles. Idiom-based approaches try to capture cinematic principles and let them dictate camera behaviour. Several works have attempted some formalization of cinematic principles. The first to attempt some automatic cinematography were Karp and Feiner [15]; their ESPLANADE system used an AI technique called Planning. Butz [3] proposed that the rules of cinematic expression are analogous to grammar in natural language, and this was the basis for his CATHI system. Christianson et al. [5] defined DCCL (Declarative Camera Control Language) and attempted a more

systematic analysis of cinematography. They describe several cinematic principles and show how they can be formalized in a declarative language. They encode 16 idioms, at a level of abstraction similar to the way they would be described in a film textbook.

Real-time camera behaviour was addressed by several projects; the Virtual Cinematographer [14] formulates some idioms as finite-state machines, which may then be used to make real-time decisions in 3D chat environments on the Web. Bares et al. investigated user modelling [1] and task sensitive camera behaviour [2]. Tomlinson, Blumberg, and Nain [22] used a behaviour-based approach; the camera is modelled as an autonomous agent, and its behaviour is based on a reactive behaviour system, with sensors, emotions, motivations, and action-selection mechanisms.

Coyne and Sproat [6] have constructed an automated tool that converts natural language textual descriptions into static images. Their work is an excellent first step towards automated animation generation. They analyze the text, and construct a synthetic 3D scene, based on a library of thousands of 3D objects. Although they do not deal with action, they deal with human postures. Coyne and Sproat deliberately do not cope with missing information in the input description; their system fails to generate an output in such cases.

## What We Have Done

Our approach is knowledge-based, and is thus similar to the idiom-based approaches. However, we claim that idioms are the wrong granularity, being too coarse to formalize cinematic knowledge. Using idioms, one needs to code a specific idiom for every possible situation. This method results in a repetitive and predictable output, which impedes user engagement. For the same reasons, using idioms does not scale to more complex situations. Our approach looks at a lower level of facts, which together comprise idioms. Another problem with idiom-based approaches is that they assume that an editing algorithm is known. We believe this view is too optimistic for a domain as complex as cinematography, and have thus opted for a declarative knowledge representation approach.

In order to evaluate our approach we have implemented a system called Mario (see figure 1).[1] The inputs to Mario are a screenplay and a floor plan. The screenplay is given in a formal language. The reasoning engine applies cinematic principles, taken from the cinematic knowledge base, to the inputs. The output is a list of the camera parameter values, according to the knowledge base. The reasoning engine also produces a log of its reasoning process. The system can create a synthetic 3D movie corresponding to the screenplay and the camera decisions, if 3D models and animations for the objects and actions mentioned in the screenplay are available. The animations are currently output to VRML97 [13].

We have surveyed a few textbooks on cinematic principles with different levels of sophistication [21, 23, 4], and used a professional film and video editor as a domain expert. We translated the principles extracted from these sources into formal rules. Mario is based on an existing system, implemented in Common Lisp, called Cake [17]: a multi-layered

[1] The system is named after one of the main characters in the telenovela that was selected as the first test case.

reasoning system developed at MIT's Artificial Intelligence Laboratory in the late 1980s. Cake's architecture includes seven layers; the bottom six layers provide generic knowledge representation and automated reasoning facilities, and a top layer called *Plan Calculus*, which includes generic facilities to implement a specialized formalism for software development. We have replaced the top layer with Mario, which is a specialized layer for reasoning about cinematography.

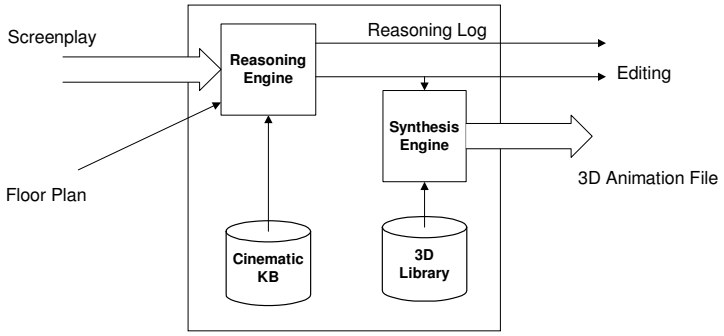


Figure 1. Mario system overview.

The technical details underlying Mario can be found in [8, 9]. Here we only provide a high-level and non-technical description of the method. There are infinitely many possibilities for camera behaviour in a scene. We use cinematic principles to reduce the search space into a small discreet set of options. We assume that at each point in time the camera has a target (or targets), a shot type (such as close up, long shot, etc), and a profile angle (such as frontal view, left profile, etc). These three features will often determine the exact camera position. Occasionally, there will also be usage of atypical angles (such as low angle shot, or a view from above), zoom, and tilt. We let the domain expert describe rules in terms of these features, and use automated reasoning to apply these rules to a given situation.

As a first genre to cope with we selected telenovela, a Latin-American form of TV soap opera, which is infamous for its simplistic use of cinematic language.[2] Here are a few rules as specified by the domain expert:

- If an actor is speaking, she is displayed in a frontal (30-degrees) medium shot.
- If an actor is walking, she is displayed in a long shot, from a ¾ angle.
- Cameras don't cross the **line-of-interest** with a cut. In our simple example, the line-of-interest is the line that connects the two-dimensional positions of the two actors.
- There are no jump cuts; specifically the angle between two consecutive shots is at least 30 degrees.

We formulated each such rule into a set of logical formulae. Clearly, this formalization required refining some details, and this was done with consultation of the domain expert or

[2] In fact, this genre is so predictable that it may be worthwhile to attempt automated screenplay generation as well.

the cinematic textbooks. The formulae are generally independent, and the interaction of many low-level formulae gives rise to high-level cinematic phenomena (e.g., see figure 2).

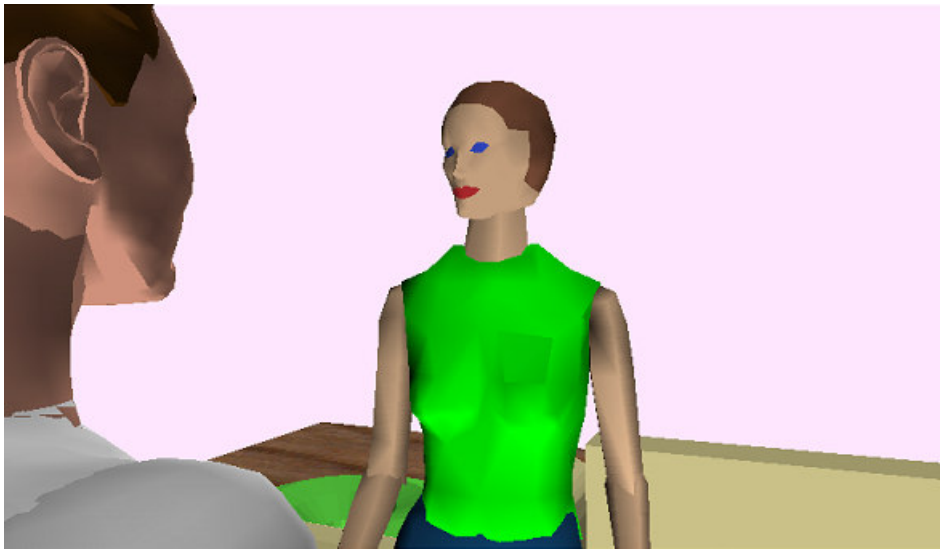

Figure 2. An over-the-shoulder shot emerges from low-level cinematic principles.

One of the unique features of our system is that we use non-monotonic reasoning; Let us explain this. Traditional logic is monotonic: if some fact is true at some point in time, it will always be true, and cannot be undone. Human thought, and cinematic reasoning probably included, is often non-monotonic: we may believe some fact to be true, but given some new evidence we may change our minds. Such non-monotonic framework is more appropriate for reasoning about cinematic principles. Our algorithm applies generic rules to a given situation, but it can also cope with exceptions.

We have run Mario and evaluated the results for over a hundred scenes. Due to a limitation in animation generation, most of our examples only included a restricted set of actions: walking, speaking, jumping, and running. The largest example we analyzed was a complete telenovela scene, which included 41 actions, resulting in an animation sequence of 2 minutes and 20 seconds. This example was tested with a variety of rule sets. The version of the knowledge base that produced the best results, according to our domain expert, includes a few dozen formulae. During the processing of the full scene, the system was loaded with over 10,000 logical terms.

In addition to Latin telenovela, we have analyzed scenes from additional genres, mainly a more dramatic genre (based on the TV science-fiction series *The X-Files*). This introduced new challenges; mainly, it includes a much richer set of situations and locations compared to telenovela. As we have refined the knowledge base and introduced new rules, we frequently confronted the problem of contradiction between rules. It is possible to keep the problem under-constrained by using retractable premises rather than axioms, but the result might be an arbitrary choice made by Mario to prefer one rule over the other.

Figure 3. A threatening character is shown from a low angle.

Conflicting assumptions are a well-known problem in non-monotonic reasoning [12], but there does not seem to be a general solution. We have examined a domain-dependent solution to this problem: classifying rules into categories of high-level goals. The goals we have identified are: spatial orientation, conveying the information explicit in the script, conveying the information that may be implicitly deduced by a viewer from the script, aesthetic considerations, and parsimony: using the minimum number of shots and cameras. In cases of conflicts, the system can try to satisfy all the goals, rather than choose preferences arbitrarily. The goals component has not been implemented.

Cinematic formalization may help the filmmaker organize the cinematic knowledge in her mind. This is useful for gaining new insight into cinematic expression. By abstracting and formalizing the domain space, the filmmaker may become aware of new options.

We can illustrate this by examining two examples of movies created by Mario, by using simple and deliberate violation of rules. In the first, we asked Mario to prefer complex shots rather than simple shots. Note that our method is fully deterministic, which means that even if there are arbitrary choices, we expect a high degree of consistency. In the case described above, Mario preferred modifying the profile angle from 3/4 to frontal, and also preferred zooming in. The result was of a very consistent style including a lot of camera rotations, which several viewers called “the Matrix” style.

An additional small modification produced a completely different style. Instead of just requiring complex shots, we require the beginning and the end of each shot to be different in **all** aspects. We expected very dynamic camera behaviour, but watching the resulting movie was still surprising. The consequence of the new rules was that in almost all of the shots, the camera rotated between the two actors. This turned out to be a very consistent style, called by some viewers “the Ping Pong style.” Some film students mentioned that this style reminded them of the Dogma 95 cinematic genre, which is characterized by unstable and rapidly changing camera positions. Filmmaker Yigal Burstein remarked that this style reminded him of an experimental film by filmmaker Michael Snow.[3] The point of this camera behaviour in Snow's experimental film was to use repetitive camera rotations to accentuate a sudden dramatic event. Thus, we see that, on the one hand, experimenting with the rules can lead to surprising effects. But on the other hand, we note that we are far from a tool that would deliberately select such a style to emphasize an event, as done by Snow.

[3] Apparently, this film has no name.

We have conducted an informal evaluation of Mario. The main idea was to conduct a kind of a "Turing test," that is, to see if viewers, and especially filmmakers, could tell the difference between Mario's editing and expert human editing. We have presented five versions of the telenovela scene in 3D animation, one of which was manually prepared by our domain expert. The audience was comprised of two groups: one including graduate film students and lecturers, and the other only included subjects with no background in film.

The complete details will be provided in the full paper [10]. The following points can be made about the results. First, only 8 out of 22 (36%) of the viewers recognized who made all three movies correctly. Due to the limitations of the experiment, we only see this as a first indication that Mario's result are comparable to human expert; this needs to be further investigated.

We also asked the viewers to rate the movies. We note that in both groups the rating for the human-made version was highest. It seems that the question whether the movie was edited by human or machine is misleading, whereas viewers do sense a difference, and judge that the human version is better.

In addition to work on Mario, we have examined a novel application of automatic camera for the automated generation of movie summaries from session in interactive virtual environments [11]. Such an application poses two challenges: the first is to automatically decide what is worthwhile showing, and the second is how to show it, which is based on the research in Mario. This gave us the opportunity to further study automated movie construction. Moreover, within the limited domain of a life simulation game, we can override the AI problem; it is reasonable to assume that the system has information about the actors, their goals, personality, and substantial common-sense knowledge relating to the game domain.

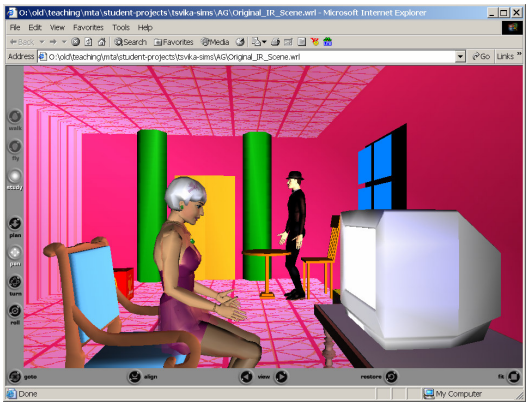

Figure 4. A snapshot from the life-simulation environment that was used for evaluating automated summarization.

## What May Be Done: The Complete Framework

Based on our experience with the two projects described above, and on some additional tools we have built, we can now describe the whole, hypothetical system (a diagram appears in figure 5).

The first step is to be able to accept the input screenplay. As mentioned before, we do not regard natural-language processing to be an essential part of the paradigm. A formal language that allows describing a rich set of scenarios should be sufficient, and it is possible to design user-interface solutions that will relieve users of learning its syntax. It is also possible to design a user interface that includes some two dimensional diagram of the location. The important point is that it should be able to convert these inputs into a scene description language, such as the one we used for Mario (see [9]).

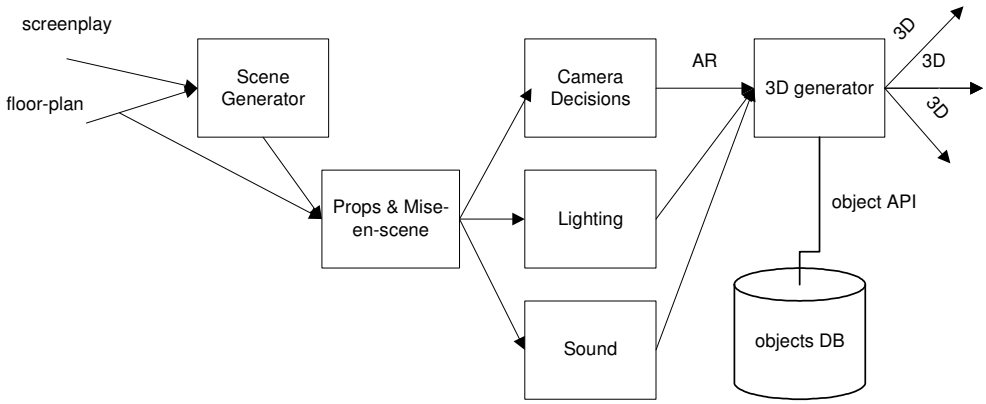


Figure 5. An overview of the hypothetical framework for automated movie construction.

Typically, even a detailed screenplay will leave a lot of information missing, as compared to a movie. A set of steps is required to complete this missing information, based on two types of knowledge: common-sense world knowledge, and cinematic knowledge. We have seen in Mario an example of how cinematic knowledge can be used to compensate for missing information in the form of camera behaviour. Similarly, we need to deal with set design and props, mise-en-scene, lighting, and sound; this is explained below.

Some of these problems are studied by the video game industry, and generally in real-time virtual environments. The reason is that in such real-time environments there is a need for automated decision-making. Our advantage is that we do not need to make the decisions in real-time, and, thus, are able to use algorithms that are more sophisticated but relatively slow, such as automated reasoning.

By set design and props we mean that for every object mentioned in the screenplay, we need to decide its precise location, size, colour, and style. Also, we may want to automatically augment the scene with typical objects; for example, if the scene takes place in an office, we would like to include at least one desk and chair, even if these are not mentioned in the screenplay. These decisions need to be coherent with respect to both common sense and artistic style. There are a huge number of facts in such a knowledge

base; while we cannot expect to have a complete description of every possible scene, we can hope to have a reasonably complete coverage of specific domains, and provide easy ways for the authors to extend this knowledge base for a given screenplay.

Mise-en-scene is cinematic jargon, which in our case stands for the location of actions and objects in a scene over time. Given the screenplay, our system needs to be able to plan the actor and object motions. This problem is very similar to path planning in computer games, which also relies on years of research in robot motion planning.

Lighting decisions can also be made based on both world knowledge and artistic considerations. Automated lighting has been attempted, to some extent, by Tomlinson, Blumberg, and Nain [22]. Sound includes sound effects that could come from a pre-recorded library, speech utterances generated by text-to-speech engines, and automated music. The last part is an interesting challenge; some work on real-time music composition for interactive environments was done by Robertson et al. [18] and Eng et al. [7].

Assume that we have been able to complete all missing information and generate a detailed scene description. It remains a major challenge to construct corresponding animation from such a description; there is still much research needed in reusable components for animation synthesis. This is especially difficult with human animation, since it is complex, and we are very sensitive to human details. Such animation synthesis issues are an active area of research within the real-time animation community.

# Conclusion

There is a growing interest in computer-generated art that displays an emergence of complex phenomena. Emergent behaviour is typically achieved by using evolutionary methods such as genetic algorithms. We have demonstrated how a form of emergent behaviour, being a property of any complex system, can be achieved using a knowledge-based approach.

Automating animation is a bottom-up process. Artists are typically reluctant to accept automation, but they gradually learn to feel comfortable using these techniques, by learning to control them and manually override the features they do not approve of. It may be possible to take things further, and automate the whole animation production process.

Our method will probably not be complete in the near future, and will not produce high quality results, as to substitute human generated animation. Yet, we believe it is feasible to construct a tool that may be useful in the scope of specific domains. Such a tool may either produce results fully automatically for visualization purposes for non-professional users, or be used as part of a semi-automated process by professional animators.

**References**

[1] Bares, W H and Lester, J C, Cinematographic User Models for Automated Real-time Camera Control in dynamic 3D Environments, *UM97: User Modeling: Proc. of the Sixth Int'l Conf.*, Jameson, A, Paris, C, and Tasso, C (eds), pages 215-226, Sardinia, Italy, 1997.

[2] Bares, W H, Zettlemoyer, L S, Rodriguez, D W, and Lester, J C, Task-sensitive Cinematography Interfaces for Interactive 3D Learning Environments, *IUI-98: Proc. of the 1998 Int'l Conf. Intelligent User Interfaces*, pages 81-88, San Francisco, California, 1998.

[3] Butz, A, Anymation with CATHI, *IAAI-97 Proc. of Innovative Applications of Artificial Intelligence*, pages 957-962, Providence, Rhode Island, 1997.
[4] Callev, H, *Cinematic Expression*, Optimus, in Hebrew, 1996.
[5] Christianson, D, Anderson, S, He, L, Salesin, D, Weld, D, and Cohen, M, Declarative Camera Control for Automatic Cinematography, *Proc. of the Thirteenth National Conf. Artificial Intelligence,* AAAI Press, pages 148-155, Menlo Park, CA, 1996.
[6] Coyne, B and Sproat, R, WordsEye: An Automatic Text-to-Scene Conversion System, *Proc. 28th annual conf. on Comp. graphics and interactive techniques*, pages 487-496, ACM Press, 2001.
[7] Eng, K, Klein, D, Babler, A, Bernardet, U, Blanchard, M, Costa, M, Delbruck, T, Douglas, R J, Hepp, K, Manzolli, J, Mintz, M, Roth, F, Rutishauser, R, Wassermann, K, Whatley, A M, Wittmann, A, Wyss, R, and Verschure, P F M J, Design for a Brain Revisited: The Neuromorphic Design and Functionality of the Interactive Space Ada, *Reviews in the Neurosiences*, volume 14, pages 145-180, 2003.
[8] Friedman, D and Feldman, Y, Knowledge-Based Formalization of Cinematic Expression and its Application to Animation, *Proc. EUROGRAPHICS*, Saarbrucken, Germany, 163-168, 2002.
[9] Friedman, D and Feldman, Y, Knowledge-Based Formalization of Cinematic Expression, *Proceedings of the Italian-Israeli Computer Science Forum on Research and Applications of Artificial Intelligence,* June, 2003.
[10] Friedman, D and Feldman, Y, Knowledge-Based Formalization of Cinematic Expression, under review.
[11] Friedman, D, Feldman, Y, Shamir A, Dagan T, Automated Creation of Movie Summaries in Interactive Virtual Environment, to be presented *at IEEE VR,* 2004.
[12] Ginsberg, M L (ed), *Readings in Non-Monotonic Reasoning*, Morgan Kaufmann, 1987.
[13] Hartman, J and Wernecke, J, *The VRML 2.0 Handbook*, Addison Wesley, 1996.
[14] He, L, Cohen, M F, and Salesin, D H, The Virtual Cinematographer: A Paradigm for Automatic Real-Time Camera Control and Directing, *Computer Graphics*, volume 30, Annual Conf. Series, pages 217-224, 1996.
[15] Karp, P and Fiener, S, Automated Presentation Planning of Animation Using Task Decomposition with Heuristic Reasoning, *Proc. of Graphics Interface '93*, pages 118-127, Toronto, Canada, 1993.
[16] Parent, R: *Computer Animation: Algorithms and Techniques*, Morgan-Kaufmann, 2001.
[17] Rich, C and Feldman, Y A, Seven Layers of Knowledge Representation and Reasoning in Support of Software Development, *IEEE Trans. Software Eng*, 18(6), pages 451-469, 1992.
[18] Robertson, J, de Quincey, A, Stapleford, T, and Wiggins, G, Real-Time Music Generation for a Virtual Environment, presented at the *ECAI 98 Workshop on AI/ALife and Entertainment*, 1998.
[19] Schank, R C and Abelson, R P, *Scripts, Plans, Goals, and Understanding*, Erlbaum, 1977.
[20] Schank, R C, *Tell Me a Story: Narrative and Intelligence*, Evanston, IL: Northwestern University Press, 1995.
[21] Thompson, R, *Grammar of the Edit*, Focal Press, 1993.
[22] Tomlinson, B, Blumberg, B, and Nain, D, Expressive Autonomous Cinematography for Interactive Virtual Environments, *Proc. of the Fourth Int'l Conf. Autonomous Agents*, 317-324, Barcelona, Spain, June, 2000.
[23] Yaron, T, *Editing Movies*, Israeli Ministry of Culture, in Hebrew, 1995.