Toward Intelligent Support of Authoring Machinima Media Content: Story and Visualization

Mark O. Riedl
Institute for Creative Technologies
University of Southern California
Marina Del Rey, California, USA
riedl@ict.usc.edu

Jonathan P. Rowe
Department of Computer Science
North Carolina State University
Raleigh, North Carolina, USA
jprowe@ncsu.edu

David K. Elson
Computer Science Department
Columbia University
New York City, New York, USA
dke4@columbia.edu

ABSTRACT

The Internet and the availability of authoring tools have enabled a greater community of media content creators, including non-experts. However, while media authoring tools often make it technically feasible to generate, edit and share digital media artifacts, they do not guarantee that the works will be valuable or meaningful to the community at large. Therefore intelligent tools that support the authoring and creative processes are especially valuable. In this paper, we describe two intelligent support tools for the authoring and production of machinima. Machinima is a technique for producing computer-animated movies through the manipulation of computer game technologies. The first system we describe, ReQUEST, is an intelligent support tool for the authoring of plots. The second system, Cambot, produces machinima from a pre-authored script by manipulating virtual avatars and a virtual camera in a 3D graphical environment.

Categories and Subject Descriptors

I.2.1 [**Artificial Intelligence**]: Applications and Expert Systems. H.5.1 [**HCI**] Multimedia Information Systems – *Artificial, augmented, and virtual realities.* J.5 [**Arts and Humanities**] – *Performing arts*

General Terms

Algorithms, Human Factors.

Keywords

Machinima, Intelligent Authoring Support, Story Authoring, Camera Control, Mixed-Initiative Systems

1. INTRODUCTION

Access to a nearly ubiquitous medium for information exchange – the Internet – and greater access to tools for media content production have led to a cultural phenomenon of user-generated content sharing. Tools exist for creating practically every type of artistic, creative, or communicative digital artifact, including pictures, music, video, and computer animation. While these tools make it technically feasible to produce creative content, they do not guarantee that the creator will produce work that is valued by a community as possessing quality or meaning. Therefore, the development of tools that support authors in creating purposeful content plays an important role in enabling and improving media content production. Tools that support the authoring of content, in contrast to those that focus on providing the technical ability to create a media artifact, are especially valuable when the authoring process is prohibitively costly, difficult, or time-consuming.

One example of technologically-supported, user-generated content is *machinima*. Machinima refers to the use of video game technology to ease the creation of animated cinema. Traditional filmmaking requires significant resources (sets, equipment, props, etc.) and talent (screenwriters, cinematographers, editors, etc.). Video games, providing rich graphics and interactivity, have been repurposed into tools for filmmaking in which users choreograph the avatars' movements to "perform" for a player whose perspective represents the camera (often adding dubbed dialogue). Some game manufacturers have embraced machinima by adding authoring tools and by providing special modes for scripting camera angles.

Despite tools that make it technically possible for non-experts to produce machinima, it is still complicated and time-consuming to design and produce cinematic narratives. Authors must, through the constraints of software, still manually position cameras and subjects. For example, [9] describes an interactive leadership training application that makes use of machinima "cut-scenes." The use of machinima simplified the creation of the cut scenes by avoiding traditional high-cost film production, but choreographing and encoding the cinematic camera shots still required over $800 in labor costs per minute of machinima video (Gordon, personal communication). Drawing from such experiences, we hypothesize that intelligent support technologies for machinima generation can offset prohibitive labor expenses and better enable non-professionals to produce cinematic content.

To support machinima authoring, we must first understand how films are made. McKee [16] describes the process of screenwriting where the idea of a story, told as a plot, is rendered first as a *treatment* and then as a script, which the film's director then renders as a completed movie. The treatment is a prose elaboration of the plot, including descriptions of where the scenes are set and what actions the characters take. The script, in turn, is an elaboration of the treatment. It adds production constraints (such as character blockings and camera angles) and character dialogue (see Figure 4 for an example of a typically formatted script). This pipeline for creative production calls for different sets of tools at each point. Such tools can be provided to non-professional filmmakers who intend to write and produce machinima but require guidance on the creative aspects of the process, as well as the technical ones.

Differences among the pipeline's stages raise questions about how to most effectively use AI for supporting machinima creation. Lubart [15] enumerates four ways in which computer interfaces can support creativity:

- **Computer as nanny:** The computer provides organizational and classification services and performs routine operations on behalf of the user.
- **Computer as pen-pal:** The computer facilitates brainstorming with functionality that captures and transmits to collaborators the user's thoughts.
- **Computer as coach:** The computer is knowledgeable about the process and can offer suggestions and stimulate creativity.
- **Computer as colleague:** The computer forms half of a human-computer team by contributing to the solution.

The computer as coach metaphor is used extensively in intelligent tutoring systems [23]. However, it is not our intention to teach a user how to author plots or produce machinima. A system-as-coach acts to facilitate improved task performance by a user, but does not attempt to solve or contribute to the problem on which the user is working. In contrast, the computer as colleague metaphor introduces automation into the creation process. Typical mixed-initiative support systems implement an expert system that is capable of solving some, or all, of the problem the human user is working on. The user and the system take turns, filling in details of the solution.

In line with these approaches to computational support of creativity, we address two of the processes in the pipeline for machinima creation: story authoring (the transition from idea to plot outline) and cinematic production (the transition from script to final movie). At these two stages of the pipeline we describe how different types of intelligent support tools can be brought to bear.

The first stage in the pipeline is to outline a plot without the details that go into the treatment [16]. Plot outlining does not require specialized knowledge per se, but non-expert human authors can benefit from critical analysis that substitutes for experience. The ReQUEST system is a computer-as-coach approach to intelligent authoring support that assists non-expert story writers with plot authoring. ReQUEST does not suggest or provide plot content, but instead operates as a surrogate audience to provide constructive feedback and help direct the user through the authoring process. In this regard, ReQUEST shares similarities with the functioning of intelligent tutoring systems [23]. Assisting non-expert writers with the transition from plot outline to treatment, and then to script with character dialogue, is left for future work.

The final stage in the pipeline is to realize the script on screen, as a visual 2D projection of activity occurring within a 3D graphical environment. Even though machinima reduces the labor and expense of traditional film production, this process requires knowledge and expertise about cinematic idioms in areas such as camera work and editing. The Cambot system [7] was originally designed to fully automate the cinematic realization of a script. In this work, we have incorporated Cambot into our machinima media content creation pipeline so that it augments a non-expert's gaps in knowledge and experience. In this role, Cambot becomes a computer-as-colleague.

2. STORY AUTHORING ASSISTANT

Story authoring is a creative act that can be challenging to novices. We propose that one method to assist a human story author is to provide an intelligent tool that is capable of delivering constructive feedback and direction. Unlike more conventional mixed-initiative approaches where an expert system is able to partner with a human user to solve a problem (in this case, the creation of a story), we are pursuing a technique that spurs creativity without the system taking the initiative to author content itself. This is analogous to intelligent tutoring, except that the goal is not to teach a process of story authoring, but to facilitate creative activity regardless of the human author's preferred process. We believe that plot outlining exemplifies the type of task where the computer-as-coach method of intelligent authoring support should be applied. It is desirable for the plot to be authored entirely by the human, avoiding the implication that the computer serves as a "co-author."

It is our belief that an intelligent authoring tool should analyze and suggest areas of the plot to which, due to the human authors' lack of experience with plot structure, improvements can be made. This should be done without explicitly suggesting plot content. This approach is in contrast to existing commercial tools for story authoring, such as Dramatica^{TM}, Power Structure^{TM}, and Truby's Blockbuster^{TM}, which primarily provide content organization based on popular approaches to screenwriting. We propose a system that uses artificial intelligence techniques to add layers of scaffolding and constructive feedback on top of traditional organizational support, independent of any prescribed approach to screenwriting.

Providing intelligent and constructive feedback on content creation requires a mechanism for acquiring and representing a story that is amenable to analysis by an intelligent system. To successfully do this, it is imperative to separate the essential aspects of a story – what happens when, and why – from the tangential aspects that are difficult for AI, such as natural-language understanding and common sense reasoning [6]. Although the complexities of natural language are highly relevant to the eventual performance of an authoring support assistant such as the one described here, they are beyond the scope of this paper.

Once a computational representation of a specific story is captured, it can be used to analyze causality and event importance [27], character intentionality [21], global coherence [24], and other content-based features. This representation can also be used to detect potential story anomalies such as causal dead-ends and un-motivated character goals and to support various types of quantitative analysis (e.g., distribution of characters, types of events, etc.). Furthermore, as we demonstrate, the knowledge acquisition process itself, when coupled with human authoring, provides opportunities for interactive support. The ideas informing this approach derive from an interdisciplinary body of work in screenwriting (e.g., [16]), intelligent tutoring, story understanding (e.g., [20][14]), authoring support tools (e.g., [18][25][17]), and psychological models of stories (e.g., [10]).

In this section we introduce ReQUEST, a prototype story authoring assistant. We begin by discussing QUEST [10], the psychological model of question answering that serves as our cognitive representation for encoding stories. We follow by describing our system, ReQUEST, which helps direct the story authoring process while concurrently constructing a QUEST model of the developing story.

2.1 QUEST Model of Narrative

QUEST [10] is a psychological model of question answering that simulates the question-answering performance of humans when responding to open-class questions about narrative content. Specifically, QUEST models encode the answers to why, how, when, enablement, and consequence-type questions. This type of model can be used to illustrate how people build cognitive representations of stories, and the manner in which these cognitive representations capture certain relationships between narrative events and the perceived goals of characters [10]. QUEST represents stories as directed graphs of plot elements, thereby capturing reader knowledge about story events, story states, and character goals. Directed links signify the causal relationships between story events and the intentionality relationships between events and character goals. Figures 1 and 2 [10] show an example story and its corresponding QUEST structure, respectively.

There are three types of nodes in a QUEST story model that are relevant to our purposes:

- **Event nodes**: Event nodes declare the occurrences of state-changing action in the story world.
- **State nodes**: State nodes declare particular snap-shots of the state of the story world.
- **Goal nodes**: Goal nodes declare the goals that characters have.

The links between nodes capture the different types of relationships between events, states, and goals. QUEST identifies the following five types of relationships:

- **Consequence (C):** Consequence arcs capture causality (i.e., event e_1 causes event e_2) or enablement (i.e., event e_1 makes it possible for event e_2 to occur). Consequence links can initiate from a state or event and terminate in a state or event.
- **Reason (R):** Reason arcs connect two goal nodes when one goal, the initiating node, can be explained as sub-goal of another goal, the terminal node. Chains of goal nodes made with reason links suggest the appearance that a character is implementing a plan.
- **Initiate (I):** Initiate arcs connect events to goals when the event can be interpreted as causing a character to adopt a goal that it previously did not have.
- **Outcome (O):** Outcome arcs connect a goal node to an event node when the event can be interpreted as achieving that goal.
- **Implies (Im):** The initiating event node implies the terminal event node.

Graesser et al. [10] illustrate the QUEST model of question answering with the following question pertaining to the story in Figures 1 and 2: "Why did the daughters stay in the woods too long" (node 5)? There are many possible answers, including:

A. Because the daughters forgot the time (node 4).

B. Because the dragon kidnapped the daughters (node 7).

C. Because the daughters were walking in the woods (node 2).

> Once there was a Czar who had three lovely daughters. One day the three daughters went walking in the woods. They were enjoying themselves so much that they forgot the time and stayed too long. A dragon kidnapped the three daughters. As they were being dragged off, they cried for help. Three heroes heard the cries and set off to rescue the daughters. The heroes came and fought the dragon and rescued the maidens. Then the heroes returned the daughters to their palace. When the Czar heard of the rescue, he rewarded the heroes.

Figure 1. Example story, "The Czar's Daughters" [10].

D. Because the heroes fought the dragon (node 18).

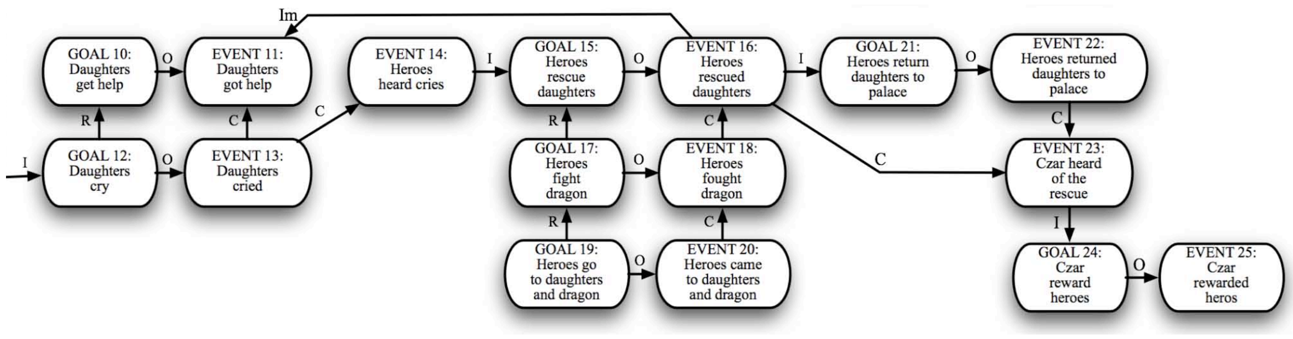
Both the question and each possible answer correspond to nodes in the knowledge structure. The QUEST model defines search procedures for each type of question (e.g. why, how, when, enablement, and consequence). The search procedures, starting at the queried node, distinguish between legal answer nodes and illegal answer nodes. That is, only nodes reachable by the search procedures are legal answer nodes. Answers (A) and (C) are legal answers. Of those two, (A) is preferred by the QUEST model because the corresponding node has a smaller structural distance from the queried node. The legality of answers and the weight of structural distance determine the goodness of answer judgments rendered.

One notable feature of QUEST is its implementation of goal hierarchies. A goal hierarchy is a sequence of events in which a character is perceived to intentionally act to achieve some goal state. A goal hierarchy can be perceived as a character plan. Nodes 15 through 20 in Figure 2 illustrate a goal hierarchy. Some event – the heroes hearing the cries of the daughters – initiates in the minds of the heroes the top-level goal to rescue the daughters (node 15), which is eventually achieved (node 16). In this example, the heroes are considered to be a single intentional entity. Nodes 17 and 19 show the decomposition of the top-level goal, denoting the intermediate goals that must be achieved for the heroes to achieve the top-level goal. Goal hierarchies are important because story world characters are perceived to be intentional agents by the audience. Breakdowns in the audience perception of character intentionality result in failure to see a character as believable [21][22] and loss of suspension of disbelief [8].

2.2 ReQUEST: Toward Story Authoring Assistance through Audience Modeling

ReQUEST is an intelligent story authoring support system that assists a non-expert author in creating meaningful narrative content. ReQUEST uses an authoring paradigm where story events, states, and character goals are individually written in natural language on notecard-like forms. Based on this information, ReQUEST generates questions that a hypothetical audience might have about the story in order to help the author further flesh out the narrative. The author can ignore, delay, or answer these questions at any time. The human author answers questions by authoring new plot elements or by referring to existing ones.

The use of forms as an authoring paradigm is not far removed from an approach advocated by McKee [16]. McKee suggests that successful writers often use index cards to develop what he calls a step-outline: "the story told in steps." The author writes events on each card that, in aggregate, constitute a story. McKee also advocates adding structural metadata on the cards. In



Im
GOAL 10: Daughters get help
O
EVENT 11: Daughters got help
EVENT 14: Heroes heard cries
I
GOAL 15: Heroes rescue daughters
O
EVENT 16: Heroes rescued daughters
I
GOAL 21: Heroes return daughters to palace
O
EVENT 22: Heroes returned daughters to palace
R
C
C
R
C
C
I
GOAL 12: Daughters cry
O
EVENT 13: Daughters cried
GOAL 17: Heroes fight dragon
O
EVENT 18: Heroes fought dragon
C
EVENT 23: Czar heard of the rescue
R
C
I
GOAL 19: Heroes go to daughters and dragon
O
EVENT 20: Heroes came to daughters and dragon
GOAL 24: Czar reward heroes
O
EVENT 25: Czar rewarded heros

Figure 2. Example QUEST model.

addition to content descriptions, each form in ReQUEST possesses a small number of metadata annotations, composed of information such as relevant characters, whether or not an event was intentional, whether the content establishes or resolves some aspect of the story, event location, starting time, and goal achievement time. In accordance with McKee's approach, much of this metadata is obtained under the pretense of organizational support, thereby minimizing the annotation burden for the author. Through these parallels, ReQUEST's approach derives from McKee's suggested *inside-out* approach to authoring.

Particularly important to narrative understanding are (a) the causal connections between events, and (b) the motivations, goals, and intentions of characters as they act in the story world. As authoring progresses, ReQUEST updates a QUEST model of the story-so-far based on input of plot elements and the answers of system-posed questions that the human author has elected to address. Relations between plot elements can only be acquired when the human author elects to answer questions posed by ReQUEST – no annotations or further input are necessary to specify causal or intentional connections within the system's plot representation. The QUEST model maintained by ReQUEST is never exposed to the user; the goal of the author is to create a story that he or she is pleased with, not to create a directed graph that satisfies the computer system. Therefore, ReQUEST serves as a surrogate audience by building a QUEST model that can be used to generate questions and serve as a mechanism for story analysis. This question-based support paradigm enables ReQUEST to provide scaffolding and assistance to the author. This is in contrast to more conventional mixed-initiative approaches where the system may take the initiative to add or modify narrative content, or to suggest additions or modifications.

While form completion is the primary mechanism for adding content to the developing story model, question answering is the method through which a QUEST model is constructed from individual QUEST nodes. There are currently two types of questions that are posed to the author:

- **Why questions:** Seek to determine what caused or enabled some event or state, or what initiated a particular character's goal.
- **Consequence questions:** Seek to determine the story consequences of some authored content, or what the content in turn enables to happen.

Additional question types are under development. Further questioning will seek to reason about more abstract descriptions of the story and requisite structural components of the narrative. However, the existing question types appear to be effective at generating traditional QUEST structures.

Although an initial version of the core authoring support mechanism, ReQUEST, has been implemented, the user interface for an authoring support tool that utilizes ReQUEST has not been designed or developed. Regardless, a future interface should allow the user to author events in any order and position the events relative to each other to determine the story's temporal order. Further, questions from ReQUEST should be presented to the user in a non-obtrusive way so as to not interrupt the *flow* [3] of the human author's creative process. This can be accomplished through a special ReQUEST dialogue pane in the interface, or through other intelligent techniques such as detecting when the user is ready to shift his or her attention. This is also related to the requirement that the user be able to ignore or delay answering questions during authoring. Due to the variability inherent in natural language, the system could pose unnecessary or irrelevant questions, making it vital that the author be able to disregard such questions.

2.2.1 ReQUEST Processing

ReQUEST utilizes a robust, rule-based approach to constructing a QUEST representation of the authored story, and providing feedback to the author. The rules are implemented in the JESS rule engine. There is no natural language component to ReQUEST, so its rules must utilize the form-types, annotations, and causal links explicitly created and inferred during the authoring process. Five different types of rules operate upon the various content types and annotations to assemble a QUEST representation of the story:

- **Node creation rules:** Automatically generates QUEST nodes based on story content that has been previously authored.
- **Relational examination rules:** Examines the relationships between story content nodes to determine whether a question can be considered 'answered' or not, marking the question appropriately.
- **Response handling rules:** Defines relations between story content nodes based upon the author's responses to questions.
- **Question generation rules:** Recognizes holes in the representation and generates questions in an effort to fill these holes. Logic about question ordering is also incorporated into these generation rules.

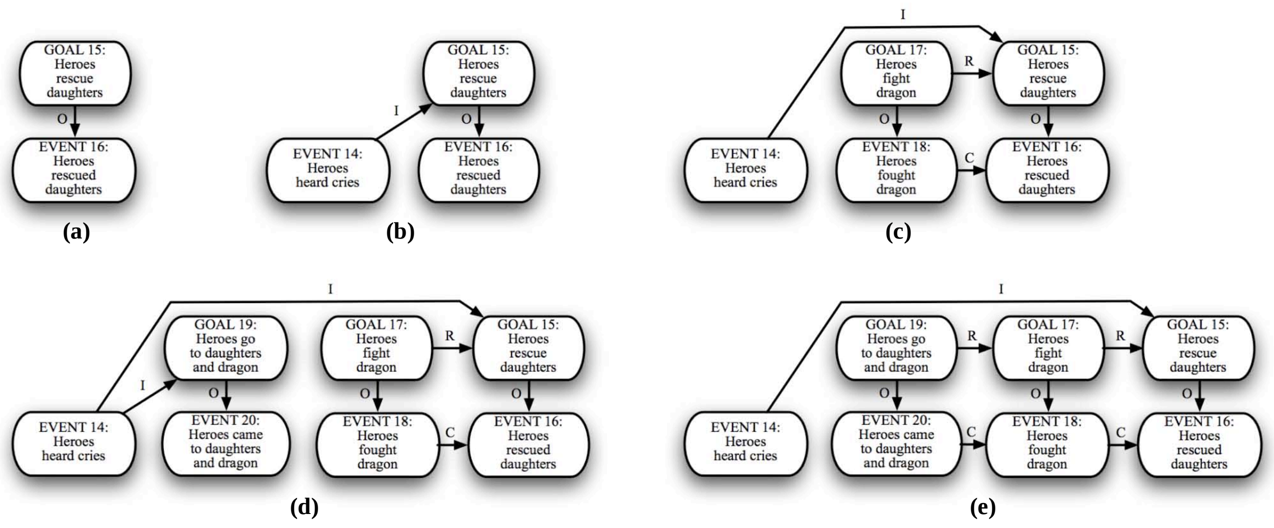

GOAL 15: Heroes rescue daughters
O
EVENT 16: Heroes rescued daughters
(a)
EVENT 14: Heroes heard cries
I
(b)
GOAL 17: Heroes fight dragon
R
EVENT 18: Heroes fought dragon
C
(c)
GOAL 19: Heroes go to daughters and dragon
EVENT 20: Heroes came to daughters and dragon
(d)
(e)

Figure 3. Example of ReQUEST processing of character goal hierarchies.

- **Internal consistency rules:** Supports robustness in question answering by recognizing anomalous causal relation patterns and correcting the QUEST model's structure.

As an event, goal, or state is authored, a corresponding node is added to the QUEST model. Depending on the content type, this new node triggers rules that result in the creation of further nodes or the generation of related questions. For example, an authored event that is annotated as intentionally performed by a story world character will trigger rules that create and link a corresponding goal node for that character. Questions are also generated about why characters want events to happen and what the consequences of the events are. Before the questions are posed to the human author, rules examine the existing QUEST model in an attempt to answer the new and existing questions. If the new questions are not currently answerable by ReQUEST, the story is considered flawed from the perspective of the audience, meaning the audience may not be able to understand the relationship between plot elements. Consequently ReQUEST poses new questions to the human author. As the author responds to questions, rules fire that interpret the answers in the context of the current QUEST model and create new links between nodes signifying causal or intentional relationships. Rules examining the new QUEST model are re-fired in an attempt at answering the earlier questions, and the questions are marked as answered or remain unanswered as appropriate.

ReQUEST is capable of assisting human authors in recreating the Czar's Daughters story (Figure 1) [10] and the Aladdin story from [21][22] – two stories with known QUEST models. The example below illustrates how authoring occurs, rules fire, questions are generated and answered, and a QUEST model is constructed.

2.2.2 ReQUEST Example

Assume ReQUEST were to assist an author in the creation of the Czar's Daughters story shown in Figure 1. In this example, we show how the Czar's Daughters story can be recreated with ReQUEST. We use an existing story because its structure is known and can serve as a basis for comparison. It is not necessary to have an existing story and neither the hypothetical author in this example nor the system has any knowledge about the final story structure ahead of time. At some point during the authoring of the story, the author fills out a form describing the event "The heroes rescue the daughters." The event form is annotated as an intentional action that neither establishes nor resolves any aspect of the story, is centered around the heroes, and has an associated location and time. The creation of an intentional action results in an Event node being constructed within the QUEST model. Since the event is annotated as being intentional, a rule in the node generation category fires and creates a corresponding goal node. An *outcome* arc is also created to connect the newly created goal node to the event node. See Figure 3a.

The existence of new content nodes in the QUEST representation causes question generation rules to fire. Because the rescuing event is not annotated as an establishing event, a "why" question is generated: "Why did the heroes want to rescue the daughters?" Note that the presentation of the question to the human author – including the exact wording of the question – is left to the user interface. The semantic meaning of the question is given in the quote marks. Asking why a character wants to do something encourages the author to think about the goals and motivations of the character. No incoming *initiates* arc or outgoing *reason* arc currently exist from the associated goal node, so ReQUEST cannot answer the question by itself. Assuming there is a flaw in the story-so-far from the audience's perspective, ReQUEST poses the question to the author.

Suppose at some point the human author chooses to address the "why did the heroes want to rescue the daughters?" question by authoring a new event, "Heroes heard cries," which is temporally positioned before the rescue event. Assume that the author annotates the event as unintentional; no corresponding goal node is created. Because the rescuing event occurs *before* the rescue event, ReQUEST believes this to indicate that hearing the cries initiates the goal of rescuing the daughters in the minds of the heroes. An *initiates* arc is extended from the new event to rescue goal node, as shown in Figure 3b.

Later, the author creates an intentional event described as "The heroes fight the dragon." Assume the system creates associated goal node, *outcome* arc, and a "Why did the heroes want to fight the dragon?" question as in the process illustrated above. At some point the author chooses to answer the question by pointing out that the question can be answered by previously authored event, “The heroes rescue the daughters.” The author is signifying that, in his or her mind, the heroes wanted to fight the dragon in order to achieve the goal of rescuing the daughters. Pointing out the rescue in response to the question triggers the response handling rules. Because the rescuing event occurs *after* the fighting event, ReQUEST believes this to indicate that fighting the dragon is part of a larger plan and that “rescuing the daughters” is a super-ordinate goal of the characters. The response handling rule that fires realizes this by creating a goal hierarchy, connecting the two relevant event nodes with consequence arcs and the two corresponding goal nodes with reason arcs, as seen in Figure 3c. Goal hierarchies always consist of chains of intentional events and goals according to temporal order.

When the author creates the event, “Heroes go to the daughters and dragon,” ReQUEST generates the related goal node, *outcome* arc, and a “Why do the heroes want to go to the daughters and dragon” question. Suppose the author immediately responds to this question by pointing ReQUEST to the event, “Heroes heard cries,” that was previously authored. Since “Heroes heard the cries” temporally occurs before the new event, an *initiates* link is established, as illustrated above. Two things happen. First, ReQUEST’s relational examination rules recognize that since both “Heroes rescue the daughters” and “Heroes go to the daughters and dragon” goals are initiated by the same event, they must be part of the same character plan. The event, “Heroes go to the daughters and dragon,” is incorporated into the goal hierarchy as shown in Figure 3d. Second, ReQUEST’s internal consistency rules recognize that it is incorrect for any but the top-most goal in a goal hierarchy (in this case “Heroes rescue the daughters”) to be the terminus of an *initiates* arc. Rules fire that destroy the initiates arc between “Heroes heard the cries” and “Heroes go to the daughters and dragon” that was created in response to the latest answer by human author to the system-posed question (see Figure 3e for the final, correct goal hierarchy for the heroes).

2.2.3 Informal Evaluation

We conducted an informal, Wizard-Of-Oz-like evaluation to assess the effectiveness of ReQUEST’s approach to story authoring support, its ability to generate a QUEST model from an authored story, and the effects of questioning on the authoring process. The single subject was a third-year male West Point cadet. As part of his deployment rotation, the cadet was tasked with outlining a fictional story about a leadership issue of his choice. Prior to the study, the participant had spent about a week and a half collecting news stories and reports from the Iraq War that pertained to a theme of his choosing: developing trust between Iraqi and US forces. He entered our study with a broad knowledge of this issues he would like to address, but, reportedly, no premeditated plot details. The participant had no professional writing experience and had never written a story of significant length before.

Unlike standard Wizard-Of-Oz studies, the subject was not unaware that a human “Wizard” was performing the role of the computer system. The secrecy was not required because we were only gauging the effectiveness of the ReQUEST algorithm, and not concerned about the subject’s judgment of the system or its usability. The “interface” through which the participant was to outline the story was a stack of 5” x 8” colored Post-it® notes that were to be written upon and adhered to a wall. The participant was instructed to keep the content on each note relatively atomic. Two assistants, who were moderately experienced writers, sat in on the session to provide a minimal support role to the participant. However, the participant was ultimately responsible for the story and the only one allowed to post items to the wall.

One of the authors of this paper represented the ReQUEST AI system. Questions generated by the ReQUEST rules were posed to the author through differently colored notes adhered to an adjacent wall. The participant was instructed that he was free to address the questions at any time or to completely ignore them. The only stipulation was that answers must come in the form of existing or newly authored notes.

The participant developed the outline of an original story in approximately two hours during the session. Approximately 28 notes were authored. Twenty-three questions were posed to the author, consisting of “why” and “consequence” questions. Of these, six questions were ignored, seven questions were answered by authoring new story content, and 10 were answered by pointing to elements that were already authored before the question was addressed.

Observations of the participant suggest that questions were not obtrusive and did not interfere with the participants authoring style. Authoring tended to occur in bursts; several events would be posted in quick succession without regard for any system-posed questions. After authoring slowed, the participant would then consult the questions that had since accumulated. Many of the questions were answerable through existing content, but occasionally a question would catalyze further authoring. Although only seven questions were directly answered by authoring new content, question-answering ultimately resulted in more than seven pieces of story content being authored. This unobtrusive questioning paradigm appeared to help direct the participant without constraining or obstructing authoring, and lead us to adopt the term “therapist” when describing how ReQUEST operated in practice.

The resulting QUEST model constructed by the application of ReQUEST rules contained 47 nodes. Space precludes us from showing it here. The QUEST model consists of fourteen disjoint sub-graphs. The disjunctions occurred due to questions that were ignored by the participant, which prevented the system from making the explicit linkages, and due to questions that were answered in unanticipated ways. For example, the participant would answer a “why” question with “because that is how the character is,” implying that intentional events were enacted because of character personality traits. These character trait considerations prompted us to extend the ReQUEST system to handle character considerations other than goals. Testing on a single user is not sufficient to allow us to draw any conclusions about ReQUEST’s performance as an authoring support tool. However, our evaluation has allowed us to test the robustness of the rules as well as provided some anecdotal evidence that the system may eventually be a valid approach to authoring support.

3. MACHINIMA VISUALIZATION AND EDITING

We have not yet addressed the stage of the pipeline in which a plot outline is rendered into a treatment [16] and then into a script. However, once a script has been created, the final stage is to “shoot” the script to render it into a completed movie. Machinima can reduce the cost and time of production, making it possible for non-experts to create animated movies. But machinima still requires the author to be proficient in visual storytelling, including appropriate camera angles, blocking (i.e., placement) of characters, and editing. Because of the need for the human author to possess highly specialized skills at this stage, both creative and technical, we believe that the previously used computer-as-coach metaphor is not appropriate during this stage of machinima creation. Cinematography can more readily be modeled computationally than plot authoring; therefore, it more readily lends itself to reasoning by fully automated expert systems. Cambot [7] was designed as an automated camera control system that manipulates avatars and camera viewpoint in a 3D graphical environment in order to visually realize scripted scenes.

We have adapted Cambot by incorporating it into the machinima creation pipeline as a *computer-as-colleague* approach to authoring support. This approach support is exemplified by mixed-initiative systems, which typically implement an expert system that is capable of solving some (or all) of the problem faced by the human user. Cambot can be utilized as an expert system in mixed-initiative authoring support because it can generate solution details (e.g. specific camera angles) where the human user is unable to or chooses not to specify them. When supporting the authoring pipeline, Cambot allows the human author to specify as many production details as he or she desires in the input script. The system then applies its knowledge to solve the problem of “shooting” the script, calculating appropriate creative choices that the user did not specify.

Cambot’s approach is modeled after the actual filmmaking process, and relies on a large database of cinematic knowledge to achieve results that are aesthetically acceptable. Sections 3.1 and 3.2 briefly describe the filmmaking process and the Cambot system, respectively.

3.1 Filmmaking

The production phase of traditional filmmaking begins with a script. The script describes at least one scene, in which a segment of action and dialogue takes place in a continuous span of time and at a single location. Scenes are made of a succession of *beats* [16], the smallest divisible segments of a scene, typically encompassing one line of dialogue or a moment of action.

The core of a scripted scene consists of character actions and dialogue acts that advance the narrative. The other elements of the script specify how the scene should look and sound. The author may, for example, give the location where the scene may take place (e.g., a street), the blocking of the characters (standing side by side), or essential information to include in the camera viewpoint (a threatening sky).

Working from this script, the director has to satisfy many overlapping constraints. Fortunately, there is a generous “search space” from which he or she can craft the best visualization of the script. There may be many appropriate locations at which to shoot a scene, several ways the director can position (i.e., *block*) the actors and a multitude of camera angles that satisfy both the script and the director’s visual style. Once the director has obtained “coverage” – that is, a shot from at least one angle for each beat – he or she can make final selections, as well as manage global considerations such as pacing, in the editing room. A single shot may last as long as a scene or as short as a fraction of a second. The resulting “reel” contains the fully realized scene, from a visual standpoint. The director then turns to the audio, adding voice-over dialogue, sound effects and music to complete the movie. Audio is outside the scope of the work reported here.

3.2 Cambot: Intelligent Cinematography for Machinima

Cambot closely models the real filmmaking process to function as a virtual director for offline machinima production. Given a script, it blocks characters, identifies possible shot compositions, and edits the best available shots into a final set of time-indexed dialogue and gesture commands. These commands are rendered by a separate visualization system. Cambot is thus not bound to a particular game engine, or to the machinima pipeline in general.

There are two types of input that Cambot needs to realize a scene. One is a *set*, a 3D environment annotated with labels that describe the types of locations that can be evoked in each constituent space (such as a street or an interior space). In this manner, the set acts as a studio back-lot, where locations can be re-used from film to film. The other input is a symbolically encoded script that is analogous to the model script described above. Structurally, the script is divided into a number of scenes, each of which consists of at least one beat. Scenes and beats contain the following types of information:

- **Character declarations:** A scene must declare which characters are present, and which of the available avatars Cambot should invoke for each character.
- **Actions:** Actions include character gestures and lines of dialogue.

Given no other information, Cambot is able to realize a scene from these elements alone. As discussed above, the restrictions used in a real script to guide the look of the scene are supported by Cambot in the form of optional constraints. There are four dimensions of constraints that a script may use to guide Cambot’s aesthetic decisions when determining how to shoot a scene:

- **Location constraints:** These indicate to Cambot which areas of the digital back-lot are appropriate for shooting the scene.
- **Blocking constraints:** Each beat may be annotated with constraints on the movements and locations of declared characters relative to one another.
- **View constraints:** The script may recommend (or require) that Cambot cover a certain beat with a certain type of shot, e.g., a close-up of a particular character, a shot that Cambot knows to be “intense,” or a shot in which the camera moves.
- **Scene constraints:** The script may guide Cambot’s aesthetic choices in assembling complete reels, e.g., to use as few cuts as possible.

The author of the input script can exercise as much control over the resulting visualization as he or she desires. That is, if the human author knows exactly how the actors should be blocked and how the scene should be shot, he or she can indicate this through the script input. Cambot will determine how to best assign values to the variables that are not explicitly authored.

Cambot uses each of the script’s constraints to select among all the assets available for realizing the scene. The constraints are matched against a hand-authored library of bits of cinematic knowledge called “facets.” Facets fall into the following types:

- **Stages:** A *stage* is an area of space that Cambot assumes to be free from occlusions and obstructions. It functions as the frame on which the other elements of a scene (characters and cameras) are mounted. A stage can be any polygonal shape.

EXT. KABUL CITY STREET - NIGHT

SERGEANT SMITH, 29 y.o. male, is standing in a street, gun at his side. CAPTAIN YOUNG, 34 y.o male, approaches him.

YOUNG
(1)What's your condition, Sergeant Smith?

SMITH
(2)Captain Jones, sir, road Beta One is secure.

We see a few Afghan civilians chatting before them.

SMITH (cont'd)
(3)The city's pretty cold tonight.

YOUNG
(4)Perez tells me you have a message from a local?

SMITH
Yes, Captain, a villager told me that his uncle would like to speak to you about the food drop.

YOUNG
(5)Probably Gol Omar again. He's scrambling for ways to get his fingers in that honeypot.

SMITH
(6)Yes sir, and word is, he's gathering a hundred men at the ridge overlooking the base.

YOUNG
(7)Tell him I'll meet him at noon outside the base. Tell him to come unarmed and alone.

SMITH
Copy that.

Figure 4. Example script. Beats are numbered in superscript.

- **Blockings:** A *blocking* is a geometric placement of abstract characters relative to the center point of a stage. A blocking must be invoked along with a stage that is sufficiently large to contain each blocked character. A blocking can specify character movements.
- **Shots:** A *shot* is defined to be the position, rotation and focal length of a virtual camera relative to the center point of a stage. The camera can move within a shot.

Stages, blockings, and shots are used in conjunction with one another by aligning their respective center points. Not all elements are compatible; that is, there is a many-to-many, but incomplete mapping between stages, blockings, and shots that can be combined. Cambot uses stages to "package" shots and blockings together in a way that guarantees freedom from occlusions and obstructions. Once combined, stages, shots and blockings are *anchored* onto the set in order to determine the concrete Cartesian coordinates that instantiate them at the chosen location.

Cambot treats user input – in the form of a script consisting of beats with possibly incomplete location, blocking, view, and scene specifications – as a set of constraints that must be satisfied in order to find a sequence of shots that cover all the beats. The input constraints define a search space comprised of compatible locations, blockings, and shots. Cambot uses a combination of breadth-first search and dynamic programming to search this space to find the highest-scoring combination of locations, blockings, and shots that cover each beat. Score is computed relative to the degree of satisfaction of user-provided (or default) aesthetic constraints. The result of this process is a sequence of shots, blockings, gestures and dialogue acts, along with precise timing information, that can be sent to a visualization engine for final rendering. More details can be found in [7].

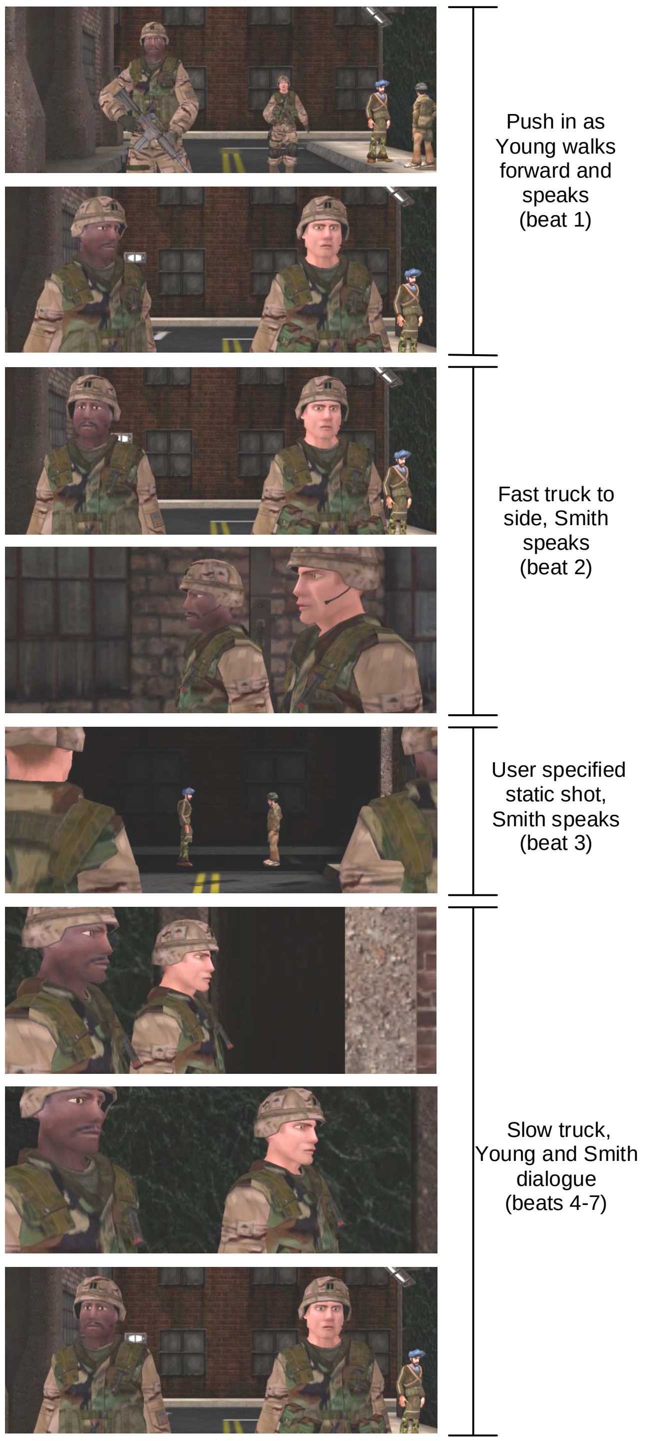

Push in as Young walks forward and speaks (beat 1)
Fast truck to side, Smith speaks (beat 2)
User specified static shot, Smith speaks (beat 3)
Slow truck, Young and Smith dialogue (beats 4-7)

Figure 5. Screenshots from visualization of the example script.

Once all scenes in a script have been processed, Cambot instructs the visualization engine to begin the rendering process via network socket. Cambot instructs the visualizer to (a) place avatars and the camera at particular Cartesian coordinates, (b) to play avatar animations or have avatars speak dialogue, and (c) to move the avatars or the camera along particular trajectories. It associates scheduling information with each instruction to ensure proper synchronization between characters and camera. Currently the visualization engine is Unreal Tournament^{TM} with modifications to accept temporally parameterized character and camera positions based on [29].

To test the system, we created the script shown in Figure 4. The script describes a situation where two deployed military personnel come together to brief one another. A symbolic encoding of the same script, along with a cityscape virtual set, were sent to Cambot, which was configured with a stylistic heuristic that favors minimal cuts and maximal camera movement. The script input did not specify any shot information except to force Cambot to use a static shot to show point of gaze of the Soldiers in beat 3. The resulting reel is shown in Figure 5.

4. RELATED WORK

4.1 Narrative Authoring and Understanding

Numerous narrative authoring systems have been developed, e.g., [17][18][25]. Most authoring systems are for interactive narrative environments due to the computational complexity of authoring branching structures. Typically these systems focus on user interface designs to facilitate narrative knowledge entry. While ReQUEST must still be coupled with a user interface, ReQUEST differs from these other approaches in that it supports non-branching narrative authoring and does so by stimulating content creation through question-answering.

ReQUEST shares similarities with narrative understanding systems, e.g., [14][4][28][20][19]. AQUA [20] in particular shares a particular methodology with ReQUEST. AQUA attempts to understand a narrative by detecting anomalous narrative statements, posing questions, and attempting to answer those questions with schemas or cases in order to explain – and thus understand – the anomalies. ReQUEST also generates questions when it detects anomalies in the story-so-far. Anomalies occur when ReQUEST detects unmotivated character goals, ill-formed character plans, and non-established events and states. However, since ReQUEST is an authoring support tool, it is possible that explanations for anomalies have not yet been authored (or that the anomalies are false-positives because ReQUEST lacks the ability to comprehend the author's natural language input). ReQUEST handles questions by asking the author to provide the explanation either by authoring additional content or by referencing existing content.

4.2 Virtual Camera Control

Cambot is a virtual cinematography system. Related work in virtual cinematography includes techniques involving visual fly-throughs [5], finite-state machine encoding of idioms [12], grammatical encoding of idioms [2], constraint satisfaction [1], genetic algorithms [11], autonomous agents [26], and planning [13]. While most related work in virtual cinematography adopts and encodes cinematic knowledge – typically in the form of idioms – Cambot attempts to model the filmmaking process itself. Cambot models the filmmaking process as a constraint satisfaction process in which the script specifies the constraints with which a set of beats must be covered by shots. This differs from other systems that use constraints (e.g., [1][11]) because those systems specify constraints on visual camera angles, whereas Cambot uses constraints that reflect the entire production process, including set location, blocking, camera shot, and affectual variables such as scene intensity.

The advantage of the approach used in Cambot, with respect to machinima authoring support, is that it can be incorporated into a mixed-initiative system. The human author provides as many of the visualization details as he or she chooses or is able. Cambot searches for a sequence of coordinated actor blockings and camera shots that satisfy the constraints inferred from the script's beats and made explicit by the author.

5. CONCLUSIONS AND FUTURE WORK

There is a demand for media modalities through which one can express views and share experiences. Computer games and other consumer graphics technologies, in conjunction with tools for customizing those technologies, enable users to create increasingly sophisticated visual narratives. However, the creative processes involved in visual storytelling are varied and, at times, complex. Machinima production is technically feasible in the sense that there are tools that can be brought to bear on the problem of visually realizing a story. Although machinima technically reduces the cost and labor involved in producing cinematic movies, it still requires creative skill and experience. We believe that artificial intelligence can be used to create authoring support tools that can augment the abilities of non-expert storytellers to create meaningful machinima.

We have described two systems, ReQUEST and Cambot, that provide intelligent support for non-expert who wish to create machinima. The first system, ReQUEST, supports plot-level authoring of stories by applying knowledge about audience question-answering in order to assess the causal and character attributes of plots as they are authored. The second system, Cambot, is an expert system that manipulates virtual avatars and a virtual camera (viewpoint) in a 3D graphical environment to produce aesthetically acceptable visualizations of an authored script.

ReQUEST and Cambot provide very different perspectives into the general field of intelligent support for authoring. The process of creating a movie from scratch, whether it a conventional film or machinima, requires many different processes [16]. Each of these processes requires different capabilities and modalities of interaction from intelligent support tools. The first creative act, plot outlining, is a computationally ill-defined task where the creative content should be a reflection of the human author's intentions. This suggests a computer-as-coach approach to intelligent authoring support in which the tool does not contribute to the plot, but analyzes human-authored content from different perspectives. Once a script has been written, the process of visually producing the script requires a completely different set of knowledge and skills. In contrast to plot outlining, visual realization requires technical competencies that a non-expert is unlikely to have. In visual storytelling, whether in conventional film or machinima, there are conventions and idioms that can be computationally encoded. To support this process in the pipeline, we believe that a computer-as-colleague approach allows an expert system to effectively supplement the human's capabilities by making decisions about visualization that the human author is unable or unwilling to specify.

One part of the process of creating machinima that we have not addressed is the bridging of the gap between what ReQUEST helps users author – a plot outline – and what Cambot requires as input, a movie script. Future work involves bridging this gap by researching and developing additional intelligent support tools that help a non-expert human user author the additional elements.

6. ACKNOWLEDGMENTS

The project or effort described here has been sponsored by the U.S. Army Research, Development, and Engineering Command (RDECOM). Statements and opinions expressed do not necessarily reflect the position or the policy of the United States Government, and no official endorsement should be inferred. Special thanks to Andrew Gordon, Julia Kim, Salvo Lavis, Lindsay Armstrong, and Eric Garcia.

7. REFERENCES

[1] Bares, W.H. and Lester, J.C. Intelligent Multi-Shot Visualization Interfaces for Dynamic 3D Worlds. In *Proc. of the 1999 Int. Conf. on Intelligent User Interfaces*, 1999.

[2] Christianson, D., Anderson, S., He, L., Salesin, D., Weld, D., and Cohen, M. Declarative Camera Control for Automatic Cinematography. In *Proc. of 13th National Conf. of the AAAI*, 1996.

[3] Csikszentmihalyi, M. *Creativity: Flow and the Psychology of Discovery and Invention*. Harper Collins, New York, 1996.

[4] Cullingford, R.E. SAM and Micro SAM. In R. Schank and C. Riesbeck (Eds.) *Inside Computer Understanding*. Erlbaum, Hillsdale, NJ, 1981.

[5] Drucker, S.M. and Zeltzer, D. Intelligent Camera Control in a Virtual Environment. In *Proc. of Graphics Interface 1994*.

[6] Elson, D.K. and McKeown, K.R. A Platform for Symbolically Encoding Human Narratives. In *Proc. of the AAAI Fall Symposium on Intelligent Narrative Technologies*, 2007.

[7] Elson, D.K. and Riedl, M.O. A Lightweight Intelligent Virtual Cinematography System for Machinima Production. In *Proc. of the 3rd Conf. on AI for Interactive Digital Entertainment*, 2007.

[8] Gerrig, R.J., and Pillow, B.H. A developmental perspective on the construction of disbelief. In J. De Rivera and T. R. Sarbin (Eds.), *Believed-in imaginings: the narrative construction of reality*, American Psychological Association, Washington, DC, 101-109, 1998.

[9] Gordon, A.S., van Lent, M, van Velsen, M., Carpenter, P., and Jhala, A. Branching Storylines in Virtual Reality Environments for Leadership Development. In *Proc. of the 16th Innovative Applications of AI Conference*, 2004.

[10] Graesser, A.C., Lang, K.L., and Roberts, R.M. Question Answering in the Context of Stories. *Journal of Experimental Psychology: General*, 120, 3 (1991).

[11] Halper, N. and Olivier, P. CAMPLAN: A Camera Planning Agent. In *Proc. of the AAAI Spring Symposium on Smart Graphics*, 2000.

[12] He, L., Cohen, M.F., and Salesin, D. The Virtual Cinematographer: A Paradigm for Automatic Real-Time Camera Control and Directing. In *Proc. of the 23rd Int. Conf. on Computer Graphics and Interactive Techniques*, 1996.

[13] Jhala, A.H. and Young, R.M. A Discourse Planning Approach for Cinematic Camera Control for Narratives in Virtual Environments. In *Proc. of the 20th National Conf. of the AAAI*, 2005.

[14] Lehnert, W.G., Dyer, M.G., Johnson, P.N., Yang, C.J., and Harley, S. BORIS – An Experiment in In-Depth Understanding of Narratives. *Artificial Intelligence*, 20 (1983).

[15] Lubart, T. How Can Computers be Partners in the Creative Process: Classification and Commentary on the Special Issue. *Int. Journal of Human-Computer Studies*, 63, (2005).

[16] McKee, R. *Story: Substance, Structure, Style, and the Principles of Screenwriting*. HarperCollins, New York, 1997.

[17] McNaughton, M., Cutumisu, M., Szafron, D., Schaeffer, J., Redford, J., and Parker, D. ScriptEase: Generative Design Patterns for Computer Role-Playing Games. In *Proc. of the 19th IEEE Int. Conf. on Automated Software Engineering*, 2004.

[18] Medler, B. and Magerko, B. Scribe: A Tool for Authoring Event Driven Interactive Narrative Drama. In *Proc. of the 3rd Int. Conf. on Technologies for Interactive Digital Storytelling and Entertainment*, 2006.

[19] Mueller, E.T. Understanding Script-Based Stories using Commonsense Reasoning. *Cognitive Systems Research*, 5, 4, (2004).

[20] Ram, A. AQUA: Questions that Drive the Explanation Process. In R. Schank, A. Kass, and C. Riesbeck (Eds.) *Inside Case-Based Explanation*. Erlbaum, Hillsdale, NJ, 1994.

[21] Riedl, M.O. *Narrative Generation: Balancing Plot and Character*. Ph.D. Dissertation, North Carolina State University, Raleigh, NC, 2004.

[22] Riedl, M.O. and Young, R.M. An Objective Character Believability Evaluation Procedure for Multi-Agent Story Generation Systems. In *Proc. of the 5th International Working Conference on Intelligent Virtual Agents*, 2005.

[23] Shute, V. J. and Psotka, J. Intelligent Tutoring Systems: Past, Present and Future. In D. Jonassen (ed.) *Handbook of Research on Educational Communications and Technology*. Scholastic Publications, 1996.

[24] Smith, E. and Hancox, P. Representation, Coherence and Inference. *Artificial Intelligence Review*, 15 (2001).

[25] Spierling, U., Weiß, S.A., and Müller, W. Towards Accessible Authoring Tools for Interactive Storytelling. In *Proc. of the 3rd Int. Conf. on Technologies for Interactive Digital Storytelling and Entertainment*, 2006.

[26] Tomlinson, B., Blumberg, B., and Delphine, N. Expressive Autonomous Cinematography for Interactive Virtual Environments. In *Proc. of the 4th International Conf. on Autonomous Agents*, 2000.

[27] Trabasso, T. and Sperry, L. Causal Relatedness and Importance of Story Events. *Journal of Memory and Language*, 24 (1985).

[28] Wilensky, R. PAM and Micro PAM. In R. Schank and C. Riesbeck (Eds.) *Inside Computer Understanding*. Erlbaum, Hillsdale, NJ, 1981.

[29] Young, R.M. and Riedl, M.O. Towards an Architecture for Intelligent Control of Narrative in Interactive Virtual Worlds. In *Proc. of the 2003 Int. Conf. on Intelligent User Interfaces*, 2003.