Cinematic Camera Control in 3D Computer Games

Ting-Chieh Lin
Department of Computer and Information Science, National Chiao Tung University
1001 Ta-Hsueh Rd,
30010, Hsinchu, Taiwan.
gis90525@cis.nctu.edu.tw

Zen-Chung Shih
Department of Computer and Information Science, National Chiao Tung University
1001 Ta-Hsueh Rd,
30010, Hsinchu, Taiwan.
zcshih@cis.nctu.edu.tw

Yu-Ting Tsai
Department of Computer and Information Science, National Chiao Tung University
1001 Ta-Hsueh Rd,
30010, Hsinchu, Taiwan.
gis91812@cis.nctu.edu.tw

ABSTRACT

Good camera control and planning techniques in 3D computer games can give players deeper feelings about atmosphere of games. However, most modern computer games use fixed point-of-view techniques to control the scene camera. We believe applications of some basic cinematic camera planning methods in computer games will enrich player's gaming experience. In this paper, we discuss the principles of camera control in real-time 3D computer games and concepts of cinematography, encapsulating both of them into camera modules. We also present a system for event-driven camera modules selection which is connected to the interactive environment in 3D computer games. Furthermore, some methods to solve the conflict between camera placement and frame coherence are also described. Finally, a practical method to avoid obstruction is used in this system to consider the visibility of important objects.

Keywords

Computer Graphics, 3D Computer Games, Cinematic Camera Control, Constraint-Based Camera Planning.

1. INTRODUCTION

In the trend of current computer games, good storytelling techniques become more and more important since designers wish players to be able to feel the atmosphere deeply while playing the game. Like motion pictures, camera techniques can enhance the viewer's experience. Current computer games often use a fixed point of view or a first-person view. These camera settings are easy to design and control but just offering a view for player to see the scene. Film industry has many experiences in camera techniques and already developed many heuristic methods and principles. However, there are still some differences while integrating them into interactive computer games. The camera in games needs to move automatically to a specific position based on cinematography without player's order. It also has to avoid bothering players' control feelings. Therefore, we implement a game to show how the cinematic camera control improves the effect of games.

WSCG SHORT Communication papers proceedings *WSCG'2004, February 2-6, 2004, Plzen, Czech Republic.*

In this paper, we propose a mechanism of camera control in 3D computer games. The system can automatically direct the camera based on some cinematic heuristics. Because a game can usually be de-composed to several specific scenes which often occur in motion pictures, we use a sequence of shots similar to cinematic heuristics to describe the camera behavior in a scene.

The goal of our system is to integrate these cinematic camera techniques into a game system and let the camera control undisturbed. Therefore, players can feel that the atmosphere the story wants to reveal more easily. Some well concepts of camera control from other researchers, such as camera module [Coh96a][Dru94a][Dru95a][Hal01a], and constraint solver [Bar00a][Dru94a][Dru95a] [Hal01a], are used. Compared to other researches about cinematic camera control or camera engines for games, our method is similar to [Hal01a] but we devote to the implementation of a playable game with camera control module and demonstrate how the cinematic camera control improves the effect of a game. Therefore, some extensions are provided to make the system more robust. The occlusion can be predicted and a new camera position is generated by the system. When the most proper camera setting is not available, an expedient one will be generated.

This paper is organized as follows. In Section 2, we review the previous related researches of camera control and take a look at the methods we reference. Section 3 describes the heuristics of camera techniques and principles in cinematography. We discuss the architecture of our system, explain each component, present a demonstrating game integrated with the cinematic camera control system, and analyze the detail techniques of implementation detail in Section 4. Finally, we make conclusions and discuss our work in Section 5.

2. RELATED WORKS

There are various areas in which related work has been explored. The camera settings of current games may be important references. They have provided players useful interfaces to play games. Some work assists users in directly manipulating camera, but the camera process should be invisible to the user in games. Allowing communication of visual goals should be enabled so that players could know what happens in the game. Another characteristic of the camera in game is that it can only be directed when the director engine knows what is likely to happen next. Its relationship to game context cannot be ignored.

Most real-time games use specialized camera routines. However, one kind of camera routines cannot handle variously visual goals. Here we discuss some famous examples. *Tomb Raider* lets player control a female adventurer. The camera follows her back with the ability to explore surroundings. Many 3D action or RPG games, such as *Phantasy Star Online* and *Ultima IX*, adapt this routine. This camera routine makes navigation passably easy, but emotional effects and visual goals are ignored. *Biohazard* divides the whole scene into several independent blocks and places one camera in each block. The game selects a camera according to the block in which the character is. While the character moves to another block, the camera corresponding to the block replaces the old one. This creates a cinematic style but an inflexible one. Moreover, this faces a problem that when a player wants his character interact with an object or character in a block different from the one your character in, for example, shooting a distant monster, the camera will not be able to show the target. An immovable camera also ignores the visibility problem such that important visual goals are sometimes occluded. *Doom*, *Quake*, and *Half-Life* are "first-person shooter" type games. The entire screen shows a straight-ahead view of what you are seeing. This first-person view allows player having a large freedom to control camera. However, it can't improve the effect of storytelling. *Ultima Online* offers top-down camera style as many adventure games use. As the player's party wanders around the world, the camera watches them from high above. This provides easy navigation but is expressionless.

[Bli88a] described the mathematics for defining low-level camera parameters. Given the geometry of the scene and the desire actor placements, some algebra techniques can decide the proper values of camera parameters. [Dru94a] and [Dru95a] presented CamDroid system which is able to assist in developing camera frameworks for widely disparate types of graphical environments. [Bar00a] uses a constraint solver to find a solution to various user-imposed requirements of camera viewing. Besides, [Hal00a] uses a genetic algorithm to find a good camera placement. [Hal01a] proposed a camera engine for computer games, and pay particular attention to a trade-off between constraint satisfaction and frame-to-frame coherence. An algorithm for consideration of visibility is also presented by finding potential visibility regions on depth map obtained from the actor's view.

The Virtual Cinematographer [Coh96a] uses the concept of idiom, and treats sequences of shots as a finite state machine. A state represents a shot, and an arc represents the available shot transition. [Haw02a] and [Haw02b] introduced the availability of cinematic camera in games. Some techniques of obtaining visual goals are provided and he mentioned the work that the virtual director and editor should do, such as how to determine the interesting shot. [Tom00a] proposed a behavior-based autonomous cinematography system by encoding the camera as a creature. This creature has motivational desires, controlling camera and lighting in order to augment emotional content. There is an animation tool providing autonomous cinematography [Ken02a]. This planning system employs concepts of expert system. Cinematographic knowledge is stored in a database and this system will choose a proper camera pattern according to the action description given by users. Besides, Cognitive Modeling [Fun99a] is also an AI method which has ability to reason the camera position from provided knowledge.

3. CINEMATIC CAMERA TECHNIQUES

The camera does so much work more than just taking a picture of a situation. It gives audience certain impression of a subject and its surroundings. Shooting it one way and the subject can appear important, dominating its environment. Shot from another angle, the subject become quite incidental. Therefore, we need to classify shots in order to organize and arrange how we are going to shoot any situation. Furthermore, knowledge of adjusting

composition for appropriate effect and principles to make the visual effect consistent is also important.

Cinematography Principles

Filmmakers have found numerous heuristics for selecting good shots and informally specified constraints on avoidance of inappropriate shots for scenes. We surveyed and picked some important principles as considerable factors of camera control. These principles are:

- **Do not cross the line:** The exact line of action depends on the number of actors in the scene. For one actor the line of action is the vector in the direction the actor is facing. For two actors the line of action is the connected line between two actors. Once an initial shot is taken from the left or right side of the line of action, subsequent shots should remain on the same side. This rule ensures the direction of the actor's motion being clear.
- **Avoid jump cut:** There should be a marked difference in composition, such as size, view, or number of actors, between the two shot crossed by a cut. An inappropriate cut creates a jerky, sloppy effect.
- **Use establishing shots:** Establish a scene before moving to close shots. When there is a new development in the scene, the situation must be re-established.
- **Let the actor lead:** The actor initiates all movement, and then the camera follow it. Therefore, camera setup should be considered before the actor.
- **Break movement:** A scene illustrating motion should be broken into two shots. When the actor appears to move across the middle of the screen, a shot is often cut to another shot.

4. CINEMATIC CAMERA CONTROL SYSTEM

The architecture of a computer game with the cinematic camera control system is illustrated in Figure 1. Beside of the camera control system, the other two parts consists of the real-time application and the renderer. The real-time application supplies the renderer with game content, including any static geometry, material properties, lights, and the story, that will affect the environmental parameters. At each time tick, the following events occur in this system:

- The real-time application sends the camera control system a list of parameters which are important to the decision making of the camera settings in that tick. According to the game context and significant scenes, these parameters may be an actor's position, orientation, equipment, action, and user input, etc.
- The cinematic camera control system uses the parameters from the real-time application plus the existent state of the camera settings (e.g. how long the current shot has lasted and the current camera settings) to produce an appropriate camera specification and then output it to the renderer.
- The renderer collects the animation parameters and description of the current environment sent by the application, and the camera specification and acting hints sent by the camera control system.

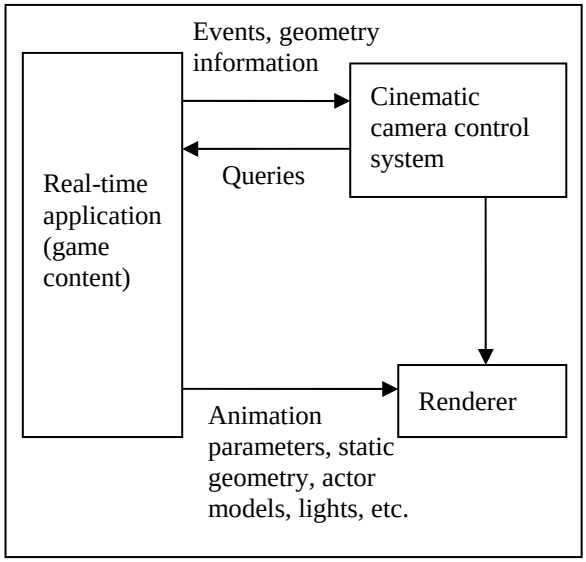

Figure 1. A game with cinematic control system.

The camera control system (Figure 2) has two components: camera modules and descriptions of shots. Camera modules implement the camera placements and transitions. Descriptions of shots are sequences of shots according to the formula of cinematography used in scenes. These descriptions are organized hierarchically, from more general ones near the top. Each shot in the descriptions of shots is corresponding to a camera module. Because camera modules can solve the constraints defined by a shot, the final camera settings are obtained.

Camera Modules

A camera module represents an encapsulation of the constraints and a transformation of user input and parameters from application to desired shots. As shown in Figure 3, a generic camera module contains three components: the local camera state, analyzer, and constraint lists.

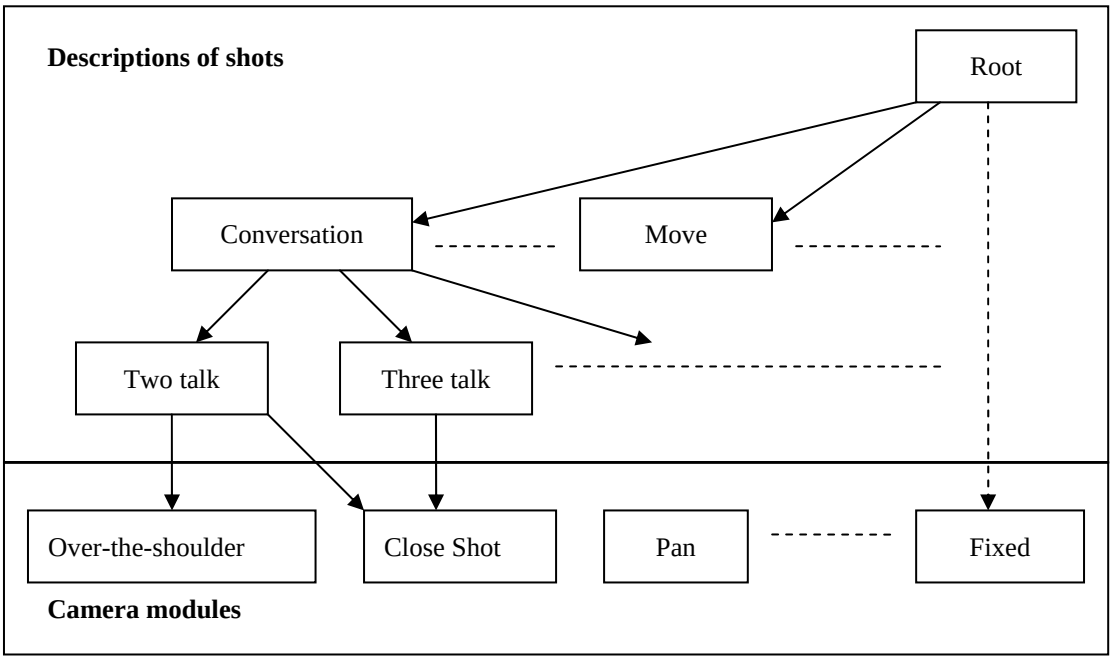

Figure 2. Cinematic camera control system.

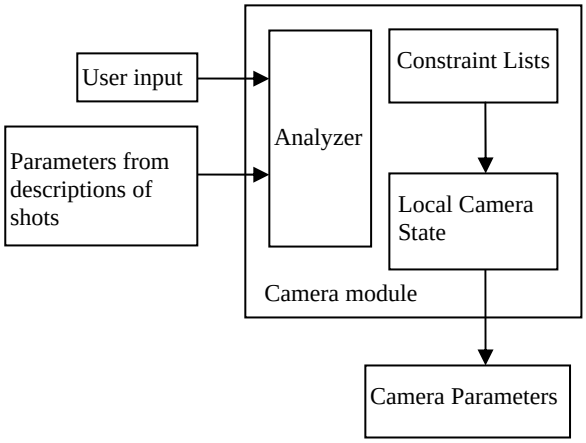

Figure 3. A generic camera module.

- **The local camera state:** This module always contains information about camera position, view normal, up vector, and field of view. It can also contain parameters for deriving camera parameters, such as value of time or other local information specific to operations of the module.
- **Analyzer:** The analyzer can add information to constraints or camera state to adjust the shot generated by this module.
- **Constraint lists:** The constraint lists contain constraints to be satisfied during the period when the module is active. Details are described in the next sub-section.

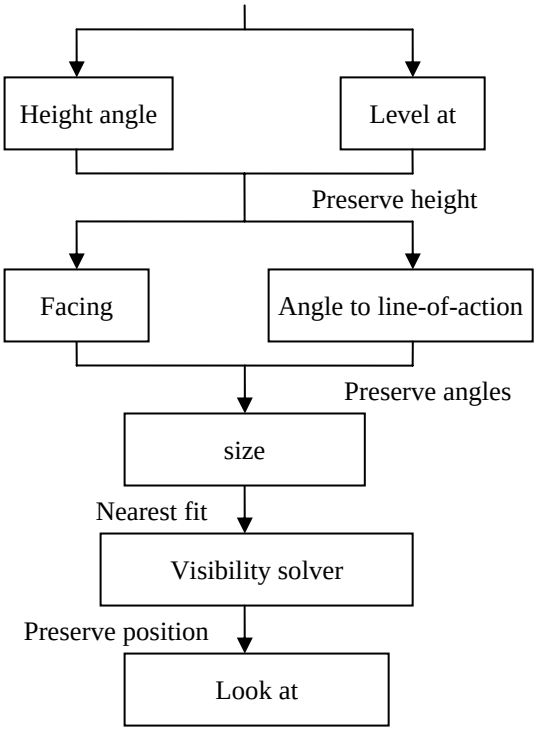

Figure 4. Constraint solver.

Constraint Lists

In our system, the constraint lists can be viewed as a black box that produces values for some degrees of freedom of the camera. We adapt the constraint

solver (Figure 4) described in [Hal01a] to solve the constraints. Experience in previous work provided the specifications of constraints that can sufficiently define a camera position and viewing direction as well as a desired shot.

- **Level at:** The camera should offset at a certain height relative to the object.
- **Angle to line-of-action:** This constraint describes the angle relative to the line-of-action.
- **Facing:** Each object has a vector which defines the front of the object. By setting the desired viewing angle relative to object's front direction can create many useful shots.
- **Size:** This constraint controls the camera's distance to the target.
- **Height angle:** It instructs the camera to watch the target at a specified height angle.
- **View at angle (X and Y):** This one determines the position of the target on the screen. X angles, other than 0.0, move the projection of the target to the left or right side of screen; the Y angle is used to push the target toward the top or the bottom.
- **Visibility:** The target should be visible to the viewer, such that obstructions are avoided.

Frame Coherence

In many conditions, a sudden jump cut will annoy players while playing the game. A jump cut that occurs during a hard fight will totally spoil the game playing experience. Moreover, a camera sometimes needs to cross the line of action for some reasons, and the only way to keep the viewer's sense of direction is to move the camera smoothly. Therefore, the analyzer in camera modules decides whether the camera should move smoothly by collected parameters, such as the camera module used previously. If the frame coherence is necessary, position and angle of the camera are interpolated in each time tick. Otherwise, the camera parameters will be replaced by active camera module immediately.

Obstruction Avoidance

In our system, we employ a less expensive technique to determine if a shot is obstructed. We use a line-intersection test that uses a line going from the camera to the bounding sphere of the actor. Besides, as mentioned in previous works, predictive camera planning [Hal01a] is necessary for frame-to-frame coherence and smooth camera movement. Therefore, we replace the line-intersection test with a cylinder when there is a demand on smooth camera movement. By using a cylinder for intersection test, obstruction can be detected before it happens. Furthermore, along the way by which the object has passed, there should be an area in which the camera can be placed without being occluded. To orientate camera to the inverse direction that the object recently headed to is a solution to avoid occlusion. Figure 5 illustrates one of the situations of obstruction avoidance by our method.

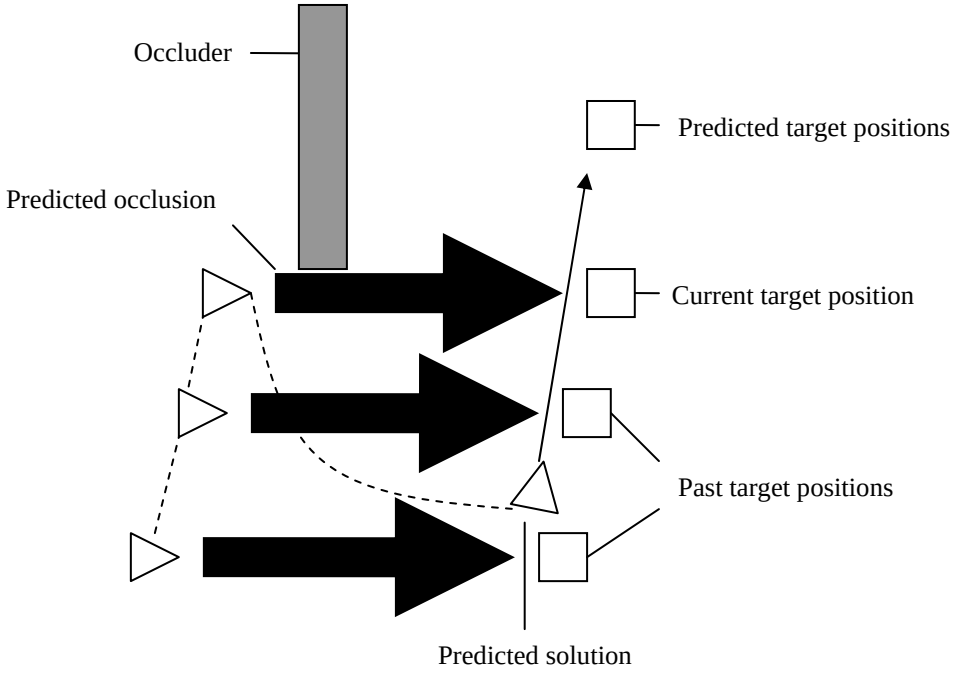

Figure 5. Obstruction avoidance.

Game Description

In our game system, players can explore the world, listen to a non-player character's words, pick up weapons, touch objects, and fight enemies with a gun or a crowbar. The player has to control his character to escape the danger area. For this purpose, the player have to defeat enemies who will try to kill the player, get information or assistance from useful people, activate some switches to solve puzzles. These event scenes are suitable for a description of shots to capture, and our mechanism will perform this duty. We classify possible conditions in the game to four states as shown in Figure 6.

Each state is an event scene corresponding to a description of shots in our system. There is also a finite state machine which describes how to capture the event scene. Transitions between states are driven by specific events and these events are defined in the game content explicitly. The description of shots for exploration is shown in Figure 7. Figure 8 illustrates the finite state machine for a scene of conversation. In the condition of gun fight and close fight, we use simply one camera module for each state to handle the task. More details about camera modules we use will be discussed in the next sub-section.

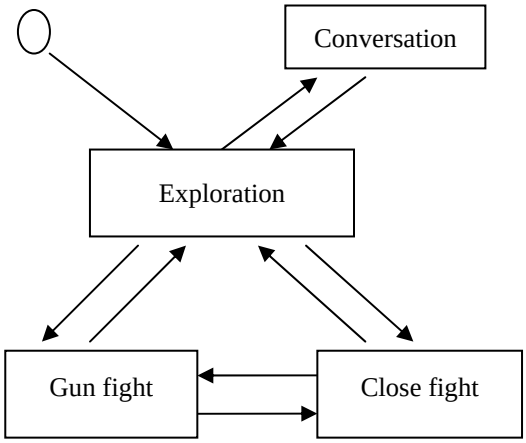

Figure 6. States in the game.

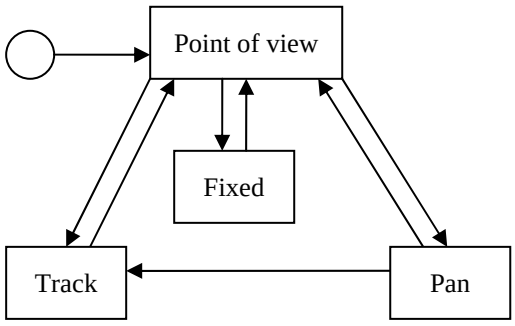

Figure 7. Description of shots for exploration.

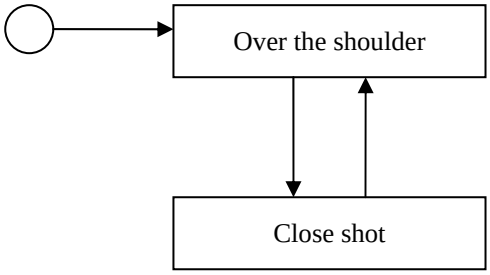

Figure 8. Description of shots for conversation.

Results

We implemented our system with C++ language and run it on a PC platform with Pentium 4 2133MHz CPU and 256 MB RAM. It includes eight camera modules to handle all necessary situations, including "fixed", "track", "pan", "point of view", "over the shoulder", "close shot", "gun fight", and "close fight". The function of each module is described in the following paragraphs:

- **Fixed:** A fixed camera is placed in the scene without movement or rotation. It is a good choice to reveal important information or handle a small space.
- **Track:** The camera focuses on the target and is kept a distance away from the target. However, the player cannot control the camera. This camera module is suitable for storytelling because it is often used in cinematography. In the situation that the "point of view" module cannot handle (e.g. obstruction occurs), the "track" module is sometimes used.
- **Pan:** The camera is rotated on a side-to-die basis. It is a well-known cinematic camera control technique. We can use it for cinematic effects.
- **Point of view** (Figure 9a)**:** The camera follows the player's back with the ability to explore surroundings. Because the camera is toward the direction that the character faces to, the player can observe the environment easily.
- **Over the shoulder** (Figure 9b)**:** It is a good composition to capture the interaction of two people.
- **Close shot** (Figure 9c)**:** It can emphasize the character in action. We often choose it to be one shot for a conversation.
- **Gun fight** (Figure 9d and Figure 9e)**:** To capture a scene of gun fight, we need to consider a handy control and make the target visible. A focus on target and the player's character appearance on screen are necessary conditions. Frame-coherence

and the principle, “do not cross the line”, should be followed because the can keep the sense of direction and avoid jump cuts.

- **Close fight** (Figure 9f)**:** In order to enhance the effect which emphasizes both fighters, a close fight should be featured by a close camera. Like “gun fight”, frame coherence is necessary, and they should also be visible on the screen.

5. CONCLUSIONS AND FUTURE WORKS

In this paper, we propose a cinematic camera control mechanism which is suitable for 3D computer games involving virtual characters and stories. The control mechanism collects and analyses information from the game content and automatically directs the camera to capture event scenes. We encapsulate cinematic camera techniques into camera modules and encode the procedure of shooting a scene into a description of shots which is actually a finite state machine. Therefore, the system can also assist designers to add effects of cinematic camera control by providing camera modules and descriptions of shots. Moreover, we described a procedure to solve constraints and discussed how it works in a camera module. The concept of frame-coherence is integrated into our system for smooth camera movement and we provide a method to predict occlusion and avoid it in some cases. With all these features, this camera control module can automatically generate shots and arrange these shots to provide a cinematic effect and suitable for game playing.

So far, our experiments have opened several topics for future research:

- Our system is based on *Half-Life*’s game engine at present time. To generalize our system to fit any graphic engine can make it more extensive and useful for game designers.
- More camera techniques and more effects can be integrated into the system. Therefore, techniques for characters’ emotional states can extend the ability of our system.
- Camera planning becomes very complex when there are too many conditions which need to be considered. Even though the system offers encapsulated camera modules and descriptions of shots, how to use them is still a difficult task. Expert system might be a solution to help designers to use or build the components in the camera control module efficiently by encoding cinematography knowledge into database and transforming cinematic principles into rules.

6. REFERENCES

[Bar00a] Bares , W.H., Thainmit, S., and McDermott, S. A model for constraint-based camera planning. In Smart Graphics: Papers from the AAAI Spring Symposium (Stanford, March 20-22, 2000), Menlo Park, AAAI Press, pp. 84-91, 2000.

[Bli88a] Blinn, J.F. Jim Blinn’s corner: Where am I? What am I looking at? IEEE Computer Graphics and Applications 8, No.4, pp.76-81, Jul., 1988.

[Coh96a] He, L., Cohen, M.F., and Salesin, F.H. The virtual cinematographer: A paradigm for automatic real-time camera control and directing. SIGGRAPH 96 Conference Proceedings, Annual Conference Series, ACM SIGGRAPH, Addison Wesley, pp. 217-224, Aug., 1996.

[Dru94a] Drucker, S. Intelligent camera control for graphical environments. PhD thesis, MIT Media Lab, 1994.

[Dru95a] Drucker, S.M., and Zelter, D. CamDroid: A system for implementing intelligent camera control. 1995 Symposium on Interactive 3D Graphics, ACM SIGGRAPH, pp. 139-144, Apr., 1995.

[Fun99a] Funge, J. Cognitive modeling for computer games. AAAI Spring Symposium on Artificial Intelligence and Computer Games, Stanford University, Mar. 22-24, 1999.

[Hal00a] Halper, N., and Olivier, P. CAMPLAN: A camera planning agent. In Smart Graphics: Papers from the AAAI Spring Symposium (Stanford, March 20-22, 2000), Menlo Park, AAAI Press, pp. 92-100, 2000.

[Hal01a] Halper, N., Helbing, R., and Strothotte, T. A camera engine for computer games: Managing the trade-off between constraint satisfaction and frame coherence. Proceedings of EUROGRAPHICS 2001.

[Haw02a] Hawkins, B. Creating and event-driven cinematic camera, part one. Game Developer Magazine, pp. 34-40, Oct., 2002.

[Haw02b] Hawkins B. Creating and event-driven cinematic camera, part two. Game Developer Magazine, pp. 36-39, Nov., 2002.

[Ken02a] Kennedy, K. and Mercer, R.E. Planning animation cinematography and shot structure to communicate theme and mood. Proceedings of the 2nd international symposium on Smart Graphics (SMARTGRAPH '02), Hawthorne, NY, USA, Jun. 11-13, 2002.

[Tom00a] Tomlinson, B., Blumberg, B., and Nain, D. Expressive autonomous cinematography for interactive virtual environments. Proceedings of the 4th International Conference on Autonomous Agents (AGENTS-2000), NY, ACM Press, pp. 317-324, Jun. 3-7, 2000.

(a) Module: point of view

Description of shots: exploration

(b) Module: over the shoulder

Description of shots: conversation

(c) Module: close shot

Description of shots: conversation

(d) Module: gun fight

Description of shots: gun-fight

(e) Module: gun fight

Description of shots: gun-fight

(f) Module: close fight

Description of shots: close-fight

Figure 9. Camera modules.