Virtual Cinematography of Group Scenes using Hierarchical Lines of Actions

Kaveh Kardan Henri Casanova

kaveh@hawaii.edu henric@hawaii.edu

University of Hawaii*

Abstract

We present an approach to generate shots for 3D computer graphics cinematic sequences from event-based descriptions of scenes of conversations between groups of actors. Our approach creates camera setups using a combination of geometric constraints and aesthetic parameters, while ensuring that the resulting cinematic sequence obeys the heuristics of good cinematography. More specifically, our main contributions are the a method for defining hierarchical lines of action and the identification and use of relevant first principles of cinematography for using these lines of actions. Our approach is more flexible and powerful than those proposed in previous work, mainly because it naturally generalizes to any number of actors in a scene.

CR Categories: I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism; J.5 [Arts and Humanities]

Keywords: computer animation, virtual cinematography, film editing, film grammar, virtual characters, computational cinematics

1 Introduction

The creation of cinematic sequences using 3D computer graphics has many applications. Of particular interest to us is the automated authoring of cinematic sequences for computer games. Both scripted sequences (known as non-interactive scenes) and the playback movies of players' recent actions in the game can benefit from automated cinematic sequence creation. Other applications include computer-assisted storytelling, pre-visualization for films, and 3D chat systems. The effort to create these sequences varies greatly with the visual complexity of the final result, but in all cases, the task of creation remains a time-consuming one. Even for the simplest of sequences, manually setting key frames for camera positions and focal lengths for every setup, and then editing together the resulting rendered footage takes a significant amount of time. In addition, there are cases where it is not possible to manually script camera positions and determine shots and edits ahead of time. One example is the case of games which may want to show a cinematic playback of events based on the actions of the player, rather than pre-scripted input.

There is therefore a clear need for a system which can aid in the creation of cinematic sequences by applying a set of rules derived from the common knowledge of cinematic conventions. Conversely, such a system can be used as a research tool for filmmaking, attempting to "derive" the aesthetic attributes that govern certain historical or genre film styles.

There is a large body of tacitly accepted, yet not very precisely defined, knowledge in the field of cinematography and editing, regarding the rules to follow when filming and editing a scene [Arijon][Katz]. Although heuristics exist, and there are well-accepted conventions in the field as to what "works", this information is not of a sort that has been easily applied to the automated creation of computer-generated cinematic sequences. The very nature of the art form of filmmaking is that every film is different, and there are many "correct" ways to shoot and edit each scene, with very few hard rules which cannot be broken for artistic purposes. Our goal is to encode these heuristics in a software system, in order to allow the generation of cinematic sequences that would be considered correct by filmmakers.

In this paper our main contribution is a generalized camera setup approach that can handle any combination of actor stagings, and that is general enough to handle scenes with 3 or more actors. Most notably, our work abandons special-case approaches to placing cameras in a scene based on the spacial locations or numbers of actors in a scene. Instead, we define hierarchical lines of actions that server as a basis for deriving proper camera setups directly from first principles of cinematography.

2 Previous Work

[Arijon] and [Katz] are valuable texts that explain fundamental concepts of cinematography, and have been used extensively as inspiration by researchers in the field of computer cinematics. Arijon's *film idioms* have been encoded and simulated in various research projects. [Christianson] and [He] have essentially implemented Arijon's idioms regarding camera placement into a software framework for doing virtual cinematography of 3D scenes. Unlike our system, all of these papers explicitly encode Arijon's idioms as formulae in their system. Most related to our work is [Li], which parameterizes these idioms to represent higher

*e-mail: {kaveh | henric}@hawaii.edu

level intent and story considerations. This parameterization provides increase flexibility and represents a marginal departure from the hardcoding of idioms. Our approach differs in that we do not encode idioms, parameterized or not, but instead build our approach on first principles (on which some of the idioms are themselves based). Note that the pervasive issues of geometric camera constraints for framing and occlusion avoidance are well understood and are addressed in [Bares00] and [Pickering]. At a higher level of abstraction, [Friedman] attempts to encode rules of filmmaking genres into an expert system for generating movies.

None of the above works demonstrate results with scenes of more than three actors. This limitation to scenes with two or three actors stems from the fact that these works encode camera setups based on hardcoded, although sometimes parameterized, film idioms as described by Arijon. Our work differs in that we present a generalized approach that relies on the first principles behind Arijon's idioms. We demonstrate that our approach produces valid cinematography for groups of actors of arbitrary size.

Also related to this work are the many research works that have strived to bring cinematographic principles to 3D computer games. An additional challenge which is faced there because cinematography must happen in real time, often responding to arbitrary user interactions. [Amerson] describes a system for bringing filmmaking knowledge to real-time interactive narratives. [Oliveros] applies Arijon's idioms to game cinematics. [Halper] addresses the issues of geometric constraints for cameras in games. In [Hornung] the camera is an agent in the game, employing cinematographic knowledge to aid in storytelling. [Cozic] investigates the use of fixed camera positions, informed by cinematographic knowledge, in games. [Bares99] uses multiple camera setups to follow the action in virtual 3D worlds. In our this work we do not consider real-time cinematography.

3 Software Testbed

In order to test our algorithms, we have developed a software framework based on an existing commercial 3D animation application. Our system is written in Common Lisp, with a socket interface to the Maya 3D application, running on the OS X operating system. Common Lisp calls are translated to MEL (Maya Extension Language) command strings. These MEL command strings are then sent to Maya via a socket, and are executed by Maya. This allows our system to execute any Maya action, such as creating and placing cameras. Values from Maya are return to our software via sockets as well. This software architecture avoids the need to develop a custom 3D graphics engine, and gives us access to the large set of functionality present in an existing commercial 3D animation system.

The two main entities in our 3D scenes are cameras and actors. Cameras have position and orientation, as well a focal length and aspect ratio. Actors are modeled as hierarchical skeletons and can perform simple procedural animation. Currently, these procedures consist of a mouth animation for speaking, and a look-at animation for orienting the head.

These animations are helpful to illustrate the events of the scene, and make the viewer understand why a framing or editing decision was made by the system. In essence they help comprehension of the scene by doing very rudimentary acting, in term allowing us to evaluate the correctness and appropriateness of the generated cinematography.

Note that the resulting animated Maya scene can then be rendered either in hardware or software. This will allow us to extend the capabilities of our software to handle lighting in a natural manner in future work.

4 Design

4.1 Approach

The main steps in the traditional filmmaking pipeline are preproduction, production, and post-production. For our purposes, the roles of interest are those of the director, the cinematographer, and the editor. We represent each of these three roles as modules (fig. 1).

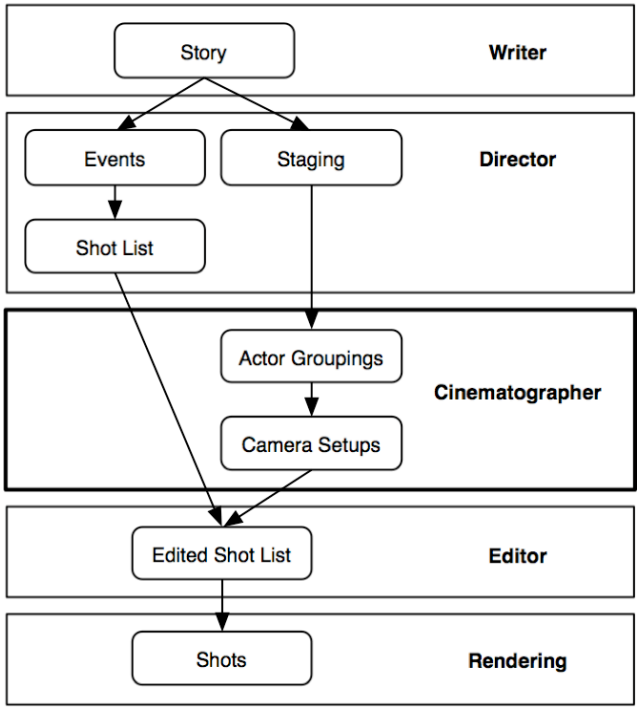

Figure 1. Data flow for cinematics creation.

The director module produces a shot list and places the actors around the stage (also known as blocking). The cinematography system then generates all possible camera setups based on actor grouping and lines of action. Finally, the editor generates an edited shot list, and the sequenced shots are rendered based on the appropriate camera setups.

In this paper, we concentrate on the cinematography step of the process, which (for our purposes) involves the creation of camera setups showing an arbitrary number of actors in a scene. Though on a film set a cinematographer (also known as director of

photography or DP) is as concerned with lighting as with camera setups, our current approach does not address this aspect of the cinematographer's role.

4.2 Scene Representation

Our system encodes data in terms of a scene, which takes place in a single location, with a number of actors. Scenes are made up of discrete events, such as one person talking, or looking at one another. Events are the atomic units of acting in our system.

A scene also contains actors, which are represented by data structures organized as a skeletal animation rig, with rigid body parts, useful for the framing of shots. In the current implementation, actors can perform very simple motions such as mouth movements and head turns, in order to roughly approximate the acting in a conversation scene. In this work we only consider static scenes, in which neither the camera nor the actors are moving.

The creation of events can be handled in a number of ways. The method used for most of the test cases in this paper is manual encoding of existing film clips. Several sample clips have been encoded to serve as test cases for the system. This allows us to compare the generated results with established "correct" results. We also use algorithmically generated test scenes to validate camera setups for a range of popular actor stagings.

Another way of generating events would be to use a higher level scripting system. Such a system would generate scenes of actors and events, perhaps from story generation systems. Another type of software which could generate scenes are *machinima* systems where users create computer-animated movies, often by repurposing recordings of video game scenes. Also, a game engine can generate scenes based on the actions players take in a game. This can serve as a means of providing players with cinematic playbacks of their game play. Finally, instant messaging and chat software could also be used to generate events, based on the text and emotes input by multiple participants.

5 Cinematography

The goal of our cinematography module is to generate a valid camera setup for each shot requested by the editing module. A camera setup is a specific camera position, orientation, tilt, and focal length. Our approach abandons case-specific approaches to camera placement in scenes, and harkens back to the first principles of cinematography for filming conversations. These principles are *look direction* and *camera framing*.

5.1 Look Direction and Line of Action

In a static scene, such as a conversation, the single most important rule in framing the actors is to preserve the look direction of an actor in the scene [Arijon]. This means than an actor looking towards the left of the screen in one shot, should keep looking in the same direction in subsequent shots (fig. 2). This is necessary to establish a consistent spacial continuity between shots, giving the viewer an imaginary location from which to view the scene. This has led to the development of the notion of a *line of action*.

A line of action can be thought of as a line connecting the two actors involved in an event. If the camera always stays on one side of this line, the "look direction" of both actors will be preserved. We refer to the half-plane delimited by this line as the working space of the camera. Camera setups are restricted to stay inside this working space (fig. 3). The resulting visual illusion is that the viewer is positioned somewhere in the working space.

A camera setup is swung off the line of action by an angle specified as an aesthetic parameter. A swing angle of zero would place the camera on the line of action, with the actor directly facing the camera. A swing angle of 90 degrees would result in a profile shot.

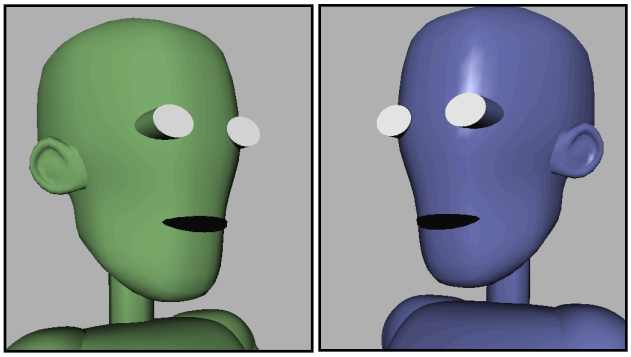

Figure 2. Correct look directions for a 2-person dialog scene.

5.2 Camera Framing

Another first principle of cinematography is the framing of the actors in the shot. Ranging from extreme close-ups to long shots, a well-established aesthetic has evolved over the decades regarding the body parts each framing shows [Arijon][Katz]. Each of these framings serves a certain emotional/aesthetic purpose, either in itself or in conjunction with contrasting framings, both in space and time.

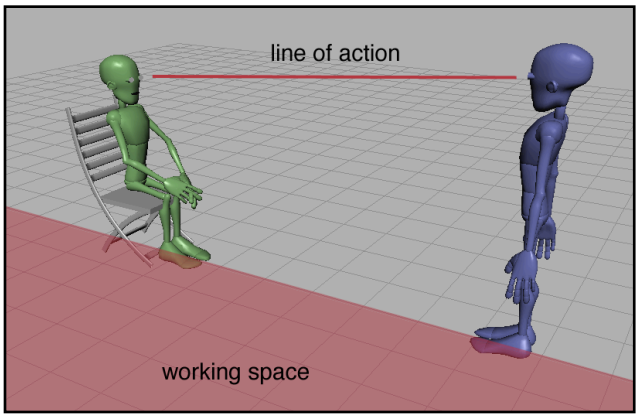

Figure 3. Line of action and working space.

A full-fledge discussion of these properties of framings, while of great interest to filmmakers, is beyond the scope of this paper.

Nevertheless, our approach takes into account aesthetic parameters that can be specified by the user. For instance, the user can specify the amount of *looking room*, i.e., the space in the frame in front of the actor's face (fig. 4). Also, the camera can be tilted up (low angle) or down (high angle).

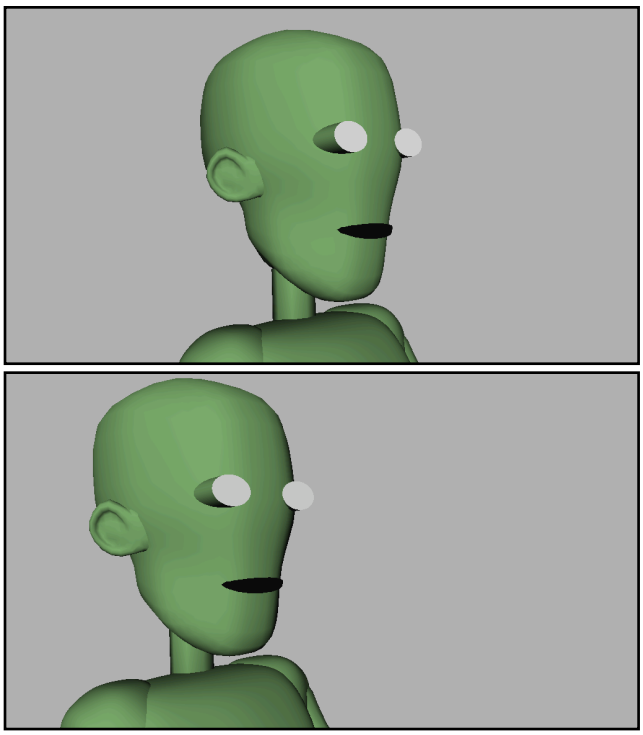

Figure 4. Framing without and with looking room.

At the lowest level of abstraction, the camera system is given a number of geometric shapes it needs to show within the frame. The information about the desired orientation of the camera is derived from the current line-of-action and working space. The aim point of the camera is set to the center of the shapes, and the camera is positioned such that the target shapes are within the field of view. Actors in our system contain a list of body part shapes attached to skeletal bones, which are queried by the camera system to create correct framings (fig 5).

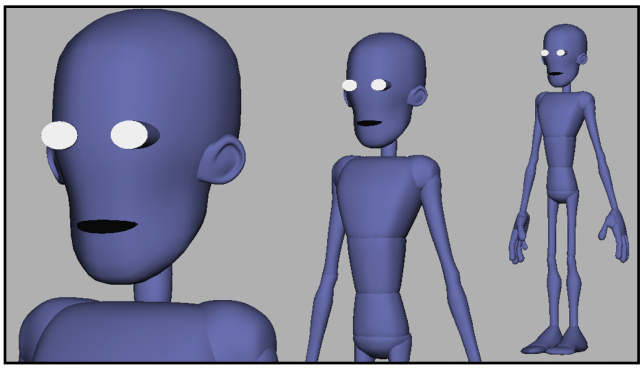

Figure 5. Body framings (from l to r): close-up, medium shot, long shot.

6 Two-actor Scenes

For dialog scenes between two actors, Arijon defines 5 camera setups (fig 6) as well as 4 possible actor stagings. These stagings are: face to face, side by side, right angles, and behind one another. For each of these stagings, the camera positions to achieve the 5 classic setups are discussed. In addition, he addresses the issues involved in camera placements when one or both actors are sitting or lying down, effectively creating a vertical distance between them.

This combination of basic camera positions and actor stagings leads to a large number of possible shot framing permutations, which Arijon discusses. Our approach is, rather than attempt to hardwire solutions for each case, to return to the first principles of look direction and framing, developing a system which can generate all of Arijon's setups in a general manner. By computing a line of action between the heads of the two actors, and a working space, we can derive the 3 types of Arijon's 5 standard camera placements as follows.

An internal shot is a camera setup with a swing angle off the line of action. The swing angle and framing are specified by the user. An external (over the shoulder) shot is the same as an internal shot except that part of the second actor's body has been added to the list of shapes to be framed by the camera. It should also be noted that it is possible to obtain an unintended external shot if the swing angle is not large enough to move the camera away from the second actor. An apex shot is a shot framing both actors, with a 90 degree swing angle.

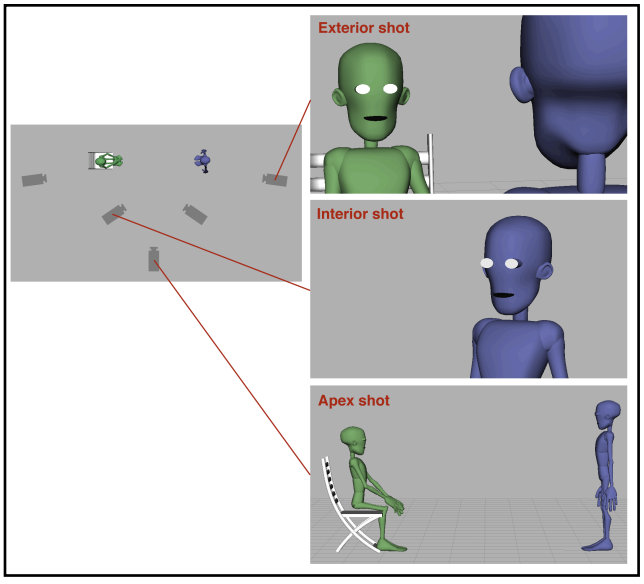

Figure 6. Classic two-person camera setups, from [Arijon].

For two-person dialog scenes, we begin our cinematic sequence with an establishing shot, which is an apex shot, and then cut between inner shots of the actors. The inner shot framings and swing angles depend on user-specified parameters, as modulated by the geometry of the scene and the aspect ratio of the frame.

Our system naturally handles the vertical distance created when one actor is sitting or lying down, since our lines of action are 3D and pass through the head locations of the actors.

7 Group Scenes

Previous work has dealt with the cinematography of three actors in a hardwired manner [Cohen]. These approaches are not readily generalizable to larger groups. Our system uses a novel approach using *hierarchical lines of action.* We generalize two person setups into a system of hierarchical lines of action that provides a methodology for generating correct camera setups for scenes containing an arbitrary number of actors.

There are two reasons for using hierarchical lines of action. One is to ensure consistent camera views for an actor by avoiding small variations in camera placement when the actor addresses groups of actors. The other is based on the notion that not only do individual actor interactions have lines of action, but the scene in general will typically also have a global line of action. Using a hierarchical approach of nested groups with lines of action helps ensure that the lower level camera working spaces are in accordance with the scene's global line of action.

Using independently computed lines of action for each event in a group scene exhibits visual properties which do not match what is seen in traditional filmmaking. Consider the case of one actor speaking, in turn to three other actors facing him (fig 7). Using independent lines of action, this would result in three camera setups for actor A: one each for his 3 lines of action.

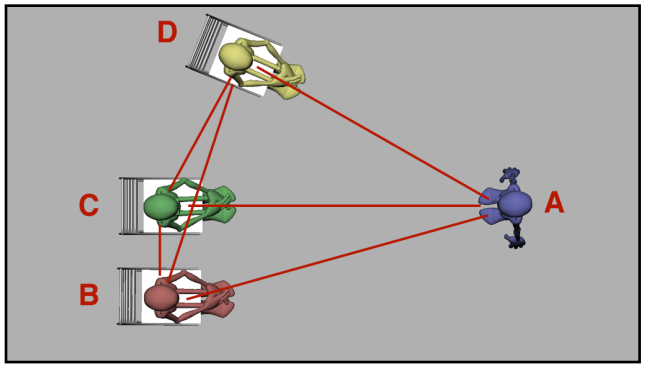

Figure 7. Group scene showing independently computed lines of action.

The swing angles would ensure that the camera maintains a constant angle each line of action, resulting in the illusion that the actor is always facing the same direction (fig. 8). In this example the actor turns his head to face each of the three other actors, from left to right (B, C, then D). As can be seen in the figure, this creates an odd visual illusion, where the actor's head remains fixed and his body turns counter to the direction of the head turn.

This undesirable illusion can be easily avoided by grouping actors in a scene into a hierarchy of groups based on spacial proximity. Lines of action are computed between the groups in this hierarchy, and are used for computing camera setups (fig. 9). This means that there is a single line of action connecting the actor in our example to the group of three actors. This results in a more natural and temporally continuous view of the actor, as seen from a single unified camera setup (fig. 10).

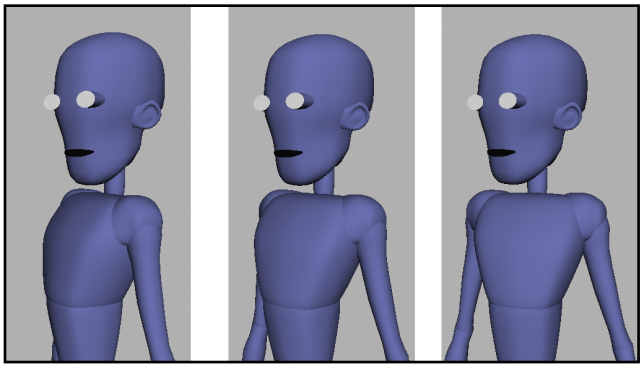

Figure 8. Head turn as viewed by 3 different camera setups as the actor looks respectively at actors B, C, and D. Head appears to stay fixed, and body turns in opposite direction.

Since the groupings are hierarchical, lines of action are traced between the other actors by forming subgroups, and camera setups for their interactions are generated as necessary, based on the events of the scene. Therefore, if actor D looks to actors B and C, the same technique will produce visually correct results.

In effect, our approach reproduces the filming process on a live action set, where a small change in a line of action does not necessitate the moving of the camera to a new setup.

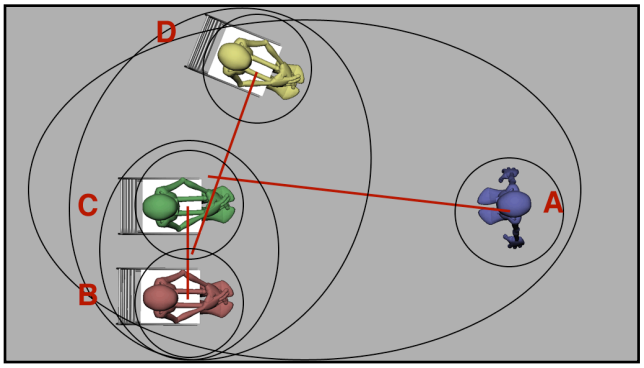

Figure 9. Group scene with actor groupings and hierarchical lines of action.

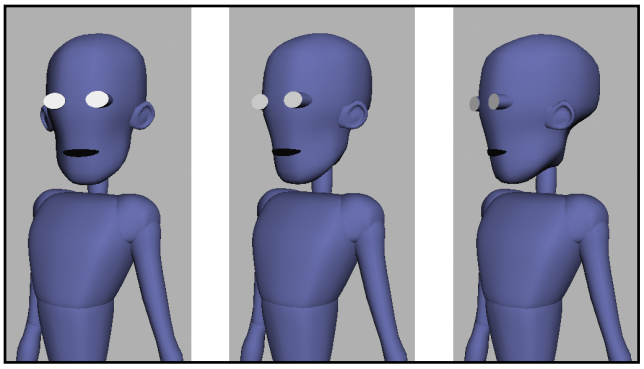

Figure 10. Head turn as viewed by a camera setup based on

actor grouping, as the actor looks respectively at actors B, C, and D. The head appears turn properly, and body remains fixed.

If the three actors in the group have an angular dispersion wider than the swing angle, then the actor interacting with them will rotate his head past the camera, breaking the look direction rule. We avoid this by computing a line of action based on the working space of the camera, selecting a line of action which represents the event with the actor closest to the swing angle. A new swing angle is then computed from this line of action, ensuring the actor never violates the facing rule.

The following example shows a large group scene, with 8 speaking actors (fig 11).

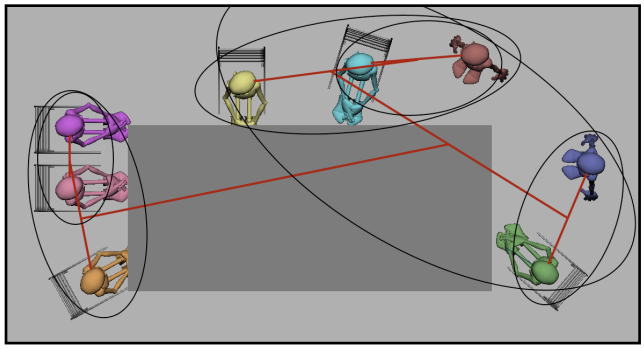

Figure 11. Groupings and lines of action for large dialog scene.

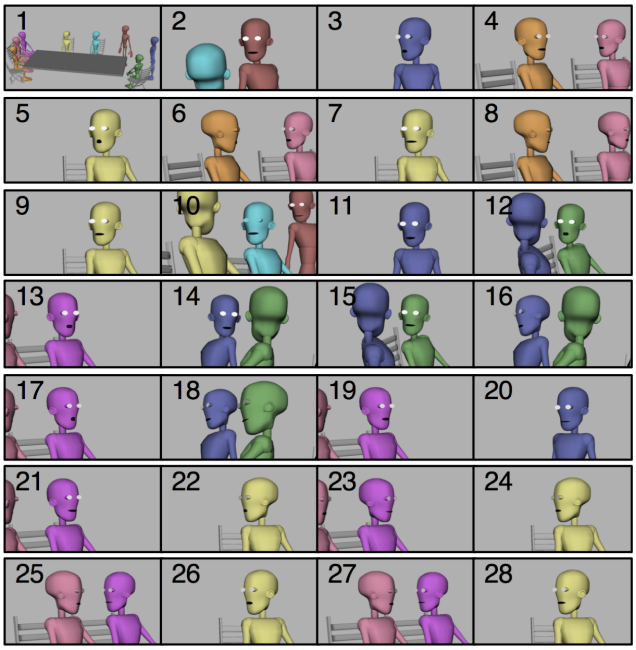

Figure 12. Shots generated for the scene layout shown in fig 11. The generated movie is available at www2.hawaii.edu/~kaveh/Research/Papers/Sandbox2008/.

The sequence of shots generated by the system is shown in fig 12. The first shot is an establishing shot, automatically generated by the system to encompass all actors in the scene. Shots 8 and 20 are reaction shots for which there are no events. These shots are placed in the sequence by the editing system.

8 Results & Evaluation

In order to test our system, we need input scene descriptions encoded as events. In addition, the geometric layout of the scene must be specified, as well as any dialog audio track. Currently, we manually create this input data by extracting the audio track of conversation scenes from films, and analyzing the timings of each actor speaking. These timings are used to create the start and end times for speaking events. The layout of the actors' positions in the scene is created by placing and posing simple 3D models of actors in Maya, based on a visual approximation of the film footage.

There is no single correct solution to depicting a given scene cinematically. Many widely diverse variations can be all correct, and their merits entirely artistic, as opposed to technical. To begin with, the number of potentially "correct" camera setups is unlimited, and depends mostly on the desired emotional response from the viewer, as opposed to acceptable placements or framings. This makes judging the results of the system a very subjective matter. In fact, it is often the films that break one or more accepted cinematographic conventions which stand out as being memorable and innovative.

Therefore, our goal is not to produce artistically interesting cinematic sequences. Instead we aim at producing a "journeyman-like job", and create visually acceptable cinematics that are not objectionable or clearly incorrect.

To evaluate our approach we first generated algorithmically test cases. More specifically, we generated all possible two-actor stagings, as described in [Arijon]. Our system produces valid results for all these cases, thereby validating that the first principles of cinematography implemented in our approach do indeed make it possible to automatically "discover" the film idioms in [Arijon].

The above result is comparable to the results obtained in previous work [Christianson] [He] [Li]. However, the power of our approach becomes clear when moving to scenes involving more actors, and most notably more than 3. We have evaluated our system for a number of short clips of group scenes from feature films, involving from two to eight actors. The generated cinematic sequences from the movie clips, along with the original clips, are available for viewing at www2.hawaii.edu/~kaveh/Research/Papers/Sandbox2008/. These clips clearly demonstrate that our system leads to cinematography that follows the accepted heuristics of the field. Our goal is not to exactly duplicate the source scenes. However, one can plainly see that there are inevitable (and very encouraging) similarities between the live action and computer-generated clips. This observation stems from the fact that both types of clips are built based on the same underlying principles of cinematography.

9 Conclusions

We have proposed an approach for computer-generated cinematography that relies on the concept of hierarchical lines of action and on first principles of cinematography. Our approach

can create valid camera setups for scenes involving any number and spacial arrangement of actors in a static conversation scene. The camera setups allow for shot sequencing resulting in natural visual style, similar to the established styles of traditional filmmaking. In addition, our approach takes into account a number of stylistic parameters that can be tuned by the user.

10 Future Work

This work can be extended in several ways. We plan to add more expressive actor animations, including gestures and body language, in order to improve the emotional content of our sample scenes. Editorial decisions often depend on subtle facial expressions and eye movements, and we would like to be able to include such decision making in our system. We would also like to extend the system to be able to handle dynamic scenes, with both camera and actor motions, as well as actors entering and leaving the scene. This would mean that lines of action would be changing, and actor groupings would be created and deleted over the lifetime of a scene. Often, it is desirable to adjust the positions of actors or objects in order to achieve a more pleasing visual composition with the frame of the image. This sort of staging can be done by allowing the camera system to tweak the geometric layout of the scene on a shot-by-shot basis. This sort of "cheating" is a staple of traditional filmmaking.

A major challenge for computer-generated cinematography is the handling of lighting for visual style and emotional purposes. To the best of our knowledge there has not been any significant work in this field. Additionally, the simulation of depth of field, and focus effects can be used for artistic purposes. Much of the motivation for selecting camera setups and shots is the emotional content of the events in a scene. We plan to add emotion and motivation tags to events, and develop rules based on these to create more compelling cinematic sequences. These tags might specify an actor's state of mind, a relationship between two actors, as well as what the director wants the viewer to feel at a given point. As we move up in abstraction from simple events, we can envision scene data that includes story and plot information, which the cinematography system can reason about, influencing the choices it makes.

A long-term research goal would be to address the inverse problem, which is to use our system to derive the cinematic "rules" that give certain historical or genre films their special look and feel.

References

AMERSON, D. and KIME, S. 2000, Real-time Cinematic Camera Control for Interactive Narratives. In *AAAI'00*.

ARIJON, D. Grammar of the Film Language. Silman-James Press, 1991.

BARES, W.H., and LESTER, J.C. 1999, Intelligent Multi-Shot Visualization Interfaces for Dynamic 3D Worlds. In *Intl. Conf. on Intelligent User Interfaces*, pages 119-126.

BARES, W.H.; THAINIMIT, S.; and McDERMOTT, S., A Model for Constraint-Based Camera Planning. In *Smart Graphics. Papers from the 2000 AAAI Spring Symposium*, pages 84-91, 2000.

CHRISTIANSON, D. B.; ANDERSON, S. E.; He, L.-W.; Salesin, D. H.; Weld, D. S.; and Cohen, M. F. 1996, Declarative camera control for automatic cinematography. In *Proceedings of the Thirteenth National Conference on Artificial Intelligence*, 148-155.

COZIC, L.; DAVIS, S.B.; and JONES, H., Interaction and Expressivity in Video Games: Harnessing the Rhetoric of Film. In *Technologies for Interactive Digital Storytelling and Entertainment,* pages 232-239, 2004.

FRIEDMAN, D.A., and FELDMAN, Y.A., 2004, Knowledge-based cinematography and its applications. In *Proceedings of ECAI*, pages 256-262.

HALPER, N.; HELBING, R.; and STROTHOTTE, T. 2001, A Camera Engine for Computer Games: Managing the Trade-Off Between Constraint Satisfaction and Frame Coherence, In *Proc. Eurographics 2001.*

HE, L.; COHEN, M.; and SALESIN, D., The Virtual Cinematographer: A Paradigm for Automatic Real-time Camera Control and Directing, *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques*, p.217-224, 1996.

KATZ, S., Film Directing: Shot by Shot: Visualizing from Concept to Screen, Michael Wiese Productions, 1991.

LI, T.Y., and XIAO, X.Y., An Interactive Camera Planning System for Automatic Cinematographer. In *Conference on Multimedia Modelin*g, pages 310-315, 2005.

OLIVEROS, D.A.M. 2004. Intelligent Cinematic Camera for 3D Games, MSc. Thesis, University of Technology, Sydney Australia.

PICKERING, J,H., 2002, Intelligent Camera Planning for Computer Graphics, PhD. Thesis, University of York, York, UK.