Proceedings of the Fourteenth Artificial Intelligence and
Interactive Digital Entertainment Conference (AIIDE 2018)

Postmortem: MKULTRA, An Experimental AI-Based Game

Ian Douglas Horswill
Northwestern University, Evanston IL, USA
ian@northwestern.edu

Abstract

Games are inherently situated within the cultures of their players. Players bring a wide range of knowledge and expectations to a game, and the more the game suggests connections to that culture, the stronger those expectations are and/or the more problematic they can be. *MKULTRA* is an experimental, AI-heavy game that ran afoul of those issues. It's interesting to hear a talk about or to see demonstrated by the author, but frustrating for players who do not already understand its internals in some detail.

In this paper, I will give a postmortem of the game, in the rough style of industry postmortems from venues such as Gamasutra or GDC. I will discuss the goals and design of the game, what went right, what went wrong, and what I should have done instead. In my discussions of the game's problems, I'll focus on the ways in which it frustrated the players' cultural expectations, and what we can learn from them for the design of future games.

Introduction

MKULTRA is an experimental, AI-heavy game intended to explore novel, AI-centered game mechanics (I. Horswill, 2014b, 2014a). Roughly speaking, it consists of a 1980s-style Prolog natural language system (Pereira & Shieber, 1987; Pereira & Warren, 1980; David H. D. Warren & Pereira, 1982), combined with a 1990s-style reactive planner (McDermott, 1978; Sibun, 1992), running in a RPGMaker-style tile-based world. The agents in the game – the player, the player character, and the NPCs – can all interact using generative natural language dialog, including questions, imperatives, and declaratives, within a restricted grammar of English and a lexicon of a few hundred words.

The game was envisioned as a mystery where the player was cast in the role of Betsy, a detective with psionic abilities. The mystery narrative provided an impetus for natural language interaction – the player must talk to characters to collect information – and the psionic abilities afforded a number of novel gameplay mechanics, most notably *belief injection* (see below).

Development proceeded as far as a playable demo with fully implement character AI and drama management. However, even the demo has proven very difficult for naïve users to play. The problem is not that the puzzles are too hard, but that learning the limits of the system is too hard.

A succession of tutorial mechanisms were added to the game: autocompletion, hinting mechanisms, active prodding from NPCs, and finally, pop-up menus to suggest promising actions to the player. But none were sufficient to allow any of the 30 or so playtesters to complete the demo level without help. As a result, development of the game has been suspended pending a fundamental redesign of the gameplay.

In this paper, I will discuss the development of the game in the style of a game industry postmortem: goals, technology, what went right, what went wrong, and what I should have done instead. The intent here is not to document specific technical innovations, since those have been written about elsewhere. It is rather to talk about why those innovations were insufficient, and how we can apply those lessons to future games.

Much of the failure of the system involves the cultural expectations players brought to the game, and the game's inability to fulfill them. These failures were twofold: the game's characters suggested a deeper understanding of human culture than they actually possessed, and the game's similarities to existing genres misled players to expect different kinds of gameplay than the game was able to support.

Goals

The game's original goal was to build a next-generation interactive narrative in the tradition of Mateas and Stern's *Façade* (Mateas & Stern, 2005). Periodic attempts by the game AI research community to build full-fledged games are important. And while there has been a great deal of work on interactive narrative since *Façade* (see, for example (Evans & Short, 2014; I. D. Horswill, Montfort, & Young, 2014;

Roberts & Isbell, 2008; Robertson & Young, 2015), the only recent attempt to build a similarly complete, AI-heavy interactive narrative piece has been Evan's and Short's *Versu* platform (Evans & Short, 2013), which did not come from the research community.

The primary technical goal was to integrate simple but generative natural language with true semantic parsing into an AI-based interactive narrative. *Façade* did something closer to categorization of utterances than the production of true logical forms (Mateas & Stern, 2004); the system could understand that you were agreeing with a character or that you had insulted them. But if you asked a character a question, the system literally couldn't represent the content of the specific question.

Finally, the project had a set of game design goals; it was an exercise in AI-based game design (Eladhari, Sullivan, Smith, & Mccoy, 2011). The central design problem it sought to solve was to design gameplay that was robust with respect to the AI system's failure modes. Any practical character AI system will have limited knowledge, vocabulary, reasoning abilities, and so on. The project sought to design the mechanics and the narrative of the game so as to compensate for those limitations (I. Horswill, 2014b).

One example of this was the *belief injection* mechanic. The player can manipulate an NPC's behavior by injecting false beliefs directly into its knowledge base. This is a fun puzzle mechanic: how do you figure out the character's current belief structure, so you can change it to accomplish your goal? It's also a narrative alibi for the character's inevitable failures of planning, inference, and social norms; the character can be forgiven for doing stupid things because it's a mind-controlled zombie. Put another way, by making AI debugging a form of gameplay, we reduce the intrusiveness of the AI's fragility. The player sees it as a failure of their own problem solving rather than as a bug in the game itself.

Another example was the surfacing of the AI's limitations in the user interface. *MKULTRA*'s vocabulary is several hundred words, not the tens of thousands of a fluent speaker. Teaching the player what the system can and cannot understand is difficult. So the system scaffolds the player's learning using a novel autocomplete mechanism (I. Horswill, 2014a). The system uses a reversible parser/generator that allows it to take a partial user inputs as they are typed, solve for its possible completions within the subset of English the system understands, and display those completions in real time, introducing the player to examples of sentences it understands. If there are no possible completions, the system knows the player has typed an invalid input and gives them immediate feedback rather than letting them continue typing a bad command.

Gameplay

The core gameplay is similar to classic parser-based interactive fiction (Jackson-Mead & Wheeler, 2011). By typing commands in English, the player explores the world, gathering information, poking systems to understand how they work, and ultimately solving puzzles. Unlike parser-based IF, the systems being poked are autonomous, AI-driven NPCs running a simple NL system and reactive planner.

All characters, including the player character (PC), go about pursuing their own limited goals in the absence of player intervention. The player intervenes by typing sentences in the UI; these "thoughts" are injected into the player character. They can be requests, questions, assertions, responses to questions, or variations such as indirect requests. If the player character is in conversation with an NPC, the sentences are spoken by the player character to the NPC. Otherwise, they are taken as being addressed to the player character. Characters, including the PC, can refuse requests or lie in response to questions. The only exception is belief injection, which characters cannot refuse.

The fundamental mechanics of the game are therefore questions, requests, and belief injection. These mechanics are then used to gather information and solve puzzles.

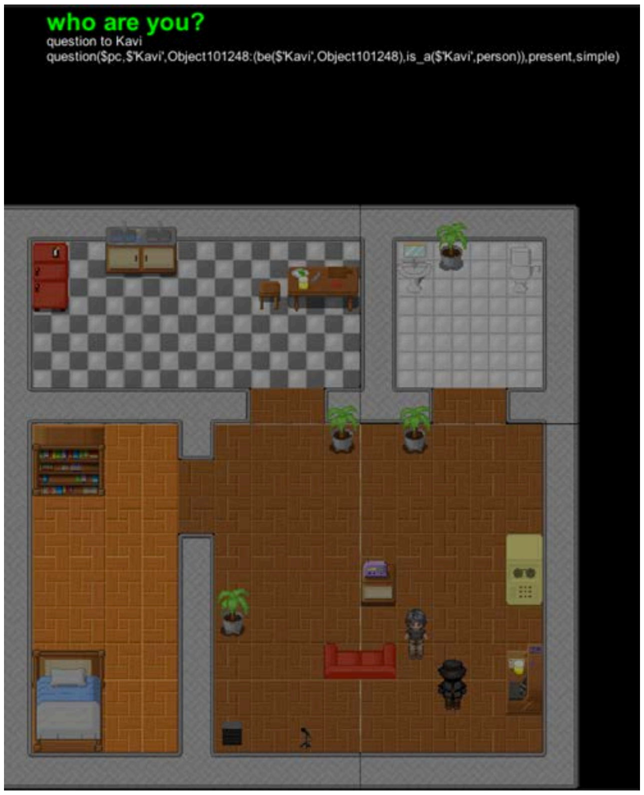
Technology

MKULTRA runs under the Unity3D game engine (Unity Technologies, 2004). The game is free, open-source software available on github.

The AI system is written primarily in a custom-built Prolog interpreter. The Prolog interpreter also implements Evans' *eremic logic* (Evans, 2010; Evans & Short, 2014), which provides a separate, tree-structured knowledge-base that has better update semantics than Prolog's assert/retract interface.

The major components of the AI system consists of:

- A reversible, semantic parser/generator based on definite-clause grammars (Pereira & Shieber, 1987; Pereira & Warren, 1980). It handles single-clause English sentences without quantifiers, in a variety of grammatical moods, tenses, and aspects.
- A committed-choice reactive planner inspired in part by the SALIX system (Sibun, 1992).
- A dialog system that can respond speech acts involving questions, answers, assertions, and imperatives. It is essentially an elaborate version of CHAT-80 (David H. D. Warren & Pereira, 1982).
- A simple ternary logic programming system reasoning about the knowledge states of characters to help them reason about the distinction between not knowing a fact and knowing the fact to be false.



```
who are you?
question to Kavi
question($pc,$'Kavi',Object101248:(be($'Kavi',Object101248),is_a($'Kavi',person)),present,simple)
```

Figure 1: MKULTRA demo level

- A lying system to allow characters to reason about whether they should answer a question truthfully.
- A limited moral reasoning system (Blass & Horswill, 2015) based on defeasible Prolog (Nute, 1993).
- A beat-sequencing system, similar to that of *Façade* (Mateas & Stern, 2002)
- An elaborate context-dependent hinting system to suggest player actions based on the world state, narrative state and the knowledge states of the characters

In addition, the game contains a large number of UI technologies to help on-board the player:

- The incremental parsing/autocompletion system discussed above.
- All objects in the world support pop-up menus showing possible interactions with the object in question. Menu entries can be nominated by the object, the current narrative beat, or specialized knowledge about the player character's goals.
- Menu entries take the form of equivalent sentences the player could type through the dialog menu, giving the player further examples of the kinds of sentences the system understands.

Demo level

Although the character AI and beat sequencing systems were implemented, only a demo level was constructed because of the playability problems with the game.

In the demo level, the player plays Betsy, a psionic spy whose has had her maguffin stolen by her friend Kavi, who is also a spy, but who works for Betsy's opponents, the illuminati. Betsy needs to find and recover her macguffin without being exposed and killed by Kavi.

The level begins with Kavi welcoming Besty and telling her to make herself at home, but warning her to stay out of the bedroom. Kavi then wanders off to the kitchen. The player can now explore the house or talk further to Kavi.

Searching can be achieved by telling Betsy to examine specific objects, or just by giving her broader directions such as *search the bedroom*, or even *search the house*. If the player searches the bedroom, they will find the macguffin, but Kavi will come and kill her.

The player can talk to Kavi by saying *talk to Kavi*, *ask Kavi to bring me the macguffin*, *ask Kavi if he's a member of the illuminati*, etc. If Betsy isn't already in a conversation with Kavi, she'll walk over to him and begin a conversation. Kavi will deny any knowledge of the illuminati or the macguffin and will attempt to kill Betsy if she finds the macguffin.

Solving the level requires disabling Kavi in some way. There are a number of ways of doing this, including:

- Brainwashing Kavi to believe Betsy is a member of the illuminati, and therefore allowed access to the macguffin. Kavi will then even deliver the macguffin if asked to do so.
- Brainwashing Kavi to believe Betsy is an inanimate object and therefore not a threat.
- Brainwashing Kavi to believe that he is hungry and that he is himself food. Kavi then fatally eats himself.
- Telling Betsy to put Kavi in the refrigerator. Because of a bug, Kavi gets stuck in the refrigerator (he doesn't realize he needs to leave it before trying to walk elsewhere). This is a sufficiently amusing bug that it was left in.

MKULTRA was playtested at a number of venues (AIIDE, EXAG, Indie City Meetup), and less formally at Northwestern University and GDC, for a total of around 30 playtesters. In addition, its codebase was used in two undergraduate classes. No player ever succeeded in completing the demo level without help.

What Went Right

Although unsuccessful as a game, there were aspects of *MKULTRA* that worked very well.

Technology

The Unity Prolog interpreter built for *MKULTRA*, which is open source and distributed on github, was an unexpected success. It has been used for a half dozen projects in the US, South America, the Caribbean, and Europe. It was also used in a successful commercial game, *Project Highrise* (Viglione & Zubek, 2016). In general, Prolog was a very useful language for AI prototyping. It allowed high programmer productivity while being sufficiently performant for the needs of the game.

Definite clause grammars (Pereira & Warren, 1980) worked well for the kinds of limited vocabularies and grammars being used in the game. The ability to embed arbitrary Prolog code in the productions was useful, as it allowed the parser to call out to the game engine for resolving certain kinds of NP references. And although DCGs are a backtracking, recursive descent parsing technology, their performance was sufficient for the game.

Evans' eremic logic (Evans, 2010) was also very useful for maintaining both the AI's state and for communicating with the underlying game engine. Without its ability to delete or replace large chunks of state with a single operation, the game would almost certainly have had large numbers of bugs related to incomplete state cleanup.

Reactive planning was sufficient for the simple tasks the characters had to solve in the game. There were few cases where the kinds of simulation-based lookahead that a full planning system (Ghallab, Nau, & Traverso, 2004) provides would have been useful. That said, there were cases where discourse planning would have been simpler if there had been some limited lookahead capability.

Gameplay

Even though the NL and planning capabilities of the characters were primitive by modern standards, they added sufficient autonomy to the characters to make them interesting systems to poke. They were sufficiently reliable and performant to demonstrate the practicality of allowing the player to task NPCs in a simple, generative language.

Players enjoyed being able to type *search the house* and have the character know what to do; it's far less tedious than manually piloting the character from one location to another. Similarly, it's nice to be able to walk up to an NPC and say "can I have an apple?" and have them go to the kitchen, open the refrigerator, extract an apple, walk back to you, and give it to you. It would be interesting to explore the use of this kind of interaction for sidekick/companion characters.

It is also (often) entertaining to watch the system behave strangely. There is considerable humor value in telling a character to eat himself or to put another character in the refrigerator.

What Went Wrong

The game confused players. Even though the system would explicitly prompt players to type, players were often unsure how to interact with the system and would simply stare at the screen.

Negative Transfer from Existing Genres

One issue was that the game has sufficient superficial similarities to existing genres that players would repeatedly try to deploy the conventions of those genres. Players would, for example, type "n" for "north" rather than "go to the kitchen," since that's a widely adopted convention of parser-based text adventures. Others would attempt to pilot the player character using the arrow keys, as if it were a typical tile-based RPG.

Many players expected the puzzles to be in the environment rather than in the NPCs. They would ignore Kavi in favor searching for trap doors or hidden safe combinations. Many players missed that they'd been given an objective and blindly explored the environment, not being sure what to look for.

Expectation Violation

The biggest problem players had was determining what actions were available to them. The natural language interface presents the appearance of open-ended interaction, and does deliver it in a limited sense. But it only understands a small fragment of English. Players spent nearly all their time trying to guess what sentences it would accept. While the autocompletion interface helped, it was better at telling them an input was invalid that what valid input to use instead.

Players also had an analogous problem with understanding the boundaries of the AI's knowledge of the world. Kavi is not an actual intelligent creature. He's a clockwork mechanism that understands two things:

- Only admit to knowledge of the macguffin or the illuminati to members of the illuminati
- Kill any people who go into the bedroom unless they're members of the illuminati

This means he can be circumvented by convincing him Betsy is a member of the illuminati, is not person, or is somewhere other than the bedroom.

Unfortunately, Kavi behaves just enough like an intelligent creature that players overattribute knowledge to him. They tried to threaten, bribe, trick, flatter, or seduce him, or otherwise compromise him in any number of ways about which the system had exactly zero understanding. The player can easily waste large amounts of time trying to guess what verb to use to threaten Kavi, when the system literally doesn't know what a threat is. Even though Kavi's AI is considerably more sophisticated than ELIZA (Wiezenbaum, 1966), it nevertheless falls prey to the ELIZA effect (Wardrip-Fruin, 2012).

This relates to the final issue, which we might call the non-systematicity of knowledge. If a word is used during the game, the player naturally assumes the system has general knowledge about it. For example, the fact that there is furniture in the demo level will cause players use commands such as *look under the sofa*. But while the game understands that the object occupying that pair of tiles is a "sofa" and sofas are a kind of container you can sit and lie on, it doesn't understand that there is normally space underneath one of them where something could be lost or hidden. Again, the player can waste large amounts of time trying to find ways of looking underneath the sofa or between its cushions, even though the game doesn't know what a cushion is. This is in some ways like Wardrip-Fruin's TaleSpin Effect (2012), wherein an AI system appears more simple-minded than it actually is. However, in this case, the system truly is simple-minded about furniture.

Similarly, the characters know some things about the CIA, e.g. that it's a government agency, but not any of its historical or cultural legacy. You can't ask characters about the CIA's role in the cold war, because they don't know what communism is; it isn't relevant to the plot. Unfortunately, players don't know it's irrelevant.

This issue is particularly problematic when it comes to scripted dialog, which can be spoken by an NPC when a narrative beat deems it necessary. Such dialog often contains words, idioms, or grammatical constructions the system doesn't understand at all.

Failure and Cultural Context

Games are always situated within a cultural context. That context provides the background for the game's intelligibility to the player. Well-designed games are matched to this context so as to leverage player expectations and understanding. This issue of cultural alignment is a useful lens for analyzing *MKULTRA*'s problems.

Game culture

Part of the cultural context is the specific culture *of games*. While different players have different levels of familiarity with a given genre or with games in general, most players come to a game with background knowledge of at least some of the tropes and mechanics of common game genres.

Consider the different kinds of role-playing game genres. Most players of parser-based interactive fiction have considerable experience with the genre and come to the game with the expectation that the game will implement the standard verbs and sentence structure of parser-based IF unless there is a clear good reason not to. These conventions did not come about overnight or at random; they were developed over a period of decades through explicit discussion amongst authors and players (Jackson-Mead & Wheeler, 2011; Nelson, 1995). Parser-based IF players are also more likely to view a certain amount of trial and error as being part of the puzzle, and hence the pleasure, of the game.

Compare these to computer RPGs, such as the *Final Fantasy* series. These take the explicitly pedagogical approach that has become standard in video game level design: the game is taught during the first few levels or scenarios, each of which teaches the player about a small set of new capabilities. If the game has a manual at all, it is largely limited to listing key or button bindings.

By contrast, table-top RPGs literally *are* their manuals. TTRPG players are expected to invest time reading and learning about the game before they ever start playing. And game masters are expected to invest hours or days.

General human culture

In addition to their specific game knowledge, players bring all the general knowledge they share with other members of their culture: their language, their beliefs about social and gender norms, their understanding of cooking and television and politics. When we interact with people we use that knowledge to interpret and predict their actions, and assume they share and use it too.

The characters in *MKULTRA* present themselves as members of some approximation to contemporary culture, or to be accurate, North American, white, middle class, English-speaking culture. They have commonsense knowledge about space, containers, and so on. They lie. They understand social norms such as excusing oneself when leaving, and ethical norms such as it being inappropriate to eat other people.

Unfortunately, even players who aren't a part of this culture know more about it than any present-day AI system could possibly know. They bring a wide range of cultural expectations that the system cannot possibly live up to. Players repeatedly bump up against these limits of the character's background understanding.

What made it a problematic game

MKULTRA is a kind of perfect storm of cultural misunderstandings. The game *looks* like a tile-based RPG, creating false expectations of what character and environmental interaction should be like. It also looks like parser-based IF, creating a different set of false expectations. Plus, unlike either of them, it has autonomous characters and generative natural language, setting up a third set of expectations it can't possibly fulfill.

These would be less of a problem if *MKULTRA* were purely a puzzle game. We could restrict the sentences the characters say to stay well clear of the boundaries of its understanding. We could foreground the limited, and mechanical nature of their understanding, making the characters into a game system for the players to reason about in the same way they'd reason about game economies. We could adopt classical, pedagogical level design, having each successive level explicitly introduce a few words of vocabulary. The player would never wonder what the game's subset of English was because it would have been explicitly taught to them.

However, these kinds of interventions are incompatible with the game's original goal of immersive interactive narrative. It doesn't work to tell the player they're role-playing a detective using generative natural language, but then say "this is level one; you can use these five words." For interactive narrative, a choice-based interface, where the player is explicitly presented with the options that are viable in the immediate circumstances would work much better than the generative-but-limited NLU of *MKULTRA*.

This, I think, is the single biggest problem with the game: it needed either to be a *Façade*-like interactive narrative that encouraged players to willfully suspend disbelief, or it needed to commit fully to being a puzzle game that encouraged the player to think of the characters not as humans but as dumb AI systems to be reverse-engineered by the player. Trying to do both created a dilemma in which the more successful it was at one, the less successful it was at the other. The semantic parsing that produced interesting puzzle gameplay actively impeded narrative immersion, and was in turn, actively impeded by the game design constraints introduced by the narrative aspects of the game.

To be clear, many games, including point and click adventures from *Myst* (Miller & Miller, 1993) onward, successfully combine puzzles and narrative. Typically, the puzzles form the gameplay, and the advancing of the narrative forms a part of the reward for solving the puzzle.[1] *MKULTRA*'s problems stem from an unsuccessful attempt to use generative natural language for both these elements at once.

Conclusion

As AI researchers, we tend to think of character AI as a set of distinct technical capabilities. But when the player plays the game, they experience the characters as an overall *gestalt*. One aspect of the *gestalt* is a continuum between what we might call *clockwork characters*, in which the mechanical nature of the underlying AI is foregrounded in the player's mind, and (for want of a better term) *lifelike characters*, in which the player is less conscious of the character's mechanical nature. This continuum is highly sensitive to both gameplay design and AI capabilities and robustness.

Gameplay that treats characters as puzzles to be solved emphasizes the mechanical nature of characters. The player must understand the AI in detail and think of it as a puzzle. The player can repeatedly play through the same sequence of actions and responses without losing the illusion of life because there was little illusion to begin with.

By contrast, lifelike characters are better for narrative immersion, at least insofar as the narrative is about the characters. The illusion of life is delicate, however.

Many of *MKULTRA*'s problems stem from unsuccessfully trying to make characters that are simultaneously clockwork and lifelike. These problems suggest some general design lessons for each type.

Clockwork characters must be as legible as possible to the player. Players must be able to predict character behavior well enough to accomplish their goals in the game. It must be easy for the player to learn character behavior, and any relevant internal state must be readily apparent to the player. The more rules and state underlying the character's behavior, the longer it will take the player to learn and master them. If the gameplay requires the player to reliably predict character behavior, then this learning time effectively limits the allowable complexity of character behavior.

Predictability is less important for lifelike characters than the need not to be overtly confusing or stupid. Their behavior can be complex and even unpredictable, so long as it is narratively appropriate. Characters need reasonable default behaviors to perform when they encounter unanticipated situations. For example, in *Façade*, characters generally grunted or entirely ignored inputs they didn't understand and continued performing their current beats. While suboptimal, this is preferable to an endless cycle of "I'm sorry; I didn't understand that, could you repeat it?" It's often preferable for a character to default to a less intelligent, or less specific, response to a situation than to risk an inappropriate behavior that breaks the illusion of life.

The clockwork/lifelike spectrum is only one aspect of the player's experience of character AI. *MKULTRA*'s problems stemmed in part from trying to be both at once, which placed competing pressures on both gameplay and technology. At the same time, it raised a set of cultural expectations in the player that it couldn't ultimately fulfill. While still interesting and engaging to see demonstrated, a truly successful version of the game will require a gameplay redesign that resolves these issues.

Acknowledgements

I'd like to thank Ethan Robison, Rob Zubek, Robin Hunicke, James Ryan, Michael Mateas, Richard Evans, all the various playtesters, and the students of the various courses that used *MKULTRA* for their kind advice and feedback. I'd also like to thank the reviewers for their insightful comments. They helped improve this document considerably.

[1] Although see (Montfort, 2005) for a much more sophisticated analysis.

References

Blass, J., & Horswill, I. (2015). Implementing Injunctive Social Norms using Defeasible Reasoning. In *Workshop on Social Believability, Proceedings of the Eleventh AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*. Santa Cruz, California.

Eladhari, M. P. E. Al, Sullivan, A., Smith, G., & Mccoy, J. (2011). AI-Based Game Design : Enabling New Playable Experiences. *Technical Report, UCSC-SOE-11*.

Evans, R. (2010). Introducing Exclusion Logic as a Deontic Logic. In *Deontic Logic in Computer Science, Proceedings of the 10th International Conference, DEON 2010, Lecture Notes in Computer Science Volume 6181* (pp. 179–195). Fiesole, Italy: Springer.

Evans, R., & Short, E. (2013). Versu. San Francisco, CA: Linden Lab.

Evans, R., & Short, E. (2014). Versu - A Simulationist Storytelling System. *IEEE Transactions on Computational Intelligence and AI in Games*, *6*(2), 113–130.

Ghallab, M., Nau, D., & Traverso, P. (2004). *Automated Planning: Theory & Practice*. Morgan Kaufman.

Horswill, I. (2014a). Architectural issues for compositional dialog in games. In *AAAI Workshop - Technical Report* (Vol. WS-14-17).

Horswill, I. (2014b). Game design for classical AI. In *AAAI Workshop - Technical Report* (Vol. WS-14-16).

Horswill, I. D., Montfort, N., & Young, R. M. (2014). Guest editorial: Computational narrative and games. *IEEE Transactions on Computational Intelligence and AI in Games*, *6*(2).

Jackson-Mead, K., & Wheeler, J. R. (Eds.). (2011). *IF Theory Reader*. Boston, MA: > Transcript On Press.

Mateas, M., & Stern, A. (2002). A Behavior Language for Story-Based Agents. *IEEE Intelligent Systems*, *17*(4), 39–47.

Mateas, M., & Stern, A. (2004). Natural Language Understanding in Façade: Surface-text Processing. In *Technologies for Interactive Digital Storytelling and Entertainment (TIDSE)*. Darmstadt, Germany.

Mateas, M., & Stern, A. (2005). Façade.

McDermott, D. (1978). Planning and acting. *Cognitive Science*, *2*(2), 71–100.

Miller, R., & Miller, R. (1993). Myst. Cyan, Inc.

Montfort, N. (2005). *Twisty Little Passages: An Approach to Interactive Fiction*. Cambridge, MA, USA: MIT Press.

Nelson, G. (1995). The Craft of Adventure. Retrieved from http://mirror.ifarchive.org/if-archive/info/Craft.Of.Adventure.pdf

Nute, D. (1993). Defeasible prolog. *AAAI Fall Symposium on Automated Deduction in Nonstandard Logics*, 105–112.

Pereira, F. C. N., & Shieber, S. (1987). *Prolog and Natural Language Analysis*. Brookline, MA: Microtome Publishing.

Pereira, F. C. N., & Warren, D. H. D. (1980). Definite Clause Grammars for Language Analysis - A Survey of the Formalism and a Comparison with Augmented Transition Networks. *Artificial Intelligence*, *13*(231–278).

Roberts, D. L., & Isbell, C. L. (2008). A survey and qualitative analysis of recent advances in drama management. *International Transactions on Systems Science and Applications*, *4*(1), 61–75.

Robertson, J., & Young, R. M. (2015). Interactive Narrative Intervention Alibis through Domain Revision. *AIIDE*.

Sibun, P. (1992). *Locally Organized Text Generation*. University of Massachusetts, Amherst.

Unity Technologies. (2004). Unity 3D. San Francisco, CA.

Viglione, M., & Zubek, R. (2016). Project Highrise. Chicago: SomaSim, LLC.

Wardrip-Fruin, N. (2012). *Expressive Processing: Digital Fictions, Computer Games, and Software Studies*. Cambridge, MA: MIT Press.

Warren, D. H. D., & Pereira, F. C. N. (1982). An efficient easily adaptable system for interpreting natural language queries. *Computational Linguistics*, *8*(3–4), 110–122.

Warren, D. H. D., Pereira, L. M., & Pereira, F. (1977). PROLOG - The Language and its implementation compared with LISP. In *Symposium on AI and Programming Languages* (Vol. 12, pp. 109–115). ACM.

Wiezenbaum, J. (1966). ELIZA. *Communications of the ACM*, *9*, 36–45.