Thesis no: MECS-2014-05

An Imitation-Learning based Agent playing Super Mario

Behavior created by combining Reinforcement- and Imitation Learning

Magnus Lindberg

Faculty of Computing
Blekinge Institute of Technology
SE–371 79 Karlskrona, Sweden

Contact Information:
Author(s):
Magnus Lindberg
E-mail: meli09@student.bth.se

University advisor:
Dr. Johan Hagelbäck
Dept. Creative Technologies

Faculty of Computing	Internet	:	www.bth.se
Blekinge Institute of Technology	Phone	:	+46 455 38 50 00
SE–371 79 Karlskrona, Sweden	Fax	:	+46 455 38 50 57

Abstract

Context. Developing an Artificial Intelligence (AI) agent that can predict and act in all possible situations in the dynamic environments that modern video games often consists of is on beforehand nearly impossible and would cost a lot of money and time to create by hand. By creating a learning AI agent that could learn by itself by studying its environment with the help of Reinforcement Learning (RL) it would simplify this task. Another wanted feature that often is required is AI agents with a natural acting behavior and a try to solve that problem could be to imitating a human by using Imitation Learning (IL).
Objectives. The purpose of this investigation is to study if it is possible to create a learning AI agent feasible to play and complete some levels in a platform game with the combination of the two learning techniques RL and IL.
Methods. To be able to investigate the research question an implementation is done that combines one RL technique and one IL technique. By letting a set of human players play the game their behavior is saved and applied to the agents. The RL is then used to train and tweak the agents playing performance. A couple of experiments are executed to evaluate the differences between the trained agents against their respective human teacher.
Results. The results of these experiments showed promising indications that the agents during different phases of the experiments had similarly behavior compared to their human trainers. The agents also performed well when comparing them to other already existing ones.
Conclusions. To conclude there is promising results of creating dynamical agents with natural behavior with the combination of RL and IL and that it with additional adjustments would make it perform even better as a learning AI with a more natural behavior.

Keywords: Artificial Intelligence, Reinforcement Learning, Imitation Learning

Contents

Abstract **i**

1 Introduction **1**
1.1 Topic . . . 1
1.2 Background . . . 3
1.2.1 Reinforcement Learning . . . 3
1.2.2 Imitation Learning . . . 4
1.2.3 Finite-State Machine . . . 6
1.3 Problem Description and Statement . . . 6
1.3.1 The Problem . . . 6
1.3.2 Research Questions and Methodology . . . 6
1.3.3 Approach to Answering the Research Questions . . . 7
1.3.4 The Testbed . . . 8
1.3.5 Aim and Objectives . . . 9
1.4 Thesis Overview . . . 11

2 Related Work **12**
2.1 Reinforcement Learning . . . 12
2.2 Imitation Learning . . . 13

3 Implementation **16**
3.1 Reinforcement by the Environment . . . 16
3.1.1 The Environment of Infinite Mario Bros . . . 16
3.1.2 Environment Translated to States . . . 17
3.1.3 Combining and Choosing Actions . . . 20
3.1.4 Reward Calculation . . . 21
3.1.5 The Transition Between States . . . 23
3.2 The Imitation Process . . . 23

4 Method **26**
4.1 Experiment Purpose . . . 26
4.2 Data Gathering . . . 26
4.3 Experiment Execution . . . 27

5 Results **30**
5.1 Learning Phase 30
5.2 Tuning Phase 31
5.3 Evaluation Phase 32
5.4 The Complete Experiment 33
5.5 Mario AI Competition 2009 Results 34

6 Analysis **36**
6.1 Comparisons 36
6.1.1 The Agents 36
6.1.2 Mario AI Competition 2009 39
6.2 Analysis of the Research Questions 40

7 Conclusions and Future Work **42**

References **44**

A Agents data **46**
A.1 Learning Phase 46
A.2 Tuning Phase 47
A.3 Evaluation Phase 49

Chapter 1

Introduction

The following section is containing the aim and purpose of this thesis, which area is in the field of artificial intelligence of video games. Another section is followed presenting more detailed information about the history and technical details of the chosen AI techniques; Reinforcement- and Imitation learning. It is followed by a short overview of the complete thesis. The last section holds information about the problem handled in this thesis.

1.1 Topic

In this research Reinforcement Learning (RL) and Imitation Learning (IL) is combined to test and evaluate if by using these techniques is possible to create a computer player that plays with natural behavior in a platform game. Computer players that play in a natural, sometimes human-like manner, is interesting for most game genres. In this thesis it is however limited to platform games because of its simpler environment compared to other game genres, which makes it a feasible testbed for the time frame of this thesis project. The purpose is to see if this combination of techniques works by comparing it to other existing AIs for platform games. If the outcome of this research is good then the result could maybe be interesting for future work in other game genres as well.

A problem in today's gaming industry is that producing a high quality game often takes long time and costs a lot of money, sometimes up to millions of dollars[1]. However there is a new trend with smaller independent development teams thanks to the increase of digital distribution like Steam[1] and Google Play[2] that is due to the internet being available to more and more people. There is therefore lesser need for the use of a game publisher which makes the investments smaller and therefore gives the developers more freedom due to the lower risk of the project. An effect of this new trend is that games with more varying gameplay have been created and which also have led to old game genres being popular again. One example of this is the increase of new platform games that once was a very popular genre but died out as the performance of graphics in

[1]http://store.steampowered.com/
[2]https://play.google.com/store

games increased and made room for other genres like First Person Shooters in 3D-environments. There have been some successful releases of new platformers lately, like Trine[3] and Rayman Origins[4]. Trine sold more than 1.1 million copies at the end of 2011[2] and has proven that there once again is a big market for platform games.

A common element in nearly all video games since the beginning of that era is the use of Artificial Intelligence, AI, which is used for tasks like controlling a computer player, pathfinding for entities etc., which makes it a big and very central part of many games[3]. This makes it an interesting area to optimize and do research about.

Even on these smaller products the customers will still put high demands on the product with features like dynamic gameplay for replayability, smarter enemies, companions behaving natural etc. that are common elements in modern games. These features often take long time to develop and test. The solution to some of these demands can partially be using a learning AI. This will make it dynamic to changes in the game environment which creates a more varying experience. Another advantage is due to its behavior of learning without supervision that at the same time also can save production costs of the development because less time is needed for balancing and developing solutions for each possible scenario the AI could exist in. This and an increasing market for platform games make further research about AI in the genre important and interesting.

One solution that has previously been used with success in AI agents is RL. In RL the AI explores the game world in a trial-and-error fashion to find a specific goal. The agent will either chose the best earlier performed action in the current environment or perform an available random action depending on the exploring rate the agent is using. A higher exploring rate increases the probability that it will choose a random action instead of the best one. After the action has been chosen it will be evaluated to either be rewarded or punished depending if the action did bring the agent closer to the goal or not. This reward/penalty is then saved to be used next time the agent is in the same or similar situation to decide what the best action to use is. By trial-and-error the agent will learn a way to complete the wanted goal[4].

One drawback with RL is that it often requires playing in the environment several iterations to learn an appropriate behavior. This can be a problem if the behavior of the AI doesn't is what wanted which often is a natural, human-like, and not a robotic behavior that always makes the right decision with no error margins. One way help with the creation of a natural behavior is to try mimicking a human player using IL. By letting the AI agent study a human playing the game and try mimic its actions in the same or similar situations. This can also increase the chance of being able to create an AI with a more natural and human-like

[3]http://trine-thegame.com/site/
[4]http://en.wikipedia.org/wiki/Rayman_Origins

behavior which in many games is wanted.

The purpose of IL is to help the RL agent to behave more natural which often is the wanted behavior for an AI. It is also more difficult to create this type of behavior which will hopefully be an easier and faster task to complete with the combination of RL and IL.

1.2 Background

A big branch in the area of AI is Machine Learning (ML) which is a system that is learning from data. ML can be used in many different ways and approaches and two types of ML is Reinforcement Learning (RL) and Imitation learning (IL) that both are learning from, although different, sets of data. Following section is describing some different approaches and algorithms of Reinforcement- and Imitation learning. Additionally some shorter explanations of techniques often used in Reinforcement- and Imitation learning are also presented. A shorter description of Finite-State Machines that are commonly used in different AI techniques will end this section.

1.2.1 Reinforcement Learning

Reinforcement learning (RL) is a technique that lets the AI study the environment it exist in and use this information to decide what action should be performed next.

The main definition comes down to three aspects:

- Sensation: The AI has to know and understand the environment in some way or another.
- Action: Decide what action to perform by chose the action with the highest (or sometimes random) old reward value for that current state. Depending on the result of the chosen action it is possible for this value to change.
- Goal: There has to be a defined goal that is wanted to be reached in the environment[5].

The AI tries to complete the specified goal by trial-and-error and after a numerous of trials learn the best way to reach that specific goal. After each trial the AI is going to do an evaluation of the last action to see if it was a step in the right direction. This is done by the use of a fitness-function that will either reward or penalize the AI depending on the chosen action in a certain state has helped the AI come closer to its goal or not. The information is then saved and remembered by the AI to help it make an even better decision next time it faces the same situation again. A common fitness-function, Q-learning, is explained next.

Q-learning

One of the most used Reinforcement Learning approaches is the Markov Decision Processes (MDPs), first presented by Sutton and Barto, which is described in this section. Additional explanation of the commonly used reward algorithm for MDPs, Q-learning by Watkins, is done below.

A regular *MDPs* consists of a state machine which purpose is to hold the information that is learned from the environment. The state machine consists of a Q-table, with pairs of state and actions. Every part of an environment will be translated to a state with a set of performable actions. This Q-table contains the Q-value of every pair which is the measure of how good an action is in a specific state; the higher the better. This algorithm consists of four main factors: *current state*, *chosen action*, *reward* and *future state*. It starts by the agent choosing the action with the highest Q-value in the start state. A reward will be received after the action is performed and the agent enters the new state. The agent will choose the highest Q-value in the new state and calculate the reward for the Q-value in the previous state. The reward is calculated according to increase and decrease rate depending on how good or bad the result of the action was which is evaluated through a set of defined criteria. The state machine will now move on to the next state and repeat the steps until the learning is finished, which can be when the agent dies or complete a map[5][6].

The following equation is the common used *Q-learning* algorithm developed by Watkins. It is used to calculate the new Q-value for the most recently performed action:

$$Q(s,a) = (1-l)Q(s,a) + l * (r + d * max(Q(s_{+1},a))) \tag{1.1}$$

$\boldsymbol{Q}$ is the Q-table with the Q-value for each state-action pair that has been discovered. The state, $\boldsymbol{s}$, is which the agent was in when it performed action, $\boldsymbol{a}$, and $\boldsymbol{l}$ is the learning rate used to determine how important the reward from the recently performed action is; the closer the value (varies between 0.0 - 1.0) is to 1.0 the more is the current performance of the action worth. The purpose of the discount rate, $\boldsymbol{d}$, is to decide how much the reward of the best action in the next state is valued. A value (varies between 0.0 - 1.0) close to 0.0 means that it does not take the future reward into any bigger account and if the value would be exact 0.0 it will not use the future reward at all[4].

1.2.2 Imitation Learning

The section below will describe the differences of two categories of machine learning principles that often are used for imitation agents; direct- and indirect learning. A brief description of a modified version of a technique used to imitate human players in a fighting game is shown in the following section.

Direct- and Indirect learning

Even though the concepts of direct- and indirect learning original derived from Machine Learning it is many times used to define what type of imitation purpose is used. With *direct learning* the AI will try to do the exact action the human player performed in each situation. This is done by saving what action was performed by the human in a certain situation or surrounding and then let the AI try to perform the same action when it exist in the same situation. When using *indirect learning* it is most common to have a fitness-method that calculates how similar the playing performance of an agent is to another agent. One of these agents contains data gathered from a human player and the other one is an AI. The AI agent is in many cases a type of RL that can have its reward/fitness-calculation easily changed. It is therefore common to have RL agents with different types of reward/fitness-functions that are compared to the data gathered from the human play to see which of these functions differ least and is most similar to the human[7][8].

Modified Ghost AI

One way to use Imitation learning (IL) is to study a human and gather data that can be used to adjust the rates and constants of other AI techniques like in RL. With the data gathered from the imitation recording the probability of choosing a certain action in a specific state could be changed or used to adjust the learning/discount rate in the Q-learning algorithm or as well the exploration rate for the reinforcement part. A change of the exploration- and the learning/discount rate could result in an agent learning with an increased chance of getting a more natural human behavior. Lueangrueangroj et al.[9] presented an algorithm used to calculate the probability to choose a certain action in a specific situation by using a combination of a weight system that kept track of how good result the actions performed as well as a list holding information about of how frequently the actions where performed in the visited situations.

To adjust the probability of the used action to be used again in a situation the following equation is used:

$$Pa = Pf + (C * (Pw - Pf)) \tag{1.2}$$

$\boldsymbol{Pa}$ is the probability that the action will be used next time the same situation occurs, $\boldsymbol{Pf}$ is the probability calculated by how frequently the action has been performed in that situation, $\boldsymbol{Pw}$ is the weight probability gotten from the imitation learning that was deciding if the performed action generated good result or not. Lastly the $\boldsymbol{C}$ is a constant that often is set to a value around 0.1. All probability variables are measured in % (percent). All probabilities are together normalized to 1.0 to keep the sum of the probabilities of all actions together to be 100%.

1.2.3 Finite-State Machine

A Finite-State Machine (FSM) is a model to hold and change between a finite numbers of states. Depending on a trigger or some fulfilled conditions a change from the current state to a new state can occur. This change between states is called transition. Depending of what type of FSM is used it could be possible to know the next transition at the same time the FSM change to a new state. The two different types of FSM are deterministic and nondeterministic. With a deterministic FSM the possible transitions to new states are known on beforehand. In contrast a non-deterministic FSM will not save transition between states and could therefore not know the possible transitions to new states in the current one[10][11].

FSM is often used for different purposes in the area of AI like for example in reinforcement learning holding the possible actions in the different environment states and handling the transition between them.

1.3 Problem Description and Statement

This section first describes the problem handled in this thesis additionally with the following section 1.3.2 that defines the research questions of this thesis and how these are approached to be able to answer them. Which testbed and the motivation of this choice is presented in section 1.3.4. Following sections shows what steps and objectives that are needed to execute to be able to find a solution to the proposed problem.

1.3.1 The Problem

Due to the high risks and investments in game development it is interesting to find techniques that can ease up the development resulting in smaller costs and risks, which always is a good thing. It is therefore the main purpose of this thesis is to study if RL can be successfully combined with IL to create an AI agent acting with a natural behavior that often is a wanted feature in modern games. To be able to evaluate the agent it is used in a set of tests that will measure its playing performance on different levels in the platform game Infinite Mario Bros. The results of these tests will be used to compare it to other existing AIs so it can be evaluated if the combination is useful for future commercial platform games and if it has potential to be an interesting subject for further research in other game genres as well.

1.3.2 Research Questions and Methodology

The main purpose of this thesis is to try to answer these following questions:

- RQ1: Is imitating a human player using Reinforcement learning a feasible technique for creating a learning based AI in Infinite Mario Bros?
- RQ2: How does the proposed AI perform against the bots from the 2009 Mario AI competition?

To be able to answer **RQ1** one have to define what a feasible technique really is and there is arguable no exact answer to that. A try do define "a feasible technique" is desribed by the following criteria which will be used in this thesis:

- **Criteria 1** - If the AI agent can complete a map that the human was able to finish it could be considered to be a feasible technique. Should the agent not be able to complete a single map it could maybe be not be considered a feasible technique, unless there are specific reasons such as the map was not possible to complete.
- **Criteria 2** - The results from the combination RL and IL should not differ too much from the performance from the imitated human; if it does some problem with the imitation process probably exist due to not imitating the behavior similarly. Due to the requirement to resemble but not exactly mimicking the specific human player the results from the AI agent is not needed to exactly match the measurements by the human player. Therefore some limit values are set to determine how similar the agent is compared to its trainer and to be able to find some indicators of natural behavior. If a difference is less than 5% for a measurement it is considered very good and similar behavior and with a difference between 5 and 10% is considered acceptable behavior differences. Differences above 10% are no longer seen as similar behavior. These limit values are set by the author and is only used as indicators how much the agent is similar to its human trainer to ease up the comparison against the human trainer, when comparing their playing results. Due to the research of how natural behavior these agents have is outside the scope of this paper it will not be researched in any larger detail.

1.3.3 Approach to Answering the Research Questions

The set of criteria mentioned in the previous section are needed to be fulfilled in a satisfying way before the combination of RL and IL could be considered to be a feasible technique. A couple of empirical experiments will be executed to test if these criteria are met before being able to analyze the combination.

The first criteria that is defined to check if the agent is capable of completing a map studied from the play of the human is tested by comparing some of the data gathered from each played level. The data is gathered from the human playing session and the test executed by the AI agent that both saves the success rate of completing a certain map. If the agent at least is capable to approximate match the success rate for some of the levels compared to the player, this criterion could be considered to be fulfilled.

The purpose of the second criteria was to control if the behavior of the agent was similar to the play of the human. This will be checked by comparing the data including the measurements of distance traveled, success rate and enemies killed. If some of these measurements of the agent are similar to the ones gathered from the human player it could be considered to be enough to satisfy this second criteria.

By satisfying these criteria it can be considered to be enough to support the idea of **RQ1** which is proposing that RL and IL could be a feasible combination to create a learning AI agent in a platform game.

RQ2 will be answered by comparing the results of this thesis' AI with data from other AI agents. Data will be gathered from official publications of the 2009 edition of the Mario AI Competition where the data was gathered from 40 selected maps. The data in the experiment will contain the information of the total distance traveled on the levels, number of enemies killed and the sum of the time left on the played levels. These measurements will be prioritized in a certain order to avoid ties in the comparisons, that was mentioned earlier in this paper[12].

1.3.4 The Testbed

The choice to keep the implementation simple is due to not complicate the evaluation of the combination of RL and IL and to be able to get good results in the limited time period of this thesis project. If the results would prove to be good a more complex game environment from another game genre could be of interest in future research. One thing that can simplify the RL system is the complexity of the environment, i.e. the number of possible states. The less dynamic and simpler surroundings the game has the easier it is to translate and less optimization is needed for memory management and search-algorithms. That is one of the reasons why the platform game, Infinite Mario Bros (IMB) by Markus 'Notch' Persson[5], was chosen. Another reasons are thanks to it is well documented, has been used in a competition that will serve as a benchmark for the AI implemented in this thesis, is open source and free to use. This, and thanks to Mario AI Competition[6][12] that was a tournament held a couple of years with the purpose of creating the best AI agent for the game IMB, has led to the good documentation. Good documentation and simple game environment made IMB to be chosen for the purpose of being the testbed for this thesis.

[5]https://mojang.com/notch/mario/

[6]http://www.marioai.org/

The reason why the 2009 year's edition of the Mario AI Competition is used as the testbed and as the result comparisons over any newer version is because of its extensive amount of documentation and descriptions of the implementation additionally to the larger amounts of results from the tournament that later year's editions are lacking.

1.3.5 Aim and Objectives

The aim of this paper is to research if reinforcement learning can be successfully combined with imitation learning for an AI agent in a platform game and test the playing performance of this combination. One of the tests is comparing the agents against other existing ones to control that the combination performs at least similar as some other already used AI techniques. However the amount of similar human behavior for the agents compared to their respective human trainer is not taken in any deeper evaluation due to the fact that the human behavior is a complete research area by itself. As stated by the work of Tencé et al.[13] believability is a subjective opinion and is complex to answer. Tencé et al. did a large research of believable human behavior and presented a proposal to describe believable human behavior but they were not able to evaluate if their proposal could be used to measure human behavior or not. They proposed the evaluation of their work as future work[13]. Therefore only a simplified approach is used to determine if some natural behavior can be found of the agents. This will is done by comparing the measurements gathered during the playing sessions of the human trainers and the AI agents. The different measurements used in the comparison are described later in this section. Additional information that is used in the comparison is controlling how many maps the agent can finish compared to other existing AIs doing the same test. More extensive behavior evaluation is outside the scope of this research and would probably fit as future work in this subject.

This list of objectives will need to be followed to be able to complete the thesis:

- **Literature review** - To be able to decide what techniques that should be used for the reinforcement- and imitation learning implementations research is needed in these fields.

- **Implement chosen techniques** - After the research some of the techniques will be chosen and used for the implementation part of this thesis. The target platform is a game called Infinite Mario bros.

- **Testing phase nr 1** – A set of 4 human players will play the same 10 levels of the game for the AI to study while using the imitation learning. The data gathered from this test will be used to adjust the exploration rate for the RL agent and also update the probability chance of choosing the different actions in different situations according to the play of the human.

- **Testing phase nr 2** - The updated RL agent will run the same set of 10 maps that they players did in the previous test while using the adjusted RL to learn a behavior as good as it can. After the learning phase consisting of 100 iterations for each of the 10 maps is over, it will change its phase to tune the behavior by running the best behavior, but still continue to learn and save the process. The purpose of the tuning phase is to tweak the agent to perform better with the help from the RL part. This tuning phase will be executed for another 100 iterations but after that the agent will stop its learning and only use the best behavior for the evaluation phase. The evaluation phase is executed for the purpose to save data for the different comparisons and this phase will run the test 10 iterations. All these testing phases will be iterated 6 times to get a mean result and reducing the risk of getting results of bad quality.

- **Measure and compare** - The last phase of the thesis is to measure how good the combination of the two learning techniques is. Data from other AIs will be gathered from publications showing the results from the different Mario AI competitions[12] and other studies using the same format. The Data used in the comparisons is a set of three different measurements and these are the *distance traveled* on the levels, total number of *enemies killed* and the total *time left* on the levels.[8]. The measurments is prioritized in the mentioned descending order and the lower prioritezed measurements are only used if there is a tie on a higher prioritized measurement. The implemented AI in this thesis will run and gather its own data in the exact way as they did in the 2009 year edition of Mario AI Competition. These measurements will be used to compare this AI with other existing ones.

The prioritized data is used when comparing with other agents to decide which is the superior one. A more detailed explanation describing these types of data is presented below together with the reasons of why they were prioritized. This thesis will follow the prioritization list used in the Mario AI Competition 2009 where they used the following list, presented in descending order:

- **Priority 1** - ***Total distance traveled*** is the most important data that is the measure of how far the agent traveled on the chosen ten maps before he either died, ran out of time or when he reached the most optimal situation that is when he complete the whole level. The reason why this is chosen the most important data is due to it is a good calculation of how good the agent can survive and advance on a level before ending up dead, running out of time or getting stuck.

- **Priority 2** - If more than one agent happens to land on the same total distance traveled the second prioritized data will be used instead which is the ***total enemies killed***. This is a measure of how good an agent can handle the different enemies which with a high number of killed enemies means that the agent most probably is dynamic to handle different situations due to the fact that the enemies differs in appearance.

- **Priority 3** - The third data on the list is the measure of how much ***time left*** the agent had when reaching the goal. A higher value means that the agent was fast with completing the level which is sought for. Sadly the time left-calculation is somewhat broken because when the player dies there is often some time left on the clock and this time will also be added to the time left calculation. Due to this is a calculation done by the benchmark from the Mario AI Competition it means that the function cannot be changed and updated to discard the time left in situations like that.

1.4 Thesis Overview

This section is briefly descibing the structure of this paper with a short description of every chapter in their following order.

Chapter 2 Related Work will present previous studies and researches that are of relevance to the aim of this paper and work that are using the techniques used for the implementation in this thesis.
Chapter 3 Implementation is describing the RL and IL implementation of the AI and how the game used as testbed looks like and how its environment is studied by the agents. It is also describing the imitation process that is performed when the human players are playing for the purpose of teaching the agents.
Chapter 4 Method contains the research questions and how they are approached. Additional information in this chapter will describe how the data is gathered during the experiments that are later used in the different comparisons.
Chapter 5 Results are presenting the data gathered from the experiments and the comparison between other existing agents against the agents of this thesis.
Chapter 6 Analysis is analyzing the results and explains why the results look like they did. The chapter is also keeping the answers to the research questions.
Chapter 7 Conclusion and Future Work is summarizing the analysis of this thesis and is proposing some further improvements of the implementation of this thesis and what area that is interesting to do future work in.

Chapter 2

Related Work

Due to the platform genre has existed for a long time now and the Mario AI Competition has been held several times already, there has been significant amount of research done in the field of AI for platform games. Sections following will provide information of related research in the areas of Reinforcement- and Imitation learning.

2.1 Reinforcement Learning

A recent paper by Jyh-Jong Tsay et al.[14] showed that Reinforcement learning can be used successfully in platform games as the authors used Q-learning for the implementation. Tsay et al. created two own versions of the algorithm because of the high memory requirement of the standard Q-learning when saving and loading complex learning environments. They did some optimizations to reduce the total number of states and actions for the purpose to reduce memory usage for their two implementedAI agents that also was tested and compared by the performance of other implementations from the 2009 years edition of Mario AI competition[1][12]. The two created agents performed better than several of the other participants in the competition[14].

Liao et al.[15] used a model free variant of Q-learning. This means that the states doesn't have any connection or traces between other states and that a current state cannot know what state will come next until an action is performed. They translated the game environment to states used in a Finite-State Machine (FSM). They also investigated what learning- and discount rate was most optimal for the reward function in different situations. The main result showed that the agent was after 5000 iterations able to complete around 90% of the levels it played.

To use RL in combination with another AI techniques seems to be a good method as for instance Shinohara et al.[16] is showing when combining Q-learning with the popular search algorithm A*. An A*'s purpose is to search for the shortest path to a defined goal. Their combination is based on a winning AI for the 2009 year edition of the Mario AI Competition. The A* created by Shinohara

[1]http://www.ieee-cig.org/cig-2009/

et al. is used to search for appropriate actions in a certain situation, that can help Mario come closer to his goal. They searched for two steps ahead in time to be able to predict the result of the best action in the future state. These searched actions is then used in Q-learning to decide which of the selected actions would be best to perform in the current environment. This proved to be a really good combination that made the agent nearly always reach the goal for every level tested.

Another approach when creating a learning AI for a platform game was done by Hou et al.[10]. They showed by their research that Genetic Algorithm (GA) can be used to enhance an AI agent created by a Finite-State Machine (FSM) that was used to keep the states translated from the game environment. Even with a simple solution they succeeded to evolve an agent that was able to complete 80% of the tested levels. This has to be considered good result and promising for future work with that combination.

2.2 Imitation Learning

Another area in AI for platform games that has gained some attention recently is imitation learning. Ortega et al. showed that using imitation learning in platform games can be successful for creating an AI that is capable of completing different levels in a game and to some extent getting behavior similar to a human playing style, when it is studying and imitating a human[8]. They used different types of AI agents that used either direct or indirect learning. The indirect learning was done by letting the player play a set of maps, and then let the AI run the same set of maps using a reinforcement technique combined with data from the playing session of the human and creating scripts for each run. After a number of iterations they compared the scripts to the player's movements to decide which script behaved most similar to the human player. Their result showed that the AI with indirect learning was the most similar to the player it studied, but it was still easy to spot the difference between the AI and the real player, according to the persons that was evaluating the accuracy between them.

A good example where learning by imitation is used is in the retail game Black & White[2]. In Black & White you play as a God and to your assistance you have a creature that will start as a small baby with no specific knowledge. As the time goes by, the creature will study the player's behavior and actions so that he can try to mimic these action as the best he can. If the player is goodhearted and for example give food to the villagers, the creature will try to find some food and do the same. But if the player is evil and throw rocks at the buildings, the creature will get an evil personality as well. The game is using an imitation learning model called "Belief-Desire-Intention model" which is using a combination of desires, opinions and beliefs to make decisions for the creature[17].

[2]http://en.wikipedia.org/wiki/Black_%26_White_(video_game)

Black & White was a successful release and showed that imitation learning can be of importance in great games.

Imitation learning is a diverse technique that is not only for platform games or controlling an agent as it has been used in other types of games as well. Togelius et al.[7] tested using IL to create interesting tracks for a car racing game by study the plays of humans. It generated maps that was argued to be fun, which can be seemed as good results. They stated that the use of IL in racing games could possibly be a feasible technique for commercial games.

Thunputtarakul and Kotrajaras[18] created an agent using imitation learning, or as they called it Ghost AI, to mimic the exact movements and actions from the play style of a human playing the fighting game Street Fighter Zero3. The way they did this was by saving the current game world including information like distance between the two players, their current health etc. to states. Every state got an action assigned to it that was recorded from the play of the human, working like a deterministic state machine. To test the precision of the imitation they for instance made the AI mimic the player action in real time without the player knowing which one he was controlling. Some of the persons that performed the test could not point out which player he was controlling. They also presented and tested some calculations used to evaluate the correctness of the AI agents playing performance. The overall result were considered good and stated to be useful for future commercial fighting games.

Additional research about the earlier mentioned Ghost AI was done by Lueangrueangroj and Kotrajaras[9] that extended the technique in the same fighting game Street Fighter Zero 3 by keeping track of how frequent every actions were performed by the human in a certain state and the actions was also weighted depending on who was damaged by the chosen action. If the human player succeeded to damage the AI agent, the action in that state would get its weight increased but if it was the human that got hurt it would instead decrease the weight meaning it was a bad move by the human. To evaluate how natural the agent was they did an experiment letting 9 humans play against it while it saved how much damage the agent took compared to the human. They did the same test for the Ghost AI implemented in the work of Thunputtarakul et al.[18] and the result showed that the damage chart was more random with the modified Ghost AI compared to the original one were the chart had a more regular pattern meaning that the player had an easier time to counter the strategies that agent. With this approach it resulted in a more natural behavior thanks to that the AI now could learn in real time and be adaptable to new strategies performed by the human player, which the unmodified Ghost AI could not.

Similar to the purpose of this thesis Ortega combined Reinforcement learning with Imitation learning in a platform game. There are some differences though. The first one is that Ortegas main aim was to create a set of agents that was trained to act as similar to the play of a certain human. They did some evulation tests to decide wich of the agents acted most similar to the human. The main aim of this thesis is to use Imitation learning to adjust the learning of the Reinforcement learning so that the agent can behave in a more natural way, and not try mimicking a certain human player. The chosen implemented IL techniques also differs alot. The other differences between Ortegas work and this thesis is that Ortega did not do any comparisons how their AI performed compared to other already existing AI agents. This comparison is one of the main purposes of this thesis that will be used to determine if future work in the subject is motivated.

Chapter 3

Implementation

3.1 Reinforcement by the Environment

This section describes the process used when the AI agent is using the reinforcement part of the learning. Subsection 3.1.1 describes how the game environment of the testbed Infinite Mario Bros looks like which is needed to understand before being able to handle the situations that the agent can end up in. The next subsection is about how the states for the reinforcement learning is created and used. The following two sections is about how Mario is choosing and performing his actions. The actions performed by the agent could either be of help for Mario to advance in the world or not at all. It is therefore important to reward and punish the behavior of Mario to help him make a better decision next time he ends up in the same situation. How this is handled is described in the last subsection of this chapter.

3.1.1 The Environment of Infinite Mario Bros

To be able to create states for the reinforcement learning algorithm one needs to understand how the environment in the game looks like. The environment in the 2D sidescrolling platform game Infinite Mario that is the testbed for this thesis is of a relatively simple structure. The player will control the little red plumber Mario that will try to find the kidnapped princess and to do this he needs to reach the end of each level that exist in the rightest side of the map. On his way to his goal he will encounter a number of monsters and enemies with a diverse set of abilities. The different types of abilities can for instance be to jump, shoot, or to be immortal. This results in a variety of different ways needed to approach enemies with specific abilities.

It is not only monsters in the world that have useful abilities, Mario has some too. He can go left or right and he can do this either by walking or running. Additionally he can also jump which could be very useful to overcome obstacles like pipes or stomping enemies. Mario also has the ability to shoot fire balls depending on which mode he is currently in. There are three modes that Mario can be in, descending from best they are fire, big and small. He can only shoot when in fire mode. The environment is a dangerous place and Mario can get hurt by walking into monsters or getting shot. When he gets hurt he is lowered to an inferior mode. If he gets hurt when in small mode, he will sadly die. Another way to die, where the mode he is in does not matter, is to fall off a cliff.

Mario and all enemies lives in a simple world consisting of objects like pipes, coins, bricks, mushrooms and so on. Some of these objects are intractable. The most important ones are mushrooms that gives Mario an upgrade (if he is not in fire mode already) and coins that are collectable and will result in a new life when he got 100 of them. A brick can be destroyed by Mario if he is located below it and jumps right into it. If he is lucky he could get a hidden coin or a mushroom from the destroyed brick. An example of how the world which Mario lives in looks like can be seen in Figure 3.1.

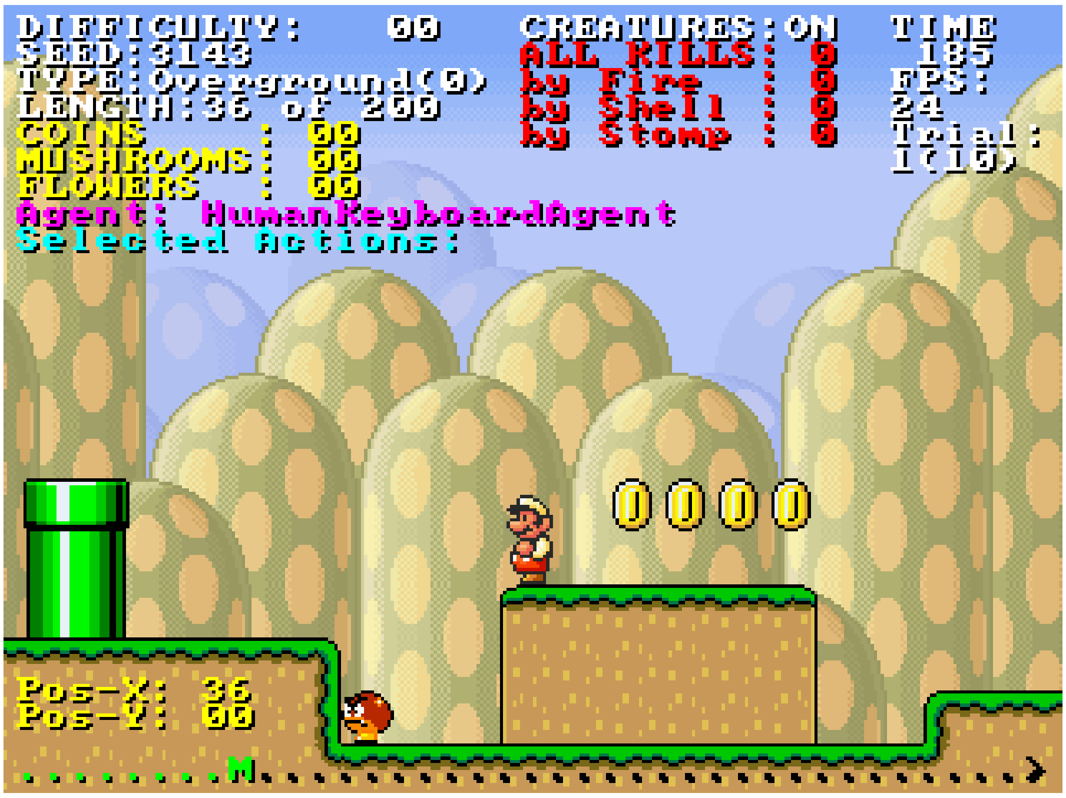

Figure 3.1: Mario in the world standing near some coins and an angry enemy!

3.1.2 Environment Translated to States

The world of Infinite Mario Bros (IMB) is represented by a grid system dividing the whole scene into 22x22 squares and this representation is implemented by two 2D byte arrays containing the information about the parts of the level scene that are currently viewed and the other keeping information about where and what types of enemies that resides in that scene at the moment. These 2D arrays has the size 22x22 which also is the size of game window and every index contain a value between 0 - 256 where 0 means an empty square that is passable area. The other 256 values represents the different types of enemies, obstacles, coins, objects and so on that the world is created of.

When the game environment is known it is possible to translate it to different states that will be used to decide what action should be performed next. The following state translation of the environment used in this thesis is partially based on Ortega's et al.[8] where they save information about the current environment and states of Mario to a list with different conditions that are describing the current situation with information like Mario's current mode, if and where close enemies are lurking, if Mario is close to a cliff etc. The complete list of conditions describing the game environment by being converted to an integer array can be found in the Table 3.1 below. To reduce vaguely created states it is only the area closest to Mario that are checked and translated due to it is where the most critical information is, like for example if there is a enemy or obstacle at the far left side of current scene it will have no or a very small impact on Mario's next decision of action. It would also reduce the precision of the translation because many more states could fit with the same conditions which would lead to the agent having a harder time learning an acceptable behavior due to not being rewarded/penalized consistently. There are thus two grids around Mario where one is used to check for enemies and the other one is used for detecting oncoming obstacles. The size of the grids are 3x3 and 4x4 for obstacles respective enemies, as seen in Figure 3.2. The reason for the larger grid for enemies is that they are not static like obstacles and therefore needs earlier detection to be able to act in an acceptable way.

This information is then later used to create states that are placed in a nondeterministic finite-state machine. The state machine does not keep track of any couplings between the states what so ever, meaning that a state cannot know beforehand which state that could be chosen next. This is due to the fact that a certain state can appear on many different places on the same level. The states also have a list attached to them that store a Q-value for each performable action, which is the measure of how good a certain action is in that specific state.

Index	Condition	Values
0	Mario's mode	0 = small, 1 = big, 2 = fire
1	Is Mario on ground?	0 = no, 1 = yes
2	Is enemy up?	0 = no, 1 = yes
3	Is enemy down?	0 = no, 1 = yes
4	Is enemy left?	0 = no, 1 = yes
5	Is enemy right?	0 = no, 1 = yes
6	Is closest enemy stompable?	0 = no, 1 = yes
7	Is Mario near a cliff?	0 = no, 1 = yes
8	Is Mario over over a gap?	0 = no, 1 = yes
9	Low obstacle infront of Mario?	0 = no, 1 = yes
10	Medium obstacle infront of Mario?	0 = no, 1 = yes
11	High obstacle infront of Mario?	0 = no, 1 = yes

Table 3.1: All conditions has an unique index used to identify them together with a value describing the state of the condition. These conditions can be combined to create up to (3*2Î0) 3072 unique states.

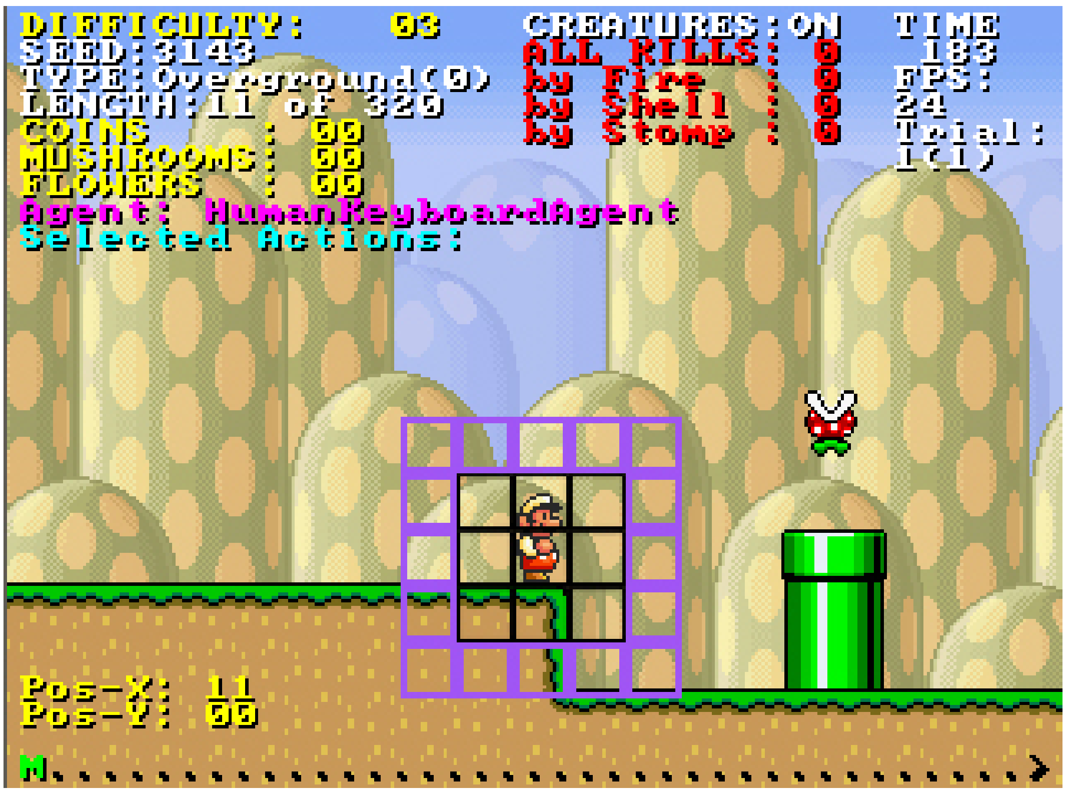

Figure 3.2: The 3x3 and 5x5 sized grids used to limit the area detecting objects and enemies in the world.

3.1.3 Combining and Choosing Actions

This section describes the thoughts of combining the possible action combinations that Mario can perform to ease up the use of more than one action at the same time and as well reduce unnecessary combinations. The implementation of how a new action is chosen is also presented in this section.

Combining Actions

In IMB there are 5 different actions that Mario can perform to make it possible for him to explore and advance in the world. The actions are walking left or right that also can be speeded up with the speed ability and additional abilities are jumping and shooting. All these actions can be combined to perform more advanced movements for example Mario can shoot while jumping left and so on. To simplify the decision of what actions to perform and reduce unnecessary action combinations a complete list of possible action combinations was created for the purpose to use in the reinforcement learning. The complete list of performable combinations can be seen in in Table 3.2. This idea to simplify the action combinations is inspired by the work of Liao et al.[15] where they also tried to remove unnecessary combinations to improve the playing performance of the agent. The possible combinations for the agents in this thesis are the same used by Liao et al. Unnecessary action combinations are those that would do nothing like pressing the left and right key (walking left and right) as the same time that would result in Mario just standing still and doing nothing.

Index	Name	Actions
0	STAND	None
1	LEFT	Go left
2	RIGHT	Go right
3	JUMP	Jump
4	SPRINT_SHOOT	Activate sprint and shoot
5	RIGHT_JUMP	Go right and jump
6	LEFT_JUMP	Go left and jump
7	RIGHT_SPRINT	Run right
8	LEFT_SPRINT	Run left
9	RIGHT_JUMP_SPRINT	Run right and jump
10	LEFT_JUMP_SPRINT	Run left and jump
11	JUMP_SHOOT	Shoot and jump

Table 3.2: All possible action combinations has been dedicated an unique index used in the Q-value table, name that is describing the combination and the actions that should be executed.

Choosing Next Action

A new action will be chosen when Mario has either reached a new state or gotten stuck, but the decision of choosing next action can be done in two ways, either by:

- Random - The new action is chosen in a completely random way and earlier action performance is not considered. This type of choosing action is used when the agent is exploring new actions or when stuck in the world somehow.
- Best Q-value - In this case the best action is always chosen by checking the Q-table for the current state and choosing the action with the best Q-value. Choosing the best action is used both under the learning phase as well as when using the optimal behavior.

To determine how to choose the next action an exploration rate is used and a control is done to see if the agent is in the learning phase or not. An exploration rate is used to decide how often the agent should try new actions and explore the world instead of choosing the best action. If the agent always would use the currently best action it would never learn any new behavior and risk to not find the most optimal way to play. The exploration rate in this implementation was set to 0.4, meaning that 40% of the time a new action should be chosen at random.

When the learning phase is over the exploration is set to 0.0 meaning that the agent will try to use the best action every time a new action should be chosen. The only time when the best action is not chosen in this non-learning phase is when the agent somehow got stuck and cannot advance in the level anymore and in that case a random action is selected to be the new action.

3.1.4 Reward Calculation

When an action has been executed and Mario reached a new state it is necessary to update the value (Q-value) for the performed action. A Q-value is the representation of how good an action is in a certain situation. This calculation can only be done if a couple of rules and criteria are set for the definition of what is a good or bad result for an action. The main goal of IMB is to reach the very right end of the current level to reach a new level while searching for the kidnapped princess. Other tasks that are not equally important but still increases the score in the game is gathering coins, eating mushrooms and completing the map with as much time left as possible. By knowing the ultimate goal and these minor important tasks it is easier to define what behavior should be rewarded and which should be penalized. The reward ($\boldsymbol{R}$) formula is presented in 3.1

$$\begin{aligned} R =&(dX * XRate) + (dY * YRate) + (dC * coinRate) \\ &+ (dM * modeRate) + (dK * killRate) + obstR \\ &+ extraP \end{aligned} \tag{3.1}$$

where $\boldsymbol{dX}$ and $\boldsymbol{dY}$ is the difference of the Mario's positions in the x/y-axis between the old and new states. The number of coins collected since the old state is noted with $\boldsymbol{dC}$ and if any change of Mario's mode has been done it is represented with the $\boldsymbol{dM}$ variable. If any enemies have been killed since last state $\boldsymbol{dk}$ will tell how many. Mario will also be extra rewarded if he was able to overcame an obstacle, $\boldsymbol{obsR}$ but also being penalized a little extra, $\boldsymbol{extraP}$, when making the choice to either walking left or standing still too long. There are also some rate constants that are used to scale the reward for the different attributes mentioned above.

This reward is then used in the Q-learning algoritm, that was presented earlier, to calculate the new Q-value for the chosen action. For a deeper explanation of the equation, please take a look in the chapther reinforcement learning in Topic Background presented earlier in this thesis. As a reminder though the equation will be shown once again here.

$$Q(s,a) = (1-l)Q(s,a) + l * (r + d * max(Q(s_{+1},a))) \tag{3.2}$$

The learning rate ($\boldsymbol{l}$) used in the equation for the reinforcement learning part had its value set to 0.3 and the discount factor ($\boldsymbol{d}$) was set to ($\boldsymbol{0.01}$). A learning rate can have a value between 0 - 1 and the closer to 0 the less important is the reward for the chosen action meaning that the agent will not learn anything new. A value close to 1 increases the importance of the current reward. The discount rate is used to decide how important the Q-value for the best action in the next state is considered and similar to the learning rate a value close to 0 means that it does not take the future reward into any bigger consideration and vice versa with a value closer to 1. The reason for a pretty medium value for the learning rate was due to prevent the first actions in a newly created state to skyrocket its Q-value, only because it was chosen first. Due to the choice of using a non-deterministic finite-state machine (FSM) for managing the states that is not keeping track of any couplings between them it is not of any bigger relevance of how good the best action is in the next state and the discount rate was therefore put to a very low number. For example the first state of a level could possibly also be the same as the last one of that level resulting in that a state could not know if it is closer to the ultimate goal or not than the last state. Due to the good results performed by the reinforcement part that was presented in Ortega's study[8] it was chosen to use the same learning- and discount rate in this implementation too.

3.1.5 The Transition Between States

A transition between states will happen when the environment and states of Mario no longer match the conditions of the current state in the FSM or if Mario has gotten stuck with the current action. If the new conditions already exists in the FSM then the agent will either choose the action with the highest Q-value from the Q-table or choose a new action at random. To decide how to choose the new action an exploration rate is used to define the chance of performing a random action instead of the best one. A higher exploration rate means a higher change of choosing the random selection. If the state has never been visited before its conditions will be added to the state machine and a random action will be chosen. When the action has been chosen and performed it will result in one of either two scenarios:

- **Getting stuck** - Mario is considered stuck if he has not moved in any x-direction for a couple of frames. In this case the chosen action will be penalized due to it hinders Mario to advance in the world. When this happens a new random action will be chosen to hopefully get Mario out of the sticky situation.

- **Reaching a new state** - The most optimal situation is when Mario is reaching a new state which means that the chosen action in some way was successful. How successful the outcome of the action was is calculated by the fitness function used in the Q-learning.

3.2 The Imitation Process

The imitating part of the AI learning is somewhat based on the modified version of the Ghost AI technique, as talked about in earlier chapters, proposed by Lueangrueangroj and Kotrajaras[9] where they by letting the AI agent study a human playing a fighting game learn in which situations to perform certain actions. They gathered statistics from the players containing information about how often an action was performed in a situation and how successful this decision was. These two measurements were together used to calculate the probability for an action to be performed next time visiting the same situation. This method to calculate the probability to choose an action in a specific state will be used to extend the reinforcement learning of the agent by letting it decide next action by using the probability chance for each action instead of the regular Q-value table. The formula to calculate the probability for an action ($\boldsymbol{Pa}$) used in this thesis is the same as presented in the work of Lueangrueangroj and Kotrajaras which is once again showed in Equation 3.3 as a reminder.

$$Pa = Pf + (C * (Pw - Pf)) \tag{3.3}$$

The weight of how good result the action have performed in the state, ***Pw***, is calculated in a percentage (%) of how often the result was good. To decide what good result is and what is not the IL used the reward calculation from the RL that is calculating the reward of the action depending on variables like how far the agent traveled since last action, number of coins gathered and so on. All details of the reward calculation were desribed earlier in this chapter. If the calculated reward was a positive number the weight for that certain action would be increased by one (1) to reward the good behavior. Should it show that the reward for the action was zero (0) or a negative number the weight would be decreased by one (1) to penalize for the bad behavior. This approach to weight the result from the action is similar to the way Lueangrueangroj and Kotrajaras[9] did when they created their learning fighting agent. The calculation of the Pw is done by dividing the total of times the action were performed with the total times the result of the action were successful. This can be seen as a success rate of the chosen action and will have a small effect on the total probability.
The use of this implemented IL is to adjust the RL in two different ways:

- **Adjust the exploration rate** - This rate is set to match the "randomness" of the human play because a human will most likely not make the same decision every time he encounters a similar situation especially when there is a lot going on like when fighting a number of enemies. In a less hectic situation it is probably easier for the player to make a good decision most of the times resulting in a lesser random behavior for that situation. The exploration is calculated by the mean value of the total times the human player performs the same action in a certain situation, calculated by a fraction representing the percent (%) of the times the action is chosen. The rate can have a value between 0 - 1 and a value closer to 1 meaning a higher chance to explore the world and choosing a random action. The standard exploration rate for the RL is set to a constant of 0.4.

- **Adjust the probability to perform an action** - One data that is gathered when the human is playing is how frequently he uses the different actions in a certain state and this data is unique for every state. How the probability for the actions in a state is calculated can be seen in Equation 3.4. This probability is measured in a percentage (%) of how often a certain action compared to the others was used in that specific situation and it will be used together with RL to choose the random action when exploring. This makes the agent explore in a way similar to the imitated human resulting learning a more natural behavior. Although the Q-value is not used to decide which action to explore it will calculate the reward of the random chosen action as usual with the Q-learning algorithm. The Q-value is still used when deciding which action is best in a specific state.

> Every state visited by the human player will have an unique probability list coupled to it to get as natural behavior as possible, but for new explored states that the human never visited was a standard probability list used that was calculated from the imitation session by calculating the mean value of all probability chances for every state visited.

The new exploration rate, ***E***, is calculated by the number of times the most frequently used action was performed, ***bA*** (best action), in the chosen state divided by the total times that all actions have been executed, ***tA*** (total actions), in this state which also is the probability for the best action in that state. This gives the %-of times the most regular action was chosen so to get the exploration rate we need to turn it around by doing 1 - the result. The complete equation can be seen below:

$$E = 1 - (bA/tA) \tag{3.4}$$

Equation 3.4: The equation to calculate the exploration rate for a state i.e. how often another action than the most used one was chosen.

Chapter 4
Method

The purpose of this chapter is to explain the methods that were used during the experiments and the purpose to execute them. Information about how data for the experiments of this thesis were gathered is presented within this chapter with additional information that will describe the prioritized measurements used in the benchmark from the Mario AI Competition. There is several types of data gathered for the experiments and results, which and how is described in the last section.

4.1 Experiment Purpose

The methods chosen to gather the data used in the results that is later analyzed for the purpose to answer the research questions of this thesis are of the types of *empirical experiments*. The experiments are used to gather *quantitative data* which is used to compare the implemented AI to other existing AIs. The result of that comparison is used to analyze if the combination of RL and IL is a feasible technique for a platform game and/or for future research in other game genres as well. The *validity* of this research can be considered good due to it is based on techniques, with small adjustments, that have been used successfully in games before which is supporting the chosen solution to these research questions to be of relevance. The set of tests that is executed to try answering the research questions will also be a measure of how good *reliability* the implementation of this thesis has.

The purpose of these following experiments is to answer the two research questions for this paper which was described in the previous section of this chapter.

4.2 Data Gathering

To be able to do an accurate and reliable comparison there is a set of data that needs to be gathered from the human player and the AI agents playing performance. All data types saved during the playing session of the AI agent are presented in Table 4.1 together with a brief description. The types of data saved during the play of the human are exactly the same as the data gathered during the AI agents plays. The data gathered from the human will be used to evaluate the result from the learning of the RL and IL combined agent so that it does not differ that much from the imitated player. Some of the data which are used to compare to other existing AIs like the ones from the Mario AI Competition 2009 are prioritized but the other data are only used for comparison between the created agents of this thesis and are not prioritized. Data that is not prioritized is noted with an X in the priority column.

4.3 Experiment Execution

The experiments of this thesis are executed in these following 4 steps:

- **Imitation phase** - A set of four players with different gaming experiences will be the teachers for each one of the AI agents that will study the behavior of the humans. The only requirements put on these four players were that they should have played Super Mario Bros before and be familiar with the controls. All players sad that they at least had tested the game before and when studying the results performed by these players it could be assumed that they were telling the truth and had earlier experience with playing Super Mario Bros. This assumption is made due to the fact that all players were able to complete at least one level each. Worth to note is that even if Infinite Mario Bros is used as the playing testbed for this experiment, it is a clone of Super Mario Bros with the same controls and graphics which makes the playing experience identical to Super Mario Bros. All the players will play the same set of 10 levels while data is gathered to be used for adjusting the RL agent and additionally save the results to be used for the comparisons between the humans and the RL and IL combined agents. The data of how often a human perform a specific action in a certain situation is recorded and is translated to a probability list containing the chance of performing the action in the current state. This probability list is later used by the RL agent to get the behavior from the human. The exploration rate, which is how often a player performs another action than the most frequently used one, is also calculated and later used by the RL.

- **Learning phase** - The second step of the experiment is to train the AI agents. The agents running these tests are the four RL agents adjusted with the action probability list and exploration rate from the imitation phase. Additionally an agent with the original standard RL behavior will also perform these tests, for the purpose to compare the performance of the standalone RL against the combination of RL and IL. The agents will play the 10 levels by starting off running these levels 100 times each for the purpose to train themself and learn how to advance in the levels. During the whole learning phase the exploration rate will steadily decrease until the end of the learning phase where the rate will reach 0. When the exploration rate has reached 0 it means that the agent will no longer try to explore the world anymore by taking random or other actions than the best one in the current situation. The only times the agent will still not choose the best action is when it gets stuck in the world and in that case try to perform random actions until it breaks lose. After playing these levels a total of 1000 times it is then time for the next phase.

- **Tuning phase** - This second phase is the adjustment phase which means that the agent will play the 10 levels another 100 times but this time not exploring anything and just trying choosing the best action available. The purpose of this phase is to tweak the agent by using the old learnings from the IL together with RL to improve the playing performance of the agent and develop its behavior even more. The agent will though still continuing to learn for the purpose to adjust some minor details in its behavior.

- **Evaluation phase** - Lastly it is time for the evaluation phase which is executed during 100 runs, 10 playing session per level. At this point the agents will stop its learning and only play with its best behavior. Data gathered from these runs is the main comparison material used to both compare the internal agents of this thesis and compare to the agents from the 2009 Mario AI Competition.

These complete experiment explained above will be iterated/done six times for each of the AI agents, to get at good mean result of the playing performance to reduce the risk of getting bad quality results. Why the number of six iterations is chosen is because of in the work of [19] they found the best evolutionary agent around the sixth run. Although their work is in a little different subject they showed good results with this number of runs and therefore it is used in this experiment too.

Priority	Data	Description
1	Distance traveled	How far Mario traveled on the current level before he either died, ran out of time or finished it.
2	Time left	Time left when Mario completes the level.
3	Enemies killed	The number of enemies killed by Mario on the current level.
X	Number of states explored	Can be used to see how much every agent has explored of the different levels.
X	Number of wins	The total number of wins during the current test. This value can be compared to the data-file from the most previous level to determine if Mario won or died on the current level.
X	Number of deaths	The total number of deaths during the current test. This value can be compared to the data-file from the most previous level to determine if Mario won or died on the current level.
X	Mario mode	The mode Mario ended the level in. If he died or timed ended his mode will be 0. The other possible values, 1 - 3, are representing his states when reaching the goal.

Table 4.1: The table with the data that is gathered during the playing sessions and that is later on used for the comparisons between agents. ***Note:*** Coins are not including due to the limitations of the benchmark which does not differ coins from an empty space in the world. Therefore can the AI agent never detect a coin and take that into consideration when making its decisions.

Chapter 5

Results

Following tables and graphs shows the results for the different AI agents during the three phases of the experiment and additionally the results from the play of the humans. The mean data from the learning-, tuning- and evaluation-phases is presented in separate tables. The complete data from all experiment iterations for each agent in each phase can be found in Appendix A. The next following section is presenting graphs of the development during the complete experiment by showing the mean win rate, number of enemies killed and total distance traveled for each one of the four agents. The purpose of these graphs is to study how the change between the different phases affects the agents. Additionally the result from the comparison against the agents from the 2009's edition of Mario AI Competition is presented last in this chapter.

5.1 Learning Phase

Learning Phase				
Agent	Distance	Kills	Time	Win rate (%)
Human 1	95157	29	1132	60.0
Human 2	35044	7	1390	10.0
Human 3	94527	31	836	60.0
Human 4	104471	40	974	60.0
RL + IL 1	63192	24	1187	30.2
RL + IL 2	48479	19	933	13.7
RL + IL 3	61405	24	1182	29.5
RL + IL 4	63321	23	1328	32.5

Table 5.1: The data for the AI agents was gathered during the learning phase of the experiments. As for the humans the data were gathered during their playing sessions when teaching the AI agents.

The Table 5.1 shows the mean values of the comparison attributes total distance traveled, total number of enemies killed, total time left and the win rate over the ten selected levels. These values were gathered during the learning phase for the AI agents where they used their action probability list, created from the human playing session, to decide the next action. The agents ran every one of the ten levels a 100 times each leading to a total of 1000 runs. The learning phase was executed six times for every AI agent to get data that was most representive to the normal behavior of the agent. The data from the humans was gathered during the imitation phase when they were playing the ten levels for the purpose of teaching the AI agents.

5.2 Tuning Phase

Tuning Phase				
Agent	Distance	Kills	Time	Win rate (%)
Human 1	95157	29	1132	60.0
Human 2	35044	7	1390	10.0
Human 3	94527	31	836	60.0
Human 4	104471	40	974	60.0
RL + IL 1	84049	31	1233	57.6
RL + IL 2	80559	31	1236	49.9
RL + IL 3	81662	28	1234	53.4
RL + IL 4	81261	30	1257	50.7

Table 5.2: The data for the AI agents was gathered during the tuning phase of the experiments. As for the humans the data were gathered during their playing sessions when teaching the AI agents.

The Table 5.2 shows the mean values of the comparison attributes total distance traveled, total number of enemies killed, total time left and the win rate over the ten selected levels while executing the tuning phase. During this phase the agents are choosing their best performable action in each situation as they continue to learn and update their behavior. These values were gathered during a test consisting of running the ten levels 100 times each, resulting in a total of 1000 runs. The tuning phase was executed six times for every AI agent to get data that is most representive to the usual behavior of the agents. The data collected from the human playing sessions was gathered during the imitation phase when they were playing the ten levels for the purpose of teaching the AI agents.

5.3 Evaluation Phase

Evaluation Phase				
Agent	Distance	Kills	Time	Win rate (%)
Human 1	95157	29	1132	60.0
Human 2	35044	7	1390	10.0
Human 3	94527	31	836	60.0
Human 4	104471	40	974	60.0
RL + IL 1	89763	32	1226	65.0
RL + IL 2	79927	29	1275	51.2
RL + IL 3	89708	31	1214	57.2
RL + IL 4	85321	31	1249	61.5

Table 5.3: The data for the AI agents was gathered during the evaluation phase of the experiments. As for the human teachers the data were gathered during their playing sessions when teaching the AI agents.

During the evaluation phase, that is the last phase of the experiment, was the following comparison data gathered; total distance traveled, total number of enemies killed, total time left and the win rate. The Table 5.3 shows the mean values of these comparison attribute that was collected over the selected ten levels that was run ten times each, resulting a total of 100 runs. The evaluation phase was executed 6 times for every AI agent to get data that is close to the mean behavior. When executing this phase the agents will no longer learn anything new or save any additional progress but only use its best behavior which is always choosing the best action for every situation. The data for the humans in the table was gathered during the imitation phase when they were playing the ten levels for the purpose of teaching the AI agents.

5.4 The Complete Experiment

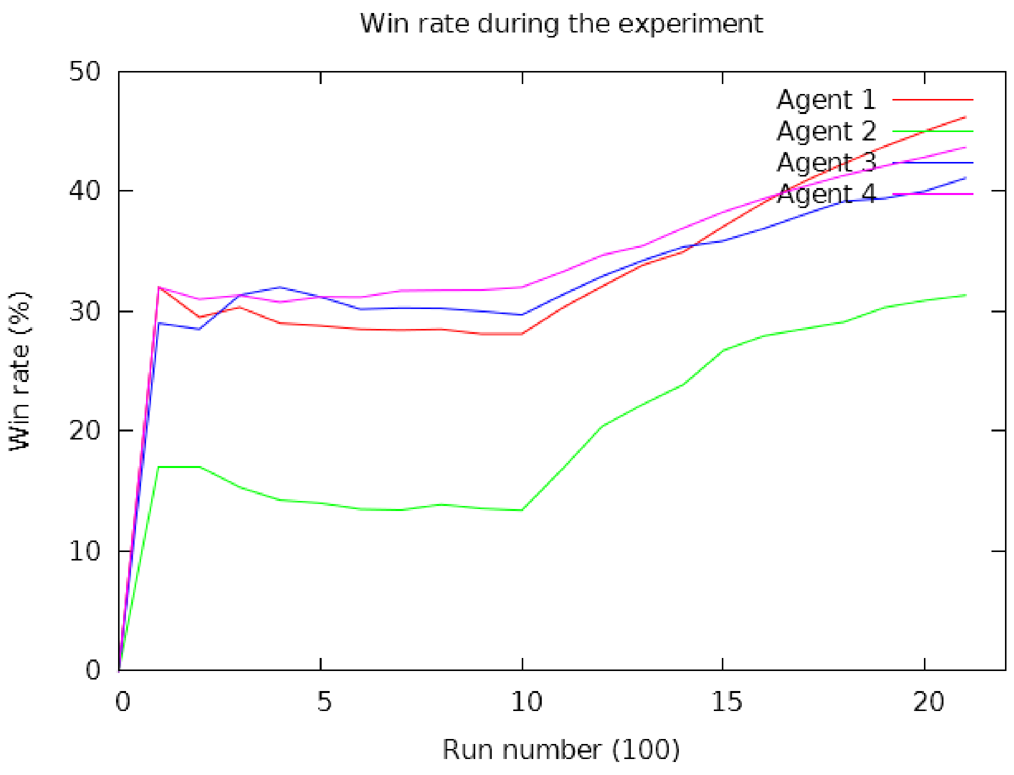

Figure 5.1: The total win rate during the experiment over the three different phases. The first phase ends at iteration number ten, the second phase ends at iteration 20 and the last phase ends at 21. A win-rate sample where gathered for every 100 run (one iteration).

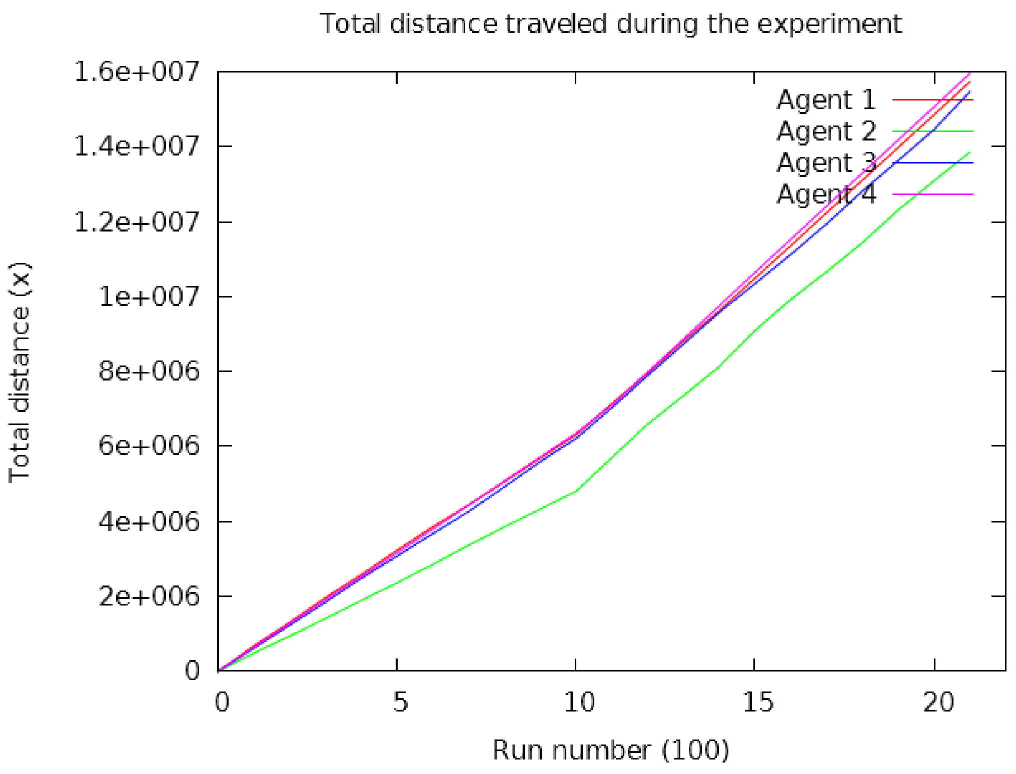

Figure 5.2: The total distance traveled by the agents on the ten selected levels during the experiment over the three different phases. The first phase ends at iteration number ten, the second phase ends at iteration 20 and the last phase ends at 21. A distance sample where gathered for every 100 run (one iteration).

5.5 Mario AI Competition 2009 Results

The way this experiment was executed was to run each one of the 6 versions of the four RL and IL combined agents of this thesis in the same environment as used in the 2009 year's edition of the Mario AI Competition. The best result for each agent was chosen to be used for the comparison in the table. As can be seen in Table 5.4 the best RL and IL combined agent reached the 11th place in the results table and was able to beat five other agents from the tournament. As can be seen on the top two contestants the winner of a tie is decided by the next prioritized data. The data is prioritized in the descending order; distance, time left, kills and mode.

Competitor	Approach	Distance	Levels	Kills	Time left	Mode
Robin Baumgarten	A*	46564.8	40	373	4878	76
Peter Lawford	A*	46564.8	40	421	4841	69
Andy Sloane	A*	44735.5	38	294	4822	67
Trond Ellingsen	RB	20599.2	11	201	5510	22
Sergio Lopez	RB	18240.3	11	83	5119	17
Spencer Schumann	RB	17010.5	8	99	6493	24
Matthew Erickson	GP	12676.3	7	80	6017	37
Douglas Hawkins	GP	12407.0	8	90	6190	32
Sergey Polikarpov	NN	12203.3	3	67	6303	38
Mario Perez	SM, Lrs	12060.2	4	170	4497	23
RL + IL 4	**RL, IL**	**11257.0**	**4**	**90**	**6647**	**33**
RL + IL 3	**RL, IL**	**10920.0**	**3**	**83**	**6313**	**27**
RL + IL 1	**RL, IL**	**9589.0**	**2**	**54**	**6629**	**19**
RL + IL 2	**RL, IL**	**7672.0**	**2**	**55**	**6131**	**19**
Alexandru Paler	NN, A*	7358.9	3	69	4401	43
Michael Tulacek	SM	6571.8	3	52	5965	14
Rafael Oliveira	RB	6314.2	1	36	6692	9
Glenn Hartmann	RB	1060.0	0	8	1134	71
Erek Speed	GA	out of memory	-	-	-	-

Table 5.4: The complete results from the 2009 year's edition of the Mario AI Competition as presented by Karakovskiy and Togelius[20] is showed here together with the agents of this thesis. A* = A star pathinding, RB = rule-based agent, GP = genetic programming, NN = neural network, SM = state machine, LRS = layered controller, GA = genetic algorithm.

Chapter 6

Analysis

There are both promising and less promising results of the experiments which is analyzed in this chapter. In section 6.1 the analysis from the comparisons between the different agents during the different phases is presented additionally with the data gathered over the complete experiment. Next following section, 6.2, is analyzing the comparison between the agents of this thesis with the other agents of the Mario AI Competition. The later section will try answering the earlier stated research questions of this thesis by analyzing the criteria appointed to them.

6.1 Comparisons

6.1.1 The Agents

Learning Phase

Differences between agent and respecitve teacher				
Agent	Distance	Kills	Time	Win rate
RL + IL 1	-31965 (33.6%)	-5 (17.2%)	+55 (4.9%)	-29.8% (50.0%)
RL + IL 2	+13435 (38.3%)	+12 (171.4%)	-154 (11.0%)	+3.7% (3.7%)
RL + IL 3	-33122(35.0%)	-7 (22.6%)	+346 (41.4%)	-30.5% (50.8%)
RL + IL 4	-41150(39.4%)	-17 (42.5%)	+354 (36.3%)	-27.5% (45.8%)

Table 6.1: The differences for the different measurements used to compare the agent with its human teacher, that was gathered during the learning phase. A difference of 5% or less is a good results, differences over 5% up to 10% is considered okay results. All results with a difference of 10% and over are considered bad.

As can be seen in Table 5.1 some of the values actually seem to be approximating similary between the human and its imitating AI agent. For example there is only a 3.7% difference in the win rate between human 2 and the AI agent 2. That agent also performed noticeable worse than the other agents did when comparing the distance traveled, which also the human teacher 2 did compared to the other humans. Another attribute that also is nearly matching each other's is the time value for human 1 and its corresponding AI agent where there is only a difference of about 4.9% (55 seconds) over the total ten levels. The largest difference in this table can be found when comparing the win rate by the AI agent 3 with its teacher human 3 which differs in 30.5%. To summarize there are both promising and less promising results to be found in this phase. The results of the learning phases seems to show that the playing session of an agent compared to its teacher fits better the poorer the teacher performed and that some similarities between the imitating agent and its teacher can be found. Another observation is that the agents that were studying the better performing humans also performed noticable better than the respective poorly performed agent. Although the difference was smaller between the agents than it was between the human players, it seems that the data gathered from the human playing session have affected the performance of every one of the agents.

Tuning Phase

Differences between agent and respecitve teacher				
Agent	Distance	Kills	Time	Win rate
RL + IL 1	-11108 (11.7%)	+2 (6.9%)	+101 (8.9%)	+2.4% (4.0%)
RL + IL 2	+45515 (130%)	+24 (343%)	-154 (11.0%)	+39.9% (399%)
RL + IL 3	-12865(13.6%)	+3 (9.7%)	+398 (47.6%)	-6.6% (11.0%)
RL + IL 4	-23210(22.2%)	-10 (25.0%)	+283 (29.0%)	-9.3% (15.5%)

Table 6.2: The differences for the different measurements used to compare the agent with its human teacher, that was gathered during the tuning phase. A difference of 5% or less is a good results, differences over 5% up to 10% is considered okay results. All results with a difference of 10% and over are considered bad.

Compared to the learning phase there have been some major changes of the results during this phase, as can bee seen in Table 6.2. Every agent has improved all their respective values (for the exception of the third AI agent which decreased its time left value by 50 seconds, i.e. 4%. This has resulted in that the differences between the agent of the poorly performed human player have increased from a 3.7% difference of the win rate up to a difference of 39.9%. At the same time, the differences for the other agents compared to their teachers have decreased. Most noticeable is the complete correctness of the number of kills between human 3 and its agent that both managed to kill 31 enemies over the ten levels during the tuning phase. Another vast improvement for that agent was also done in the win rate section where it earlier had the largest difference (30.5%) but is now the lowest one with a difference of only 2.4%. During this tuning phase there have been some improvements in the behavior compared to the teachers for three of the four agents, but for the other agent there have been some increased differences compared to its teacher. The analysis of this phase suggests that the better the human teacher performed during the imitation stage, the lower is the differences of its values compared to the values of its learning agent during this phase.

Evaluation Phase

Differences between agent and respecitve teacher				
Agent	Distance	Kills	Time	Win rate
RL + IL 1	-5394 (5.7%)	+3 (10.3%)	+94 (8.3%)	+5% (8.3%)
RL + IL 2	+44883 (128%)	+22 (314%)	-115 (8.3%)	+41.2% (412%)
RL + IL 3	-4819(5.1%)	+0 (0%)	+378 (45.2%)	-2.8% (4.7%)
RL + IL 4	-19150(18.3%)	-9 (22.5%)	+275 (28.2%)	+1.5% (2.5%)

Table 6.3: The differences for the different measurements used to compare the agent with its human teacher, that was gathered during the evaluation phase. A difference of 5% or less is a good results, differences over 5% up to 10% is considered okay results. All results with a difference of 10% and over are considered bad.

This is the last and shortest phase of the experiments that lasts for ten runs each of the ten levels. In this phase the agents are using the best behavior they have learned and this is shown by their results that is presented in Table 6.3. For three of the agents have the total distance closed in on the value of the respective human teacher. The number of kills and time left values has stabilized as well, and does no longer change its value too distinctively. The win rate for the three mentioned agents has also been increased and even more closing in on the wanted values from the human teachers since the last phase. A couple of values for some of the agents has even passed the wanted value. Overall they have values for all attributes close to their respective teacher's and could possible behave in a natural way. As for the fourth agent, named "RL + IL 2", there has been no additional progress and it has basically stagnated in its behavior development. This agent now has a 41.2% higher win rate than the human it tried to imitate had, which is not that good considering the big difference. Overall this phase has shown that at least two out of four agents have nearly reached the main statistics, distance and win rate, of their teachers and could possibly have some natural behavior. The common denominator for these three agents is that they have studied players that performed arguable well during their playing session and that the other agent that does not remotely resemble its teacher studied a player that played poorly.

Complete Experiment

As seen in Graph 5.1, the win rate for all cases are showing a noticeable change around the 1000th run. This is most likely due to the change from learning phase to tuning phase. During the learning phase the next action was chosen by the action probability list for the purpose of teaching the agent a behavior as similar to the play of the human as possible. The purpose with the next phase, tuning phase, was to tweak the agent with the help from the RL to perform better. In many cases this was the truth but it was affecting the poorly performed agent so much that it more than doubled its win rate which was on a similar level to its teacher before the tuning phase. For the other agents they had a similar increase rate of their win rate before reaching the matching win rate of their human teachers. Due to this similar increasing of better behavior for all four agents with the steady and similar increase of the win rate it is possible that the imitation learning did only affect the RL part in minor ways.

Although before that switch between the phases one of the agents actually was approximately matching the win rate of the play of its human teacher, with a difference of only 3.7% as can be seen in Table 5.1. For others it performs well below the playing performance of the human but still noticeable higher than the low performing AI agent. This indicates that there actually exist some similarities between some of the agents and their human teachers which mean that the IL during the first phases to at least some smaller extent works.

During the evaluation phase one can see that the win rate development has started to stagnate for agent 2 but is still steadily increasing for the other agents. This can suggest that the IL in the end actually in some way has affected the learning curve of the RL to match the performance of the imitated human.

As can be seen in Graph 5.2 the total distance traveled by the agents is proportionally increasing as the win rate increases. The biggest change can be seen around iteration ten where the distance traveled for the second agent started to close up to the other agents distance as well as the win rate did.

6.1.2 Mario AI Competition 2009

The reason why the agents of this thesis did not perform any better against the other AIs, as can be seen in Table 5.4, is due that they were not trained for the levels of the competition. The only training they had were from the ten selected maps that the human players played that the agents played a total of 2100 times. One other holdback was that the ten maps that the agents used for training had the lowest set of difficulty which reduces the numbers of enemies, gaps, obstacles etc. but the 40 levels of the competition had an increased difficulty every ten levels. This made it even harder for the agents to compete fairly against the other agents of the tournament. Despite this they still managed to reach 11th to 14th place and were able to beat five of the other agents. When comparing against the closest better placed agents we can only see a small difference in distance traveled, which is the highest prioritized value. Even up to the 7th placed agent the distance traveled performance was as low as 11.2%. One funny note is that there are no other, either RL or IL agents, except the agents of this thesis, in the whole tournament. With a better and more variation of the training the results could have been even greater. Due to all these hazards the total results of this comparison could be considered to be good for all the agents of this thesis.

6.2 Analysis of the Research Questions

The answer to the two research questions which is the main purpose of this thesis is presented below.

- **RQ1:** Is imitating a human player using Reinforcement learning a feasible technique for creating a learning based AI in Infinite Mario bros?

Answer: Short answer: yes and no. The good result was that the combination managed to create agents that performed well compared to the agents from the Mario AI Competition despite the lack of qualitative training. In this case did RL and IL complement each other well due to RL tweaking the behavior from the IL part to adapt to new environments. The less promising results showed that none of the agents could perform similar to its human teacher during all the different phases which would lead to the first criteria failed and as a result to that not be able to consider the combination of RL and IL to be a completely feasible technique. As noticed by the results it was not possible to integrate the modified Ghost AI technique, as presented by Lueangrueangroj and Kotrajaras[9], directly into the Q-learning to create a feasible combination. This is probably due the different approaches when deciding the next action and calculating the reward for the RL respective the IL part. With the RL part the agent wants to find an optimal way to the current goal which in the case of IMB is to reach the rightest side of the level as fast as possible to be able to reach the next level. In the case with the IL part it wants to mimic a certain behavior that not always needs to fit the goal of the RL. Therefore the goal of the RL and IL parts of this thesis' implementation does not always point in the same direction giving the agents a hard time deciding the next action. A possible solution to this problem would be if the IL part could affect the reward calculation in the RL part to make the goals of the two techniques match some more.

Although there were still some promising results due to some of the agents actually performed similar to the human teachers during periods of the experiment. Additionally the agents also complied with the first criteria of completing at least some of the levels finished by the human players. Due to the result showing that the agents at least at the end of the experiment either had a close matching or a much higher win rate compared to their respective human teacher the criteria can be considered to be met.

- **RQ2:** How does the proposed AI perform against the bots from the 2009 Mario AI competition?

Answer: The agents of this thesis actually performed well compared to the other agents and managed to reach the places 11th to the 14th. Despite the lack of qualitative training before the competition all agents were able to beat five of the agents from the competition. The best performing agent of this thesis was only 11.2% behind the agent on the 7th place regarding the highest prioritized data, total distance traveled on the levels.

Chapter 7

Conclusions and Future Work

To summarize the research of this paper one could say that both promising and some less promising results were found during the experiments. One positive result was gathered from the research with the purpose to answer how good the agents performed against the agents from the Mario AI Competition 2009 year's edition, which is the RQ2. It showed that they performed well despite the lack of suitable training before entering the contest. All the agents were able to beat five other agents and reached the places from 11th to 14th. The best one of this paper's agent had only traveled about 11% less distance, which is the main measurement, than the agent on the 7th place. This indicates that the combination of RL and IL can perform well against other existing implementations even with small amounts of training.

Although the positive results from the earlier mentioned comparison, the answer to RQ1, which proposed the question if it was feasible to combine RL and IL to create a learning AI agent, could not be explicit stated with a positive answer due to the unstable behavior of the agents during the different phases. The main conclusion from the experiment is that the different phases fitted different types of agents. During the learning phase the agents used their action probability list gathered from the human playing session and the result from this phase showed that agents with more poorly performing teachers was able to learn a more similar behavior compared to its teacher than the agents that had a better performing teacher did. Although the IL affected the other agents as well as they performed noticeable better than the poorly performed agent. During the second phase all agents improved their playing performance drastically and the agents with a well performing teacher were now matching their teachers in terms of winning rate. The poorly performing agent also increased its performance but was also increasing its difference in behavior compared to its teacher. During the last phase the poorly performing agent have stagnated in its behavior development and reached its optimal state, as in contrast to the other better performing agents continued to improve their behavior and actually in some cases surpass their teachers playing performance. The conclusion is that the RL and IL combination in its current state is not able to handle different performing agents to match their teachers during all phases of the experiment and that adjustments to the combination are needed for further work.

A suggestion of future work could be to enhance the combination of RL and the IL technique Ghost AI that was used in the implementation of this paper. As for now though it is not possible to completely state that RL and IL is a feasible technique before some adjustments are made. The first adjustment that is needed is to make it possible for the IL algorithm to update and tweak the reward-function of the RL part. As of now it is not possible which makes the RL static and not able to adapt to other behaviors than the standard one. The currently version of RL has the goal to complete the level as fast as possible, which was the target in the Mario AI Competition. This leads to problems/conflicts when humans play with a different goal in mind, when he for example wants to gather as many coins as possible or trying to kill all enemies before advancing on the level. If the reward calculation of RL cannot adapt to other behaviors than the most optimal one it will penalize the behavior of the human which results in hindering the AI agent to mimic the human correctly. Due to the results of this thesis that are still showing some promising results of creating natural human behavior by the use of the combination of RL and IL it is possible that these minor changes could possible increase the playing performance and creating agents with a more natural behavior.

Another interesting idea for future work would be to study how fast an agent with regular RL can learn to complete a certain task compared to an agent using the combination of RL and IL. In would be good to know how time-consuming these solutions are in different situations.

Due to some promising results it is motivated to continue research in the area of the combining RL and IL for the use of creating an adaptable and learning AI agent for platform games and hopefully for other game genres as well.

References

[1] Kirsten Acuna. 'Grand Theft Auto V' Cost More To Make Than Nearly Every Hollywood Blockbuster Ever Made. *http://www.businessinsider.com/gta-v-cost-more-than-nearly-every-hollywood-blockbuster-2013-9*, 2013. Retrieved 2014-02-12.

[2] Jonathan Downin. Trine crosses 1.1 million sold. *http://www.gamespot.com/articles/trine-crosses-11-million-sold/1100-6347196/*, 2011. Retrieved 2014-02-12.

[3] Brian Schwab. *AI Game Engine Programming.* Course Technology, Boston, USA, second edition, 2009.

[4] Christopher Watkins. *Learning from Delayed Rewards.* PhD thesis, King's College, London, England, May 1989.

[5] Richard Sutton and Andrew Barto. *Reinforcement Learning: An Introduction.* MIT Press, London, England, first edition, 1998.

[6] John McCullock. A Painless Q-Learning Tutorial. *http://mnemstudio.org/path-finding-q-learning-tutorial.htm*, 2010. Retrieved 2014-04-04.

[7] Renzo De Nardi Julian Togelius and Simon M. Lucas. *Towards automatic personalised content creation for racing games.* PhD thesis, University of Essex, United Kingdom, 2007.

[8] Julian Togelius Juan Ortega, Noor Shaker and Georgios N. Yannakakis. Imitating human playing styles in Super Mario Bros. *Entertainment Computing*, 2012.

[9] Sarayut Lueangrueangroj and Vishnu Kotrajaras. *Real-Time Imitation Based Learning for Commercial Fighting Games.* PhD thesis, Chulalongkorn University, Thailand, 2008.

[10] Chin Kim On Ng Chee Hou, Niew Soon Hong and Jason Teo. Infinite Mario Bross AI using Genetic Algorithm. *IEEE Conference on Sustainable Utilization and Development in Engineering and Technology*, 2011.

[11] David R. Wright. Finite State Machines. *http://www4.ncsu.edu/ drwrigh3/docs/courses/csc216/fsm-notes.pdf*, 2005. Retrieved 2014-04-04.

[12] Sergey Karakovskiy Julian Togelius and Robin Baumgarten. The 2009 Mario AI Competition. *IEEE Congress on Evolutionary Computation*, 2010.

[13] Julien Soler Pierre De Loor Fabien Tencé, Laurent Gaubert and Cédric Buche. Chameleon: online learning for believable behaviors based on humans imitation in computer games. *Comp. Anim. Virtual Worlds 2013*, 24:477–496, 2013.

[14] Chao-Cheng Chen Jyh-Jong Tsay and Jyh-Jung Hsu. Evolving Intelligent Mario Controller by Reinforcement Learning. *IEEE Conference on Technologies and Applications of Artificial Intelligence*, 2011.

[15] Kun Yi Yizheng Liao and Zhe Yang. *CS229 Final Report Reinforcement Learning to Play Mario.* PhD thesis, Stanford University, USA, 2012.

[16] Shunsuke Shinohara, Hiroharu Kawanaka Toshiaki Takano, Haruhiko Takase, and Shinji Tsuruoka. Search algorithm with learning ability for mario ai — combination a* algorithm and q-learning —. *13th ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing*, 2012.

[17] Richard Evans. *AI Programming Wisdom*, chapter Chapter 11.2: Varieties of Learning, pages 567–568. Newton, USA, first edition, 2002.

[18] Worapoj Thunputtarakul and Vishnu Kotrajaras. *DATA ANALYSIS FOR GHOST AI CREATION IN COMMERCIAL FIGHTING GAMES.* PhD thesis, Chulalongkorn University, Thailand, 2007.

[19] Jan Koutník Julian Togelius, Sergey Karakovskiy and Jurgen Schmidhuber. Super mario evolution. *IEEE Symposium on Computational Intelligence and Games*, 2009.

[20] Sergey Karakovskiy and Julian Togelius. *The Mario AI Benchmark and Competitions.* PhD thesis, St. Petersburg State University, Russia, 2012.

Appendix A

Agents data

A.1 Learning Phase

Agent RL + IL 1				
Iteration	Distance	Kills	Time	Win rate (%)
1	63083	24	1187	28.1
2	62813	24	1191	32.1
3	63890	25	1183	30.3
4	61942	22	1200	28.9
5	63800	25	1180	30.2
6	63622	25	1179	31.8

Table A.1: The data from the AI agent which ran the learning phase of the experiment six times to get a good mean value. This agent is created by the data from the first human player.

Agent RL + IL 2				
Iteration	Distance	Kills	Time	Win rate (%)
1	48000	19	937	13.4
2	48768	19	928	13.5
3	49225	19	917	14.0
4	48824	19	924	14.2
5	48042	18	949	13.3
6	48016	18	943	14.0

Table A.2: The data from the AI agent which ran the learning phase of the experiment six times to get a good mean value. This agent is created by the data from the second human player.

Agent RL + IL 3				
Iteration	Distance	Kills	Time	Win rate (%)
1	62094	25	1147	29.7
2	62089	24	1150	29.1
3	61457	25	1162	30.1
4	60226	23	1176	29.4
5	61521	23	1288	29.2
6	61045	23	1171	29.6

Table A.3: The data from the AI agent which ran the learning phase of the experiment six times to get a good mean value. This agent is created by the data from the third human player.

Agent RL + IL 4				
Iteration	Distance	Kills	Time	Win rate (%)
1	63496	27	1331	32.0
2	63194	22	1330	32.8
3	63325	22	1325	33.0
4	63763	22	1323	32.1
5	63091	23	1330	33.2
6	63059	21	1331	31.8

Table A.4: The data from the AI agent which ran the learning phase of the experiment six times to get a good mean value. This agent is created by the data from the fourth human player.

A.2 Tuning Phase

Agent RL + IL 1				
Iteration	Distance	Kills	Time	Win rate (%)
1	85376	33	1219	61.9
2	82311	31	1208	52.2
3	84718	35	1189	51.9
4	71281	26	1345	48.8
5	94509	31	1169	64.8
6	86100	28	1270	58.2

Table A.5: The data from the AI agent which ran the tuning phase of the experiment six times to get a good mean value. This agent is created by the data from the first human player.

Agent RL + IL 2				
Iteration	Distance	Kills	Time	Win rate (%)
1	82836	32	1187	48.4
2	79754	34	1247	47.7
3	79962	24	1252	46.2
4	77804	25	1317	48.4
5	83321	38	1167	56.3
6	79677	31	1243	51.9

Table A.6: The data from the AI agent which ran the tuning phase of the experiment six times to get a good mean value. This agent is created by the data from the second human player.

Agent RL + IL 3				
Iteration	Distance	Kills	Time	Win rate (%)
1	82403	30	1233	50.3
2	88893	33	1199	59.9
3	77585	27	1294	51.4
4	82400	25	1253	51.2
5	79566	26	1149	55.1
6	79122	27	1276	52.9

Table A.7: The data from the AI agent which ran the tuning phase of the experiment six times to get a good mean value. This agent is created by the data from the third human player.

Agent RL + IL 4				
Iteration	Distance	Kills	Time	Win rate (%)
1	87052	19	1227	53.7
2	82278	35	1259	55.7
3	81610	28	1270	47.9
4	77333	30	1314	47.6
5	79983	29	1268	50.9
6	79310	40	1203	48.4

Table A.8: The data from the AI agent which ran the tuning phase of the experiment six times to get a good mean value. This agent is created by the data from the fourth human player.

A.3 Evaluation Phase

Agent RL + IL 1				
Iteration	Distance	Kills	Time	Win rate (%)
1	87544	31	1223	70.0
2	98759	39	1106	70.0
3	90333	35	1212	60.0
4	68958	27	1396	50.0
5	105734	30	1113	80.0
6	87252	28	1297	60.0

Table A.9: The data from the AI agent which ran the evaluation phase of the experiment six times to get a good mean value. This agent is created by the data from the first human player.

Agent RL + IL 2				
Iteration	Distance	Kills	Time	Win rate (%)
1	76160	31	1317	40.0
2	83972	42	1185	58.0
3	84876	22	1236	59.0
4	87795	22	1251	50.0
5	74417	36	1297	60.0
6	72339	21	1361	40.0

Table A.10: The data from the AI agent which ran the evaluation phase of the experiment six times to get a good mean value. This agent is created by the data from the second human player.

Agent RL + IL 3				
Iteration	Distance	Kills	Time	Win rate (%)
1	100842	54	1068	63.0
2	100870	34	1137	60.0
3	75532	24	1342	50.0
4	90219	24	1239	60.0
5	81684	23	1266	60.0
6	86102	25	1237	60.0

Table A.11: The data from the AI agent which ran the evaluation phase of the experiment six times to get a good mean value. This agent is created by the data from the third human player.

Agent RL + IL 4				
Iteration	Distance	Kills	Time	Win rate (%)
1	88697	22	1234	60.0
2	94417	47	1183	80.0
3	93035	21	1211	60.0
4	78512	22	1329	59.0
5	76807	28	1322	50.0
6	80460	45	1213	60.0

Table A.12: The data from the AI agent which ran the evaluation phase of the experiment six times to get a good mean value. This agent is created by the data from the fourth human player.