1.

How Players Speak to an Intelligent Game Character Using Natural Language Messages

Fraser Allison, Ewa Luger, & Katja Hofmann

Transactions of the Digital Games Research Association

November 2018, Vol 4 No 2, pp 1-49

ISSN 2328-9422

***All authors were at Microsoft Research Cambridge, UK, during data collection stage.**

ABSTRACT

AI-driven characters that learn directly from human input are rare in digital games, but recent advances in several fields of machine learning suggests that they may soon be much more feasible to create. This study explores the design space for interacting with

such a character through natural language text dialogue. We conducted an observational study with 18 high school students, who played Minecraft alongside a Wizard of Oz prototype of a companion AI character that learned from their actions and inputs. In this paper, we report on an analysis of the 186 natural language messages that players sent to the character, and review key variations in syntax, function and writing style. We find that players' behaviour and language was differentiated by the extent to which they expressed an anthropomorphic view of the AI character and the level of interest that they showed in interacting with it.

Keywords

Natural language, AI, human-agent interaction, Wizard of Oz, Minecraft

INTRODUCTION

In recent years, advances in machine learning have driven rapid improvements in artificial intelligence (AI) techniques, both within and outside of videogames. Game worlds make effective testing grounds for learning AI due to their ability to simulate real-world challenges in incremental, constrained stages in a controllable and measurable environment ("Why AI researchers like video games" 2017). As a result, games have been the focus of a great deal of AI research involving machine learning (Yannakakis & Togelius 2015; 2017), from digitised board games (Silver et al. 2016) to real-time arcade games (Mnih et al. 2015; Shaker et al. 2013) and three-dimensional gameworlds that more closely approximate physical space (Johnson et al. 2016).

Despite this, game developers have been slow to take up machine learning techniques for in-game character AI.Games that do feature AI characters that learn from player inputs have been either notable for their novelty, such as *Black & White* and *Forza Motorsport*, or confined to academic projects, such as *NERO:*

NeuroEvolving Robotic Operatives (Stanley et al. 2005). Both the tools and the imagination needed to change this are surfacing. On the tools side, several companies including Unity (Juliani 2017) and Microsoft (Johnson et al. 2016) have released open-source platforms that facilitate the training of machine learning agents in game environments, and these platforms can be adapted for imitation learning (in which agents learn from actions demonstrated by a player or another agent) and reinforcement learning (in which agents learn from a reward signal provided by the environment, such as a game score). On the imagination side, the idea of AI characters that learn from players is becoming more prevalent in games. For example, the 2017 stealth-action game *Echo* is built around the design conceit that enemy characters learn and replicate the actions taken by the player (Robertson 2017). And in 2014, *Middle-Earth: Shadow of Mordor* featured a system in which individual enemy characters were permanently transformed and adapted by their encounters with the player (Taljonick 2014). However, these conceptual examples do not appear to be driven by machine learning technology in any real sense. Shaping the user experience for a game character that actively learns through interaction with a player remains challenging, as there are concerns that the interaction may be inconsistent, difficult for players to understand, or simply not fun (Muñoz-Avila et al. 2013; Yannakakis & Togelius 2015).

The present study is positioned as a pilot study designed to support the long-term aim of developing AIs that can learn to make sense of complex game environments (Johnson et al. 2016), especially in multi-agent settings that include AI and human players. Its purpose is to take a speculative look into the near future, and consider what the user interface considerations would be for an AI character that dynamically changes its behaviour based on what it learns from the player. As this character would have the potential to be less predictable and more adaptable than a character with static AI, an ideal interaction modality should allow for a wider variation of player inputs than a traditional gamepad. It should also be expressive enough to convey detailed state changes

in the AI back to the player, including ones that may not be obvious in the character's actions. Despite this increased scope for complexity, it should remain intuitive enough for the player to understand and formulate new input combinations without needing to spend a great deal of time learning how to do so. A natural language dialogue system is a common proposal for an interface that meets these criteria, as evidenced by the near ubiquity of spoken interaction among science fiction AIs, from the androids in *Westworld* to Holly the ship's computer in *Red Dwarf*.

Natural language interaction presents an intriguing mode for interacting with relatively independent and teachable AI agents. In principle, it could give players great flexibility to direct and interact with an AI character in an "off-script" fashion, without the need to learn and navigate an extensive graphical or physical user interface. And language understanding systems have improved rapidly in recent years, driven by the success of neural network models of machine learning.[1] Machine reading comprehension has approached and even exceeded human standard in some constrained scenarios (Eckersley et al. 2017), although open domain language understanding by computers remains far below human level. However, natural language interaction also introduces design challenges. For example, the conversational mode of dictating action to a computer is a departure from the widely accepted interaction paradigm of "direct manipulation" (Shneiderman 1982), in which the player's point of control is represented as a clear and direct link between the physical controls in their hands and a singular locus of manipulation in the computer system or gameworld (Bayliss 2007). This unfamiliarity and the unbounded nature of natural language can make it difficult to immediately formulate the "right" thing to say, and can require a process of learning how to adapt one's phrasing to the system

1. Many of the recent improvements in natural language processing have come from the use of neural network models, particularly recurrent or recursive neural networks (RNN), and specifically variations of long short-term memory (LSTM) and gated recurrent unit (GRU) RNN models. For a full explanation of the use of neural network models in natural language processing, see Goldberg (2016).

(Luger and Sellen 2016, 5289). On the developer's side, the openness of natural language can make it difficult to anticipate what kind of syntax and concepts players will use.

To contribute towards addressing these challenges, we conducted a study of players interacting with an autonomous, learning-capable game character using natural language. This study employed the Wizard of Oz research method, which involves presenting participants with a convincing replica of an automated system in which some of the functions are secretly operated by a human (Kelley 1983). We designed a conceptual prototype of a plausible near-future AI agent that learns from player actions and uses natural language text messages to communicate, which we named help_bot. We invited participants to perform tasks in the videogame, Minecraft, with the assistance of help_bot, and provided intentionally minimal instructions for how to do so, so that we could observe the ways that players spontaneously attempted to engage and speak with an agent of this type. We conducted a content analysis on the natural language messages that players sent to help_bot, to study what type of syntax they used, what kind of commands they gave and how the use of language as opposed to traditional game controls created opportunities and problems for the interaction.

Given the rapid progress in machine learning-based natural language processing, it is our belief that these techniques have the potential to dramatically change in-game interaction using natural language. However, it is extremely unlikely that the available technology will immediately jump to human-level language understanding; much more partial and constrained language interactions appear likely in the near term. Therefore, potential usage scenarios and player perceptions and behaviours need to be thoroughly understood, to allow game developers to craft language-based interactions that suit players' expectations and desires within the constraints of the available technology. We see the key contribution of this work in mapping this space of player

perceptions and behaviours, paving the way towards the required understanding and development of novel designs.

PRIOR WORK

In the past decade, there has been a steady increase of research interest in AI applications in digital games (Yannakakis & Togelius 2017, 19-20). Much of this work has focused on training game-playing agents using reinforcement learning (RL), in which an agent is taught to associate combinations of actions and environmental conditions with a reward signal (such as the character's health or a score counter), and learns through trial and error to maximise the reward signal through its choice of actions (Sutton and Barto 1998). Famously, Google DeepMind used a combination of RL and supervised learning from expert human moves to train a Go-playing program (Silver et al. 2016) that beat one of the world's best human Go players, Lee Sedol. RL research is also being conducted to train agents in real-time digital games, from older Atari 2600 games (Bellemare et al. 2012) to more recent games with complex spatial environments such as Doom (Kempka et al. 2016), Starcraft (Farooq et al. 2016) and Minecraft (Johnson et al. 2016). A related branch of research has looked at developing agents that learn from player actions, either from pre-recorded play data or through direct interactions with players. The goal of this work can be to learn higher-level performance strategies, create more convincingly human-like game characters or adapt to individual players' preferences and playstyles (Bakkes et al. 2012).

Studies in interactive machine learning look at scenarios in which a human actively provides feedback to a learning agent to update its behaviour. Researchers in this area have consistently found that users exhibit strong preferences for teaching styles that do not always align with the learning model of the agent (Amershi et al. 2014). Whereas RL-based agents are often designed to learn from explicit feedback on their recent actions, human teachers

give relatively little explicit feedback, and instead focus on communicating the desired behaviour conceptually through demonstrations and positive prompts (Amershi et al. 2014; Kaochar et al. 2011; Knox et al. 2012). When required to give repetitive and simplistic input, users often experience impatience and frustration, and a resulting decline in their performance as teachers (Cakmak et al. 2010; Guillory and Bilmes 2011). In a study by Fischer et al. (2013), human users were better at adapting their teaching behaviour for a learning robot when the robot's feedback mimicked the human's social behaviour (in the form of gaze), which indicates that their mental models of how the robot was learning and attending to things were influenced by their knowledge of human learning and attention. Similarly, a study by Koenig et al. concluded that human users' failures to adapt their teaching behaviour effectively based on feedback from a robot learner resulted from a "tendency to map a human-like model onto the capabilities of the robot" (2010, 1111). A review of interactive machine learning studies by Amershi et al. (2014) concluded that a wide range of interactions are possible for human teaching of agents, but studying the human users of these systems will be critical to ensuring their success.

A common barrier to studying user behaviour with both intelligent agents and natural language interfaces is the difficulty in implementing such systems to a high level of reliability. When the research question is not how users respond to the current state of the art but how they would respond to a hypothetical version of the technology, implementing the technology can be prohibitively difficult or expensive. To circumvent this, researchers in human-computer interaction often implement Wizard of Oz prototypes instead. In the Wizard of Oz method, an interface is presented to the user as being fully automated, but is operated out of sight by a human facilitator without the user's knowledge (Maulsby et al. 1993). This approach, first developed for studying user responses to natural language interfaces (Kelley 1983), is also commonly used for studying user interactions with intelligent agent systems (Goodrich and Schultz 2008; Riek 2012). Bernotat et al. (2012)

used a Wizard of Oz design to test how people responded to a futuristic "smart home" without specific instructions, and found that most users defaulted to speech control, demonstrating that language-based interaction is associated with intelligent systems in the public imagination. In another Wizard of Oz study, Xu et al. determined that users could recognise an unsignalled change in the behavioural pattern of an agent, and adapt their own behaviour to suit. These studies demonstrate that the Wizard of Oz approach is well suited to an exploration of how users interact with intelligent agents, particularly in language-based interactions.

A great deal of research has been conducted on natural language interfaces, but for the purposes of this study we are primarily interested in studies of user behaviour in natural language interactions with embodied virtual characters. Most prominent in this field is the work of Cassell, who formulated the concept and early prototypes of the "embodied conversational agent" (2000). Cassell focused on the role of non-verbal behaviours in sustaining the experience of human-like conversation, and the ways in which these factors make conversation fundamentally multimodal. Mateas and Stern (2005) incorporated expressive and affective embodied conversational agents into an interactive drama game, Façade, which was built around natural language interactions; Sali et al. (2010) compared this version of Façade with alternative versions wherein the player selected dialogue responses from a menu rather than typing their own, and found that although the natural language interface generated frustrating errors and reduced players' feeling of control, it was still the most preferred modality as it provided the greatest sense of presence and engagement. More recently, Lessard has produced several natural language interaction games designed around "conversational puzzles" (2016, 6), making conversation itself more of a game mechanic than a pseudo-social interaction. Lessard concludes that the natural language interaction in his games is easy for players to understand and to start playing with, but that the highly scripted nature of current game dialogue systems restricts the ability of games to take

advantage of more emergent gameplay possibilities that would theoretically be possible with natural language.

WIZARD OF OZ CHARACTER DESIGN

Following a review of literature on AI research in games and other fields of application, we extrapolated a set of abilities that we thought represented a reasonable approximation of what an autonomous agent in a game-world such as Minecraft could be made capable of within a few years' time. We named this hypothetical agent help_bot, and defined its abilities in a manner that we could represent through a human-controlled character.

Help_bot could "see" the same visual input as a player, and use this vision to understand and navigate unfamiliar terrain. It could recognise simple objects by sight within the game-world, including objects that were defined items in that game (such as the block types in Minecraft) as well as geometric shapes and patterns formed from the arrangement of objects in the game. It could add new objects to its recognised list through being given labelled examples (such as learning to associate a new shape with the label "pyramid"). It could learn and imitate behavioural patterns by watching the actions of a player-controlled avatar, and update its behaviour based on positive and negative feedback from the player, as well as behavioural prompts such as being hit or being given a particular tool or material. It had a limited ability to infer a higher-level goal from a player's actions, such as predicting what larger shape a player might be constructing from the initial placement of a few blocks.

Notably, help_bot was designed to be a "friendly" or companion character, in contrast to the majority of AI-controlled characters in digital games who take an "enemy" or oppositional role. Help_bot's behaviour followed a simple loop. Its starting state was to follow the player's avatar from a short distance and observe what they did. Periodically, it would categorise the player's current action (e.g. building with bricks) and infer a short-term goal of

that action (e.g. building a straight wall out of bricks). Help_bot would then attempt to assist in that task by continuing the action, such as by adding more bricks onto the wall to extend it in the same direction. At irregular intervals, or when prompted by direct interaction by the player, help_bot would reassess what the player was doing and either continue its current action or choose a new action accordingly.

The player could override this behaviour by sending help_bot messages through Minecraft's built-in chat channel. Help_bot understood natural language input through this channel, within constraints. It could distinguish between commands, questions, statements and acknowledgements and choose an appropriate response. It looked for verbs in a message that matched an action in its behavioural repertoire, such as "build", "follow" and "attack", and it looked for a grammatical subject and object to determine what the verb referred to. In this way, long or fragmented sentences could have their meaning inferred from key elements without fully understanding every word, but more nuanced or obscure meanings would not be understood. Messages that were understood prompted standard responses from help_bot: <ok> for commands, <done> after the command was completed, and <yes> or <no> in response to questions. Messages that were not fully understood prompted a request for clarification: <show me where>, <show me how> or <?> (see *Responses to prompts for additional information*).

In accordance with the Wizard of Oz research protocol, help_bot was secretly controlled during the study by a researcher in a separate room. As described above, the Wizard of Oz approach is often used for research on intelligent agent and natural language system prototypes (Goodrich and Schultz 2008; Riek 2012). Indeed, the method was first developed for studying the responses of "computer-naïve, first-time users" (Kelley 1983, 193) to a natural language application. One of the considerations of this method is that the deception should not be too obvious, which goes hand-in-hand with ensuring that the prototype is not unrealistically

high-performing. In our case, the researcher controlling help_bot was instructed not to make its behaviour appear too intelligent or natural. They controlled its movements entirely through a keyboard, rather than a mouse and keyboard, to reduce its fluidity of movement. Occasionally, the researcher made help_bot make deliberate "errors" by choosing actions that were plausible but against the player's apparent intentions, such as building over an open space that the player had created. This was to reinforce the impression that help_bot was computer-controlled, and to allow us to observe how players attempted to correct unwanted behaviour. To further support the impression of being an AI-controlled character, help_bot was given a robotic appearance.

METHODOLOGY

We recruited students from two high schools in the United Kingdom to participate in an observational user study. Excluding two participants who dropped out, we had 18 participants (11 female, 7 male) aged between 11 and 15 complete the study. As a rough indicator of sufficiency, this is equal to the mean sample size for in-person user studies presented at the ACM Conference on Human Factors in Computing Systems in 2014 (Caine 2016, 986). Parental consent was obtained for all participants, and parents were given the option to be nearby and observe the study. All participants were required to have played Minecraft before, and their level of experience varied from a few hours to over a hundred hours of play.

The study was conducted across two weeks, with each participant in a separate session. Each session lasted approximately 90 minutes. After greeting the participant and their parent and explaining the study, the facilitator showed the participant to a private room with a computer running Minecraft and a video camera. Participants were asked whether they would like to opt out of having their image appear in any publications about the study, which two did. The facilitator then interviewed the participant

briefly about their experience with Minecraft: how often they played, which game modes and activities they preferred, and whether they played in single-player or multiplayer mode. Each participant was set three building tasks to complete in sequence, in a pre-saved Minecraft gameworld created for the purpose.

The first task was to build a model boat, without assistance. This was a warm-up task, which gave the player a chance to get used to the game controls if they needed to, and to become accustomed to the study environment. It also allowed the researchers to observe how the player behaved in "normal" solo Minecraft play, to understand their habits and strategies so that they could be contrasted with how they played in the subsequent tasks. Five minutes was allowed for this task.

After the first task, the facilitator explained that they would be introducing an AI assistant character named "help_bot" into the world for the following tasks. Help_bot was described as an experimental prototype developed by the researchers, which could learn how to act in Minecraft by watching and interacting with players. Players were told that help_bot would try to assist them in their next building task, and that they could teach it or show it what to do if they wanted.

In either the second or third task, the player was also told that help_bot could understand messages that were sent through Minecraft's chat function. If the text interaction was introduced in the second task, for the third task the player was told that the text interaction was disabled. The order of the language and non-language interaction conditions was rotated, ensuring a balanced allocation of age, gender and previous Minecraft experience for participants.

The facilitator deliberately avoided giving specific instructions or examples of how to interact with help_bot. Before the non-language-input condition, the player was told that help_bot would learn from what it saw them do; that it would try to help them

with whatever they were doing; and that they could teach it things or show it what to do. Before the language-input condition, the player was again told that help_bot would learn from what it saw them do, and also what they wrote in the chat channel; that it could understand normal sentences; and that they could try to tell it what to do, teach it things, or give it feedback on what it had done. The instructions did not specify that players were required to interact with help_bot, and once the task had begun the facilitator did not direct the player further except to answer questions. Players who ignored help_bot or lost interest in their building task and moved on to other activities were allowed to do so.

Both the second and third task lasted 15 minutes (although players were allowed to go overtime by up to five minutes to finish what they wanted to do in the game). The instruction for the second task was to build a house, and the instruction for the third task was to add to it with a construction of their choice. After the first eight participants this task was changed, as most participants were familiar with the task of building a house from previous Minecraft experience, and so created it too quickly and often with little planning required. For the remaining participants, the instruction was to build a maze.

After each task, the facilitator conducted a short semi-structured interview with the player, prompting them to explain what they had been thinking at various moments. Players were also asked what they thought of help_bot; what strategies they used when they wanted help_bot to change its behaviour; how well they felt help_bot understood what they wanted, and what made them think so; how playing with help_bot compared to playing with another human; and what features they would change or add. After the final interview, the Wizard of Oz research approach was explained and participants were informed that help_bot had been a human-controlled character. Prior to this debriefing, no participant indicated a suspicion that help_bot may not be computer-controlled.

Content analysis of message logs

Across the 18 sessions of the study, players sent a total of 186 messages through the chat channel. These messages, along with the responses from help_bot, were saved in a log file. We conducted a content analysis on these messages to study the language players had used. These messages were analysed and coded by the first author in an iterative open coding process. Each message was coded according to seven main categories:

- Message syntax (e.g. interrogative)
- Message function (e.g. query)
- Message subject (e.g. “I”, “you”)
- Message direct object (e.g. “me”, “this”)
- Amount indicator (e.g. “a”, “some”)
- Location indicator (e.g. “here”, “back”)
- Repair process (e.g. reformulation)

Our analysis approach was informed by conversation analysis (Sacks et al. 1974), as the high-level goals of this study are to some extent aligned with the goals of conversation analysis. We focus on understanding the structures and patterns that can be discerned among pairs or longer sequences of “utterances” (to use the conversation analysis term), and how the structure of messages relates to the ways in which they are employed to elicit specific actions, rather than pure exchanges of information. We also look at players’ strategies for repairing failures of communication through their messages (Schegloff et al. 1977). However, the full method of conversation analysis is not suitable for this context, as it is substantially concerned with the mutual organisation of dialogue, ordinarily in the form of verbal speech. In this study, by contrast, the organisation of dialogue was one-sided, with the player initiating and directing nearly all of the conversation under the expectation that help_bot would act as a passive responder.

Accordingly, we draw on conversation analysis conceptually in the definition of codes, but pragmatically take a content analysis approach that is more tailored to the log transcript data available.

In analysing the messages as action-oriented inputs, we consider each one as a "speech act" (Searle 1969), intended by the player to serve a functional purpose. We note that this purpose can be different to the literal meaning of the sentence, and so a complete analysis must determine the true function of a message from contextual information. We draw on both the in-game context and players' interview comments to infer the function of each message, and we identify indirect speech acts (Searle 1975) in which the literal and functional meaning of the message disagree.

We do not intend through this analysis to lay down firm or fixed rules of conversational procedure for natural language-based interaction. However, as Button and Sharrock (1995) argue, we believe that examining the function and form of players' spontaneous natural language messages will provide useful guidance for natural language interactions with game characters, by pointing out what naïve players might want or expect such a character to be able to understand and respond to appropriately.

RESULTS

The focus of this paper is on the natural language interactions between players and help_bot. However, these textual inputs were highly multimodal with the players' actions using the traditional game controls, so some discussion of these actions is also required. To capture this, we use the following notation when describing interactions from the study: text inputs in angle brackets, verbal comments in quotation marks, and physical or in-game actions in italics. For example:

Player:<come back and give be the wood>

Help_bot:<?>

Player: “Question mark. So do I have to do every command in one line? Because I did two commands there.”

Player reads their previous message again.

Player: “Oh, because I wrote ‘give be the wood’.”

The intended meaning of a text command was often dependent on the virtual space in which the player’s avatar and help_bot were standing. Some commands directly referred to a visible object, as in <kill the zombie>, while others carried an implicit expectation that they would be carried out in the nearest relevant location, as in <can you get some wood please>. A few commands were paired explicitly with the player’s actions, as in <copy me>, or with help_bot’s actions, as in the following exchange:

Player:<break blocks>

Help_bot mines blocks directly in front of it, leaving other blocks above.

Help_bot:<done>

Player:<there is some at the top>

Help_bot:<?>

Player:<look up>

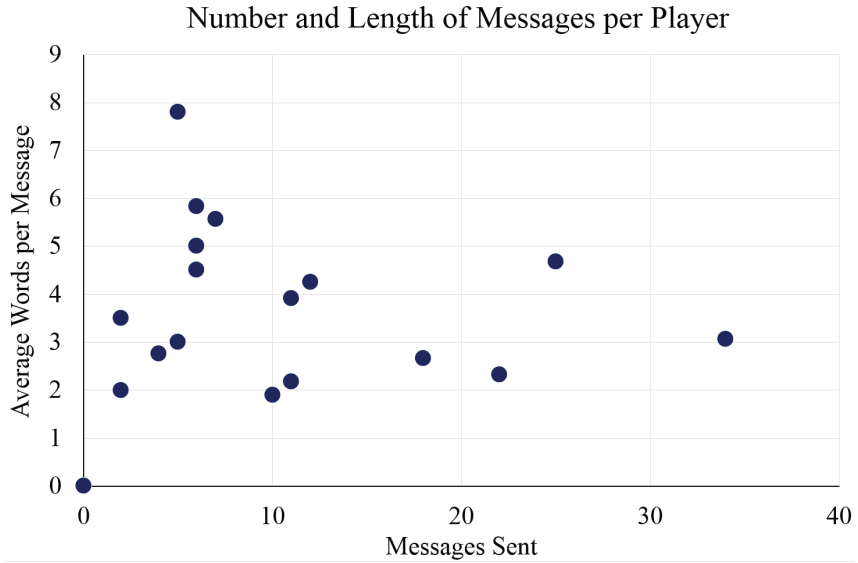

Figure 1: Scatter plot of players in the study by the number of messages they sent (horizontal) and the average length of those messages in words (vertical).

The extent to which players engaged with help_bot through the natural language system varied considerably (see figure 1). Several players sent less than five messages in total, including one who sent none at all. Conversely, several players sent help_bot more than one message per minute across the 15-minute task, with 34 being the highest.A few players wrote full sentences of up to 11 words (see figure 2), including polite phrasings and compound sentences, but the majority wrote primarily in terse phrases that consisted of no more than three words. There was no strong correlation between the length of a player's messages and the number of messages they sent.

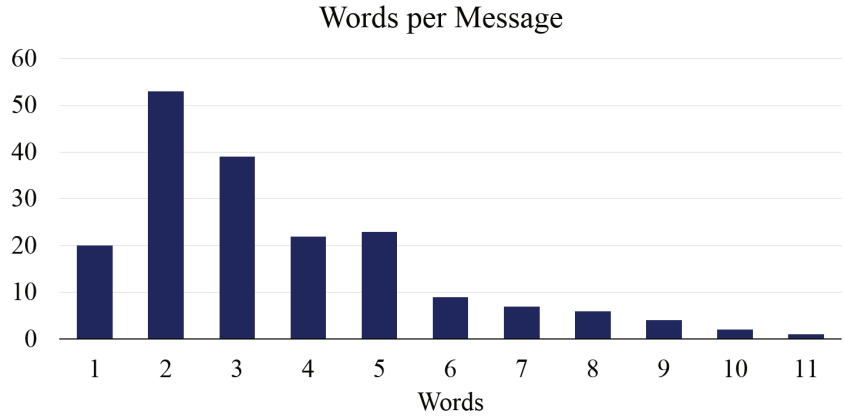

Figure 2:Distribution of messages by the number of words per message.

In the post-task interviews, some players said that they had kept their messages short to be sure that they would be understood. Although they had been told that help_bot could understand normal writing, players pre-emptively avoided longer or more complex sentences on the assumption that they would not be understood, as one player explained:

> "It's quite hard to get the wording correctly to get it to do stuff. Because like when I told it to bring in the material I was going to say 'collect this and give it to me' – I wasn't sure if it would understand that. So you have to be quite simple."

At the same time, however, players were concerned that their messages could be lacking in necessary detail, and that help_bot might make incorrect assumptions due to a lack of specificity in their instructions:

> "[I needed] to be quite simple and not overcomplicate it, so it would understand. [. . .] Probably if I were to say 'collect wood', it might have got *any* wood, so you have to be quite specific with that, what type of wood."

As a result of these conflicting tensions, players often hesitated over the wording of their messages and expressed uncertainty about how to engage help_bot in more complex tasks. A few

players suggested that a customisable menu of commands might be useful for defining some more complicated requests. We observed that the moments of uncertainty often came after a player had begun to type a request, but was unsure how to finish it; a contextual autocompletion function could provide timely assistance.

The exact wording used in messages was diverse. Of the 186 messages sent throughout the study, 128 were unique in their wording. A further 11 messages were sent twice by the same player. Only 12 messages were sent by multiple players, including four that were sent by three different players: <thank you>, <follow me>, <come back> and <build a house>. The latter was influenced by the construction task we assigned the first eight players, which was to build a house. Only two messages longer than three words were used by multiple players: <bring me oak wood> and <give me the wood>.

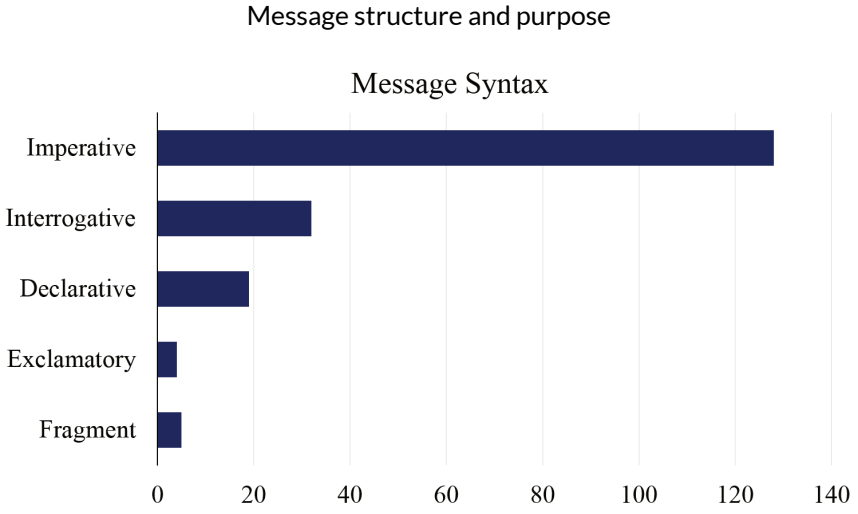

Figure 3:Distribution of messages by sentence type.

Most messages were structured as *imperatives* (see figure 3). These were usually short phrases such as <get stone> and <come here>. Some imperatives were longer, including a few compound

sentences containing multiple commands, such as <come to me and give me the oak wood>. Every imperative message was either directions to undertake an activity (*command*) or directions to cancel the current activity (*stop*). We show the relative proportion of message functions in figure 4.

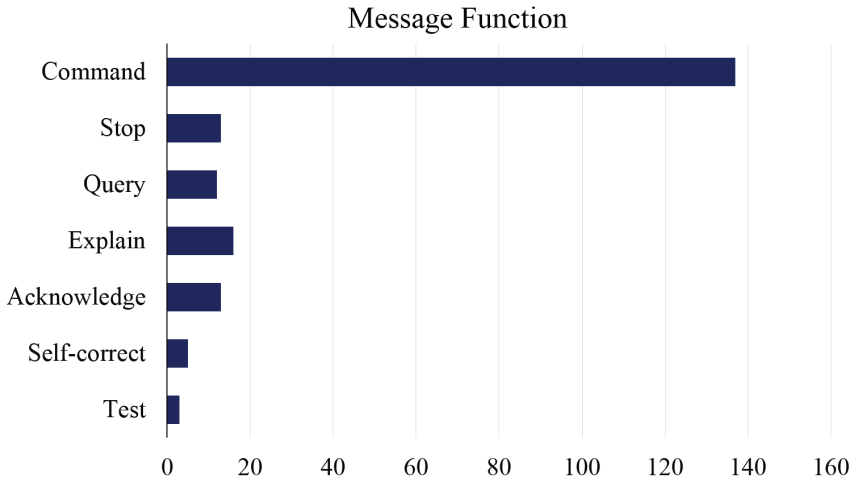

Figure 4:Distribution of messages by intended purpose, inferred from game context and player comments.

A large majority of messages omitted a grammatical subject (see figure 5). This was because most messages were imperatives, and standard English grammars omits the subject in an imperative phrase – the implied subject is the receiver of the message. One message specified its subject by naming help_bot (<help bot come here>). Six messages lacked an explicit subject, because the intended subject was the same as the previous message; we label these as “Antecedent (implied)”.

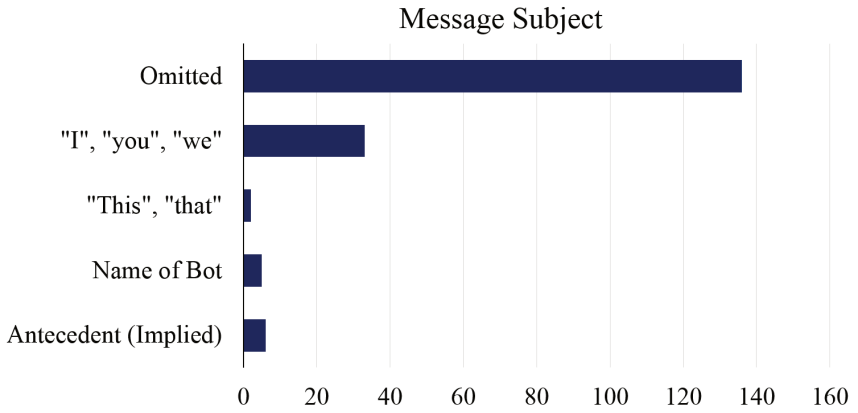

Figure 5:Distribution of messages by intended purpose, inferred from game context and player comments.

The direct object of most messages was indefinite rather than specific, as in <get stone> or <get some stone> as opposed to <get the stone> or <get that stone>. Where the direct object referred to “me” or “you”, this was usually a learning command, as in <copy me>, or an acknowledgement, as in <thank you>. The frequency of each type of direct object is shown in figure 6.

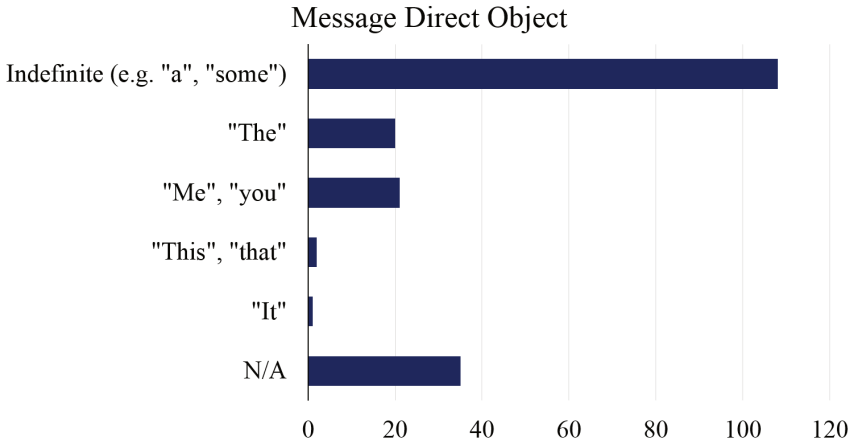

Figure 6:Distribution of messages by intended purpose, inferred from game context and player comments.

Most command imperatives were conceptually simple, requesting actions that had only one step (as in <come to me>) or two steps (as in <bring me oak wood>, requiring help_bot to collect oak and bring it to the player). Players also tested more conceptually complex requests, such as <build a house>. In response to these, help_bot would prompt the player to <show me how>. At this point, players typically simplified or abandoned the request, although some proceeded to demonstrate or explain the task (see *Responses to prompts for additional information*).

Stop imperatives, such as <stop> and <don't mine that>, were typically used when help_bot had completed a task to the player's satisfaction, or when it had made a categorical error. By this, we mean an action that was different in kind from what the player intended, such as when this player tried to ask help_bot to collect fruit:

Player:<get food>

Help_bot:<ok>

Player: "Oh, it's about to go and get food! Yay! I really hoped that would work."

Help_bot moves towards a cluster of sheep and cows.

Player:"Don't kill everything please. I'm going to follow it and make sure it doesn't kill everything. Oh – I'm not going to watch, because I feel like it's going to kill everything. . . Maybe I should tell it don't kill everything."

Player: <don't kill everything>

Help_bot:<?>

Player: "You don't. . ." *[nervous laugh]* "OK then, I'm just not going to watch and pretend that help_bot isn't slaughtering animals behind me."

Player: <stop>

Players did not use stop commands when help_bot made smaller-scale mistakes, such as placing blocks in the wrong location or digging a hole too deeply. In these cases, players often verbalised their frustration, but in the game they simply reversed help_bot's actions using their own avatar. In the post-task interview, players said they wanted help_bot to interpret either a <stop> command or a reversal of its recent actions by the player as implicit negative feedback, so that it would be less likely to take those actions in the future. That is, to update its behavioural algorithm. However, there were concerns that textual feedback may be too ambiguous, so help_bot might unlearn the wrong behaviour.

Interrogative or questioning phrasing was the second-most common message structure. Only two-fifths of these messages were punctuated with a question mark, as would be grammatically expected. We inferred that less than half of the interrogatives were truly intended as questions (e.g. "do you have any wood?"), and the remainder were indirect commands (e.g. "can you get me some coal please"). Interestingly, the use of a question mark was a strong indicator of a genuine question: three-quarters of the queries, but only one-fifth of the interrogative commands ended with a question mark, as figure 7 shows. One participant surmised that a question mark might be required for a message to be understood as a question:

Player:<do you have any wood?>

Help_bot:<yes>

Player: <how much>

Player: "I probably should have done a question mark."

Help_bot:<?>

Player: <how much wood do you have?>

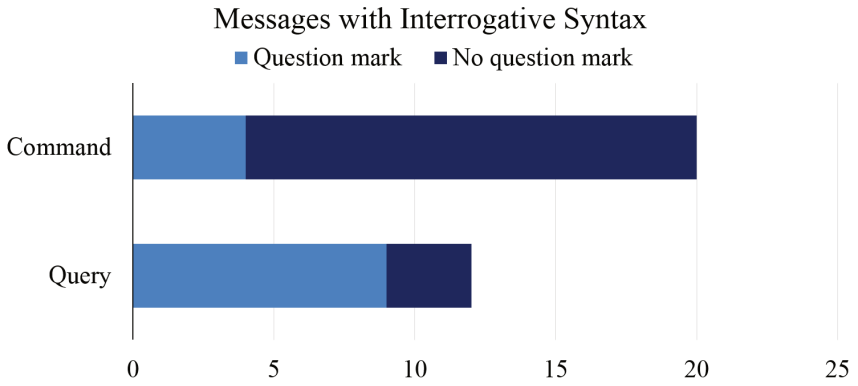

Figure 7:Breakdown of messages with interrogative syntax. Most of those intended as queries ended with a question mark, whereas most of those intended as commands did not.

Interrogative queries were used to learn about help_bot itself. Questions referred to either the contents of its inventory (<do you have any wood?>), the types of actions it was capable of (<what can you do?>), and its status (<are you lost?>). Players used these questions to gain information about help_bot that was not available through other means.

The third-most common message type was *declarative*, or straightforward statements of fact. The primary uses of declarative messages were to identify an object (<this is a shelter>) or to acknowledge help_bot's actions (<thank you> or <well done>). In a few cases, declarative statements were paired with learning commands to teach help_bot a behaviour that it could later use. For example, one participant instructed help_bot to <watch> as they built a simple hut shape, typed <this is a shelter>, then typed <build a shelter>. Other participants described more elaborate versions of this teaching approach as a way of automating repetitive work:

"One of my friends, every world he makes he always has this thing where he has [. . .] a boat with a pig in it and it always just spins around eternally. He could name it 'pigspin' or something like that,

and every world he goes in he could give [help_bot] the things it needs and go ‘build pigspin’.”

Two players used declarative messages as indirect stop commands for help_bot (<you don’t need to make any more planks> and <that’s fine>).

In the post-task interviews, players mentioned that they thanked or complimented help_bot as a form of positive feedback, with an idea that this might reinforce its learning of the recent behaviour:

“It’s about being polite, and also saying ‘thanks, you did the *right thing*’.”

Four messages were *exclamatory* in format, of which three were greetings (<hello> and <hi>) and one was a celebratory statement (<yay we finished the maze!>). We classified the purpose of all four as acknowledgements to help_bot, with the latter also being an explanatory message intended to reinforce the idea that what the player and help_bot had just built was a <maze>.

Finally, five messages were incomplete fragments. Two of these were self-corrections by the player (<bring me the wod> followed by <wood>), intended to update the meaning of the preceding message. One was similarly an addition to a prior message, adding <two blocks high> after <build wall>. The remaining two messages were nonsensical (<jeff> and <s>), and at least one of these was deliberately so. The player explained in the post-task interview that they had entered a nonsensical message to test help_bot’s responses:

“I was trying to see – because I [previously] did ‘thank you’ and it said ‘ok’, so I wasn’t sure if that was a response to ‘thank you’ or if it was just its generic response if it doesn’t understand something. And I did some random stuff and [learned that] the generic response if it doesn’t understand something is [a] question mark.”

Indeterminacy of amounts, places and boundaries

Specificity was an issue for many participants. Players were forced to confront the fact that their everyday phrasing contains a great deal of ambiguity, which is resolved by human conversation partners through context and common sense. The Minecraft setting facilitated references to objects in the world, as it contains only a limited set of clearly labelled and categorised objects. However, players were uncertain how to specify locations and amounts in a way that help_bot would understand. Players also expressed uncertainty about help_bot's understanding of boundaries.

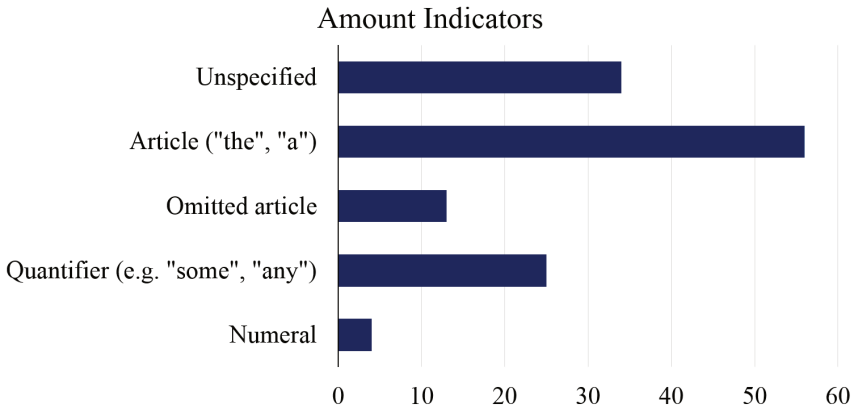

Figure 8:Breakdown of messages by how they indicated an amount for their direct object.

Amounts were rarely specified with a numeral; only one player did so (see figure 8). Most references to multiples of an object left the amount unmentioned, as in <collect bricks>. The rest of the time a linguistic quantifier was used, as in <get some wood> and <can you get me more inc sacks please>. Players would then leave help_bot to collect the objects until they felt it had enough, and stop or call it back at that point. Single objects were less difficult, as players included the relevant article ("a" or "the"), depending on whether the required object was generic (as in <build a box>) or specific (as in <bring me the coal ore>). In some cases, the article

was implied but omitted, as in <craft sign>, which could increase the difficulty for a natural language system to parse correctly.

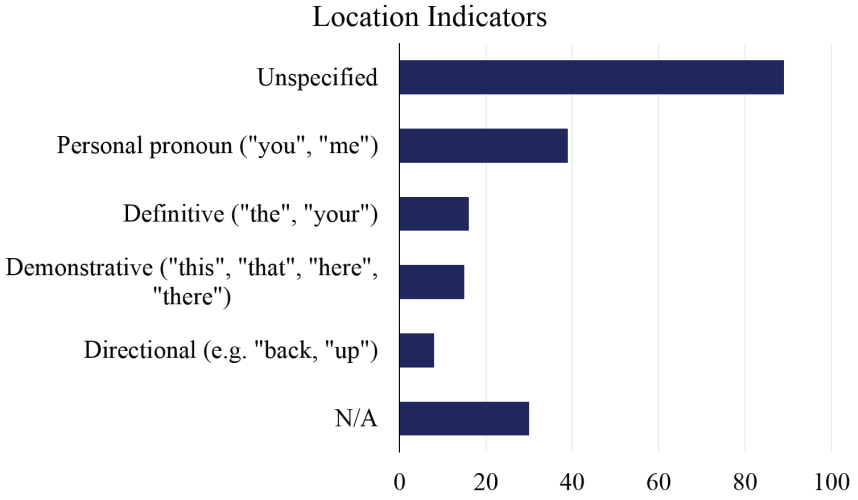

Figure 9:Breakdown of messages by how they indicated a location for the action of their message.

Locating the object of the message introduced further difficulties. In most cases, no location was specified, but there was an implicit deixis to many of these messages (see figure 9). For a typical fetch request, such as <get wood>, the player wanted help_bot to take the item from a specific location (usually the nearest source, unless that source was something the player had built) and bring it back to the player. Similarly, a typical build request such as <build a box> contained an unstated assumption that help_bot would place the building on a suitable flat piece of ground near to the player, but not close enough to interfere with their current activity.

“It built it quite close to where I built it. Like when I did the three-by-three square, it built right next to it. With this it didn’t really matter, but I think if you could tell it where to build that would be quite good. . . Maybe like coordinates or something, I don’t know.”

Some players attempted to be more specific about the location for their requests by using demonstrative terms such as “this” and

"here" as linguistic pointers. These were paired with the player's avatar moving to the relevant location, or even tapping on a specific spot with the avatar's hand, as in this exchange:

Player: <get me some spruce wood>

Help_bot: <ok>

Help_bot starts mining a house that the player had built out of spruce blocks.

Player: "So it is mining what I've done earlier, which is really weird."

Player: <don't mine that>

Help_bot stops mining: <ok>

Player looks at a tree: <mine this>

Help_bot starts mining the tree.

Player: "At least it knows what 'this' is."

The need to demonstrate the location with the player's avatar meant that proximal locations were much easier to point out than distal locations. Many players expressed frustration at their inability to designate specific places at a distance.

As mentioned previously, players expressed some concern about help_bot's judgement regarding the boundaries of locations and objects. There were times when a player might request more of a resource that they were currently working on, and worried that help_bot might mine that resource straight out of their built structure, as it was the closest source.

"Does it know when to stop? And if I had a house full of oak wood, and I said 'go get some oak wood', would it take the oak wood from my house?"

Responses to prompts for additional information

We defined three prompt messages for help_bot to use in response to messages that were outside the parameters we had set for its understanding. These indicated varying levels of comprehension, and elicited different types of responses from the players.

The first prompt was used when the player entered a request that was comprehensible to help_bot, but for which there was no clear or default location. This included requests for a resource type that was not currently visible, and requests for building a shape that did not specify where the building should go. In the pilot phase, the prompt text was <where?> or <from where?>, but players expressed confusion about how to answer this. One player asked if they needed to enter coordinates on Minecraft's grid reference, which was not a feasible solution. For the remainder of the study, we changed help_bot's prompt to <show me where>, and observed that players showed less uncertainty about this prompt, and consistently responded by physically moving their avatar to the desired location. We inferred from this that a query is easier for the player to respond to if it suggests a form of response, compared with one that is less prescriptive about how it should be answered.

The second prompt was <show me how>, used when the player entered a message that was recognisable in form as a building, collection or other action request, but which required knowledge that help_bot was not initially given. A typical case was a request to construct a complex object, since help_bot recognised only the names of geometric shapes unless taught other configurations by the player. Players most often responded to this prompt by giving up the request, but in several cases responded by demonstrating the behaviour they wanted and asking help_bot to copy it, as in this example:

Player: <build a tree>

Help_bot: <show me how>

Player: <no build a maze>

Help_bot: <show me how>

Player: <just watch>

Help_bot: <ok>

Player constructs a maze out of blocks.

Player: <now build a maze>

In these demonstrations for help_bot, players consistently marked either the start or the end of the action sequence with an explicit message, but not both. Help_bot was expected to recognise the completed structure as a Gestalt "object", and thereby understand where the relevant actions had begun or ended.

In contrast, a few players used more detailed messages to explain what they wanted, rather than demonstrating and expecting help_bot to replicate their actions:

Player:<help build a house>

Help_bot:<show me how>

Player:<put some cobble on top of each other>

Help_bot:<ok>

Help_bot begins stacking blocks of cobble. Player places more blocks alongside it to form a box shape, and help_bot follows this shape.

Help_bot:<done>

Player:<put some cobble on the top to make a roof>

Help_bot:<ok>

When messages were not recognisable as a command or a request, help_bot's fallback prompt was <?>. This was used in response to

spelling mistakes, sentence fragments and verbs that were outside help_bot's abilities (such as <die> and <write hello>). When players received this response, most were quick to assume that help_bot was unable to perform the action they had requested, and simplified or abandoned their request. However, in many cases they were mistaken: the action was within the abilities we had defined for help_bot, but the message formulation was not. Spelling was particularly notable, as players sometimes did not notice their own spelling mistakes, which left them with the false impression that help_bot could not understand the message they *intended* to write, rather than the message that they actually wrote. From this, we infer a need for feedback messages that are specific about the source of the lack of understanding. For example, rather than saying <?> or <I didn't understand that>, a message might say <I don't recognise the word 'whool'>.

Feedback on natural language input

Despite the challenges of natural language input, most players commented that they enjoyed it more than interaction via the traditional game controls or menus. A few players noted that menu-based interactions, such as those that exist for some friendly characters in Minecraft, would be preferable to natural language text as they provide more structure and clarity about the character's abilities. However, most considered the natural language input style an overall benefit as it opened up the possibility space for what they could potentially do with the character.

Players used the dialogue to consider how help_bot "thought", in some cases actively probing it with questions about its abilities or variations on a text prompt to test how it responded. The text responses provided a relatively clear channel to understand what was happening inside help_bot's "brain", and gave a sense that it was updating its behaviour.

> "With the one that I couldn't speak to, if I had tried to get wood or something, it didn't have the same feel that it was learning. So the text one, it felt like it was learning because it was saying ok, yes, I know how to do this."

Players also stated that natural language gave help_bot a greater sense of being alive and engaged with the player, compared with the non-speaking version, which made it more enjoyable to interact with:

> "Typing feels more interactive, like you're talking to a real person. Pressing stuff doesn't feel like that, like you're just talking to a computer."

Two players commented that they would like to have the option to talk to help_bot through speech rather than text messages, to further extend the feeling of a living character. However, they were uncertain about the ability of speech recognition technology to work well enough to support this.

> "It would be cool if you could do voice commands. But then again, you'd have to have the most to-your-country accent, otherwise it wouldn't understand it."

Expectations and cues for understanding the agent

The way in which players interacted with help_bot was influenced by their prior knowledge in several domains. Most obviously, players' experiences with Minecraft guided many of their initial attempts to understand and engage the character. Players compared help_bot to Minecraft's villager and wolf NPCs (non-player characters), for example in surmising that help_bot would attack any enemy NPC that the player attacked, because this was the behaviour for a tamed wolf. Players also drew on their experiences with other humans in the multiplayer game; two players tried to engage help_bot by repeatedly crouching their avatar in front of it, which is a social custom in online Minecraft equivalent to waving "hello". Finally, players compared the text inputs in our

study to the console commands in Minecraft. These commands allow players to edit the state of the game by entering a text string beginning with a forward slash – for example, </time set 6000> to change the in-world clock to midday. Several players initially started their messages to help_bot with a forward slash, until the facilitator pointed out that it was unnecessary. One player asked whether command strings in the same format could be used to manage help_bot's learning, such as </train help_bot X> to learn a current action and </set help_bot X> to repeat that action at a later time. These influences show that the context in which an AI character is deployed will influence the way users understand it and expect it to behave.

Expectations were also drawn from sources beyond Minecraft. Players who had programming experience compared the natural language inputs in our study to the programming language that they had used, and this guided their thinking about what might be possible. Real-world social cues were also applied: players expressed discomfort when help_bot followed their avatar too persistently, stood too close to it, or stared at it for too long. And unsurprisingly, several players referred to film depictions of robots and AI characters, such as *The Terminator*:

> "At one point I was like, 'ok ok ok ok, that's enough!' It kind of reminded me of in a film where there's like robots and they go out of control. That's why I was afraid to start digging the ground to get a flat bit, because I was afraid it would just start levelling the whole world."

Players' preconceptions about help_bot were, in some cases, well suited to the protocols we had designed for it, and in other cases beyond its abilities. What we found notable was that these expectations were sometimes assumed to be true, if only unconsciously, without having been tested. For example, one player travelled far away from help_bot and was confused when it was unable to find its way to them, as their experience with other NPCs had taught them to expect friendly characters in Minecraft to teleport near the player if they strayed too far away. This raises

the importance of providing the right contextual cues for players to form the right mental model of how an AI character works, as incorrect expectations may otherwise be set and not tested.

Variation in engagement and anthropomorphisation

We observed substantial variations between players in the ways that they engaged with, reacted to and spoke about help_bot. Putting these differences together, we hypothesise that they reflect two main dimensions on which attitudes towards help_bot varied. The first was level of engagement, or the extent to which players were interested in interacting with help_bot. The second was anthropomorphisation, or the extent to which players acted as though help_bot had human-like thoughts and feelings.

Differences in the level of engagement were apparent in the time each player spent interacting with help_bot during the tasks. As figure 1 shows, more than half of the players sent no more than seven messages to help_bot throughout the task, or less than one every two minutes, whereas several players sent more than twice this many messages. This variation carried over to other behaviours as well, including the amount of time the player spent watching help_bot, and the amount of interest they expressed verbally during and after the tasks. Highly engaged players spent more time experimenting with help_bot to determine its capabilities and asking the facilitator questions about it, and in some cases largely abandoned the construction task we had set in favour of playing with help_bot. We observed that several of the players who showed the greatest interest in help_bot also held higher expectations that it was capable of complex behaviour, although whether there was a causal relationship is unclear; it could simply be that these players thought and spoke more about the possibilities.

There are several behaviours wrapped up in what we are calling “anthropomorphisation”, each of which represents an attitude that the AI character has human qualities. Players varied in the

language they used in text messages, from those who entered only terse verb-noun commands such as <kill sheep> to those who greeted help_bot with a <hello>, framed their commands as polite requests such as <can you bring me some birch wood please>, and thanked help_bot for completing tasks. Players showed varying levels of empathy for help_bot, from those who casually hit it with an axe when it was in the way, to those who expressed concern about its wellbeing. One such player avoided clicking on help_bot, concerned that they would hit it by accident, and expressed guilt at making their avatar eat food in front of help_bot:

> "I feel kind of bad eating it – can I give this to you?"
>
> *Player gives some of the food to help_bot and demonstrates eating it with their avatar.*
>
> "Did they eat it? I don't see it, I assume they ate it. Now I feel slightly less bad."

When talking about instructing help_bot, some players described it in terms of a brainless instrument that could be programmed to perform repetitive actions, whereas others gave it tasks that required more independent, sophisticated and arguably human-like judgement. As an example of the latter, one player repeatedly set up pits for help_bot to fall into, explaining that they were trying to teach it to avoid the situation by looking out for and filling in any pits that it encountered in the future. Players also expressed an expectation that help_bot would prioritise tasks in a common-sense fashion, so that when a hostile creature attacked, for example, they were surprised if help_bot did not automatically come to their assistance. As such cases happened only rarely during the test, it was not certain whether this expectation was higher among participants who had higher expectations for other aspects of help_bot's judgement.

The behaviours that indicated low or high anthropomorphisation appeared to cluster together in individual participants. A player who expressed empathy for help_bot was also frequently one who

gave it higher-level instructions with more room for autonomy, and one who described it more as a character with a mind than as a plain instrument. That is not to say that players who showed higher anthropomorphisation believed help_bot had human intelligence or emotions, but they appeared more inclined to act as though it did.

	Low anthropomorphism	**High anthropomorphism**
High engagement	Treated agent as an instrument to be programmed	Treated agent as a character capable of judgement
Low engagement	Inattentive to agent	Polite to but uninterested in agent

Table 1: 2×2 model of player attitudes to the AI agent (help_bot), showing how variations in engagement and anthropomorphism resulted in different behaviours.

Notably, the level of engagement and the level of anthropomorphisation were at least partially independent of each other. Some players showed relatively little interest in help_bot, but addressed it courteously in the few messages they did send. Other players spent considerable time testing out help_bot's abilities and talked about it with enthusiasm, but as an interesting tool that they could program rather than as a character. There were players who liked help_bot as a potential sidekick character that could exercise independent judgement, and those who largely ignored it and said little to suggest that it had an inner life. The variations are summarised in table 1. This study is too small-scale and unstructured to draw firm conclusions about these variables, but we put them forward as a possibility to investigate in later research.

DISCUSSION

Our analysis shows that players' messages to help_bot primarily used simple syntax and direct commands, but that there was substantial complexity and variation in the details of wording and the way in which text messages were paired with in-game actions. Few messages were repeated between different players, and players invested considerable thought into their choice of words due to the difficulty of communicating with an AI that does not have the common ground of knowledge and judgement shared by most humans. Indirect speech acts (Searle 1975) were common, particularly in the form of commands with interrogative phrasing, which highlights the need for a natural language interface to either distinguish between direct and indirect commands, or remind players to use direct syntax. A contextual autocompletion function would seem suitable, to provide guidance to players as they are formulating the phrasing to translate their intention into words, which was often a moment of hesitation.

The findings also show that messages rarely contained all the information needed to interpret them correctly within the words themselves. Contextual information was also required. Much of this missing information could be inferred quite simply in the scenario we studied, but would become ambiguous in other contexts. For example, it could be assumed in our study scenario that help_bot was the subject of imperative messages as there was no other conversation partner in the game, but in a multiplayer or multi-agent game situation the subject would need to be stated by the player or inferred by the agent. Other messages required reference to the dynamic game state to be accurately interpreted, as in the use of deictic words such as "there", "that" or "away" matched with the player's avatar's location and gaze direction. (This is suggestive of one of the earliest multimodal interface models, *Put-That-There*, which combines speech, gesture and gaze to determine the user's input (Bolt 1980).) Players were aware of the inherent indeterminacy of their language, and expressed concerns about help_bot's ability to make judgements that would

seem sensible for a human, such as distinguishing between the "natural" and built environment (that is, between what the game generated and what the player constructed). Thus, the challenge for natural language agents in games is not only to make correct judgements about the player's intentions, but also to communicate the results of those judgements to the player.

The need to communicate the agent's internal decision-making is even greater in the context of interactive machine learning tasks, wherein the player is directly teaching the agent to learn new behaviour or change its existing behaviour. Help_bot's messages reassured players that it was learning, and went some way towards clarifying what it was learning and what it was not, despite consisting of only a few words. This appears to be linked to the sentiment that help_bot felt more like a living, thinking person when it talked: these signals conveyed the sense that it had a mind, rather than just a behavioural algorithm. Accordingly, players' strategies for teaching help_bot focused on communicating concepts through demonstration and example, rather than training help_bot through frequent feedback on its actions. (Note that players were told before each task that help_bot could learn from their feedback.) This is consistent with previous studies (Amershi et al. 2014; Kaochar et al. 2011), and poses a technical difficulty as many interactive machine learning approaches rely on user feedback to iteratively adapt the agent's behaviour (for example, Knox 2013). Some players did perceive themselves to be giving help_bot feedback, although this came in the form of implicit signals, such as acknowledging messages like <thank you> and contradictory actions like undoing help_bot's work.

Users may be encouraged to take a more direct teaching role by giving them tools to make their feedback more precise, and by providing clear feedback on what has been learned. Participants in our study were hesitant to engage in teaching, partly due to a perceived ambiguity about precisely what help_bot was learning from their actions. In addition, the fact that our help_bot was designed to continuously observe and adapt its behaviour to the

player's actions made it hard for our participants to recognise behaviour that was permanently learned, as opposed to momentary imitation. Participants generally preferred to be able to control when and where help_bot was taking in information for its own learning.A suggestion to reduce the ambiguity around help_bot's learning was to give players the ability to toggle it between a learning mode and a non-learning mode.

The way in which a subset of players anthropomorphised help_bot is consistent with past studies of conversational interaction (Luger and Sellen 2016) and empathic agents (Paiva et al. 2017), and representative of a wider effect in human-computer interaction: the tendency for people to respond to computers as though they are human, which Reeves and Nass have dubbed the "Media Equation" (1996). Nass and Moon argue that users "mindlessly" (2000, 82) apply social rules and expectations to computers, and Nass and Brave (2005) suggest that this effect is particularly strong for interactions involving speech. The strength of this effect has been challenged (Shechtman & Horowitz 2003; Lang et al. 2013), but there is some evidence that users apply more social behaviour to computer characters as their appearance becomes more human-like (Gong 2008). This implies that natural language interaction is a modality that will elicit more social reactions, as players in our study ascribed a greater sense of humanity and intelligence to help_bot when it used text, despite the text being limited to only a few formulaic phrases.

These variations in players' attitudes and expectations towards help_bot are important because they show that players have different mental models (Norman 1983) of how the agent works and what it is capable of. Our findings suggest that there is no universal starting point or blank slate in how players will perceive an agent. Expectations about an agent's degree and form of "intelligence", its adaptability and its responsiveness to different inputs are influenced by both the presentation and context of the agent, and background knowledge drawn from science fiction, previous game experiences, and real-world human social customs.

To facilitate players having a smooth experience with an agent like help_bot, designers will need to evaluate the context of the game genre and the appearance of the character to anticipate what kinds of expectations players may have, and consider how both implicit cues and explicit messages may serve to guide players to adopt the right mental model for working with their character.

Limitations and future directions

Our concern in this study was not only to evaluate what players said to the agent, but to observe players' actions in the context of gameplay, and to examine their reasoning in the post-task interviews. By nature, this was a limited study of a relatively small group of participants of one age bracket in one geographical location in a single language. A broader-based study would be needed to determine how representative our measurements are of natural language interactions with game characters by other groups of users, or to make statistical comparisons of message patterns under different conditions. One intriguing question, which we plan to address in a future study, is whether players use language differently when (knowingly) communicating with an AI compared to another human player, and in what respects their language is different or similar.

Our study has not been designed to elicit comparative preferences between natural language messages and conventional game dialogue systems such as branching conversation trees. As we have reported, our participants did compare the natural language system to other methods of interacting with game characters, but unsurprisingly these comparisons centred on Minecraft's own system, which is mostly wordless and oriented towards trading goods. In the most directly relevant comparative study, Sali et al. (2010) found that players preferred typing natural language messages to game characters over choosing messages from a menu, even though they encountered a higher number of errors, and frustration with the natural language mode. This is consistent with our findings. However, both studies looked at only a single

play session, and involved participants for whom natural language was a novel way of interacting with game characters. It remains to be seen whether a preference for natural language would continue over a longer time period.

CONCLUSION

Overall, our study shows that there are substantial commonalities in the syntax and concepts that players use in natural language interactions with a game character with learning AI, but also significant variations in the specific wording, behaviour and expectations for the character, which are driven by players' prior knowledge and contextual cues. While natural language interaction offers the promise of a flexible, engaging and intuitive way to interact with and teach AI agents in games, much work will be required to realise this prospect. Our study did not specifically set out to measure the extent of anthropomorphism in players' mental models of the character, but we have identified this as an area for further investigation to determine how it can be used either as a design resource to shape the interaction, or as a pitfall to be avoided. In future work we will look more systematically at the effects of anthropomorphism on language-based interaction with an AI character.

ACKNOWLEDGEMENTS

This work was supported by Microsoft Research and the Australian Government Research Training Program.

BIBLIOGRAPHY

Amershi, Saleema, Maya Cakmak, W Bradley Knox, and Todd Kulesza. 2014. "Power to the People: The Role of Humans in Interactive Machine Learning." *AI Magazine*35 (4).

Bakkes, Sander C. J., Pieter H. M. Spronck, and Giel van Lankveld. 2012. “Player Behavioural Modelling for Video Games.” *Entertainment Computing* 3(3): 71–79. https://doi.org/10.1016/j.entcom.2011.12.001.

Bayliss, Peter. 2007. “Notes Toward a Sense of Embodied Gameplay.” In *DiGRA ’07 – Proceedings of the 2007 DiGRA International Conference: Situated Play*, The University of Tokyo, Japan. http://www.digra.org/digital-library/publications/notes-toward-a-sense-of-embodied-gameplay/.

Bellemare, Marc G., Yavar Naddaf, Joel Veness, and Michael Bowling. 2012. “The Arcade Learning Environment: An Evaluation Platform for General Agents.” *ArXiv:1207.4708 [Cs]*, July. https://doi.org/10.1613/jair.3912.

Bernotat, Jasmin, Birte Schiffhauer, Friederike Eyssel, Patrick Holthaus, Christian Leichsenring, Viktor Richter, Marian Pohling, et al. 2016. “Welcome to the Future – How Naïve Users Intuitively Address an Intelligent Robotics Apartment.” In *International Conference on Social Robotics 2016*, edited by Arvin Agah, John-John Cabibihan, Ayanna M. Howard, Miguel A. Salichs, and Hongsheng He, 982–92. Cham, Switzerland: Springer International Publishing. http://dx.doi.org/10.1007/978-3-319-47437-3_96.

Bolt, Richard A. 1980. “‘Put-That-There’: Voice and Gesture at the Graphics Interface.” In *Proceedings of the 7th Annual Conference on Computer Graphics and Interactive Techniques*, 262–270. SIGGRAPH ’80. New York, NY: ACM. https://doi.org/10.1145/800250.807503.

Button, Graham, and Wes Sharrock. 1995. “On Simulacrums of Conversation: Toward a Clarification of the Relevance of Conversation Analysis for Human-Computer Interaction.” *Cambridge Series on Human Computer Interaction*, 107–25.

Caine, Kelly. 2016. “Local Standards for Sample Size at CHI.” In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, 981–992. CHI ’16. New York, NY: ACM. https://doi.org/10.1145/2858036.2858498.

Cakmak, Maya, Crystal Chao, and Andrea L Thomaz. 2010. “Designing Interactions for Robot Active Learners.” *IEEE Transactions on Autonomous Mental Development* 2(2): 108–18.

Cassell, Justine, Joseph Sullivan, Scott Prevost, and Elizabeth Churchill. 2000. *Embodied Conversational Agents*. Cambridge, MA: MIT Press.

Eckersley, Peter, Yomna Nasser, Yann Bayle, Owain Evans, Gennie Gebhart, and Dustin Schwenk. 2017. “AI Progress Measurement Project.” Electronic Frontier Foundation. 12 June 2017. https://www.eff.org/ai/metrics.

Farooq, Sehar Shahzad, In-Suk Oh, Man-Jae Kim, and Kyung Joong Kim. 2016. “StarCraft AI Competition Report.” *AI Magazine* 37(2): 102–7.

Fischer, Kerstin, Katrin S Lohan, Chrystopher Nehaniv, and Hagen Lehmann. 2013. “Effects of Different Kinds of Robot Feedback.” In *Proceedings of the 5th International Conference on Social Robotics*, 260–69. New York, NY: Springer. https://doi.org/10.1007/978-3-319-02675-6_26.

Goldberg, Yoav. 2016. “A Primer on Neural Network Models for Natural Language Processing.” *Journal of Artificial Intelligence Research* 57: 345–420.

Gong, Li. 2008. “How Social is Social Responses to Computers? The Function of the Degree of Anthropomorphism in Computer Representations.” *Computers in Human Behavior*24 (4): 1494–1509. https://doi.org/10.1016/j.chb.2007.05.007.

Goodrich, Michael A., and Alan C. Schultz. 2008. “Human–Robot Interaction: A Survey.” *Foundations and Trends in Human–Computer Interaction* 1(3): 203–75. https://doi.org/10.1561/1100000005.

Guillory, Andrew, and Jeff A. Bilmes. 2011. “Simultaneous Learning and Covering with Adversarial Noise.” In *Twenty-Eighth International Conference on Machine Learning (ICML 2011)*, 11:369–76.

Johnson, Matthew, Katja Hofmann, Tim Hutton, and David Bignell. 2016. “The Malmo Platform for Artificial Intelligence Experimentation.” In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence (IJCAI-16)*, 4246–47.

Juliani, Arthur. 2017. “Introducing: Unity Machine Learning Agents.” *Unity Blog*(blog). 2017. https://blogs.unity3d.com/2017/09/19/introducing-unity-machine-learning-agents/ (accessed Feb. 2018).

Kaochar, Tasneem, Raquel Torres Peralta, Clayton T. Morrison, Ian R. Fasel, Thomas J. Walsh, and Paul R. Cohen. 2011. “Towards Understanding How Humans Teach Robots.” In *Proceedings of the 19th International Conference on User Modeling, Adaption, and Personalization*, 347–352. UMAP’11. Berlin, Heidelberg: Springer-Verlag.

Kelley, J. F. 1983. “An Empirical Methodology for Writing User-Friendly Natural Language Computer Applications.” In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, 193–196. CHI ’83. New York, NY: ACM. https://doi.org/10.1145/800045.801609.

Kempka, Michał, Marek Wydmuch, Grzegorz Runc, Jakub Toczek, and Wojciech Jaśkowski. 2016. “ViZDoom: A Doom-Based AI Research Platform for Visual Reinforcement Learning.” *ArXiv:1605.02097*.

Knox, W. Bradley, Brian D. Glass, Bradley C. Love, W. Todd Maddox, and Peter Stone. 2012. "How Humans Teach Agents." *International Journal of Social Robotics*, 4(4): 409–21. https://doi.org/10.1007/s12369-012-0163-x.

Knox, W Bradley, Peter Stone, and Cynthia Breazeal. 2013. "Training a Robot via Human Feedback: A Case Study." In*International Conference on Social Robotics*, 8239:460–470. Berlin, Heidelberg: Springer.

Koenig, Nathan, Leila Takayama, and Maja Matarić. 2010. "Communication and Knowledge Sharing in Human–Robot Interaction and Learning from Demonstration." *Neural Networks*, Social Cognition: From Babies to Robots, 23 (8–9): 1104–12. https://doi.org/10.1016/j.neunet.2010.06.005.

Lang, Helmut, Melina Klepsch, Florian Nothdurft, Tina Seufert, and Wolfgang Minker. 2013. "Are Computers Still Social Actors?" In *CHI '13 Extended Abstracts on Human Factors in Computing Systems*, 859–864. CHI '13. New York, NY: ACM. https://doi.org/10.1145/2468356.2468510.

Lessard, Jonathan. 2016. "Designing Natural-Language Game Conversations." In *Proceedings of the First International Joint Conference of DiGRA and FDG*, Dundee, Scotland. http://www.digra.org/digital-library/publications/designing-natural-language-game-conversations/.

Luger, Ewa, and Abigail Sellen. 2016. "'Like Having a Really Bad PA': The Gulf Between User Expectation and Experience of Conversational Agents." In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, 5286–5297. CHI '16. New York, NY: ACM. https://doi.org/10.1145/2858036.2858288.

Mateas, Michael, and Andrew Stern. 2010. "Structuring Content in the Façade Interactive Drama Architecture." In *Proceedings of the First AAAI Conference on Artificial Intelligence and Interactive*

Digital Entertainment, 93–98. AIIDE'05. Marina del Rey, CA: AAAI Press.

Maulsby, David, Saul Greenberg, and Richard Mander. 1993. "Prototyping an Intelligent Agent through Wizard of Oz." In *Proceedings of the INTERACT '93 and CHI '93 Conference on Human Factors in Computing Systems*, 277–284. CHI '93. New York, NY: ACM. https://doi.org/10.1145/169059.169215.

Mnih, Volodymyr, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, et al. 2015. "Human-Level Control through Deep Reinforcement Learning." *Nature*, 518 (7540): 529–33. https://doi.org/10.1038/nature14236.

Muñoz-Avila, Hector, Christian Bauckhage, Michal Bida, Clare Bates Congdon, and Graham Kendall. 2013. "Learning and Game AI." In *Artificial and Computational Intelligence in Games*, edited by Simon M. Lucas, Michael Mateas, Mike Preuss, Pieter Spronck, and Julian Togelius, 6:33–43. Dagstuhl Follow-Ups. Dagstuhl, Germany: Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik.

Nass, Clifford, and Scott Brave. 2005. *Wired for Speech: How Voice Activates and Advances the Human-Computer Relationship*. Cambridge, MA: MIT Press.

Nass, Clifford, and Youngme Moon. 2000. "Machines and Mindlessness: Social Responses to Computers." *Journal of Social Issues*, 56 (1): 81.

Norman, Donald A. 1983. "Some Observations on Mental Models." In *Mental Models*, edited by Dedre Gentner and Albert L. Stevens, 7–14. Mahwah, NJ: Lawrence Erlbaum Associates.

Paiva, Ana, Iolanda Leite, Hana Boukricha, and Ipke Wachsmuth. 2017. "Empathy in Virtual Agents and Robots: A Survey." *ACM Transactions on Interactive Intelligent Systems*, 7(3): 11:1–11:40. https://doi.org/10.1145/2912150.

Reeves, Byron, and Clifford Nass. 1996. *The Media Equation: How People Treat Computers, Television, and New Media like Real People and Places*. CSLI Publications and Cambridge University Press.

Riek, Laurel. 2012. "Wizard of Oz Studies in HRI: A Systematic Review and New Reporting Guidelines." *Journal of Human-Robot Interaction*, 1(1): 119–36. https://doi.org/10.5898/JHRI.1.1.Riek.

Robertson, John. 2017. "Echo: Ex-Hitman Devs Bring Machine Learning to Stealth Games." Ars Technica. 15 August 2017. https://arstechnica.com/gaming/2017/08/echo-hitman-preview/ (accessed Feb. 2018).

Sacks, Harvey, Emanuel A. Schegloff, and Gail Jefferson. 1974. "A Simplest Systematics for the Organization of Turn-Taking for Conversation." *Language*, 50(4): 696–735. https://doi.org/10.2307/412243.

Sali, Serdar, Noah Wardrip-Fruin, Steven Dow, Michael Mateas, Sri Kurniawan, Aaron A. Reed, and Ronald Liu. 2010. "Playing with Words: From Intuition to Evaluation of Game Dialogue Interfaces." In *Proceedings of the Fifth International Conference on the Foundations of Digital Games*, 179–186. FDG '10. New York, NY: ACM. https://doi.org/10.1145/1822348.1822372.

Schegloff, Emanuel A., Gail Jefferson, and Harvey Sacks. 1977. "The Preference for Self-Correction in the Organization of Repair in Conversation." *Language*, 53 (2): 361–82. https://doi.org/10.2307/413107.

Searle, John R. 1969. *Speech Acts: An Essay in the Philosophy of Language*. Cambridge, UK: Cambridge University Press.

Searle, John R. 1975. "Indirect Speech Acts." In *Syntax and Semantics, Volume 3: Speech Acts*, edited by Peter Cole and Jerry L. Morgan, 59–82. New York, NY: Academic Press.

Shaker, Noor, Julian Togelius, Georgios N. Yannakakis, Likith Poovanna, Vinay S. Ethiraj, Stefan J. Johansson, Robert G. Reynolds, Leonard K Heether, Tom Schumann, and Marcus Gallagher. 2013. "The Turing Test Track of the 2012 Mario AI Championship: Entries and Evaluation." In *IEEE Conference on Computational Intelligence in Games (CIG)*, 1–8. IEEE. https://doi.org/10.1109/CIG.2013.6633634.

Shechtman, Nicole, and Leonard M. Horowitz. 2003. "Media Inequality in Conversation: How People Behave Differently When Interacting with Computers and People." In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, 281–288. CHI '03. New York, NY: ACM. https://doi.org/10.1145/642611.642661.

Shneiderman, Ben. 1982. "The Future of Interactive Systems and the Emergence of Direct Manipulation." *Behaviour & Information Technology*, 1 (3): 237–56. https://doi.org/10.1080/01449298208914450.

Silver, David, Aja Huang, Chris J. Maddison, Arthur Guez, Laurent Sifre, George van den Driessche, Julian Schrittwieser, et al. 2016. "Mastering the Game of Go with Deep Neural Networks and Tree Search." *Nature*, 529 (7587): 484–89. https://doi.org/10.1038/nature16961.

Stanley, Kenneth O., Bobby D. Bryant, and Risto Miikkulainen. 2005. "Evolving Neural Network Agents in the NERO Video Game." *Proceedings of the IEEE*, 182–89.

Sutton, Richard S., and Andrew G. Barto. 1998. *Reinforcement Learning: An Introduction*. 1st ed. Cambridge, MA: MIT Press.

Taljonick, Ryan. 2014. "Shadow of Mordor's Nemesis System Is Amazing–Here's How It Works." GamesRadar. 29 August 2014. http://www.gamesradar.com/shadow-mordor-nemesis-system-amazing-how-works/ (accessed Feb. 2018).

The Economist. 2017. “Why AI Researchers like Video Games.” 13 May 2017, sec. Science and Technology. https://www.economist.com/news/science-and-technology/21721890-games-help-them-understand-reality-why-ai-researchers-video-games (accessed Feb. 2018).

Yannakakis, Georgios N, and Julian Togelius. 2015. “A Panorama of Artificial and Computational Intelligence in Games.” *IEEE Transactions on Computational Intelligence and AI in Games,* 7 (4): 317–335.

Yannakakis, Georgios N., and Julian Togelius. 2017. *Artificial Intelligence and Games*. Springer.