University of New Orleans

ScholarWorks@UNO

University of New Orleans Theses and Dissertations

Dissertations and Theses

Fall 12-20-2019

Multi-Agent Narrative Experience Management as Story Graph Pruning

Edward T. Garcia
University of New Orleans, etgarci1@uno.edu

Multi-Agent Narrative Experience Management as Story Graph Pruning

A Thesis

Submitted to the Graduate Faculty of the
University of New Orleans
in partial fulfilment of the
requirements for the degree of

Master of Science
in
Computer Science
Artificial Intelligence

by

Edward Garcia

B.S. University of New Orleans, 2005
M.B.A. University of New Orleans, 2009

December, 2019

ACKNOWLEDGEMENTS

The work produced here was a joint effort of students and faculty at the University of New Orleans Narrative Intelligence Lab. I give my sincere appreciation to my master's thesis committee, Dr. Stephen G. Ware, Dr. Ben Samuel, and Dr. Md Tamjidul Hoque, and to my fellow graduate students, Rachelyn Farrell, Alireza Shirvani, and Dharmesh Desai. Thank you.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

What is this paper about?
In this thesis I describe a method where an experience manager chooses actions for non-player characters (NPCs) in intelligent interactive narratives through story graph representation and pruning.

Description of our experience manager
The space of all stories can be represented as a story graph where nodes are states and edges are actions. By shaping the domain as a story graph, experience manager decisions can be made by pruning edges. Starting with a full graph, I apply a set of pruning strategies that will allow the narrative to be finishable, NPCs to act believably, and the player to be responsible for how the story unfolds. By never pruning player actions, the experience manager can accommodate any player choice.

How did we evaluate the experience manager?
This experience management technique was first implemented on a training simulation, where participants' performance improved over repeated sessions. This technique was also employed on an adventure game where players generally found the NPCs' behaviors to be more believable than the control.

Keywords
KEYWORDS: Planning, Classical Planning, Narrative Planning, Experience Manager, Artificial Intelligence

INTRODUCTION

Description of Interative Environments/Simulations

An interactive virtual environment is an effective medium for education, training, therapy, and entertainment. Simulations provide a safe environment where trainers can teach trainees about situations that may otherwise be dangerous. Simulations can be more cost-effective than using actual equipment, both in terms of training costs and equipment repair. Military personnel, firefighters, doctors, nurses, and police employ simulations for training (Hays & Singer, 1989).

Why use an Interactive Narrative?

When relying on human actors, these kinds of role-playing scenarios can be time-consuming and cost-prohibitive to create, run, and evaluate. Even considering small training domains there could be hundreds or thousands of possible stories the participant can experience. A human author in charge of creating this experience would manually have to keep track of all these stories which can be difficult, time-consuming, and error prone. We look to the field of interactive narrative planning to help ease the burden and cost of creating and reasoning about the simulation.

A simulation can be represented as a planning domain where an author describes the world using characters, locations, actions, intentions, goals, the initial state of the world, and more. Using the domain, a narrative planner can find all possible stories using the author's description of the world. This alleviates the human author from having to keep track of all possible stories, and instead only think about the factors of the world that is important to portray.

However, when considering interactive stories, a player entity introduces uncertainty when reasoning about the domain. There are now choices that can be made by the player at any time which will not be known until the simulation is running. It is up to the simulation to react. An experience manager can be used to keep track of the state of the world, monitor player actions, and control the simulation at every given state. Using an experience manager, the player can then take any possible action, and the simulation will be able to react.

Police Use of Force Simulation Introduction

I have created a prototype training simulation that allows police officers to explore the consequences of use of force decisions. It takes place in a virtual world where a police officer responds to a call about a potentially dangerous suspect. The participant takes the role of the officer and is free to take any action available in the virtual world. The goal is to teach key concepts defined by the Police Executive Research Forum (2012) via a learn-by-doing approach.

Even in this limited scope training domain, which contains 11 types of actions, 5 endings, and 5 measures of player knowledge, there are 125,688 unique states and 752,741 possible transitions between those states. Hence, we use a planner-based experience manager in place of human actors to make decisions in our simulation.

Camelot Study Introduction

Camelot is an adventure game that allows a player to explore and create a narrative in a medieval setting. It takes place in a virtual world where the participant takes control of a young man who is tasked to bring back medicine to his sick grandmother. There are 4 locations, 3 NPCs, and 6 items with which to interact.

What did I contribute and how?

In order for non-player characters (NPCs) to act, an experience manager must inform the simulation of which actions to take in a given state. Hence, experience management can be viewed as graph traversal. Nodes in a story graph represent states of the virtual environment and edges represent actions that change the state. We use various pruning techniques to make intelligent choices given a state.

My contribution is the application of an automatic experience manager using pruning strategies outlined in this thesis. I have created an intelligent training simulation that uses this automated experience manager to create interactive stories that teach use of force policies. Additionally, I was a main contributor to the adventure game that demonstrates using the defined experience manager techniques creates more believable characters than that of the control.

RELATED WORK

Overview of section

In this section, we will cover the basic concepts of planning, narrative planning, experience management, and virtual environments.

Planning

What is planning?

Planning is the science of reasoning about a sequence of actions which achieve some goal. Three components represent a planning problem (Russell, S.J. and Norvig, P., 2016): initial state, action, and goal. A state is a representation of the entire environment, where logical propositions define its configuration. An action is a step to perform that transitions one state into another. Actions are specified in terms of preconditions, that must be satisfied immediately before it can be executed, and effects, which are applied to the state immediately after an action occurs. The goal is a logical proposition that must be true at the end of a plan. A valid plan is any sequence of actions which results in a state where the goal is true.

Classical Planning

Classical Planning Assumptions

Classical planning assumes that the environment is fully observable, static, and deterministic (Russell, S.J. and Norvig, P., 2016).

- Fully Observable: When making decisions, the planner has complete knowledge of the world.
- Static: Environment does not change while agent deliberates.
- Deterministic: An action from one state always leads to a next predictable state.

In the real world, a planner would neither have complete knowledge of the world nor complete control over it. However, in a virtual world, the environment is fully observable, static, and deterministic except for actions that player may perform in the world.

Domain Dependent vs Domain Independent Planning

In a domain dependent planner, the facts presented will involve the domain about which the system is expected to reason (Ginsberg, Matthew; Geddis, Donald F., 1991). Wilkins (1983) explains, domain-specific planners are designed to work efficiently in a single problem domain. When finding a solution, a domain-specific planner depends upon the structure of the domain to run efficiently. Because of that, the underlying algorithms may not be applicable in another domain.

However, a domain independent planner is generated by a planning technique which is applicable in many domains and provides general planning capabilities (Wilkins, David E., 1983). There are widely renowned examples of domain independent planning such as SRI's STRIPS (Fikes, Richard E.; Hart, Peter E.; Nilsson, Nils J., 1972), Penberthy and Weld's UCPOP (1992), Haffmann and Nebel's Fast Forward (Hoffmann, J; Nebel, Bernhard, 2001), etc.

Narrative Planning

What is a narrative and interactive narrative?

For the sake of this thesis, a narrative (Riedl, Mark; R, Michael Young, 2014) is a predetermined, temporally ordered set of actions or events. An interactive narrative (Riedl, Mark; R, Michael Young, 2014) is a form of digital media in which users create or influence a dramatic storyline through actions, either by assuming the role of a character in a fictional world or by issuing commands to an autonomous, virtual non-player character.

How does Narrative Planning differ from Classical Planning?

Narrative planning is a variant of classical planning which searches for a sequence of actions to achieve the author's goal, such that all actions are clearly motivated and goal-oriented for agents who take them. In 2012, Haslum (2012) and Riedl and Young (2014) explained that Narrative Planning is a type of planning with additional conditions. It places additional constraints on a planner's solution: some system level goal called the author's goal must be achieved, but agents must only act in service of their individual goals, possibly cooperating and competing with one another in the process (Ware S. G., 2014). The main difference between narrative planning and classical planning is the notion of intentionality. Agents behave intentionally in such a way that the agent has some motivation behind the actions.

Strong Story vs Strong Autonomy

When considering implementation of narrative generating systems, we can consider a scale between *strong story* and *strong autonomy* (Riedl & Bulitko, 2013). On one side of the spectrum, *strong story* is a system that guarantees a unified plot. Whereas *strong autonomy* guarantees accurate simulation of each character.

One possible downside of a completely *strong story* narrative approach is that the plot produced does not consider characters' goals and may contain actions that may not make sense for a character to take. On the other hand, *strong autonomy* considers characters' goals but may produce stories that do not exactly meet the needs of the author.

Intentional planning is a compromise between the two where it ensures the author's desired outcome while generating believable character behavior (Ware S. G., 2014).

Narrative Planning Detailed Explanation

The following are all the parts necessary for an intentional planner to create a plan that starts at an initial state and ends at a goal state considering both author's goal and agents' goals. These definitions are derived from Ware et al (2011) and (2014).

Constant: A defined component of the world that does not change.
Typed Constant: Constants sharing a characteristic or set of characteristics. Examples include characters, places, and items that exist in the world.
Character: A special type of constant which represents an agent with intentions.
Fluent: A feature of the state which can be assigned one of several values. For example, the officer's location is a fluent and its possible values are all the locations available in the domain (Street, Sidewalk, etc).
Literal: A statement, which either can be true or false, which asserts that fluent has a specific

value. For example, "the officer is at the sidewalk" is a literal.
State: Single function-free ground predicate literal or conjunction of literals describing the story world.
Initial State: A set of literals that completely describe the world before the start of the plan
Author Goal: Literal or conjunction of literals that must be true at the end of the plan
Character Goal: Literal or conjunction of literals that a character wants to be true. Character goals do not need to be achieved but can be used to explain character actions.
Step/Action: An event that occurs that transitions a state to another state. A step preconditions, effects, and consenting characters.
Preconditions: A set of literals that must be true immediately before the step can be taken.
Effects: A set of literals that become true immediately after the step has been taken.
Consenting Characters: A possible empty list of characters, who are responsible for taking the actions and must have a reason to take the action before the action can be executed.
Planning Domain: A list of constants, characters, initial state, author goal, character goals, and actions that make up the story world.
Plan: A sequence of actions where the preconditions of actions are true immediately before the action taken.
Causal Link: Explanation of how the effect of an earlier step satisfies the precondition of a later step. A causal link is denoted as $s \xrightarrow{p} u$ where u is the step with the precondition literal as proposition p and s is a step with an effect p. Hence, we say p is true, because s made it so. A causal link only exists if no step between s and u negates p.
Intentional Path: A sequence of n alternating actions and propositions $\langle a_1, p_1, a_2, p_2, \dots, a_n, p_n \rangle$ taken by a character in service of a goal literal p_n. The character must have the goal p_n before the start of this sequence and the last action of the sequence a_n must have effect p_n. For i from 1 to $n-1$, there must exist a causal link $s_i \xrightarrow{p_i} s_{i+1}$ (that is, all steps in the path must be causally linked). No proposition can appear twice, and an intentional path may not contain a proposition and its negation.
Explained Step: A step is explained if and only if, for all of its consenting characters there exists an intentional path for that character that includes this step and such that every other step in that path is also explained (Note: The actions of this intentional path do not need to appear in the plan. In other words, a character can intend to take actions that never actually happen). Put another way, an explained step is one where every consenting character has a reason to perform this step and the other steps used to explain this step are also reasonable.
Valid Intentional Plan: A plan that ends in a state where the author's goal is achieved and where every step is explained.

Experience Management

Brief description of interactive drama/story

An Interactive Drama (Roberts, David L.; Isbell, Charles L., 2008) is one where a player is an active participant in how the story unfolds. Here the user makes the decisions for one agent. A player is able to explore different parts of the environment and able to engage other players and non-player characters by taking specific actions. In turn, the non-player characters will react to the behaviour of the player instructed by the experience manager. This helps in making an engaging and player-driven experience. Thus, the author of the narrative is responsible for designing specific situations that can be expected to happen during play.

Role of the Experience Manager

An Experience Manager (Roberts, David L.; Isbell, Charles L., 2008) is a coordinator. It tracks the narrative progress by directing roles and responses of objects for achieving specific narrative or training goals. In many cases, the user makes the decision for one agent and the experience manager makes decisions for all other agents. These experiences are complex and can deliver agency (Wardrip-Fruin N. , Mateas, Dow, & Sali, 2009) to the player to influence the way in which the experience unfolds. In this thesis, we explain an experience manager technique that informs NPCs how to behave at any given state.

Mediation

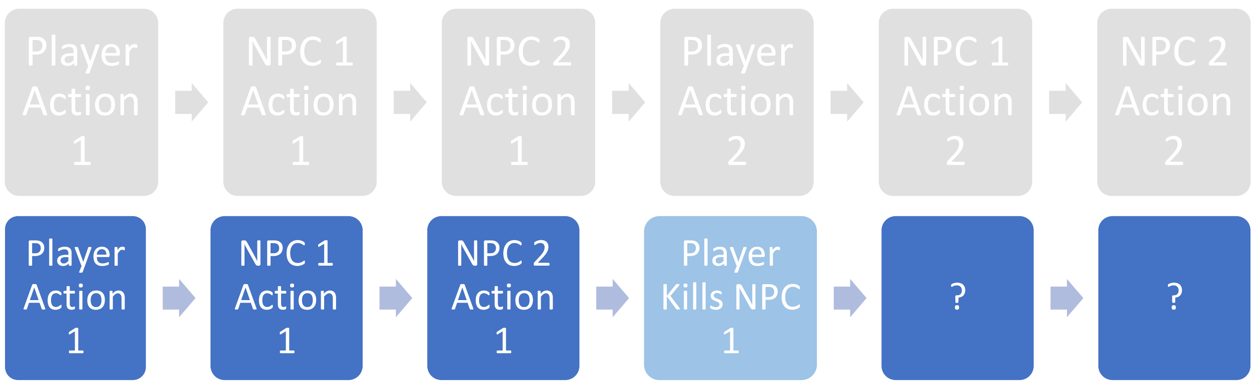

Figure 1: Expected Narrative compared to Unexpected Narrative

Actions taken by a player in the simulation are unpredictable. In *Figure 1*, we see two sequences of actions between Player and two NPCs. The first sequence is a story the author intends to tell. The second sequence contains the action "Player Kills NPC 1". This action directly affects the intended story, and the story above can no longer be executed. We call this action an exceptional action (Harris & Young, 2005).

Reactive Accommodation

The author at this point can ask the following questions. Can the story continue without NPC 1? Can a different set of actions still meet the intended story goals? Starting after the exceptional action, reactive accommodation finds another set of actions that can still reach the author's goal. For example, perhaps NPC 2 can perform that action that NPC 1 was supposed to perform.

Reactive Intervention

Reactive intervention is another strategy that stops the player from taking exceptional actions. One solution is to modify the effects of an action, such as a player attempting to kill NPC 1, but the NPC does not die and is "just hurt" in the process. Another solution is to stop the action altogether. For example, the player tries to kill NPC 1, but before the player makes the attempt, NPC 2 grabs his arm and stops him from performing the action.

Limitations

Handling this issue at the point of the exceptional action has some limitations. It can be frustrating to the player if he takes a long series of action to reach the point of the exceptional action only to have it stopped or modified. Additionally, modifying or stopping a player action may be perceived as a rather extreme action by the system to the player. However, intervention may be necessary as the alternative would be a story that cannot reach a goal state.

Proactive Accommodation

This strategy is similar to reaction accommodation, but we consider alternatives *before* the exceptional action. For example, after "Player Action 1" in *Figure 1*, can NPC 1 and NPC 2 take different actions to prevent the player from killing NPC1 and still meet the author's goals? Is there another sequence of actions that reach the goal state?

Proactive Intervention

Proactive intervention is based on the same concept as reactive intervention, but the actions before the exceptional action can be modified or stopped to allow the story to reach a goal state.

THE SIMULATION

Overview of Simulation Creation

In this section, we will discuss two simulations that were created to test the effectiveness of our experience manager.

Police Use of Force Training Domain

In-Depth Description of the domain

In the Police Use of Force Training Domain, the participant plays the role of a police officer (Officer), who is responding to a call from a young man's mother. The young man (Suspect) has been recently kicked out of the house. He is on the porch, banging on the door, and possibly has a knife.

The domain has been written in STRIPS-style format where all objects, actions, and goals are specified (Fikes, Richard E.; Hart, Peter E.; Nilsson, Nils J., 1972). The simulation uses a client/server configuration, where the client takes input from the participant to control the officer and sends player actions to the server. The client was created with the Unity Game Engine. The participant controls the client using either Screen and Keyboard controls or Virtual Reality controls. This was not the focus of this thesis, but other experiments used the comparison of both control types to analyze presence in terms of a story generated by a narrative planner (Garcia, Ware, & Baker).

The server stores the current state of the simulation, and depending on the state, the participant is allowed to take certain actions. The server also directs the client on which actions the NPCs will take.

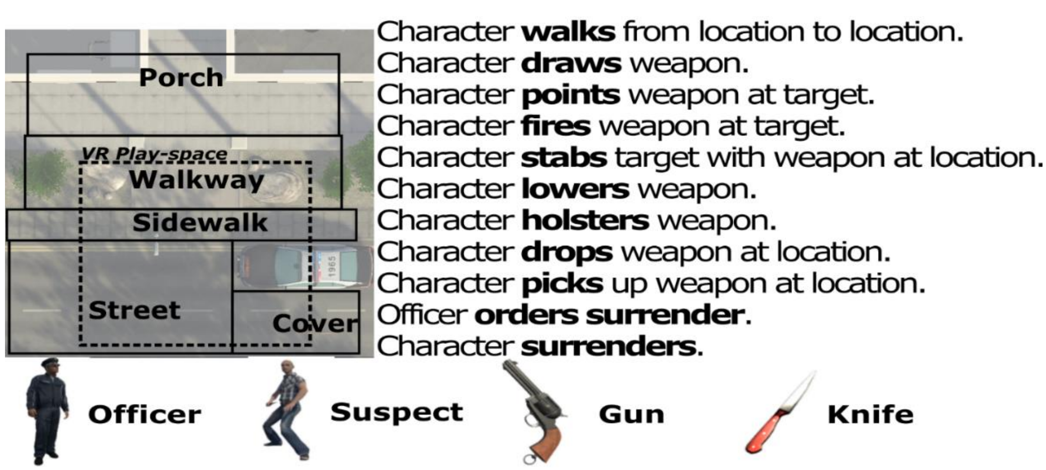

Figure 2: Police Use of Force Training Simulation Domain Description

The virtual reality controls use the HTC Vive virtual reality system in room-scale mode, where the room represents the area shown in *Figure 2*. The client allows the player to freely move around a continuous space and perform actions in this space. However, the play-space and actions taken by the client are discretized for processing by the planner on the server. Our simulation contains five locations, two characters, two weapons, and eleven actions as shown in

Figure 2. Each session begins with the Officer at the Cover location and the Suspect at the Porch location. A session is designed to last about 1 minute. Each action in the simulation takes roughly the same amount of time as it would take in the real world. Every executed character action (by the player or an NPC) is logged for analysis. At the end of each session a score is displayed on the screen depending on how the simulation ended. Scores rank the possible endings from worst to best based on the safety of the officer and the suspect.

Simulation Background

The Police Executive Research Forum (PERF) (2012) has identified best practices regarding use of force that are designed to ensure that the officer, suspect, and bystanders remain safe in a potentially dangerous situation. PERF also speculates that some officers may leave the academy with a bias toward using force because many training simulations assume that force is always necessary. There are many tools for teaching officers how to shoot, but too few for teaching them how *not* to shoot. They call for innovative methods to address this problem.

When an officer is faces with having to deescalate a situation, getting more time leads to greater probability of success. Our simulation is designed to teach one use of force policy in particular: *distance + cover = time*. When an officer keeps distance and cover between himself and a suspect, he can buy time to achieve a peaceful resolution. Policies like this one demonstrate the advantages of interactive narrative training simulations over traditional shooting range simulations because, depending on the trainee's actions, force may not be needed at all.

In this implementation, we are trying to teach the participant about keeping distance, keeping cover, and not escalating the situation. A higher score shows that the participant performed actions that demonstrated desired behavior within the simulation. A lower score is given if the participant takes undesired actions within the simulation. For example, the simulation starts off with the suspect being calm. If the player points the gun at the suspect, the suspect is angered and starts approaching the officer and threatens the officer. If the suspect threatens the officer, a lower score is given.

Description of Scores

- Score 0: The suspect stabbed and killed the officer.
- Score 1: The officer shot the suspect, but the suspect never threatened the officer with the knife.
- Score 2: The officer shot the suspect after being threatened with the knife.
- Score 3: The suspect surrendered, but only after threatening the officer with the knife.
- Score 4: The suspect surrendered and never threatened the officer with the knife.

Description of Knowledge Attributes

We determined five specific features about which trainees might demonstrate knowledge or ignorance. We call these *player knowledge attributes* and define them based on the states in which the trainee finds himself and the actions he takes or does not take. Knowledge or ignorance of these attributes can thus be measured automatically by analyzing a session log. We must note that these represent our own non-expert interpretations of use of force policies. Before using this simulation to train actual police officers we must obtain feedback from experts.

- Keep Distance: The officer should keep distance between himself and the suspect. If the participant and suspect get within arm's reach of one another (that is, occupy the same location as shown in *Figure 2*) this concept is not known. Otherwise, this concept is known.
- Use Cover: The officer should keep cover between himself and the suspect, even if he must retreat. If the participant walks back to the Cover location (behind the car), this concept is known. Otherwise, this concept is not known.
- Justified Force: PERF (Police Executive Research Forum, 2012) mentions that holding a knife is not the same as brandishing a knife. If the suspect raises the knife in a threatening way and the officer uses deadly force, this force was justified. If the officer finds himself in a situation where force is justified and uses it, this concept is known. If the officer finds himself in a situation where force is justified and does not use it, this concept is not known.
- Unjustified Force: If the officer used deadly force when the suspect was not close and/or has not raised the knife, force was not justified and should not have been used. If the officer uses force in this way, this concept is not known. Otherwise, it is known.
- Agitation: The suspect is nervous, and the way the officer deals with him can either calm him down or further agitate him. If the officer points his gun at the suspect while the suspect is not angry, the suspect becomes angry and aggressive. If the officer angers the suspect in this way, this concept is not known. Otherwise, it is known.

Additionally, these knowledge attributes are an approximation of the knowledge of the player. They are assigned based on the actions of the player. If a player "gets lucky", and performs all the correct actions, the simulation will assume the player knows these concepts, where, he may not actually know about these concepts.

Camelot Study Domain

Figure 3: Camelot domain locations

Camelot takes place in a medieval setting where the player controls a young man who is tasked to retrieve medicine for his sick grandmother. The simulation at the time of the study consisted of four locations, three NPCs, and six items. The possible actions are:

- Character **walks** from location to location
- Character **buys** from seller an item with coin
- Character **takes** an item from container

- Character **puts** an item into container
- Character **attacks** victim with weapon
- Character **steals** from victim an item with weapon
- Character **reports** criminal to guard

Camelot was created using a client/server architecture, where the client was created in Unity3d game engine and the server is the experience manager much like the police use of force training simulation. Hence the experience management keeps track of state and communicates all decisions to the client.

The simulation starts in Grandma's cottage where grandma asks the player to go to the market and get some medicine. Once leaving the cottage, the simulation begins and all player choices from this point forward may affect how the story unfolds. The story then ends when the author's goal of the player arrives back to the cottage with the medicine. However, to keep the story interesting, we have decided that the outside world can be a dangerous place, hence the player can die in the attempt to bring home the medicine.

In-Depth Description of the domain

Figure 4: Camelot domain characters

The player begins at home, where they learn their grandmother is sick. She gives the player a gold coin that can be used to buy medicine. The game features three NPCs. A merchant is in the market selling medicine and a sword. The town guard is in the market watching for criminals. A bandit waits in his camp. The bandit has a coin of his own that he keeps in a chest, but he is hoping to acquire more items of value such as money and medicine. There are three main locations: the player's house, the market, and the camp. A fourth location, the crossroads, connects all three. The game ends when the player returns home carrying the medicine or when the player dies.

Seven kinds of actions are available. Characters can walk from one place to another. Characters can take items out of the chest in the bandit's camp. Characters can buy items from the merchant for 1 gold coin each. If a character is armed, they can steal an item from an unarmed character. One character can attack and kill another, unless the attacker is unarmed, and the victim is armed. Characters can loot items from slain characters. Finally, a character who knows the bandit's location can report him to the town guard. Despite its small size and simplicity, this domain yields a surprising number of interesting ways the player can accomplish their goal or die in the attempt.

States and Actions

State Space Representation

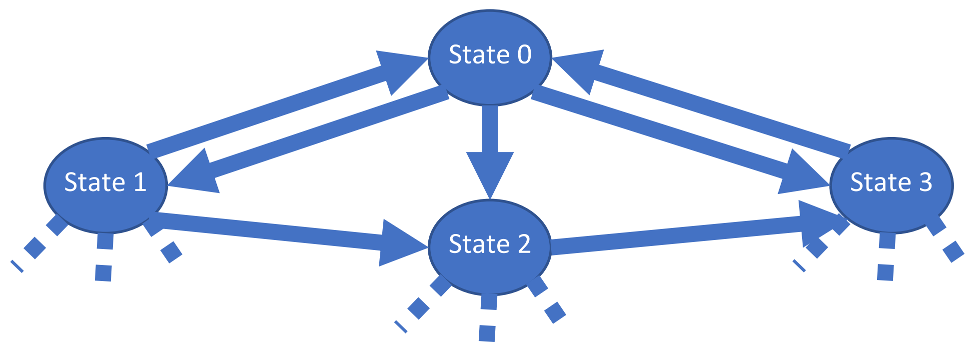

Figure 5: Initial creation of a story graph

The state space of the simulation can be represented by a story graph where nodes are unique states and a directed edge $s_1 \xrightarrow{a} s_2$ may exist from state s_1 to state s_2 for action a if a's preconditions are met in state s_1 and taking a in s_1 would results in s_2.

A state is a set of propositions which completely describes three things: the configuration of the physical world, including all characters and items, the intentions of all agents, and the current state of the player model if one is being tracked (i.e. knowledge attributes).

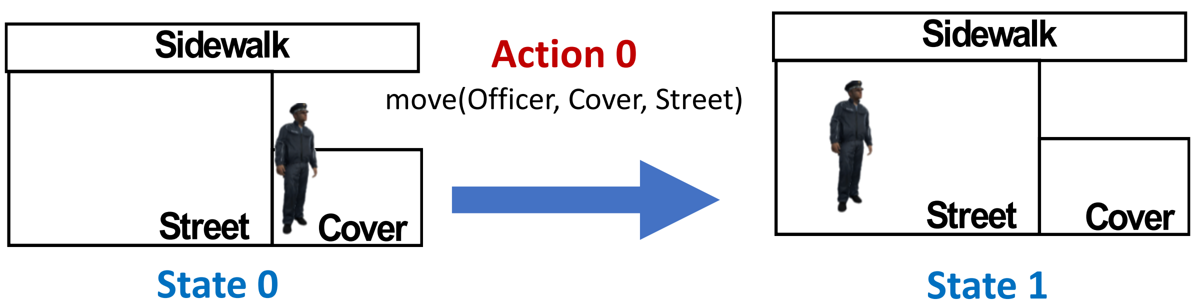

Figure 6: State Space Representation

Both domains are written in a STRIPS style format using multi-value variable representation of state. *Figure 6* shows an example of State 0 where `location(Officer) = Cover`. Action 0's preconditions are true in State 0 and the effects can be applied creating State 1 where `location(Officer) = Street`.

An example of the contents of one state node in the Police Use of Force Training are as follows:

- `alive(Officer) = True`
- `alive(Suspect) = True`

- `location(Officer) = Cover`
- `intends(Officer,`
- `location(Gun) = Officer`
- `location(Suspect) = Porch`
- `location(Knife) = Suspect`
- `holding(Suspect) = Knife`
- `target(Suspect) = Null`
- `surrendered(Suspect) = False`
- `intends(Officer, intends(Suspect, surrendered(Suspect) = True)`
- `path(Cover, Street) = True*`
- `path(Street, Sidewalk) = True*`
- `path(Sidewalk, Walkway) = True*`
- `path(Walkway, Porch) = True*`
- `audible(Cover, Street) = True*`
- `audible(Cover, Sidewalk) = True*`
- `audible(Cover, Walkway) = True*`
- `audible(Street, Sidewalk) = True*`
- `audible(Street, Walkway) = True*`
- `audible(Street, Porch) = True*`
- `audible(Sidewalk, Walkway) = True*`
- `audible(Sidewalk, Porch) = True*`
- `audible(Walkway, Porch) = True*`
- `knowsUseCover = Null`
- `knowKeepDistance = Null`
- `knowsJustifiedForce = Null`
- `knowsUnjustifiedForce = Null`
- `knowsAgitation = Null`

As shown above, here are most of the multi-valued variable assignments that make up one state node of the Police Use of Force domain. We have made a closed world assumption where any unspecified value is assigned False (for example, `audible(Cover, Porch) = False`). The goals of the characters are also represented via the "intends" property. Finally, the player model is represented via three-valued variables `knowsUseCover`, `knowsKeepDistance`, `knowJustifiedForce`, `knowsUnjustifiedForce`, and `knowsAgitation` with possible value `True`, `False`, or `Null`.

Action Timing Management

Each action in our domain takes about the same amount of time it would take in the real world. One of the challenges we faced was how to handle actions in continuous time when our domain was represented in a discretized state space. To handle this issue, every action had a starting point, ending point, and ability to be cancelled at any time between both points. Every time an action reached the end, it was communicated to the experience manager, and the experience manager would respond whether other running actions should continue or cancel.

Story Graph Pruning

A story graph (Li, Boyang; Lee-Urban, Stephen; Johnston, George; Riedl, Mark O, 2013) defines the space of legal story progression and ultimately determines possible events at any given point in time. Our simulations are represented via story graphs where nodes are states and edges are actions. The experience manager is responsible for making decisions about NPC's action at every given state. However, there are many possible actions from each state that can be taken by the player, a NPC, or both. Hence, our experience manager prunes edges at every state until a decision is made for each NPC.

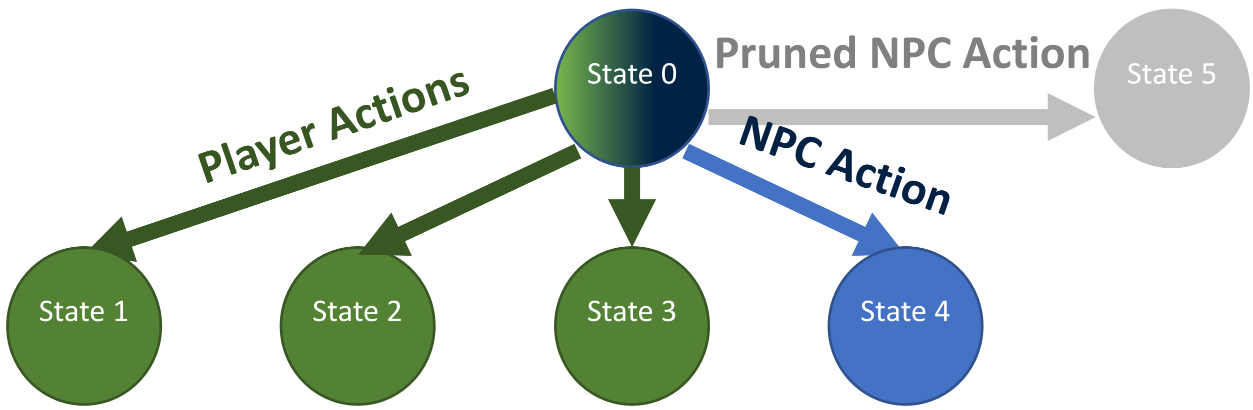

Figure 7: Pruning story graph so that only 0 or 1 actions exists per NPC

Each action can be categorized as either a player action, an NPC action, or some are both where both player and NPC must consent to the action. In the simulation, the player should always be free to perform any action when its preconditions are met. By using an automated experience manager, our system is able to react to a player taking any of these actions without needing to hand author every possibility. In some cases, the experience manager may direct an NPC to take an action. In other cases, the experience manager may decide it best for the NPC not to act at all.

Pruning Strategy	Police Use of Force Training Simulation		Camelot	
	Nodes	Edges	Nodes	Edges
Full Story Graph	126,688	752,741		
Intentionality	125,428	554,319	388,318,086	1,028,110,791
Shorter Plan			93,608,267	248,440,557
Lazy NPC			58,191,971	148,928,950
Unique Endings	125,428	550,447	52,262,059	138,072,434
Player Knowledge	125,428	544,491	52,262,059	138,072,434
Goal Priority			30,149,245	76,006,520
Cyclic			23,159,543	56,783,502
Arbitrary	125,428	534,991	20,365,197	49,669,363
Dead End			20,365,187	49,669,351

Table 1: Story Graph Reduction Results

Full Story Graph

The experience manager's decisions have been precompiled using the methods below, but the same criteria could be applied (or approximated) in real-time systems. We begin with the full story graph for our domain which is the entire state space representing every state and every possible action reachable from the initial state. We then prune this graph intelligently until every NPC has at most one action to perform in each state, thus making the experience manager's decisions unambiguous. We never prune player actions (i.e. we never prevent a player from taking an action which should be possible in the current state). Pruning the story graph at design time allows us to fully consider the long-term consequences of every decision on the space of possible stories that can be told. The full story graph sizes are shown on *Table 1*.

Note, the Police Use of Force Training Simulation did not use all the pruning strategies shown in *Table 1*. The domain was smaller than the Camelot domain, and the pruning strategies which were used was enough to create our experience manager for this study. Also note that Camelot is missing Full Story Graph numbers. In the Camelot study, our base was a story graph that was already pruned.

Intentionality Prune

Studies show that virtual characters appear more believable when they act intentionally. That is, they appear to be working toward their goals (Riedl & Young, 2010). We use Shirvani, Farrell, and Ware's (2017) model of intention: any action that an agent does not believe will contribute to achieving the agent's goal should be pruned. After intentionality pruning, the story graph sizes are shown on *Table 1*.

Without Intentionality Prune	With Intentionality Prune
Simulation starts	Simulation starts
Suspect surrenders	Officer approaches Suspect
	Officer orders Suspect to surrender
	Suspect raises knife
	Officer takes cover
	Suspect surrenders

Table 2: Example of Intentionality Pruning

Table 2 shows an example of possible stories to reach the goal `surrendered(Suspect) = True`. If we do not consider the goals of the suspect, a possible action is for the Suspect to surrender right way. However, if we do consider the goals of the Suspect, we observe a story where the suspect does not want to surrender right away, but only after various actions does the suspect decide he wants to surrender.

Shorter Plan Prune

Given two plans for the same agent to achieve the same goal, we prefer the shorter plan. After shorter plan pruning, the story graph sizes are shown on *Table 1*.

Longer Plan	Shorter Plan
Guard observes Player killing Merchant	Guard observes Player killing Merchant
(Guard wants to attack player)	(Guard wants to attack player)
Guard picks up sword from Merchant	Guard attacks Player with Guard's Sword
Guard attacks Player with Merchant's Sword	

Table 3: Example of Shorter Plan Prune

At the state where the guard observes player killing merchant, we prune the Guard picking up the sword from the merchant as that leads to a longer plan to achieve his goal of killing the player.

Lazy NPC Pruning

Given a choice between a plan with mostly NPC agent performing actions to achieve the agent's goal and a plan with mostly a player performing actions to achieve the agent's goal, we prefer the latter. One of our design principles is to give the player the most opportunity to explore the space and feel agency on how the story unfolds. In order to do this, we prefer plans that have more player actions. This also has the added benefit of not having all the NPCs converge to the player hoping the player will perform some action at the beginning of the story.

More NPC Actions	More Player Actions
Simulation starts	Simulation Starts
(Merchant wants a coin)	(Merchant wants a coin)
Merchant walks to Crossroads	Players walks to Crossroads
Merchant walks to Cottage	Players walks to Market
Player buys medicine from Merchant	Player buys medicine from Merchant

Table 4: Example of Lazy NPC Pruning

For example, consider Camelot player's goal to buy the medicine. The player could travel to the market, buy the medicine from the merchant, and then travel back home. Alternatively, the merchant could travel to the player's home and sell them the medicine without requiring the player to ever leave the house. Though both plans make intentional sense and achieve an author goal, we prefer the former, because it gives the player more opportunity to explore and find their own way to achieve their goals. This is the Lazy NPC principle. After this pruning, the graph reduction can be seen in *Table 1*.

Unique Ending Prune

The author's goal in this domain is a disjunction of various possible ending states. The experience manager is neither cooperating with the player to achieve a good ending nor opposing the player to achieve a bad one. Rather, the actions taken by the player (not the NPCs) should be responsible for the ending earned. Hence NPCs should prefer actions which keep the higher number of possible endings available. This is a tie breaking prune, which means that if there exists only one edge for an NPC, it will not be pruned using this technique. After unique ending pruning, the story graph reduction is shown in *Table 1*.

Less Unique Endings	More Unique Endings
Player and Bandit walks to Crossroads	Player and Bandit walks to Crossroads
(Bandit wants a coin)	(Bandit wants a coin)
Bandit kills Player	Bandit steals coin from Player
Bandit takes coin from Player	

Table 5: Example of Unique Ending Prune

The order in which these pruning strategies are employed is important. We perform intention pruning first because it is important for characters to be believable. If unique ending pruning were to happen first, it is possible that the characters would act unbelievably or not at all to ensure more endings stay available. For example, say the Officer angers the Suspect. The Suspect approaches the Officer and then raises the knife. If unique ending pruning has occurred before intentional pruning, the Suspect would simply do nothing from this point on, because stabbing the officer would remove a unique ending. Instead, we want the Suspect to follow through with his plan, even if it reduces which endings are available.

Goal Priority Prune

When a character has multiple goals, the experience manager will run into situations where a character is split between two goals. Without any sort of priority, the experience manager will proceed to select actions towards the characters first goal. This however leads to problems where characters may endless try to reach a recently achieved goal that was undone to proceed the next goal as shown in the following example.

No Priority Goals	Higher Priority Goals
(Guard wants to be at the Market) ✔	[1] (Guard wants to be at the Market) ✔
Player reports to the Guard the Bandit at the Crossroads	Player reports to the Guard the Bandit at the Crossroads
(Guard wants to be at the Market) ✔ (Guard wants to kill the Bandit) ✘	[1] (Guard wants to kill the Bandit) ✘ [2] (Guard wants to be at the Market) ✔
Guard walks to Crossroads	Guard walks to the Crossroads
(Guard wants to be at the Market) ✘ (Guard wants to kill the Bandit) ✘	[1] (Guard wants to kill the Bandit) ✘ [2] (Guard wants to be at the Market) ✘
Guard walks to Market	Guard kills the Bandit
(Guard wants to be at the Market) ✔ (Guard wants to kill the Bandit) ✘	[1] (Guard wants to kill the Bandit) ✔ [2] (Guard wants to be at the Market) ✘
Guard walks to Crossroads, etc.	Guard walks to Market
(Guard wants to be at the Market) ✘ (Guard wants to kill the Bandit) ✘	[1] (Guard wants to kill the Bandit) ✔ [2] (Guard wants to be at the Market) ✔

Table 6: Example of Goal Priority Prune

Cyclic Prune

The above prune does not prevent all cycles, so we detect cycles of 3 or more edges and break them. When an NPC has multiple actions, they can take in a state, we prune those which are part of a cycle. If every edge in a cycle is that NPC’s only action for that state, we prune the one which is part of the longest plan (i.e. we prefer to remove a step that requires two more steps after it to achieve the agent’s goal over one that only requires one more step after it).

Player Knowledge Prune

When the simulation starts, each player knowledge attribute is set to unobserved. These attributes are the simulation's model of player knowledge. Given multiple NPC actions, the action that leads to observing a player knowledge attribute (whether known or unknown) as quickly as possible is preferred. This is also a tie breaking prune; if there exists only one NPC edge, it will not be pruned. After player knowledge pruning, the story graph is reduced as seen in *Table 1*.

We prioritize keeping unique endings available over learning about the player. If we had done the reverse, the following example could have occurred: The Officer angers the suspect, so Suspect immediately raises the knife. This is the quickest way for the simulation to determine if the officer will use justified force. However, this also immediately eliminates all endings where the Suspect did not threaten the Officer, making it impossible to achieve the best score.

Instead, the Suspect approaches the Officer first, then raises the knife, keeping more unique endings available and still allowing the simulation to learn if the Officer will use justified force.

Arbitrary Prune

At this point, there may still be a few states that have multiple possible NPC actions. We treat all of these actions as equally good. Consequently, we arbitrarily prune by choosing the first action. After arbitrarily pruning, the story graph sizes are as shown in *Table 1*.

Dead End Pruning

It must always be possible for the story to end. The story ends when one of the author's goals is achieved. We define a dead end to be a node from which it is impossible to reach a terminal node. In the final round of pruning, we remove NPC edges to ensure that no dead ends are reachable. Note that we only ever remove NPC edges, never player edges; in other words, we avoid the need to intervene by ensuring the narrative never reaches a state where intervention might be necessary. After dead end pruning, the graph reduction is shown in *Table 1*.

EVALUATION

Police Use of Force Training Simulation

In this simulation, we want to show that participants performed better as time progresses using our experience manager.

Methodology

Participants / Experiment

This study was conducted with 21 civilian participants consisting of university students and staff. The study took up to one hour per participant. The participant was assigned a unique ID number to preserve anonymity. A video was shown to the participants about the simulation they were about to experience and instructions about which buttons performed which actions in the simulation. This simulation was used for other studies outside of this thesis, hence we asked participants to play both virtual reality and screen and keyboard inputs. The first half of our participants were asked to play the simulation between two and ten times using the screen and keyboard input. Whereas the second half of our participants were asked to play the simulation between two and ten times using virtual reality (VR) input. All participants then played the alternate set of inputs between two and ten times.

Experimental Design

Hypothesis 1 for Police Use of Force Player Knowledge Score

The null hypothesis is that there is no relationship between the score received by the participant and the time played. The alterantive hypothesis is there is a signficant and positive relationship that shows the more time a participant plays, the higher the participants score. The participants played between two and ten times, and at the end of each session the score was displayed to the user (as well as recorded for the study).

Hypothesis 2 for Police Use of Force Player Knowledge Attributes

The null hypothesis is there is no relationship between the overall player knowledge attributes tracked for the participant and the time played. The alterantive hypothesis is there is a signficant and positive relationship that shows the more time a participant plays, the higher the tracked player knowledge. The participants played between two and ten times, and at the player knowledge was recorded for the entire study and never shown to the participant.

P-Value Significance

We consider results for which $p <= 0.1$ marginally significant, and results for which $p < 0.05$ significant.

Results

Results 1 for Police Use of Force Score

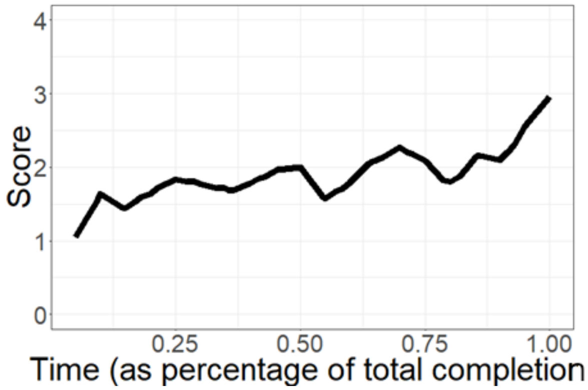

Figure 8: Average score of all participants as they progressed through the simulation

A mixed-effect multilevel model predicting scores from treatment group, controls used, and session number nested within participant revealed that participant scores increased over repeated sessions ($t_{145} = 4.609$, $p < .0001$) as shown in *Figure 8*. This supports our alternate hypothesis where participants score higher the more they play.

Results 2 for Police Use of Force Player Knowledge Attributes

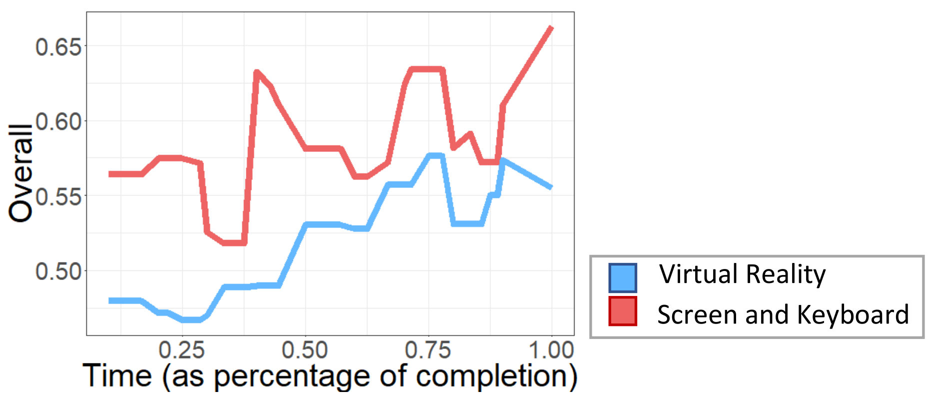

Figure 9: Overall average knowledge attribute scores for all subjects as they progressed through the simulation

As an alternative measure of performance, we automatically analyzed each subject's session log based on the five player knowledge attributes identified earlier. Each component of the player knowledge vector can be represented with the following values:

- Unobserved: Simulation does not have enough information to determine whether the knowledge attribute is known.
- Known: Simulation has observed an action that indicates the subject knows the attribute.

- Not Known: Simulation has observed an action that indicates the subject does not know the attribute.

A subject's overall average knowledge can be represented as the number of known attributes divided by the total number of known values (i.e. non Unobserved values) *Figure 9* shows that overall knowledge significantly increases as the participant progresses through the simulation ($t_{278} = 2.198$, $p = .0288$). This validates that the simulation can teach basic use of force principles (assuming a correct operationalization of the knowledge attributes) as time progresses.

Some subjects reported difficulty using the virtual reality controls. Multiple subjects reported that they were afraid to walk backwards for fear of bumping into walls. Walking backward was the most common way for subjects to take cover, and taking cover was required to reach the maximum score, so this may have limited some subjects' performance and may help to explain why overall score on the knowledge attribute was generally higher when using the screen and keyboard controls. These difficulties may be mitigated as virtual reality hardware becomes more common. Alternatively, we may extend the tutorial to include walking backwards so that future subjects will know it is safe.

Camelot Study Domain

In this simulation, we want to show that participants find NPC behavior more believable in our experimental experience manager over the control experience manager.

Control vs Experimental

When an audience observes a sequence of events, there is a human tendency to try to make narrative sense of these events (Bruner, 1991). An audience assumes that characters are acting intelligently even if these actions are random. Hence, we use a random prune story graph as our control.

Our experimental experience manager uses the pruning strategies outlined in the *Story Graph Pruning* section.

Participants / Experiment

This study was conducted with 20 participants consisting of university students and staff. The study took up to 30 minutes per participant. The participant was assigned a unique ID number to preserve anonymity. A video was shown to the participants about the simulations they were about to experience and instructions about which buttons performed which actions in the simulation. The first half of our participants were asked to play the simulation between two and ten times using our control experience manager. Whereas the second half of our participants were asked to play the simulation between two and ten times using our intelligent experience manager. Then all participants then played the alternate experience manager simulation between two and ten times. After playing both versions, a questionnaire was given to participants asking them to compare the believability of the NPC behavior and agency, the power to take meaningful actions and see the results of one's choices (Wardrip-Fruin N. , Mateas, Dow, & Sali, 2009).

Experimental Design

Hypothesis for Camelot Study Believability

The null hypothesis is that participants showed no preference for either experience manager. The alterantive hypothesis is the participants will report increased believable NPC behaviour and agency in the experimental experience manager over the control experience manager. Half the participants played the control version first, where the other half played experimental version first. The participants then played the alternate experience manager and compared believability of NPCs between both experience managers in a questionnaire.

P-Value Significance

We consider results for which $p <= 0.1$ marginally significant, and results for which $p < 0.05$ significant.

Results

Results for Camelot Study Believability

Question #	Prefer Intelligent	Prefer Random	P-Value (corrected)	Effect Size
1	16	4	0.008	0.4
2	13	7	0.132	0.7
3	18	2	< 0.001	0.2
4	16	4	0.008	0.4

Table 7: Camelot within participants believable behavior results

A binomial exact test confirms our hypothesis for three of the four questions in *Appendix A.1: Camelot Questionnaire after Second Version* at the $p < 0.05$ level as shown in *Table 7*. Hence participants found that the behavior in the experimental experience manager was more believable than that of the control experience manager.

DISCUSSION

We were able to successfully implement automatic experience managers by using the pruning strategies in this thesis where participants:

- Successfully learned police use of force knowledge as time progressed using two measures of performance
- Agreed that NPCs were more believable than the control experience manager

These pruning strategies were able to be used on two very different domains where Police Use of Force Simulation was for training and the Camelot Simulation was for entertainment. One of the pruning strategies we employed was the Dead End Prune. This allowed us to create simulations where we never intervened with the player. However, these pruning strategies only affect NPC actions and not the Player actions. The dead end pruning strategy does not guarantee that the story will be finishable. It only guarantees that at any given state, the pruning strategy will not allow an NPC action that will put the story in an unfinishable state. Depending on the domain, a player action may be responsible for making a story unfinishable. Hence other mediation strategies, such as intervention, may need to be employed in other domains to gaurntee that the possible stories are finishable.

FUTURE WORK

The domain for these prototypes were small enough that a story graph could be generated entirely offline. Story graphs, even for small domains, can get very big very fast. As the content of the simulation expands in size and complexity, pruning a complete story graph will be intractable for most domains. However, this work was instructive because it allowed us to consider the long-term consequences of every experience manager decision. In the future, we hope to investigate how these principles can be adapted to graphs being generated in real-time.

In future versions of the police use of force training simulation, the only content provided by a human author will be the domain description and descriptions of possible wrong beliefs about that domain that the simulation should target during training. Wrong beliefs represent a different version of the domain. For example, an officer may not realize that approaching the agitated suspect will cause him to get angry. This misunderstanding can be represented as a version of the domain where the get angry axiom either does not exist or has different preconditions. The set of things a person would do differs based on their beliefs about the domain, and these differences can be used to diagnose what the trainee knows and does not know. This is how we derived the knowledge attributes used to measure subject performance—by identifying actions that only a person who knows or does not know that information would do.

Perhaps the most limiting assumption of this initial work is that the story graph is Markovian. Stories are non-Markovian; different action sequences leading to the same state may require different conclusions. In future work, we hope to explore how tracking the history of events can improve experience management and NPC believability.

REFERENCES

Bruner, J. (1991). The narrative construction of reality. *Critical inquiry*, (pp. 1-21).

Fikes, Richard E.; Hart, Peter E.; Nilsson, Nils J. (1972). Learning and executing generalized robot plans. *Artificial Intelligence, 3*(1972), 251-288.

Garcia, E. T., Ware, S. G., & Baker, L. J. (n.d.). Measuring Presence and Performance in a Virtual Reality Police Use of Force Training Simulation Prototype. *Thirty-Second International Florida Artificial Intelligence Research Society Conference*, (pp. 276-281).

Ginsberg, Matthew; Geddis, Donald F. (1991). Is there any Need for Domain-Dependent Control Information? *Proceedings of the Ninth National Conference on Artificial Intelligence*, (pp. 452-457).

Harris, J., & Young, R. M. (2005). Proactive mediation in plan-based narrative environments. *International Workshop on Intelligent Virtual Agents*, (pp. 292-304).

Haslum, Patrik. (2012). Narrative Planning: Compilations to Classical Planning. *Journal of Artificial Intelligence Research, 44*, 383-395.

Hays, R. T., & Singer, M. J. (1989). Simulation fidelity in training system design: Bridging the gap between reality and training. Springer-Verlag New York Inc.

Hoffmann, J; Nebel, Bernhard. (2001). The FF planning system: fast plan generation through heuristic search. *Journal of Artificial Intelligence Research, 14*, 253-302.

Li, Boyang; Lee-Urban, Stephen; Johnston, George; Riedl, Mark O. (2013). Story Generation with Crowdsourced Plot Graphs. *Proceedings of the 27th AAAI Conference on Artificial Intelligence*, (pp. 598-604).

Penberthy, J. Scott; Weld, Daniel S. (1992). UCPOP: A Sound, Complete, Partial Order Planner for ADL. *Proceedings of the 3rd International Conference on Principles of Knowledge Representation and Reasoning, Volume 92*, pp. 103-114.

Police Executive Research Forum. (2012). *An Integrated Approach to De-escalation and Minimizing Use of Force.*

Riedl, M. O., & Bulitko, V. (2013). Interactive narrative: An intelligent systems approach. *AI Magazine*.

Riedl, M. O., & Young, R. (2010). Narrative planning: balancing plot and character. *JAIR 39*, (pp. 217-218).

Riedl, Mark; R, Michael Young. (2014). The Importance of Narrative as an Affective Instructional Strategy. *Adaptive Instructional, 2*.

Roberts, David L.; Isbell, Charles L. (2008). A survey and qualitative analysis of recent advances in drama management. *International Transactions on Systems Science and Applications, Special Issue on Agent Based Systems for Human Learning*, *4*, 61-75.

Russell, S.J. and Norvig, P. (2016). *Artificial Intelligence: A Modern Approach.* Pearson Education.

Shirvani, A., Ware, S. G., & Farrell, R. (2017). A possible worlds model of belief for state-space narrative planning. *In Proceedings of the 13th AAAI International Conference on Artificial Intelligence and Interactive Digital Entertainment*, (pp. 101-107).

Wardrip-Fruin, N., Mateas, M., Dow, S., & Sali, S. (2009). Agency reconsidered. *In Proceedings of the Digital Games.*

Wardrip-Fruin, N., Mateas, M., Dow, S., & Sali, S. (2009). Agency reconsidered. *Proceedings of the Digital Games Research Association.*

Ware, S. G. (2014). Glaive: a state-space narrative planner supporting intentionality and conflict. *In Proc. of AIIDE*, (pp. 80-86).

Ware, S. G., & Young, R. (2011). CPOCL: a narrative planner supporting conflict. *Proceedings of the 7th AAAI International Conference on Artificial Intelligence and Interactive Digital Entertainment*, (pp. 97-102).

Wilkins, David E. (1983). Representation in a Domain-Independent Planner. *Proceedings of the Eighth International Joint Conference on Artificial Intelligence*, (pp. 733-740).

APPENDIX

Appendix A.1: Camelot Questionnaire after Second Version

[Answers: The first version, The second version]

1. In which version did the characters feel more realistic?
2. In which version did the characters do a better job of reacting to things they saw and ignoring things they did not see?
3. In which version did the characters do a better job of trying to accomplish their goals?
4. In which version did you feel like your actions had more effect on the story?

VITA

Edward Thomas Garcia obtained his Bachelor of Science degree in Electrical Engineering with Computer Engineering Concentration and Communications Minor in 2005. He obtained his Master of Business Administration degree in 2009. He is employed by the University of New Orleans as a computer programmer working with the university's Student Information System. He joined the Computer Science graduate program to pursue a PhD degree with a concentration in Artificial Intelligence starting 2014. This research work was performed under the supervision of Dr. Stephen G Ware and presented in 2019.