From Abstraction to Reality: Integrating Drama Management into a Playable Game Experience

Anne Sullivan, Sherol Chen, Michael Mateas

Expressive Intelligence Studio
University of California, Santa Cruz
{anne, sherol, michaelm} @ soe.ucsc.edu

Abstract

Often, video game designers must choose between creating a restrictive, linear experience and designing an open world with many different story lines that fail to form a tightly crafted narrative arc. A drama manager (DM) can provide a solution to this dilemma. A DM monitors an interactive experience, such as a computer game, and intervenes to shape the global experience so that it satisfies the author's expressive goals without decreasing a player's interactive agency. Previous work with declarative optimization-based drama management (DODM) has focused on simulated player studies in abstract game worlds. In this paper, we present the integration of DODM into a real-time adventure-style dungeon game called EMPath. We describe the game world as well as the modifications that had to be made to the DM as the result of integrating with a concrete, playable experience. These changes include the development of new story evaluation features, improving drama management move refiners and creating a new player model. We also describe the results of a play test comparing player experiences with the drama manager on and off.

Introduction

When creating a video game, designers are faced with the decision of how to integrate a compelling and cohesive story line for the player to experience. Currently, most commercial games tend towards one of two strategies. The first solution involves producing a fairly linear experience which allows great control over the player's choices, ensuring a cohesive story. The second strategy is generating a very open world that allows the player many options but allows for less control and, therefore, less cohesion within the experience.

A drama manager (DM) is an alternative solution to this dilemma. A DM makes it possible to balance authorial goals without decreasing the player's agency. It accomplishes this by monitoring the game play and then intervening to shape the global experience.

Drama management was first proposed in the context of interactive drama (ID), dramatic worlds in which players experience a dynamic story arc and interact with socially capable, personality-rich autonomous characters. In addition to drama management, the creation of such experiences introduces many design and technology challenges such as autonomous characters, dialog management, and natural language understanding and generation.

Because of the complexity related to creating complete, playable IDs, much of the drama management literature has focused on technical proofs of concept that are not integrated into a playable game, making it difficult to assess whether the DM is positively affecting player experiences. However, drama management can be applied more generally than to IDs; it can be applied to any interactive experience in which the author would like to express preferences for sequences with specific properties while still allowing player agency and non-linear events. Within existing game genres, drama management can perform tasks such as non-linear mission, encounter, and player discovery sequencing.

Drama management can support a level of non-linearity in existing game genres that would be difficult to impossible to achieve using the traditional approach of scripting and local triggers. Additionally, drama management opens up new game design possibilities that would traditionally be complicated or impossible to conceive of without the capabilities of a DM.

In this paper we report on the application of declarative optimization-based drama management (DODM) to a *Zelda*-style adventure game called EMPath. An advantage of using a traditional genre game is that it allows us to directly compare the player's experience in the game with and without the DM active. Such comparisons are not possible for emerging genres such as interactive drama that may depend on the presence of a DM to even be coherent.

We then describe related DM work, present the DODM framework, discuss the design of EMPath, explain the extensions we made to DODM to support EMPath, and present the results of our user tests. Finally, we discuss the changes we made to our player models based on the results of our player tests.

Related Work

Search-based drama management (SBDM) was first proposed by Bates (1992) and developed by Weyhrauch (1997). Weyhrauch applied SBDM to *Tea for Three*, a simplified version of the Infocom interactive fiction *Dead-*

line; achieving impressive results in an abstract story space with a simulated user. Lamstein & Mateas (2004) proposed reviving this work.

More recent work has generalized SBDM as DODM, for which search is one optimization method. Other optimization methods have been tried, including reinforcement learning, and an alternative formulation of the DM problem as targeted trajectory distribution Markov decision processes (TTD-MDPs) (Roberts et al. 2006). Nelson and Mateas (2005) have compared the search and TTD-MDP formulations of DODM.

The *Mimesis* architecture (Young and Reidl 2003) constructs story plans and monitors for player actions that might threaten causal links in the story plan. If a threat is a detected, the architecture re-plans or prevents player action. The *Interactive Drama Architecture* (Magerko 2005) strives to keep the user on a prewritten storyline by taking corrective action guided by a state-machine model of likely player behavior. In contrast to both approaches, DODM seeks to incorporate player action into a dynamic plot, rather than guide player action onto a pre-written plot.

Both the beat-based DM in *Façade* (Mateas and Stern 2003) and the *PaSSAGE* system (Thue et al. 2007) employ a content selection model of drama management. In *Façade*, the beat-based DM maintains a probabilistic agenda of dramatic beats. Each beat coordinates autonomous characters in carrying out a bit of dramatic action while supporting player interaction during the beat. Similarly, *PaSSAGE* contains a library of character encounters in a role-playing game, dynamically selecting the next encounter as a function of a model of the player. Unlike DODM, neither system projects future story arcs to make a decision.

Thespian (Si, Marsella, and Pynadath 2005) and *U-Director* (Mott and Lester 2006) both employ a decision-theoretic approach to DM. In *Thespian*, decision-theoretic goal driven agents select actions to maximize goals specified as reward functions over state. DM is thus decentralized across characters. *U-Director* employs a centralized DM that selects actions to maximize narrative utility, measured with respect to narrative objectives, story world, and player state. *U-Director* shares with DODM the ability to trade off among multiple, potentially contradictory narrative features.

DODM has traditionally been evaluated in discrete textual spaces. In contrast, we explore applying DODM to a graphical game world. Ontañón, et al. (2008) take a similar approach, creating a graphical representation of the *Anchorhead,* but maintaining the discrete game world and discrete time used in the interactive fiction. In this system, the player interacted with the story using typed commands which were interpreted with an NLU module. The number of actions the player could perform was greatly increased by this, and their system uses a hybrid montecarlo expectimax algorithm to focus the DM searches. This reduces the amount of game tree searches the DM performs, allowing it to respond in real time. *EMPath* has significantly different game mechanics from *Anchorhead*, thefore requiring more complicated applications of the abstract drama manager actions to handle the real-time nature of the game.

The DODM Framework

To configure DODM for a specific world, the author specifies *plot points*, *DM actions*, and an *evaluation function*. Plot points are important events that can occur in an experience. Different sequences of plot points define different player trajectories through games or story worlds. Examples of plot points include a player gaining story information or acquiring an important object. The plot points are annotated with ordering constraints that capture the physical limitations of the world, such as events in a locked room not being possible until the player gets the key. Plot points are also annotated with information such as where the plot point happens or what subplot it is part of. The exact set of annotations is flexible and depends on the story.

DM actions are actions the DM can take to intervene in the unfolding experience. Actions can *cause* specific plot points to happen, provide *hints* that make it more likely a plot point will happen, *deny* a plot point so it cannot happen, or *un-deny* a previously denied plot point.

The evaluation function, given a total sequence of plot points that occurred in the world, returns a "goodness" evaluation for that sequence. This evaluation is a specific, author-specified function that captures story or experience goodness for a specific world. While an author can create custom story features, the DODM framework provides a set of features that are commonly useful in defining evaluation functions.

We used the following four features for the EMPath story evaluation function: *Motivation*, *Thought Flow*, *Manipulation*, and *Story Density*, *Motivation* measures whether events that happen in the world are motivated by previous events. For example, in EMPath, the player can learn from a note that they must acquire a candle before leaving the dungeon; if they encounter this note before finding a boss monster in a room with candles, then the encounter with the boss is motivated.

Similarly, *Thought Flow* measures the degree to which plot points associated with the same topic appear together. In EMPath, this feature prefers stories in which plot points associated with each of the quests are grouped together without interleaving.

Story Density is a measure of how tightly grouped activated plot points occur. In *EMPath* it is desirable for plot points to happen at a steady pace – not all at once, nor largely spread apart.

Manipulation measures how coercive the DM is in a given story sequence. Manipulation costs are associated with each DM action rather than being associated with plot points. Manipulation counter-balances other features, forcing the DM to take into account manipulation costs when optimizing other features. The author associates plot points or DM actions with specific evaluation features.

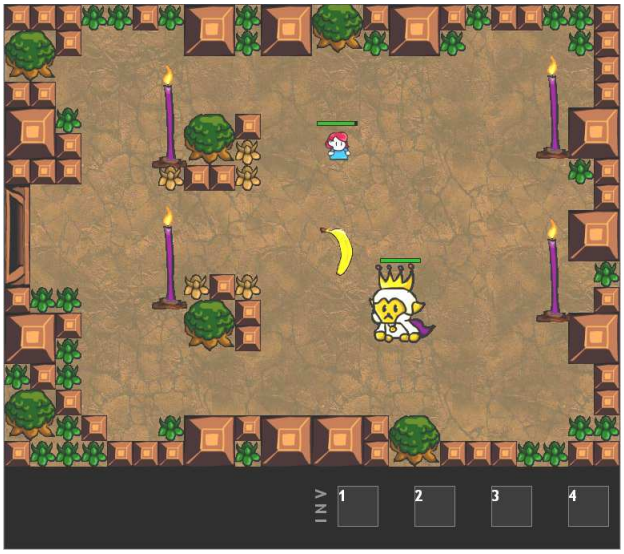

Figure 1: Screenshot of EMPath. The player (in blue) is in the monkey king's room avoiding the thrown bananas.

When DODM is connected to a concrete game world, the world informs the DM when the player has caused a plot point to happen. The DM then decides whether to take any action, and tells the world to carry out that action. In EMPath, for example, the player may be moving between rooms and fighting non-boss monsters. When the player finds a note describing a nearby prisoner, the game tells the DM that a plot point has happened. The DM might then choose the "deny candle location information" action, telling the game world to remove a specific note.

Given this model, the DM's job is to choose actions (or no action at all) after the occurrence every plot point so as to maximize the future goodness of the complete story. For EMPath, we perform this optimization using game tree search in the space of plot points and DM actions, using expectimax to backup story evaluations from complete sequences. In order to perform this look-ahead search, DODM requires a player model to predict future player action at the plot point level.

For more information on the DODM model, see Nelson, Roberts and Isbel (2006) and Nelson et al. (2006).

EMPath

EMPath (Experience Managed Path) is a classic *Zelda*-like adventure game that we developed specifically to test DODM in a traditional game genre. EMPath was designed to be small, so that it can be completed quickly and players can experience a clear sense of progression and completion from beginning to end. It also was designed to be playable without the DM, yet amenable to drama management.

Game Description

The EMPath world is a 25 room dungeon populated with enemies, fire traps, a prisoner, and two bosses. The goal is to reach the stairs and escape from the dungeon, however the stairs are too dark to traverse and blocked by rubble; the player must find the special items required to escape.

There are two quest lines in the game. The first involves finding a candle so that the player can see their way up the stairs. There are notes that describe the location and use of the candle. The candle is in the possession of the monkey king, whom the player must kill to acquire the candle.

The second quest line revolves around a prisoner, who possesses a magic talisman that can clear the rubble blocking the stairs. A note in the game world informs the player of the existence of a prisoner imprisoned in a jail cell in the dungeon. When the player finds the prisoner, they are told that the large guard next door (boss) has the key. Once killed, the guard will drop the key to the cell. When the player frees the prisoner, they receive the magic talisman.

Plot Point	Description
1. info_loc_candle Thought: candle	player receives information about the location of the candle
2. info_use_candle Thought: candle	player receives information about the use of the candle
3. king_dead Thought: candle Motivation: plot point1, plot point2	player kills the monkey king
4. get_candle Thought: candle Motivation: plot point3	player gets the candle
5. info_key_guard Thought: prisoner	player receives information that the guard holds the key

Table 1: Sample set of plot points used in EMPath.

Plot Points

The game flow described above is the player's experience when the drama manager is off. As in standard adventure and RPG game design, some of the event sequences are linearized via trigger logic, while others can appear in different orders, including orders that are not as satisfying.

For instance, reading a clue in a note after the player has already accomplished the task is a not uncommon experience in contemporary games. In order to create the drama managed version of the game, it is necessary to decouple significant game events from their default realization in the non drama-managed game. This is accomplished by defining plot points.

Plot points are the significant events that influence the player's overall experience in the game. We defined ten plot points for EMPath; five of them appear in Table 1. Each plot point is annotated to support the evaluation features, in this case thought flow and motivation. Note that plot points are defined so as to generalize them from their

DM Action	Description
mob_drop_loc_candle Type: hint, Manip: 0.2	Info_loc_candle note drops off next enemy killed
room_drop_loc_candle Type: hint, Manip: 0.1	Info_loc_candle note shows up in next room
npc_talk_loc_candle Type: causer, Manip: 0.6	Enemies walk in chatting about info_loc_candle
deny_loc_candle Type: denier, Manip: 0.1	Temporarily remove info_loc_candle note from game (other hints and causers disabled)
reenable_loc_candle Type: reenabler, Manip: 0.0	Adds info_loc_candle note back into the game and re-enables other hints and causers
king_attack_player Type: causer, Manip: 0.9	Causes king_dead by having the monkey king chase down the player, following from room to room

Table 2: Sample set of DM Actions used in EMPath

default implementations. For example, instead of a plot point "find note about candle location", there is a plot point "learn candle location" as there are multiple ways to learn about the candle location.

Plot points are organized into a directed acyclic graph (DAG) to capture ordering dependencies that are enforced by the physical logic of the world. In EMPath, we had very few dependencies; **get_key** is required before **get_talisman** can happen and **get_candle** and **get_talisman** are prerequisites for exiting the dungeon. Most plot points can appear in any order; however the evaluation function will encode a preference for certain orderings.

DM Actions

For each plot point, we defined a list of actions the DM can perform to influence plot point occurrence. For each plot point, we considered hints, causers, and deniers, defining a total of 33 DM actions. Six example actions appear in Table 2. We include manipulation costs, which are used by the manipulation evaluation feature.

Game Design Challenges

One of the biggest challenges in creating a drama managed game is authoring multiple story lines. For example, events can happen in many possible orders and in many possible ways, physical objects may move, and connectivity of map areas may change. Accomplishing this level of dynamism with purely local trigger logic would be a Herculean task. For each plot point, it would be coded as local tests that somehow looked at the history of what happened so far, and trading off between the various evaluation features, e.g. exactly when the note should be discovered may involve trade-offs between thought flow and motivation.

The difficulty with trigger logic is that it does not gracefully scale to handling a large amount of interacting state in making decisions, and, more critically, *can only consider the past, not the effect of actions on possible futures*. DODM can take into account the interactions between various decisions, and, by projecting possible futures, it is able to weigh the costs and benefits of making various game world interventions. However, the game designer must still design how the DM actions affect the detailed game state, also called the *refinement* of DM actions.

For example, what happens when the player kills the monkey king if the DM has chosen a different method to deliver the candle to the player? As this is clearly a boss battle, having nothing happen would confuse the player, so the designer should provide content to handle this situation. All sequencing events related to the candle – whether the player received all the information before finding the candle – were left to the DM. However, we did have to worry about the details of *how* the player finds the candle. The designer must implement the content in the game that handles how these details interact with other events in the game.

Finally, plot points must be carefully chosen so as to provide the DM with a view of all the events important to the flow of the experience. Without knowledge of all the significant events, the DM will not be able to reason about where to fit it into the experience, nor have actions associated with them.

DODM Adjustments

Prior DODM research has focused on abstract game models defined purely in terms of plot points and DM actions. It has been previously believed that everything to do with sequence-level or story-level importance in the game world was captured by the DAG. However, when connecting DODM to a concrete game world for the first time, we had to modify DODM to account for aspects of the player's experience influenced by the physical layout of the world. To incorporate the new spatial importance of the world, we created a new evaluation feature, story density.

Story Density

Before EMPath, all the DODM evaluation features we developed had no dependencies on physical details of the game world. Consider the three prior evaluation features that we are reusing for EMPath: thought flow, motivation and manipulation. Values for these three features are computed based entirely on static annotations on plot points and DM actions. As we defined the evaluation function for EMPath, however, we realized that none of the features in our toolbox adequately capture a notion of *story density*.

Story density is a measure of how many plot points happen per unit time. As authors, we felt that for EMPath long periods of wandering between plot points should get a lower evaluation score, as should plot points happening too close together. For a room-based dungeon crawl like EM-

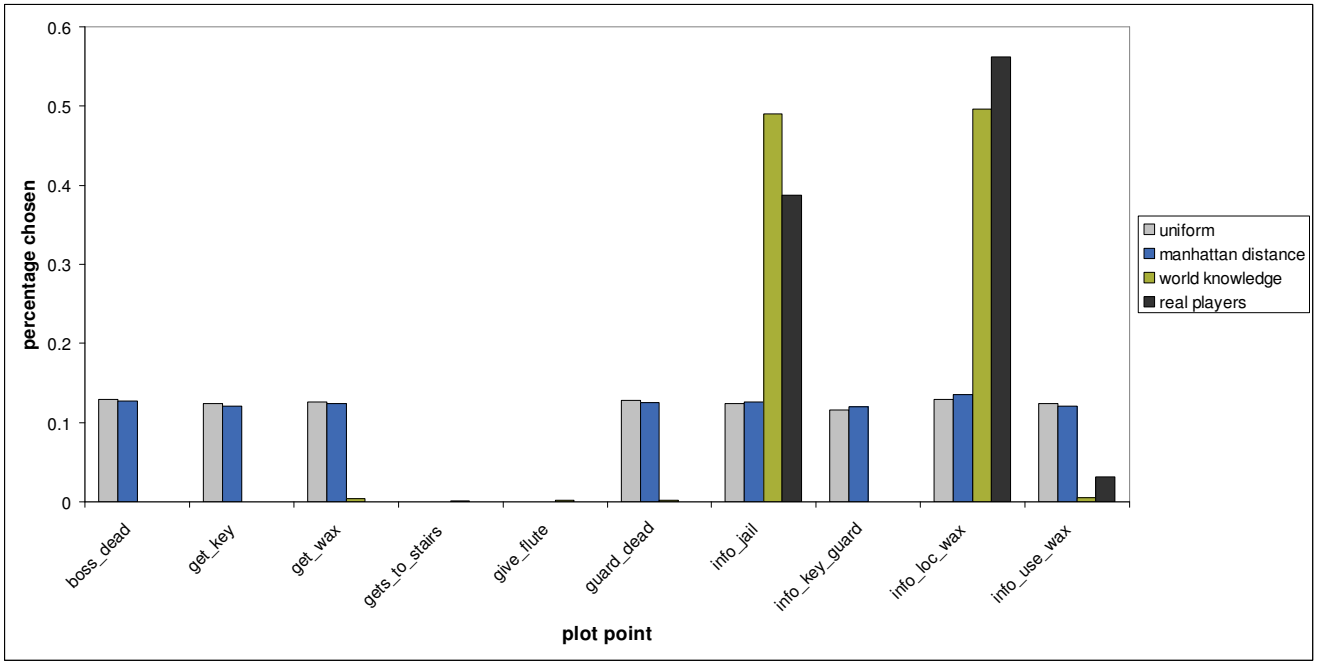

Figure 2: The world knowledge player model most accurately reflected the choices real players made. In this graph, we compare the first move chosen by each of the player models, as well as the players themselves. It is clear that the World Knowledge player model had the most accurate results.

Path, we captured "time" in terms of the number of room transitions between plot points.

This evaluation feature explicitly depends on the physical details of player game world actions, not just on static annotations on plot points. To implement story density, the DM records the room in which each plot point occurs and calculates the 1.5 times the Manhattan distance, in number of rooms, between consecutive plot points (in EMPath, rooms are arranged on a square grid). Using this calculated distance provides an estimate of how many rooms the player passed through between consecutive plot points, without having to account for the concrete connectivity of the world. Given the small size of the EMPath world, story density gives maximum score to consecutive plot points separated by one to four room transitions, giving negative scores to plot points happening in the same room or plot points separated by more than four room transitions.

Player Models

The Drama Manager relies heavily on an internal player model to create its predictive game tree. For each plot point that happens in the game, the DM chooses which action to perform (which can also include the null action). The logic for choosing this action is based on creating a look-ahead game tree of possible future plot points and DM responses. Each leaf of the tree is evaluated using the evaluation features specified by the author. The sub-tree that ends with the highest evaluated leaf is chosen as the DM action. Ties are chosen in a non-deterministic fashion.

Future plot point probabilities are provided by the player model, and therefore the player model is the foundation of the Drama Manager's choices. It stands to reason that the more accurate the player model is the better choices the Drama Manager will be able to make.

Uniform Player Model

This basic player model assumes that each remaining plot point has an equal probability of happening. When hints are active, it adds a weight to the plot point that is being hinted out. This is the most simplistic of the player models, and is currently widely used.

The strength of the uniform player model is that it is the easiest to create and requires the least amount of knowledge of the game world. Because the player model is required to recalculate the probabilities at each node of the game tree, it is called a large number of times. With this player model, the speed is not an issue, and the Drama Manager can choose its next action in a fraction of a second.

With this simplicity comes a large cost. A uniform player model is not generally an accurate model of what a real player would do in the game world. The player model does not take the player's position into account. It seems obvious that a player is more likely to trigger closer plot points than those further away.

By assuming a uniform distribution of probabilities, the Drama Manager may chose an action that will affect a plot point that realistically will not be encountered by the player until much later in the game. We believe these false

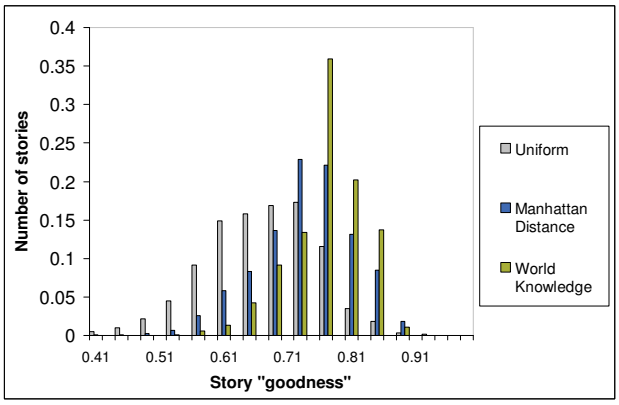

Figure 3: Each player model was used in 5000 test runs. The evaluation scores of each player model are shown above. From this graph it is easy to see that the world knowledge player model has consistently higher scoring stories.

assumptions are the cause for our evaluation numbers in our user tests.

Manhattan Distance Player Model

Due to the limitations of the uniform player model, we first created a player model that utilizes the Manhattan distance to weight the probabilities of each plot point occurring. With this player model, only limited knowledge of the world is known to the drama manager. The coordinates of each plot point is used to calculate the Manhattan distance between plot points. In our game, not every room is connected to each other, so players are not able to necessarily move along a path that is equal to the Manhattan distance. To encode this, we multiply the Manhattan distance by 1.5. This additional weighting gave some allowance for the maze-like nature of the world.

This created a much more realistic probability mapping of the plot points without adding too much complexity. Since players are much more likely to encounter plot points closer to them, this simplified distance measurement takes this into account.

However, there are room configurations where the Manhattan distance is not a good approximation tool. For instance, rooms with locations right next to each other will have a low Manhattan distance. If these rooms are not actually connected by a doorway, but requires traversing a U-shaped path through a series of rooms, this Manhattan distance is a poor approximation.

The other issue with this player model is that the world is small enough that the weighting from the Manhattan distance is not enough to make a discernable difference. (Figure 3) Scaling the weights would help account for this, but the inaccuracies of using the Manhattan distance led us towards a different player model.

World Knowledge User Model

Using the Manhattan distance did not give us the spatial accuracy we wanted, so we created a player model which has an internal representation of the world layout, as well as the locations of the plot points. Using this map, it is possible to calculate much more realistic probabilities of each plot point happening next.

To calculate the probabilities, we traverse one hundred random walks, noting the first plot point encountered. Afterwards, we tally these numbers and normalize to reach the final probability distribution. When performing a random walk traversal, the player model stops at the location containing the first plot point it encounters 95% of the time, continuing the walk the other 5%. This models the fact that that players will sometimes walk past an interaction associated with a plot point (e.g. ignoring a note, not picking up an important object, etc.), though this is unlikely.

This weighting was added due to the results of our player tests. In practice, most users would activate a plot point when it was in the same room as them – by picking up a note or fighting a boss. However, in some cases, players would instead enter a room, then either ignore the plot point device, or back out of the room entirely.

The tradeoff of this player model is the added time and complexity of creating the game tree. For every node in the game tree, one hundred random walks are performed which can add up to a large overhead. By optimizing the random walk, we were able to reduce the decision time of the Drama Manager to approximately 1 second, well within reasonable game time limits.

By comparing the three player models and the actual player traces, we found that the World Knowledge player model is the most accurate of the three. (Figure 2) We compared the first plot point chosen by each player model in five thousand runs as well as those chosen by our players from our user tests. We looked primarily at the first plot point choice since all other plot point choices were affected by DM actions. Because the World Knowledge player model followed the actual player choices the most accurately, we used this as our player in our follow up test runs.

We then performed 5000 evaluation runs where the player was emulated by the World Knowledge player model, and the search tree was created using the three player models. We found that the World Knowledge player model consistently had the highest scoring stories. (Figure 3)

User Studies

As an exploratory study of the affect of the drama manager on the player's experience of EMPath, we performed play test with 31 players: 10 players played the game without the DM, 11 players played the game with the DM turned on and an equal weighting of the four evaluation features, and 10 players played the game with the DM turned on but with the manipulation feature ignored by the DM. In this last condition, the DM could be maximally manipulative in optimizing the experience. The purpose of this study was to gather survey data characterizing how the

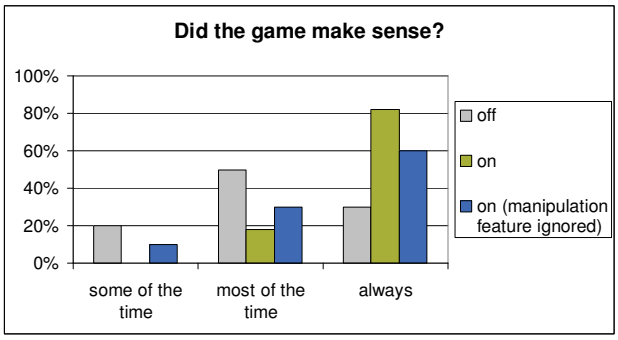

Figure 4: With the Drama Manager on, players found the game made sense more often. None of the participants answered that the game made sense "none of the time."

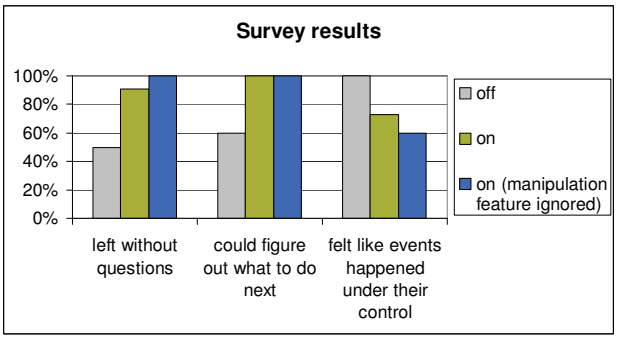

Figure 5: With Drama Management, players had fewer questions about the plot and felt lost less often. A majority of players felt in control of play experience, even with drama management on.

DM affected various aspects of the player's experience. The results of this study can then be used to guide the design of a more formal, quantitative, statistical study.

We hypothesized that players would prefer the flow of the game, find the order of events to make the most sense, and get lost less often with the DM turned on. We hoped this could be accomplished with minimum feelings of manipulation.

Methods

Players initially filled out a survey describing their prior gaming experience. Then they were then given instructions for the game, including the goal of finding the exit and leaving the dungeon. The users were asked to talk aloud as they played. After finishing the game, an interviewer asked them six questions about the experience. The entire session was videotaped.

Results

As this was one of the first times DODM has been integrated into a real game world, the goal of these preliminary tests was to see if there were any positive effects of the DODM. We are extremely encouraged by the results we have found, and are currently working on a more complex protocol with a larger sample size.

Question 1 asked if the players felt that the order in which they learned things made sense. With the DM off, players tended to express confusion about the flow of events in the game. Player 23 complained that "finding things so late in the game, it didn't make sense", while for player 27, who happened to discover the prisoner in the jail before learning anything about this sub-quest, "...the guy in the jail was random."

In contrast, with DM turned on, players were more likely to report the story made sense. Player 26 mentioned that the order made sense because "you got the notes before doing what they wanted." Likewise, player 16 felt that "the events that happened fit together." Overall, 82% of the participants playing with drama management reported the events always made sense, while only 30% of the control group felt the events always made sense. (Figure 4)

Question 2 asked players if they had any questions about items they had received in the game after playing. Our results show that players tend to have fewer questions when the DM is turned on. Player 13, who played without the DM, reported that, "I didn't know why I needed the candle until I reached the stairs," while player 10 who played with the DM turned on commented, "anything I found had information about it." In our tests, 50% of the players in the control group said they still had questions when they had finished playing, while only 9% of the group with DM turned on had questions, and no one was left with questions in the group where DM was turned on and maximally manipulative. (Figure 5)

Questions 3 and 4 dealt with whether the players felt lost. Question 4 specifically asked the player if they sometimes didn't know what to do next. Without the DM, player 19 said that they got "lost and had a lot of backtracking", while player 40, who played with the DM, reported they had "very little wandering" and that "stuff would come up immediately." With the DM turned off, 40% of the players reported feeling lost, while no one from either of the drama managed groups stated that they had felt lost during the game.

Players did have some feelings of manipulation with the DM turned on. Unsurprisingly, more players felt manipulated with the maximally manipulative DM. Player 40 commented that "notes would appear in rooms I'd been to before." However, others liked this dynamic feeling to the dungeon. Player 32 reported that the "candle melting was unexpected and neat." This DM triggered event is an alternative way to receive the candle. No one in the control group felt manipulated, while 27% in the first DM group and 40% in the maximally-manipulative DM group felt manipulated.

Not surprisingly, in the DM condition with the manipulation feature included, the DM chose to make far fewer moves. The DM would typically step in to help the player stay on one quest at a time, or to ensure the player had information before finding objects within the dungeon. The fact that infrequent guidance could dramatically improve

player's perceptions indicates a decent "default" design for the game. One way we could have cheated the play-test is to purposefully design a poor default layout for the game world, requiring significant DM intervention for the world to make sense. This result shows that this is not the case.

One surprising result from our user tests showed that drama managed players had lower evaluation scores than those who did not. This was exactly opposite of our qualitative data. After discussing possible reasons, we began to suspect that the player model we were using was not sufficient for real gameplay.

Future Work

We believe that these tests and results are interesting and positive enough to warrant a wide array of further work.

We hope to soon be able to run more user studies to test the effects of the World Knowledge player model on real gameplay cases. We hope that this will correct the incongruity of the evaluation numbers seen in our previous user test, as well as lead to a better player experience.

An interesting and obvious extension of this work would be to implement a DM with a much larger game and test the scalability of the results. Intuitively it seems that a larger game would magnify the results of the DM. However, there would be some concern that the DM may not scale well to the added complexity of a large story world, since the search space grows exponentially with the number of possible plot points and drama management moves.

The Drama Management system itself is also currently not as well suited to typical genres as it could be. For instance, the DM currently only takes action after a plot point has happened. In cases where the user is wandering for long periods of time, presumably lost as to what to do to move the experience forward, DODM will take no action. By incorporating a pro-active component into the model, the DM could take action in the form of a hint or a causer when a long period with no plot point activation is detected. .

Finally, it is not entirely intuitive for a designer to choose and create DM actions or the complex interaction of the DM and the evaluation features. It would be helpful to create a graphical authoring tool that aids the designer in this process. Such a tool could, for example, allow the author to create a graphical representation of the plot point DAG, plus additional annotations about the story world, and use this information to guide the author in defining appropriate DM moves and evaluation features.

References

Bates, J. 1992. Virtual reality, art, and entertainment. *Presence: The Journal of Teleoperators and Virtual Environments*, vol. 1, no. 1, pp. 133–138.

Lamstein, A., and Mateas, M. 2004. A search-based drama manager. In *Proceedings of the AAAI-04 Workshop on Challenges in Game AI.*

Magerko, B. 2005. Story Representation and Interactive Drama. *Artificial Intelligence and Interactive Digital Entertainment Conference*, AAAI Press.

Mateas, M. and Stern, A. 2003. Façade: An Experiment in Building a Fully-Realized Interactive Drama. *Game Developers Conference, Game Design track.*

Mott, B.W. and Lester, J.C. 2006. U-director: a decision-theoretic narrative planning architecture for storytelling environments. *Proceedings of the fifth international joint conference on Autonomous agents and multiagent systems*, ACM Press: 977-984.

Nelson, M.J. and Mateas, M. 2005. Search-based drama management in the interactive fiction Anchorhead. *Proceedings of the First Annual Conference on Artificial Intelligence and Interactive Digital Entertainment*, AAAI Press.

Nelson, M.J., Roberts, D.L., Isbell Jr, C.L., and Mateas, M. 2006. Reinforcement learning for declarative optimization-based drama management. *Proceedings of the fifth international joint conference on Autonomous agents and multiagent system*: 775-782

Nelson, M.J., Mateas, M., Roberts, D.L., and Isbell Jr, C.L. 2006. Declarative Optimization-Based Drama Management in Interactive Fiction. *IEEE Computer Society*: 32-41.

Ontañón S., and Jain A., and Mehta M., and Ram A. 2008. Developing a Drama Management Architecture for Interactive Fiction Games. *First Joint International Conference on Interactive Digital Storytelling.*

Roberts, D.L., Nelson, M.J., Isbell, C.L., Mateas, M., and Littman, M.L. 2006. Targeting Specific Distributions of Trajectories in MDPs *Artificial Intelligence and Interactive Digital Entertainment International Conference.*

Si, M., Marsella, S.C., and Pynadath, D.V. 2005. Thespian: Using Multi-Agent Fitting to Craft Interactive Drama. In *Proceedings of the fourth International Joint Conference on Autonomous Agents and Multi Agent System.*

Thue, D., Bulitko, V., Spetch, M., and Wasylishen, E. 2007. Interactive storytelling: A player modeling approach. *Proceedings of the third Artificial Intelligence and Interactive Digital Entertainment conference.*

Weyhrauch, P.W. 1997. "Guiding interactive drama," Ph.D. dissertation.

Young, R.M. and Riedl, M. 2003. Towards an architecture for intelligent control of narrative in interactive virtual worlds. *Proceedings of the 8th international conference on Intelligent user interfaces*: 310-312.