Task-Sensitive Cinematography Interfaces for Interactive 3D Learning Environments

William H. Bares   **Luke S. Zettlemoyer**

Dennis W. Rodriguez   **James C. Lester**

Multimedia Laboratory
Department of Computer Science
North Carolina State University
Raleigh, NC 27695-7534
(919) 515-7534   Fax: (919) 515-7896
{whbares,lszettle,dwrodrig,lester}@eos.ncsu.edu

ABSTRACT

Interactive 3D learning environments can provide rich problem-solving experiences with unparalleled visual impact. In these environments, students interactively solve problems by directing their avatars to navigate through complex worlds, transport entities from one location to another, and manipulate devices. However, realtime camera control is critical to their successful deployment. To create effective learning experiences, a *virtual camera* must in realtime "film" their activities in a manner that most clearly depicts the salient aspects of the tasks students are performing. To address this problem, we have developed the *cinematic task modeling* framework for automated realtime task-sensitive camera control in 3D environments. Cinematic task models dynamically map the intentional structure of users' activities to visual structures that continuously depict the most relevant actions and objects in the environment. By exploiting cinematic task models, a cinematography interface to 3D learning environments can dynamically plan camera positions, view directions, and camera movements that help users perform their tasks. To investigate the effect of the cinematic task modeling framework on student-environment interactions, we have constructed a full-scale cinematography interface and a 3D learning environment testbed. Focus group studies suggest that task-sensitive camera planning significantly improves students' interactions with complex 3D learning environments.

KEYWORDS: 3D environments, task models, camera planning, learning environments, educational applications.

INTRODUCTION

Rapid advances in graphics technologies have paved the way for a new generation of 3D learning environments. By enabling students to immersively explore and manipulate 3D worlds depicting complex electronic, mechanical, or biological systems, 3D learning environments can provide rich problem-solving experiences with unparalleled visual impact. With the proliferation of inexpensive graphics accelerators, they are quickly becoming a cost-effective means of delivering customized learning and training, and their captivating game-like qualities may hasten the arrival of systems that blur the distinction between education and entertainment.

Realtime camera control is critical to the successful deployment of dynamic 3D learning environments. Students interactively solve problems by directing their avatars to navigate through complex worlds, transport entities from one location to another, and manipulate devices. To create effective learning experiences, a *virtual camera* must "film" their activities in a manner that most clearly depicts the salient aspects of the tasks they perform. Previous work on 3D camera control either (1) depends on offline operation and does not permit user interactivity [11, 6], (2) requires users to directly control low-level camera positioning and orientation parameters [10, 13, 18, 24], which is problematic when users must perform complex tasks while simultaneously issuing camera control commands, or (3) provides automated realtime camera control [4, 9, 5, 21] but does not take into account the inherent intentionality of users' activities as they perform complex navigational and manipulative tasks.

To address this problem, we have developed the *cinematic task modeling* framework for automated realtime task-sensitive camera control in 3D environments. Cinematic task models dynamically map the intentional structure of users' activities to visual structures that continuously depict the most relevant actions and objects in the environment. Building on our previous work on customized camera planning that considers users' visualization preferences [4], this framework has been implemented in UCam,[1] a user-sensitive realtime camera planner. By exploiting cinematic task models, UCam dynamically plans camera positions, view directions, and camera movements that help users perform by composing

---

[1]User-Customized Automated Montage

Figure 1: The CPU CITY 3D Learning Environment

shots that at each moment present the most clear views of users' navigation and manipulation activities.

To investigate the effect of the cinematic task modeling framework on student-environment interactions, we have constructed a 3D learning environment testbed that provides students with navigational and manipulative controls. In CPU CITY (Figure 1), students learn about the fundamentals of computer architecture by directing their avatars to perform the activities comprising a computation. They navigate through multiple environments of a 3D world housing a virtual computer, they transport "address tickets" to RAM, they retrieve operands from the harddrive, and they execute instructions in the CPU. Focus group studies in which users interacted with the CPU CITY avatar while UCAM automatically planned shot compositions in realtime are encouraging. They suggest that task-sensitive camera planning significantly improves students' interactions with complex 3D learning environments.

CINEMATOGRAPHY IN 3D ENVIRONMENTS
Dynamic 3D learning environments hold great promise for a broad range of educational and training applications. By navigating their avatars through virtual worlds that represent the structure of complex systems and by interactively manipulating entities in these worlds, students can acquire an intimate knowledge—perhaps even deep kinesthetic memories—of the structure and function of complicated devices. Similar benefits may be provided by immersive training systems in which students monitor system behaviors, diagnose faults, and enact repairs. Moreover, the growing body of evidence provided by the entertainment industry, e.g., the phenomenal popularity of the Nintendo 64 series, suggests that avatar-based immersive environments can be enormously engaging and may play a strong motivational role in returning students to a learning environment time and time again.

Realtime Cinematography
As students solve problems in 3D worlds by steering their avatars through expansive landscapes populated by complex architectural structures, learning environments must continuously present views of their activities that are most helpful in assisting the students perform their tasks. Viewing problems are particularly acute in dynamic *unconstrained motion* 3D environments where fast moving, joystick-driven avatars can

go anyplace in the world, move through portals, and pick up, carry, and deposit objects at will in order to accomplish their goals. Unless learning environments can carefully track avatars' activities and, on a moment-to-moment basis, show precisely the aspects of the scene that are most relevant to students' current goals, students can suffer from severe disorientation and be unable to effectively manipulate objects.

Realtime camera planning in 3D learning environments entails selecting camera positions and view directions in response to changes in the avatar's position, orientation, navigation direction, and manipulations of objects. A virtual camera must track activities by executing *cuts* (instantaneous changes from one shot to another without an intervening transition), *pans*, *tracking*, and *zooms* (pull-ins and pull-outs) to make on-the-fly decisions about camera viewing angles, distances, and elevations. Planning camera shots and camera positions while preserving continuity requires solving precisely the same set of problems that are faced by cinematographers, with the additional constraint that they must be solved in realtime.

In dynamic 3D environments, camera planners must continually make decisions about shot types and camera positions. *Shot types* are characterized by the viewing angle, elevation, and distance relative to the subject, which collectively determine the size of the subject as it appears in the frame. For example, an avatar occupies all of the frame for a close-up shot. Different shot types are more useful in particular situations. For example, the camera can be placed slightly below the subject and gaze up towards it to exaggerate its size, or high above the subject gazing downwards to show the scene layout. High and far shots present more information about the scene but tend to be less interesting [14].

State of the Art
Dynamic 3D learning environments call for an approach to camera planning that can represent and draw inferences about the visual structure of information to be presented to users. A growing number of projects have addressed the fundamental problems of multimedia planning [2, 15, 16, 12, 25], automated illustration [22], and animated interface agents [23, 3, 19]. Work has also begun on camera control, for which three approaches have been proposed. In the *offline* approach of ESPLANADE [11] and the "batch" version of DCCL [6], camera planning systems operate in "batch" mode and hence do not permit interactivity. In the *direct control* approach [10, 13, 18, 24], camera systems require users to directly control low-level camera positioning and orientation parameters. This is problematic when users must perform complex tasks while simultaneously issuing camera control commands.

In the *realtime planning* approach [4, 5, 9, 21], camera systems dynamically plan camera position and orientation. While this approach comes closest to delivering the necessary functionalities required of dynamic 3D learning environments, previous work on automated 3D camera planning has not taken into account the inherent intentionality of users' activities. CAMDROID [9] allows the user to design a network of camera modules and constraints but has no task model. The JACK system [18] for simulating the interaction of virtual

humans in an environment employs automatic camera control with occlusion avoidance for smooth visual transitions but also does not employ a task model. The VIRTUAL CINEMATOGRAPHER [21] uses film idioms to successfully maintain camera shot sequences that are consistent with film conventions, but it does not address camera planning for complex user navigation and manipulation tasks in complicated environments. CATHI [5], which is part of the PPP project [3], permits users to state visualization preferences such as the use of spotlights, depth of field, and animation duration, as well as animation preferences that include two cinematic styles. However, the cinematic styles are not based on a representation of the goal structure of users' activities.

DYNAMIC TASK-SENSITIVE CAMERA PLANNING

To enable students interacting with 3D learning environments to focus their attention exclusively on problem-solving activities rather than on directing the camera, 3D learning environments must provide them with an interface that automatically shows the most relevant features of the virtual world at each moment. The interface must make realtime decisions about which subjects in the environment to focus on, how to compose shots that will most effectively depict avatars' activities, and how to plan the smooth tracking and panning of the camera as students navigate the environment and manipulate objects. To address the problem of task-sensitive camera planning, we have developed the *cinematic task modeling* framework for 3D learning environment interfaces. This framework is based on the following premise:

> **Intention-based Cinematography:** By exploiting knowledge about the task a student is performing in a 3D learning environment, a dynamic camera planning system can create a narrative visual structure that reflects the intentional structure of the student's behaviors and thereby support his or her goal-driven problem solving.

In effect, a "virtual director" can take an *intentional stance* [8] toward avatars' actions in the environment. By forming expectations about what students' are seeking to accomplish, it can monitor their progress as they achieve (or fail to achieve) their current goals. Together with precise knowledge about avatars' direction of travel, their location in the world relative to key architectural structures and devices, and their manipulative behaviors, it can use these expectations to dynamically orchestrate effective immersive learning experiences.

Creating an intention-based approach to cinematography for 3D learning environments entails developing (1) a task-sensitive camera planning architecture, (2) a dynamically maintained representation of students' tasks that can be used to produce an immersive experience in which only the most relevant aspects of the world are shown at each moment, and (3) a computational model of task-sensitive shot composition and camera motion planning that uses the intentional representation to support realtime problem solving. Each of these is discussed in turn.

The Task-Sensitive Cinematography Architecture

Camera planning begins the moment a student enters a 3D learning environment. At each tick of the clock, the task-sensitive cinematography system (Figure 2) renders a new frame by employing five central knowledge sources:

- *Task Network:* Encodes a network-based representation of the student's task, where goals are achieved by the student's problem-solving actions, and transition conditions are represented with predicates on a world model and on the avatar state (described below). For example, in the virtual computer world of the CPU CITY testbed, the task network encodes the avatar's sub-goals for operand handling which include obtaining the address of an operand from the CPU, carrying a data value from RAM to the CPU, and inserting a data capsule into a register.

- *Intentional Index Structure:* Semantic indices that map problem-solving goals to shot composition guides. The camera planning system uses knowledge about goals in the task network to index into the shot composition guides that serve as the basis for composing effective views of relevant subjects in the world. For example, manipulative goals are mapped to shot composition guides that can clearly show an avatar's picking up an object or inserting an object into a receptacle.

- *Shot Composition Guides:* Represent abstract shot composition plans for creating shots that will effectively help the student in problem solving by artfully framing the avatar's behaviors and the relevant objects for those behaviors. Shot composition guides, like film idioms in DCCL [6], specify a sequence of one or more shots, each with its own recommended subjects, camera elevation, viewing angle, camera-subject distances, and duration. For example, to help the student locate an object in a landscape, one shot composition guide encodes a high-elevation establishing shot plan for depicting the avatar and the object it is attempting to find.

- *Avatar State:* Represents the avatar's position, orientation, navigation vector, current behaviors, and accouterments. For example, the avatar state of the CPU CITY world might encode its coordinates in the CPU room and indicate that it is walking toward an input register as it carries an address capsule.

- *World Model:* Encodes the layout of the world, including (1) scene geometries, including portal connectivities, (2) 3D models of architectural structures, devices, and portable objects, and (3) the properties and functionalities of manipulable devices and objects. For example, the CPU CITY's world model represents the layout of the buildings representing the CPU, RAM, harddrive, power supply, and portals that connect them, the 3D models for these structures, the locations of all of the devices and objects in the simulated computer (e.g., registers and data capsules), and their properties (e.g., how they are operated and whether they can be picked up).

As the student navigates through an environment and manipulates the objects in it, the system monitors his or her behaviors, updates the cinematic task model, and composes shots. At each tick of the clock, it first translates changes to the joystick's x-y control axes and trigger buttons to the avatar's navigation and manipulative behaviors. After updating the avatar's position, orientation, navigation vector, current behavior, and accouterments, it updates the modified properties of manipulable objects in the world model. Cam-
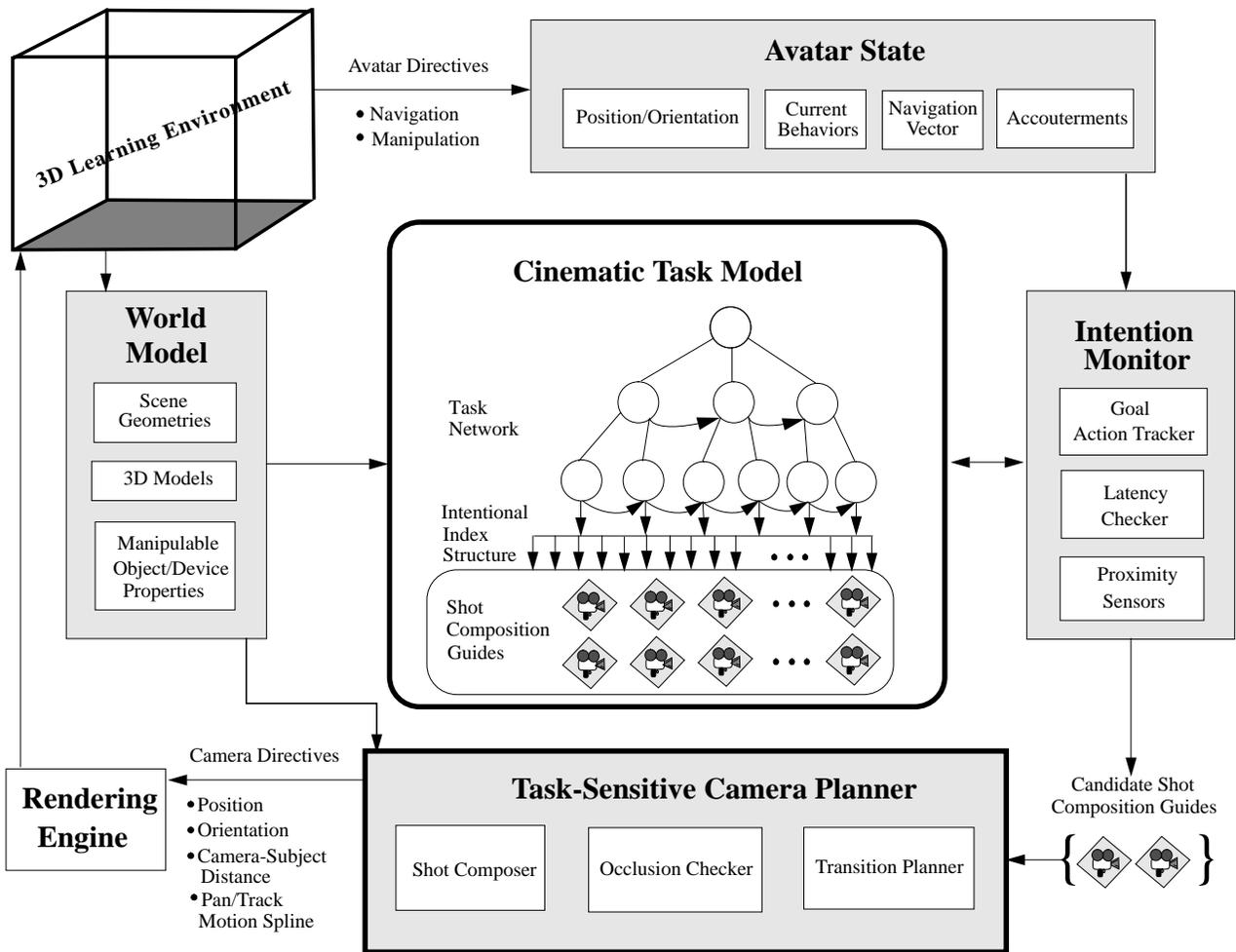
3D Learning Environment

Avatar Directives
- Navigation
- Manipulation

Avatar State

Position/Orientation | Current Behaviors | Navigation Vector | Accouterments

World Model

Scene Geometries

3D Models

Manipulable Object/Device Properties

Cinematic Task Model

Task Network

Intentional Index Structure

Shot Composition Guides

• • •

• • •

Intention Monitor

Goal Action Tracker

Latency Checker

Proximity Sensors

Rendering Engine

Camera Directives
- Position
- Orientation
- Camera-Subject Distance
- Pan/Track Motion Spline

Task-Sensitive Camera Planner

Shot Composer | Occlusion Checker | Transition Planner

Candidate Shot Composition Guides

Figure 2: The Task-Sensitive Cinematography Architecture

era planning begins when the system invokes the *intention monitor*, which maintains the cinematic task model and selects a shot composition guide. The intention monitor checks goal satisfactions, notes proximity sensors that have been triggered, and calls the latency checker. If any goals have been achieved, any proximity sensors have been triggered, or the latency checker indicates a lack of student activity, the intention monitor updates the cinematic task model. It deactivates goals in the task network that have been achieved, enacts the relevant transitions, and updates goal activations. It then uses the intentional indices to index into the shot composition guides and passes the selected guide to the camera planner.

The *camera planner* performs three families of operations: shot composition, occlusion checking, and transition planning. First, it interprets the selected shot composition guide in the current world to create a shot (or sequence of shots) that depicts the focal subjects relevant to the current goal. Second, it checks for occlusion constraint violations. By inspecting the geometry of the current scene, it determines if structures in the scene block the view of the focal subjects. If so, it selects an alternate viewing angle. Third, it plans shot transitions that either cut from one shot to another or,

more likely, smoothly track and pan the camera to the next viewpoint.

The product of these computations is a set of executable camera directives specifying *shot type* (e.g., close-up, long), *viewing angle* (e.g., front-left), *viewing elevation* (e.g., high, medium, or low), *transitions* (cut, tracking, panning), *shot duration*, and *panning* and *tracking speeds* to depict the focal subjects. These directives are then passed to the renderer, which composes the next frame depicting the 3D environment. The net effect of viewing these rapidly rendered frames is an immersive experience that is customized for students' tasks.

Cinematic Task Modeling

Cinematic task modeling is the process of tracking students problem-solving activities and determining the most appropriate types of shots to help them achieve their goals. By observing students' avatars' behaviors and exploiting knowledge of the task that they are performing, learning environments can infer their intentions and recommend plans for shot compositions. A cinematic task model must provide a solution to each of the following problems: (1) how to represent the student's task, (2) how to infer the student's intentions

from the behaviors of their avatars, and (3) how to select an abstract shot composition plan that, when interpreted in the context of the current scene, will create a view of the world that helps the student achieve their current goal.

Procedural Task Networks A representation of students' tasks should easily encode procedural task knowledge, permit rapid updates for realtime performance, and be simple and extensible to facilitate the economic deployment of 3D learning environments. In contrast to more complex and labor-intensive approaches to task modeling such as task-action grammars (TAGs) [17] or production systems [1], the cinematic task modeling framework encodes task knowledge in a network-based representation reminiscent of early planning systems [20]. *Task networks* represent procedures as a network of goals and the temporal relations holding between them. While task networks lack the flexibility of TAGS, production systems, and sophisticated planning representations, they offer a practical alternative for 3D learning and training environments, in large part because of their ease of construction and maintenance for procedural tasks [19].

Task networks encode the intentional structure of the problem-solving activities students perform with their avatars. Each goal node contains a representation of (1) its supergoal, (2) participants, (3) successor goals, (4) goal type, (5) conditionalized transitions (expressed as predicates on the world model and the avatar state) from predecessor goals to the current goal and (6) from the current goal to successor goals, (7) relevant actions, (8) side effects (optional), (9) pedagogical advisory overlays, which consist of brief textual reminders about the current goal, and (10) latency, which denotes the time threshold above which it can be inferred that the student is not making sufficient progress towards achieving the goal. For example, Figure 3 shows a `Put-In-Receptacle` goal from the CPU CITY learning environment. It is a manipulative goal, i.e., achieving it requires the avatar to manipulate an object or device. Its participants include an *actor* (an avatar named Whizlo), an *object* (a data capsule), and a *receptacle* (a register in the CPU). For the goal to become active, its entrance transition conditions must be satisfied. In this case, Whizlo must be in the CPU room, and he must be carrying a particular data capsule. After he performs the relevant action (inserting the data capsule into a specific register), the exit transition conditions are asserted (the fact that the data capsule occupies the register), as are the side-effects (the data capsule disappears from the input register, and the result value appears in the output register). The pedagogical advisory overlay specifies a relevant help message, and the latency indicates that the student should spend a maximum of 15 seconds to achieve the goal.

Cinematic Intention Monitoring Dynamically creating views of the world in order to help students perform their tasks requires the learning environment to recognize their intent, but plan recognition is a notoriously difficult problem [7]. To address it, the cinematography system exploits the procedural knowledge encoded in the task network and, in Andersonian model-tracing style [1], carefully tracks students' goals and

Put-In-Receptacle(Whizlo, Data-Capsule-B, CPU-Input-Register-2)
SUPER-GOAL: Operand-Handling
GOAL-TYPE: Manipulative
PARTICIPANTS: Actor: Whizlo
                Object: Data-Capsule-B
                Receptacle: CPU-Input-Register-2
ENTRANCE-TRANSITION-CONDITIONS:
        In-Room(Whizlo, CPU)
        Carrying(Whizlo, Data-Capsule-B)
PEDAGOGICAL-ADVISORY-OVERLAY:
        "Place the value of B in Register 2."
ACTIONS: Insert(Whizlo, Data-Capsule-B,
        CPU-Input-Register-2)
SUCCESSOR-GOAL:
    Remove-From-Receptacle(Whizlo, Result-Value,
                CPU-Output-Register)
EXIT-TRANSITION-CONDITIONS:
    In-Receptacle(Data-Capsule-B,
            CPU-Input-Register-2)
SIDE-EFFECTS: Disappear-In-Receptacle(Data-Capsule-B,
                CPU-Input-Register-2)
        Appear-In-Receptacle(Result-Value,
                CPU-Output-Register)
LATENCY: 15 seconds

Figure 3: A goal node from the CPU CITY task network

doesn't permit them to stray far from the correct solution path. These functionalities are provided by the intention monitor. As students direct their avatars through a learning environment, the intention monitor inspects the world model and the avatar state to dynamically maintain the cinematic task model while formulating shot composition recommendations for the camera planner. It performs three families of tasks:

- *Goal Activation Tracking:* At each tick of the clock, the intention monitor examines the world model and the avatar state to determine if the *current goal* G_C should be updated to its successor goal $G_{C'}$. If the avatar has previously satisfied G_C's entrance transition conditions and, since the last clock tick, it has performed the actions associated with G_C, then G_C is deactivated, its exit transition conditions are asserted, and $G_{C'}$ is activated, indicating that the student is now poised to perform the relevant actions to achieve $G_{C'}$. For example, if a student's current goal is to pick up a data capsule from the harddrive, then he/she will transition to the successor goal when `Pick-Up(Whizlo, Data-Capsule-A)` becomes true. The camera planner will use its knowledge of goal activations to compose shots that clearly depict the participants of the current goal.

- *Latency Checking:* At each tick of the clock, the intention monitor performs two types of latency checking to assess the student's progress. First, it conducts a *goal latency check* to determine if the student is making satisfactory progress towards achieving the current goal G_C. If the clock indicates that the time spent by the student to achieve the active goal exceeds G_C's permissible latency, a latency

violation is marked. For example, if the student does not achieve the `put-in-receptacle` goal shown in Figure 3 before 15 seconds of effort, the intention monitor marks a latency violation. Second, it conducts an *idle latency check* to determine if the student has continued to be active in problem solving. If the clock shows that the student has been inactive for a period of time longer than idle threshold, an idle-time-exceeded violation is marked. The camera planner will use its knowledge of latency and idle time violations to compose shots that will be helpful to students who may be experiencing problem-solving impasses (as described below).

- *Proximity Sensors:* To gauge the student's progress at a finer granularity, the intention monitor employs proximity sensors. When the student steers his or her avatar to a region surrounding an object that is a participant in the current goal, a proximity sensor in the intention monitor is triggered. Proximity sensors mark the achievement of *silent goals*. These are implicit goals that do not formally belong to the task but nevertheless suggest progress towards explicit "first class" goals that do. For example, a silent goal preceding a `pick-up-object` goal is achieved when a student directs her avatar to a region guarded by a proximity sensor attached to the relevant object in the environment. If it is not achieved in a reasonable period of time, a latency violation is marked so the camera planner can provide compose appropriate shots; in contrast, when it is achieved, it offers strong positive evidence that the student is making progress towards the first class goal.

Realtime Task-Sensitive Camera Planning
The intention monitor performs its goal activation tracking, latency checking, and proximity sensor activities to enable the camera planner to make informed task-sensitive decisions about how to best depict the world in order to assist the student's problem solving. Taking advantage of cinematography conventions that have developed over the past century [14], the camera planner operates in three phases to plan shots and transitions between them:

1. **Goal-Based Shot Composition:** The intention monitor exploits the results obtained from its goal tracking and other activities, together with intentional indices, to find a set of one or more shot composition guides.

2. **Scene Interpretation:** The camera planner selects one of the shot composition guides and interprets it in light of the current scene geometry to avoid potential occlusions.

3. **Camera Motion Planning:** The camera planner uses the selected shot composition guide, together with cinematic continuity conventions, to plan (a) the motion splines for tracking and panning and (b) cuts from one shot to another.

Goal-Based Shot Composition In the shot composition phase, the intention monitor uses the type of the activated goal and latency violations to select one or more candidate shot composition guides. Problem-solving goals in 3D learning environments are of two types: navigational and manipulative. If the student is attempting to achieve a *navigational* goal, the intention monitor will bias its selection toward views of the world that will help the student reach his or her destination. Because destinations are represented as features of navigational goals and the world model encodes scene geometries and locations of all entities, the system can compose views that clearly depict the relevant architectural structures, devices, and objects. If the student is making adequate progress towards achieving a navigational goal, the intention monitor will propose shot compositions that feature the avatar moving through the environment. In contrast, if the student is experiencing difficulties in achieving the goal, as indicated by latency violations, the intention monitor will propose shot compositions that clearly show the location of destination objects, either in a single shot with the avatar or with alternating cut shots between the avatar and the destination structure, device or object. For example, if a student interacting with the CPU CITY environment has difficulty finding one of the "keys" to the CPU, he will be shown views of the avatar and the key (if they are in close proximity) and alternating cut views of the avatar and the location where the key resides. The intention monitor also employs wide and high establishing shots at the beginning of interactions to set the stage for problem solving.

If the student is attempting to achieve a *manipulative* goal, the intention monitor should propose shot compositions that feature views of the avatar and the subject it is manipulating. If the manipulative goal involves both an object and a device, views depicting all of the participants should be composed. For example, if the task model suggests that the student will soon be attempting to insert a data capsule into a register, the intention monitor should compose a shot that clearly depicts all three of the participants (avatar, data capsule, and register).

Scene Interpretation The result of the goal-based shot composition phase is a set of one or more instantiated *shot composition guides*. A shot composition guide specifies the number of shots in a sequence. For each shot, it specifies a camera elevation, a viewing angle, the desired distance of the camera from the "primary" subject (typically the avatar), and the proposed duration for the shot in milliseconds. Table 1 summarizes the types of shot compositions that UCAM currently employs. For example, an establishing shot in the CPU CITY testbed shows a far, wide view of the relevant characters (the avatar), manipulable objects ("keys" to gain entrance to the CPU), and architectural structures (the RAM, harddrive, etc.). During the scene interpretation phase, the camera planner randomly selects one of the chosen candidate shot composition guides and then interprets each of its shots in the current scene. To obtain an unobstructed view of the avatar and the relevant structures, objects, and devices (indicated by the participants in the current goal), it performs a series of occlusion evaluations by raycasting for obstacles between the camera and avatar. If occlusion is detected, it modifies camera shots by gradually sweeping around the avatar until a clear view is found.

Camera Motion Planning Maintaining visual continuity across time is critical for creating a coherent immersive learning experience. However, achieving visual continuity

Shot Composition Guide	View Summary
Establishing	One view to include avatar and relevant objects, prefer high elevations
Opposing Dramatic	View of subject A, cut to view of subject B, prefer tighter, lower views
Opposing Informative	View of subject A, cut to view of subject B, prefer higher, farther views
Tracking 2 Angles	2 views of moving avatar, different angles
Circling 4 Angles	4 views swinging around subject
Manipulative Left	Rear left view of avatar with object
Manipulative Right	Rear right view of avatar with object

Table 1: UCAM shot composition guides

in 3D environments, particularly complex avatar-based environments which require a variety of camera shots, is difficult. Camera motions (panning, tracking, zooming) cannot be chosen arbitrarily without producing visual discontinuities [14], nor can they abruptly cut from shot to shot without introducing confusing context switches. To address this problem, the camera planner uses the recommendations of the selected shot composition guide to makes its decisions. For example, some shot composition guides such as opposing views force cut transitions between their shots. If not, the camera planner computes a motion spline for the camera to travel between its current position and that needed for the newly selected shot and estimates the amount of time required for the camera to travel that path and then (1) evaluates camera positions along the path for occlusions, which if found, cause a cut to the new shot. (2) It evaluates intermediate positions for visual awkwardness, e.g., it checks for the appearance of shots directly over the head of the avatar, which are frequently confusing in the context of a fast track/pan. (3) Because jump cuts from one shot to another which is only slightly different produces a jarring effect [14], it computes the the angular difference between the current and new shots and opts for a track/pan if this is excessively small. The intention monitor and the camera planner perform all of the tasks above at each tick of the clock to produce a continuous immersive experience as the student interactively solves problems in the learning environment.

IMPLEMENTATION AND EVALUATION

UCAM is a full-scale realtime implementation of the cinematic task modeling framework.[2] To investigate UCAM's

---

[2]UCAM is implemented in C++ and employs the OpenGL graphics library for 3D rendering. It runs at 15 frames/second with texture mapping in a 640x480 full-screen window on a 200 Mhz Pentium Pro computer running Windows NT 4.0 with a Permedia 2 3D graphics accelerator and 64 MB memory. UCAM and the CPU CITY testbed consist of approximately 27,000 lines of code.

behavior, we constructed CPU CITY, a 3D learning environment that provides immersive learning experiences to non-technical students learning about computer architecture. CPU CITY is built from more than 50 3D models and employs a large but linear task network.

The student is initially set with the task of gathering the two keys that will unlock the entrance to the CPU (and other structures) so that they can begin to perform the operation (C = A - B). This involves finding and picking up first a red key, then a blue key, and then placing them into their respective receptacles. At this time the student will be able to enter the CPU where they will need to get the first memory address from the control unit. The memory addresses refer to the locations in RAM that the actual values that the operation will perform on reside. The student then takes the first memory address and carries it to RAM where it must be deposited into the "IN" slot. This will cause the value of that memory address to appear in the "OUT" slot. Similar activities are used to obtain the value of B from the harddrive. Finally, the student stores the result of this computation, C, in RAM.

To gauge the effectiveness of task-sensitive cinematography, we conducted a focus group study involving 12 subjects. Each subject interacted with two versions of UCAM. In the first version, all of the automated camera control was disabled and the camera was fixed relative to the avatar's coordinate system with "behind-the-back" shots; if subjects wanted to change the viewpoint, they had to do it manually with a joystick or menu-based interface. In the second version, the full functionality of task-sensitivity was in play. Results of the study suggest that task-sensitive camera planning can improve learning environment interactions. Establishing shots proved very useful in reorienting the student as they attempted to achieve navigation goals. The automatic occlusion avoidance was very favorably received. Although most subjects were not patient enough to remain idle for the 2 seconds needed for the automatic establishing shots and alternating cut "hint" shots to come into play, it appears that with a lower threshold they will be very helpful. Perhaps most telling of the results were that subjects rarely if ever took control of the camera in either version—it was simply too complicated to attend to both the learning task and camera control parameters.

The study also suggested several areas for future work. These include reducing the use of cut transitions which tend to distract students, allowing students to trigger establishing shots to depict an overall "map" view, enabling students to express specific preferences to direct the automatic camera planning, and improving shot quality when viewing multiple objects simultaneously.

In summary, intention-based cinematography holds much promise for interactive 3D learning environments. By exploiting knowledge about the task a student is performing, a dynamic camera planning system can create a narrative visual structure that reflects the intentional structure of the student's behaviors and effectively support his or her goal-driven problem solving.

REFERENCES

1. J. Anderson, A. Corbett, K. Koedinger, and R. Pelletier. Cognitive tutors: Lessons learned. *Journal of the Learning Sciences*, 4(2):167–207, 1995.

2. E. André, W. Finkler, W. Graf, T. Rist, A. Schauder, and W. Wahlster. WIP: The automatic synthesis of multimodal presentations. In M. T. Maybury, editor, *Intelligent Multimedia Interfaces*, chapter 3. AAAI Press, 1993.

3. E. André and T. Rist. Coping with temporal constraints in multimedia presentation planning. In *Proceedings of the Thirteenth National Conference on Artificial Intelligence*, pages 142–147, 1996.

4. W. H. Bares and J. C. Lester. Cinematographic user models for automated realtime camera control in dynamic 3d environments. In *Proceedings of the Sixth International Conference on User Modeling*, pages 215–226, 1997.

5. A. Butz. Anymation with CATHI. In *Proceedings of the Ninth Innovative Applications of Artificial Intelligence Conference*, pages 957–62, 1997.

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

7. J. Chu-Carroll and S. Carberry. A plan-based model for response generation in collaborative task-oriented dialogues. In *AAAI-94: Proceedings of the Twelfth National Conference on Artificial Intelligence*, volume 1, pages 799–805, 1994.

8. D. C. Dennett. *The Intentional Stance*. MIT Pres, Cambridge, Massachusetts, 1987.

9. S. Drucker and D. Zeltzer. CamDroid: A system for implementing intelligent camera control. In *Proceedings of the 1995 Symposium on Interactive 3D Graphics*, pages 139–144, 1995.

10. M. Gleicher and A. Witkin. Through-the-lens camera control. *Computer Graphics*, 26(2):331–340, 1992.

11. P. Karp and S. Feiner. Automated presentation planning of animation using task decomposition with heuristic reasoning. In *Proceedings of Graphics Interface '93*, pages 118–127, 1993.

12. S. Kerpedjiev, G. Carenini, S. Roth, and J. Moore. Integrating planning and task-based design for multimedia presentation. In *Proceedings of the Third International Conference on Intelligent User Interfaces*, 1997.

13. J. Mackinlay, S. Card, and G. Robertson. Rapid controlled movement through a virtual 3D workspace. In *Proceedings of ACM SIGGRAPH '90*, pages 171–176, 1990.

14. J. Mascelli. *The Five C's of Cinematography*. Cine/Grafic Publications, Hollywood, 1965.

15. K. R. McKeown, S. K. Feiner, J. Robin, D. Seligmann, and M. Tanenblatt. Generating cross-references for multimedia explanation. In *Proceedings of the Tenth National Conference on Artificial Intelligence*, pages 9–15, San Jose, CA, 1992.

16. V. Mittal, S. Roth, J. D. Moore, J. Mattis, and G. Carenini. Generating explanatory captions for information graphics. In *Proceedings of the International Joint Conference on Artificial Intelligence*, pages 1276–1283, 1995.

17. S. Payne and T. Green. Task-action grammars—a model of the mental representation of task languages. *Human-Computer Interaction*, 2:99–133, 1986.

18. C. Philips, N. Badler, and J. Granieri. Automatic viewing control for 3D direct manipulation. In *Proceedings of the 1992 Symposium on Interactive 3D Graphics*, pages 71–74, 1992.

19. J. Rickel and L. Johnson. Integrating pedagogical capabilities in a virtual environment agent. In *Proceedings of the First International Conference on Autonomous Agents*, pages 30–38, 1997.

20. E. D. Sacerdoti. *A Structure for Plans and Behavior*. Elsevier, New York, 1977.

21. D. Salesin, L. wei He, and M. Cohen. The virtual cinematographer: A paradigm for automatic real-time camera control and directing. In *Proceedings of ACM SIGGRAPH '96*, pages 217–224, 1996.

22. D. D. Seligmann and S. Feiner. Supporting interactivity in automated 3D illustrations. In *Proceedings of Intelligent User Interfaces '93*, pages 37–44, 1993.

23. B. A. Stone and J. C. Lester. Dynamically sequencing an animated pedagogical agent. In *Proceedings of the Thirteenth National Conference on Artificial Intelligence*, pages 424–431, 1996.

24. C. Ware and S. Osborn. Exploration and virtual camera control in virtual three dimensional environments. In *1990 Symposium on Interactive 3D Graphics*, pages 175–184, 1990.

25. M. X. Zhou and S. K. Feiner. Top-down hierarchical planning of coherent visual discourse. In *Proceedings of the Third International Conference on Intelligent User Interfaces*, 1997.