Bot Colony – a Video Game Featuring Intelligent Language-Based Interaction with the Characters

Eugene Joseph, North Side Inc.

Abstract

Bot Colony is a single-person online adventure game by North Side Inc., in which the player needs to converse with robotic characters in order to accomplish an investigative mission. Besides entertaining its players, Bot Colony is also intended to help some of them learn English in a new, fun way - by playing a video game. A player's utterance – in unrestricted, free-form English - is processed by a dialogue pipeline where it is parsed, disambiguated, co-referenced resolved, and reasoned upon, following which an answer is generated. This paper describes the dialog functionality available in the Beta version of Bot Colony, and discusses the applications of intelligent interactive conversation in video games. Some advances in automated knowledge acquisition, entailment of dictionary definitions, paraphrase, co-reference resolution to 3D scenes, spatial relations, and using natural language to program 3D animation – enabling user-generated content - are presented. The paper is based on results of our R&D work on language/AI and 3D graphics, and the interaction between them.

1 Game Setting

The Bot Colony game (BCG) is based on the science-fiction novel of the same name, the Bot Colony book (BCB) (Joseph, 2010). In the book, Nakagawa Corp., a fictional Japanese robot manufacturer, moves its operations to the island of Agrihan in the Marianas to thwart industrial espionage attempts. Agrihan serves as a robot test bed before deployment in the future human colony on Mars, and the robots must operate autonomously. As the island is primarily populated by robots, the player has to converse with them in order to carry out his mission. The player gradually discovers the back-story of the game through these conversations, which are the main investigative tool. The player's ability to converse with the characters in unrestricted English is the key innovation of the game.

In the game, the player is given the mission to investigate the disappearance of three cutting-edge sensors extending robotic perception with smell, touch and taste. Later, the player is told that a spy has penetrated Nakagawa's sanctuary and the mission becomes finding this spy. Through his interactions, the player discovers that the spy has destabilized the islands' robots by programming a 'do as people do' law into them – a deeply flawed law of robotics that motivates interesting behaviour by the robots who 'crave' to become more human.

2 Motivation

The man-machine dialogues featured in the BCB set benchmarks for robotic disambiguation, co-reference resolution, natural language generation, inference, modality, world knowledge and linguistic performance in general. We wanted to meet the performance challenges set in the book, and transform science fiction ideas into working software technology.

The example presented below illustrates an actual player- robot dialogue in Bot Colony (BCG). Pi and Ri stand for player and robot utterance i, respectively:

P1: Who are you?
R1: `My name is Jimmy. I am a domestic robot.`
P2: Where are you?
R2: `In the house.`
P3: And where is the house?
R3: `On Agrihan island.`

P4: Who lives here?[1]

R4: Masaya, Ayame, Takeshi and Hideki.

P5: Who is Masaya?

R5: Masaya is a man.

P6: What does he do?

R6: He is the head of the sensor lab of Nakagawa Corp.

P7: And Ayame?

R7: What does Ayame do? Ayame is Masaya’s wife.

P8: I see. Are Takeshi and Hideki their children?

R8: Yes.

P9: How old are they?

R9: Takeshi is eight and Hideki is four.

P10: How do you know?

R10: Ayame told me.

P11: What did you say?

R11: I said, “Ayame told me”.

P12: How do you do your job?

R12: I prepare meals,I wash the dishes, I do the laundry and I babysit the kids.

P13: Why do you do this?

R13: To do my job.

P14: What is this?[2]

R14: It is a vase.

P15: How do you know?

R15: Because I can see it.

P16: Pick it up![3]

R16: (Robot picks up the vase).

P17: Put it in the center of the table.

R17: Which table, the dining room table or the living room table?[4]

P18: The first one.

R18: (Robot puts vase in the center of the table)

P19: You’re a genius!

R19: Why, thank you!

P20: Go to the big table.[5]

R20: OK (robot goes to table).

P21: Align the chair next to you with the table.[6]

R21: OK (robot lines up the back of the chair with edge of table)

P22: Now pick up the ceiling.

R22: I can’t, it doesn’t move.

P23: Do you snore?

R23: I don’t do that, but I know people do.

P24: Am I evil?

[1] co-reference resolution of ‘here’

[2] player clicks on object to learn its name, ESL feature

[3] command, co-reference resolution to 3D scene

[4] interactive co-reference resolution of similar objects in 3D scene

[5] dining-room table is bigger than living-room table, the property can be used for co-reference resolution of the intended table

[6] ‘next to you’ is a spatial, Boolean relation

R24: `You're asking if you enjoy harming others? I have no data about this in my memory.`

P25: I am evil.

R25: `Got it, you are evil.`

R26: Am I a nice person?

R26: `No.`

P27: How do you know?

R27: `You just told me.`

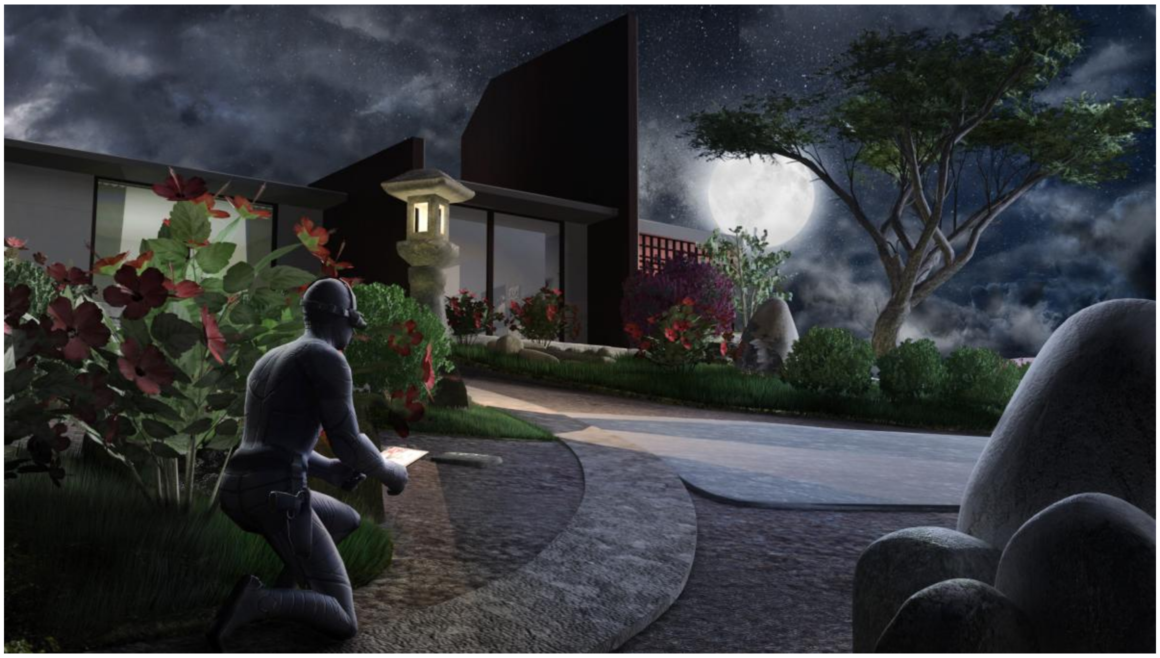
Fig. 1 Third-person gameplay, the player controls his ninja character in the garden. The player will communicate with the robot in the house using the tablet.

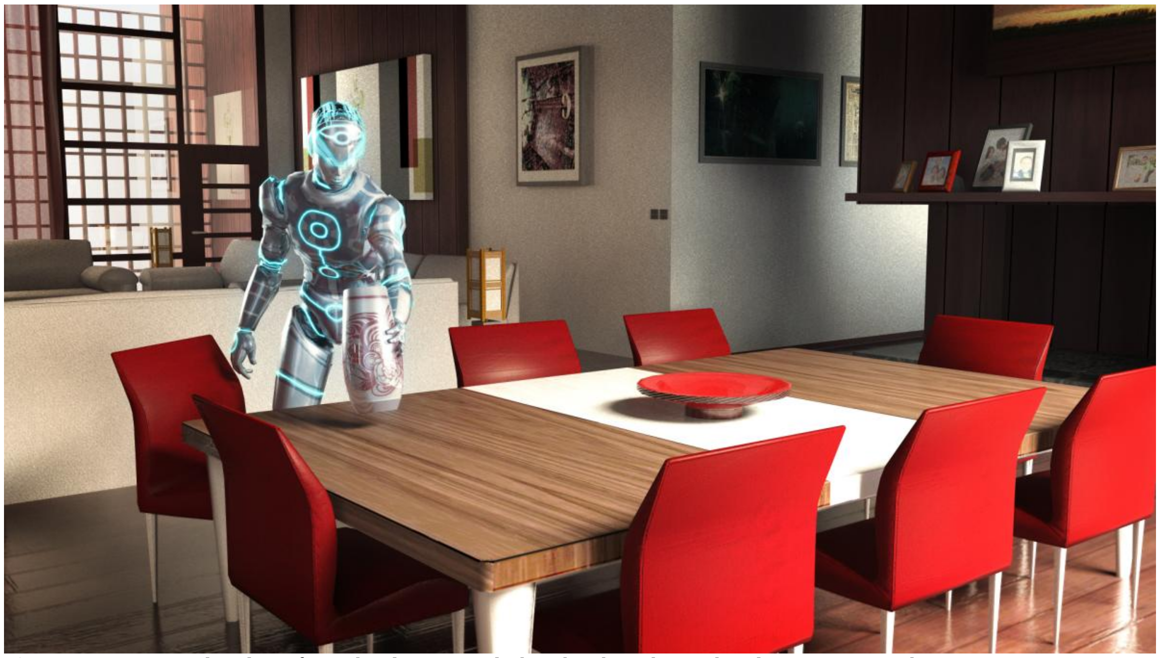
Fig. 2 First-person, mediated interface: The player controls the robot through speech and can converse with it.

3 Language-based real-time interaction with virtual characters

Early on, we made the choice that the characters the player can interact with through language be limited to robots: while it is acceptable for robotic interlocutors not to understand what a person is saying, this does not hold for human interlocutors. A human character not understanding what he or she is told would immediately dispel the suspension of disbelief - an essential element of an engaging game. If one intends to build a game featuring interactive conversation with human characters, the level of language understanding must be exceptionally high. See reference to LA Noire below.

4 Bot Colony architecture

Bot Colony has a client-server architecture. The client manages the 3D world, handles speech-to-text and text-to-speech, and communicates with the server, which handles dialogue.

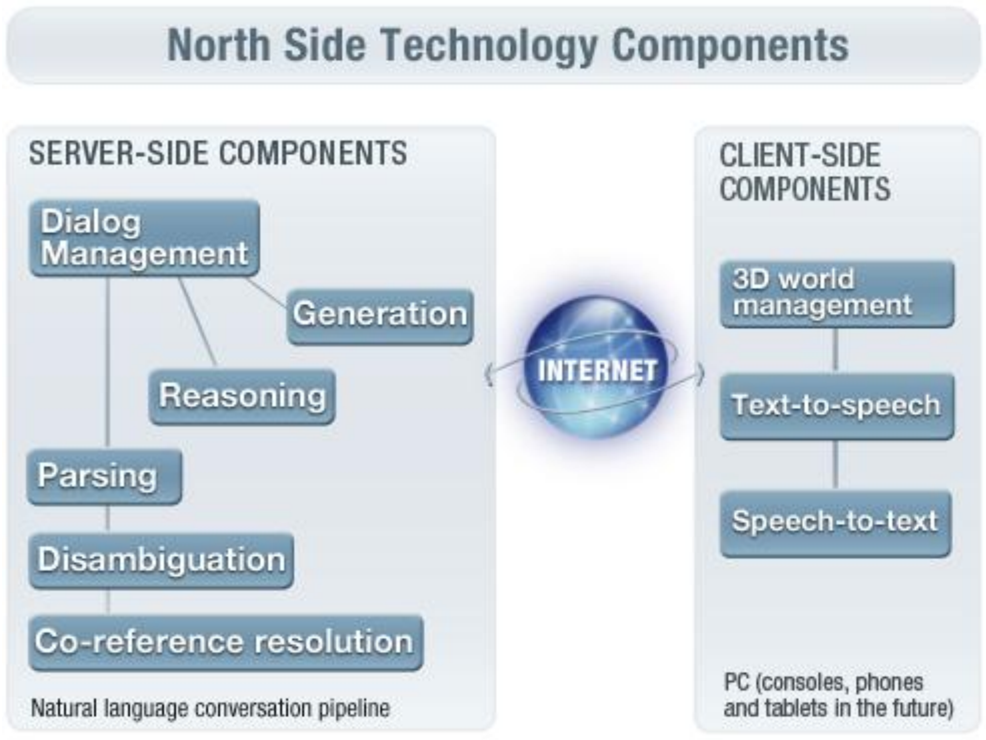

Fig. 3 The North Side linguistic server supports English conversation about user-specified facts, rules and database objects.

The language pipeline was designed to be generic, to enable its use in other conversational applications such as Intelligent Assistants, mobile computing, e-commerce, and language teaching. We plan to offer access to our language servers to game publishers interested in building interactive dialogue games, after the pipeline proves to be robust. We have put in place an architecture that will enable individuals to develop and publish 3D character animation content, and eventually interactive dialogue video games. See Natural-Language based Scripting, below.

The major components of our dialog pipeline are dialog management, parsing, disambiguation, co-reference resolution, Question Answering (QA) and reasoning and generation. Every one of them contributes to shaping 'the answer' - what a character will utter in response to what the player said. We will emphasize those aspects of the dialog pipeline that are of particular interest to online video games deployed internationally. Response time is always important, and needs to be roughly similar to a conversation between people, to maintain the feeling of interactive dialogue.

4.1 Speech recognition

Interaction starts client-side, with the player speaking or typing in English. We use the Nuance Dragon software for transliteration of spoken input into text. The player needs to invest about 30 minutes in training his or her acoustic profile, which continues being refined as additional dialogue takes place. Native English speakers achieve precision rates close to 100% under optimal conditions (all proper names defined, acoustic profile trained extensively). As the closed-Beta has not yet started at the time of writing, we have not yet experimented with ESL (English as a second language) players, expected to form a large part of our player community. Consequently, we are not yet able to predict the rates of speech recognition for non-native speakers with strong accents.

In order to give a player the opportunity to check transliteration accuracy, the text is only sent to the server after the player hits Enter. Since this slows down the dialogue, we will detect cases where players don't need to make corrections, and offer these players to send their text to the server without waiting for Enter.

Typing is supported as an alternative input method for players keen to jump in and try the game, without first training their acoustic profile.

4.2 Discourse planning and management

One of the major challenges in Bot Colony was integrating dialog smoothly with the 3D world. We wanted players to feel at home in the traditional environment of a 3D game - moving in the environment, looking around, or interacting with objects – while being able to converse with the characters at the same time.

As soon as the player is sufficiently close to a character, he or she can converse with it by speaking or typing. As illustrated above, the player can ask factual **questions** about location, time, objects, people, events, how to do something or why his interlocutor does something (R1 – R9). Answering questions such as 'How do you know X?' (R10, R15) and being environment-aware (see Reasoning) lends credibility to the virtual characters.

The player can also ask robots to **manipulate** objects in various ways in order to advance in the game (R16, R21). A key game mechanic in Bot Colony is extracting information from a robot in exchange for helping it learn about humanity (the robot is motivated by the 'do as people do' law). The player practises speaking to robots in the training level (Intruder), where he controls a robot remotely - through language - to erase all traces of an intrusion into a home. Other examples of language-controlled interactions or manipulations are:

- playing 3 card Monte against a robot,
- walking a robot through preparing maki-sushi
- helping a maintenance robot diagnose and repair another robot
- correcting yoga poses
- choreographing a circus routine for robotic animals

Getting a character to manipulate objects in 3D using language requires spatial support. Utterances containing prepositional phrases are resolved all the way down to geometry. Take, for example, the command P21 "Align the chair next to you with the table." We resolve such a command by doing co-reference resolution with the 3D scene and processing the spatial relationships between the objects referenced. The availability of 3D models grounds spatial relation recognition,avoiding the need for labelling approaches such as (Kordjamshidi, Bethard, and Moens, 2012).

Other dialog acts detected in BCG: The player make offer and counter-offer dialog acts when negotiating the price of an item in the Bazaar, or describing the emotions of human participants in cartoons (opinion dialog act).

To phrase the examples above in Dialog Management language, *dialogue act recognition* is at the heart of our dialog management. Bot Colony attempts to do automated dialogue act recognition. Very challenging areas are the extraction of meaning from various formulations of the same semantic message. As a rule, these are processed using various forms of entailment (see Reasoning below).

The distribution of dialogue acts in Bot Colony differs from the distribution in the Switchboard Corpus in Stolcke[6] as QUESTIONS are more frequent. COMMANDS are also important in Bot Colony, as the player can instruct robots to perform tasks. The Switchboard Corpus contains telephone conversations and is unlikely to contain commands, which indeed these are not listed in Stolcke[6]. In the Beta version, Player STATEMENTS are remembered by the robot, but not yet checked for consistency with previous statements. OPINIONS are detected and remembered. YES ANSWER, NO ANSWER, AGREEMENT, DISAGREEMENT, CONVENTIONAL OPENINGS, OFFERS, OPTIONS AND COMMITS, etc. are also supported.

To make the game interesting, it is important that the player not be the only one initiating dialogue. Mixed initiative dialogue lets a robot take initiative, and address relevant utterances to the player. Proximity to an object can be used as a trigger for the robot to point to it, and make a comment or ask a question about it.

Teaching a robot new concepts or new animations sequences is a very important requirement for personalizing a robot and increasing the replay value of the game. This teaching is done through a multi-step dialogue, involving clarification. Multi-step dialogue and ellipsis is another challenging area in Dialog Management.

4.3 Parsing

When the player's input gets to the server, it is parsed. Precision and recall are critical (and contradictory) goals, as common parsing errors jeopardize dialog success. The most common errors are part of speech errors (for example, interpreting a verb as a noun) or wrong phrase attachment.

There are no limitations as to the vocabulary that can be used by the player, and Bot Colony supports both idioms and phrasal verbs. To achieve good parsing results (leading to the robot understanding the player), our players will have to use well-formed English, at least at the beginning of the closed-Beta program. We recognize that this limitation does not jive very well with the mission of the game to be an ESL tool, since players who want to learn or improve their English will likely make many mistakes. We foresee investing significant effort in categorizing English errors by nationality and correcting the player's English in the ESL flavour of the game.

4.4 Disambiguation

Since English is highly polysemic, precise disambiguation is critical to extract the right semantics out of the player's utterance. Picking a wrong sense has nefarious consequences, sending the dialogue on the wrong track. If a robot is hot, is it trendy or at a high temperature? Is that window on the screen, or in the house?

In order to achieve high-performance in disambiguation, our disambiguation module integrates syntactic and semantic criteria with domain knowledge and the accumulated dialogue context.

4.5 Co-reference resolution

Any video game incorporating full-fledged interactive dialogue in natural language will require a new kind of co-reference resolution venturing into pragmatics. Traditional co-reference resolution does not take into account the 3D world, but in video game interactive dialogue, this is a necessity. In Bot Colony, we use the 3D scene both for co-reference resolution and to resolve spatial relations (R17, R18, R21). This enables the player and the characters to use language to refer to the objects surrounding them in the 3D environment, and is a key component of maintaining suspension of disbelief. Using the 3D environment for co-reference resolution requires an art pipeline in which assets are rigorously named.

Additionally, both the player and the character should be able to refer to what the other said (P11, P13), or what the other did. These are not entities, but rather situations. Co-reference resolution to utterances in dialogue is supported in BCG.

Completely simulating environmental awareness of characters - being able to ask a character to describe what he sees in front of him, or asking him about a noise that was just heard - poses a number of significant challenges. A BCG character is 'self-aware' of the consequences of its actions – for example, it knows that it holds a vase it just grabbed. It is important to meet these modelling challenges in order to maintain suspension of disbelief.

4.6 Question Answering and Reasoning

The front-end modules in our pipeline (dialog manager, parser, disambiguation and co-reference resolution modules) cooperate in generating a precise logic form that is sent to the QA and reasoning module. The latter is responsible for producing most (but not all) of the answers the player will hear or see printed on the screen.

Once the technical obstacles of producing a good logic form have been overcome, and the machinery needed to manipulate this logic form and match it with a knowledge base works well – a difficult task, as the two logic forms can be widely different - the next barrier is the availability of world-knowledge.

Even when the player stays on track and asks relevant questions and utters relevant messages, generating good answers can be a challenge due to the lack of world knowledge. It is reasonable to impose the following legitimate limitation on the dialogue with a robotic character (as it is done in BCG): a robot needs to understand as a minimum the concepts and utterances related to its morphology (body-parts), its tasks, its immediate physical environment, the events it witnessed first-hand, as well as concepts and utterances related to its perceptual abilities, its cognition, and its communication abilities. This is no small task. One consequence of this commitment to environment-awareness is that a third-person interface must be supported, since the player can ask a robot to describe *him*. In BCG, both first-person and third-person cameras are supported.

Questions that are not relevant to the player's mission need still need to be dealt with, since they will inevitably come. A big challenge is how to do this with the limited knowledge available, or how to acquire the missing knowledge. We pursue a knowledge mining strategy to acquire consensus reality knowledge used in dialogue (example R23). A robot will attempt to learn new concepts by paraphrasing them and co-referencing entities to his environment. We have independently developed an entailment approach that has objectives similar to those described in (Szpekor and Dagan, 2009) and (Ben Aharon, Szpektor and Dagan 2010), namely representing argument mapping as entailment rules. In (Szpektor and Dagan, 2009) these rules are augmented with subcategorization frames and functional roles, which the paper correctly identifies as a generally-needed extension for predicative entailment rules. As described in (Szpektor and Dagan, 2009) in order to apply a rule, the entailing template must be first matched in the text, which includes matching the template's syntactic dependency structure, functional roles and frame. In (Szpektor and Dagan, 2009) and (Ben Aharon, Szpektor and Dagan 2010), texts need to be annotated with this information, to compensate for the lack of precise word sense disambiguation (WSD). In our dialogue pipeline we rely on more precise parsing and WSD, so prior annotation is not required (and it is impractical in interactive dialogue). To achieve greater recall, our entailment works between a machine-readable dictionary entry and its definition (example R24 shows entailment between an adjective and a verb, identified as future research in (Szpektor and Dagan, 2009)). This entailment capability has applications to QA through simplification of query and fact base, simplification for ESL (Specia and Jauhar, 2012), interactive clarification through paraphrase, and automated knowledge acquisition.

Since answering in a cogent way is paramount to the quality of the experience, we're introduced in BCG a game mechanic designed to help keep player utterances tractable. The Humanity Detection Meter is displayed next to the head of a robotic interlocutor suspicious that the player (who passes as a robot by wearing a robot's badge which broadcasts the latter's identity) may actually be human. Therefore, one of the challenges of the player is to pass as a robot with unfriendly robots. If he fails, he is imprisoned for a while to 'provide services to the community', a euphemism for teaching robots about human beings so that robots can serve humans better.

4.7 Natural Language Generation

The QA and reasoning module generates a logic form as a result of inference or question answering processing. The generation module then realizes this logic form as surface text. Generation will resolve pronouns, ensure syntactic agreement, do aggregation, produce relative clauses, and so on.

In a video game application, generation needs to work in tandem with animation modules to increase realism: for example, the lips of the character (an android robot, for example) should be synchronized with the sound we hear (lip synch). In Bot Colony, we attempt to depict cognitive operations as they take place. States like the character looking for answer, not knowing, hesitating, experiencing a quasi-emotion are depicted through body language combining posture and gestures, or by using a 2D cognitive avatar in the case of non-humanoid robots.

Since our answers are computed dynamically, voice actors cannot be used with the technology described here. This introduces an artistic limitation. Solutions for infusing synthetic speech with emotion, as (Schröder & J. Trouvain, 2003), are very much a requirement for video games like ours, in order to render emotion dynamically.

Last but not least, a high quality, continuous language synthesis solution supporting multiple distinct voices is required. We use both Neospeech and Cepstral as language synthesis solutions.

4.8 Natural Language Scripting

Early on, we made the commitment to use controlled-English as the programming language for Bot Colony. Controlled English made a lot of sense for programming the game, as it provided the most natural way to describe the basic data elements required, all of them situations. Our programming paradigm is based on goal-context-plan (Dan Tecuci, 2007), a natural way to think about Intelligent Agents (robots). Here we use the term scripts to refer to programs written in controlled-English that handle *events* originating from the Client.

In our scripting facility, *events* is used in a broad sense, encompassing

- a spoken or typed utterance (the Client contains the Dragon SDK software) ,
- selection (clicking with the mouse on graphics of interest)
- an entity entering a 3D zone of interest,

- the distance between 2 objects being below or above a threshold
- timeouts of various timers

As a result of handling events in context, the script often dispatches animations and speech. Since these need to happen in parallel, our scripts support concurrent actions.

Our script engine works in an intuitive way: when an event occurs, it searches for a script that consumes it, based on location and interlocutors. For example, assume the event is the command “Pick up the vase!”. This script has a goal (‘Agent A pick up object X’), prerequisites (object X exists, object X is reachable, Agent A has hands), a plan (Agent A approaches object X, Agent A reaches for object X, Agent A grabs object X, Agent A bring hand(s) close to body), and side-effects (Agent A holds object X). The script engine checks the prerequisites for a goal before actually running the script.

Scripts are written for all Intelligent Agents (the player, or robots, in the case of Bot Colony). Since we don’t control the player, the scripts are used to try to guess his goals based on what he does or says (both of which are events). Scripts are also used to provide task-specific help. As far as the robots go, the script controls their behaviour completely, specifying the animations to be played and the utterances to be uttered. Robot utterances are not exclusively produced by scripts – as mentioned, a robot answer may originate from the QA and reasoning module or another specialized module such as path planning, in the case of instructions for getting to a place. However, our scripts have the ability to override answers to selected questions – we don’t want to give away the answer that will enable the player to immediately complete the level just because he happened to ask the right question.

A key consideration when designing our natural language based scripting facility was to enable players to make their own games. Events, conditions, goals, plan steps are all in English, which opens the medium to non-programmers. Using natural language will enable end-users to design 3D animated content (user-generated content): after the initial Beta, we plan to preface the training level with a new level (Party Time) that tests the player’s creativity in preparation for giving him the mission. In Party Time, the player has to design a robot party and upload its video to Nakagawa (the video can be posted to the game’s bulletin board for other players to view). The player will create his video through a point and click scripting facility providing access to events and animations of intelligent objects, specified in English This approach of specifying events and animations in natural language and through a point and click interface will i) enable non-prefessional animators (so basically everyone) to create 3D animations and games using natural language and ii) ensure that whatever a user scripts can actually be executed by the game engine - in particular, that all animations are available and well-specified for their characters.

5 Automated gameplay evaluation through language analysis

Semantic understanding of the player’s input facilitates automated evaluation of the quality of the gameplay experience on a global basis. We plan to semantically process dialog data collected from the closed-Beta program to detect problematic interactions and game situations, and improve them. These situations are of particular interest:

- player does not understand the robot
- robot does not understand the player
- the player does not understand his mission
- the player is lost in the game-world
- the player does not know how to solve a particular game task
- the player is frustrated with his interlocutor
- the player is frustrated with the game
- the player is pleased with the interlocutor/game

6 Comparing approaches to dialogue in games

6.1 Dialog Trees and keywords

Several video games (Mass Effect, Heavy Rain, Dragon Age) have used dialogue trees to interact with the player. Dialog trees are a linear way of traversing a game. There is no natural language processing with dialog trees, because the text selected by the player is not really 'understood' by the character with whom the player 'converses'. Since there is no need to actually process the text displayed in menu choices, game writers are free to make dialogue responses as witty, stylish or arcane as they wish.

LA Noire is a game in which the detective Cole Phelps solves police cases by investigating crime scenes and interrogating witnesses. LA Noire uses simplified dialogue trees: a player can accept a witness/suspect's account, doubt it, or challenge it as a lie citing evidence from the investigator's notebook. The character will respond in consequence. LA Noire excels in rendering the emotional range of the characters: realistic facial expressions and body language are perfectly synchronized with the lines delivered by the voice actors. While the Bot Colony technology could provide a lot of flexibility in interacting freely with the characters, the emotional response - body language, facial expression and voice parameters - would be extremely challenging to generate automatically. The subject of getting a virtual actor to approximate the performance of a human one is certainly a worthy a goal for future research.

Language has already been used in games for giving commands (in Ubisoft's Endwar and various text adventure games), and games like Façade and Spaceship Titanic made attempts at generating relevant pre-recorded output based on keywords detected in the input text.

6.2 Deep Semantic Understanding

Deep-semantic understanding of text is a new approach to handling dialogue in games. The player speaks for himself instead of picking ready-made utterances from a list created by the game writer; therefore, the player engages in real dialogue, not a linear script re-enactment subject to a preset order. The immersion is higher, as the character's response directly addresses the semantic content of the player's utterance, as in a conversation between two people.

In Bot Colony, we process player utterances using the language processing pipeline described above. While our technology does not yet pass the Turing test, we have made significant inroads into building natural language technology that enables an Intelligent Agent to converse using for input a fact base, events accumulated during gameplay, information drawn from the 3D environment, as well as rules and domain knowledge. We hope to understand language sufficiently well to enable players to complete the Bot Colony levels while having fun, while offering them a glimpse into what *realistic* interaction with robots will be like – a geeky, often funny feel absent in dramatizations like Asimov's I Robot or Blade Runner, whose Nexus 6 robots are just too close to humans to be believable. A strategic goal is to establish our technology as an innovative, fun way to learn English, especially for people learning English as a second language (ESL).

We believe that natural language understanding and generation opens up exciting opportunities for new game genres, new gameplay and a richer, more immersive game experience.

7 Conclusions

Building a functional dialogue pipeline integrated with 3D scenes is a formidable research task. North Side embarked upon language and AI R&D some five years before starting Bot Colony development in

October 2007. We continued R&D on language processing and its integration with the 3D scene in parallel with the development of the Bot Colony game (BCB was written at the same time). We have achieved sufficient precision in parsing, disambiguation and co-reference resolution to be able us to use entailment based on automated argument mapping to definitions for language understanding – a technology we believe to be of strategic importance. We have made advances in argument mapping to definitions enabling automated common-knowledge acquisition, concept simplification and paraphrase, co-reference resolution to 3D scenes, spatial relations, using natural language to program 3D animation – enabling user-generated content, as well as performace analysis through semantic language analysis.

We are at the beginning of the evaluation of Bot Colony and our dialogue pipeline. We are convinced that deep-semantic understanding of language has the potential to enrich the gameplay experience and increase player immersion in video games.

Acknowledgements

I thank the members of our language and graphics R&D teams for their contributions to North Side's core technology, and our artists, programmers and linguists for their contributions to the Bot Colony video game.

Parts of this work we supported by

- the National Research Council's IRAP program contract Natural Language Processing for Simulation and Videogame Applications, IRAP project 605841 June 2006, and Natural Language Processing for Simulation and Videogame Applications , IRAP project 697850 April 2009.
- the Defence Industry Research program contracts DIR-583 Concept-Proof Prototype Demonstrating the Feasibility of a Natural Language Interface to Complex Military Systems, contract W2207-07EM01, May 2007 and DIR-652 Development of a Natural Language Processing Spatial Awareness and Reasoning Capability for Virtual Crewmembers – contract W2207-10EM01, April 2010.
- Telefilm Canada and the Canadian Media Fund
- the Canadian federal SR&ED tax credit program, the Quebec SR&ED tax credit program, and the Quebec multimedia tax credit program.

The author wished to thank Prof. Guy Lapalme of the University of Montreal for his review of this paper. The author remains solely responsible for the claims made in the paper and any errors or omissions within.

References

Eugene Joseph. 2010. Bot Colony, published by North Side Inc.

Andreas Stolcke et al., Dialog Act Modelling for Automatic Tagging and Recognition of Conversational Speech. 2007. *In Computational Linguistics, Volume 26, Number 3.*

M. Schröder & J. Trouvain. 2003. The German Text-to-Speech Synthesis System MARY: A Tool for Research, Development and Teaching. *International Journal of Speech Technology.*

Dan Tecuci. 2007. A Generic Memory Module for Event. *PhD Disseratation Computer Sciences Department, The University of Texas at Austin, August 2007.*

Idan Szpektor, Ido Dagan. 2009. Augmenting WordNet-based Inteference with Argument Mapping. *In Proceeding of the 2009 Workshop of Applied Textual Inference, ACL-IJCNLP 2009.*

Roni Ben Aharon, Idan Szpektor, Ido Dagan. 2010. Generating Entailment Rules from FrameNet. *In Proceedings of the ACL 2010 Conference Short Papers.*

Parisa Kordjamshidi, Steven Bethard, and Marie-Francine Moens. 2012. SemEval-2012 Task 3: Spatial Role Labeling. *In First Joint Conference on Lexical and Computational Semantics (*SEM), Montreal, Canada, June 7 – 8, 2012.*

Lucia Specia, Sujay Kumar Jauhar and Rada Mihalcea. 2012. SemEval-2012 Task 1: English Lexical Simplification. *In First Joint Conference on Lexical and Computational Semantics (*SEM), Montreal, Canada, June 7 – 8, 2012.*

Phlip K. Dick. 1968. Do Androids Dream of Electric Sheep. The literary work that inspired Hampton Fancher's 1977 and David Peoples' 1980 Blade Runner scripts.