A Happier Life Through Sad Mode - Designing Automated Players for Single Player Games

Saili Chola

1170448

Master of Arts by Creative Practice in the field of Digital Arts

Supervised by: Kirsten Du Preez and Kieran Reid

June 2023

Waiver Letter Number: H21 10 05

A dissertation submitted to the Wits School of Arts, University of the Witwatersrand, Johannesburg, in fulfilment of the requirements for the degree of Master of Arts

University of the Witwatersrand, Johannesburg

School of Humanities

SENATE PLAGIARISM POLICY

Declaration by Students

I, Saili Chola (Student number: 1170448), am a student registered for Master in Digital Arts by Creative Practice in the years 2021-2022. I hereby declare the following:

- **I am aware that plagiarism (the use of someone else's work without their permission and/or without acknowledging the original source) is wrong.**
- **I confirm that ALL the work submitted for assessment for the above course is my own unaided work except where I have explicitly indicated otherwise.**
- **I have followed the required conventions in referencing the thoughts and ideas of others.**
- **I understand that the University of the Witwatersrand may take disciplinary action against me if there is a belief that this in not my own unaided work or that I have failed to acknowledge the source of the ideas or words in my writing.**

Signature: _____  **_____ Date: 5th June 2023**

Contents Page

List of Figures

Acknowledgements

I want to thank my loving family for seeing me through this expansive journey. It wasn't easy, and I was a husk of a human for the better part of two years, but I've finally finished. Thank you to both of my supervisors for being detailed with their feedback and critique but for doing so respectfully and challenging me to be better than I realised I could be. To my closest and longest-standing friend Siya for making sense of the ocean of emotions. To my newest and oldest standing friends for giving me constant assurance for the path I'm on appreciated infinitely. I'm grateful for all that have touched my life and co-founded what I'm able to do today and in the future. The rest is confetti.

Abstract

Solo games are a keystone of tabletop board gaming for players and designers alike. While they are numerous and enjoyed by many members of the community, there is a noticeable lack of clarity and exploration of what principles make these games uniquely interactive and enjoyable experiences for players. This project responds to this inadequacy through the development of a playable game and a research report. The game demonstrates and tests the virtues of solo game play mechanics while the report expands and discusses the interpretable results and qualities of said solo game mechanics.

Preface

This project was developed in two halves. The first half is a playable game and the second is the final report you are currently reading. My work is organised in this format through the definition of a Research through Design methodological approach. The purpose of this approach being to link the practical components of playing a game with the documented info as forms of understanding a topic. As such the game provide an interactable object that demonstrate as well as substantiates the concepts I am investigating.

I suggest you interact with the project in the following way:

The first two chapters should be read in full to establish the context of this project. Chapter one provides the reader with the motivation for this project's endeavours, the goals it is aiming for and how it plans to do so. Chapter two then expands on what information is necessary to understand the nature and choices that inform the discussion of the creative component of the project.

Chapter three is the moment where a split interaction with the project is encouraged. The chapter is comprised of an overview followed by three sections of discussion of each iteration. Each section should be read and then the corresponding iteration of the game should be played.

To access the playable game, you will need to own *Tabletop Simulator* (2015) on steam and download the three iterations from the steam workshop links provided.

It is not necessary to play the game to understand the report, nonetheless they cannot be separated from each other. You will undoubtedly be able to understand all major concepts and talking points about this project but contained within the creative component and its accompanying appendices are a level of nuance, insight, and volume of information far beyond what this report can capture.

Tabletop Simulator:

https://store.steampowered.com/app/286160/Tabletop_Simulator/

Iteration one:

https://steamcommunity.com/sharedfiles/filedetails/?id=2925044571

Iteration two:

https://steamcommunity.com/sharedfiles/filedetails/?id=2925046805

Iteration three:

https://steamcommunity.com/sharedfiles/filedetails/?id=2925047862

Chapter 1 - Link Start

"The illusion of reality lies not in the machinery itself, but in the users' willingness to treat the manifestations of their imaginings as if they were real."
— Raph Koster, Postmortems: Selected Essays Volume One, 385

Introduction

In my time studying the craft, artistry, and design of tabletop games, very few game genres have captured my interest quite like solo board games. I find solo games fascinating because of how dramatically the gameplay experience changes based on the simple adjustment of included players. In a solo game setting, you are no longer playing against your friends or an unknown competitor. Instead, you are playing against a designed system, or more curiously yourself (Engelstein and Shalev 12).

In an exclusively analog context, solo games contend with the task of structuring[1] play and its mechanics[2] into an enjoyable experience for players. Many solo games encounter this position when they are developed from multiplayer game mechanics. The difficulty of this task lies in adapting multiplayer mechanics for a solo game context that remains true to its base game while also being satisfying for its new context.

The complexity of this task lies in organising play into a coherent objective, series of challenges and sequence of difficulty. The solution that solo games trend towards has been a collection of game mechanics and rules defined as an artificially intelligent (AI) system (Engelstein and Shalev 12). Despite this solution being so prevalently in use professionally, the theoretical backing of why or how solo-game AI functions or contains valuable methods of creating enjoyment for players is not well-defined. As I will venture to explain, AI is merely the "machinery" of playing a solo game (Koster 385). That is AI are the moving technical part of the game experience, but their mere inclusion does not translate to an inherently fun experience. In this sense, the enjoyment of solo games lies in how the machinery makes its actions purposeful to what players believe is enjoyable in the game environment. This is a complex idea so let us start by explaining the ephemeral term of AI in game design.

As it originates in the field of computer science, AI systems capably "perceive, understand, predict, and manipulate a world far larger and more complicated than itself" (Russell and Norvig 1). In simpler terms, it is also understood that AI can act independently and or "autonomously" in their given environments (18). In the field of analog solo game design, however, AI describes a system that players "play against" and is not actually "sentient" (Engelstein and Shalev 12). Solo game AI noticeably act with a static predesigned purpose and understanding of the game environment. Through this method of interaction solo game AI can facilitate gameplay for players despite not

[1] The structure of a game describes its foundational win conditions and or orientation of play. For instance, a game experience could be built around a specific player winning or losing and played over three successive play sessions (Engelstein and Shalev 1).
[2] Mechanics in game design theory describe a variety of highly debated concepts each with their own set of nuances but for this paper we will describe mechanic as "a single action" or "compound activity composed of a suite of actions" (Salen and Zimmerman 290).

being inherently intelligent or malleable towards their environment as described in computer science.

The disparity in definitions highlights how the game design field values the play experience[3] of players. It is through the interactions that a player has with the system that a game is viewed as successful or lacking. To understand solo games, it is thus important to define and explore the role AI systems have in creating play experiences for players.

From my own personal gaming experiences, *Onirim* (2010), *Root* (2018), and *Scythe* (2016) have had the strongest impression on me through their solo game AI systems. The ingenuity behind the design of these games lies in how their AI systems interact with the player. *Onirim* is a deck-based solitaire card game where there is no direct opposition that the system poses to the player. Instead, the play experience is arranged by encountering obstacles, mechanics, and rules in predesigned, but organically occurring order for the player to win or, in most cases, lose. Additionally, the play experience can be customised to the player's preference or skill level through various expansions[4]. *Scythe* and *Root,* on the other hand, are territory-based war games with an AI system that is physically present and directly opposing the player. As a necessity for the game to function, they require players to manage opposing player turns. The biggest difference between *Scythe*, *Root* and *Onirim* is the management responsibilities they must undertake.

In both games, there is an overarching design goal for the system. Looking at various rulesets and developer reports, *Root* and *Scythe* emphasise mechanics mimicking real players (Britt 7). Through this mimicry, they seek to capture their multiplayer counterparts' spirit and complex gameplay loops (Engelstein and Shalev 13). *Onirim*, on the other hand, leans into being a system with an abstract presence in the game space. Through this, its functionality can depend on mechanics and systems that deepen the solo player experience instead of posing as a direct opponent (Ibid). Through these three games we have two methods of creating a play experience. There is no right or wrong method for achieving play for each. Both games are well beloved by me and many people in the community. This instead highlights those solo games appeal to players for different reasons. Dale Leorke, a games researcher, and consultant, established insightful statistics about player experiences with solo games through the paper *Solo Board Gaming: An Analysis on Player Motivations* (2018) that we can use to further define this idea.

One of the reasons solo games have risen to popularity is due to the player base "embracing" the unique qualities they provide as a game experience (Leorke 3). Leorke identified that these qualities can be defined as player motivations (Ibid). Social motivators describe how solo games can have a relaxed state of social responsibility or obligations (18). Genre motivators describe the medium's unique eccentricities, game design exploration or analogue tactility (19). Finally, playstyle motivators describe immersive, experimental, or uncommitted methods of playing solo games (Ibid). These reasons explain most of what players seek from their solo game experiences. More importantly, is that these motivations specify what solo game mechanics can be considered to create moment-to-moment enjoyment and long-term engaging experiences for players.

[3] I am defining experience generally as a dynamic of playing. The game can be exciting, frustrating, strategic, a space for learning, a space for testing ideas and so on.

[4] Expansions for board games provide additional content for players in the form of mechanics or a game play environment.

With all that said, this project revolves around investigating AI mechanics in solo games and their interaction with players. My aim is to explore how agent system design creates enjoyment and engagement for solo games.

The Project

Exploration can be useful so long as it is guided and or measured by objectives and expectations. My methodology, which will be discussed in detail further on, stood as the basis of defining these objectives and expectations to successfully work through my aim. My main objective is to craft a game that demonstrates what design techniques enable enjoyable and engaging play experiences for solo players.

Over seven months, I took on the task of adapting the multiplayer version of *Root* (2018) into a solo game. The game I designed sought to unify principles of solo game design found throughout my research. These principles covered the areas of solo game design, enjoyment theory and AI design practices. From this research, I designed three iterations of a solo mode for *Root*. These iterations created a context in which many solo game mechanics I personally defined as valuable for solo play could exist, be testable and verifiable by myself and others. To conclude what these playable iterations displayed I took to documenting and summarising my findings into this research report and accompanying documents[5].

Root: A Tale of Woodland Might and Right

Root is a two to four-player board game designed by Cole Werhle and published in 2018 by Leder Games. The gameplay revolves around woodland creatures seizing control of a forest through cutthroat, politically referential area control gameplay, strategy, and tactics. Each player plays a single faction out of a possibility of four factions. Each faction is designed asymmetrically; thus, how they interact with and win the game differs. The Marquise de Cat, for instance, uses the playstyle of engine building which can be defined as “getting together a few elements that create a virtuous cycle of increasing productivity” (Engelstein and Shalev 438). Beyond their playstyle, each faction’s asymmetry spans a unique point-scoring method, unit choice and deployment, pre-game setup and card/ability usage. To keep the game consistent between players, there are still shared components of their victory conditions, resource usage/access and phases/performable play actions.

[5] This document is quite focused on discussing solo game design theory and concepts as much as possible. That said sometimes we will not be able to talk about solo game design theory without fundamental game design theory. Much of this document thus uses the work of American game designers and researchers Katie Salen and Eric Zimmerman and their seminal book Rules of Play: Game Design Fundamentals (2003) as a backbone of supporting game design theory.

Figure 1 - Image of the game board of Root and its various pieces in Tabletop Simulator

The game plays through twelve board locations, defined as clearings. Each clearing has a range of one to three pathways that connect them to other clearings. Clearings also have slots where structures for each faction can be constructed. Each player goes through three phases of gameplay on their turn. These phases, with some variance between factions, allow the player to act towards winning the game. Actions range from interacting with other players, the board, the game state, or scoring points to progress towards the victory condition. The game ends as soon as a player gains thirty victory points.

Root has issued and updated many expansions since its original release in 2018. As of 2022, the game boasts ten game expansions that offer either an update or additional game mechanics to the core game experience. Additionally, there is also a digital version of the game. I have played through the majority of the expansions to inform my design process.

My Designs: Iterations 1 to 3

Using *Root* as the basis for my designs was essential to realise this project because it offered me three advantages. Firstly, I was not starting from scratch. I did not have the added burden of verifying whether an original game I designed would be fundamentally engaging and functionally playable. The game I was building from was an already proven and engaging game experience. Making a game from the ground up would have emphasised development and taken away time from what would be iterating and analysing solo systems in detail. Secondly, *Root* already had a trove of fans, opinions, casual and competitive playthroughs and finally community developed projects[6]. The variety of sources provides a large amount of information that can be parsed to draw ideas and design considerations for solo mechanics. Thirdly, *Root* already has an ideal gameplay experience through its multiplayer design. This unique position means that developed ideas can be measured against the intended experience for players or, even better, an expected feeling that players have for specific mechanics. There will likely be some variance in how the solo or multiplayer version of a game establishes player engagement, but regardless the blueprint for the experience the game should capture through its mechanics is readily visible.

[6] *The Root: Clockwork Expansion* is based on the work of the community member Benjamin Schmauss who developed the *Build a Better Bot* project. In that project Benjamin approached core issues the community felt existed in the original *Root* solo game mode. Leder games eventually took his design on that would later be what we now know as the Clockwork Expansion.

Root also already contains a solo mode that comes with the original game. The solo mode we shall henceforth refer to will be the revised versions present in *Root:* The *Clockwork Expansion* (2020). The AI system revolves around a rote and easily interpretable manner of playing the game to fill out vacant opposing roles in the game environment. If you have two players, the AI system can take on the roles of the third or fourth for example.

Each of the base factions are represented within this system and generally achieve a semblance of the original faction through the actions they perform on their turn. On their turn, a human player will be tasked with following a set of instructions that describe all the actions the AI system performs. The AI systems function off an environment that can be generally interpreted by way of markers and rigidly predefined action sequences.

The biggest difference between an AI and the player-controlled faction is the way each turn plays out. The AI system gains points each turn until a victor is declared while the player gains victory points through the original game mechanics of *Root*. In most cases, the experience of playing is like racing the AI system to reach the victory point total which contrasts heavily with the idea of out-strategizing your opponent from the multiplayer version. The AI system is at times also freely granted actions, units, or buildings instead of having to pay a game-related cost of actions or resources. The AI can be customised with a variety of game-related stat or game logic changes to increase the difficulty it presents to players. My analysis suggests this is both a method of keeping the system competitively active in the game and simplifying the requirements of operation.

In analysing the changes between the original solo and the expansion solo system, I found a lot of clarity on the player experience. If I had questions about the intent of a particular game design decision, the other game would provide the answer. Following suit, I rationalised that my project could also document each significant milestone achievement to show the divergence or evolution of my conclusions. This amounted to three design iterations that I classify holistically as the creative component of my work.

Iteration one deconstructs and then reconstructs the AI system of *Root*. It takes the main mechanics of *Root* and organises them into executable rules for the AI to function and facilitate playing against an opponent loosely. **Iteration two** focuses on refining the mechanics of iteration one. It does this by finely considering the effects of all AI interactions, direct or indirect, in the game. This is a shift from a functional game system to one that maximises mechanics and provides players with more engagement. **Iteration three** then took to implementing experimental/unstable design ideas. These ideas would typically be more convenient for players to interact with. The rationalisation, however, is that these experimental mechanics produce “new information” from the resultant failed or stressed interactions that are present (Salen and Zimmerman 2).

More than my own opinion, it is helpful to interpret the impression of my designed mechanics. Another pursuit of this project was establishing a dedicated group of people to test my iterations alongside me. The groups ranged from scholars and designers with design and interaction in their field of study and communities that regularly play *Root* amongst other board games. I gathered playtest reports from these groups that could expand and interrogate their experience playing individual iterations. The collected information would serve as insight into how the interactions and engagements of how the game functions separate from my intended outcomes.

Towards a Research Methodology

As the computer science and game academic Espen Aarseth describes, games are artefacts to be understood “through play” (7). That might seem obvious, but the more significant lesson behind this assertion is that the entire experience of a game is realised when it is being played first-hand. This approach prevents “severe misunderstandings” of the intended game experience or inner workings of the game mechanics (3). From this perspective, a solo game that could be played and in turn reviewed by analysts would reveal more information about the topic of solo game mechanics than an approach that simply theorised and conceptualised principles around the nature of solo games.

Aarseth suggests that “Scholars, gamers, critics and developers all have different needs and needs for different methods”, meaning that the methodological process should be chosen according to the position of the research inquiry and the type of information needed from it (Aarseth 6). Another part of this statement is that the established role of the analyst informs what information we can get from a research inquiry. As such, I would like to clarify my two roles for this project.

The first role is that of a game designer. The baseline technical description of a designer is an individual, group or culture that is a design practitioner. A core philosophy in design is that “in order to design, you must practice design” (Brathwaite and Schreiber 11). In the context of this project, the value of a game design idea does not depend on previous design experience or ability but that there is value in that experience. My experience thus guides which ideas this project endeavours to explore. Beyond that bias we can affirm that game design at its core is creating “rules and structures that result in an experience for players“ (Salen and Zimmerman 26-27). The tasks involved in design revolve around conceptualisation, implementation, and documentation. Conceptualisation speaks to identifying necessary spaces of emotional or functional player experience in a game and then proposing “design decisions” in the form of new mechanics or rules (Salen and Zimmerman 9).

Implementation is then including these solutions into the game as is most viable given the development context the game exists in. A design decision that has a time frame of two weeks, as opposed to two months, will have different implementation solutions. Documentation is the paper trail of design decisions and, more finely, the statistics of the game experience as it evolves between decisions. According to industry accomplished game designers Brenda Braithwaite and Ian Schreiber, documentation allows a “memory” of changes to be collected and for the designer to provide “communication” in the form of accessibility of their ideas with anyone that should be involved or curious about the design of the game (Salen and Zimmerman 4).

Even though these are formally defined terms, design work is less tangible than it seems. Design is a process driven as much by playing games as it is by crafting and writing design decisions down. As Salen and Zimmerman state:

> “it is not possible to fully anticipate play in advance. It is never possible to completely predict the experience of a game. Is the game accomplishing its design goals? Do the players understand what they are supposed to be doing? Are they having fun? Do they want to play again? These questions can never be answered by writing a design document or crafting a set of game rules and materials. They can only be answered by way of play. Through the iterative design process, the game designer becomes a game player and the act of play becomes an act of design.”
> (Salen and Zimmerman 19)

The information a designer garners from playing can directly feed back into the design decisions the game moves forward with. For instance, a game might intend to be a challenging experience but

upon play be relatively simple to play through. A designer thus creates “a context to be encountered by a participant, from which meaning emerges” (Salen and Zimmerman 26-27).

Additionally, meaning is generated through “the way that players interact with the game in order to play it” (Salen and Zimmerman 27). Through meaning, the game’s experience or “context” is formed. A good example of this can be seen through how games suggest player counts but players interpret an optimal number of players for a fun play through. *Root* for instance suggests a player count of 1-4 but a particular player might enjoy their play experience simply as a 1 on 1 or three player game.

To affirm which experiences are meaningful designers employ what is known as “iterative design” practices (Salen and Zimmerman 25). Iterative design describes a “play-based process” of having the game played and, in turn, experienced throughout its development while focusing on “fundamental rules and core mechanics” (Ibid). The participants in this play-based process are the players, who can include the designer but should not exclusively be the designer.

The other role I fit into is that of a researcher which Richard Herriott, the PHD in universal design and researcher on design practices[7] will help us define. As Herriott describes a researcher uses “systematic enquiry into X to discover communicable knowledge” (2). We say the research process is a systematic enquiry because it “seeks to answer some question; is goal-directed because its objects of inquiry are defined by the task description; and is knowledge-directed” (Ibid). This project as such acquires its direction from difficulties observed with playing, define what these difficulties are through research amongst a broad category of sources and then moves towards a solution through testing and review processes. In addition to all these research goals Herriott defines, it is equally important that this information is made to be communicable to a community and that the “findings must be intelligible” (Ibid). Intelligibility explains how the information is well constructed, and organised to be interpreted by others which defines a prominent reason for this research reports length and detail.

The final note about the role and practice of research is for a consistent consideration of bias and objectivity. The academic researcher Ranjit Kumar suggests that, beyond explicit paradigms, lenses or methodologies of research, a researcher must adhere to a “control of bias” and a “maintenance of objectivity” within the research process and outcomes achieved (Kumar 11). Raph Koster[8] argues that bias is a component of game design practice. As he mentions that at certain points in the design phase, designers must act on personal “feelings” and well-considered instincts to make games enjoyable for players (Koster 40). This is not to say that the two roles are incompatible but rather that some flexibility or at the very least precise definitions of the expected outcomes of a research project is necessary. This is where the methodology of Research through Design becomes important to congruently unify these roles.

[7] Herriott disagrees with how prominent Research through Design paper written by John Zimmerman et al posture that the research approach can exclusively reveal a unique perspective about design that other approaches cannot. I will thus describe here and further on the capacity in which I am using Research through Design to ensure my approach is well established.

[8] Taken from the preface of their community and industry recognised book “Raph Koster is a veteran game designer who has been professionally credited in almost every area of the game industry”. They’re quite accomplished as a “top thinker” in the game design field.

Research Through Design

Research through Design (RtD) is a helpful methodology because it combines the strengths of a designer that iterates on a gameplay experience and the strengths of a researcher who systematically and scientifically interrogates a subject. RtD is defined by its acknowledgement and use of the design process as a research inquiry system (Herriott 2-3). The approach emphasises crafting interactable "artifacts" (Zimmerman et al. 1). These artefacts then act as means to relink "weakly" contextualised theoretical models and ideas to a "strongly" contextualised experience through an artefact (Herriott 2). This project will be able to verify whether ideas are valuable and in what ways as opposed to simply proposing this value purely from theory and previous play experiences.

Additionally, RtD is useful in situations where a lack of reliable knowledge is present. The artefact justifies information through how it includes society. Through the participation of society, artefacts create "socially robust knowledge" (Ibid). An artefact enables solo game mechanics to interact with players before they are finalised and are thus strengthened by this interaction and review process. Furthermore, the results of the "inquiry" will only be visible in the artefact and not exclusively through the documentation accompanying it (Zimmerman et al. 2; Herriott 3).

This methodology essentially unifies the roles of designer and researcher into a design researcher in the game design field. The design researcher would "identify opportunities for new" new mechanics that can positively impact the game design field; they perform "problem framing" to highlight theoretical gaps in the field; they undertake "evaluating the performance and effect of the artifact" to "discover unanticipated effects"; they also suggest supportive models, theories, or ideas to bridge the holes in the current theoretical understanding of the topic (Zimmerman et al. 4). All described properties serve to provide insight into the topic of solo game mechanics and are ones that I plan to fully use.

RtD also emphasises the objectives and motivations behind making an interactable artefact. RtD describes shifting the design process from how we are making something to how we could make it. Interactable artefacts produced under RtD are also "novel" (Zimmerman et al. 4). A novel artefact is unique in its occurrence and made beyond a need to recreate an executed idea (Ibid). The purpose of the artefact in this way can be said to "produce knowledge for the research and practice communities" and "not to make a commercially viable product" (7). The decisions that are pursued in creating an artefact can appear unintuitive, unreasonable, and unviable because they exist to become "research artifacts". The research artefact, in this way, creates the "right" gameplay experience as opposed to an artefact that focuses "on making commercially successful" artefacts (Ibid).

In the context of game design, the RtD process is an effort to move games from their "current state" to a "preferred state" of characteristics (Zimmerman et al. 5). RtD highlights and isolates the design decisions in the original and similarly designed artefacts through a "holistic" design approach (6). The holistic approach describes synthesising information from various fields, artefacts and conflicting opinions into a single working system (6). In meeting specifications to make the artefact, RtD then advocates re-evaluating the object by asking questions that probe the artefacts effectiveness, appearance, or presence in the world (Ibid). To reach the "preferred state" for any research artefact, RtD outlines four parameters of process, invention, relevance, and extensibility to ensure a project is valuable as a research contribution (7). The descriptions of each parameter are as follows:

Process: Constant documentation of the evolution of the project to allow reproduction of the product and rationale for the decisions made. (7).

Invention: Displays how the design combines and synthesises various subject matters towards addressing a specific scenario. Emphasis will be placed on situating the designed product amongst existing literature in the research and cultural community (7).

Relevance: an articulation of how this work has benefited the community and how this state has been identified as the preferred state of the work (7). The state has been interpreted as why the work has been chosen and must appear as it is as a finished product.

Extensibility: The knowledge gained from the project should provide space for further exploration or leverage knowledge that can be directly used by the community (7).

With that said we have sufficient premise, objective, and methodology to dive into uncovering the topic of solo games.

The Research Report

The report's role is to document and explain all information gathered around the topic. The objective is to establish relevant literature or concepts for discussing solo games. Additionally, it serves the purpose of presenting my process, observed outcomes, and conclusions about the design of solo game agents and how they create engaging gameplay experiences. The report, as it will be discussed beyond this chapter, works as follows:

Chapter two will be a literature review of what information is relevant towards the topic of relevant values, concepts and mechanics that go towards designing solo game agents. Chapter three presents my three design iterations and summarises the actions and information that emerged from the project. Only the most prominent points of what was intended, what was achieved, and what was observed about the nature of solo game agent design are presented. Chapter four, the final chapter, concludes the observed outcomes of this research project and what can be summarised about solo game design.

Chapter 2 - Building a Fort

Introduction

Aarseth identifies that game design and development are “ergodic” (6). This means that they function and should be perceived as an equal contribution of the individual disciplines that develop them. Similarly, to achieve “thorough game analysis”, a “collective pool of experience” is valuable because it “will always bring new aspects forward” (Ibid). To analyse an aspect of game design and use a singular experience or single source of information is, in many ways, insufficient. Aarseth goes as far as to say that “drawing on the experience generated by others is crucial, not merely useful” (ibid). My literature review consists of three sections of information: A discussion on Root as a network of knowledge production, a brief discussion on solo games and how they function, enjoyment and engagement as applied to player behaviour and finally AI technology as it might be applied to solo game design.

Root as a Network of Knowledge Production

One challenge with researching solo games is the minimal amount of analogue game design theory. An excellent way to compensate for that is through what Aarseth defines as three levels of acquiring experiential knowledge about games.

> “Firstly, we can study the design, rules, and mechanics of the game, insofar as these are available to us, e.g., by talking to the developers of the game. Secondly, we can observe others play, or read their reports and reviews, and hope that their knowledge is representative and their play competent. Thirdly, we can play the game ourselves. While all methods are valid, the third way is clearly the best, especially if combined or reinforced by the other two.”
> (Aarseth 3)

Through this outlined hierarchy of information, I would suggest *Root* has a network of knowledge that can be accessed. This network is established through playable games and the various forms of documentation, review or design that are produced from community interest of the *Root* intellectual property. This network is useful as my own experience isn’t the only source of robust information. Professional players, journalists or hobbyists can all be referenced in some manner to improve the design process of this project.

Design work requires a “critical self-awareness” to maintain the validity of anything claimed when these instances occur (Aarseth 7). On top of being self-aware of myself I can use the information of the network to contrast community ideas with my own to conceptualise solutions and arguments. One solution is to gather supporting material from individuals who have interacted with the source and presenting their own viewpoints. Resources such as online playthroughs, guides, cautionary or enthusiastic reviews, and interviews all form a part of the knowledge network of *Root*. These sources can inform the design process constructively by being separated from my own context and experience of the game. For instance, I am not a competitive *Root* player but interacting with players’ experiences with higher competency provides me with information. It enables more opportunities to iterate how mechanics function within this project instead of mistakenly having to “commit” to a “severe misunderstanding” (1).

Similarly, when concerned with avoiding misunderstandings, there are instances when a game cannot be played, but its rules and mechanics can be studied. In such cases, the direct rulebook or the developer's entries on the subject can be exceptionally insightful. Games used as sources can also be studied from the perspective that they retain a similarity or contrasting mechanical idea, such as *Scythe*. Through these three levels, we thus have a network of information that makes knowledge generation possible and valuable despite a formalised informational source to access.

However, *Root* and these sources can only explain so much through a shared network of experiences. More is required to understand the functional/structural reasoning behind the design of solo games and, by extension, the foundation of enjoyment in games.

Zooming in on Solo Games

To understand *Root* and the design of solo games at large, we must expand on how solo games function as a genre as generally as possible. In the book *Building Blocks of Tabletop Game Design,* Geoffrey Engelstein and Isaac Shalev mention how an essential aspect of solo gaming revolves around "playing against the game itself" (13). As they further explain, the 'game itself' describes how designers might use one of three types of play experiences to motivate playing a solo game.

The first type is "goal-based" play experiences, where the player seeks to complete a larger predefined objective (Engelstein and Shalev 1). *Friday* (2011), a game by Friedemann Friese, has the player survive a dangerous jungle and win by defeating a looming pirate attack. The second is "record based" play experiences which revolve around the mastery of the game's mechanics and subsequently achieving ever higher scores through successive playthroughs (12). *Indian Summer* (2017) by Uwe Rosenberg, a game where players are trying to cover a woodland floor with specifically shaped blocks that represent leaves, demonstrates this by having its solo mode designed to have players combine the highest scoring two of three playthroughs. After playing three times, players combine the two highest scores of their playthroughs to arrive at a high score they can share with others or compare to their last complete play-through of the game. Lastly, solo games use "AI-based" systems that are physical representations of a player that oppose human players throughout a game (13). Games like *Root* and *Scythe* recreate the original multiplayer interactions by introducing AI systems defined as bots or automata, respectively. These systems have access to essentially the same mechanics, objectives, and rule regulations that a regular player would.

Considering the finer characteristics of AI-based play, Engelstein and Shalev have a few more observations to offer. The first is that players distinctly operate AI systems. The player follows algorithmic gameplay instructions that "mimic" the "outputs" of a "human player's choice" (Engelstein and Shalev 13). A common trend of these systems is to include "randomizers" that can vary the choices and presence of the agent in the game (Ibid). This can be an important feature of agent design as it positively affects how players interact with the system by creating "uncertainty about the AI player's choices" (Ibid). This uncertainty removes rote or completely predefined play patterns that a human player would find tedious.

There are no strict rules or guidelines on how AI systems can be designed, as the field of solo games currently exists. Engelstein and Shalev note that most AI is notably "representational" in design (Engelstein and Shalev 13). This term describes how the agent's actions and capabilities in a game mimic a regular player's actions and capability (Ibid). Other game designers such as Morten Monrad Pederson design AI systems according to their own philosophical principles. Penderson's solo game AI are developed under the "automata" framework for instance. Visible in games like *Scythe* automa

function to create an “impression” of the presence of a player during play (Ibid). This is preferred instead of mimicking players as closely as possible.

The value being that automata can function towards the needs of the player experience rather than obeying the expansive or rigid rule structures that a player would have to follow to function throughout play. In scythe for example, the AI system teleports instead of using move actions as the player would. This simplifies the complexity of managing the AI turn while also capturing the general feeling of moving. An automata AI system exemplifies that AI design can mould specific play experiences towards fundamental interactions to create enjoyment for players.

Contrary to this achievement in AI design, I feel there is a detriment to AI systems organised in this manner. From my experience, AI systems that move away from representational mechanics tend to feel unfair or even outright unenjoyable to play against. Throughout play the system will have to make use of mechanics that are intended to maintain competitive edge against the player. The player will however not be able to use the same mechanics and thus feel like there is unfair assistance granted to the player. In the scythe example for instance, the AI may be able to teleport to locations, but the player still must manage their movement properly with extended limitations towards legal move operations.

A similar situation occurs in *Root’s* AI design, where factions gain stats or free actions that ignore the game’s core gameplay logic. Having to work for the same benefits another player can get without the same effort does not feel enjoyable to play against. The enjoyment an AI can provide is also based on its execution and understanding of how its players determine enjoyable mechanics. The value system players inherently refer to when engaging with a game was thus the next topic that needed some clear definition.

Learning is Fun

Players value a game based on how much fun they experience while playing (Koster 40). However, fun as a concept can be nebulous and cover a wide range of emotions for players. Although there is a general connotation of positive emotion, the terms help enjoyment and engagement are concepts in game design theory that explain what fun is and how it is sustained for players.

Raph Koster says, “Fun is all about our brains feeling good” (Ibid). Fun is the enjoyment we feel from the stimuli games provide during play. Stimuli refer to “physical stimuli, aesthetic appreciation,” anything that provides “direct chemical manipulation” within our brains (Ibid). This chemical manipulation he speaks of is more accurately described as a “dopamine” reward our brains provide us for every situation we successfully and positively learn from (Ibid). Within this state of learning is the ability to comprehend, solve and master situational events we encounter (Ibid). Games that allow players to learn new things about themselves, profound ideas, or the world around thus “fun” from this perspective of game design theory (Ibid).

The feeling that opposes fun is then “boredom”. Boredom describes how a game is unenjoyable for players to participate in (Ibid). It is felt by players when a game is unable to present any new “cognitive stimuli”, “data”, or “permutations” of game-related events and prevents the learning aspect of playing a game from continuing (34). In further defining fun, we can mention that fun occurs when players are given “meaningful” actions to perform during play (Salen and Zimmerman 25). As Salen and Zimmerman describe that:

> "Meaningful play in a game emerges from the relationship between player action and system outcome; it is the process by which a player acts within the designed system of a game and the system responds to the action. The meaning of an action in a game resides in the relationship between action and outcome"
> (Salen and Zimmerman 28)

While helpful in describing meaningful play, this extract describes a baseline interaction that can occur in all games. This form of meaningful play is purely "descriptive" (Ibid). Play is enjoyable when actions are "emotionally" and "physiologically" substantial as well as "integrated" and "discernable" for players (50-51). Discernable play describes player actions that have "perceivable" outcomes within the game, while integrated play describes game-related actions with long-term "significance" in the game well after execution (Ibid). Both terms are essential for making play meaningful for players as they provide feedback on what happens when an action is taken and how it affects the rest of the game.

Meaningful play in its different forms also calls attention to time as a component in crafting an enjoyable experience. Games manage the introduction of stimuli to form "patterns" of play (34). Patterns are the perceivable outcomes and rules of the experience they are engaged with—essentially, the establishment of what is possible or conversely impossible within the game.

Pattern based play is useful for solo games as a means of creating fun. Players can learn the bounds of their own skillset or the game environment through the interactions and limits of the AI system's play patterns. The key being to keep congruency between all play patterns possible amongst all players in their environment. This idea calls back to my frustrations with the *Root* AI where the system can gain abilities that cannot be accessed by a regular player. The player's perception of play patterns within the Root game experience is incongruent with what is possible. In such cases the design of the play pattern focuses solely on the AI system instead of what those play patterns mean for the player.

Finally, adding or subtracting variations on patterns of play and stimuli enables a "richly interpretable" play experience for players. However, if stimuli and their subsequent patterns are not carefully managed, they risk pushing players into a state of anxiety or boredom via sensory "overload" or "deprivation" (Koster 42). When an AI system varies its play patterns to create an interesting game environment it also creates a potential game experience that is too difficult or too simple for a player to engage with. It is through the curation of stimuli and learnable patterns throughout a game, players enter a state of optimal engagement, which we will now describe (Juul 8-9).

Player Engagement through the GameFlow model

Engagement is the alternate method of interpreting fun. Engagement speaks towards a player's active participation or "interactivity" with a game (Salen and Zimmerman 49). Engagement from players advances a game's play state and enables a cyclical loop of interaction between the player and the system until the game's conclusion.

There are three interactional modes that a solo game can make use of to keep a player engaged. Cognitive interaction speaks towards the "psychological, emotional, and intellectual participation" interaction between player and system (Salen and Zimmerman 49). For example, in *Root,* a subtly distinct group is known as the small folk are represented as the playing cards players use to

strengthen their faction or perform actions. The small folk represent the working class to which stronger factions nonchalantly use in their game secure power and presence in the forest. The mechanical function and importance of the small folk directly speak towards an explicit political presence in the game (Thurot).

The next form of interaction is functional which describes any physical or virtual interfacing action that occurs (Salen and Zimmerman 50). Interfacing refers to supportive game-related activities that enable the game to be played as opposed to actions that progress and affect the state of the game. Handling a deck of cards, setting up the board for play or managing supportive components like a die are all forms of interfacing within a game. Finally, direct interaction describes "participation" with the games "designed choices" and mechanics inherent in playing (Ibid). Through this description, it should be apparent that being engaged with a game transcends simply acting in a game. The thoughts and perceptions of players can be equally thrilling if a game makes indirect interactions enjoyable for players. If a game can maintain a steady sequence of meaningful interactions, players usually enter what is known as a state of "flow".

Flow is a state of being coined by the late psychology researcher Mihaly Csikszentmihalyi in his book *Flow: The Psychology of Optimal Experience* (1990). The book qualifies why and how people maintain long periods of interaction with their chosen activity. As described by Csikszentmihalyi, the flow state can be "so gratifying" that a person will be "willing to do it for its own sake, with little concern for what they will get out of it" (qtd. In Sweetser and Wyeth 2). This definition captures the hobby of solo play quite well in that there is little to no external value that can be gained from playing. Csikszentmihalyi also asserts that we can understand enjoyment through a "universal" set of eight categories (qtd. In Sweetser and Wyeth 3).

To bring flow theory closer to game design contexts the game researchers Penelope Sweetser and Peta Wyeth revised these eight categories under the title of "GameFlow". GameFlow is more appropriate because of how the context emphasises the experience of maintaining enjoyment with games amongst player relevant contexts and ideas. This is favoured from relating enjoyment to the generalised everyday person experience. Sweetser and Wyeth note their own model as "refined and extended" for game design research contexts as it accounts for the "heuristic literature" of designing games that focus on usability and user experience (3). The eight areas of GameFlow are described as "concentration, challenge, skills, control, clear goals, feedback, immersion, and social" as necessary conditions for players to achieve a flow state (Ibid). Each category can be summarised as the following:

Concentration: Games should require and facilitate concentration. The more attention it requires, the more absorbing and enjoyable it can be (Sweetser and Wyeth 7-8).

Challenge: Games should strive to be appropriately complex for the player to attain goals. Too tricky, and they risk inducing "anxiety" not challenging enough, and they incur apathy (Sweetser and Wyeth 7-8).

Skills: Games must support skills development, emphasising how the experience is taught and eventually mastered (Sweetser and Wyeth 8).

Control: Games should allow players to transfer their intention into an equivalent in-game action. In addition, games should enable players to customise the game to fit their learning or playing style or realise an impact on the game environment or experience (Sweetser and Wyeth 8).

Clear goals: The game should provide clear goals at appropriate times for the player to achieve or gradually build up to an overarching goal (Sweetser and Wyeth 9).

Feedback: The game should communicate a player's progress towards an objective or their status in the game environment (Sweetser and Wyeth 9).

Immersion: The game should provide a deep but effortless involvement. If a player can lose concern for the environment, time, and to an extent, self, players will be absorbed into the game experience (Sweetser and Wyeth 10).

Social: Not a condition as it relates to the original concepts and flow requirements, but games are unique in how players engage with them for social interaction. Games, by extension of this fact, should facilitate opportunities for social competition, cooperation, or connection between players. This is not necessarily important in the game itself but can be visible through virtual communities or how games enable sharing experiences (Sweetser and Wyeth 10-11).

While useful, these conditions still need to be applied to a context. Without context, each condition only speaks to the potential value they provide to a game experience. With context, the most enjoyable or dysfunctional aspects of playing a game may be considered with more depth and precision. For example, the *Root* AI are perceptively given unearned upgrades. In the context of being just a player, these rule definitions seem to reward the AI during play without a clear reason. On the opposite hand the player must earn all their upgrades at a cost of actions, cards or acquiring specific territories on the game board. Through the context of design however it is apparent that the AI is designed to challenge a specific skill level of player. Without this upgrade system the AI is unable to maintain an evolving competitive threat. The tension in this decision for players is thus how this change affects players of all skill levels instead of just the intended group and defines where an update in design should be applied. As illustrated these conditions are most appropriate when discussed alongside the analysis of a system and its intention rather than without it.

With a reasonably rounded description of what is fun for players and a general understanding of solo games, we now need to consider a keystone of solo game functionality. That is the origin, purpose, and design of artificial intelligence.

The Artifice of Intelligence

When I took my first steps into uncovering AI in a game design context, I was at odds with how little information was documented compared to how much the term seemed to be used in practical development. Through my various readings, I assume that AI in solo games is in many ways developed from "borrowing" or optimising mechanic ideas from other solo games that use AI as a central solo game feature (Salen and Zimmerman 298).

Game design is not a field faulted for being self-referential because of its uniqueness as a medium. Still, in doing so, sometimes unknown rough edges of the design are reinforced instead of ironed out of the practical development process. This is most prevalent when considering the design of solo game AI and ideas that seemed to prevail despite evidence that players did not enjoy their appearance. For this reason, instead of reproducing any intended or unintended quirks of designing an AI system, it is necessary to rebuild a definition of foundational information required to design AI for solo games. The text that helps me define this is the book *Artificial Intelligence: A Modern Approach* (2010), written by computer science researchers Peter Russell and Stuart Norvig.

Let's restate that AI is artificial intelligence (Russell and Norvig 1). It is a term brought from the fields of science and engineering (Ibid). The field of AI is primarily concerned with how to "build intelligent entities" and pursuing an understanding of what makes something intelligent (Ibid). An AI entity is also what we define as an agent that "perceives its environment through sensors and acting upon that environment through actuators" (4). The environment is the space to which the agent acts or has access (Ibid). Its sensors interpret, identify, and catalogue environmental information (Ibid). The actuators then enable it to interact with its environment (Ibid).

If we were to describe a human playing a game of *Root* as an agent, we would say the game board and all its associated game bits are the environments; the sensors, the eyes, ears, brain, and skill or cultural experience of a player; the actuators are then not only the hands to manipulate the environment but also what actions are present in the game of *Root* such as initiating a battle or crafting a card. When an AI system can perceive its "environment" through these sensors and manipulate it through actuators, we describe it as an agent (Russell and Norvig 35). Agents can have many roles in human society. Still, generally, they are devised to excel where we as humans have dwindling interests and a lack of physical or mental capability (1). From this point onwards, we will refer to the design of AI purely in its agent form.

Aside: The Semantics of AI, Agents and… AP?

From my initial interactions with the solo game genre and its continued use of the term AI, I've been quite puzzled about how to settle an internal debate about terminology. It's necessary to have a focused and well-researched discussion, but acknowledging my puzzlement will at least highlight something interesting about language within design practice.

At first glance, this may appear to be a semantic disparity between how AI is understood from a computer science context and used in a game design context. Still, the use is essential in understanding the gravity of this difference. There have been many foundational schools of thought for AI (Russell and Norvig 5-16). As such, the only way to correctly speak on the topic is to define what the definition of AI means as it is relevant to a particular field which will be something one must tailor to fit the context.

The most accurate term, from my perspective, would be automated player because this deemphasises the perception of an "intelligent" system (Russell and Norvig 2). From my own design experiences, I've felt it also refocuses the importance of the player in all interactions involving the system. It's never that the system acts in the game, but rather the player makes the system act. That extra step always highlights a level of interaction that will be apparent for players, and hopefully designers, every time a layer of mechanical fun or complexity is designed into an AI system.

Approaches in Agent Design

A general method of considering agent design is first determining its reason for existing and then designing it from what is known as agent approaches. An approach defines not only a general ideology of designing an agent but also the boundaries of its expected interactions and metrics to be perceived as "intelligent" (Russell and Norvig 2).

> "In Figure 1.1 we see eight definitions of AI, laid out along two dimensions. The definitions on top are concerned with thought processes and reasoning, whereas the ones on the

bottom address behaviour. The definitions on the left measure success in terms of fidelity to human performance, whereas RATIONALITY the ones on the right measure against an ideal performance measure, called rationality. A system is rational if it does the "right thing" given what it knows."
(Russell and Norvig 1-2)

Thinking Humanly	**Thinking Rationally**
"The exciting new effort to make computers think . . . *machines with minds*, in the full and literal sense." (Haugeland, 1985) "[The automation of] activities that we associate with human thinking, activities such as decision-making, problem solving, learning . . ." (Bellman, 1978)	"The study of mental faculties through the use of computational models." (Charniak and McDermott, 1985) "The study of the computations that make it possible to perceive, reason, and act." (Winston, 1992)
Acting Humanly	**Acting Rationally**
"The art of creating machines that perform functions that require intelligence when performed by people." (Kurzweil, 1990) "The study of how to make computers do things at which, at the moment, people are better." (Rich and Knight, 1991)	"Computational Intelligence is the study of the design of intelligent agents." (Poole *et al.*, 1998) "AI . . . is concerned with intelligent behavior in artifacts." (Nilsson, 1998)

Figure 1.1 Some definitions of artificial intelligence, organized into four categories.

Figure 2 - Visual graph of Rational, Humanly, Acting, Thinking AI Approaches

Each approach suggests an aspect of acting in an environment that displays what we define as intelligence (Russell and Norvig 1-2). Going through each in concise detail, we can say that:

Thinking Humanly is how an agent models cognitive behaviour. In essence, an agent will be designed to match human behaviour to the "input–output behavior" of a system (Russell and Norvig 3). To derive a pattern of behaviour, one can usually use three methods of self-inspection of our thoughts, psychological experiments observing others, or the more easily and readily available brain imaging technology while said brain works through a set of actions (Ibid).

Acting Humanly is how an agent processes language and subsequently communicates, stores new and old information, uses said information to answer questions/draw new conclusions and finally detects and adapts to unique circumstances (Russell and Norvig 2). Each of these abilities forms the criteria for passing the formalised Turing Test developed by Alan Turing to qualify whether there is "operational intelligence" in how an agent conducts itself (Ibid). Additionally, the test does not need to do so with a physical representation as "physical simulation of a person is unnecessary for intelligence" (Ibid).

Thinking Rationally describes an agent's ability to handle the reasoning process by "codifying" the "right thinking" for an agent to determine correct or valuable conclusions about its environment (Russell and Norvig 4). As this process of logical thought is concerned, an AI would be able to use constructed "argument structures that always yielded correct conclusions when given correct premises" (Ibid). A simple example would be if all x are y and all y are z; then all x are z which in "theory" always leads to the "correct" answer (Ibid).

Acting Rationally emphasises the agent's ability to act autonomously (Russell and Norvig 4). Without interference, it can act "so as to achieve the best outcome or, when there is uncertainty, the best expected outcome" (Ibid). This describes an ability to "perceive the environment", "adapt to change and pursue goals", and "persist over a prolonged period" (Ibid). Also noteworthy is how

the agent can engage with different ranges of rational thought. Acting rationally can mean "reflexively reacting" to stimuli, acting despite "no provably correct thing to do", and finally engaging many of the previous approach processes to form a response to environments or events (5).

Humans are very perceptive of the presence of other humans in a game environment. In the paper, *Virtual Humans and Persuasion: The Effects of Agency and Behavioral Realism* (2007) Guadagno et al. state that humans invest emotionally in game "environments" that incorporate entities with likenesses closer to humans (1). Any "member" of the game environment that appears alien or outside of what a player identifies as characteristics of being human reduces immersion where the opposite heightens it (Ibid). For solo games this would mean that an agent that is behaviourally similar to a player will increase the amount of enjoyment felt by the player.

Behavioural likeness describes designing agents with "realism" (Guadagno et al. 2). Achieving realism in an agent is a process of encoding agents with actions and functions that are "representations" of the "human" equivalent action or procedure (Ibid). The agent also needs to operate with an air of "sentience", but only to "the extent to which the participant believes they are interacting with another sentient human being" (3). Sentience is synonymous with intelligence in the way agents only need to perform the "right" action as the player perceives (Russell and Norvig 1-2).

From this information, we can conclude that **Acting Humanly and Rationally** is a reasonable basis for this particular agent design. The foundation for this agent design is rooted in the multiplayer enjoyment of the game *Root* (2018). That is the head-to-head nature of *Root* and all its multiplayer interactions (Kinne). These aspirations mean the agent design must recreate the many interactions associated with playing against another human player. Playing as a human refers to taking logical actions that are aware of the context and consequences of such actions. How these logical actions become possible is through the idea of rationality and task environments.

Agent Rationality

From the cyclical nature of perceiving an environment and acting within it, an agent uses the concept of "rationality" (Russell and Norvig 37). Rationality is an agent's capability and process for "behaving" well, given its current environment. It is the reason agents are perceived as intelligent. Environments are never static because actions are being performed in them and thus shift how they may be perceived (34). For that reason, rationality is essential for an agent to distinguish the most valuable performable action given the environment it finds itself in (Ibid).

The most rational action in an environment is achieved by "considering the consequences of the agent's behaviour" (Russell and Norvig 34). While existing in an environment, an agent will invariably perform a sequence of actions informed by its sensory system. The "history of everything the agent has ever perceived" is then defined as its percept sequence (Ibid). This sequence enables the agent to measure its performance and inform its most rational action choice to achieve what we will expand on as a "desirable" environment state.

Environments all have characteristics. All these characteristics can be quantified and, by extension, manipulated to define different states of an environment. By identifying preferred metrics of these qualities, a designer can determine an agent's performance measure. Performance measures then enable the agent to move towards an environment state that is desirable (Russell and Norvig 38).

Performance measurement is a task of “minimising” or “maximising” preferred metrics so an agent may perform a rational action towards the desired environment state.

Another point on rationality is that the agent cannot, in most cases, use itself solely to measure its performance (Russell and Norvig 37). Doing so leads to a false perception of achieving goals by “deluding” itself of its apparent performance (Ibid). An agent could, for instance, strive to achieve a specific victory point total, but the relevance of when it does this in conjunction with other players is just as important as a parameter to win a game. Going too far in defining and measuring the performance of the agent design can lead to a misunderstanding that rationality is not the same as omniscience.

Omniscience for agents is knowing the results of what will happen before it happens, which is not a goal of agent design (Russell and Norvig 38). In being able to perceive an environment, an agent is not intended to understand the “outcome” of its actions fully but rather act in a way that produces an “expected” outcome (Ibid). Environments will naturally have complex moments, so knowing the best action in every situation is impossible. Designing agents need only consider the history of actions and events instead of the potential of actions and events that might occur (Ibid). Doing so prevents negative environment states that couldn't have been predicted and filters into the agent’s deduction for a rational action (38). In extreme cases, the agent might even deduce that inaction is the best action (Ibid). The agent could, for instance, refuse to play the game because it understands that playing can lead to losing the game. Instead, the agent must act according to the percept sequence “to date” and perform actions until the most rational action is performed (39).

While omniscience is something to filter out of agent design, information gathering to “modify future percepts” is entirely acceptable (Russell and Norvig 39). Information gathering serves the purpose of using the sensor inputs of the agent to perceive the environment and decide on a rational action based on what is discovered. In this way, the agent exercises autonomy by solving situations the designer could not anticipate. Autonomy isn’t always essential for an agent, depending on its motivation for existing in an environment. Still, some level of autonomy is always preferable to “compensate for partial or incorrect prior knowledge” about an environment that a designer could not foresee (Ibid).

Given all that has been said about rationality, we can conclude that a rational agent observes these traits:

- The desired state of the environment
- The undesired state of the environment
- Actions that may be performed
- History of perceived information about the AP’s actions, presence, and the properties of its environment.

When performing rational actions, agents interact with a set of conditions of the environment that we call the task environment. Task environments depict the environment through four parameters: performance measures, environment, actuators, and sensors (Russell and Norvig 40). These conditions create the boundaries of rational actions and how they affect the consideration or manner of execution, the agent might take. They are as follows:

- Performance Measures: What the agent identifies it must pay attention to realise a desirable/undesirable environment state.
- Environment: The physical space, objects and entities that occupy the space.
- Actuators: Anything the agent may use to manipulate the environment state perceptively.

- Sensors: All that enables the agent to perceive the environment it occupies.

These traits are often necessary to define before any design can take place for an agent. Task environments represent the “problem” that agents solve through rational actions. The details of how these solutions are enacted exist in the design of the agent’s agent program.

Agent Programs

A rational agent can differentiate the value of performing one action over another but what enables an agent to perform said action is what we define as the agent program (Russell and Norvig 46). Agent programs use “pseudo-code” instructional language to enact a desired rational action (Ibid). These instructions use the perceptual information available at the moment of undertaking rather than using or adjusting its actions according to the entire history of actions (Ibid). There are four notable trends of agent designs that act as “skeletons” that “embody the principles underlying almost all intelligent systems” and scale rationality (Ibid). They are as follows:

Simple Reflex Agent:

The first noteworthy feature of this agent design is that it selects actions it should perform through its “current percept” sensory information of its task environment (Russell and Norvig 48). The second feature is that this agent design extensively uses **if condition then action** rule definitions to realise behaviour in an environment (Ibid). For example:

If there are enemies here, **then** initiate attack action.

Finer rational choice can also be designed based on lists that correlate performance measure conditions to a set of actions. For example:

If it is my turn, **then** initiate actions 1-3.

1. Recruit units
2. Move units
3. Attack with units

The caveat of this program is its simplicity means its rational action must be similarly simple as rational actions to be performed accurately that can be achieved is generally low.

Model-Based Agent

This agent creates an internal state of its environment to deepen the range and distinction of performable rational actions (Russell and Norvig 48). This is only possible when the agent can interpret how the environment changes “independent of the agent” and how its actions affect the environment (49). In this way, the agent creates a “model” of the world that informs what it deems rational. Besides this, it performs actions similar to the simple reflex agent (50).

Goal-Based Agent

This agent program combines a model of the world with its own set of goals to inform its rational action choices (Russell and Norvig 52). It is no longer operating to simply operate but directing its actions based on reaching specific goals that lead to what it personally defines as a desirable environment state (Ibid). In this way, the agent must make extensive use of searching and “planning” to identify goals and “action sequences that achieve the agent’s goals” (Ibid). This agent

also advances its “decision making” process by considering whether actions lead closer to goals or are only perceived as the best action at the time (53). Through this, the agent takes into consideration its percept sequence history as well as what are likely signs in the environment that define preferable expected outcomes (Ibid). The strength of this state thus lies in its adaptability to reach desirable environment states.

Utility-Based Agent

The final structure focuses on creating and managing an internalised performance measure for the agent (Russell and Norvig 53). Goals provide a sense of which action is rational, but what happens if multiple goals are present and lead to a desired environment state (Ibid)? Instead of each option being equally viable, the agent would rather use their “internal performance measure” with “external performance measure” to deliberate on which goal provides optimal and efficient progression to a desired environment state for the agent (Ibid). All of which generally venture into the realms of “perceiving, representation, reasoning and learning” (Ibid). In most cases, this agent succeeds by its ability to use consistently refined instructional language and robust computation[9] systems.

Conclusion

Even though we have engaged in a lengthy discussion about the two fields of solo games and agent designs, this chapter only captures some things that could be said about any topic. Instead, I have only mentioned what I felt were the most relevant or standard talking points that create context around this specific design endeavour. To recap this literature review, we first established that *Root* as a game exists as a variety of experiences that can all be drawn from three levels to conceptualise, challenge or support mechanical design ideas. Solo game design theory assisted in establishing how solo games function, are considered, their established trends and standardised innovations.

Moving on from solo games, we established how fun is captured through enjoyment and engagement. From this, we defined that learning is an active part of why games are enjoyable to players. Engagement is then the sustained enjoyment of players in game. Players remain engaged with a game as they encounter various conditions of GameFlow. Finally, agent design is the distilled understanding that AI are the sum of an agent approach and rational actions in a task environment enacted by agent programs.

[9] Computation describes an ability to process information to arrive at an outcome (8). AI systems can often have high computational demands to perform as intended or realise rational actions.

Chapter 3 - The Reactive Automated Player Mod:

Reports and Insight of Iterations 1 through 3

One important point to keep in mind: We will see before too long that achieving perfect rationality—always doing the right thing—is not feasible in complicated environments.
(Russell and Norvig 5)

Introduction

The remainder of this report will discuss solo agent design through my design process, observations, and the result of each iteration. Much information has been addressed until this point, so an effort to link or introduce ideas for readability will be made where appropriate. The agent design will be discussed as it appeared in three separate milestone achievements. The simplest way to consider each iteration is through the idea that each describes a design phase. Iteration one establishes a foundation of mechanical ideas for later ideas to build on or extend. Iteration two is a simple refinement of the core and supporting aspects of the game experience. Finally, iteration three extends iteration two's ideas into obtuse, overly designed mechanics. The exercise intends to discover new information via designing to the point of failure. Each iteration will describe a goal, a summary of how it was designed, and an overall conclusive summary of what could be observed about the phase in the project. The agent system I've designed is the reactive automated player (RAP) agent.

Contextual Background

The term Reactive Automated Player (RAP) agent was coined during iteration one because its most captivating actions occur in response to the player's actions. Automated player emphasises that a player is required for this system to function. There is no argument for its intelligence being made in its title and instead highlights that there is a functional role that the player must perform to ensure the game is playable.

The RAP agent is made to play *Root* strictly against a single player, given that this is a solo game. The RAP agent uses the Marquise de Cat faction to play the game, while the player is strictly obligated to use the Eyrie faction. These factions are preferred because they both have a perceivably balanced and "interesting" set of interactions with each other (Wehrle 13). In a one-versus-one setting, the rulebook determines that in situations where player counts are variable, this duo can allow an enjoyable game experience (Ibid). Balance in this capacity is referred to as advantages and disadvantages to win, create favourable interactions and achieve gameplay goals throughout play. For instance, the Marquise and the Eyrie have similar opportunities to win, given their similarly aggressive but nuanced playstyles. Their similar interactions can also be seen in how they deploy units, how they occupy territories through moving and battling and finally, how they score points based on erecting buildings.

To make my designs, I used a game program called *Tabletop Simulator* (2015). This game is a virtual representation of a traditional tabletop experience. The program was chosen because it provides representational interactions such as moving pieces or card shuffling and an ability to craft one's games with enough tinkering. I was lucky enough to find that the original publishers of *Root*

uploaded the original game as a mod[10]. This mod provided many pieces and layouts to create an aesthetic link to the original game. As my design developed, I introduced my own designed game pieces, tokens, and rulesets to make the game playable and supportive of all the solo game mechanics introduced.

Design Goals

Morten Monrad Penderson's Automata[11] agent system sticks out to me as an excellent starting point for realising my own agent system. The vital definition in the automata design being the ability to embrace a "philosophical" underlying of how agents are designed to create play experiences for players (Engelstein and Shalev 13). This is the difference between a game experience supplemented by including an agent and a game that is enjoyable because of its agent. While making something remarkable isn't the goal of this project, a cohesive and refined experience is valuable for players to enjoy a game. To achieve this, I determined I would need to establish a "directed design process". This design process would then stipulate my project goals and core principles of solo game design. Here are the three design goals I outlined.

1. The AP must play as the player does

As mentioned in the section describing behavioural likeness, an important design goal will be to keep the mechanics and play patterns as similar to a regular player as possible. The critical psychological emotional and functional benefit being the GameFlow condition of immersion[12] via behavioural similarity of an agent to a player (Sweetser and Wyeth 10). Emotionally, the player must feel the ups and downs of being outsmarted, underestimated, or aggressed or able to agree with what a regular player would provide. Functionally, however, the player must feel like actions taken by the agent have relevance to the game's outcome. The actions and moments that involve the agent must have as much relevance to affect the game as anything that happens on the player's turn.

2. Mechanical design can be complex depending on its reception with players

GameFlow defines how a game facilitates a player's ability to concentrate and contributes immensely to the enjoyment of the game (Sweetser and Wyeth 5). The game "shouldn't be burdened with tasks that don't feel important" (Ibid). More precisely, we can say that we want to avoid complexity in most scenarios.

In the same breath, however, it is important to acknowledge this agent can only exist through complexity. Complexity is a natural component for achieving detailed rational actions for the RAP agent. An inevitability is that attaining a representationally human agent means increasing the amount of necessary information that can be processed. This increase in "computational demand" increases how rational the agent will appear but then also obscures the presence of "clear goals" and decreases "immersion" and the sense of "control" that the player would have in their game environment (Russell and Norvig 5) (Sweetser and Wyeth 8-10).

[10] The term mod refers to a modification of the core game and its processes. In this case mods are somewhat like downloadable content that allow you to play board games as they were intended with all their pieces, rules and sometimes automated systems for ease of use.

[11] See Zooming in on Solo games page 10-11

[12] Immersion is summarised on page 19.

To resolve this conflict this project must be informed by the player's tested experience. For example, the regular board game reviewer The Cardboard Herald reframes complexity as a positive when comparing the agent designs between the original game of *Root* and its sequel design, *Root: The Clockwork Expansion* (Herald*)*. The latter is praised because complexity is a tool to support a rational agent design which in turn creates an enjoyable game experience. The *Clockwork Expansion* agent design makes it clear that complexity can actively form interesting gameplay moments and become simple over time for the player (Ibid). More importantly however, the complexity is validated by players which is a central assertion presented in this paper.

Based on these interpreted values, my conclusion is that complexity must be designed with the game supporting the player's ability to operate the agent and must make complex mechanics relevant to the player game play experience.

3. The design must support the player's skill development/mastery of the game

Root, at its core, is a game about competitive territory occupation. This defines an inherent challenge for players to succeed through superior strategic prowess. Challenge considered through GameFlow is thus a valuable condition to realise enjoyment in a game like *Root*. If the RAP agent can "sufficiently" match, challenge or ideally support a player's skill development, then the player will remain engaged with the experience (Sweetser and Wyeth 5).

Additionally, in this solo game scenario, the agent can have customisable or designed adaptive mechanics to challenge players appropriately to their skill level. If we consider the notion that fun is pattern recognition and learning, then the system would ideally benefit from having an agent that can introduce or take away game play challenges to maintain engagement. In this way, my goal will be to have the game challenge the player enough to win if they cannot determine a viable strategy. However, in most cases, it should allow the player to pull ahead and win.

The Reactive Automated Player's Task Environment

Before the design process can start, a definition of all the problems, accessible conditions, and limitations that the RAP agent would encounter needs to be established in what we define as the Performance Environment Actuators and Sensors (PEAS)[13] task environment (Russell and Norvig 40).

Task Environment PEAS Description of Root

Agent Name	Performance Measure	Environment	Actuators	Sensors
Reactive Automated Player	All players victory point condition progress All players resources and countable values	Game board Opponent Play Style	Hands All actions and mechanics relevant to Marquise De Cat faction	Eyes Ears Memory

The performance measures correlate to values of victory points and resources/countable values which are necessary for the RAP agent to succeed. Victory points help the agent determine its performance, but as the iterations develop, they also correlate with the enjoyment of the game, as we discuss further. The resources and countable values themselves determine the precision of rational actions that may be taken to play the game with intelligence.

Moving onto the environment, we can note that there are two states of a literal and a dynamic hypothetical environment. The literal environment is the game board while the hypothetical environment is the opponent's play style. The literal environment of the game board provides the boundaries of direct game-related actions available to the RAP agent. The hypothetical environment, on the other hand, considers that the player can adopt a variety of playstyles that change how the agent perceives its own objectives to win the game. The player plays as the Eyrie, an aggressive/expansive/fast-acting faction by design. Despite their knack for this playstyle, they possess the capability to play in various defensive, passive, or slow ways. The design style of each faction also means that pivoting between playstyles and strategies isn't particularly difficult. Through all these values, the RAP agent must be able to interpret the player's objectives and outvalue their manoeuvres through direct or indirect action to ensure it is providing a suitable challenge to the player.

The actuators and sensors speak of human-owned traits because a human operates the agent. The agent will, in this way, have the same essential capabilities to manipulate the game as the player. When we mention the Marquise de Cat's relevant actions and mechanics, however, we are speaking towards the idea that everything available to the faction is available to the agent to challenge the player. On the other hand, memory as a sensor describes an ability to remember actions and moments that have occurred in the game environment. With all that said, the first iteration establishes some foundational mechanics and agent programming for the RAP system.

[13] The acronym PEAS is defined on page 24

Iteration 1: First Steps towards an Agent System

Overview

To start my design process for the RAP agent, I broke my focus into two areas. The first was establishing mechanics from the solo play experience that the RAP agent could define itself around. This occurred through distilling the Marquise de Cat's (henceforth referred to simply as the Marquise) actions into a simple reflex agent program of instructional language to enable the agent to play the game. The second was conceiving a percept[14] history of moments in the environment that the agent could reference to inform its actions, which I define as its memory "architecture" (Russell and Norvig 46). The memory architecture is an additional perceptual input for the agent, but we will discuss this in greater detail further.

As far as gameplay is concerned in this iteration, the agent is only functionally able to play the game. By this, I mean the RAP agent is incapable of enacting winning strategies. Its understanding of rational action is rudimentary as it defines actions in terms of possible or not. It can only act in the environment as strictly defined in the rulebook while omitting complex mechanics complicating its ability to function reliably. Playing against it provides no challenge as its action choices are easily exploitable to win the game. I will start the discussion of the design process by establishing the core actions that the RAP agent must use to play the game.

Starting from scratch

Root has many actions that players may engage in to play the game. Different actions may be used to achieve different feats in the game, and so my design process sought to deconstruct the value and necessity of actions for the RAP agent's use. This assessment broke actions into fields of general and special actions.

General actions describe actions that are widely available to all players and are necessary to functionally play the game. Not being able to recruit a unit makes playing the game impossible for many factions and so is a general action. The remaining actions are defined as special because they are unique, and valuable to factions indirectly or directly, but also not necessary for a faction to function in a game. The Marquise, for instance, has an action called "Field Hospital" that specifies that they can retrieve units that die in combat if specific conditions are met. The action is valuable and unique, but a player can play the entire game without using the action once.

As mentioned earlier functionality of the agent was paramount to completing this iteration. Some special actions were included, but others, such as the ability to generate wood through structures and "overclocking", as mentioned in the figure below, were omitted. The reason is that they don't disable the system's functionality and can be introduced in later iterations with more careful consideration of how they affect the RAP agent's ability to play.

Additionally, these actions can be quite complex to integrate and keep coherent with my design goal of maintaining the complexity of interaction relevant to players. At this point, it was far more valuable to establish general actions as the stable framework for how the RAP agent interacts with

[14] Percepts describes the "perceptual inputs" of the environment agents have at their disposal (Russell and Norvig 34)

the game space and, by extension, the player at later stages of design. With that said, the general actions I outline are determined from the figure below that describes Marquise's turn structure.

Figure 3 - Marquise de Cat Faction player board depicting nested phases of play, possible actions, and action limitations.

The general actions available for the RAP agent are determined as follows:

Battle: Initiating battles between RAP agent's units and opposing units

Recruit: Recruiting Units Marquise units to the board at established "Recruiters" buildings

Build: Placing Buildings that range from "Recruiters" to recruit units, "Sawmills" to generate wood resources and "Workshops" to enable the crafting of cards ability

March: Moving Units between locations designated as clearings on the board

With these actions determined, the next step was forming a turn from these actions for the RAP agent. *Root* turns are split between the three phases of birdsong, daylight, and evening. Birdsong contains preparatory actions and instructions, daylight allows direct and active interaction with other players, and evening draws cards for the factions while concluding their influence in the game environment. The daylight phase enables opportunities to engage in the non-linear execution of actions. These actions shift the game environment and its conditions in various ways. If a player cannot recognise the dynamic application and execution of these actions, they will eventually be overcome by someone who does.

Due to *Root* using linear phases and action sequences, a simple reflex agent program[15] is beneficial for the RAP agent to progress through the game until completion. A simple example of this program's method of delivering instructional language that a player would execute is described as follows:
If it is the RAP agent's turn, perform steps 1-4

1. Recruit Units
2. Move Units
3. Initiate Battles
4. End turn.

[15] Simple agent programs are defined on page 24.

This is the basis of how the RAP agent would enact even as far as iteration three. The next task is to define a motivation for which action should be performed over another and, most importantly, when. In doing so, the agent establishes a crucial part of playing as a human does in defining its own tactical and strategical inclination of how to win the game.

Aside: Indirect mechanics

It is also noteworthy that some mechanics in the grander scheme of playing *Root* have indirect mechanics and value towards winning the game that is important to consider. We can't detail each mechanic's nuance in this paper specifically, but their relevance is not forgotten in the design process. An example of these sub-mechanics can be seen in placing buildings.

Figure 4 - The Marquise de Cat's Building Track

When the Marquise faction places recruiting buildings on the board, they gain fewer victory points than when they place down other buildings. The benefit, however, is that they can draw more cards and recruit units as part of subsequent recruit actions. In such cases, there is an interesting nuance of which action is more valuable—gaining victory points or controlling and influencing the game environment through buildings. *Root's* mechanic design embeds many of these and similar choices throughout play. As such, no correct answer can be made separate from context. The option to do either is possible and is merely a set of sub-mechanics that the RAP agent must use to challenge the player adequately.

Strategy and Tactics through Action Sequences

Performing enough general actions defines a method of playing to win the game we define as a strategy. To strategize describes using an intentional set of actions which we can describe as tactics to achieve a long-term goal (Brathwaite and Schreiber 88-87). Tactics are noted as intended because they directly realise short-term goals (Ibid). Tactics build into and mutate a strategy in this way. A

player might, for instance, have the strategy of being aggressive by using many tactics that focus on being directly oppressive or confrontational towards the opponent. Aggressive tactics carry the exact definition, but the nuance of how that aggression is performed is emphasised. Aggressive tactics could mean occupying key territories with overwhelming forces, preventing opponent goal completion over your own or fighting wherever a fight is to be had.

I identified two considerations for building a strategy and tactical method of playing the game for the RAP agent. The first, which is more functional, is that the agent must select an action sequence through each turn. Each action sequence would be enacted in the daylight phase.

The RAP agent's action sequences were designed from common strategy and tactical action sequences performed by players in the community. Common sources of insight came from observing professional players in tournaments, advice integrated from guides for specific factions, and my definition of Marquise play after playing many games against the Eyrie. These action sequences would be designed to build off the value identified in the environment, informed by human experience in a similar situation and support the previous action taken.

Each sequence title would correlate to what value it provides on the RAP agent's turn. For example, an action sequence would read "Defensive Action" to illustrate that the RAP agent identified a reason to be defensive. The result would be executing actions to provide value towards its assertion of the environment. From this, I established broad strategy categories overseeing general tactical action sequences visible as follows:

Strategy – Aggressive

Tactic 1 – Contest the Enemy's Presence

1. Recruit units
2. Move all units towards a group of enemy units that are less than 3.
3. Initiate a battle against enemy units
4. End Turn

Tactic 2 – Disrupt Enemy Buildings

1. Move largest group of AP units to the closest enemy building.
2. Move 2nd largest group of RAP agent units to 2nd closest enemy building.
3. Initiate a battle where the least number of enemy units are present in a location with a building.
4. End turn

Figure 5 - A visual depiction of the "Tactic 1 – Contest the Enemy Presence"

The weakness at this design stage is that the RAP agent's action sequences do not acknowledge enough known **if-then** conditional statements into actions before they are performed to inform the RAP agent's decision-making process. The update of the action sequence would thus appear as follows:

Strategy – Aggressive

Tactic 1 – Contest the Enemy Presence

1. **If** there are more than 2 recruitment buildings, **then** Recruit units
2. **If** a group of RAP agent units is equal to 3 or more units **then** move them to a clearing with the most enemy units.
3. **If** the RAP agent's units outnumber enemy units, **then** initiate a battle
4. End turn.

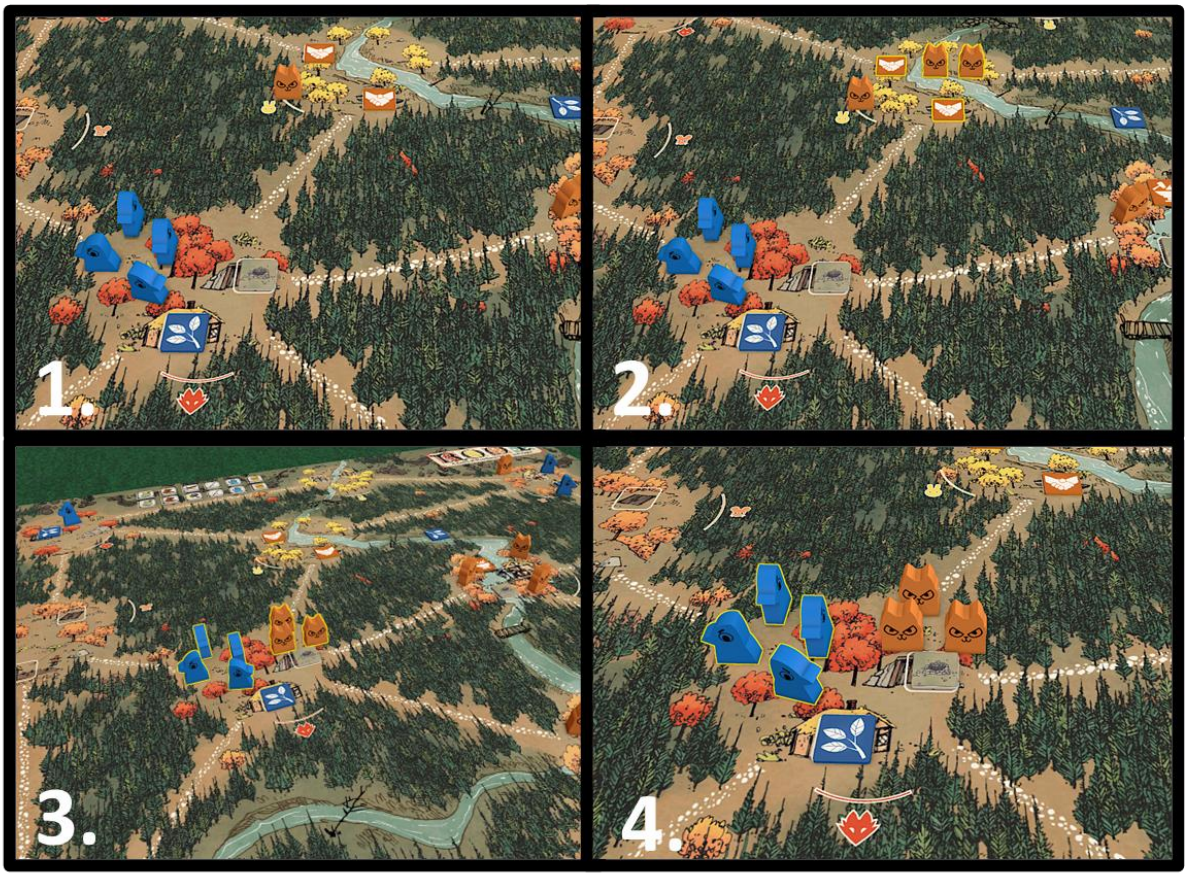


Figure 6 - A visual depiction of the "Tactic 1 – Contest the Enemy Presence" expanded with if-then conditional statements

Let's describe how the RAP agent interprets this. First, the strategy defines that the RAP agent has been motivated to act aggressively throughout this game. Its tactic choice is thus motivated by the overarching strategy and has further that a contestable enemy presence is present. The RAP agent then proceeds to enact the action sequence.

The first step in the action sequence questions whether recruitment buildings are available to gain the full advantage of the recruitment action. The condition is true, so the RAP agent recruits units. Next, it questions whether the number of units is three or more to contest the enemy's presence. It does have enough units and proceeds to contest the enemy unit's presence. The RAP agent identifies that the enemy's presence makes battling less valuable and doesn't initiate.

As you may have noticed in the description above, there are poorly defined motivations behind the strategy and tactic choice by the RAP agent. That is because there is no distinction between the value of performing one action sequence over another. The distinction can only be made if the RAP agent contains a sense of rationality and the ability to react to events.

The Reactive Mechanism

One part of why *Root* is an engaging game relates to the ebb and flow of strategic thinking and execution. From my experience watching and playing the game, I have concluded that sticking to one's strategy is not enough to win. Players must perform a complex strategic game of observation

and counteracting to win in *Root*. As such, plans to counter your opponent, double down on your ideas or press an observed advantage shift constantly. I realised the key to learning this kind of play is thus identification of the past and present while also considering the future. In this way, the RAP agent needed to act based on its perceptual history of the environment.

To achieve this, I quantified the game environment into notable events in gameplay. Notable events were defined as any gameplay interaction that contributed to the Marquise gaining an advantage or disadvantage in the game environment. In that sense, a notable event could be anything from an establishing a building, earning victory points, the opponent crafting a card or units that die in a battle. These notable events were then represented in the game as event tokens. An example of this token system would look like the figure below.

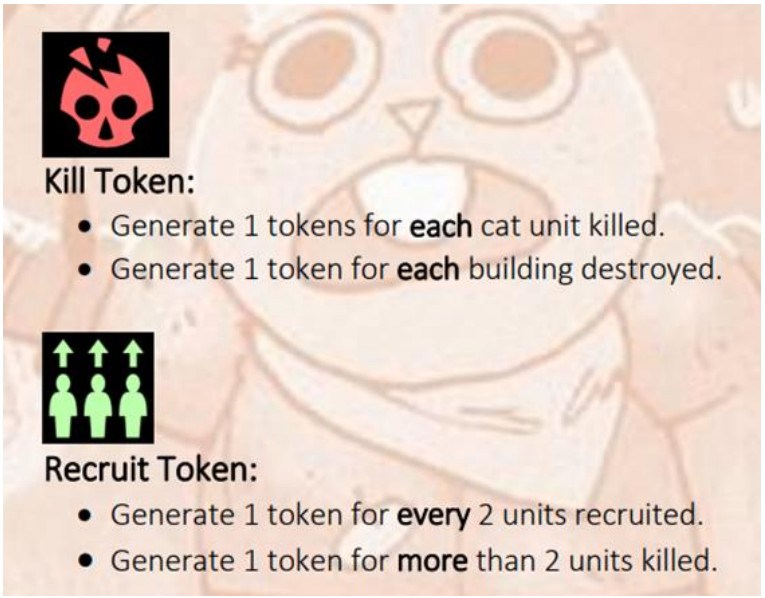


Figure 7 - Depiction of the Token generation system from the rulebook

The idea behind this token system being that before the RAP agent even decided to perform an action sequence it would require the player to identify what notable events had occurred in the environment. Each event would correlate to a token that would be generated when its defined conditions were met. These token when placed into a structured system of identification that I will soon define that would motivate the RAP agent to distinguish which action sequence would be the most rational action to respond to the environment state.

Strategy – Aggressive

Tactic 1 – Contest the Enemy Presence

If the following tokens of **Recruit Token**, **Kill Token** and **Building Token** are present, perform the following actions:

1. **If** there are more than 2 recruiters buildings, **then** Recruit units
2. **If** a group of units is equal to 3 or more units and two spaces away from enemies in a clearing, **then** move units to the specified clearing.
3. **If** AP units outnumber enemy units, **then** initiate a battle
4. End turn.

Memory Architecture

With the basics of the token system in place, it became necessary to consider how this system integrates with the turn of the RAP agent. After all, the token system quantified moments in the game but still had no logic behind which moments were important and when to reference said moments to inform a rational action. Towards this end, I began defining the design of an agent architecture based on memory.

When combined with agent programming, architecture for agents establishes necessary sensory functions or "physical" attributes that define an agent's existence (Russel and Norvig 46). For instance, architecture for an agent that must deliver parcels via the road might be a car and cameras that capture information on its environment to navigate to its destination (Ibid). In a similar sense, this agent uses an architecture that stores and retrieves information as a memory system. Throughout the game, this feature, defined as the "reaction bag", ensures the RAP system can collect and reference event tokens in the game environment.

The reaction bag is a container to hold, conceal and randomise the tokens before the RAP agent uses them to perform its turn. Holding tokens emulates player memory the same way a player would keep track of all notable events during the game to form a response. Concealing the tokens in a bag imitates the nature of obscuring what the player can perceive about their opponents. Randomisation is an essential component of perceiving a rational agent (50). The reason is that in "competitive environments", it prevents the agent from acting predictably by the player (43). How the reaction bag fits into the turn structure works as described below.

Figure 8 – The memory architecture as represented in game through a bag and predefined token system.

Before the RAP agent acts in the environment, it will first draw several tokens. The combination of passes is then linked to a key in the ruleset. This key defines which tactic the RAP agent should perform. After the RAP agent has executed an action sequence, it discards the associated tokens. This is done to acknowledge that the objective linked to said token events has been completed and that the RAP agent should consider other possibilities to win the game.

The final point I'd like to present on how this system functions is through the addition and subtraction of tokens. Drawing tokens from the bag is random and concealed by nature, so to have the RAP agent more deliberately choose its strategy, tokens are added and subtracted from the being considered at different points of the agent's turn. At the beginning of the RAP agent's turn, tokens are added, while at the end of its turn, tokens are fractionally discarded. Adding tokens to the

reaction bag serves to emphasise a particular strategy choice. The RAP agent will be more likely to select a strategy based on how many tokens are present in the bag that relate to it. The same is valid for discarding tokens but devaluing the chance of selecting a specific strategy.

If this consideration wasn't added, the RAP agent's choice of strategy becomes completely chance-based through a random selection of tokens each turn. Purely chance-based selections would create instances where the action sequences do not make rational sense or provide the value the RAP agent needs to win the game.

Conclusion

With the reactive system in place on top of a variety of other general supportive rules to facilitate playing a game, iteration one was technically done. The noteworthy aspects we can observe are as follows:

The RAP agent:

- Can play the game functionally but cannot challenge the player.
- Has a rudimentary understanding of rational actions through the reactive system.
- Has a flawed memory architecture that cannot distinguish between the relevance of events.
- As a result of all previous points plays passively instead of actively in its environment.

The RAP agent can play the game reasonably through its general action list. Some mechanics, such as crafting cards and using wood as a resource, that were central to the Marquise should have been included, but due to their complex integration needs, they were omitted. Their inclusion would have impacted functional play negatively, if not impossible. In general, more design and development time is needed to properly introduce these mechanics to the RAP agent in a relevant way to the player and lower the complexity of interacting with the RAP agent.

As the measurements of rational[16] action apply, the agent works well, if not in a rudimentary manner. It can act in the environment, reference the game environment history, and simply differentiate between desired and undesired environment states. All these conditions, however, are towards the simple goal of being able to function in the game environment. Defining the engagement these systems provide can only be vague. We have or would need to cover the research projects' more extensive interests. This then signals the goals that iteration two can hope to cover in more detail, given that its foundation has been set.

The memory architecture has failed through its specific ability to reference all events instead of just relevant events. A simple way to illustrate the gravity of this point is through an example of our lives lived. Various things are important to us at different points in our lives. As a child losing a toy could affect us significantly, but as we become adults, we learn that many items in life are replaceable and thus that one moment is hardly anything important. As the RAP agent exists, every single moment in its environment is important until the game is concluded. Due to this, its decision-making is very much obscured by the relevance of some events over others. An aggressive action in turn one can still affect turn ten. This occurrence will be felt even when the game and strategy have shifted considerably. The RAP agent's failing in this sense is that there are no means to re-evaluate or discard events that occurred earlier than I could conceive.

[16] The summary of what points qualify an agent as rational action is visible on page 24

A questionably problematic characteristic of this design is that the RAP agent relies on the player to perform any move. If the player is passive, the RAP will be passive; if the player is aggressive, the AP will similarly be aggressive. Passivity doesn't describe doing anything, but rather it will not confront the player in any way that is threatening to the player or valuable towards winning the game. This occurrence is notably atypical of a *Root* game. Compared to the *Root: The Clockwork Expansion* agents, however, we will see much activity in occupying territories, placing recruits, and moving them to battle. The RAP agent is very exploitable with how it plays the game in this iteration.

A significant milestone for the next iteration is addressing how the agent opposes the player and progresses towards the win condition. Once sorted, the iteration must also refine the values mechanics that support the RAP agent's capability to interact with the player meaningfully.

Iteration 2: Refining Engagement

Overview

Iteration one established foundational game interaction for the RAP agent in a simple context. The next step in development was to refine how the RAP agent performed these interactions to progress through the game and engage the player.

I determined two pursuits could achieve this. The first is redefining what strategy and tactics mean to the RAP agent in terms of a "grand strategy" (Brathwaite and Schreiber 88). This grand strategy essentially makes considerable efforts to define the goals and objectives of action sequences. Additionally, it refines how action sequences connect to other sequences by being reverent of actions that could be informed by the past or the future.

The second is that the holistic interaction of the RAP agent needed optimisation. Various supporting features or mechanics needed redesigned interactions to affect players positively. This would range from introducing custom icons, player mats and game pieces for specific interactions and generally refining how certain interactions felt for players through the elaborate process of playtesting the game with players. The goal was to ensure that as many interactions as possible with the RAP agent on a moment-to-moment basis or over a long period were enjoyable and engaging. When both design pursuits were achieved, the RAP agent's design would be ready for playtest review sessions to affirm the values of solo game agent design.

Redefining Tactics

What is immediately noticeable about the RAP agent is that it cannot win or challenge the player in a changing gameplay environment. A prominent reason for these difficulties is my inaccurate incorporation of the faction playstyle mechanics into the strategy layer. Additionally, RAP agent strategy and tactics were not defined well enough to progress towards the game's conclusion. Let us first consider the notion of a playstyle and its role in how *Root* is played.

A playstyle in *Root* establishes which interactions provide value to their faction based on the ease of acting and the outcome they can expect from acting. All playstyles are defined on the back of the faction board. The Marquise has a playstyle that excels at spreading units around the board because move and recruit actions are simple to perform. In this way, some factions find it simpler to play in one method over another.

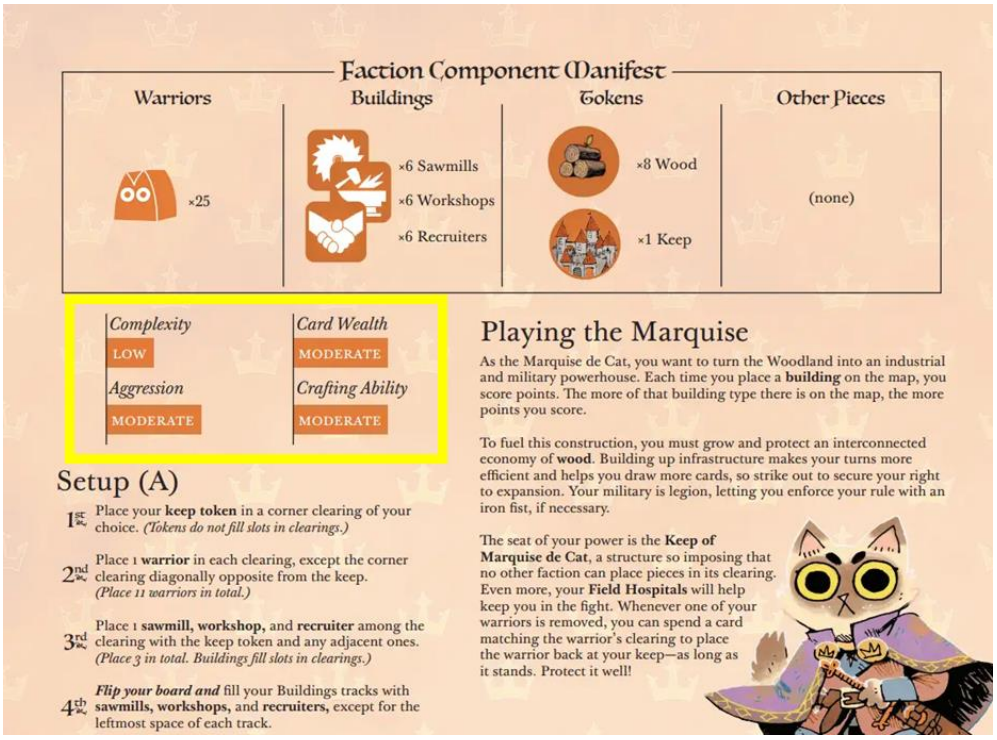


Figure 9 - Depiction of the Marquise de Cat's playstyle definition

These playstyle considerations highlight a weakness in the RAP agent's conceptualisation of strategy and tactics. That strategy was designed separately from playstyle and broadly relates to playing war games in general play styles of aggressive, defensive etc. From these broad strategy descriptions, the tactics were "unstructured" as the analog game design researcher Sven Gaudl might define (18). To improve the gameplay experience of playing against the RAP agent, the RAP agent would need to introduce directed and well-considered action sequences (19).

To structure the RAP agent's strategy and tactics meant establishing a grand strategy. Grand strategy defines overarching goals and objectives the RAP agent is trying to achieve. Through this, the RAP agent can achieve greater rationality through well-considered goals, strategy, and tactics that use its various positive and negative characteristics to its advantage. Unfortunately, the design I arrived at doesn't function as well as intended. I will discuss further in this iteration review as a design consideration of the RAP agent. Nonetheless, it is essential to examine the thought process of how this system came to be and the design I arrived at.

Redefining Tactics - The Grand Strategy

Grand strategy is defined as an "overarching" means to achieve a "long term goal" (Brathwaite and Schreiber 88). Several smaller strategies support a grand strategy. Within these smaller strategies, tactics are arranged to achieve short-term goals (Ibid). Below is my depiction of how the RAP agent appears in each stage of the grand strategy definition.

Figure 10 – Grand Strategy Overview

For the RAP agent, the overarching goal is to challenge the opponent in a way that requires skill to overcome. To keep with the goals of the design project, it then also specifies that if not possible, it proceeds to attempt to win the game itself. To achieve these goals, the RAP agent is designated to perform an overall status of area control throughout the game, which is defined as:

- Organising units and buildings into proximity or allied units and buildings, most notably their starting position.
- Maintaining board presence through high unit numbers throughout the board in locations it deems important.
- Attacking the opponent in locations they have been observed to neglect.
- Moving troops out of locations in a way that does not leave itself exposed from its key location.

The RAP agent then achieves these goals by playing the game within one of the three smaller supportive strategies depicted. These strategies are organised in a way that clarifies how it perceives its status in the game environment. Each strategy is designed to linearly move the RAP agent from the undesirable state of losing to the desirable state of winning. This is achieved by how the strategies' objectives grant the RAP agent resources, territories and victory points. Each strategy will now be described in greater detail[17]:

Survival:

This playstyle assumes the RAP agent has been pushed into a position where their ability to act functionally in the environment has been disrupted. As per the grand strategy, the RAP agent's goals have been compromised in some way. Either a building chain has been disrupted, their unit presence is dwindling or non-existent, or their ability to contest the player is unlikely. The solution

[17] The attached appendix also goes through the strategies in gameplay specific contexts and terms.

then being the RAP agent will attempt to claim unguarded or neutral territories, seek to re-establish essential resources and faction structures, or organise remaining units into defensible/uncompromisable formations to achieve a minor goal. The RAP agent also sparsely attempts to catch the opposing player off guard by employing aspects of the aggressive playstyle.

Aggressive/Intercepting:

This playstyle attempts to gain control of the game environment through aggressive actions such as initiating battles, constructing buildings in critical territories, establishing a unit board presence, and generally occupying opposing player territories. The RAP agent will move or recruit units from well-defended or minor clearings into essential territories of the opposing player. The RAP agent also attempts to push the player into making choices based on dilemmas (Brathwaite and Schreiber 88). Dilemmas simply being a "trade-off" of value between interactions or resources in the game environment (87). For instance, a dilemma could be saving a single territory from RAP agent's units attacking multiple territories. Each location might yield different costs and benefits if control is lost.

Furthermore, it assembles the RAP agent into a looming and recognisable threat poised to attack if left unchecked. Both situations also notably affect enjoyment through competitive "social" opportunities identified in GameFlow (Sweetser and Wyeth 10). The RAP agent can encourage a reason to interact with its actions which is enjoyable to players.

Opportunistic/Observant:

This playstyle contains the most evaluations of the game environment and the player. The evaluations assist the RAP agent in performing actions that create opportunities to expand its empire. Expanding the empire describes performing actions that gain territory, such as moving units and establishing buildings. In addition, the RAP agent will also try to gather victory points and craft abilities.

Tactics in Detail

From these strategies, I determined it was necessary to deconstruct how tactics are logically constructed from the outlined values of their parent strategy. The outline helps describe a process of rational thinking that the RAP agent uses to determine its tactics. The RAP agent clarifies the human player's actions in the outline. Next, the RAP agent defines its interpretation of the value or threat of these actions as an abstract intent. The RAP agent then describes actionable direct and indirect rational actions to act in response or proactively against these abstract intents. Finally, these direct and indirect actions are placed into context and constructed into action sequences. The figure below demonstrates this process.

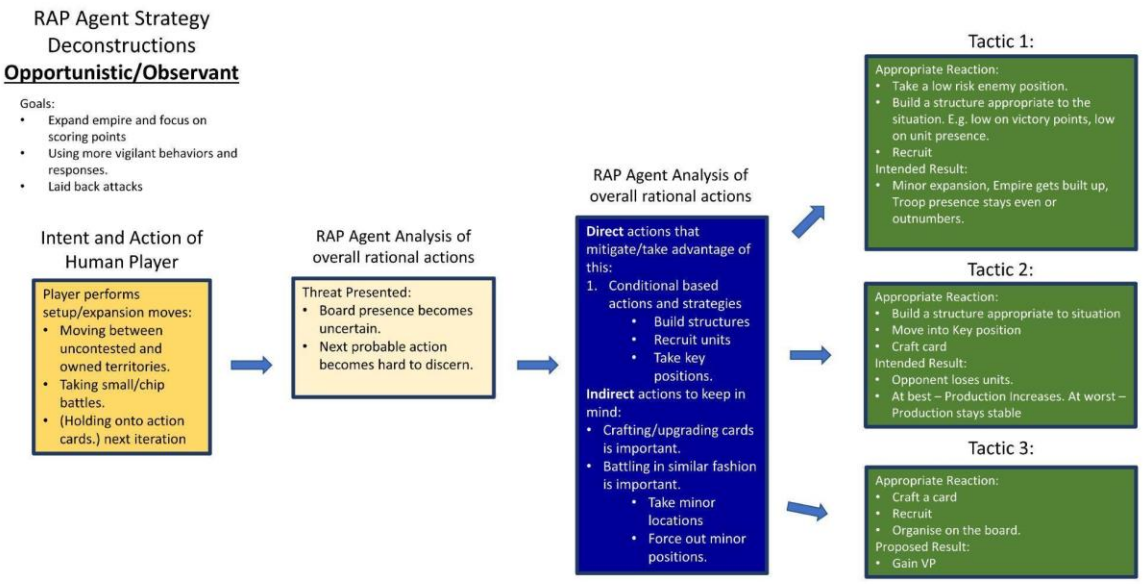


Figure 11 – Opportunistic/Opportunity tactic deconstructed.

At this point, the strategy and tactics of the agent were fully redesigned and could be woven back into the ruleset for the game. The deconstructed strategies presented in figure 10 would appear in the ruleset as depicted in figure 11.

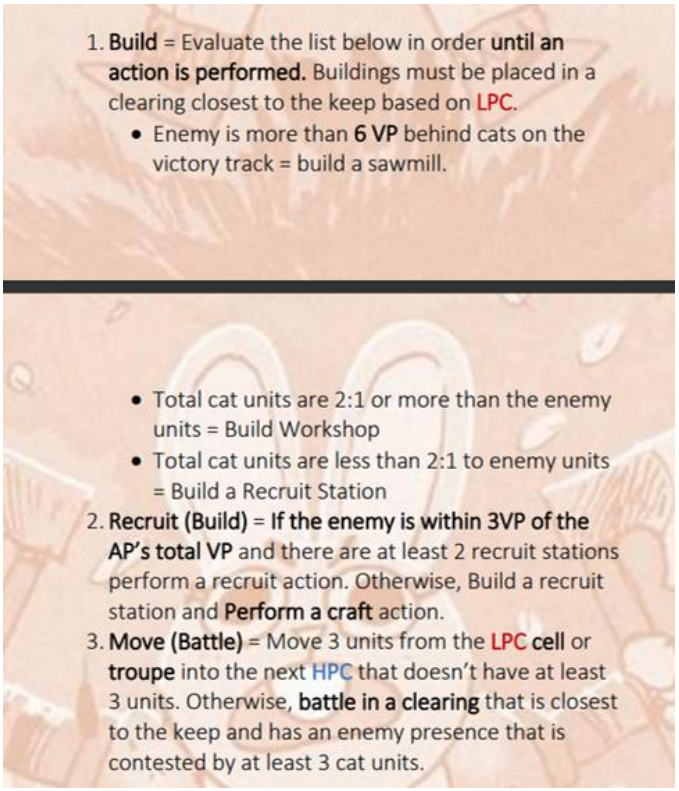


Figure 12 – Tactic two as it appears in the Iteration 2 ruleset for the RAP system.

Let us illustrate the inclusion of grand strategy by explaining how figure 11 would be interpreted by the RAP agent[18]. The RAP agent finds itself in a game environment where the player's actions or presence are not clearly identified. That means its broader strategy is defined as being Opportunistic/Observant. As a result, it determines that it can focus on expanding its empire depending on evaluating the player's board presence.

Through tokens[19] drawn from the reaction bag, the RAP agent selects tactic two as the most valuable and rational action sequence to perform. This tactic determines that its short-term goal should be to contest the player's board presence in minor locations while improving their board presence. The actions that can realise this goal are building, recruiting, and moving. Tied to each action is an evaluation of the game environment that enables the RAP agent to make complex rational decisions based on its next move. The most complex of these evaluations appears in the build action.

The build action assigns a priority to buildings that should be placed in the game environment. This priority is based on the RAP agent's perception of the game environment. The relevance is that some buildings are more valuable in different contexts. For instance, the first condition analyses the distance between the player and the RAP agent in terms of victory points. If the player is behind the RAP agent considerably, then it constructs a sawmill. Note that constructing the sawmill isn't that useful for the RAP agent because the wood resource function is not integrated, and the building itself only provides low victory points. Building a sawmill thus relates to the grand strategy's overarching goal to challenge the player but not actively pursue the most optimal method of winning. The optimal method of winning through build actions would be constructing the most present building type in the environment to gain the next tier of victory points defined on the Marquise game board[20].

Suppose the player is within victory point catch-up range. In that case, however, the RAP agent then moves on to its following rational action by considering the generally valuable building of a workstation. However, the condition for whether this action is valuable is determined by how many units the player has in the game environment. In this instance, if the RAP agent's units do not outnumber the player units by 2:1, then it deems the final building type of a recruit station is more appropriate.

Despite how granular the choice of building a specific type of building is for the RAP agent, each option still fulfils the strategic goal of expanding the RAP agent empire. Action sequences gain or grant value to past, or future turns through these redesigns.

The second action, "Recruit (Build)", demonstrates this when the RAP agent evaluates the game environment. Before committing to recruiting units, the RAP agent first assesses whether this action would be valuable based on the player's victory points amount and, more importantly, whether it has established a certain number of recruitment stations. If the condition is not met, it invests in establishing recruitment buildings to empower future recruitment actions.

The RAP agent in this current state is noticeably capable of acting in the environment and responding to player actions in various well-considered ways. Even better, actions contained a sense of logic as the game proceeded. Actions made in earlier turns were relevant and sometimes crucial to the success of future turns.

[18] Appendix 4 contains the grand strategy goals, strategy, and tactic descriptions in full detail.

[19] Notable events that define the history of past events in the game environment.

[20] Visible in figure 4

Reintegration of Card Crafting/Upgrading Mechanics

As mentioned in iteration one, the actions that the RAP agent could engage in were differentiated as general actions or special actions. Of these actions, some were omitted to establish functional interactions for the RAP agent with its game environment. In this iteration, however, I came across a simple means to reintroduce the mechanics of card crafting as two separate mechanics of upgrades and general card crafting.

The mechanic takes shape entirely in iteration three, but it is worth mentioning here that card crafting was still present in a rudimentary form. This worked because the RAP agent would have tokens associated with action sequences that included crafting a card or generating an ability. Crafting a card had the singular purpose of granting victory points while upgrading worked as a means of gaining passive skills that enhanced the capability of the RAP agent. When upgrading an ability, the RAP agent would flip over a card describing mechanics that would apply to its faction until the game's conclusion.

The purpose of the craft and upgrade ability is to increase behavioural similarity. Noteworthy about this system is that even an unfinished mechanic changes the game's challenge for the player. This is achieved while still being within the scope of abilities the player could receive by crafting cards.

Supporting Features and Mechanics

In solidifying the RAP agent's core gameplay interactions and reintroducing omitted mechanics, I refined the enjoyment of the game surrounding direct gameplay. This was a process of analysing and redesigning how the implementation of game interface-specific supporting systems, rule definitions and mechanics impacted the enjoyment and engagement of the game. The guiding principle for this work is the GameFlow condition of immersion as a "visceral involvement" in the games (Sweetser and Wyeth 5). This condition aims to provide enjoyment to the player even when they are not playing through their turn. In this solo game context, this could easily be half the time given that the player plays for themselves and the RAP agent.

In this regard, many aspects of the game's interfacing design must be optimised. I can't necessarily go through all changes that were made, but I can still the most prominent design and development actions I performed to optimise interacting with the RAP agent.

User Experience Design Principles

Players can only ever play games through the interface that a game designer provides them (Schell 254). In this sense, the methods incorporated into interacting with the game considerably influence the enjoyment and engagement of a game. Within the design of an interface, a few design principles can be used to maintain the "immersive" of interacting with a play through its interface (Ibid).

The main goal of any interface will be to create a sense of "control" as a player interacts or attempts to achieve something within the game environment (Ibid). Essentially the interface "should be easy to use, intuitively organized, and should not sacrifice readability and functionality for aesthetics" (Sweetser and Wyeth 8).

For the RAP agent, that occurred through a variety of solutions. For ease of use, gameplay-related numbers were reduced to their essential amounts. Intuitive organisation and readability meant

formatting the rule set descriptions of mechanics, reducing the possibilities of rule discrepancies, and generally designing diagrams to illustrate the intended means of interacting. The RAP agent's rules, supportive board layouts and mechanic descriptions were never considered during iteration one. This meant that aesthetic direction and formatting were necessary to improve readability. A simple example is that the original game of *Root* does not include any symbols for tokens and organising them. So, I took it upon myself to find some symbols to make them easily recognisable through a colour-coded system and include a simple token mat that indicates the draw order and separates it from the play space after use.

Figure 13 - Token mat for players

Individually all these considerations and changes are very small, but if we consider each change as a positive emotional response, the player will have the changes are notable. Twenty small changes, even if unnoticed, amount to twenty positive interactions outside of direct interaction. "Invisible" changes that do not disrupt the immersion of a player are definitively valuable (Schell 254). That said some interactions require more than formatting to immerse the player in the game such as the complexity of the ruleset.

The Player Decides

It is undeniable that grand strategy has complicated the RAP agent's approach to acting in a game environment with the output being rational actions. The cost of this rationality however is that the computational demands of the system become very complex (Russell and Norvig 5). As a result, interpreting the rules or what the action should do becomes quite challenging for the player.

Conditions and values need to pass through the player before they can be executed by the RAP agent. This common event is detrimental to the players feeling of immersion because they are reminded that they are "participating in the medium" instead of just playing the game (Sweetser and Wyeth 10). My solution was thus to reframe the complexity through the player's own judgment and active participation. When a complex rational problem appeared, the player would use their own

experience to conclude the event with the RAP agent. In this way, I achieved what could be considered a “robust” method of distilling the player’s complex understanding of how *Root* should be played into the RAP agent (Schell 254).

In a game scenario, this could look like the following. When the RAP agent had two equally viable attack locations during the game, the player would be asked to make their most informed decision on how the Agent should proceed. Through this feature, the agent adopts the player’s decision-making. This adoption process has the added benefit of adjusting the RAP agent to the skill level of the individual playing the game. This occurs because players tend to perceive a game environment differently depending on their skill level and can only make choices appropriate to that skill level (Hristova, Guid and Bratko 728).

In moments such as these, the game’s complexity decreases because what would have taken an undecided number of rules/pages to delineate the exact legal move to make is instead considered intuitively by the player. This circumvents the computational demand of operating the system by having players use their complex knowledge, goal definitions, and decision-making process to act in the game. These moments aren't holistically common, but when they occur, the player is no longer meticulously following rules and taking themselves outside the feeling of immersion. Instead, they make a meaningful decision or interaction that will affect the game environment (Salen and Zimmerman 27).

At this point in my design process, the RAP agent played the game intuitively and contained many enjoyable and engaging interactions. The RAP agent provided surprising moments of intrigue and rational decision-making while having an interface optimised for interaction. As the designer, I could not insist on these observations alone to affirm my stances on solo games, so conducting playtest sessions with a range of players other than myself became the next task.

Playtesting

For games to confirm their intended experiences, they need to be placed in the hands of people. For this project, playtests were constructed to gather opinions on the play experience that the RAP agent provided.

Playtests were conducted by contacting various communities and asking whether they would be interested in participating in a research project focused on solo games. Once interested parties were contacted, they were divided and asked to play through iterations of the RAP agent I determined.

Play-testers are valuable for confirming or highlighting this design's achievements, pitfalls, and alternate opinions. Finding play-testers, unfortunately, proved quite difficult despite multiple local and international calls. My sample size was relatively small, which inherently brought into question how the information gained from this event could be used.

Due to my outline of the project being geared towards people who were largely familiar with playing *Root,* the information was already not entirely credible as a source. The information my play-testers provided was still valuable if considered an impression and insight into the game’s enjoyment. Answers given could be used to justify observations, but their experiences were not inherently dismissible. I could not, for example, ask how they compared playing with the *Root: The Clockwork Expansion* solo agent and my own, but I could ask for comments on how they perceived specific interactions with my RAP agent.

Summarising Playtest Review Answers

The summary will be taken from players playing through iteration 2, version 4. Both players finished their game with the RAP agent and won. Both note that they did not fully adhere to all the rule regulations. Unfortunately, this also means I cannot use any number values they provide to support points. Instead, I will focus on their emotive and descriptive play experience. At this point in the design, a general sentiment about the game is that it is complex to comprehend on top of *Root's* already complex method of playing. Nonetheless, the game is still enjoyable through the interactions that the RAP agent facilitates.

Below is my summary of all the play-tester's opinions through a question that best captures the range of answers they present[21].

Question: What was your experience playing against the RAP agent?

Both play-testers answered that the experience changed how they perceive solo games. One elaboration mentions that the token system and range of possible reactions to their play made the RAP agent particularly involved in the game. The RAP agent, however, lapses into feeling inconsequential as sometimes both players admit that they feel as if they are playing against their faction's complex play pattern rather than considering the plans of the RAP agent. Both players note that the RAP agent made them feel "challenged but excited" and generated emotions along with "frustrated, admiration and triumph".

Question: Where do you feel the RAP agent fell short?

An overwhelming weakness of the system is noted as either the complexity of interpreting the correct way to interact with the RAP agent or the RAP agent's overall strategy and individual action executions. One player noted that the RAP agent's strategy was relatively "passive" at times, allowing the player to execute their plans undisturbed. The other play-tester also pointed out that the connection between actions the RAP agent was trying to perform was still not connected well enough. This resulted in scenarios where battles and attempts to capture points of interest were not well supported and thus poorly executed. Interestingly the other player did not note this disconnect and instead felt that the RAP agent acted convincingly with its action execution.

Question: Which moments or mechanics were noticeably interesting to you?

One play-tester noted that the upgrade system helped adjust the difficulty of the RAP agent despite being rather slowly introduced. Similarly related to the sense of challenge, the other play-tester felt that the fluid nature of the RAP agent's ability to challenge them was valuable in keeping their gameplay interactions enjoyable. When they misplayed, they were let back into the game, and when they achieved an advantage, the system seemed to respond with more proactive actions.

Considering an Overall Flaw in Design

Much was achieved in refining the RAP agent's interactions and interface while conducting playtests. In my review of the RAP agent's performance, I felt it was still acting inconsistently despite how many revisions I performed. Sometimes these moments would derail the experience, and others just

[21] The anonymous playtest reviews can be found in appendix 5.

felt like a quirk of the system. Due to this, it became necessary to spend some time diagnosing the issue in the system through all my design notes, playtest observations and playing myself. My diagnosis of this inconsistent agent performance was that my method of implementing the agent program was fundamentally flawed for the rational design I was aiming for.

Playing through a game of *Root* can be quite complex, with various decisions, faction-specific interactions, and scenarios the player is involved in. The RAP agent cannot rely on a linear execution of actions to play this game. No matter how finely I adjust the construction of the available tactics, the permutations of game scenarios are too vast to have a well-reasoned response in just eight proposed action sequences. Additionally, the action sequences are predefined solutions for commonly encountered game situations.

Combining the complexity of *Root* with the linearity of the systems execution results in obtuse and counterintuitive action executions by the RAP agent. For instance, if the outcome of an attempt to capture a territory concludes with more casualties than anticipated, it would be valuable to let the territory go. The RAP agent, on the other hand, insists on capturing said territories even if its potential to hold onto it would be slim if possible next turn. This example highlights that the inherent fault of the RAP agent's design is its inability to interact flexibly with the game.

This may have been a logical set of actions at the beginning of its turn, but the environment changed during this tactic. Without the capacity to consider this or play with more dynamism in its action selection, the system is confined to having recurring obtuse moments during play.

This is still a symptom of the problem rather than the root cause of it. The inherent problem is that the system should be able to create tactical solutions as situations appear dynamically. Dynamic in this context describes actions that are not dependent on the following action to succeed and can be taken out of context for their overall strategic layer if necessary. A need for more aggressive actions could be substituted for passive building actions, for example. If the system could act as if it is tracking the environment, the engagement of the experience would be significantly higher. This kind of feature is only possible when a system uses carefully considered information instead of probable ones.

The reason the RAP agent contains probable information relates to revealing tokens to select tactics. Despite its value for obscuring the information from the player operating the agent system, the token drawing system is still somewhat random and unreliable. The problem this presents is both an issue of rationality but also one of insufficient challenge. In a game like *Root*, one's proficiency at playing increases each turn, and individual actions become equally important in realising the win condition. As such, the game's challenge can never be fully met if the system makes incorrect tactical decisions and inconsistently plays the game as a formidable presence (Sweetser and Wyeth 7-8). Situations such as these become less enjoyable and exploitable weaknesses as players become more experienced.

An ideal solution would be the RAP agent making decisions based on a memory system that can compute all events, actions, and resources in the game environment. Through the computation, it could then choose or create individual actions that form a tactic. These tactics would then move the agent system closer to its performance measures. The main complication of implementing this ideal solution is how it affects the player's experience of operating the system. As I have explored, a process like this tends to be a tediously complex computational management position that players are forced to participate in to get the game to work. Doing so would affect players' intelligibility and subsequent meaningful enjoyment and engagement.

Despite ultimately being a lacklustre solution, I implemented conditionally nested actions to give more options to the RAP agent for atypical scenarios it encounters. The agent would perform action x until the rule becomes illegal or impossible. Should this happen, it would retract all actions and perform y instead.

Action X (Action Y): X rule description. Y rule description.

While this reduces the amount of irrational tactical decisions that can occur for the RAP agent, the system's inability to consider scenarios dynamically prevents this issue from being solved permanently. The exact nature of this problem will be discussed in iteration three.

Conclusion

Despite being at its second iteration, this phase of design confirmed many considerations I had around the nature of solo agent design. Through the design, playtesting, and review of this agent, I could confirm the following:

- Solo agents can use grand strategy to change the player's perception and sculpt the interactions that occur in the game for the sake of enjoyment and engagement.
- Solo agents in competitive environments can feel odd to play against if their ruleset is only partially similar to a human player.
- Solo agent designs are as much enjoyed by their interface and supporting features as they are by their direct game environment representation and actions.
- A sense of challenge and adapting challenge to be appropriate for a player's game skill are valuable sources of enjoyment and engagement.

The RAP agent acts rationally by mimicking many play patterns a human player performs. The manner of achieving this is through the idea of a grand strategy comprised of a hierarchy of goals, strategies, tactics, and individual actions. The importance of grand strategy is that it encourages interaction between the player and the RAP agent, which changes how the player perceives the intelligence of the RAP agent. Grand strategy supports the emotional and social experience of interacting with a solo agent.

This design also confirms that the cumulative experience of interacting with a solo agent is essential for players. Through careful consideration of all the minor interactions manipulating game pieces or interpreting the rules can be enjoyable on a moment-to-moment basis. Similarly, taking steps to improve the readability of rules, the ease of interpreting solo agent-related actions and generally imparting moments of playing into moments where it is still the RAP agent's turn can positively affect engagement.

Root is a game that is based on competitive territorial occupation. While I don't believe solo agents need to strictly be behaviourally similar, in a game environment such as this, representing the agent as close to the player has a remarkable effect on the enjoyment of interactions. There is clear "feedback" on how actions progress or hinder players through similar actions performed in the environment that can be mimicked. If the RAP agent performs a manoeuvre, the player can incorporate that line of thinking into their strategy because it is transparent how it is used and affects play. In this way, the game also teaches the player in some senses to "master" the game and how it can be played. One play-tester suggests this when they mention their interpretation of Eyrie's

playstyle and their reinterpretation of how general mechanics could be used advantageously, even when the outcome appeared to be a disadvantage, such as initiating battles.

Iteration two has largely been successful in addressing the overall project's goals. This report does not mention a fair amount of this success, but it can be experienced through playing the game or reading its manual. That said, my work isn't finished yet, and there are still some rocks I'm eager to turn. As mentioned in my design flaw consideration, there are still either fundamental flaws I can experiment with or ideas I can push to extremes to uncover some virtues of solo agent design. Iteration three will thus be my space to interact with topics around the adaptable challenge for players and move the RAP agent closer to the newly identified dynamic nature of *Root* as a game.

Iteration 3: Extensions of Solo Game Design Features

Introduction

At this point, the RAP agent was stable and capable of playing the game compellingly for the enjoyment of players. It was developed to play the game to completion and included an array of specifically designed solo mechanics to confirm ideas around solo game design.

In completing iteration two, there were concrete examples of how increasing the agent's rational capacity and behavioural similarity impacted players' enjoyment. Confirmation is helpful but only gives us a description of charted territory in the field. As much as iteration two confirmed solo agent design, it also highlighted explorable uncharted territory.

The uncharted territory I'm referring to exists through two ideas. The first is that the rationality of the RAP agent can be further increased if the agent program and game environment are moved from a simple reflex program operating in a static game environment to a model-based program operating in a dynamic environment. The second idea is increasing the fidelity to which the RAP agent can adjust the challenge concurrently during play through dynamic difficulty adjustment.

Overview

Iteration three was unique because the RAP agent functioned well without these additions. It is more accurate to say that despite the value the designed mechanics provide to the RAP agent, this iteration is the most tedious and cumbersome means of interacting with the project. This observation doesn't devalue the work but confirms that complexity can enrich as well as disrupt solo game principles.

All ideas explored in this iteration were pushed to extremes instead of being precisely playable. The iteration is still playable, but the tasks and processes required to play are overwhelming. The experience of playing suffers as a result. This iteration should only be played by individuals who are familiar with how to play *Root*, and its various factions, know how to play *Root: The Clockwork Expansion* solo mode, and have also played iteration two for added clarity. One play-tester who fulfilled the first two conditions mentioned that even with their experience, they found the entire experience quite complex and cumbersome.

Despite all that, the efforts I went through to design this complex system and the interactions it presents are valuable sources of information about solo agent design. With that said, I can divulge the design process and observations about the two topics of dynamism and dynamic difficulty adjustment in solo agent design.

From Simple Reflex to Model Agent Design

As identified in iteration two,[22] the action sequence method of playing the game displayed a limit to how rationally the RAP agent could distinguish its environment. In that passage, I alluded to a

[22] Considering an overall an overall flaw in design subheading on page 49

solution that could work but, at the time, proved far too complex for that design stage. I couldn't merely weave a new solution in and instead needed to reconsider how the agent's actions were designed and constructed. The first step in solving this problem was then a reconsideration of the value of simple reflex agent programming to execute actions.

As described earlier, the basis of the RAP system design lies in what is defined as a simple reflex agent (Russel and Norvig 49). Its functionality is derived from a ruleset that linearly constructs and executes actions to be perceived as rational in its environment. A general problem with this is its inability to internalise its environment. The result is that the agent would sometimes perform irrational or obtuse actions despite the efforts taken to evaluate the environment.

The more significant issue as I continued developing this iteration was that the agent approaches a dynamic game environment with statically constructed action solutions. This, however, could not be solved in the remaining development time for this project and would require a fundamental redesign of the action execution system, which was already linked to several other mechanics, such as the grand strategy. For that reason, I ventured to solve the problem by expanding on the detail to which the agent program could identify its environment via the principles of the model agent program.

The primary purpose of the model agent program is to align the agent's functionality to use as many aspects of the environment as possible to inform its actions (Russell and Norvig 50). In this sense, the RAP agent would need a means to "internalise" how it perceived its environment and how those perceptions could be used to extend the precision of its rational action choices and execution.

Background before Proceeding

To understand the changes, I introduced for this iteration, we first need to clarify that *Root: The Clockwork Expansion* already contained a system for defining its environment. The reason I endeavoured to change this mechanic will revolve around a discussion on static vs dynamism in the game environment. Before that, I must outline how the original system was designed.

The system worked through what is defined as "clearing priority". This feature functioned by giving each clearing on the *Root* map a priority number from 1 to 12. The preference for which number should be considered first is further distinguished by the terms low and high. Low would determine that the solo agent should consider clearing from 1 to 12, while high would invert this order into 12 to 1. The figure below demonstrates how numbers were assigned to the game map.

Figure 14 – Clockwork Expansion Clearing Priority definitions.

The purpose of defining clearing priority is to enable the clockwork solo agent to perform actions in competitive and rationally logical locations. When working through iterations one through two, I reasoned that this system worked well enough to interpret the environment, so I didn't disturb how the RAP agent used this already-designed feature. The RAP agent's ruleset would thus look as follows:

Tactical Set 1 -

1. Do X action in the **Highest Priority Clearing**.
2. Perform X action in the **Lowest priority clearing** with two units
3. Recruit units into the **Highest priority clearing** connected to the home base

While this method of determining the value of clearings worked, it only did so simply and statically. It is simple because there is no guiding principle as to why a clearing is more important than another. It is also static because the priority definition stays until the end of a game. In this manner, clearing priority is arbitrary because it doesn't consider what makes a clearing valuable and how these value systems can shift during play. A simple way to exemplify this is by considering whether clearing 12 would be valuable despite the human player not using it. The answer in most cases will be no, but this system would still define that the clearing is a high-priority clearing. That being the case, I redesigned how clearings are interpreted from statically, once at the start, to dynamically, multiple times throughout the game.

Away from Static and Towards Dynamic Clearings

The overall change I needed to make seemed simple: clearings needed a dynamic value. What impacts that value was only somewhat straightforward. From the community, my playtests and my own experience playing *Root*, I organised the value of clearings into the traits of game board presence and clearing potential.

Each faction will use a clearing at some point to establish a game board presence. Although the use may vary between factions, consistency can be seen in that clearings allow factions to establish a presence in the environment through units, tokens, or buildings. The value goes up or down depending on what the faction pieces enable the faction to do. Additionally, clearings confer victory

points for factions if captured from neutral or enemy occupation or held throughout the game. The eyrie, the faction the human player would be playing, gain victory points based on how many clearings they control with their faction building.

All clearings possess strategic potential based on the number of building slots present and their location on the game board. Clearings in two-player settings have either one or two building slots that can be used to expand their empire by constructing faction buildings. Some clearings are connected to other clearings via multiple pathways, whereas others are limited to one or two channels of entry. Pathways of clearings have a dynamic value in that they can be beneficial or detrimental to a player trying to use them.

Clearings with multiple pathways can be useful, allowing better movement to other clearings. They can act as a forward position to decrease the actions required to gain value in the game environment. On the other hand, they can also be targeted from many potential points and thus be difficult to defend. Limited pathway clearings can have worse value because they are hard to reach and require investment to capture and hold. They can still be useful to specific factions depending on how they are used.

Even with these two traits, the value of a clearing can still be hard to define and dynamically track. This is compounded by the fact that for the RAP agent to use these definitions, it must consider the value of clearings from the perspective of its position and the value it provides the regular player. To account for this, the clearings are divided into two systems of prox and clearing tokens.

Dynamic Environments - Prox

The first system for clearings I describe as a “Prox” system. The purpose of prox, short for proximity, is to enable the RAP agent to determine the most optimal locations to act in when concerned with empire expansion and securing its board presence.

Prox functions by assigning a clearing a rating based on units or buildings located in or adjacent to the clearing in question. The rationale for this value system is that units and buildings are crucial to enabling the Marquise’s actions. They enable many general actions for the RAP agent, determine the proximity of allied game pieces and correlate to gaining victory points through establishing buildings.

Each unit grants a prox value of 1 to a clearing depending on whether it is in or connected by one pathway to a clearing. However, buildings grant a prox of 2 per building occupying the clearing. The values are organised this way to ensure that the system can determine which clearings enable their gameplay actions. Clearings with buildings enable crafting or recruiting, for instance, and thus pose more gameplay potential or risk if captured or lost. Units provide prox to adjacent clearings connected through pathways because nearby units decrease the action required to perform rational actions in the environment. Moving two groups of nearby units into supportive positions can be more beneficial than a singular group further away.

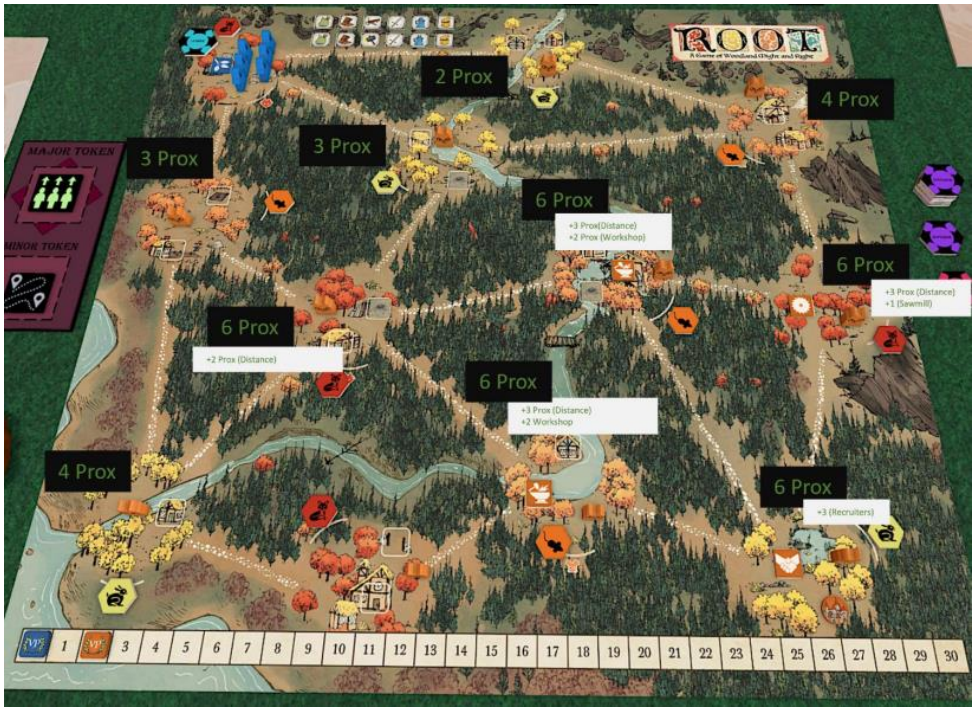


Figure 15 – Prox Values according to a simulated game scenario

These prox values can sometimes cause ties for clearing priority, so a tiebreaking system is included. The system works by first considering the type of building occupying a clearing and then considering the distance from the keep if the tie persists. Building types help differentiate rational action by the value the type of building provides the RAP agent. Further delineation via the keep is functional because the keep is an essential part of the mechanic that allows unit recovery after battles. With all that said, here is a table and chart that defines how the prox system identifies the value of allied clearings.

Prox:

+ 1 Prox	+2 Prox
Each cat unit inside or adjacent to a clearing	Buildings inside a clearing

Ties should be broken using the following prox modifier:

+1 Prox	+2 Prox	+3 Prox
Three Clearings away from keep	Two Clearings away from keep	Adjacent to keep
Clearings has Sawmill	Clearings has Workshop	Clearings has Recruiters

For ease of use you can use the following markers to define which clearings have a prox above the following numbers:

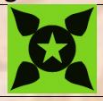
Medium Prox +4	High Prox +7	Clearing with highest Prox
		

Figure 16 – The table for Prox Values

The final note I will add about the prox system is that actions are considered in relation to the value of the prox a clearing has. An action may read as “Perform X action in the clearing with the highest prox” or “Place a building in a clearing with lowest prox closest to the keep”. With this system covered, I captured a method of valuing clearings from the RAP agent’s perspective. Next, I needed to consider the value of clearings from the opposing faction’s perspective.

Dynamic Environments - Clearing Tokens

Clearings occupied by the player are valued through a mechanic defined as clearing tokens. The goal of this mechanic is to create a dynamic definition of clearing value through the frequency of clearing use. I reason that if a clearing receives constant use, it is highly valuable to enable the player’s winning strategy.

Clearing tokens function on a simple premise of evaluating the use of clearings after each opponent’s turn. When clearings are evaluated, they are given a token. Each token signifies that the clearing fulfils a list of criteria to determine its perceived value by the RAP agent. The criteria determine where the player acts in turn and how many actions are performed (or not performed) in that clearing. On top of this, clearings must contain the signature Eyrie roost building to be evaluated.

As identified by playtest reviews and my play experience, roosts enable the faction to perform many actions and earn victory points. Losing roosts severely affects their ability to function in the game environment. This means that clearings must be valued by how adverse their loss would be to the faction. The figure below describes the criteria that clearings must fulfil to be observably valuable to the RAP agent.

Figure 17 – Clearing Tokens their status and conditions.

With these two clearing priority mechanics working in tandem, the RAP agent undoubtedly possessed a finer perception of the game environment. In play, it displayed many moments of rational decision-making and well-considered aggression. The next step, however, was ensuring this aggression was correctly suited towards the player playing. To achieve this, I began designing a mechanic to adjust the challenge the RAP agent represented dynamically.

Considering Difficulty

Dynamic Difficulty Adjustment (DDA) is a design practice that dynamically interacts with the GameFlow condition of challenge (Dziedzic 38). Dagmara Dziedzic, a game design researcher describes this game design theory in detail through the paper *Dynamic Difficulty Adjustment Systems for Various Game Genres* (2016). DDA is valuable because of where it finds common ground between its goals and the GameFlow conditions of skill. As GameFlow describes that “games should provide different levels of challenge for different players” to maintain enjoyment (Sweetser and Wyeth 5).

Through DDA, I hoped to design a system that could respond to the player’s comfort or discomfort of playing against the RAP agent as quickly as possible. The speed directly ties into maintaining a stable state of flow players where they are neither frustrated about losing to difficulty nor bored from winning too easily (Dziedzic 38). To understand how this works in practice, however, we must first dive into some theory around game challenge and difficulty.

A challenge is a task or action a player is asked to complete (Dziedzic 38). The collection of challenges and how a player "exerts effort to influence the outcome" formulates what we understand as difficulty (Ibid). In this definition, it is essential to note that we are moving away from a generic description of difficulty that covers "a wide range of psychological aspects from emotional problems to intellectual and physical challenges" (Ibid). The game design researchers Maria-Virginia Aponte, Guillaume Levieux and Stéphane Natkin further strengthen this definition by explaining that game experiences and a generalised definition of difficulty can be understood as "combinations or sequences of predefined challenges" (Aponte, Levieux and Natkin 8).

An important quality to realise about game design and difficulty is that all players are different. Some games may be similar enough to see patterns in defining difficulty, but as Dziedzic mentions:

> "there is no simple way that allows reducing the complex world of the game to only one parameter, which would correspond to a level of difficulty. Additionally, this approach cannot cope with the diversity of player's skills and their unequal knowledge of the game's mechanics, different capability to learn and adapt to the new situation over time." (Dziedzic 37)

Through this extract, we can note that difficulty shouldn't be statically consistent in its implementation. That difficulty should consider the proficiency of people playing to establish engaging play experiences. As a game design practice, it also exemplifies the intention of how a problem exists in a game, its purpose, and how mechanics are isolated as alterable parameters or interactions are essential.

From this perspective, managing the challenges faced by the player dynamically forms a core design goal. Dziedzic describes DDA as arranging players into a flow channel and keeping the level of challenge they experience relative to their ability to complete it (Dziedzic 38). The principle is simple in that it manages anxiety and boredom depending on how complex or easy a challenge is to overcome (Juul 8). In both scenarios of needing help to situate the player to an appropriate challenge correctly, the game becomes less enjoyable to players. It's also important to note that the game's difficulty assumes that the player is constantly improving their ability to play the game over time and adjusting accordingly (Ibid). Achieving this results in something we can call optimal game flow (Ibid).

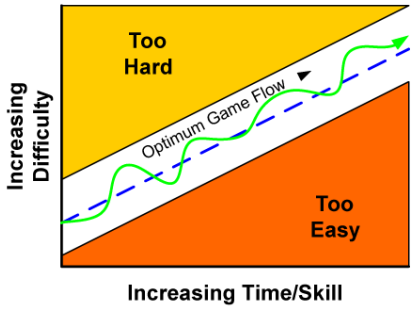


Figure 18 – The Optimal Game Flow Channel Graph

To influence how a player occupies the optimal game flow channel, the RAP agent can manipulate any parameter or mechanic of the game that affects difficulty. These conditions should ideally be altered quickly, with a defined intent and discreetly (Dziedzic 39). However, solo agents are operated by a player, so doing anything discreetly is mostly impractical. The problem with discreetness or any form of hidden information in a solo game is that the player manages the game.

Through being a manager, the player must be cognisant of all the moving parts of the game environment to ensure the game is running as intended.

As this applies to a solo agent, the player is generally aware of what the system is doing each moment of play because they are enacting the agent's moves. As far as the RAP agent is concerned, the only form of discreet play I have managed to design is that of the token system. As noted in iteration two[23], this type of discreet play ventures too far into randomness to be valuable. The randomness the token system introduces decreases the precision of rational actions. Using DDA isn't impossible as much as it is about reconsidering the value of discreet systems for realising engaging play. Furthermore, it is still possible to design alternate systems that either ignore or create a similar value to ensure a DDA system works as suggested.

The next thing to consider in realising a DDA system is defining the difficulty parameters that can be changed. Difficulty parameters are determined through traits of topography, pace, and mutability (Dziedzic 40). Topography describes overlapping actions and movements in a game environment (Ibid). In this category, geometrical games describe an experience of fluidly connected actions and positions in space that are not restricted. Topography describes play organised into individual mechanics performed one after the other and separated from occupying alternate player positions (Ibid). *Root*, for instance, is topological in its actions, such as recruiting, battling, and moving groups of units and territory occupation rules (Ibid).

Pace describes when players or entities may take individual actions (Ibid). This can either be turn-based or continuously active between players (Ibid). Lastly, mutability describes "possible ways of influencing the player's behaviour" through new mechanics or systems between the ranges of static environment, gaining temporary mechanics or permanent gaining mechanics (Ibid). From these terms, we can classify *Root* into the categories of being topological, turn-based and split between temporary and permanent mutable experiences for players. The most easily tangible feature to work with altering the difficulty of the game of *Root* was then the mutable aspects of the game. The system of how these conditions are gained, removed, or persist was already built into the core game. All I needed to do was centre my next designed system around the notable design choices and fit them into what I rudimentarily designed as the upgrade system in iteration two.

Aside: The Clockwork Expansion difficulty adjustments

It is worth mentioning that in *Root: The Clockwork Expansion* (2020), there exists difficulty modifiers to aid in controlling the challenges players experience. The clockwork expansion uses cards that give additional stats or functionality to the faction it is attached to. For instance, a card might read, "Hits against your troops in a battle are reduced by 1". This method of difficulty management does work but possesses two inconsistencies as it functions towards the clockwork's outlined design goals (Britt 7).

Either they compensate for the inability of the system to challenge the player, or they function as unique traits to change the experience players are opting into. Both cases do not align with the original multiplayer experience of playing *Root*. As mentioned in my initial goals for this system, the agent must play as close to that of a regular player as possible. An agent can be crafted to provide a unique experience for players to go against, but this change feels misplaced in the efforts to maintain similarity to the multiplayer experience. That said, a strength of using modifiers at the

[23] Read subheading 'considering an overall flaw in design' page 42

beginning of the game is that players can opt into the challenge they want to face. The weakness, however, is that this difficulty will be constant and static, which does not positively maintain a flow state for players. The goal of my design will be to keep the player in the optimal flow channel dynamically.

Upgrading the System Throughout Play

As mentioned, *Root* already contained mutable mechanics. These mechanics are identified by the card crafting system that grants the player once-off or permanent passive abilities. Iteration two already had a simple upgrade system, so my task was mainly introducing DDA into the design of the mechanic. As a result, the mechanic introduces empowering abilities to the RAP agent and challenges players that get bored in their optimal flow channel.

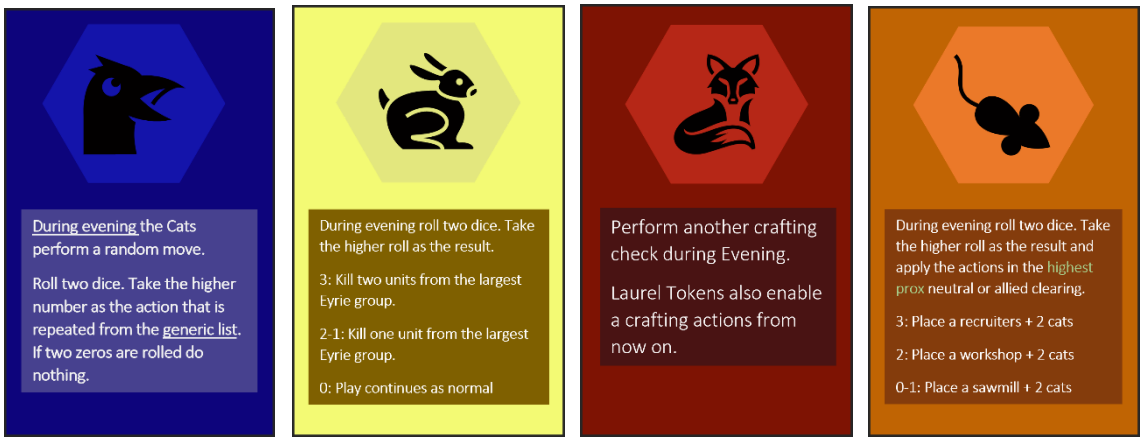


Figure 20 – Upgrade abilities present in Iteration three

The system is realised by acknowledging a competitively impressive feat from the player. A competitive feat could mean occupying a specific number of territories or gaining many victory points in a single instance. These feats generate a newly added upgrade token into the reaction bag. The first instance of revealing said token from the reaction bag anytime during play triggers the upgrade action.

The upgrade action selects one random passive ability from a list of four abilities that empowers the RAP agent with a new ability, as depicted in the figure above. Only three of these four abilities may be gained in a single game. The difficulty of the upgrade mechanic is based on combining these abilities rather than increasing the number value intensity. Additionally, all these abilities are redesigns of abilities the player can get through crafting cards as opposed to uniquely powerful abilities only the RAP agent can acquire.

With this final mechanical addition, my venture into solo agent design was concluded.

Conclusion

Most of these design choices are impractical. Both mechanics can positively affect the enjoyment of solo game agents but need to be simpler to operate. The RAP agent becomes an unwieldy mechanical system akin to a series of rule enforcement tasks. I don't doubt some people might have the patience to operate it, but even I, as the creator, can see that the RAP agent doesn't facilitate an enjoyable game experience in this designed state. Regardless, this iteration was valuable because it provided insight into solo game design in a way that hasn't been extensively demonstrated. What was learned from this iteration is that:

- A model-based agent system provides rational decision-making through the dynamic clarity of its environment.
- Dynamic difficulty adjustment can valuably contribute to a player's position in the optimal flow channel if reframed to work in a solo environment.

The dynamic clearing system creates a fine distinction of the game environment for the agent. The RAP agent can perform even greater rational and informed actions throughout the game, albeit at the cost of the complexity of the user experience. Most remarkable is that clearings are considered on two axes instead of one simplified one. Furthermore, the clearings are considered on a scale of importance rather than through defined number values. This illustrates a robust method of rational decision-making and logical play. Vital clearings aren't always the top priority depending on the RAP system's evaluation methods. A clearing that is optimal to attack but well-defended will be ignored for a less valuable clearing with a higher chance of being captured.

Speaking directly to the upgrade mechanic present, I can identify two achievements. The first comes through the encouragement of unique challenges for the player to overcome. Each *Root* game is always different based on the player's presence in the game and their choices. The upgrade system heightens this by adding challenges via the abilities the RAP system can gain. Some abilities will make the RAP agent more defensively capable, while others enable the system to push towards the game's win condition more easily or frequently. In any scenario involving upgrades, players must adjust how they approach the game to keep these newfound abilities from overwhelming them. It is through this adaptation and combination of abilities that the solo agent can gain that unique game experiences are achieved.

The second comes through the system gaining abilities throughout the game and how well the player performs. Instead of applying difficulty to the system in a static manner that assumes the player's skill level, this new system can introduce a player to a gradual challenge. Through this dynamic adjustment, the system ensures that the player's proficiency isn't an obstacle to participating in a game experience. Often, these changes occur too slowly, and players can overwhelm the RAP agent. This would happen because the player naturally progresses through the game, occurs before the upgrade action is applied to the RAP agent, and then proceeds into a snowball victory[24] for the player. The RAP agent conceptualised as a reactive system has held back the ability to adjust faster.

[24] A snowball victory occurs when a player contains too many advantageous to be disrupted on their path to winning. Often times an earlier advantage leads to gaining successive advantages.

Chapter 4: Summarising Solo Agent Design and Considering the Future

Overall Conclusion

Despite the length and detail of this project, I have no sweeping statements about what was objectively wrong or suitable for the design of solo agents. I only have observations and notes on the effects of specific systems on engagement. This report aimed to explore and define the properties and conditions of solo games by analysing the relationship between solo agents and player enjoyment. Towards that end, these are my final observations about this project and what was learned about solo games.

- Players enjoy mechanics that directly acknowledge or interact with their play within a game environment.
- Behavioural similarity, as well as rational actions, promote a variety of game-related experiences for players when encountered.
- DDA can actively maintain a sense of engagement for players if applied in non-strict definitions.
- Solo agents designed around a range of self or predefined challenge-adjusting parameters are valuable for creating enjoyable experiences for players.
- Solo mechanics should be defined not only from the game context they exist in but also from the purpose they should serve to interact with the player.
- Solo agents that incorporate grand strategy, perceptively flexible tactics and well-considered strategic goals into their design impart a greater sense of presence into the game environment for players.

Before summarising these findings, I'd like to talk more broadly about the project and the observations I've made.

Broad Observations about the Project

I want to mention that there is a symbiotic relationship between theoretical and practical development practices. Throughout the project, there were many moments where the literature or the interactable object would lead or follow the project's development. In iteration three, practical development led to the discovery of dynamic clearings and prox systems, while theoretical ideas refined their presence in the finished design. However, the hierarchy was reversed with the design of the agent system. It started with its practical existence, loose goals, philosophies, and rules and then was expanded by practical development into an executable grand strategy and simple reflex agent design.

This relationship between theory and practice also highlights that RtD helps organise the research process. The value of the information will depend on the person reading. Still, this method of documenting the design approach shows evidence of how a rational design approach of solo agents starts and evolves. Taking this into account, the overall successes and failures of the system also contribute to future efforts in commenting on or improving this system.

That also isn't to say that the RtD approach is an inherently correct method of researching a topic of this nature. Instead, RtD enables a unique method of observation, interaction, and analysis of interactable artefacts in a specified environment are allowed to be. Through these research processes, the interactable artefact presents additional unforeseen and uncapturable (this document has a page limit, after all) insight into the object through interacting with it (Ibid). A good example is that while I had predicted that rational design created engagement for players, I couldn't have reached the cornerstone of fully realising this rational design without creating the memory architecture. It was only through making the memory architecture tangibly that I developed a solo agent that was engaging to play against.

Furthermore, the mystique of the memory architecture formed the foundational playstyle of the RAP agent as reactive and, in some cases, passive. While a proactive playstyle would have been preferable, the reactive playstyle still established the many observable virtues of solo agent design. In this way, it is the making of "implicit" information into "explicit" information that RtD is useful (Herriott 6). In that sense, I'd like to reiterate what is explicit about solo agent design through each iteration I've completed.

With all that said, the practical component of this project is a centre stone of not only discussing but experiencing first-hand all the notable features of solo design mentioned above. To make sense of this, we can recount the design process and observable achievements in each iteration's summary.

Iteration 1 Summary

The first thing necessary to create the RAP agent was describing the context in which the agent should exist and the goals it aimed to accomplish.

In my case, this translated to crafting an agent that acted rationally and primarily focused on representing the play patterns of a human as accurately as possible. This goal was created to capture the multiplayer experience of *Root* as a noted significant source of enjoyment for players. A rational acting agent meant implementing what is known as simple reflex agent design and breaking down the basic mechanics and interactions of *Root* into executable actions. Simple reflex was determined as applicable because they presented a method of creating executed play patterns that were visibly common in *Root*. These play patterns could also be woven into a system that organises these interactions into logical moves towards predetermined performance measures, an environment that was both literal and player dependent and sensors and actuators that were a mix of human and game-related interactions. The performance measures ultimately are the guiding sense of logical action.

My system for organising logical action was achieved through a memory architecture[25]. The memory system would catalogue a determined notable game event for the RAP agent and then use that information to decide on a tactic to perform. These tactics were decided on randomly, achieving satisfactory results in some sense but ultimately problematic ones as development continued. The memory system was useful because it exemplified the intrigue of playing against another player. As a long-time solution, however, it never consistently interpreted the game environment and the relevance of the information it was considering when it received. As a result, the tactics it performed could have been more logical.

[25] Described on page 33 under Memory Architecture subheading.

Despite this, it was apparent that mechanics that can rationally use mechanics to act towards the win condition of a game establish engagement within a solo environment for players.

Iteration 2 Summary

In iteration two, I sought to finalise the systems and mechanics I had developed in iteration one through a refinement process. Refinement meant developing and focusing on how the agent system engaged with the player.

How the RAP agent executed its tactics throughout play needed greater purpose and cohesion. Grand strategy was the solution I defined would solve this for the RAP agent. Through this mechanic, I defined various levels of overarching goals, strategy, and tactics for the RAP agent. These conditions could be used to navigate the complex and varied game state scenarios the agent system would find itself in and execute rationally finer decision-making. Even more so, the actions appeared to be interconnected in nature. This was seen by how actions could magnify the value of past or future actions. This also suggested to players that the system had a logical understanding of how the game of *Root* functioned and what was necessary to progress towards winning.

To further support the player's feeling of enjoyment through immersion, my next task was to refine the many supporting features and mechanics of interfacing with the RAP agent. Despite being small and numerous, the cumulative value of these small adjustments displayed a worthwhile value of increasing enjoyment noticeable through my own play and in playtest reviews. In concluding the design work for this section, I underwent playtesting sessions to confirm the impression the RAP agent gave players. Despite play-testers not fulfilling the original ideal candidate requirement, play-testers still provided many insightful answers on what was enjoyable, frustrating, or notable about my RAP agent design.

Iteration 3 Summary

In the final phase of my project, iteration three expanded complex topics to reveal more information about solo agent design.

The first half of my design work was re-establishing the agent design I started the project with. That being a simple reflex program and moving into a model-based design that could interpret its game environment with dynamic clarity. The initial problem was that the RAP agent approached the game statically and somewhat randomly. At the same time, *Root* required dynamic interpretations of its environment and precision in its action choices to be a challenge for players. Model-based agent programming couldn't solve this problem, but it decreased irrational play present in the game by using mechanics I defined as prox and Clearing tokens.

These two mechanics enabled the RAP agent to interpret the game environment with finer detail. Finer detail described a game environment that split its understanding of locations in the game along the two axes of player and the RAP agent locational value perception. The RAP agent would make more well-considered and informed decisions through these separated definitions.

The final part of my iteration three design work sought to establish a method of appropriately challenging players to maintain player engagement with the game. The expansion of these ideas was derived from the notion of DDA. The goal is to keep players in a state of optimal flow by managing

how difficult the RAP agent appears to players as a ramping threat. The method of achieving this came from re-establishing the upgrade mechanic as a set of mutable abilities. The RAP agent would then give these abilities as the player performed competitively impressive feats. Upon being upgraded, the RAP agent would move the player out of bored emotional states by combining upgrade abilities to increase difficulty.

Both the dynamic environment and upgrade mechanics were enjoyable features. They highlighted the value of more profound rational action precision and challenge adjustment for players. However, the complexity of interfacing with all these new mechanics dulled the experience of interacting with the solo agent.

Considerations for the future

My considerations for what solo agent design could achieve in the future in a solo game context are limited. The reason being the topic was considered in the particular context of a game designed around *Root*. As a result, many of the areas I would mention were worth considering would be primarily defined improvements I would have implemented in a revised version of the RAP agent instead of generally for the field. That said, some considerations that can be separated from *Root*'s context are the agents used to craft solo games and the general need for more information on solo game modes.

Missing Middle of Solo Agent Design

One of the reasons that I pursued solo agent design was because the field contained enough information to compile a comprehensive argument from established knowledge in the field. That said, as noted in the book[26] that provided the primary descriptions for solo agent design, the remaining two categories of solo agent design were sorely under-defined as record and goal based solo games. While both are aligned towards discussing solo games as end states, the number of games that use these conditions as their premise is unmistakable. Some of the most popular solo games, such as Friday and Indian Summer, easily fit into these categories, so uncovering what they establish about solo games would be worthwhile (Engelstein and Shalev 12).

The other side of this solo game consideration is that I have only discussed a particular type of solo game agent. While my solo agent stipulated wanting to be as human as possible, there is room for agents to take an opposing design stance. That being an agent that understands their role as a non-human and uses those traits to the benefit of crafting experiences for players. The game Onirim already achieves this by being compelling through its ability to be a customisable experience and an abstract opponent for the player (Ibid).

Legacy Agents

[26]Building blocks of Tabletop Design by Geoffrey Engelstein and Isaac Shalev.

Given the positives and negatives of the upgrade system I developed, I have had the recurring impression that *Root* isn't the best game to implement dynamic challenge adjustment for players. My overall thought is that in games where single turns can sway the outcome of an entire game, a DDA system can't function reliably well. Regardless, in the instance where the game contained a back-and-forth of pitfalls and achievements, it enhanced the game's engagement. Given these two circumstances, I would say DDA could provide solo games with some advantages for player engagement when the context is more agreeable to its core requirements for being adjusted promptly and discreetly (Dziedzic 38).

Additionally, I'll add that the problem with the current system is that its percept sequence is built up from scratch every time a new game is started. The nature of *Root* means that strategies are defined over a few turns which in many cases can shift an opponent into a succession of wins or an uphill battle to stay in the game. Combining the practice of solo games with the genre and principles of legacy games might, in some way, help circumvent this. Legacy games are "a multisession game in which permanent and irreversible changes to the game state carry over from session to session" (Engelstein and Shalev 25). Outside the context of *Root*, legacy modes would give more tabletop games an extended period of time to identify the player and adjust the challenge they face per session they play through.

Root would be interesting for this format because, as noted in figure 13, the pathways that connect locations are shiftable. Any number of parameters could be used to place *Root* into a legacy game orientation. The benefit from an agent design perspective would be that the agent could gradually build a profile of the player and act more intuitively towards challenging their play patterns. These challenges could be adjusted after playthroughs instead of during them to enable stable agent competition.

Final Statement

My goal for this project was to establish the design decisions and mechanics that enable solo gameplay through an agent-based AI system. From this project I crafted a playable game that through its various iterations show the value and limitations of individual solo mechanics or concepts. My conclusion is that players enjoy and are engaged by solo mechanics that mimic the presence of another player in as many ways as possible, provide justifiably low time and effort consuming interfacing designs and ultimately interacts with a game environment logically and intelligently.

Works Cited

Aarseth, Espen. *Playful Research: Methodological Approaches to Game Analysis*. *Proceedings from the Digital Arts and Culture Conference*. 2003. 1-7.

Aponte, Maria-Virginia, Guillaume Levieux and Stéphane Natkin. *Difficulty in Videogames: An Experimental Validation of a Formal Definition*. *Proceedings of the 8th International Conference on Advances in Computer Entertainment Technology, ACE 2011*. Lisbon, 2011. 1-18.

—. "Measuring the Level of Difficulty in Single Player Games." *Entertainment Computing Vol 2* (2009): 1-27.

Braithwaite, Brenda and Ian Schreiber. *Challenges for Game Designers*. Boston: Charles River Media, 2009.

Britt, Marshall. *Root: The Clockwork Expansion 2 | Developer Diary*. Leder Games, 3 March 2021. 1-8. Document. Accessed 25 January 2023. <https://ledergames.com/blogs/news/root-the-clockwork-expansion-2-developer-diary>.

Davis, Ian L. *Strategies for Strategy Game AI*. *AAAI Technical Report* (1999): 24-27.

Dziedzic, Dagmara. *Dynamic Difficulty Adjustment Systems for Various Game Genres*. *Homo Ludens 1 (9)* (2017): 37-50.

Dziedzic, Dagmara and Wojciech Włodarczyk. *Approaches to Measuring the Difficulty of Games in Dynamic Difficulty Adjustment*. *International Journal of Human-Computer Interaction 34:8* (2018): 707–715.

Engelstein, Geoffrey and Isaac Shalev. *Game Structure. Building Blocks of Tabletop Game Design*. CRC Press, 2020. 1-27.

Friese, Friedemann. *Friday*. 2F-Spiele, 2011. Board Game.

Games, Berzerk. *Tabletop Simulator*. Berzerk Games, 5 June 2015. Digital Game.

Gaudl, Swen. *Design and Refinement of NPC Rules in Digital Board Games*. (2009): 1-73.

Gunn, E.A.A., Bart Craenen and Emma Hart. *A Taxonomy of Video Games and AI*. *Adaptive and Emergent Behaviour and Complex Systems (AISB 2009)*. 2009. 1-11.

Herald, The Cardboard. *Root: The Clockwork Expansion Review*. 7 April 2020. Internet Review. Accessed 25 January 2023. <https://www.youtube.com/watch?v=diwJGSc1_eA&t=362s>.

Herriott, R. *What kind of Research is Design Research*? *Design Revolutions* (2019): 1-11.

Hristova, Dayana, Matej Guid and Ivan Bratko. "*Assessing the difficulty of chess tactical problems*." *International Journal of Intelligent Systems* (2014): 728 - 738.

Juul, Jesper. *Fear of failing? the many meanings of Failure in Games*. *The Video Game Theory Reader* (2009): 1-13.

Kinne, Oliver. *Root: A Game of Woodland Might and Right Review*. 11 February 2022. Review. Accessed 11 January 2023. <https://therewillbe.games/articles-boardgame-reviews/8941-root-a-game-of-woodland-might-and-right>.

Koster, Raph. *A Theory of Fun for Game Design (Second Edition).* O'Reilly Media, Inc, 2013.

—. *Postmortems: Selected Essays Volume One*. Altered Tuning Press, 2018. Book.

Missura, Olana and Thomas Gaertner. *Online Adaptive Agent for Connect Four*. (2010): 1-8.

Nikki, Valens. Mansions of Madness (Second Edition). Fantasy Flight Games, 2016. Board Game.

Orkin, Jeff. *Three States and a Plan: The A.I. of F.E.A.R*. *Game Developers Conference 2006*. Monolith Productions/M.I.T. Media Lab, Cognitive Machines Group, 2006. 1-18.

Rosenberg, Uwe. *Indian Summer*. Edition Spielwiese, 2017. Board Game.

Russell, Stuart and Peter Norvig. *Artificial Intelligence: A Modern Approach (Third Edition)*. New Jersey: Pearson Education Inc., 2010. Book.

Salen, Katie and Eric Zimmerman. *Rules of Play - Game Design Fundamentals*. London: MIT Press Cambridge, 2004.

Schell, Jesse. *The Art of Game Design - A Book of Lenses.* Pittsburgh: CRC Press, Taylor & Francis Group, 2015. Book.

Schmauss, Benjamin and Cole Werhle. *Root: The Clockwork Expansion*. Leder games, 2020. Board Game.

Shadi, Torbey. *Onirim*. Z-Man Games, 2010. Board Game.

Stegmaier, Jamey and Jakub Rozalski. *Scythe*. Stonemaier Games, 2016. Board Game.

Sweetser, Penelope and Peta Wyeth. *GameFlow: A Model for Evaluating Player Enjoyment in Games*. *ACM Computers in Entertainment Vol 3 No 3* (2005): 1-24.

Thurot, Dan. *Foucault in the Woodland, Part One: The Small Folk*. *Space Biff*. 4 November 2021. Accessed 25 January 2023. https://spacebiff.com/2021/11/04/foucault-in-the-woodland-1.

Weibel, David, et al. *Playing online games against computer- vs. human-controlled opponents: Effects on presence, flow, and enjoyment*. *Computers in Human Behavior 24(5)* (2008): 2275-2291.

Werhle, Cole and Kyle Ferrin. *Root*. Leder Games, 2018. Board Game.

Werhle, Cole. *The Law of Root*. Leder Games, 13 July 2020. Rulebook.

Zimmerman, John, Jodi Forlizzi and Shelley Evenson. *Research through design as a method for interaction design research in HCI*. *Proceedings of the 2007 Conference on Human Factors in Computing Systems*. San Jose: Carnegie Mellon University, 2007. 493-502.

Zohaib, Mohammad. *Dynamic Difficulty Adjustment (DDA) in Computer Games: A Review*. *Advances in Human-Computer Interaction* (2018): 1-12.