Autonomous self-learning agents in 3D virtual worlds

A performance comparison of cognitive agent architectures

Yusri Dollie
University of Cape Town
yusri@zhc.co.za

William Grant
University of Cape Town
willmgrant@gmail

Jonah Hooper
University of Cape Town
jonah.graham.hooper@gmail.com

KEYWORDS

Artificial Intelligence, Stochastic Environments, Hybrid BDI agent architecture, POMDP planning, POMDP architecture, Agent Evaluation, Agent performance metrics, Unity3D

1 PROJECT DESCRIPTION

In the field of artificial intelligence (AI) and agent learning, the truly cutting edge and interesting problems lie in creating agents which are able to learn, plan and make decisions in unpredictable and noisy environments. To this end we seek to develop a cognitive agent prototype combining the Belief-Desire-Intention (BDI) agent architecture with a Partially Observable Markov Decision Process (POMDP) framework. To showcase the efficacy of such an agent we also seek to develop a testing apparatus combined with a collection of metrics to evaluate the effectiveness of an agent in a 3D environment.

2 RELATED WORK

This research project delves into the field of cognitive agents in simulated worlds, looking specifically at the performance evaluation of a cognitive agent which uses a combination of architecture designs.

In order to provide the reader with context and clarity the following sections detail the QCog framework which forms the basis of the project into which new agents will be integrated and discusses current developments within the field, focusing on combined BDI-POMDP architectures, the use of BDI agents with probabilistic notions and knowledge representation and the use of POMDP's as an agent framework. Finally we discuss the commonly accepted metrics used when testing cognitive agents in simulated worlds.

2.1 QCog Architecture

QCog is an architecture designed for adaptive self-learning agents in 3D environments that are both complex and unpredictable. It is designed to be an experimental platform for agent development and evaluation using the Unity3D Game Engine[7]. Currently QCog contains a reinforcement learning mechanism that uses a dynamic policy selection mechanism that enables a cognitive agent to adapt to unknown situations in its environment[12]. Entities placed in the simulated world are designed to be generic and extensible for ease of use within the architecture. QCog features adjustable simulation settings which can be used to set the number of iterations you wish to perform, a simulation speed control mechanism, a data recorder which records important data metrics during the simulation, and finally a playback engine that allows simulations to be recorded and played back for further study whenever the user desires[12].[1]

[1]See Appendix B for architecture diagram detailing a QCog agent

2.2 Defining BDI Architecture

In systems which require high-level management of many different objectives, specifically control tasks in complex dynamic environments where the application of conventional techniques have proven to be difficult and expensive to build, verify and maintain. The BDI architecture represents just one possible solution to this problem in an agent-oriented system. It views the system as a rational agent which has certain mental inclinations of Belief, Desire and Intention (hence the acronym BDI), these inclinations represent respectively: the information, motivational and deliberative states of the agent and determine the agent's behavior [19].

In these environments it is a BDI's flexibility to reason over different goals that allows it to adapt to changing situations by focusing on the most appropriate objective at any particular time [21]

2.3 Defining POMDP's

A basic Markov Decision Process (MDP) can be described as a tuple $\langle S, A, T, R \rangle$, where

- S is a finite set of states of the world
- A is a finite set of actions
- T: S x A $\rightarrow \Pi(S)$ is the state-transition function giving each world state and agent action a probability distribution.
- R: S x A $\rightarrow \mathbb{R}$ which is the reward function giving the expected immediate reward gained by an agent for taking each action in each state.

In the MDP model the next state and reward depend only on the previous state and action and no other prior state-action pairs, this is what is defined as the Markov property. Agents utilizing the MDP framework attempt to act optimally by calculating an optimal policy, that is a policy which provides the maximum future discounted rewards at the end of a given execution loop. In a partially observable environment however the agent is no longer able to determine its current state with complete reliability. Thus we now consider the Partially Observable Markov Decision Process framework (POMDP). A POMDP can be described as tuple $\langle S, A, T, R, \Omega, O, \rangle$, where

- S, A, T & R describe a Markov decision process
- Ω is a finite set of observations the agent can experience of the world
- O : S x A $\rightarrow \Pi(\Omega)$ is the observation function which gives each action and resulting state a probability distribution over all possible observations.

Succinctly a POMPDP is simply a MDP where the agent is unable to observe the current state and must instead make an observation based on the action and resulting state, while the goal of maximizing future discounted rewards remains the same.

2.4 Defining Agent Performance, Metrics & Scenarios

Agent performance can be described as how effective an agent is in a given case using some means of scoring[3, 11, 13, 18].The score itself is dependent on environment and scenario. An example of a score would be the time taken for an agent to complete a given goal/task. In order to evaluate the performance of an agent and its adopted learning strategies we require a controlled environment in which we conduct experiments using appropriate measures of agent performance (metrics) so that a strategy can be meaningfully evaluated and compared[9]. This environment is known as a simulated world and within the context of this research project refers to the environments implemented using the QCog architecture[12]. The experiments mentioned above will be run in this simulated world as different testing scenarios with an underlying goal that the agent is required to complete. It must be noted that metrics do not strictly refer to an agent's performance but also refer to the measurement of the changes made to the parameters of the environment where applicable[18].

2.5 Common Metrics

From existing works the most common metrics used for evaluating agent performance are scoring, time, sensing, planning, deliberation, rate of world change and commitment towards goals. Alongside this there are other areas that can be measured such as the degree of complexity and noise/unpredictability within a given scenario[1, 3, 4, 6, 8–10, 13, 14, 18, 22, 23, 25]. Each of these can be broken down into various components such as: rates of occurrence; the number of occurrences; all of which can be represented as a factor of time. When discussing time it is important to note that there are issues with evaluating an agent's performance in real time and it is better to use a simulated world clock[11, 18]. Planning also includes a subset of actions that are reflexive to changes in the environment and these require separate measures[6]. Deliberation is also referred to as re-planning and is costly towards an agent's performance. To counter this many of the existing works filter possible actions that can be deliberated on[6, 10, 11, 14, 18, 23].

2.6 Existing Testing Frameworks

There already exist a number of testing frameworks that are used to evaluate performance of cognitive agents as well as various learning strategies that an agent may have. Examples of these types of learning that these strategies utilize are reinforcement, supervised, unsupervised and deep learning. Some of these frameworks are designed using a custom framework to create simulated worlds with agents placed in them.[5, 6, 11, 18] Other test frameworks use existing game engines as a foundation that are further extended to allow for AI research.[1, 3, 4, 8, 13, 14, 25]

2.7 Combining BDI-POMDP Agent Architectures

Different agent architectures excel under differing conditions. While BDI agents excel at long term goal planning and strategizing they traditionally have no notion of uncertainty and probabilities, and while POMDP agents excel at acting optimally in partially observable environments they are unable to manage multiple long and short term goals effectively.[16][17][15] Traditional BDI architectures do not generate plans, and those that do cannot handle stochastic actions and probabilistic observations [21]. To this end there has been a shift to combining the traditional BDI agent architecture with other architectures such as POMDP's to facilitate such planning. AgentSpeak^{+} , an extension to the AgentSpeak(L) programming logic and allows for the creation of agents that are able to complete probabilistic planning using the POMDP framework. Beliefs are represented as epistemic states allowing agents to reason about uncertain observations and through a POMDP, optimal actions are selected in pursuit of a goal within an uncertain environment. This in itself provides a platform for developing BDI agents in stochastic environments, where a policy library may not exist or represents incomplete domain knowledge [2]. Another approach is the extension of the BDI architecture to include a POMDP planning module, in order to facilitate the creation of plans which include stochastic actions[20].

In this regard the challenge lies in knowing when to change the current goal and then assign the new goal [24]. To account for this, agents maintain a level of intensity of desire, analogous to a human emotion. The more intense the sense of desire the greater priority with which a goal is sought out, this intention must also be satisfied in the agent's current belief state. As an agent acts the belief state is updated, and so are the intensities of its desires, which in turn update the set of intentions held by the agent. This can be formalized as a "Hybrid POMDP-BDI architecture", a cognitive agent which is able to not only plan over a number of different goals but also react and adapt within stochastic environments [20]. This combined architecture represents a novel approach to adaptive agent learning, leveraging the strengths of both of the individual architectures.

3 RESEARCH QUESTIONS

This research project seeks to answer the following:

What is the efficacy of utilizing a combined BDI-POMDP agent, when planning and acting in a stochastic and noisy environment, when compared to the performance of agents using either a BDI agent or POMDP architecture?

This problem can be broken down into four smaller research questions.

(1) **Can an hybrid BDI agent be implemented with the notion of probabilistic actions and planning, with performance comparable to the existing reinforcement learning agent in QCog?**
(2) **What benefits are there to utilizing a POMDP based model learning framework in comparison to an MDP based framework?**
(3) **What new metrics can be developed to evaluate the performance of a cognitive agent given various architectures and learning mechanisms, when used in conjunction with currently accepted testing measures?**

(4) **How does a cognitive agent created through the combination of a BDI and POMDP agent architecture that leverages the strengths of each respective architecture perform, in comparison to the individual architectures themselves and the existing reinforcement learning agent in the QCog architecture.**

This project aims to develop and evaluate the performance of a combined BDI-POMDP agent in a stochastic environment. We then contrast this performance to the performance of a standalone BDI agent, a POMDP based agent and the existing reinforcement learning agent in QCog, all evaluated within the same testing environment.

4 PROCEDURES & METHODS

Development of this project is undertaken in two distinct phases. The division between phase one and two is between what is considered the minimum viable product which also merits a postgraduate honours project and an implementation of a novel architectural design achieved through the combination of the separate parts into one complete system. This is done in an effort to ensure all members of the team are able to work in parallel with minimal dependencies on one another's work.

4.1 Phase One

Phase one is designed to produce three independent deliverables. Each member of the team is responsible for a specific deliverable and each one of these has been chosen so that it is considered the minimum viable product to showcase and merit a complete honours project as it stands.

4.1.1 ***Yusri*** *- A Hybrid BDI agent with probabilistic notions, knowledge representation and reasoning. This agent will be integrated within the QCog architecture and evaluated against the existing reinforcement learning agent currently implemented in the QCog.*

4.1.2 ***William*** *- A fully functional testing scenario implemented in the QCog architecture and research results relating to both the chosen metrics for evaluation, and the areas of improvement or extension that could be made to the current QCog architecture.*

4.1.3 ***Jonah*** *- A POMDP based learning Agent, the agent will attempt to learn a POMDP and use the learned model for planning. The agent will be integrated within the QCog architecture and evaluated against the existing reinforcement learning agent currently implemented in the QCog which is MDP based.*

4.2 Phase Two

Phase two culminates in the combination of the BDI and POMDP agent architectures, resulting in a more sophisticated cognitive agent which leverages the strengths of each of the constituent architectures. This subsequent agent is then evaluated against each of the individual agents as well as the existing reinforcement learning agent within QCog.

4.2.1 ***Yusri & Jonah*** *- The implementation of a cognitive agent using the combined BDI-POMDP agent architecture. This agent will be integrated within the QCog architecture and evaluated against the existing reinforcement learning agent implemented in the QCog architecture as well as the individual agent architectures created during phase one.*

4.2.2 ***William*** *- Additional implementation of testing scenarios in the QCog architecture, and evaluating the BDI and POMDP agents created in phase 1 with the results being used to further improve the current testbed.*

4.3 System Prototypes

The end product of this research project is to produce a cognitive agent prototype formed from the combination of the BDI agent and POMDP architectures as well as a robust and comprehensive testing environment for evaluating the performance of different cognitive agents.

Throughout the development process three separate agent prototypes will be developed namely; a hybrid BDI agent with probabilistic extensions, a POMDP agent and a combined BDI-POMDP agent.

4.4 Design Features

This project will produce three adaptive agent prototypes, using two well researched agent architectures namely: the BDI agent architecture and POMDP framework. We seek to produce a hybrid BDI agent with the extension of probabilistic notions, knowledge representation and reasoning as well as a POMDP agent with online planning. Inspired by current novel research we seek to combine these agent architectures to form a third cognitive agent which leverages on each of their respective strengths. The existing testbed will also be evaluated and extended where necessary to facilitate a more robust and comprehensive evaluation and comparison system for the performance of various cognitive agents with different learning strategies.

4.5 Constraints

- The prototype agents must be implemented using the QCog architecture.[12]
- The prototype agents produced must perform at a minimum, the same as the existing QCog reinforcement learning agent.

4.6 Development Platform

The entire system and all prototypes will be integrated into the QCog architecture. The QCog architecture is built on the Unity3D game engine, thus scripting for the testing environment will be completed primarily in C#. The development of the agents however will be implemented primarily using Java.

4.7 Evaluating the system

The QCog architecture provides a testing and evaluation environment for agent performance. However one aspect of this project is to evaluate and extend the system where necessary. The resulting agent prototypes from phase one will be evaluated using this testbed. At the completion of phase two the resultant combined agent prototype and the individual agents will be evaluated within

a more refined and comprehensive test environment developed by extending and enriching the current testing environment.

The performance of each of the different cognitive agent architectures are to be evaluated, with respect to their ability to adapt to changes in its environment while aiming to complete a given task.

5 ETHICAL, PROFESSIONAL & LEGAL ISSUES

This research project has one possible issue, the management and use of the intellectual property of the QCog architecture, which is currently part of unpublished M.SC dissertation of Michael Waltham[12].

6 ANTICIPATED OUTCOMES

6.1 System

The main result expected from the system is to showcase the performance and possible benefits of combining the BDI agent architecture with the POMDP framework to create a more sophisticated cognitive agent capable of planning and decision making in noisy and non-deterministic environments. Specifically comparing the performance of the combined agent architecture to that of agents using individual architectures within the same environment.

6.1.1 ***Design Challenges***. There are three major design challenges to consider in this project.

(1) The extension of a BDI agent to include probabilistic notions
(2) How a BDI agent architecture and POMDP framework can be combined to form a single agent
(3) Deciding on a collection of testing metrics and ensuring that they remain independent from the testing environment as best as possible while giving an effective performance evaluation of a cognitive agent

6.2 Measures of Success

The project will be considered a success if the resulting cognitive agent prototypes are shown to equally if not better than the existing reinforcement learning agent in QCog.

6.3 Impacts of Work

Through the completion of this project we will evaluate and extend the current QCog architecture furthering its development and testing capabilities , while forwarding research into the creation of cognitive agents utilizing the combined BDI-POMDP architecture.

7 PROJECT PLAN

7.1 Risks

7.1.1 Integration failure. There is a risk of failing to integrate the BDI architecture and POMDP framework into one cohesive architecture for the agent. The concept is novel and given that this is a new area of research there could be unforeseen circumstances possibly leading to failure.

7.1.2 Loss of Team Member. If a project member is unable to complete their area of work due to unforeseen circumstances. The minimal viable postgraduate honours project - phase one- that has been discussed in this document can be completed individually which mitigates the effect of this issue. The completion of phase 2 is based on components created in phase one.

7.1.3 Equipment failure. A computer or laptop used by a team member fails. This could result in the loss of work and data. This risk can be managed by backing up data on cloud hosting providers and cloud source control systems.

7.1.4 Redesign and Component Re-evaluation. During the research and implementation phase certain components may be determined to be unfeasible or unsuitable. Resulting in time wasted to accommodate additional research and implementation. This risk is managed through the inclusion of additional slack time in the estimation of tasks which can be absorbed to accommodate the increased implementation time if necessary.

7.2 Timeline

See Gantt chart in Appendix A for full breakdown of start and end dates of individual tasks.

7.2.1 ***Project Start***. 1 July 2017

7.2.2 ***Phase 1 Deliverable***. Final deliverable of independent modules due 22 August 2017

- Hybrid BDI agent
- POMDP Agent
- Additional testing metrics and scenario implemented in the test environment

7.2.3 ***Phase 2***. Final deliverable of combined BDI-POMDP agent due 17 September 2017

- Combined BDI-POMDP cognitive agent
- Refined and extended test environment with additional testing scenarios

7.2.4 ***Project End***. 23 October 2017

7.3 Deliverables

7.3.1 Probabilistic BDI Agent. An implementation of an adaptive agent designed using the BDI architecture, with the extensions of probabilistic notions, knowledge representation (i.e P-SHIQ & P-SHIN) and probabilistic reasoning through the use of the PRONTO reasoner. This agent will be integrated and operate within the constraints of the QCog architecture.

7.3.2 ***POMDP*** *Agent.* A **POMDP** planning agent implemented in *C#* that operates within the constraints of the QCog architecture. The agent must perform at least comparatively to the existing **MDP** based planning component in QCog. In addition, the agent must perform it's planning online. The agent can engage in model free or model based learning. There can also be an explicit **POMDP** specified for the environment. The agent will implement a variant of Monte Carlo based search to evaluate the **POMDP** if it uses model-based learning.

7.3.3 A collection of testing metrics, scenarios and a further extended QCog architecture. A collection of existing and newly created metrics that can be used to evaluate the performance of a cognitive agent. Using the QCog architecture[12] a collection of test scenarios will be implemented incorporating this collection of metrics

within a complex and unpredictable 3D world. This will allow for an overall evaluation of agent performance based on a number of key areas tested through the different scenarios. The QCog architecture will also be further improved and extended based on the results of testing the various cognitive agents within QCog and the observations made during phase one of the project.

7.4 Milestones

These dates are fixed milestone dates not subject to change.

- **12 June:** Project Proposals Presentations
- **30 June:** Final Project Proposals Submissions
- **06 July:** Start Project Website for documentation & planning purposes
- **24 July:** Project Paper Plan/Scaffold
- **14 August:** Initial Software Feasibility Demo
- **15 August:** First Implementation, Performance Test & Writeup
- **05 September:** Outline of Final paper
- **11 September:** Project Weighting Decision
- **12 September:** Final Draft of Paper
- **22 September:** Final Submission of Paper
- **02 October:** Final Code Submission
- **03 October:** Final Project Demonstration
- **09 October:** Project Poster
- **12 October:** Final Project Website
- **23 October:** Reflection Paper

7.5 Required Resources

7.5.1 Unity3D. The Unity3D Game Engine is required to run the QCog architecture[12]. The personal edition of Unity3D is available freely from their website[7] and has all the necessary tools for the possible improvement and extension of the current QCog architecture.

7.5.2 Pronto Probabilistic Reasoner. A Java library utilized for reasoning with probabilistic knowledge representation. The source code can be found on their GitHub page.
https://github.com/klinovp/pronto

7.5.3 Computational Requirements. CPU: 3.0 GHz dual core or better
RAM: 4 GB
Operating Systems: Windows 7 or later, macOS 10.11 or later
Video Card: DirectX 9 compatible with 512 MB video RAM or better (NVIDIA GeForce 210/ ATI Radeon HD 5470)
Sound Card: Yes

7.6 Work Allocation

Work is distributed amongst the three team members, for phase one each team member is responsible for the design and implementation of their individual modules relating to the entire system as a whole. The project is divided in such a manner that phase one is completely decoupled and independent for each team member and they will produce their own individual deliverable. Phase two represents the combination of the individual agent architectures to create a hybrid cognitive agent and then evaluating its performance.

7.6.1 ***Yusri.***

- Design and implement a BDI Agent with probabilistic notions, knowledge representation and reasoning
- Combine initial agent design with Jonah to create a combined BDI-POMDP agent

7.6.2 ***William.***

- Research into existing and new metrics for evaluating cognitive agent performance and the design of a testing scenario within the QCog architecture to provide evaluative feedback for both cognitive agents designed by Yusri and Jonah.
- Develop additional testing scenarios for more in depth performance evaluation and extending and refining the QCog architecture. Return evaluative feedback on the combined BDI-POMDP agent.

7.6.3 ***Jonah.***

- Design and implement a POMDP Agent, extending the currently implemented MDP agent in QCog
- Combine initial agent design with Yusri to create a combined BDI-POMDP agent

B APPENDIX B

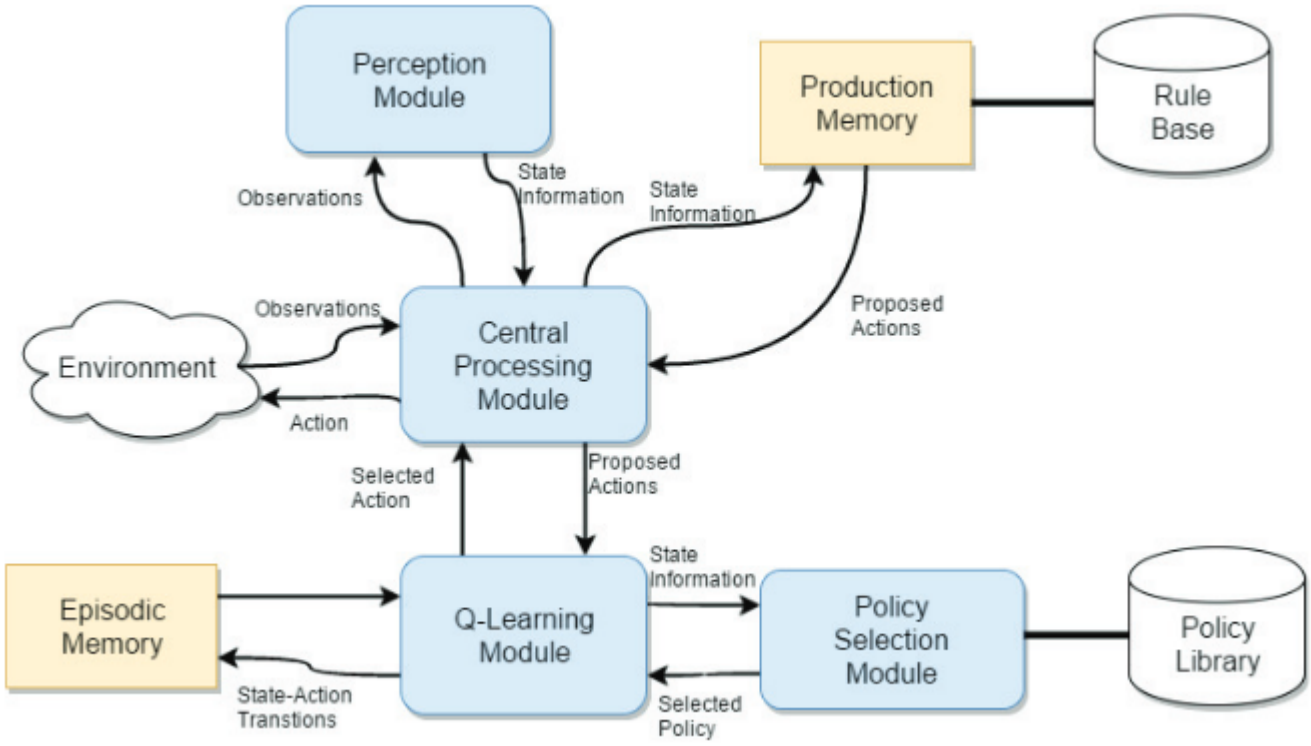

Figure 1: Architecture Diagram of Proposed QCog Agent Architecture

[12]

REFERENCES

[1] Rogelio Adobbati, Andrew N. Marshall, Andrew Scholer, Sheila Tejada, Gal A. Kaminka, Steven Schaffer, and Chris Sollitto. 2001. Gamebots: A 3d virtual world test-bed for multi-agent research. In *Proceedings of the second international workshop on Infrastructure for Agents, MAS, and Scalable MAS*, Vol. 5. Montreal, Canada.

[2] K. Bauters, K. McAreavey, J. Hong, Y. Chen, W. Liu, L. i. Godo, and C. Sierra. 2016. *Probabilistic Planning in AgentSpeak using the POMDP framework*. Springer International Publishing. http://eprints.uwe.ac.uk/31001/

[3] Marc G. Bellemare, Yavar Naddaf, Joel Veness, and Michael Bowling. 2013. The Arcade Learning Environment: An evaluation platform for general agents. *J.Artif.Intell.Res.(JAIR)* 47 (2013), 253–279.

[4] Chris Brown, Peter Barnum, Dave Costello, George Ferguson, Bo Hu, and Mike Van Wie. 2004. Quake ii as a robotic and multi-agent platform. *Robotics and Vision Technical Reports,[Digital Repository],(2004 Oct.), Available at HTTP: http://hdl.handle.net/1802/1042* (2004).

[5] Paul R. Cohen, Michael L. Greenberg, David M. Hart, and Adele E. Howe. 1989. Trial by fire: Understanding the design requirements for agents in complex environments. *AI magazine* 10, 3 (1989), 32.

[6] Paul R. Cohen, Michael L. Greenberg, David M. Hart, and Adele E. Howe. 1990. Real-time problem solving in the Phoenix environment. (1990).

[7] Unity Game Engine. Unity Game Engine-Official Site. *Online][Cited: October 9, 2008.] http://unity3d. com* (????), 1534–4320.

[8] Matthew Johnson, Katja Hofmann, Tim Hutton, and David Bignell. 2016. The malmo platform for artificial intelligence experimentation. In *International joint conference on artificial intelligence (IJCAI)*. 4246.

[9] D. Kinny and M. George. 1991. Commitment and Effectiveness of Situated Agents. In *Proceedings of the twelfth international joint conference on artificial intelligence (IJCAI-91)*. 82–88.

[10] David Kinny, Michael Georgeff, and James Hendler. 1992. Experiments in optimal sensing for situated agents. In *Proceedings of the Second Pacific Rim International Conference on Artificial Intelligence*. 1176–1182.

[11] Michael Lees. 2002. A history of the Tileworld agent testbed. *School of Computer Science and Information Technology, University of Nottingham, Nottingham* (2002), 2002–2001.

[12] Waltham Michael. Design and implementation of the Q-Cog Architecture. Unpublished manuscript. (????).

[13] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. 2013. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602* (2013).

[14] Christopher Moriarty. 2007. Learning Human Behavior from Observation for Gaming Applications. (2007).

[15] Ranjit Nair and Milind Tambe. 2005. Hybrid BDI-POMDP Framework for Multi-agent Teaming. *J. Artif. Intell. Res.(JAIR)* 23 (2005), 367–420.

[16] Praveen PARUCHURI, Emma Bowring, Ranjit Nair, Jonathan Pearce, Nathan Schurr, Milind Tambe, and Pradeep VARAKANTHAM. 2006. Multiagent teamwork: hybrid approaches. (2006).

[17] Diego Rodrigues Pereira, Luciano Vargas Gonçalves, Graçaliz Pereira Dimuro, and ACR Costa. 2008. Constructing BDI plans from optimal POMDP policies, with an application to agentspeak programming. In *Proc. of Conf. Latinoamerica de Informática, CLEI*, Vol. 8. 240–249.

[18] Martha E. Pollack and Marc Ringuette. 1990. Introducing the Tileworld: Experimentally evaluating agent architectures. In *AAAI*, Vol. 90. 183.

[19] Anand S Rao, Michael P Georgeff, and others. 1995. BDI agents: From theory to practice.. In *ICMAS*, Vol. 95. 312–319.

[20] Gavin Rens and Deshendran Moodley. 2016. A hybrid POMDP-BDI agent architecture with online stochastic planning and plan caching. *Cognitive Systems Research* 38, 22 (Nov 2016), 185. https://doi.org/10.1016/j.clinmicnews.2016.10.005

[21] Gavin B. Rens, Alexander Ferrein, and Etienne van der Poel. 2010. A belief-desire-intention architechture with a logic-based planner for agents in stochastic domains. (Aug 18, 2010). http://hdl.handle.net/10500/3517

[22] Stuart Rusell and Peter Norvig. 2003. *Artificial Intelligence: A Modern Approach* (third ed.). Prentice Hall. ID: Rusell-Norvig03.

[23] Stuart Russell and Eric Wefald. 1991. Principles of metareasoning. *Artificial Intelligence* 49, 1-3 (1991), 361–395.

[24] Martijn Schut, Michael Wooldridge, and Simon Parsons. 2004. The theory and practice of intention reconsideration. *Journal of Experimental and Theoretical Artificial Intelligence* 16, 4 (Oct 1, 2004), 261–293. https://doi.org/10.1080/09528130412331309277

[25] Hiroto Udagawa, Tarun Narasimhan, and Shim-Young Lee. 2016. Fighting Zombies in Minecraft With Deep Reinforcement Learning. (2016).