Proceedings of the Seventeenth AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment
(AIIDE 2021)

Crosston Tavern: Modulating Autonomous Characters Behaviour through Player-NPC Conversation

Elisabeth Oliver, Michael Mateas
Expressive Intelligence Studio, University of California, Santa Cruz
elaolive@ucsc.edu, michaelm@soe.ucsc.edu

Abstract

NPCs (Non-Player Characters) are a staple of video games, filling all kinds of supporting roles. This work seeks to flip that paradigm and place the player in the role of support for the goals of a small collection of NPCs enabling a new kind of AI-driven gameplay. Built on a world simulation where NPCs can take action according to their goals and knowledge of the world state and a conversation space in which the NPC is able to report their actions and exchange information with the player, this prototype AI-based game design explores a new player-NPC interaction in which player conversational actions indirectly influence the NPC simulation. In this paper we discuss the architecture, provide a design postmortem, and report the results of play testing.

Introduction

Few video games are complete without a host of non-player characters (NPCs) to accompany the player on their journey through the game's strange world. Some are quirky and loud, brimming with personality and acting as the main cast of the story. Others provide a sense of background presence, serving as "extras" in the game world, such as the townspeople found in many RPGs. Many NPCs provide direction for the player by offering quests, directly asking the player to accomplish specific tasks for them, ranging from killing monsters, to collecting objects, to saving the world.

But what if NPCs wanted advisors and not gophers? What would a game look like if the NPCs didn't want the player to collect ten apples for them, but instead just wanted to figure out where they could get those apples? What if the game focused on helping NPCs by providing information and direction?

Enabling this kind of gameplay is the focus of this work. *Crosston Tavern* is a prototype AI-based game centered around helping a small cast of NPCs find love, friendship, and good food. The player takes the role of a tavern keeper with a small group of townies who are their regular customers. Every evening these townies wander in, order food, and share their struggles and dreams with the player. And every evening, it is up to the player to share stories of other townies or provide suggestions as to what they might try next to help their goals. During the simulated day, these townies wander their world trying actions based on what they know and what they have been told with the hopes of fulfilling their dreams.

This work seeks to flip the standard paradigm of player-NPC interaction. Instead of stationary or scripted NPCs providing support to the player's goals, here we explore the player providing support to autonomous, goal-driven NPCs, exploring an AI-based game design whose main gameplay loop is organized around AI interaction (Treanor et al. 2015; Eladhari 2015). In the remainder of this paper we discuss related work, describe the gameplay, provide a description of the architecture, and evaluation results from playtesting.

Related Work

Crosston Tavern borrows its setting from the Farming Simulation (Farming Sim) genre of video games. This genre is characterized by playing as a newcomer to a small, rural town. Usually, the player character has just inherited the (now heavily dilapidated) family farm from a deceased grandfather (Amccus 1996; ConcernedApe 2016; Interactive 2007; AQL 2021). Cleaning up and rebuilding this farm becomes the player's initial goal. Additionally, the player is encouraged to meet, befriend, and romance the other townspeople.

These social and resource mechanics became the inspiration for the actions available to NPCs in *Crosston Tavern's* world simulation. NPCs want to form interpersonal relationships with one another. Each NPC has also has a "profession" related goal, which is somewhat analogous to the player's "fixing up the farm" goal. The characters have access to such actions as talking to each other and giving one another gifts (just as the player can talk to and give gifts to NPCs in farming sims), tending their crops, fishing in rivers, and foraging for wild harvestables.

Many video games are built on top of agent-based simulation systems of some sort, such as the games of the *Sims* franchise (Maxis 2014). In these games, play revolves around directing a set of semi-autonomous characters (called sims) in a simulated world. *Crosston Tavern* is similarly built on a simulation. Like the sims, characters form relationships with each other over platonic and romantic dimensions and work to fulfill certain personal goals. However, where the player can directly select the actions of sims, in *Crosston Tavern* the player indirectly influences their actions through

conversation.

The story generator *Tale-Spin* (Meehan 1977) makes use of means-ends problem solving to simulate the the behavior of characters. The resulting stories are the problem solving traces of the characters. Like *Crosston Tavern*, characters within the *Tale-Spin* simulations have goals and are able to reason about the state of the world and other characters to pursue their goals. However, where the focus of *Tale-Spin* was to autonomously generate stories based on character problem solving using an early version of hierarchical task network planning, *Crosston Tavern* simulates, via a combination of goal decomposition and heuristic action selection, the daily actions that the characters talk with the player about each evening.

Prom Week (McCoy et al. 2013) is another game built around simulated actions, this time focusing almost exclusively on those in the social sphere. Set in the week before prom in an average American high school, players help direct a cast of high schoolers through a series of social puzzles to get them ready for prom. What actions are available to a given character is determined by the rule-based utility system CiF (McCoy et al. 2014). CiF and its successor system the *Ensemble Engine* (Samuel et al. 2015) use these influence rules to decide which goals an NPC wants to pursue during the desire formation phase, and which action to perform towards the goal during intent formation. While *Crosston Tavern* also manages and pursues character goals, it makes use of goal decomposition and weights actions according to how well they accomplish goals rather than via authored influence rules.

Versu (Evans and Short 2014) and its logic-based agent model *Praxis* is a social simulation architecture organized around social practices, shared social contexts that provide context-specific actions and state sequencing to characters. Action selection occurs heuristically via one-step lookahead comparing against agent goals. Similarly to social practices, world features in *Crosston Tavern*, such as a pond or field, provide context-appropriate actions to characters. While action-selection also takes place heuristically by comparing action outcomes against goals, sequencing occurs via goal decomposition rather than via the state-machine-like sequencing of social practices.

Talk of the Town (Ryan 2018; Ryan et al. 2015) is a social simulation that models knowledge propagation and mutation that was used in the live action game *Bad News* (Samuel et al. 2016). NPCs in their simulated world observe physical and social traits of others around them and share these observations with one another. As they propagate this information, their observations can mutate, either unintentionally as characters misremember or intentionally as characters lie. Similarly, the NPCs of *Crosston Tavern* maintain separate subjective models of world state that they use for goal decomposition and action selection. Rather than *Talk of the Town's* focus on modeling primarily physical trait knowledge, *Crosston Tavern* focuses on modeling knowledge about world locations and resources (supporting the resource mechanics) and theory of mind models of social state (e.g. what A thinks B thinks about A) supporting the social dynamics. While *Talk of the Town* associated confidence values with knowledge to support knowledge change over time, *Crosston Tavern* NPCs do immediate knowledge updating.

In contrast to *Crosston Tavern*, *CiF* (McCoy et al. 2014) characters share global knowledge, including knowledge of global preferences of cultural props (e.g. zombie movies and roses). *Crosston Tavern* NPCs have individual item preferences (in the current implementation, primarily food) and individually modeled knowledge of other character's preferences. Part of the gameplay is telling other characters these preferences as the player learns them, supporting the NPCs in activities such as gift giving.

Description of Gameplay

Gameplay begins with one of the three NPCs entering the player's bar. After exchanging greetings and ordering food, the player is able to ask the NPC questions such as what the NPC did that day, what their goals are, and what they think of other NPCs, as well as offer advice about actions to pursue and share knowledge about other NPCs and the world (Figure 1).

The answers NPCs provide to these questions are derived directly from the state of the world simulation that happens between each night at the bar. For example, the player might ask "What did you do today?" This queries the NPC's knowledge base of remembered events and filters for the ones that happened on the current day. These events are then curated to a short list of "interesting" (from the NPCs point of view at least) events to tell the player.

By sharing knowledge with characters about other characters activities, such as their foraging exploits, this updates the character's knowledge of the world, indirectly influencing their actions during the next simulation phase. Direct suggestions from the player are registered with the NPC as high priority goals. Up to three such goals can be specified for each NPC, and remain in effect until the player tells the NPC to stop pursuing them, though other character goals, including goals determined via decomposition, will be pursued in parallel.

The player is not given a specific gameplay goal to pursue, though many possibilities are suggested through conversation with the characters, such as helping or hindering them in their professions and manipulating the relationships between the NPCs. At the start of the game, the three NPCs are in a love triangle to provide the player with a fraught social situation to work with.

System Description

The architecture can be split into two parts (Figure 2): the *world simulation* and the *conversation space* (set in the town's tavern). The actions taken by NPCs in the world simulation populate the content of NPCs' dialogue in the following conversation. Players can provide information or suggestions to NPCs in the tavern's, altering what actions NPCs take in the simulated world.

Action selection in the world simulation consists of three major subsystems: building the pool of possible actions available to each NPC (the *Action Builder*), selecting which

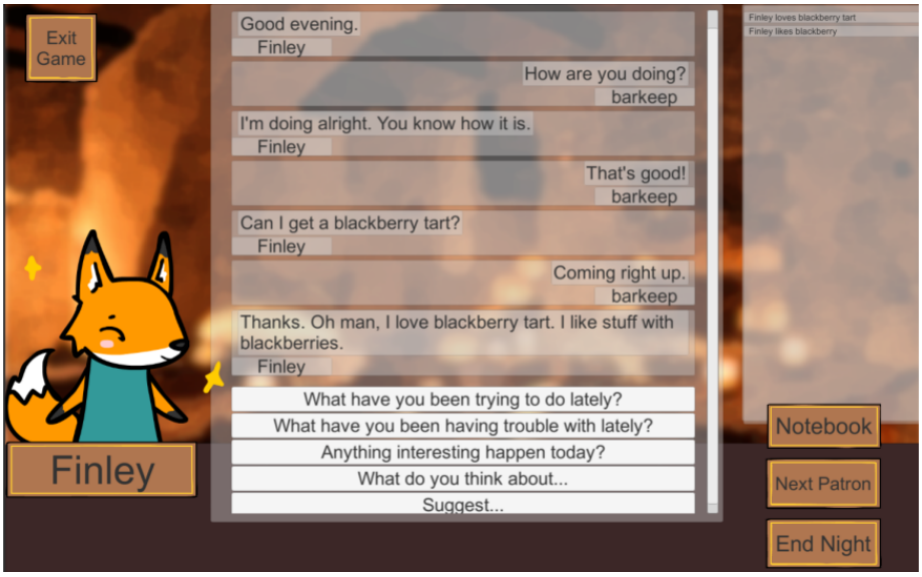

Figure 1: The player has a variety of things they can say to an NPC. Pictured are some of those that do not require any player knowledge.

of those possible actions to take (the *Action Heuristic Manager*), and executing and recording the results of the selected action (the *Action Execution Manager*).

The conversation space consists of two subsystems: a system for selecting the social moves available to the player under the current context and for choosing the next response for the NPC (the *Conversation Engine*) and the system for turning the system representation of this data into English text understandable by a human player (the *Social Move Verbalizer*).

Our approach to goal-based characters is richer than typically found in goal-based character architectures for games. Our architecture includes autonomous goal setting via the goal modules (in contrast to many goal-based characters which maintain a fixed hierarchy of goals), theory of mind, reasoning about probabilistic effects, parallel goals, and opinion formation based on outcomes. While all of these features can be found individually in different goal-based agent architectures, they are rarely integrated into a single game AI architecture to enable gameplay. We describe each of these systems in more detail below.

Goal Management

NPC goals change with their changing knowledge of the world context. Since this can change with every action taken, these goals must be regenerated before action selection at each time step in the simulation.

Goal modules are used to specify a prerequisite and a collection of goals to pursue if that prerequisite is true. In *Crosston Tavern*, each character has the same collection of goal modules. As an example, each character has a goal module that determines when a character would have the goal of dating another character. The prerequiste for this goal is: the actor feels an above threshold romantic attraction towards the target, the actor has the highest romantic attraction towards the target out of all the characters, and the actor and target are not currently dating.

This allows the NPC's goals to change to reflect their current opinions or understanding of world state. For example, the game beings with Sammy in love with Finley and so having the goal to date them. However, through player interaction, Avery may become the most romantically attractive character to Sammy, causing them to drop their goal of dating Finley and activate a goal of dating Avery.

In *Crosston Tavern*, the goal modules currently include:

- Dating (active as described above)
- Becoming best friends (active when the friendship relation is above a threshold)
- Becoming mortal enemies (active when the friendship relation is below a threshold)
- Improving profession skills (active if they have a particular profession)
- Eating food they like (active when they have that food in their possession).

These are all considered top-level goals and direct their general behavior. Top level goals can also be activated through the player's conversation with characters.

From these top-level goals, additional subgoals are created from the preconditions of actions or probability modifiers of outcomes that would help progress that goal. Action preconditions require certain world state values, while probability modifiers (which we discuss in more detail below) result in higher probabilities if those values are above certain thresholds. Example action preconditions and outcome modifiers that result in subgoal creation include inventory items, relationship values with characters, and the profession skill level.

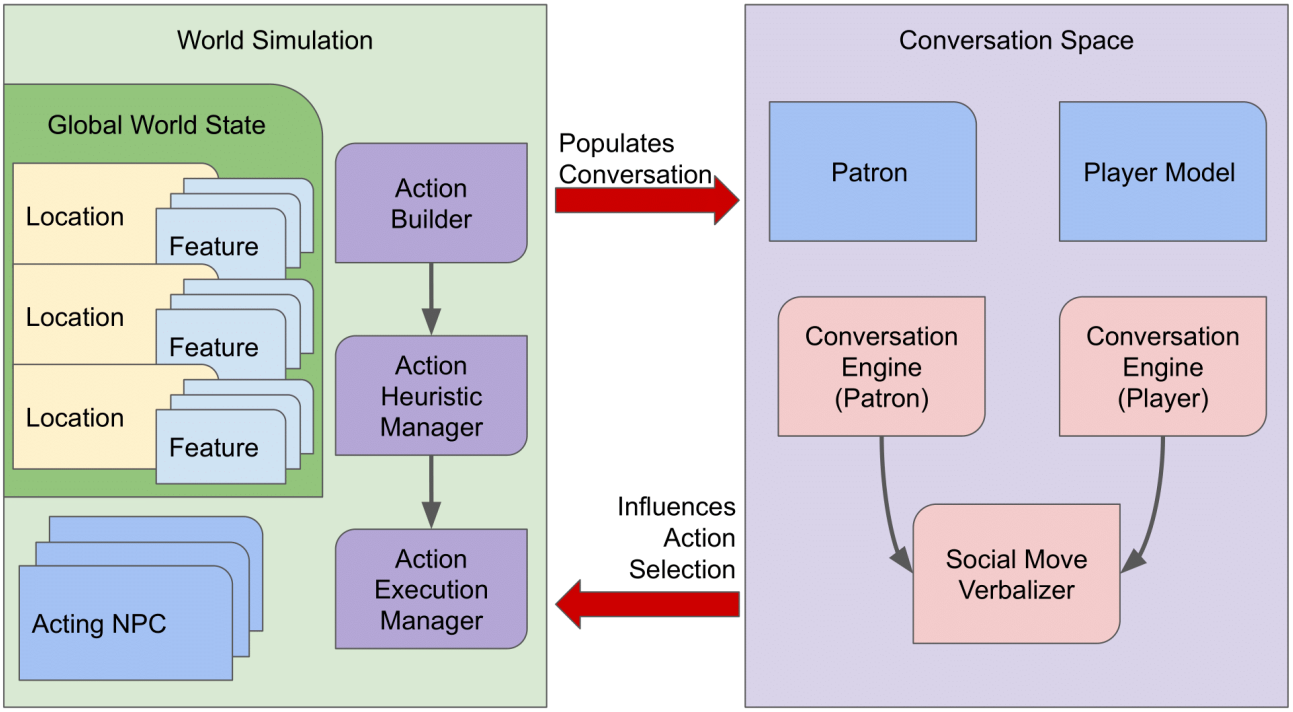

Figure 2: *Crosston Tavern* alternates between characters taking simulated action in the World Simulation during the day and talking with the player in the Conversation Space at night.

Character Actions

Similarly to Probabilistic PDDL (Younes 2003), actions in *Crosston Tavern* have probabilistic effects. Thus, unbound actions are represented by preconditions, a list of potential effects and their probabilities, and a list of variables that require binding. Once goals have been determined and action selection for a given simulation timestep has resulted in an executed action, the fully specified and executed action remembered in a character's memory includes:

- who took the action (the actor)
- what they acted on (the feature or character)
- where and when the action took place
- reasons for taking the action (weighed against goals)
- a list of rejected and invalid alternatives
- a list of realized effects (the actual results of the action).

These completed action representations are used to generate question and answers during the player's conversation with characters.

Actions with probabilistic outcomes may have static probabilities for each outcome or probabilities computed as a function of the bound state. For dynamic probabilities, the basic construct linearly interpolates between two thresholds from 0-100% for state variables that take numeric values, such as NPC relationships, inventory items, or skill levels. These individual state variable interpolations can be combined via multiplication, addition and inversion to create a wide range of dynamic probabilities for outcomes.

Below we describe the subsystems responsible for action selection and execution.

Action Builder. The Action Builder is responsible for creating the list of bound potential actions that a character could execute, including the actions offered by features in a character's current location. This includes binding potential action targets as well as objects involved in actions (e.g. gift items being transferred in a gifting action).

Action Heuristic Manager. The Action Heuristic Manager weighs how much the character wants to perform each of the bound actions against their goals and selects one for execution from among the highest weighted actions.

To determine action weightings, action effects are compared against goals. The desire for any given effect is calculated based on how well the effect advances the NPC's progress toward a goal multiplied by the weight of the given goal, summed over all the NPC's goals. The sum of an NPC's desire for each effect to occur is then multiplied by the chance that outcome will occur, then added to the values obtained by the other potential outcomes. Each pair of goal and effect that progresses that goal are recorded as part of the "reason" this action is desirable. This is used to answer questions during the conversation with the player. The action to execute is selected probabilistically among the top five weighted actions.

Action Execution Manager. Here the action selected for execution changes the simulated world state based on the outcome effects. In cases where the action has multiple probabilistic outcomes, an outcome is first selected based on the computed (or static) probabilities. Note that this set of potential outcomes may be different than what the character considered in the Action Heuristic Manager, as the character may not have yet learned about all the possible effects (e.g. the possible fish that can be caught or vegetables foraged at a particular fishing or foraging location). In addition to changing the true world state, the actor's knowledge of the world state and memory of actions that have occurred, as well as the knowledge of memory of any characters in the same location, are updated.

The actor also forms an opinion about how they feel about that action based largely on what they thought the results of the action were going to be versus what the actual results turned out to be. This is done by comparing the actual results against the probabilities determined by the Action Heuristic Manager to determine whether the outcome was disappointing or exciting. This opinion is used during conversation to filter what actions NPCs tell the player about when reporting their day.

Conversation Space

Crosston Tavern's play takes place in the form of conversations with NPCs. The player and the NPCs take turns taking actions, called *social moves* to differentiate them from the actions taken in the world simulation. Each social move consists of an abstract description of the move being made and, optionally, a list of references to world state (referred to as "facts") being conveyed by the social move. For example, when the NPC tells the player about their day they are taking the *tellAboutDayEvents* social move accompanied by a collection of discrete events from the most recent day. We shall discuss the structure of these social moves and the facts they may contain below.

Which move (or moves) is available, with which facts attached, is determined by the Conversation Engine. It has two primary responsibilities: picking the single move taken by the NPCs each turn and providing the list of social move options to the player. Then, once either chosen for the NPC to say or included as an option in the player's menu of dialogue choices, each move is then converted into readable English that the players can understand via the Conversation Verbalizer.

Facts included in social moves are learned by the listener, updating the listener's understanding of their simulated world. This process is symmetrical between the player and NPC, as both store their understanding of world state in an identical manner. Through this modeling of player knowledge, we can include among their dialogue options statements about facts the player has learned about. Through these statements, the player can subtly influence the actions of other NPCs. For more direct control, the player also has the ability to suggest new goals for the NPCs to pursue, allowing the player to push the world state in wildly varying directions.

Conversation Engine. Both the player and the NPC in a given conversation have a dedicated conversation engine that is responsible for selecting what social moves are appropriate in the given context. The primary difference between the two versions of this system is how many social moves they return when given input. The Barkeep Engine (the one which handles the player's side) generally returns many social moves for the player to choose from while the Patron Engine (the one that handles the NPC's side) always returns a singular social move for the NPC to say to the player. Additionally, which social moves are generally available to each side is different. Every social move has its own set of appropriate responses, usually generated from a generic social move and either the speaker's knowledge of world state or the facts attached to the prompt.

Patron Engine. The Patron Engine always returns a single social move in response to any prompt. For every question the player can ask and every statement the player can make, the Patron Engine can generate a response. In most cases, for a given abstract social move, there is an appropriate abstract of social move to use in response. These pairings were authored during the design of this system. For example, for any *askAboutGoals* social move, a patron should respond with a *tellAboutGoals* social move. Of course the details of the *tellAboutGoals* move will be determined dynamically from the NPC's knowledge.

There are cases, however, where there is more than one social move as a potential response. This occurs most frequently when potential responses were designed to be equally relevant to the prompt and can thus be selected randomly to provide variety to the NPC's responses. The response to *greet* is an example of this. In other cases, however, there are multiple responses because there could be multiple reactions to a player's statement or answers to a player's question. For example, when the player tells an NPC about the item preferences of another (*tellPreference* social move) the NPC can either respond that they already knew that fact or that this is new information for them as appropriate.

Barkeep Engine. The single biggest difference between the conversation engine for the player and the conversation engine for the patron is that the one for the player usually returns many social moves instead of just one. For example, in response to the NPC asking for a food recommendation (*askForRecomendation*), the barkeep engine will return a set of *recommend* social moves, one bound to each food item, to be presented in the player's dialog menu.

For many social moves the NPC may make, there are no specific barkeep potential responses. In this case, the Barkeep Engine responds with a list of player social moves corresponding to a "main dialogue menu." These include static social moves such as *askAboutGoals* or *askAboutDayHighlights* as well as social moves generated off of the player's knowledge base such as a *tellAction* move for each executed action the player has learned about.

Verbalization. To display both dialog menus and responses in readable English text, we generate natural lan-

guage from social move repesentations. This is done using hierarchical templates, a combination of context-free grammars such as Tracery (Compton, Kybartas, and Mateas 2015), with templates conditioned on world state. The hierarchical decomposition consists of three main levels: the top-level grammar patterns for social moves, intermediate level patterns for facts that are referenced in social moves, and lowest-level patterns to reference objects and low level world state. Intermediate level patterns have variations for handling past, present and future tense. Low level patterns handle generic, singular and plural forms, for example "stewed trout', "a bowl of stewed trout" and "many bowls of stewed trout'. Additionally, the pattern language includes general support for pronoun substitutions.

Interaction Between World Simulation and Conversation Space

The core of the gameplay loop is the player indirectly influencing the world simulation though conversation, with the world simulation then determining the options and answers in the next round of conversation. As an example of gameplay, the player may want to pair up Sammy and Avery romantically. Since Avery already likes Sammy, the player's focus should be on helping Avery get Sammy to like them more. One method to do this would be for Avery to give Sammy their favorite food: strawberry cake. Sammy is not at all shy about telling the barkeep that they love strawberry cake, so it quickly becomes an option to tell Avery. Once Avery knows that Sammy loves strawberry cake, their Goal Manager is able to unroll Avery's "I want to date Sammy" goal into the desire to "give Sammy a strawberry cake". To give a strawberry cake, though, Avery needs to first have a strawberry cake, opening up crafting and foraging subgoals that the player can help with.

The player can also suggest actions for the NPCs to perform. Players can use this for good or for ill, for example suggesting characters "ask out" others, either because the player knows they both secretly love each other and they just need that last push, or because they wish to sour relations between two characters that don't feel that way toward each other. Additionally, suggestions can be used to tell NPCs about actions they didn't know were available in the world simulation.

Evaluation

In evaluating *Crosston Tavern*, we had seven participants do initial playtesting on the game demo. Players were first asked about their experience with video games. This included what genres of games they typically play and if they had any experience specifically with Farming Sims, Animal Crossing, or the Sims. These were selected to be of particular interest as *Crosston Tavern* is meant to be an experience in the same larger genre of casual social game as these three.

Participants were then asked to play the game for at least twenty minutes but were allowed to continue for as long as they wanted up to fifty minutes. They were asked to think aloud as they played, and we took notes on their reactions. It was suggested that players try to manipulate the relationships between the NPCs.

After participants were finished, we asked a series of follow up questions:

- Did you enjoy yourself?
- Did you find what the NPCs had to say interesting?
- Did you feel you were able to manipulate their relationships?
- Was there anything you wanted to ask the NPCs or say to them that you were unable to?
- Were there any particular moments that stick out to you from your playthrough?
- Would you have liked to talk to more townies, or was it too much juggling three as it was?
- Would you have liked to hear about the actions of additional background characters who did not show up in the bar?

Discussion

Overall, the reception of Crosston Tavern was positive, with players saying they enjoyed playing it. The average playtime was about half an hour, with the longest playtime forcefully ended at fifty minutes. Two players stated they wanted to play it more after the playtest was complete.

Most playtesters reported finding the dialogue initially interesting but that they quickly found the way in which the NPCs talked repetitive. One playtester enjoyed the repetitiveness of the dialogue, saying it was "nonsensical" but that "it was natural to the setting". Of the things the NPCs said, most playtesters reported being more interested in dialogue about inter-character relationships and were less interested in statements about the physical state of the world or resource collection.

Frequently players tried to romantically pair up characters. A few players changed their minds about who they wanted to help get together over the course of play as they learned more about the character's interactions. For example, one player had initially wanted to pair Finley and Sammy, but after hearing Finley had turned Sammy down and then insulted them they switched their goals to supporting Sammy and Avery.

A smaller number of players wanted to see all three characters get along as friends (or at least actively did not want them to be enemies), although only one player said this was their primary goal.

One player, partway through play, decided to become "an agent of chaos" with the explicit goal of causing drama and driving wedges between the characters. This particular player expressed an almost guilty pleasure in purposefully suggesting actions to the NPCs which would sour their relationships, such as suggesting they insult one another. Frequently they said things such as, "Oh, god, I feel so bad. I am a terrible person" while grinning ear to ear at the chaos their most recent action had sown.

Another player expressed a similar sentiment. They described the feeling of playing *Crosston Tavern* that of being responsible for the lives of the NPCs. That "there is

guilt attached to messing up but also delight" and that they liked "the hubris in the responsibility because the animals do whatever you tell them to."

Unfortunately, not every playtester felt that their actions strongly affected the NPCs. Two of the seven playtesters reported feeling like they were unable to manipulate the world state, while another three described it as feeling like their ability to change things was moderate or would be more pronounced given a longer playtime. The two who felt most strongly that their actions affected the system were those who had (accidentally or intentionally) ruined a relationship between two characters.

Overall, players were reasonably satisfied with the range of things they could say to the NPCs. A few players wished that there were more dialogue options relating to character relationships, for example, they wanted to tell Sammy about Finley's opinion of Avery, while the current system only allowed the player to tell Sammy what Finley might think of Sammy or what Avery might think of Sammy.

When asked if there were any moments that stood out to players, most replied with an event centered on the NPC's emotions. Examples include Finley entering the bar frustrated three days in a row, Sammy crying as they told the player about being rejected in asking out Finley, and characters blushing after being told their crush likes them. Players also strongly remembered events which they felt responsible for or which they felt some guilt about, such as when they lied to one of the characters about how another NPC felt about them or when they heard their suggested action backfired.

When asked how players felt about the size of the cast, opinions were largely unanimous. All players agreed that three cast members were the minimum that sounded interesting. This makes sense as all players showed the most interest in inter-NPC relations and reducing the cast size from three to two greatly reduces the potential character dynamics. Some players suggested that perhaps starting with a smaller number of NPCs per night, then selecting three to visit from a larger cast as the nights progress might be interesting as well. Additionally, there was overwhelming interest in having a cast of background characters who do not appear in the bar but which the bar patrons talk about. This was part of the original plan but was cut because of time constraints.

Overall, most players were most interested in the social relationships between NPCs as well as events that triggered an emotional response in those NPCs while dialogue related to resource gathering was considered significantly less interesting or noteworthy. Interestingly, however, several players mentioned liking that NPCs had non-relationship goals, even though those goals largely had to do with resource-gathering actions. One player mentioned that "it was cute that the characters all had their own goals" and that it was "almost more interesting than the dating stuff".

Conclusion

Crosston Tavern is an exploration of an AI-based game design featuring conversational interaction with a character simulation. We have attempted to flip the standard roles of NPCs and players in video games, by having the player take a supporting role in helping NPCs accomplish their goals. NPCs autonomously pursue their personal goals within a simulated world and report on those actions and their understanding of the world during conversations with the player. The player uses this reporting to guide the NPCs in their efforts, either by sharing information or suggesting specific actions to take next.

Initial playtests have been positive. Players are excited by the potential for drama this system supports. They suggest there is much interest in exploring inter-NPC relationships through gameplay and delight in seeing emotional reactions from player action. We look forward to seeing this approach to player-NPC interaction or inter-NPC relationships used in future contexts and hope this is only the beginning for this line of research and design.

References

Amccus. 1996. Harvest Moon. SNES.

AQL, M. 2021. Story of Seasons: Pioneers of Olive Town. Nintendo Switch.

Compton, K.; Kybartas, B.; and Mateas, M. 2015. Tracery: an author-focused generative text tool. In *International Conference on Interactive Digital Storytelling*, 154–161. Springer.

ConcernedApe. 2016. Stardew Valley. https://www.stardewvalley.net/.

Eladhari, M. P. 2015. AI Based Game Design. Presented in GDC Education Summit.

Evans, R.; and Short, E. 2014. Versu—A Simulationist Storytelling System. *IEEE Transactions on Computational Intelligence and AI in Games* 6: 113–130.

Interactive, M. 2007. Harvest Moon: Tree of Tranquility. Wii.

Maxis. 2014. The Sims 4. https://www.ea.com/games/the-sims/the-sims-4.

McCoy, J.; Treanor, M.; Samuel, B.; Reed, A. A.; Mateas, M.; and Wardrip-Fruin, N. 2013. Prom Week: Designing past the game/story dilemma. In *FDG*.

McCoy, J.; Treanor, M.; Samuel, B.; Reed, A. A.; Mateas, M.; and Wardrip-Fruin, N. 2014. Social Story Worlds With Comme il Faut. *IEEE Transactions on Computational Intelligence and AI in Games* 6: 97–112.

Meehan, J. R. 1977. TALE-SPIN, An Interactive Program that Writes Stories. In *Ijcai*, volume 77, 9198.

Ryan, J. 2018. *Curating Simulated Storyworlds*. Ph.D. thesis, UC Santa Cruz.

Ryan, J.; Summerville, A.; Mateas, M.; and Wardrip-Fruin, N. 2015. Toward Characters Who Observe, Tell, Misremember, and Lie.

Samuel, B.; Reed, A. A.; Maddaloni, P.; Mateas, M.; and Wardrip-Fruin, N. 2015. The ensemble engine: Next-generation social physics. In *Proceedings of the Tenth International Conference on the Foundations of Digital Games (FDG 2015)*, 22–25.

Samuel, B.; Ryan, J.; Summerville, A. J.; Mateas, M.; and Wardrip-Fruin, N. 2016. Bad News: An experiment in computationally assisted performance. In *International Conference on Interactive Digital Storytelling*, 108–120. Springer.

Treanor, M.; Zook, A.; Palosaari Eladhari, M.; Togelius, J.; Smith, G.; Cook, M.; Thompson, T.; Magerko, B.; Levine, J.; and Smith, A. 2015. AI-Based Game Design Patterns.

Younes, H. L. 2003. Extending PDDL to model stochastic decision processes. In *Proceedings of the ICAPS-03 Workshop on PDDL*, 95–103. Citeseer.