Automatic Identification and Generation of Highlight Cinematics for 3D Games

Mike Dominguez
FactSet Research Systems
601 Merritt 7
Norwalk, CT 06851
mdominguez@factset.com

R. Michael Young
Department of Computer Science
NC State University
Campus Box 8206
Raleigh, NC, 27695-8206
young@csc.ncsu.edu

Stephen Roller
Department of Computer Science
University of Texas
Austin, TX
roller@cs.utexas.edu

ABSTRACT

Online multiplayer gaming has emerged as a popular form of entertainment. During these games, the players' main focus is usually placed on achieving the objectives that must be completed to win the game. Over the course of the game, however, their interactions may result in interesting emergent narratives that go unnoticed. Afterthought is a system that monitors player activity, recognizes instances of story elements in gameplay and renders cinematic highlights of the story-oriented game play, allowing players to view these emergent narratives after completing their gameplay session. This paper outlines Afterthought's design at a high level.

Categories and Subject Descriptors

K.8.0 [**Personal Computing**]: General,games

General Terms

Machinima, automated cinematography, 3D camera control

Keywords

Cut-scene generation, highlight reels

1. INTRODUCTION

Over the course of an online multiplayer gameplay session, interesting narratives can emerge through the interactions between players. In many cases, these narratives may go unnoticed by the participants, possibly due to the players' primary focus on completing the game's objectives or their lack of complete knowledge of the dynamic state of the virtual world. In order to experience these narratives, players need a method of reviewing past actions that took place during gameplay.

Many commercial games provide methods to analyze past moments of gameplay. A typical player-controlled replay system gives users free control of the viewpoint, putting the burden on the player to find the most interesting moments and to position the camera such that they can adequately see the action taking place. Some replay systems give users the ability to pause, rewind, and fast forward through the action [12]. Others include more advanced features such as clipping a replay to a certain segment and the integration of a mechanism to share clips with other players [13]. Some games can automatically generate a video showcasing a highlight, however they may be limited to featuring only the most obviously important moments, such as capturing a flag or scoring a goal [9].

While these existing systems provide many useful features, they lack the ability to both intelligently recognize complex interactions between players during gameplay and subsequently effectively create a video illustrating them.

When a viewer experiences a visual narrative, the way that the actions of the story are presented to them can dramatically impact their perception and understanding of events. Filmmakers have taken advantage of this since the advent of the medium in order to communicate details about the story to their audience (e.g. [2]).

Careful crafting of the presentation of a sequence of events can be used to affect an audience's emotions and their understanding of the events taking place. By allowing for more expressive cinematography to be used during replays of past gameplay, the events that take place can be more effectively communicated to the audience. This paper presents *Afterthought*, a system designed to create videos that depict emergent narratives that occur during gameplay. The problems addressed in the creation of this system include a) the logging all of the significant actions that take place in a gameplay session, b) recognizing the narratives that occur in game logs, c) determining the most effective way of filming these narratives, and d) rendering the video file.

Within Afterthought, as a multiplayer game executes, actions performed by the players will trigger log messages that are sent from a modified game to a module that matches these messages against a set of pre-defined finite state machines corresponding to the description of narrative patterns. The system keeps track of all partial matches on the FSMs in parallel, noting when each FSM reaches a terminal state (and thus corresponds to an entirely matched narrative pattern). Information about these matched patterns are then paired with the cinematographic instructions for filming them. A subsystem responsible for re-playing

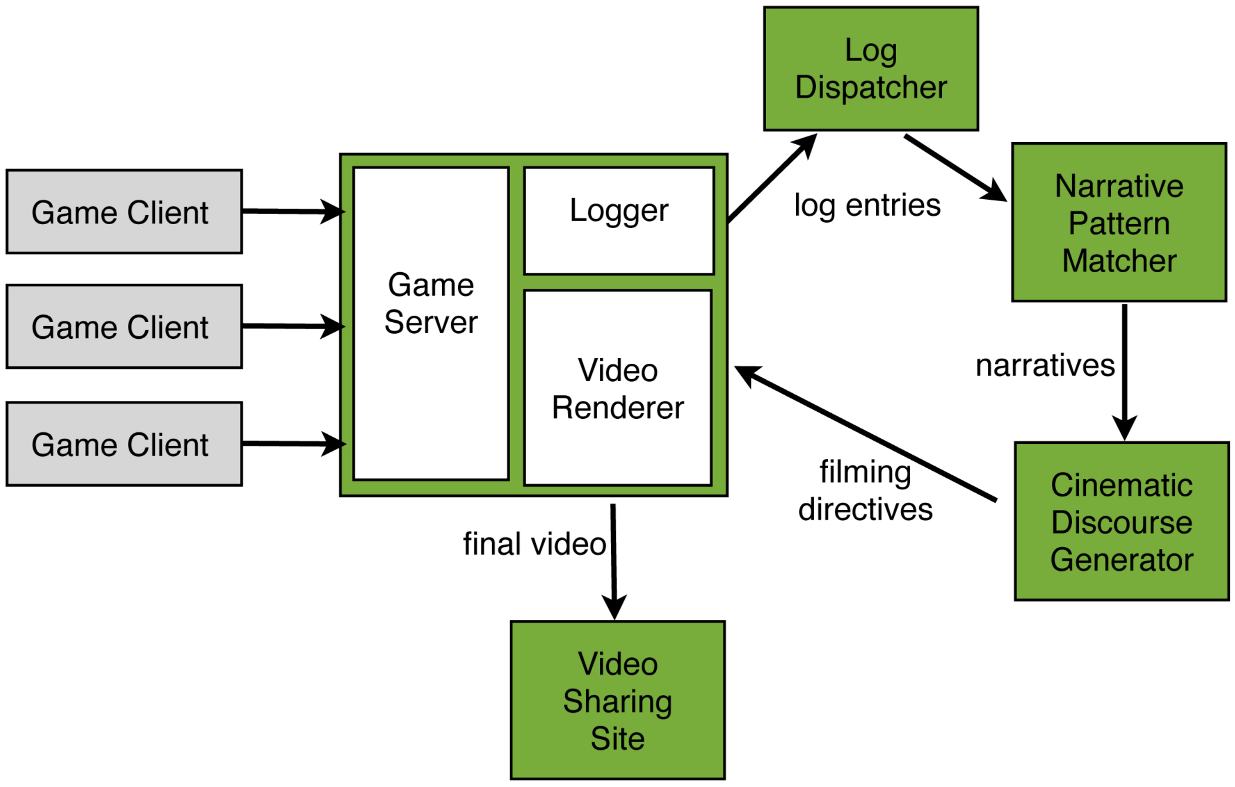

Figure 1: Afterthought's architecture.

and recording the gameplay processes these instructions to generate the final video. The resulting video is automatically encoded into a common video codec and uploaded by the system to a video sharing site for access by users. This system has been fully implemented for the Capture the Flag game type in the first-person shooter *Unreal Tournament 3* [8]. Space limitations prevent us from describing the details of the implementation at length.

2. RELATED WORK

A number of systems have been developed to analyze the logs of gameplay for both virtual and live-action games. Cheong et. al. developed a system called ViGLS (Visualization of Game Log Summaries) that automatically creates video summaries of past gameplay [4]. Friedman et. al. also worked on creating video summarizations based on logs of activity in virtual environments [7]. In their approach, the actions in a log are compared against predefined interesting actions, characters, and series of actions.

Halper and Masuch created a system that can extract and evaluate action scenes in computer games [10]. They introduced an evaluation function used to determine the interest level at a particular time during gameplay. The interest level depends on a number of in-game variables, including changes in a player's health, amount of ammunition, rate of fire, etc. Using this interest function, they divide the game session's timeline into distinct scenes of action. Their goal was to utilize these techniques for a spectator mode, in which outside observers view the gameplay live, as well as for gameplay summarization. For the live coverage problem, they discuss a number of methods used to most effectively place a spectator's viewpoint at each moment in time.

While the focus of the work reported here is not on automated cinematography, a significant and growing amount of work has been done on the automatic generation of cinematic sequences [6, 3, 5, 1, 11]. With the exception of work by Jhala and Young [11], most of this work focuses on the lower level geometric constraints that bound a shot rather than the composition of narrative sequences that tell stories through the camera.

3. OVERVIEW AND DISCUSSION

Afterthought consists of five components - the modified game to be used with the system, the log dispatcher, the narrative pattern matcher, the cinematic discourse generator, and the video renderer. These components are shown in Figure 1. A host system initializing the multiplayer match acts as the server for the game session. The systems of the other players in the match connect to the server as clients prior to the beginning of the game. The log dispatcher also connects to the game server over a socket. As the game is being played, actions performed by the players trigger log messages that are sent from the game server to the log dispatcher. The log dispatcher then forwards the messages to the narrative pattern matcher and to the cinematic discourse generator. The narrative pattern matcher finds every series of gameplay actions that match the narrative patterns it has been provided. At the conclusion of the game match, the narrative pattern matcher sends these sequences, called *narratives*, to the cinematic discourse generator. The cinematic discourse generator then selects one or more narratives from this collection to film and determines how they should be shot and edited. The resulting instructions for filming are then sent to the video renderer, which replays the game session, records the videos and uploads them to a video-sharing web site.

For this work, a mod that provided for integration with Afterthought was created using *Unreal Tournament 3* (UT3). Simplified scripts that control the camera for filming are defined in UT3's scripting language for each shot type created in the cinematic discourse generator. These scripts are parameterized so that each shot will demonstrate sensitivity to context during its filming (e.g., simple occlusion avoidance, optimal distance of the camera from the principal actors, relative height, rotation speed and translation speed of the camera relative to the action).

Afterthought identifies complex interactions between players in 3D games and generates videos showcasing them. It accomplishes this through the use of a modular system that takes as input a set of narrative pattern descriptions, preprocesses them for use in a pattern matcher, and then provides matched narratives to a rendering system that ranks action sequences and selects shot constraints for rendering during a replay session in-game.

A preliminary evaluation was performed to determine the effectiveness of Afterthought's dynamic camerawork during a replay. The data gathered provided statistically significant support for the claims that the custom camerawork provided for more humorous and more dramatic cinematics in comparison to the standard third-person view. Qualitative text responses from subjects included both positive remarks and constructive criticism indicating where improvements need to be made. Unfortunately, space limitations preclude a full discussion of the experiment's design and results. We can say, however, that the results of our study have prompted us to consider several areas for extension of Afterthought. Increasing the run-time performance of the pattern matcher would allow Afterthought to identify patterns in real-time during game play, decreasing the time needed to generate cinematics. The discourse generator could be improved by extending the system's ability to capture narratively significant actions that are temporally close (or overlap) in time but spatially distant. Currently, the solution to address this problem is to alter the speed of the camera shots used to capture them. One solution may be to show the audience multiple viewpoints simultaneously similar to techniques used in the television series *24*. This technique was implemented successfully by Bares in ConstraintCam [3] and additional work would allow for this to be integrated into Afterthought. Finally, more direct integration of Afterthought with a commercial game engine (that is, with direct access to engine source code) would allow the video rendering component to side-step a playback system and directly manipulate the game state to jump from replay point to replay point.

4. REFERENCES

[1] D. Amerson, S. Kime, and R. M. Young. Cinematic camera control for interactive narratives. In *Proceedings of the ACM International Conference on Advances in Computer Entertainment*, pages 365–370, 2005.

[2] D. Arijon. *Grammar of the Film Process.* Silman-James Press, 1991.

[3] W. Bares and J. Lester. Real-time constraint-based cinematography for complex interactive worlds. In *Proceedings of the international conference on innovative applications of artificial intelligence*, pages 114–120, 1998.

[4] Y. Cheong, A. Jhala, B. Bae, and R.M. Young. Automatically generating summaries from game logs. In *Proceedings of AIIDE-08*, pages 167–172, 2008.

[5] D. Christiansen, E. Andersen, D. Li-Wei He, D. Weld, M. Cohen, and D. Salesin. Declarative camera control for automatic cinematography. In *Proceedings of AAAI-95*, pages 148–155, 1995.

[6] S. Drucker and D. Zeltser. Camdroid, a system for implementing intelligent camera control. In *Proceedings of the 1995 symposium on 3D graphics*, pages 139–144, 1995.

[7] D. Friedman, Y. Feldman, A. Shamir, and T. Dagan. Automatic creation of movie summaries in interactive virtual environments. In *Proceedings of IEEE Virtual reality*, pages 191–199, 2004.

[8] Epic Games. *Unreal Tournament 3.* Epic Games, 2007.

[9] Next Level Games. *Mario Strikers: charged football.* Nintendo Company Ltd., 2007.

[10] N. Halper and M. Masuch. Action summary for computer games: extracting action for spectator modes and summaries. In *Proceedings of the second conference on the application and development of computer games*, pages 124–132, 2003.

[11] A. Jhala and R. M. Young. Cinematic visual discourse: Representations, generation and evaluation. *IEEE Transactions on Computational Intelligence and Artificial Intelligence in Games*, 2:69–81, 2010.

[12] EA Sports. *Madden NFL 2011.* Electronic Arts, 2011.

[13] Bungie Studios. *Halo 3.* Microsoft Game Studios, 2007.