UNIVERSITAT POLITÈCNICA DE CATALUNYA BARCELONATECH

UNIVERSITAT ROVIRA i VIRGILI

UNIVERSITAT DE BARCELONA

MASTER IN ARTIFICIAL INTELLIGENCE
FACULTAT D'INFORMÀTICA DE BARCELONA (FIB)
FACULTAT DE MATEMÀTIQUES I INFORMÀTICA (UB)
ESCOLA TÈCNICA SUPERIOR D'ENGINYERIA (URV)

MASTER THESIS

Ethical conversational agent for performing in-game user experience questionnaires

Eric Roselló Marín

Supervisor: Dr. Maite López Sánchez
Departament de Matemàtiques i Informàtica (UB)
Co-supervisor: Dr. Inmaculada Rodríguez
Departament de Matemàtiques i Informàtica (UB)

May 17, 2022

Abstract

User experience questionnaires are used at the end of a playtesting session to assess the quality of video games and improve their design. Performing a survey after the experience can lead to the user not remembering details and undergo boring and survey fatigue. We propose to integrate the survey in-game, by means of a conversational agent. We aim at endowing this agent with a value-aligned behaviour, using our definition of the moral value of *respect* to guide the agent not to disturb the user engagement. Endowing this value by means of reinforcement learning and the *ethical embedding* algorithm, which redesigns an environment to promote ethical behaviour, ensuring the agent learns to pursue its individual objective while fulfilling the ethical objective. A review of the state-of-the-art showcases that the novelty of our work is twofold: firstly, the application of *ethical embedding* outside toy problems; and secondly, the enrichment of a survey oriented conversational agent with this moral value. Results highlight the advantages of the embedding of a moral value in front of an agent with no ethical notion, with the ethical agent improving to avoid disturbing engagement compared to the unethical counterpart, setting the pavement for future ethical conversational agents.

II

Contents

List of Figures

List of Tables

Chapter 1

Introduction

For decades, one of the objectives in the field of artificial intelligence has been to develop agents capable of coexisting in harmony with people and other systems. Literature highlights the importance of considering social interactions to develop artificial agents [5]. Recently, the attention has shifted towards how morality can be embedded in these systems, ensuring an ethical behaviour from the agents, a topic known as *machine ethics*.

Nowadays, given the inclusion of artificial intelligence in our environment, through means such as unmanned vehicles, intelligent houses, humanoid robots or as avatars in games and virtual worlds, to name only a few, the necessity of machine ethics is more evident than ever. With the rapid increase of the role of technology in our lives, it is becoming more important to build agents capable of engaging in moral behaviour for this technology to cope with our values and motivations [6]. Therefore, the alignment with moral values is a particularly challenging task when trying to ensure that agents learn to behave ethically. A variety of models have been recently introduced that aim to tackle how these values can be embedded in the design of an agent.

It is within this context that lecturers from the University of Barcelona (UB) proposed this master's thesis, as a point where several of their lines of research converge. On the one hand, the UB has been collaborating with the Artificial Intelligence Research Institute (IIIA), a research center of the Spanish National Research Council (CSIC), in the development of formal and algorithmic tools that build upon Multi-Objective Reinforcement Learning to design ethical environments for agents to learn in, ensuring value-alignment

[2] when introducing a moral value in social dilemmas. On the other hand, the UB has also been working on gamification and *serious games* [7, 8, 9], those designed both to be attractive and appealing as well as to meet specific educational goals [10]; as well as on how machine learning can improve *conversational agents*, focusing recently on their integration in virtual reality (VR) experiences for gathering user feedback [7], and on incorporating sentient capabilities similar to the ones humans have [11].

The development of games and Virtual Reality (VR) experiences comes hand in hand with usability and user experience questionnaires, a widely popularized and versatile tool in Human-Computer Interaction (HCI). Researchers at the UB aim to integrate these questionnaires, normally carried out after the experience, as part of the actual game. This could be achieved by integrating an agent into the experience that carries out the survey. For example, a conversational agent that accompanies the user, as part of the narrative, asking questions in-world while users are immersed in the experience.

This is specially important in the context of VR environments, where the conversational agent can be introduced as a virtual character, a facilitator for the conversational in-world questionnaire. It would allow for a seamless experience and provide a more natural way for the user to interact when compared to the manipulation of 2D widgets in VR [7]. Introducing the questionnaire in the game could improve the response the user has towards it, as answering all items at the end can lead to the user not remembering some of the details of the experience, as well as become tiresome and result in fatigue in some situations [12]. But, nevertheless, introducing the survey in-game should be done respecting as much as possible the user engagement and immersion.

The motivation behind the thesis is to bring these different lines of research closer. To build a simple conversational agent to carry out a game experience questionnaire, while having a notion of the ethical value of *respect*. This value should guide the agent in order not to disturb the user experience, using the aforementioned tools on Multi-Objective Reinforcement Learning to design an environment that ensures the alignment with this value. This would transform the environment into one in which the ethical value is to be fulfilled effectively *embedding* the ethical value in the agent. We will consider *respectful* to ask questions when user engagement is minimum. To do this, we need to identify the different stages of engagement in a game, to confirm that the agent learns to interact with the player only when opportune.

Notice that it is not our objective to develop a fully fledged conversational agent integrated inside a complex game. There is vast literature on how to build chatbots to improve the user experience [13, 14]. In this work, we focus on providing a value-based moral notion to our agent, so it is able to recognize when is the right time to question the user.

Our approach of incorporating moral values into a conversational agent that carries out a survey has not been previously explored in the literature. Most similar article we have found is a theoretical discussion that sets the pavement for the introduction of moral values into pedagogical agents for online learning environments [15]. The algorithm used for the embedding of the ethical value has also not been explored outside the context of moral dilemmas, much less used in an application that interacts with the user. Therefore, the contribution and motivation of this master thesis is in part based on the novelty of our application. We hope to set the first steps towards the development of ethical conversational agents capable of performing in-game surveys, for games and VR experiences.

Given the novelty of our objective, and the variety of research areas combined, Chapter 2 will be dedicated to a review of the state-of-the-art of these lines of work. We will showcase the different approaches that have been recently used in incorporating reinforcement learning in games, the state of conversational agents for information elicitation, and the different morality systems proposed in the field of machine ethics. This review will help not only to introduce the reader to these different fields, but also to highlight the difficulties in bridging the gap between them.

In Chapter 3, we will explain the design choices and simplification required to bring these lines of work closer, describing how we have approached our problem. There is in fact an example in the health sector of a conversational agent that performs a questionnaire expressing questions in natural language and interpreting the user's response to map it to a Likert scale [16]. In our case, our focus is not on the aspect of performing the survey using natural language, but on how to integrate the agent in a game while ensuring an ethical behaviour. It is outside the scope of this master thesis to build a virtual character capable of carrying out a conversation in natural language to perform a questionnaire, while also integrating machine ethics. Given how value-based morality systems in research have mainly been applied to social dilemmas, jumping from these scenarios directly to natural language conversations would greatly exceed our initial objectives.

Chapter 4 will then detail the morality system we are to use in incorporating a notion of a moral value of *respect* in our basic conversational agent. We describe, following a recent methodology, how the environment is built in order to ensure that our agent learns to behave ethically. This includes the definition of the Markov Decision Process considered, the learning algorithm, the specification of the different objectives and rewards, and how to solve the *ethical embedding problem* to align the agent with our ethical value.

Subsequently, Chapter 5 showcases the results obtained from this embedding, where different metrics are selected in order to demonstrate we have achieved the desired behaviour. We compare our ethical agent to an agent with no notion of *respect*, with our agent recognizing periods of low engagement as the best situations to question the user, performing better at avoiding to disturb engagement.

Finally, Chapter 6 will present the conclusions obtained and set the steps to be taken in the future in order to increase the complexity of the system, extending its capabilities, and to incorporate it as a virtual character in new games.

Chapter 2

State-of-the-art

There is little to no literature on the alignment of ethical principles in conversational agents through the use of values and norms. Such a specific topic is addressed only in articles that aim to pave the way for future research, suggesting the necessary steps to take in order to provide these agents with a notion of ethics using morality systems [15], without discussing design or implementation details.

Given the novelty of the application we develop and how different disciplines and lines of research contribute, we start this master thesis with a review of the state-of-the-art in order to provide the reader with the necessary context. Therefore, we introduce the current state of three lines of research and their respective perspectives on the topic at hand:

- The application of Reinforcement Learning in games and, particularly, in education, given the increasing availability of online education systems, we introduce how RL is used to adapt and personalize the students' experience to their needs, focusing on improving learning gains while maintaining the motivation and engagement.

  We present the main approaches used to obtain this goal, as well as the challenges these have, and how the recent tendency points towards pedagogical agents, capable of both adapting the curriculum to the learners' needs and provide support in order to enrich the experience.

- Conversational agents, which allow for a more natural interaction, increase engagement from the user in a variety of tasks. We will highlight

the different types, the approaches for their design and some recent applications.

Specifically, we are interested in how these agents are being applied for performing interviews, and their effectiveness in information elicitation, carrying out questionnaires as conversations, promoting flexibility for user responses and engagement due to how these agents recreate social interactions.

- Machine ethics, a field that has gained attention in the last decade, focused on how to embed moral values in agents. We are to showcase the different approaches taken on how to provide agents with morality systems that encode these values.

  This review will allow us to better locate the approach we will be using in the following chapters to incorporate an ethical value into our agent and illustrate the challenges to be expected during our implementation.

2.1 Reinforcement Learning for games

Reinforcement learning has achieved incredible results in game playing in the recent years, but the application on video games goes beyond achieving agents that perform better than humans at playing video games, AI can help improve the design of games, as well as provide insight on how players interact with them [17]. For instance, AI has been used to adapt the experience to the user's profile, which has been particularly interesting for the application in serious games, adapting the curriculum content to help improve the learning process and gains of students [10].

The purpose of serious games is not only to be fun and entertaining, but also educational, appealing to an audience while also meeting some educational goals. Their value is not only academic, as they provide skill practice and entertainment through exposure. We find a wide range of artificial intelligence techniques being applied to serious games in order to improve the experience, with its main purpose found in educational games and in training for the health sector. A possible classification for the most popular techniques is provided in [10], in which two main categories are considered: decision-making and machine learning.

The first, decision-making, focuses on exploiting previously collected knowledge in order to determine the following action. The techniques and algo-

rithms in this category are very popular in the development of serious games, mostly to model the player experience using elements from the gameplay (from the interaction between the user and the game), procedural content generation (PCG) and game data mining. This category would include techniques such as:

- Decision Trees, used primarily to customize the gameplay experience based on the user progress and previous interaction with the game [18], modelling what is called *game-flow*. They are also used to trace the motivation and users' state, providing the system with information on the users' behaviour [19].
- Fuzzy Logic, also popular to adapt the gameplay experience or to classify users given their interaction [20].
- Markov Systems, mostly used to collect information from the player by following the users' state and modelling their actions [21].
- Goal-Oriented Behaviour, as with commercial games, used for controlling NPC behaviour [22].
- Rule-Based Systems, assisting decision-making and tactical actions in games [10], are used to evaluate the user given some guidelines or to use them to guide the player [23].
- Finite-State Machines, commonly user to control game-flow, for example, switching between levels, but also controlling NPCs behaviours and virtual characters [24, 25].

The second category, machine learning, optimizes performance by learning from experience in order to improve how a task is carried out. Its main disadvantage in serious games is the amount of usage time needed to train effective systems. These techniques are also used mostly to model the player experience using elements from the gameplay and game data mining, but are being applied also in order to model the experience using data collected by monitoring the players' physiology. In this category, we find techniques such as the following:

- Naive Bayes Classifier, used to analyse the players' behaviour and classify them given their performance [26].

- Artificial Neural Networks, mostly focused on altering the game-flow, but also for user evaluation and classification as well as for controlling NPCs [27, 28, 29].

- Case-Based Reasoning, mainly to customize the contents of the game automatically [30].

- Support Vector Machines, to predict, evaluate or classify the players given their behaviour [31].

The increase in interest in these techniques for serious games is partly because of the success when adapting the game-flow to customize the experience, being computationally efficient and not too intrusive. Serious games that dynamically adapt to the needs and performance of the player have proven to be, in some cases, more efficient when comparing to applications that do not adapt [10].

It is this second category that would contain reinforcement learning (RL) approaches. RL emphasizes learning by an agent from direct interaction with its environment [32], by having the agent learn a policy dictating what to do in different situations. Being educational games the most popular purpose of serious games, and due to the sequential nature of the interactions between teacher and student, RL has seen a surge of interest in its application for education.

The widespread presence of online education systems leads to the possibility of personalized learning at scale. This makes high-quality education more accessible and highlights a need for these educational systems to tailor the curriculum to each student's needs [33].

As an example of how this is being carried out, we can look at some lines of work presented at the Educational Data Mining conference from 2021 on Reinforcement Learning for Education:

- Personalizing curriculum across tasks, just as with the rest of artificial intelligence techniques in education, the most popular application is to adapt the content provided to the user, in this case working as a tutoring system. A policy can map the student's history of responses to the next task to be performed to maximize learning [34].

- Providing hints, scaffolding, and quizzing; train policies to provide hints

during a task, specially useful in scenarios where scaffolding is important in the learning process, helps to maintain engagement [35].

- Adaptive experimentation and A/B testing, based on observing the student's outcome so that next students receive a version that has been more effective previously [36].
- Model a human student, useful to design more effective feedback from the teacher, as well as to use as a simulation to evaluate the policies used to teach [37, 38].
- Content generation, or procedural content generation, to come with more material for the students [39].

Developers at the UB have worked on block-based visual programming games, to teach users programming principles, a popular task on which to apply RL, mainly to decide which problem the student should complete next and to provide hints during the exercise [35, 40].

As mentioned before, an RL agent typically needs numerous episodes in order to learn a policy, as it is a data-hungry approach. This implies that usually online learning with students is not a viable option, which leads to a need for realistic simulators or models of students to train the system or the availability of the adequate historical data to be used with offline RL methods [32].

Additional challenges for RL in education would include the limited observability of the environment (the students' knowledge) and the significantly delayed and noisy outcome measures [32] usually found in this setting. This limited observability is illustrated in [41], in which the authors aim to be able to determine the sequence of content that will both keep the student engaged and maximize learning gains, discussing how to find the best representations for the state. Given that the domain is high stakes, it is not possible for them to evaluate the representations online, so they take a data-driven approach, using historical data to evaluate representations offline.

The data challenge for reinforcement learning can also be seen in the development of [33], where the authors build a system to determine a personalized sequence of questions for each student that best predicts their knowledge state, with the intention of improving their learning outcomes. Their study focuses on how to accurately infer each student's knowledge state, as their level of understanding of some concepts, through questions. The response

history should allow choosing as the next question the one revealing the most about their knowledge state. In order to train the system, and alleviate the problem of the need for large amounts of data, they develop a synthetic dataset of simulated knowledge states, based on graph representations, allowing to demonstrate the effectiveness of their system in front of a simple set of heuristics before using it on an online educational platform. We can see how these examples illustrate the need for alternative ways to train our systems, either using historical data or a simulator. In our case, the lack of historical data motivated us to build a simulated user to test our agent, which will be detailed in Section 3.4.

These advances in several directions converge for the development of pedagogical agents, i.e., virtual characters that interact with the users to facilitate learning opportunities [42]. They provide help and guidance in online education systems, deciding what to advise, when to do it, how to express the advice and how to communicate the advice effectively; accompanying the student interacting through natural language, either voice or text messages.

Literature on the topic is grounded in the Computers as Social Actors paradigm, which states that we interact with media in inherently social and human ways [43]. Pedagogical agents draw from the previous work on educational learning to scaffold the students by providing questions, exercises or hints, but can also provide cognitive support and enrich the experience by exhibiting empathy. Users may perceive the experience in a more positive way as a result of interpreting the agents as social actors.

There is evidence supporting that the agent influences the attitude, perception and behaviour of the user to improve the learning process [42]. They can serve as a model for the student, promoting task engagement, since their presence increases the students' interest and attention, but can also do so by adapting to the learners' background and behaviour, having great versatility.

We find two categories of pedagogical agents [43]:

- *Conversational agents*, which are able to hold conversations with learners.
- *Teachable agents*, characters that the students teach to complete exercises.

We are interested in the first group, on which researchers from the UB

already have experience developing virtual tutors for educational applications, able to engage users in complex conversations and incorporate sentient qualities [11].

2.2 Conversational agents for Information Elicitation

The development of conversational agents started in 1966 with ELIZA [44], considered the first chatbot, as it aimed to simulate human conversation. These agents are based on the idea that the social conventions that guide interpersonal behaviour can also be applied to human-machine interaction, with research highlighting how interaction with systems that present a more social and affective behaviour is more natural and effective than with those that do not [45].

Chatbots can be classified in two classes [46]:

- *Open domain dialogue systems*, which do not limit the conversation topic to a specific domain, and are able to generate new dialogues but usually without a goal.
- *Task oriented dialogue systems*, aimed to solve a specific task through dialogue.

Open domain systems can benefit from learning-based approaches, as a system can learn states and actions from the data provided without a need for designing explicit rules. Again, the main restriction would be how data hungry these approaches are, with some studies performing experiments on simulated data generated by building different models on the users' profiles and preferences [46].

In task oriented chatbots, a dialogue manager is to help complete the task the agent is designed to support [47]. An example of task oriented chatbot would be those based on Artificial Intelligence Markup Language, based on *topics* and *categories*, consisting the second of patterns indicating user input and the template to generate an answer [48]. The use of predefined templates allows to build chatbots as reactive agents, which need to be extended with a different mechanism to address the queries not covered, for example,

literature shows how to compensate for this cases applying rules and logic to a knowledge base to find a suitable answer [14]. Reinforcement learning can be applied to this second category of chatbots to choose between dialogues, for example, by associating a utility with each dialogue strategy, being these strategies the aforementioned templates [49].

Traditionally, chatbots have worked based on simple pattern recognition, usually in a task oriented manner, but recent advancements in the fields of machine learning and natural language processing have allowed to expand their capabilities [45]. The increase in the complexity and capabilities of task oriented chatbots has lead to an increase in the variety of their applications. An example would be how chatbots offer a new way to collect information through conversational surveys [50], a particularly interesting case for our work, given that we aim to build an agent that performs this same task in the context of a video game. This allows to substitute a traditional online survey with an agent asking open-ended questions and interpreting free-text responses from the user.

The effectiveness of interviewing chatbots for information elicitation has led to the research on how this kind of one-on-one interaction impacts the user engagement and the quality of the information obtained [51], compared to the aforementioned online surveys. These two dimensions, response quality and level of engagement, often determine the success of an interview. Moreover, the use of these agents is aimed also at reducing survey-taking fatigue, where the longer a survey is, the less time the participant spends in each question and the more probable it is that the survey is dropped [50], which is only exacerbated with open-ended questions on traditional surveys, as more effort is needed to reply.

Conversational agents allow for users to communicate in natural language, providing more flexibility to express themselves freely. However, if the questions are not communicated clearly, or the user input cannot be correctly recognized, there may be a negative impact on the experience which may result in the abandonment of the interview [51]. Still, studies on the topic find that chatbots both increase the user engagement and the quality of the information elicited in terms of informativeness and relevance [50, 51], illustrating the effectiveness of conversational agents for this task.

For our work, an interesting example of such tool is Perla [16], a conversational agent designed to perform interviews based on the Patient Health Questionnaire (PHQ-9), where responses are to be provided on a Likert scale.

The chatbot allows the user to express naturally, assuming responses to be referring to how frequently a depression symptom is experienced. The response is then mapped to the corresponding Likert item, given predefined synonym phrases and fuzzy matching. The authors found that transforming the Likert scale of the traditional questionnaire into a structured interview did not imply a significant loss of reliability, while it also increased engagement, being much preferred in their experiments.

Therefore, using chatbots to carry out questionnaires is an approach that has had success in prior experiments, as an alternative to traditional surveys. For our purposes, when considering UX questionnaires for games and VR experiences, we aim to take an step further by integrating this questionnaires in-world, i.e. as part of the game, evaluating the experience without interrupting it. This is specially interesting in VR, as transitioning from virtuality to reality to perform a survey can lead to systematic bias [52]. Embedding this process as part of the experience may ease participation and avoid bias, allowing to stay closer to the context of an ongoing exposure and elude a negative impact on the user's immersion.

Given the effectiveness and popularization of these tools, as well the importance conversational agents give to replicating social interactions, it is only natural for recent research to try to make them as human-like as possible, in order to make them more relatable for the user, with interest in endowing chatbots with the capability to perceive and display emotions [53] to emphasize with the user. Some concern has also arisen on the ethical perspective of these agents, as they are not to engage in private or sensitive topics [51]. These chatbots are to interact with users, and, therefore, should be designed with some moral considerations in mind, ideally, they should be capable of identifying situations or actions that can inconvenience or even harm the interlocutor in the context of a social interaction.

In our case, in-world questionnaires have the advantage of not interrupting the immersive experience [7], but has also the risk of disturbing the game flow [54] if the chatbot does not identify correctly when to prompt the user. We are to tackle this challenge by embedding in the chatbot a notion of the moral value of *respect*, which should guide the agent not to disturb the user experience while performing the questionnaire.

2.3 Machine ethics

As we previously introduced, one of the objectives of artificial intelligence is to allow agents to coexist in harmony with people and other systems, which has lead to a recent focus on ensuring these agents behave ethically in the context of social interactions. Machine ethics aims at developing agents that can engage in moral behaviour, that can handle in some way our values and motivations. Several approaches have been developed in the literature to do so, this section will provide an overview on the theory in which these methods are based and a taxonomy proposed to classify them. We will use this taxonomy as a way to situate in the field the approach we select to use in our work.

These approaches provide agents with ethical mechanisms, which allow them to consider moral issues that may arise from the interaction with humans, usually focusing on moral dilemmas, which has been defined as situations where the action of the agent can lead to the detriment of the human [3]. This encapsulates scenarios in which the agent has to choose between two actions and also those in which it has to choose between performing or not a particular action, having these moral implications such as harm, security, privacy or discrimination, to name a few.

When looking at introducing ethical values in conversational agents, we can find some attempts in the literature on enhancing the capabilities of such agents by introducing an empathy module [11] that monitors some indicators to approximate the user's state, with predefined rules based on dynamic thresholds to trigger a specific behaviour.

Very little literature is found on alignment of ethical principles in conversational agents, even less implementations. Some discussions set the pavement by suggesting the necessary steps to take in the future, highlighting the need to furnish conversational agents with ethical awareness, and debate the nature of ethical dilemmas in the context of online learning environments [15]. These deliberations highlight the need for morality systems that encode these complex notions, pointing to machine ethics and moral alignment advances for this task. One future application suggested is to introduce into pedagogical conversational agents a notion of the development of course skills, the students' marks, their dedication and motivational state, which would allow the agent to understand the impact its actions have on these dimensions, avoiding decisions with discriminatory or unfair collateral results, preventing

students to feel overwhelmed and disengage from the content.

Alignment of these values is a particularly difficult task, as there is no consensus in the ethical principles to be embedded, these greatly depend on the society in which the agent is designed to act. But, setting aside philosophical questions on morality, multiple models have been recently introduced in order to align an agent to some moral beliefs.

In order to express or represent these beliefs for the agents to learn them, different goals of alignment have been proposed in the literature [55]:

- *Instructions*, the agent does what we instructed it to do, with the disadvantage of excessive literalism, where the agent may follow instructions too strictly even in situations where it is not desired.
- *Expressed intentions*, the agent does what we intended it to do, the challenge being that the subtleties of expression of these intentions may cause misunderstanding and side effects.
- *Revealed preferences*, the agent does what our behaviour reveals we prefer, learning from observation. However, in reinforcement learning, it is difficult to infer a reliable reward function, in part due to the lack of information on rarely observed situations.
- *Informed preferences or desires*, the agent does what we would want it to do if we were rational and informed. Here the challenge lies in how to limit the capabilities of the agent in order not to act in the detriment of others for the sake of the provided preferences.
- *Interest or well-being*, the agent does what is in our interest, or what is best for us, defining well-being by means of attributes such as physical health, security and nutrition to name a few. But difficulty arises when the agent has to choose whose needs to align to, as the best interest of someone might not be an action we are morally entitled to do.
- *Values*, the agent does what it morally ought to do, as defined by the individual or society, with facts about what is right or wrong to punish or promote actions. In this approach, the agent would align to a set of beliefs, being the challenge deciding which principles to align to.

We can see how different goals for alignment come with their own challenges, with the last approach, *value-based alignment*, being the most popu-

lar. Even when expressing preferences, some approaches opt to define these as a set of values indicating preferred states [56].

Given the diversity in the lines of work explored for value-based ethical agent, and in order to locate the approach we base our work in the literature, we first need to introduce the reader to the different approaches that have been proposed in the last decades. For this, we can look at a popular taxonomy to classify Artificial Moral Agents according to the strategies and criteria used to tackle ethical problems [3].

Authors of this taxonomy understand ethics as the philosophical discipline that studies the moral dimension of human beings, being morality a system of values (hence, value-based) and behaviours to live in harmony with others, describing what is good, right and proper, associated with virtue, righteousness and justice.

To develop agents, the design is usually based on *normative* ethics, concerned primarily with the articulation and justification of the fundamental principles or values that govern how people should live and what they morally ought to do [3]. Two types of normative theories would be the *utilitarian*, which focuses on maximizing utility functions, and the *deontological*, which defines duties and limitations as norms describing what to do.

Then, for the taxonomy, a first classification, proposed by Moor [57], would be based on how these ethic values are provided to the agent:

- *Implicit ethical agents*, internally built to show an ethical behaviour, but unable to distinguish good from bad.
- *Explicit ethical agents*, can process ethical rules expressed through formalisms.
- *Full ethical agents*, which would have human-like ethical capabilities.

From the design perspective, they can be classified in:

- *Top-down*, design is based on some ethical theories.
- *Bottom-up*, design is not based on an imposed ethical theory, but on using learning mechanisms and values to guide the agent.

- *Hybrid*, combining both the previous approaches, able to present a flexible moral judgement.

As mentioned before, advancements in machine ethics have focused on addressing ethical problems based on *moral dilemmas* [3], situations in which an agent is morally ought to do two different actions but cannot do both, either because one action is simply not doing the other or because there is a restriction in the world preventing doing both.

An example would be the *commons dilemma*, where and agent is located in an ecosystem with limited resources that slowly regenerate, if it follows its individual interest, the resources will be depleted to the detriment of all participants.

Depending on the type of dilemma and its ethical rules, a final classification would be:

- Obligation dilemma, where all feasible actions are mandatory, but the agent cannot carry out more than one, having to choose which.
- Prohibition dilemma, all feasible actions are forbidden, but the agent must choose one to carry out.

Category	Strategy	Criterion	Description
Implicit ethical	Implicit	Non-malicious codes	Agents avoid unethical behaviours, but they are not aware of it
Explicit ethical	Top-down	Normative ethics	Agents use a specific normative ethical theory to make decisions
		Situationism	Agents use more than one normative ethical theory to make decisions
	Bottom-up	Empirical	Agents derive ethical behaviour by themselves based on learning mechanisms through trial and error
	Hybrid	Situationism	Based on both top-down and bottom-up ethical criteria, their decisions are situation-specific
Full ethical	Top-down	Normative ethics	Agents use a specific normative ethical theory to make decisions
		Situationism	Agents use more than one normative ethical theory to make decisions
	Bottom-up	Empirical	Agents derive ethical behaviour by themselves based on learning mechanisms through trial and error
	Hybrid	Situationism	Based on both top-down and bottom-up ethical criteria, their decisions are situation-specific

Table 2.1: Taxonomy of ethical agents [3].

The resulting taxonomy for the classification of artificial moral agents can be seen in Table 2.1. Notice that full ethical agents are similar to explicit

ethical agents [57], as highlighted by the table given that their entries have the same descriptions. The difference resides in that a full ethical agent is attributed human features such as consciousness, intentionality and free will, being still in debate if a machine could be a full ethical agent in the future [3].

In this work, we will focus on an approach based on designing an environment that incentivizes agents to behave ethically [58] by adding a *normative dimension* and an *evaluative dimension*, that dictate actions to be punished and actions to be praised. With this in mind, the approach would fit in the taxonomy as one that focuses on explicit ethical agents and is based on a bottom-up strategy, which means, as we can see in Table 2.1, that elements are provided for the agents to learn an appropriate behaviour without imposing an ethical theory.

The approach proposes an algorithm to follow when designing ethical environments, formally guaranteeing that the agent learns to behave ethically while pursuing its individual objective [59]. We will describe the specifics of this approach when detailing the design.

Chapter 3

Problem formulation

Having introduced the reader to the state-of-the-art of the different lines of research that interact in our work, we can now continue by defining the domain in which our agent is to learn to behave ethically. This is, to specify the environment that defines our problem, and the necessary simplifications to then integrate our agent. Notice that these simplifications are due to the novelty of our work, given that the ethical embedding approach that will be used in Chapter 4 has only been used on toy problems and we aim at integrating it into a conversational agent that interacts with the player in a video game. Simplifying the environment allows us to bridge this gap.

This chapter first introduces the reader to the notion of *engagement*, in Section 3.1, as our moral value of *respect* is expected to take it into consideration, in order to avoid disturbing the user experience. We select a definition of engagement from the literature and present the different phases which compose it. Then, in Section 3.2, we present a simple game that is to be used in our experiments and identify the different phases of engagement in the experience. By selecting a game with such simple mechanics, where engagement phases are clear, we will be able to determine if our agent correctly learns to *respect* the user's engagement.

Section 3.3 presents experience questionnaires and their application in games, indicating the measuring tool chosen and why we consider it adequate. These questions will build the pool used by our chatbot to query the user. This section also displays the way the agent interacts with the user: the interface used by the chatbot, and the possible responses to the questions.

Finally, Section 3.4 will tackle the additional problem of the need for data in order to train our RL agent. We introduce the reader to the notion of *simulated users*, the motivation behind their use in the literature, their advantages and disadvantages, as well as different types. Then, we define a simple yet representative simulated user that substitutes human users in order to train our agent in the subsequent chapters.

3.1 Engagement

As mentioned in the introduction, our objective is to embed in our agent the moral value of *respect*. In our context, when asking questions of a survey while the user is playing a game, we consider *respectful* to do so when the *engagement* is low. Therefore, to continue our work, we first need to define what is engagement.

In the literature, engagement is a major theme of research within Human-Computer Interaction (HCI) [60], a concept that is considered important not only for the design and implementation of interfaces and applications but also to allow for these to adapt to the users [61]. By taking into account engagement, applications should adapt to the user and act appropriately according to the situation [62].

However, it is unclear exactly what and how to measure in terms of engagement, what the important measures are for specific scenarios, as well as how different aspects of engagement are related [63]. Therefore, engagement is a topic that continues to pose challenges for researchers and designers. This is partly because of the numerous interpretations of engagement found in the literature, which conflict in their definition and make hidden assumptions or imprecise generalizations [60]. For example, engagement has been viewed both as a reflection of user involvement and interaction [64], and as a process by which some participants establish, maintain and end their perceived connection [65], being the later a popular definition that positions engagement as a component of social connection [60].

In games, engagement has been regarded as the first in three levels of immersion, in which the user is to invest time, effort and attention in learning how to play the game and familiarize with the controls [66]. The multiple approaches to interpretations of engagement are usually based on either theories that focus on the actor's processes or theories that focus on the structure

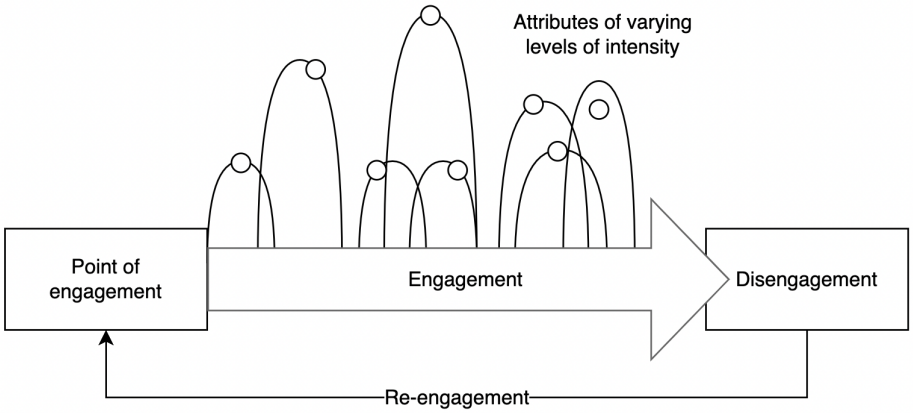


Figure 3.1: Model of engagement [1].

processes. The first group would include flow theory, social presence theory and cognitive load theory, while the second would include theories of motivation, self-determination theory and narrative theories [60].

In our case, we are to focus on the actor processes, as we want to avoid disturbing the engagement from the player's point of view. Particularly, we consider the framework proposed by O'Brien and Toms [1]. This approach is based on flow theory, which assumes the existence of a state of optimal and enjoyable experience [67], authors suggest that engagement shares some attributes with flow, such as focused attention, feedback, and intrinsic motivation. But, in contrast to the original definition of flow, authors theorize that engagement may not be dependent on the user forming specific goals for the interaction. An engaging experience is encouraged by the sensory appeal of the system and the level of feedback and challenge the user receives from the system, but the impression made by the experience may not have more meaning than that it was enjoyable or challenging [1].

They propose a definition of engagement based on its attributes, i.e., characteristics of the interaction that influence the engagement:

> Engagement is a quality of user experiences with technology that is characterized by challenge, aesthetic and sensory appeal, feedback, novelty, interactivity, perceived control and time, awareness, motivation, interest, and affect [1].

Therefore, using this definition, engagement would be viewed as a process,

in which different stages are distinguishable given their inherent attributes [1]. The authors define the following phases, illustrated in Figure 3.1:

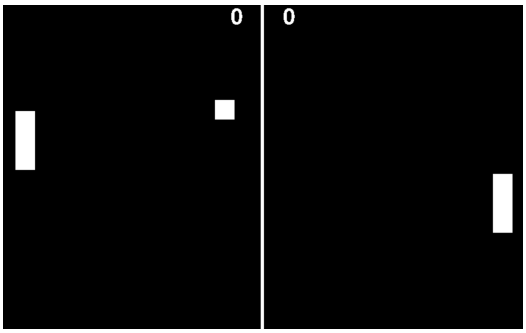
- Point of engagement, where elements such as aesthetic, informational composition or novelty capture the users' attention introducing them into engagement.
- Period of engagement, in which the users' attention and interest are maintained during the interaction, through feedback, novelty or challenge, for example.
- Disengagement, which can be caused by an internal factor, the user's decision to stop, or an external factor in the environment, such as being interrupted or distracted.
- Re-engagement, which happens when the users are disengaged before being ready to do so, returning to see what is next in a game or after having attended the task that disengaged them.
- Nonengagement, if the user is not able to re-engage, due to multitasking or interruptions preventing to engage with the primary task.

Notice that a single session of interaction with the application is composed of multiple engaging experiences which vary in intensity. Our objective is then that our conversational agent is able to recognize periods of high engagement, in order to avoid interrupting and disengaging the user by asking questions.

3.2 Pong

To develop our agent, we need to choose a simple game in which the different phases are easily recognized, allowing us to confirm that the agent actually recognizes periods of engagement and disengagement. We chose this game to be Pong, one of the first games ever created, released by Atari in the 1970s, it is a simple game based on table tennis [68]. The game is composed by 2 paddles and a ball, it can be played by 2 human players or one player against a paddle controlled by the computer. The paddles are located at the sides of the screen (see Figure 3.2d) and can move vertically in order to hit the ball that bounces back and forth. It is a very simple game, being the

(a) Starting menu. (b) Menu between levels. (c) End menu.



(d) In-game image.

Figure 3.2: Screenshots of the implemented Pong game.

most complex mechanic the fact that the ball bounces in a different angle depending on where it hits. The further the ball hits the paddle's centre, the more pronounced is the angle when bouncing. If a player misses the ball, a point is given to the opponent. The game ends when one of the players reaches an arbitrary limit of points.

In our implementation of the game, we set the user to play against a computer controlled paddle with a very simple behaviour. The computer tries to align the centre of the paddle with the centre of the ball. In order to allow the computer to miss, it has only vision on its half side of the screen (see Figure 3.3), staying put when the ball is on the opposite half. Our game is composed of levels, of increasing difficulty. The higher the level, the faster the components move and the smaller the paddle is. Between levels, a simple menu indicates who wins and allows the user to proceed to the next level when ready (see Figure 3.2 for images of the game and menus).

Given the previous definition of engagement, we can safely assume that the periods of higher engagement will be those of higher enjoyment, challenge, attention, feedback and interactivity, which are the levels themselves. On the other hand, we can argue that screens between levels (which we will name *menus*) can be, therefore, either periods of lower engagement or points of disengagement, being the player re-engaged at the start of the following

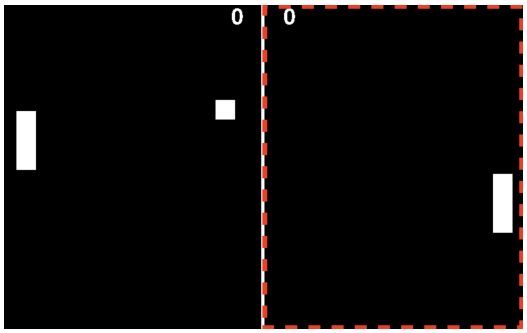


Figure 3.3: Area perceived by the opponent, the computer controlled paddle, highlighted in red.

level. Therefore, we can either interpret menus as periods where the engagement is sustained but lower or if it is stopped, in both interpretations we can assume that menus would be the best moment to ask survey questions without interrupting periods of high engagement. This implies that, if our agent correctly learns the moral value of *respect*, it will tend to avoid asking questions during a level. Notice that it is the simplicity of the game chosen what allows us to make these assumptions about engagement. Knowing already which are the engaging periods in the game will allow us to corroborate that the value is learned correctly. We should now address next which questions should our chatbot make.

3.3 Survey

With the increasing amount of video games being developed every year, games research has become prominent in Human Computer Interaction (HCI). To assess the quality of a video game and improve their design, developers resort to playtesting, with users playing the game in order to then provide feedback about the experience after the play session concludes [4]. In order to evaluate this experience, various self-report measuring instruments have been developed.

The Game Experience Questionnaire (GEQ) [69] has become one of the most prevalent instruments to measure key dimensions of the player experience, being a widely applied option, used for multiple genres, user groups, gaming environments and purposes [70]. Despite its popularity and appearing to be a versatile tool, the psychometric properties of GEQ have been

questioned due to a lack of validation [71, 70].

When trying to verify the 7-factor structure of GEQ (consisting of 33 items), researchers reported inconsistent results, with the original factor structure not being replicable [72, 73]. When developing scales, Confirmatory Factor Analysis (CFA) is used in order to verify the structure, which is then expanded by applying Exploratory Factor Analysis (EFA) to pinpoint potential problems [70].

CFA results in the literature suggest that the proposed model of GEQ does not acceptably fit the data. These conclusions, combined with EFA results, indicate several problems in regard to items sharing substantial variance with factors other than their proposed factor [74]. The results lead to the conclusion that the original factor structure of the GEQ is not adequate, and, as a consequence, alternative proposals of new structures have emerged, obtained by collapsing some items and building new ones [70, 74]. For this reason, some researchers avoid using GEQ [75].

Authors of the Game User Experience Satisfaction Scale (GUESS) highlighted the lack of psychometric validation testing for the GEQ and offered GUESS as an alternative [71]. GUESS is an assessment tool with a 9-factor model and 55 items, a model for which the authors provided psychometric validation. Since its release, the GUESS has been applied in domains such as healthcare simulation, mixed reality, social interaction, and virtual reality gaming; but with such number of items, responding to all of them can be cumbersome when repeated assessments are necessary [4].

A shorter version, GUESS-18, was developed in order to be used in iterative game design, testing, and research [4]. The authors conducted a CFA on GUESS to reaffirm that the model has a good fit, and then truncated the scale to reduce the number of items, performing a CFA again on the result to assess that the model fit was maintained. Two items were selected per dimension, with results concluding that an 18-item survey was a well-fitting model of game user satisfaction when assessing the nine original constructs. The authors still recommend the use of the original GUESS given the detail provided by the extra items, but offer GUESS-18 as an alternative for iterative design, as it takes around 3 to 5 minutes to complete, compared to the 10 to 15 of the original GUESS. They recommend its use in situations where other measures are used or survey fatigue is a concern, as well as in environments as serious games, where long surveys may not be feasible due to constraints, such as the availability of participants, time limitations or

Constructs	Statements	Used
Usability/Playability	I find the controls of the game to be straightforward.	Yes
	I find the game's interface to be easy to navigate.	Yes
Narratives	I am captivated by the game's story from the beginning.	No
	I enjoy the fantasy or story provided by the game.	No
Play Engrossment	I feel detached from the outside world while playing the game.	Yes
	I do not care to check events that are happening in the real world during the game.	Yes
Enjoyment	I think the game is fun.	Yes
	I feel bored while playing the game.	Yes
Creative Freedom	I feel the game allows me to be imaginative.	Yes
	I feel creative while playing the game.	Yes
Audio Aesthetics	I enjoy the sound effects in the game.	No
	I feel the game's audio (e.g., sound effects, music) enhances my gaming experience.	No
Personal Gratification	I am very focused on my own performance while playing the game.	Yes
	I want to do as well as possible during the game.	Yes
Social Connectivity	I find the game supports social interaction (e.g., chat) between players.	No
	I like to play this game with other players.	No
Visual Aesthetics	I enjoy the game's graphics.	Yes
	I think the game is visually appealing.	Yes

Table 3.1: Items of the GUESS-18 [4], indicating those we use.

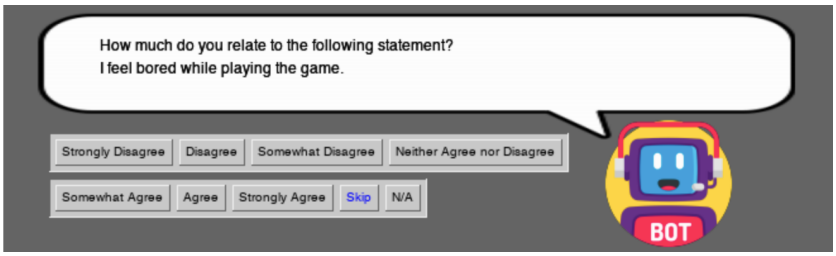
using an iterative design, which would need repeated testing.

Table 3.1 illustrates the constructs and items in GUESS-18, as well as an indication of which of them we consider should be applied to in our case, for the game we have chosen to use. In our system, items indicated will serve as a pool of questions for our conversational agent, who will take from it to ask the player. The user may respond using a 7-item Likert scale, depending on how much they relate to each statement (see responses in Figure 3.4b).

Given that the survey will take place ingame, i.e. during the experience, we add as possible responses two options, "Skip" and "N/A" (as in *No Applicable*, used when the player has no answer). The first option can be used to avoid responding to a question, which will make the chatbot remove it from the pool not to ask it again; while the later option indicates that the user does not know an answer yet, which makes the chatbot return the question to the pool in order to ask it again afterwards. With the chatbot learning from experience, we expect users to ignore or *skip* questions during the game, and to indicate that they have no answer if they are in the first stages of the game. These options should allow the chatbot to learn when questions are not desired and when it is considered too soon to ask about the experience.

The chatbot interface can be seen in Figure 3.4. Notice that we are using a Windows, Icons, Menus, and Pointers (WIMP) interface, instead of using natural language. As highlighted in previous chapters, our objective is to

(a) Chatbot not asking.



(b) Chatbot asking.

(c) Chatbot finished.

Figure 3.4: Screenshots of the possible states of the chatbot interface, using resources from Flaticon, made by Freepik.

introduce a notion of a moral value, not to build a fully fledged conversational agent capable of interaction through natural language. Once we can ensure that the agent learns our notion of *respect*, future work could build upon it to obtain a more complex conversational agent that allows the user to interact in such a way. To do so, we could follow the steps of the aforementioned Perla [16], based on the Patient Health Questionnaire (PHQ-9), which can interact through natural language and map responses to the corresponding Likert item. Giving the agent anthropomorphic features, such as voice, a face or a name, should also make the experience more appealing and facilitate engagement through personification [76]. But we consider these additional capabilities to be out of the scope of this master thesis.

3.4 Simulated user

This project faces an important challenge that is the amount of data necessary to train RL agents. Being a data-hungry approach, online learning is usually not a viable option. We have already presented some approaches taken in the literature to alleviate the problem of the need for data, with authors using simulators, i.e., models of the users that consider profiles and preferences, or synthetic datasets (historical data), to train the agents [32, 33, 46]. Here, we focus on the first option, as no historical data is available for our problem.

The impact of the need for data is even greater in interactive RL (IntRL), where a human is involved in the learning process [77]. The agent requires the interaction of a human, normally the user, using the information provided by this user through the interaction to learn to improve its behaviour. But despite the fact that having a human in the process is considered a strength for the interactive agent, it is also an obstacle for the learning process [78].

Some characteristics of human interaction can impact their applicability to train RL agents [79]:

- Information source may not be available all the time or respond in time.
- Concept drift may impact the process as the intentions of the user shift over time.
- The preconceived thoughts about the agent may introduce a bias.

When training RL agents, numerous episodes are usually needed in order to learn a policy, as a vast space of states and strategies are to be explored [80], and it is common to repeat the process in order to experiment with different combinations of parameters. The time required for the interaction may reduce the possibility of performing a large number of experiments. To require a human every time for the agent to learn is undesirable, as human trials are expensive and time-consuming, with repeatability and the acquisition of participants also being a challenge [79].

In some scenarios, building these interactive systems that learn through trial-and-error is not feasible without the use of automatic user simulation tools [80]. The purpose of these tools is not to model an individual user with

detail, but to provide an alternative to be used during the development of IntRL agents [81].

A simulated user is not as creative or wise as a human, but it readily allows for controlled experiments, being their obvious advantage the possibility to repeat experiments indefinitely while having complete control over the variables [82]. Simulated users have been used as a method to evaluate RL agents that require human input, allowing for repeatable testing with controlled parameters [79], studying the impact these variables have on the learning process and resulting behaviour, as well as comparing different learning algorithms [79].

To recapitulate, some authors hypothesize that, during the first stages of the development of interactive RL agents, it is much more convenient to use simulated users [79], as they allow for rapid development while providing the following advantages [80]:

1. Allow for longer and repeated training, in order to better explore the space of policies.
2. Allow to explore scenarios that may not be contemplated in historical data.
3. Allow not to fix the internal representation of the agent, state space and action set, as the system may be modified, meaning it would need to be retrained.

In this project, we can therefore, define a *simulated user* as an automated user model that reproduces human characteristics, capable of replicating human interaction in some degree, with the purpose of rapid and controlled training and testing [79]. These agents are designed to act as a human would, reflecting the properties of a human in a given situation, which, for example, implies that a simulated user should be noisy, to emulate human-sourced information. Some principles are suggested in the literature when designing simulated users [79]:

- Consistency, indicating that simulated users should not take actions that human users would not.
- Completeness, indicating that simulated users should produce every action that the human user may take.

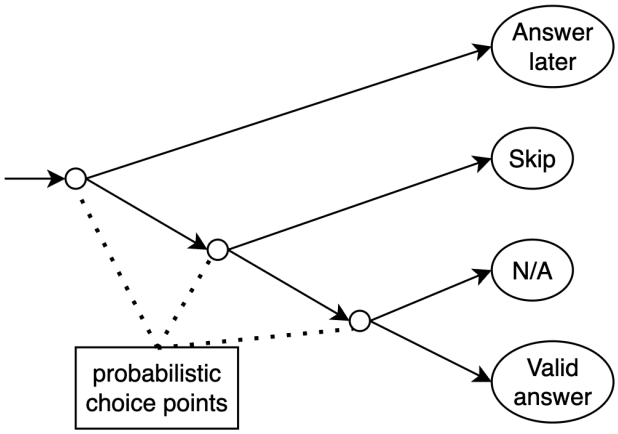


Figure 3.5: Model of our simulated user.

- Variation, indicating that simulated users should not replicate average behaviour, performing outlier or unintended actions that human users may take despite unlikely.

The simulated user may be designed based on three types of models or a combination of them [79]:

- Probabilistic model, a data-driven approach where the behaviour is defined by probable choices based on the observed behaviour of real users [83].
- Heuristic model, a deterministic approach, usually using hierarchical patterns [84] and rule sets [85].
- Stochastic model, an approach to simulate processes that fluctuate over time, simulating indeterminacy.

These models can be built based on data collected in human trials, available datasets or reverse engineered from the environment dynamics [79]. We focus on the second model, which is useful when there is little information on the user, but that information is sufficiently accurate and reliable. In our case, we have no historical data to be used in our setting, so we are to use a predefined set of heuristics, combined with the probabilistic approach in order to allow a degree of "lifelike" randomness in the behaviour [81].

Popular techniques to implement simulated users include rule trees and ripple-down rules [86], in particular, for dialogue management, designers of-

ten opt for rules or attribute-value pairs [87]. In our approach, we decide to follow the literature and use a rule tree (see Figure 3.5), which can be seen as a graph-based model where arcs represent actions and nodes represent "choice points" [88]. Non leafs would be probabilistic choice points, that represent a random decision to be taken by the simulated user, with suitable probability values being set considering our knowledge and experience. These values can then be modified once experimental data is available, or changed in order to represent new profiles of users, as well as to see the impact on the learning procedure.

Notice in Figure 3.5 that the option "answer later" implies continuing playing the game, encapsulating the necessary behaviour to do so. When deciding to answer later, the user simply ignores the chat and continues playing the game. The user continues playing the game normally, without clicking any answer button, in which case the question remains waiting for an answer. The performance of the user playing the game does not impact the behaviour of the chatbot, but it is still necessary for the execution, as the interaction of the user with the game provides information on the different engagement periods. No matter if a user wins or loses, the agent is to *respect* the engagement equally, therefore, the simulated user does not need to have a particular performance playing Pong.

In order to simulate the user playing Pong, a simple rule set is used, similar to the one defined for the opponent in the previous sections. In this case, the user can see the whole screen and tries not to hit the ball with the centre of the paddle, as hitting with the centre makes the ball go straight while hitting with the sides results in the ball bouncing with a larger angle. It is a simple system that uses a common strategy to try to win the game.

In order to model the behaviour of the user when questioned by the agent, the structure of the tree is based on the following assumptions:

- User is playing the game, but show herself/himself collaborative on answering the survey.
- User can decide to "answer later" if asked during a game, reconsidering answering the question when the level is finished (the engagement is either lower or it ends).
- User is more likely to "skip" an answer or provide "N/A" while playing the game than in a menu, as the attention is focused on the challenge of the level.

	Answer later	Skip	N/A
Starting menu	0%	60%	100%
In-game	10%	30%	$40\% * 0.5^{level}$
Menus between levels	0%	5%	$30\% * 0.5^{level}$

Table 3.2: Probabilities for the choice points at different stages of the experience.

- User gains knowledge throughout the experience necessary to answer the questions, therefore, being more likely to answer "N/A" at the beginning of the experiment, simulating a form of concept drift.

It is important to highlight that, for the simulated user, any answer in the Likert scale is equal, as it is not providing an actual opinion on the game, therefore, we can abstract all these options as "valid answers".

If we take a look at Table 3.2, we can see the probabilities chosen for the choice points of Figure 3.5, taking into account the prior assumptions and based on our expertise. Different values for these probabilities would showcase different profiles of users, and can be changed to study the impact on the resulting behaviour of the agent. Notice that the "Start menu" needs a separate set of probabilities, since the user has not even played the game yet. We should also highlight that the option "N/A" varies over time (as indicated in the assumptions), by taking into account the number of the level to decrease the probability of not knowing the answer the further the user is immersed in the experience. Notice that these values could be improved if based on historical data, but this is not possible at the moment. Nonetheless, we consider this behaviour to be representative enough for the initial development of our agent. Further iterations on this work could introduce these changes.

When combining Figure 3.5 with the values in 3.2, we obtain the behaviour of our simulated user. Having this model of the user, we can start training our system to learn the moral value of *respect*, with the simulated user representing different stages of engagement while playing Pong.

Chapter 4

Introducing an ethical value

This chapter will provide an overview on the approach used for the alignment of our agent behaviour with our selected moral value. We will follow the work of Rodriguez-Soto et al. [2], which has been applied on moral dilemmas with success, to see how it can help in our scenario.

Their approach provides algorithmic tools that build upon Multi-Objective Reinforcement Learning, helping design an ethical environment in which the agent fulfils its ethical objective while pursuing its individual objective. In our case, we would want the agent to pursue its individual objective of asking survey questions while fulfilling the ethical objective of not disturbing engagement, which we call the value of *respect*.

We start with Section 4.1, which introduces the background necessary to apply the aforementioned tools, including the use of Markov decision process (MDP) to characterize the agent's environment, how an agent can learn a behaviour based on this MDP, how a multi-objective problem can be converted into a single-objective one if preferences on the objectives are defined, and, finally, how we can study these preferences in a multi-objective problem. In our case, giving preference to the ethical value. This would allows us to determine how to ensure the ethical objective is prioritized while pursuing the individual objective.

The ethical embedding problem then consists on designing an environment in which the agent is guaranteed to learn to behave ethically. Section 4.2 details how this ethical embedding should be defined, how to introduce a moral value into a multi-objective MDP to obtain an ethical multi-objective

MDP, and the definition of an ethical policy and an optimal-ethical policy.

Section 4.3 uses the background and definitions of the previous section to present the ethical embedding algorithm [2]. It details the steps necessary in order to build an ethical environment in which an agent learns to behave ethically. We showcase how to transform the original multi-objective MDP into a single-objective MDP, by means of a scalarization function that ensures the agent fulfils the ethical objectives while pursuing its individual objective.

Finally, Section 4.4 displays how we use these tools in the specific problem we aim to tackle. This includes the definition of our MDP, the individual objectives of our conversational agent, the ethical objectives that represent our moral value of *respect*, and the application of the previous algorithm in order to ensure our agent learns to survey the user respectfully, to avoid disturbing the user while she/he is engaged in the game experience.

4.1 Background

In order to understand how the ethical embedding problem is approached, we introduce first the needed background: the definition of an MDP, how value iteration and Q-learning can be used to learn a behaviour, how to convert a multi-objective problem into a single-objective one, and how to find policies in a multi-objective problem depending on the preferences over these objectives.

Multi-Objective Reinforcement Learning makes use of a Markov decision process (MDP) to characterize the environment in which the agent acts. An MDP is defined as a tuple of five elements, $\langle S, A, R, T \rangle$ [89]:

- S is a finite set of N states.
- $A = \{a_1, ..., a_k\}$ is a set of k actions.
- $R(s, a, s')$ is a reward function that provides the expected reward for each tuple of state s, action a and future state s'.
- $T(s, a, s') = Pr(s'|s, a)$ is a probability distribution, specifying the probability that executing a in state s leads to s'.

Some authors include $\gamma \in [0, 1)$ in the tuple, a discount factor which indicates how much the agent prefers short term reward over long term reward. It is not always included in the tuple that defines a MDP, as its nature is not as related to the environment as it is to the learning algorithm used.

The usual MDP definition needs to be extended to fit our needs in this case, since we are considering an agent with multiple objectives. In an n-objective Markov Decision Process (MOMDP), the reward is defined as $\overrightarrow{R} = (R_1, ..., R_n)$, a vector where each R_i is a reward function associated to an objective, obtaining then the tuple $\langle S, A, \overrightarrow{R}, T \rangle$.

The agent's behaviour is dictated by a *policy*, π, which indicates for each state-action pair the probability of performing the given action in that state. Policies can be evaluated by computing the expected discounted sum of rewards obtained when following them. Consider the following functions:

- $V^\pi(s)$, the value function, the expected return when following π starting from state s. It provides the value of an state.

$$V^\pi(s) \doteq \mathbb{E}[\sum_{t=0}^{\infty} \gamma^k \overrightarrow{r}_{t+k+1} | S_t = s, \pi] \text{ for every state } s \in S \tag{4.1}$$

  An *optimal policy* is, then, one that maximises the expected long term discounted reward for every state, using an *optimal value function* V^*.

- $Q^\pi(s, a)$, the expected return when following π starting from state s and taking action a. Providing the value of an action in an state. With the relationship being:

$$V^*(s) = \max_{a'} Q(s, a') \tag{4.2}$$

Notice that learning V^*, or *value iteration*, is useful only when we have knowledge on the reward and transition functions [90], being a model-based approach. On the other hand, *Q-learning* can be used to train the agent online without this knowledge, discovering the information through exploration.

When considering a single objective, an optimal policy can exist given that there is a total order between policies, we can always determine whether $V^\pi < V^{\pi'}$ or $V^\pi > V^{\pi'}$ or $V^\pi = V^{\pi'}$ for any pair π, π' [2]. Despite the fact that multiple optimal policies can be found in an MDP, all of them share

the same optimal value function [91]. But, when looking at an MOMDP, a policy π can be better than another π' for a given objective ($V_i^{\pi} > V_i^{\pi'}$) while being worse for another objective ($V_j^{\pi} < V_j^{\pi'}$), providing only partial order between policies [2].

For this reason, it is not possible to determine the optimal policy of an MOMDP without some additional knowledge, some preferences over the objectives. We can only determine a subset of uncomparable policies that are better than the rest. This leads to the possibility of defining two solutions for an MOMDP [2]:

- Single-policy MORL, where a *scalarization function* is used to reduce the dimensionality, becoming a single-objective problem.
- Multiple-policy MORL, where we compute a set of policies that *could* be optimal for some hypothetical scalarization function, referred as *undominated* policies.

In this second case, the one we are interested in, authors of the approach focus on linear scalarisation functions. Value functions are combined with a weight vector $\overrightarrow{w}$, allowing to compute the undominated set as a *convex hull* [92]. This convex hull would be the subset of policies that are maximal for some weight vector $\overrightarrow{w}$ [2]:

$$CH(M) \doteq \{\overrightarrow{V}^{\pi_*} | \pi_* \in \Pi^M \wedge \exists \overrightarrow{w} \in \mathbb{R}^n : \overrightarrow{w} \cdot \overrightarrow{V}^{\pi_*} = \max_{\pi \in \Pi^M} \overrightarrow{w} \cdot \overrightarrow{V}^{\pi_*}\} \quad (4.3)$$

Being Π^M the set of policies in the MOMDP, π_* an optimal policy and n the number of objectives. Obtaining the convex hull will allow us to study which linear preferences make different policies maximal.

The convex hull can be obtained applying Convex Hull Value Iteration, an algorithm which allows learning optimal policies for all linear preference assignments over the multiple objectives at the same time [93]. The procedure can be seen in Algorithm 1, a modification of the original value iteration. In the definition, $\mathring{Q}(s, a)$ represents the vertices of the convex hull of possible Q-value vectors for taking action a at state s. Notice that the authors define the necessary operations on convex hulls in order to replicate the original value iteration using them, namely: translation and scaling, summing, and extraction of the Q-value.

Algorithm 1: Convex Hull Value Iteration [93]

Initialize $\mathring{Q}(s,a)$ arbitrarily $\forall s, a$;
while *not converged* **do**
  forall $s \in S, a \in A$ **do**
    $\mathring{Q}(s,a) \leftarrow \mathbb{E}[\overrightarrow{r}(s,a) + \gamma \text{hull} \bigcup_{a'} \mathring{Q}(s',a') | s,a]$;
return X

By defining our MOMDP with the agent's individual objective as well as the ethical objective combined in a linear function, then applying Convex Hull Value Iteration, we can identify the set of undominated policies, and when each is maximal for the different objectives given $\overrightarrow{w}$. But we first need to illustrate how the ethical objective is formally introduced in the MOMDP.

4.2 Ethical embedding problem

The problem of value alignment would be divided in two steps [2]:

- *Reward specification*, transforming ethical knowledge into ethical rewards.
- *Ethical embedding*, ensuring the rewards incentivize the agent to be ethical.

The approach focuses on the ethical embedding, as a value alignment problem with an ethical objective already specified. An agent is to fulfil this ethical objective while pursuing an individual objective [2], being the goal for the agent to learn a behaviour in alignment with the moral value in consideration.

When considering a moral value, the authors call their model the *signature* of a moral value, in order to highlight how it does not capture all the complexity and richness of moral values, but a workable model [58]. This signature is based on normative ethics, on norms defining penalized actions as well as praiseworthy ones. Two dimensions compose this ethical objective:

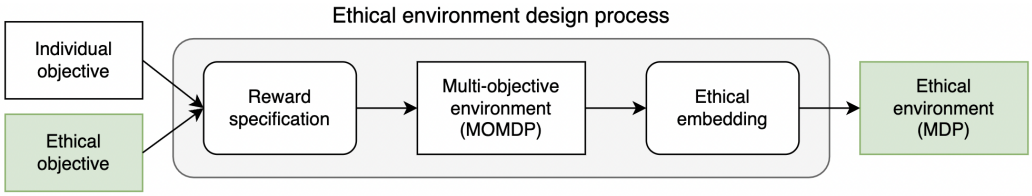


Figure 4.1: The process of designing an ethical environment is composed of two steps: reward specification and ethical embedding. The approach taken focuses on the later [2].

- *Normative dimension*, punishing non-compliance with norms, actions that go against the moral value.
- *Evaluative dimension*, rewarding morally praiseworthy actions, those that are not mandatory but positively viewed when considering the moral value.

An *ethical MOMDP* is then defined as an MOMDP where the agent must consider an individual objective and an ethical objective, specified with a tuple $\langle S, A, (R_0, R_N + R_E), T \rangle$ [2], where:

- R_0, the reward function provided by the individual objective.
- R_N, the normative reward function punishing violations of normative requirements with negative reward.
- R_E, the evaluative reward function that provides positive reward to praiseworthy actions.

In this definition of ethical MOMDP, we consider an *ethical policy* one that abides to all the norms while behaving as praiseworthy as possible [2]. The value vector for such policy, $\overrightarrow{V}^{\pi_*} = (V_0^{\pi_*}, V_N^{\pi_*}, V_E^{\pi_*})$, is optimal for both the normative and evaluative components. Moreover, if the policy is ethical and maximizes the individual objective, we consider it an *ethical-optimal policy*. Just as before with the non-ethical MOMDP, in an ethical MOMDP, all ethical-optimal policies share the same value vector.

We should remind from the previous section that we can use a scalarization function to reduce the dimensionality of our problem and obtain a single-objective MDP. The authors approach is, therefore, to use this encoding of an ethical MOMDP and define an embedding function in order

to obtain a single-objective MDP. This function should guarantee that the agent can only learn ethical optimal policies [2]. As highlighted before, in order to be able to find the undominated set of policies as a convex hull, they use a linear combination of the objectives:

$$f(\overrightarrow{V}^{\pi}) = \overrightarrow{w} \cdot \overrightarrow{V}^{\pi} = w_0 \overrightarrow{V}_0^{\pi} + w_e(\overrightarrow{V}_N^{\pi} + \overrightarrow{V}_E^{\pi}) \tag{4.4}$$

Being w_0 the individual weight (fixed to $w_0 = 1$), and w_e the ethical weight, whose value we must find. The ethical embedding should then allow us to define an ethical MOMDP as a tuple $\langle S, A, w_0 \overrightarrow{V}_0^{\pi} + w_e(\overrightarrow{V}_N^{\pi} + \overrightarrow{V}_E^{\pi}), T \rangle$, where we are to find a value of $\overrightarrow{w}$ that guarantees all optimal policies to be also ethical-optimal. In other words, the process allow us to transform the original MOMDP into a single-objective MDP where any optimal policy is ethical-optimal (see Figure 4.1). Authors also highlight that this value should be as little intrusive as possible with the learning process, therefore, we want to find the minimal weight w_e that satisfies the condition. As a reminder, this weight is the one to be computed by means of using Convex Hull Value Iteration.

4.3 Ethical embedding algorithm

Having introduced the necessary background and the definition of an ethical MOMDP an specification of the reward and weights, we can now present the algorithm proposed by Rodríguez-Soto et al. [2] (see Algorithm 2) to combine this with the Convex Hull Value Iteration in order to find the desired value of w_e, finding a solution weight for the ethical embedding problem. This weight allows us to combine individual and ethical reward in a single reward, using the aforementioned linear function, creating an environment in which the agent will learn an ethical behaviour.

Having defined our ethical MOMDP, with the corresponding rewards, the first step would be to compute its convex hull. Notice that we do not need the whole convex hull, as the ethical-optimal value vector $\overrightarrow{V}^*$ is the same for all ethical-optimal policies [2]. Any subset with at least one ethical-optimal policy is enough, which means we could constrain the search space

Algorithm 2: Ethical Embedding [2]

function EMBEDDING(Ethical MOMDP $\langle S, A, (R_0, R_N + R_E), T \rangle$) ;
Compute the convex hull for weight vectors $\overrightarrow{w} = (1, w_e)$ with $w_e > 0$;
Find $\overrightarrow{V}^*$ the ethical-optimal value vector, and $\overrightarrow{V}'^*$ the second-best value vector in the convex hull;
Find the minimal value for w_e that satisfies Eq. 4.5;
Return $\langle S, A, R_0 + w_e(R_N + R_E), T \rangle$;
end function;

for the Convex Hull Value Iteration to consider only weight vectors of the form $\overrightarrow{w} = (1, w_e)$.

The idea is that, once computed the convex hull, we can extract from it the policy that maximizes the ethical reward function $(V_N + V_E)$, the ethical-optimal policy $\overrightarrow{V}^*$, and the policy with the second-best value $\overrightarrow{V'}^*$. Then, w_e can be obtained as the minimal value that satisfies:

$$V_0^*(s) + w_e[V_N^*(s) + V_E^*(s)] > V'^*_0(s) + w_e[V'^*_N(s) + V'^*_E(s)] \tag{4.5}$$

Having obtained the value of w_e, all optimal policies in the single objective MDP defined as $\langle S, A, R_0 + w_e(R_N + R_E), T \rangle$ will be ethical [2]. Therefore, the weight vector serves as solution to the ethical embedding problem. The agent can now learn in this ethical environment using an algorithm such as Q-learning, ensuring an ethical behaviour while pursuing its individual objective.

4.4 Embedding *respect*

We can now finally tackle our particular scenario, applying the ethical embedding approach described in order to introduce the moral value of *respect* in our conversational agent. The value alignment is divided in two steps: first we need to formally specify the environment and the ethical rewards, then we can apply Algorithm 4.3 to ensure that the agent is incentivized to behave ethically.

The environment described in Chapter 3 needs to be specified as a multi-objective Markov decision process, a model for the agent to learn in. We need to describe the tuple $\langle S, A, R, T \rangle$, starting with states, actions and transitions, explaining later the individual and ethical objectives and how these are specified as rewards. States in our MOMDP will be described by the following attributes, notice that we aim at designing a set of states that should not be difficult to apply to other games:

- *Last valid answer*, a binary attribute indicating if the last question was answered with a valid option. As stated in Section 3.3, valid answers are those in the Likert scale, all but "Skip" and "N/A". This provides a notion of when obtaining an answer should be rewarded, as invalid answers provide no useful information for the survey.

- *Response time*, a binary attribute indicating how long it took the user to answer the last question asked, converting time into a binary value by considering if it is under (indicated by a value of 0) or above a threshold (with value 1). We establish this arbitrary threshold of five seconds, in which we assume the user can answer the agent, being such short questions (see Table 3.1). A long response time indicates that the user is not available to answer at the moment, therefore, probably highlights that we asked in a period of high engagement.

  These first two attributes are not dependent on our particular game, Pong, they can be applied to any experience where the agent is applied. Even if the response format is modified, answers can be classified as those valid and those not. The response time threshold may need to be adjusted if a different set of questions is used.

- *Input*, a binary attribute indicating if the user has provided input in the game in the last second. Just like with the previous attribute, this value is based on the time since the last input, the higher the time the higher the value. Specifically, we set a value of 0 if the last user input was less than a second ago, and a value of 1 if the last user input was more than a second ago. Notice that *what* input is not important in this case, the user can only move up or down, we are interested in the fact that there is an input to the game.

  This attribute provides a notion of when the user is participating in an engaging activity. Notice that, in a more complex game, we would not include all input, it should only consider keys denoting high engagement. For example, in an adventure game, movement input does not

necessarily indicate high engagement, but combat input would do. In our case, there are only two keys to play Pong, and both indicate high engagement due to the nature of the game.

It is through this attribute that the game designer can provide some knowledge of the game to the agent, as we are indicating which input we know denotes engagement. In the case of Pong, both moving up or down would be denoting engagement.

- *In game*, a binary attribute indicating if the user is in game or in a menu, a boolean where 0 indicates not in-game and 1 in-game. We could argue that this attribute could have high correlation with the previous one, as there should be no input in the menu, but this assumption may not be true for all users (depending on their skill or how collaborative they are), and would change for different games.

- *Current level*, an attribute indicating which level the user is playing. In order not to have a large state space, we limit this attribute to 3 values, indicating if the user is in the first level (with a value of 0), the second level (with a value of 1) or a higher level (with a value of 2). Notice that the starting menu (remember Figure 3.2a) is considered part of the first level.

  In other games, where levels are not differentiated, an alternative way should be offered to provide a notion of how long the user is into the experience. This can be easily changed in most cases, taking into account other attributes that increase with how far the user is in the experience, such as time or progression milestones.

Being the state defined by 5 attributes, all of them binary but one (which has 3 different values), we have a total of 48 states (encoding of the states can be seen in Table A.1). A relatively low number, which should help the learning process. We avoid a too complex state representation and a large state space in order to better generalize new games in the future.

Actions of our agent have already been suggested in Section 3.3, having only two possibilities, either *ask* or to *wait* for a better moment. We will allow the agent to take an action every second during execution. Notice that, if a question is asked, the agent will not be able to execute another action until the user answers, as we cannot ask two questions at the same time.

Finally, the transition function specifying the probability of moving from a state to another given the action performed is provided by the execution of

the game and simulated user combined. Notice that it is not a deterministic function, as there are probabilities embedded in the users' behaviour. Given that value iteration requires a model of the environment, we require specifying the table $T(s, a, s')$ in order to perform Convex Hull Value Iteration. This table can be approximated by simply executing the environment with a random behaviour and storing the probabilities observed for each transition (see Tables from A.2 to A.11).

Having defined the previous elements of our MDP, we can now introduce the reward specification. We aim to embed the moral value of *respect* in order for the agent to learn when to question the user without disturbing engagement, avoiding having a negative impact on the experience. The objectives would then be:

- The *individual objective* of the agent is to fill the survey, obtaining a reward when an answer is valid.

$$R_0(s, a, s') \doteq \begin{cases} 1, & \text{if } s'.last\_valid = 1 \text{ and } a = 0 \\ 0, & \text{otherwise} \end{cases} \tag{4.6}$$

  Where $s, s' \in S$ and $a \in A$, and $a = 0$ represents the action of asking (with $a = 1$ being the wait action). This objective should motivate the agent to ask questions to the user.

- A normative reward, punishing questions that have long response times or invalid answers, as well as asking when the user input is high.

$$R_N(s, a, s') \doteq \begin{cases} -2, & \text{if } ((s'.response\_time = 1 \text{ or} \\ & \quad s'.last\_valid = 0 \text{ or } s.input = 0) \\ & \quad \text{and } a = 0) \text{ or (s.input} = 1 \text{ and a} = 1) \\ 0, & \text{otherwise} \end{cases} \tag{4.7}$$

  The agent should learn when it is not adequate to ask a question, either because no answer is provided or because it was prompted in a moment in which the user is not available, taking into account both the user response and the user input.

  Notice that the penalization can be divided in two parts, one for asking in the wrong moment, and one for not asking when there is no input from the user. The first part ($s'.response\_time = 1$ *or* $s'.last\_valid =$

0 *or* $s.input = 0$) *and* $a = 0$), penalises the agent for obtaining invalid answers, responses that take too long or interrupting input. The second, ($s.input = 1$ *and* $a = 1$) penalises waiting instead of asking when it seems to be possible due to low input.

- An evaluative reward, promoting questions that are answered with a short response time and with a valid response.

$$R_E(s, a, s') \doteq \begin{cases} 1, & \text{if } s'.response\_time = 0 \text{ and} \\ & s'.last\_valid = 1 \text{ and } s.input = 1 \\ & \text{and } a = 0 \\ 0, & \text{otherwise} \end{cases} \tag{4.8}$$

  This should incentivize the agent to learn when questions are correctly answered and in a quick manner, taking also into account that there should be low input, indicating that the user was available at that moment.

We are introducing two concepts, the first, by taking into account response time and valid answers, should make the agent remember situations when prompting the user was adequate. The second, taking into account the user input, uses the information provided by the game designer when indicating user input that we know is related to engagement, therefore, providing information to the agent on when not to disturb. Notice that, if we provided only the first concept, if the user is collaborative and responds in-game, the agent could then learn to ask in-game, even if disturbing engagement. If we considered only the second component, the agent could learn to ask any instant the user stops the input, even if only for a few seconds.

Notice that these rewards are only encouraging to learn when to *start* making questions. Once the engagement is low, the agent should start querying the user, but answering questions will keep the engagement low. Therefore, we can expect the agent to continue asking questions indefinitely. The agent has no notion on when to *stop* asking, in order to introduce this notion, we would need to incorporate an additional moral value that takes into consideration, for example, survey fatigue. This implies that we would also need for the simulated user to replicate this survey fatigue to train the agent (redefining the behaviour described in Section 3.4). However, considering multiple ethical objectives is outside the scope of our work.

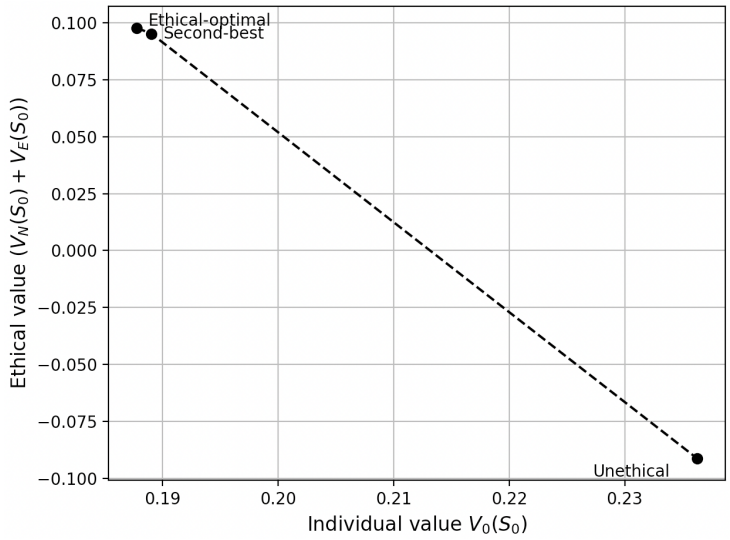


Figure 4.2: Visualization of the convex hull, providing the ethical-optimal value vector, the second-best value vector, and the policy that maximizes the individual objective.

However, although this is outside of the scope of our work, we still need to limit the interactions for the agent not to be able to ask the complete survey in a row. Since the agent should learn to initiate the survey, and we aim to integrate this in the experience, in the form of multiple short interactions instead of a longer one at the end, we limit the amount of consecutive questions the agent can make to five. If the agent reaches this limit, it will need to wait for some seconds before being able to question the user again, specifically, we set the agent to wait two to three seconds. Having selected 12 items from GUESS-18 (see Table 3.1), only 3 interactions asking the maximum amount of questions would be needed to fill the survey, which can be distributed throughout the levels in the game.

Having described the MDP and the different rewards, we can apply Algorithm 2 to obtain the ethical environment. The first step would be to compute the convex hull (with Algorithm 1) using the previously estimated $T(s, a, s')$ (again, values can be seen in Tables from A.2 to A.11). Figure 4.2 illustrates the result, providing only the ethical-optimal value vector, the second-best value vector, and the policy that maximizes the individual objective (added in order to ease understanding the distribution in the figure).

The two first value vectors are the ones with the greatest ethical value, which are now used to calculate the weight w_e. In order to do so, we need to

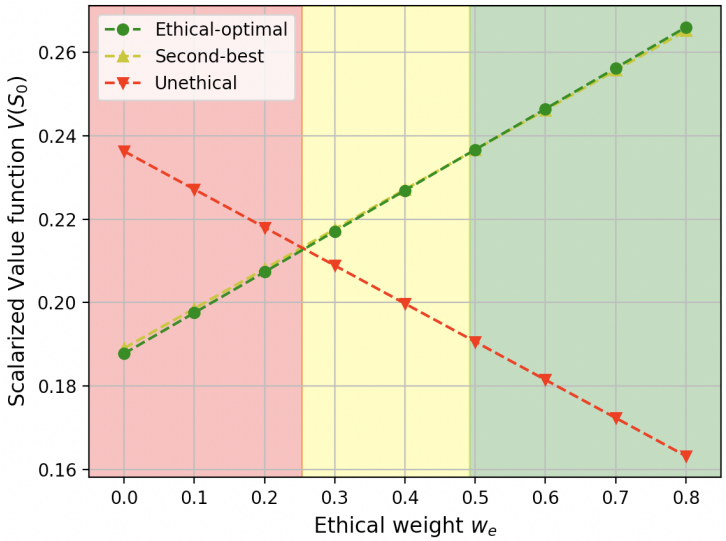


Figure 4.3: Representation of the weight space of the convex hull, highlighting where each policy becomes optimal.

solve Eq. 4.5:

$$0.18777 + w_e \cdot 0.09777 > 0.18906 + w_e \cdot 0.09515 \tag{4.9}$$

When solved, we obtain a value of $w_e > 0.49237$. This process is also represented in Figure 4.3, where we can see how the Scalarized Value function for each policy responds when changing the value of w_e, with the y axis providing the result of the Scalarized Value function and the x axis the evolution of w_e. This allows us to see when each policy is optimal depending on the value of w_e, represented by the areas matching the colour of the corresponding policy. We are interested in the point at which the policy from the ethical-optimal value vector becomes optimal.

This value of w_e can now be used in our MDP $\langle S, A, R_0 + w_e(R_N + R_E), T \rangle$, using, to set a value, $w_e = 0.55$ ensuring the agent learns to behave ethically. We can now study the impact of introducing the value of *respect*, by applying Q-learning to train our agent, comparing the performance to a baseline agent that does not have *respect* embedded. Q-learning allow us to obtain $Q^\pi(s, a)$ through:

$$Q'(s, a) = (1 - \alpha) \cdot Q(s, a) + \alpha \cdot (R(s, a) + \gamma \cdot \max_a(Q(s', a)) \tag{4.10}$$

Chapter 5

Results

Having created our ethical environment, we can now apply Q-learning to confirm the behaviour learnt by the agent is the one desired. To compare and understand the impact of the value of respect in the learning process, experiments in this chapter are carried out by both our ethical agent and an unethical agent, which takes only into account the individual objective.

We start with Section 5.1, which specifies the experiments performed. We also provide a first look at the learning process, by analysing the results of the accumulated reward.

Section 5.2 breaks down the results by providing more detail into how behaviour learned impacts engagement. We illustrate this by examining the distribution of questions between menus and in-game, the different type of answers obtained, and the arrangement of the questions asked throughout the game.

Section 5.3 follows an execution example of the behaviour of our ethical agent. A particular example which should aid the reader better contextualise the results. Moreover, this should help the reader situate the concepts seen throughout the document in an actual scenario, and illustrate the expected behaviour in a session playing the game.

Finally, Section 6.1 consolidates the observations made into conclusions on the behaviour learned and the influence of introducing our value of *respect*.

5.1 Experiments

As previously introduced, we are to use Q-learning to showcase the results of embedding the value of *respect* into our agent. This learning algorithm takes two parameters, the learning rate and the discount factor, which are set to $\alpha = 0.7$ and $\gamma = 0.7$, using a ϵ-greedy policy for exploration [91].

The experiment performs 1000 episodes, being each one a whole playtrough of our short game, with 5 repetitions of the experiment to average results. The game is composed of 3 levels, of increasing difficulty. Notice that this number of episodes could be considered low for an RL algorithm, but thanks to having a low amount of states and actions (only 48 states and 2 actions, as described in Section 4.4) this number of iterations is sufficient. If we consider an average game to last around 3 to 5 minutes, we would need from 50 to 80 hours of play only to carry out a single repetition of the experiment, which is, in most cases, not feasible with human testers, not to consider the 5 repetitions. This emphasizes the need for our simulated user for the first stages of development of such agents.

As a first approach to the global results, averaging repetitions and showcasing the learning process, Figure 5.1 illustrates the results for the rewards through the learning, with the x axis indicating the current episode and the y axis the accumulated reward obtained. The unethical agent (in red) does not change much through the learning process, managing growth but maintaining a value not far from the initial. On the other hand, the unethical agent

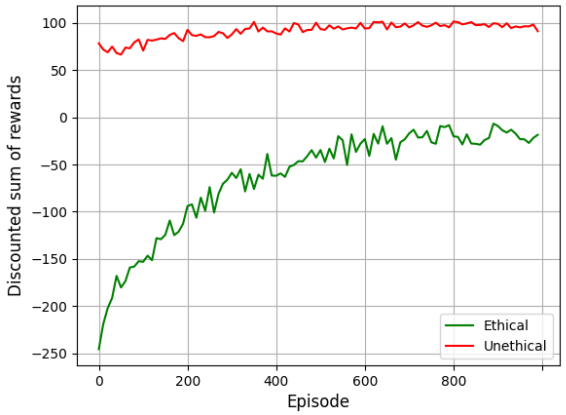


Figure 5.1: Accumulated discounted reward.

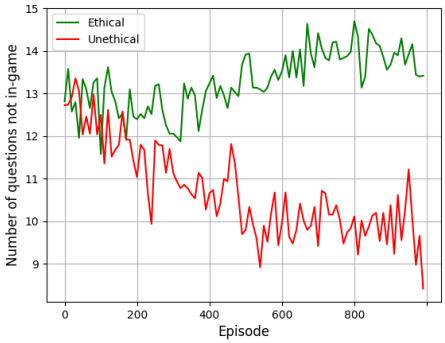
(in green) starts with a negative value due to penalizations of the normative component of the ethical objective, but shows a more noticeable growth through the episodes, converging around episode 600. Notice that the unethical agent obtains a higher reward, despite the ethical agent being able to gain a reward surplus from the ethical component over the individual objective. But this actually denotes how the individual reward comes into conflict with the ethical one, with the unethical agent being able to freely ask and interrupt during the experience. The ethical agent, on the other hand, would be limited by the normative component of the ethical objective, and focus on asking only in situations where the evaluative component promotes it as adequate. Notice the fact that the ethical agent converges around a negative value, this is due to the strong normative component defined in Section 4.4, as we are penalizing both asking and not asking in different situations.

These results show that the unethical agent, by following the individual objective, finds a more rewarding policy that is not allowed when using the ethical environment, one that is not considered *respectful*, therefore. The approach applied ensures that the individual objective is pursued while fulfilling the ethical objective, priorizing the ethical component, it is this second condition the one we see reflected in the difference between the rewards. But we should confirm this by looking at other metrics in subsequent sections.

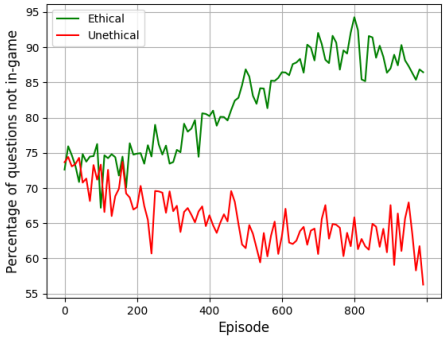
5.2 Impact on engagement

This section will provide insight on how the value of *respect* has impacted the behaviour when considering engagement. We should remind the reader that the agent can perform questions in-game, during the start menu and in menus between levels. We determined in Section 5.2 that the user will be more engaged while playing the game, therefore, in order not to disturb engagement, we expect the agent to avoid asking during a level and wait for a menu.

Let us start by looking at Figure 5.2, which illustrates how the number of questions made in menus evolved during the learning process. We provide both the total amount of questions in menus and the percentage when considering all the questions made during the game (both in-game and during menus). The unethical agent (again, in red) keeps a value of around 10 questions asked in menu, while the ethical agent (in green) performs around 13. We should remember that the ethical agent is penalised for not asking in



(a) Number of questions in menus.

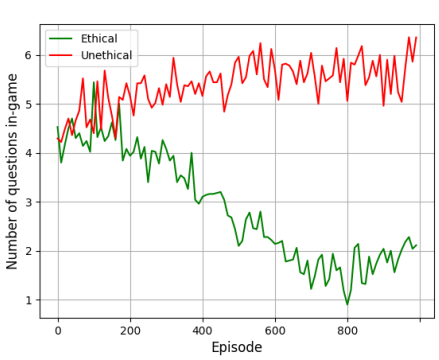


(b) Percentage of questions in menus (not in-game) over the total number.

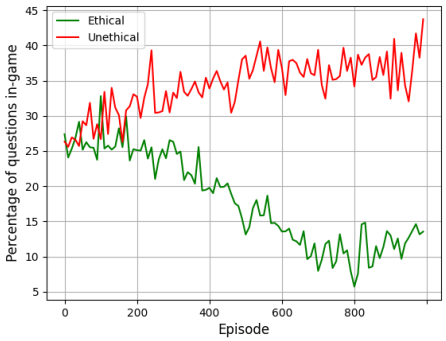
Figure 5.2: Questions prompted in menus, adding questions at the menu at the start, end or between levels.

situations where the user input is low, this naturally leads to the agent trying to maximise the number of questions in menu. If we take into consideration the percentage of questions made in menu, we can see in Figure 5.2b that the unethical agent is decreasing the proportion, meaning that the agent is learning to also ask in-game instead. The unethical agent converges around a value of 60%, meaning that Figure 5.2 is only illustrating about half of the total interactions of the unethical agent, being the other 40% in-game. On the other hand, the ethical agent converges in a value of 90% approximately, which is much higher, already denoting how the ethical agent, as expected, focuses on asking in menus between levels.

When looking at the questions made throughout the experience, we find that both agents have found moments in which they could ask the user in-game (see Figure 5.3), despite being a period of high engagement. This is to be expected of the non-ethical agent, but not from the counterpart. It would be highlighting the fact that there were moments in-game of low engagement. The ethical agent is finding these moments of lower engagement in-game, caused by the simulated user considering the possibility of the player deciding to answer during a level. This could be promoted by situations of low user input in-game, meaning, the user is playing the game but stops the input for some seconds, which activates the penalisation for not asking on low input. Notice that the agent is avoiding asking in these situations even if the user input is low. Asking would avoid this last penalisation, but the agent has learned it would not be adequate given past responses. On the



(a) Number of questions in-game.



(b) Percentage of questions in-game over the total number.

Figure 5.3: Questions prompted in-game, during actual gameplay, most probably disturbing the experience.

other hand, the unethical agent is managing to ask around 6 questions in-game, in a session of Pong of only 3 levels, which has a huge impact on the experience. While the ethical agent is able to reduce this value to around 1 or 2. Despite our specification of the ethical rewards improving the results, it is allowing these scenarios in which questions are asked in-game. This behaviour might be promoted by the simulated user, the lack of human data forced us to build the simulated user based on heuristics and experience, future work should test these results on human users, in order to corroborate the results and improve the simulated user. Still, the use of a simulated user is allowing us to introduce the value of *respect* and improve the behaviour of our agent, without the need to start directly testing on humans with an unethical behaviour.

Remember that, in Section 4.4, when defining the ethical objective, we had to consider that simply rewarding questions in the menus would not be a viable approach, as the desired behaviour would be easily achieved with a set of rules. We provided the agent with the information that questions answered with a valid answer and without a large response time are appropriate. Introducing also the information that some user input indicates engagement. This approach should allow our agent to learn in other games, as the information on valid answers and response time is always available and the game designer will always be capable of indicating a set of user input keys that denotes engagement.

In summary, the ethical agent reduces from 40% to 10% the amount of

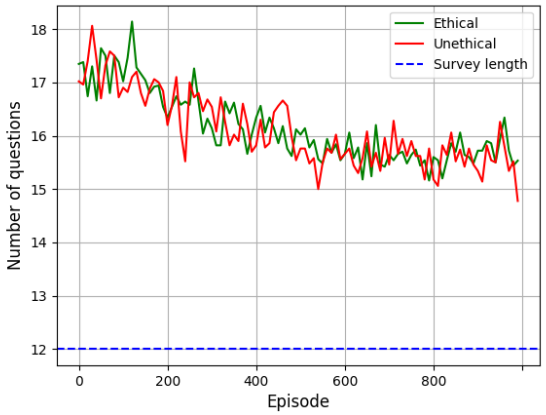
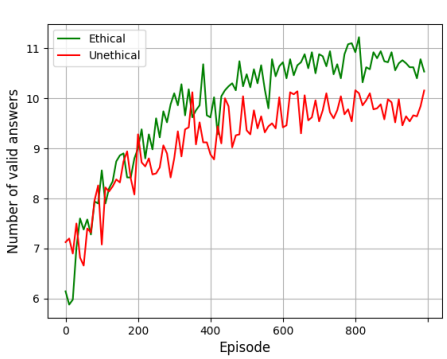


Figure 5.4: Number of questions along the learning episodes (x axis), with the amount of questions in the survey marked (on the y axis).

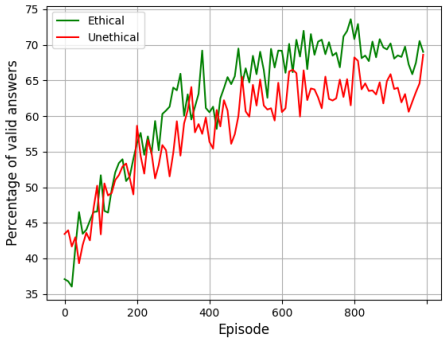
questions in-game. Moreover, the ethical agent maintains a high percentage of questions in menus, while the unethical actually learns to decrease this number. In proportion, the ethical agent performs better at avoiding interrupting the experience. The introduction of *respect* is improving the distribution of questions between menus and levels.

If we combine the previous results (see Figure 5.4), we can see that both agents are asking the same amount of questions. Both agent are also asking more questions than available in the survey pool, remember from Table 3.1 that we are using 12 of the 18 items in GUESS-18. This means that both are getting "N/A" answers and are forced to ask again later. Let us now study the distribution of valid and invalid answers. We should remind the reader that an answer is considered valid when it is inside the Likert scale, while invalid answers are either "Skip" or "N/A".

When looking at Figure 5.5, we can see that, despite asking the same amount of questions, agents are not obtaining the same amount of valid answers. Notice that the unethical agent obtains a reward only when the answer is valid, and, therefore, also aims at increasing the number of valid answers. When looking at Figure 5.5a, both agents manage to get about the same number of total valid answers, being the ethical agent slightly higher, which is confirmed in Figure 5.5b. If we consider the distribution of the questions between menus and levels, the ethical agent is achieving a high number of valid answers by prompting the user in adequate situations, while the unethical agent achieved this amount of valid answers by asking as much



(a) Total number of questions with valid answer.



(b) Percentage of questions with valid answer over the total number of questions.

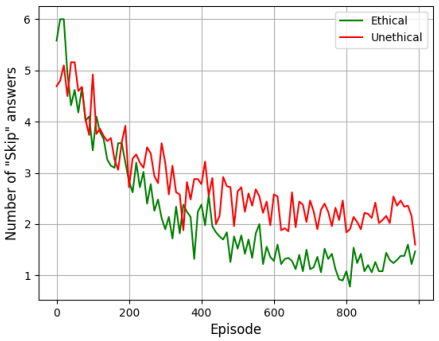
Figure 5.5: Total number of questions (in-game and menu) asked by the agent that received a valid answer from the simulated user.

as it can. We should remember how the simulated user allows to show a more collaborative conduct, meaning that the user can answer questions in-game instead of ignoring them, being her/his engagement disturbed.

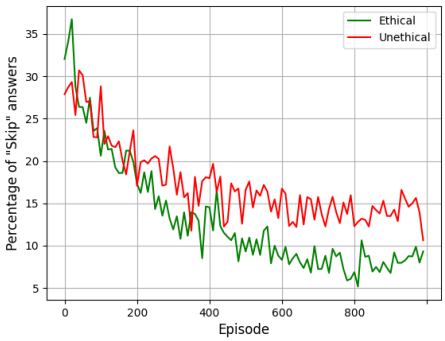
As a summary up to this point, results in these figures only reassure our previous conclusions, with the unethical agent focusing on asking more to achieve a higher reward, while the ethical agent is learning to ask in particular moments. This should improve the user's experience, as the agent would prompt questions in more adequate situations.

Focusing on the invalid answers, we can appreciate that both agents are trying to minimise the amount of "Skip" answers (see Figure 5.6). Only the ethical agent is penalized when receiving an invalid answer. But the unethical agent receives no reward for invalid answers, and remember that skipping a question removes it from the question pool, reducing the total possible reward. Therefore, both agents have a higher potential reward if they avoid situations in which they are answered with the "Skip" option. Notice that, as the ethical agent has learned to ask mostly in menus, the amount of "Skip" answers is lower compared to the unethical counterpart. This value is also approaching the value specified in the definition of the simulated user (see Table 3.2).

On the other hand, it is not the same case when considering the "N/A" option (see Figure 5.7), with the ethical agent being only slightly lower at



(a) Number of questions.



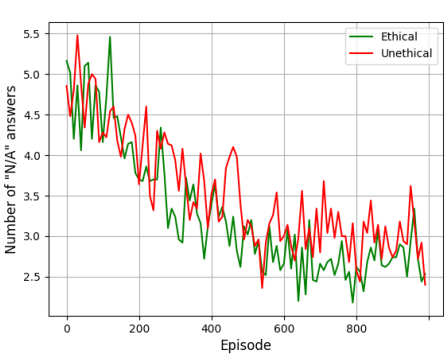
(b) Percentage over the total number of questions.

Figure 5.6: Questions that received "Skip" as answer.

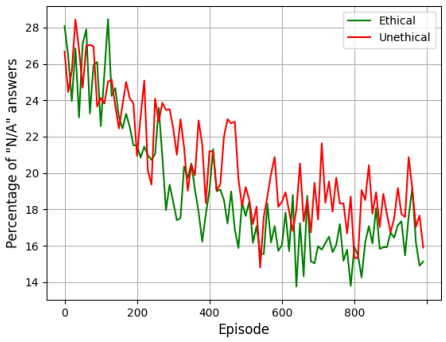
the graphs. Answering "N/A" has no disadvantage for the unethical agent, as it can ask later again with no penalization whatsoever. The ethical agent receives a penalization when this option is selected, as it is not considered valid. Still, both agents achieve about the same amount of "N/A" answers. Notice that this could be due to how rapidly the probability of the simulated user answering this option decreases. "N/A" answers from the simulated user depend on how far into the experience the user is. If we recall the behaviour for the user described in Section 3.4, the simulated user is more probable to respond "N/A" at the beginning of the experience, decreasing as the user familiarizes with the game.

Ideally, we want to avoid an "N/A" response, as it provides no information and returns the question to the pool. To do so, the agent should delay the questions, for the user to have time to familiarize with the game and have an impression before asking, reducing the probability of receiving this type of answer. But we specified the value knowing that our game is short and simple, so the user would be able to provide answers without the need for lots of playing experience with the game, meaning that the probability of answering "N/A" decreases rapidly. These behaviours, again, arise from the specification of the simulated user, which, as mentioned before, should be revisited in the future when human data is available, in order to better represent actual human behaviour.

In this regard, when looking at the distribution of questions throughout the experience (see Figure 5.8), both agents have learned to ask most questions between level 0 and level 1. This reflects how rapidly the probability



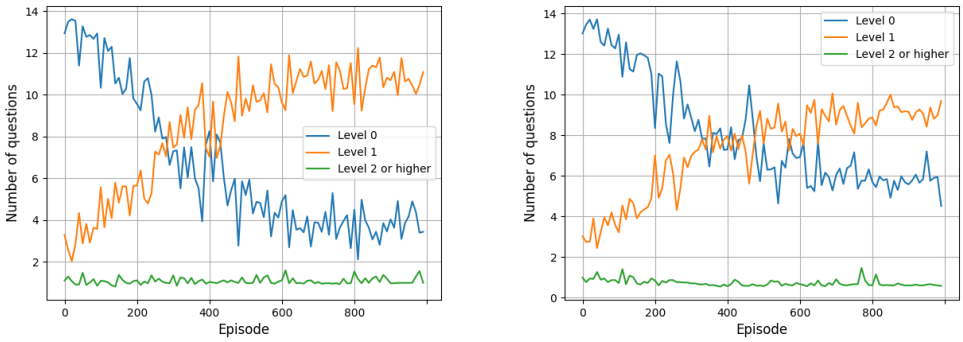
(a) Number of questions.



(b) Percentage over the total number of questions.

Figure 5.7: Questions that received "N/A" as answer.

of "N/A" answers decrease, being better to ask in the latter levels. The unethical agent is asking questions in-game (as we saw in Figure 5.3), it asks about the same amount of questions in level 0 and 1, being the second slightly higher, which is due to being a higher level and, therefore, having lower probability of "N/A" answers. The ethical agent, asks fewer questions at level 0, concentrating most of them at level 1. Notice that we set a time of around 2 to 3 seconds of waiting between groups of consecutive questions, but the ethical agent is still managing to ask more in the level 1 menu, due to the simulated user taking some time before moving to the next level. Being such a short game, we do not consider this to be inadequate, as there is not much space to distribute questions. In longer versions of the game, with more levels, we could increase the waiting time between sets of consecutive questions, forcing a distribution. Moreover, upon further experimentation on this distribution, we discover that, when performing longer games (with more levels), the ethical agent would change its behaviour only if the probability of "N/A" answers of the simulated user (from Table 3.2) is higher, meaning that most answers at the first levels are invalid. This highlights the impact of the simulated user and reiterates the need for human data to validate its specification. Still, let us remember that we focus on building an agent that avoids disturbing engagement, being the distribution of questions in the game outside the scope of our work.



(a) Ethical agent. (b) Unethical agent.

Figure 5.8: Distribution of the questions throughout the game.

5.3 Execution example

We illustrate now a particular example of the behaviour of our agent. To do so, we use the previously trained agent in a session of our game. Figure 5.9 serves as a representation of this play session using the ethical agent, while Figure 5.10 showcases the same scenario for the unethical agent. In these figures, the x axis represents the time playing the game in seconds, while the y axis the number of consecutive questions asked in a given timestamp. Regions are coloured depending on if the user is in-game or in a menu, a screen between levels. This means that green areas correspond to the menus in Figures 3.2c, 3.2b and 3.2a, while blue areas correspond to Figure 3.2d.

First thing we can notice in both figures is how menus are usually short if no question is prompted, with the duration increasing as the user takes some

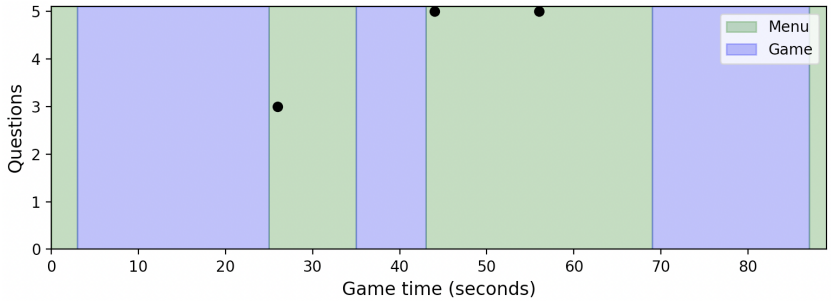


Figure 5.9: Execution example of the ethical agent in a 3 level game session, indicating periods in-game and between levels.

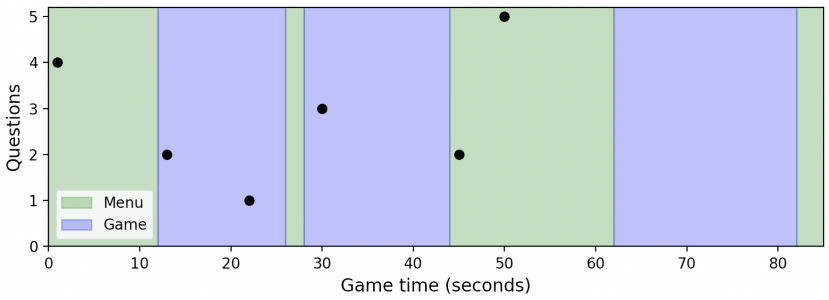


Figure 5.10: Execution example of the unethical agent in a 3 level game session, indicating periods in-game and between levels.

time to respond. Notice in Figure 5.10 that the unethical agent has started a total of 6 interactions, 3 of them in-game. The ethical agent (Figure 5.9), in comparison, has only started 3 interactions, all of them during menus.

Not only the unethical agent doubles the amount of interactions in games, it also asks 17 questions in total, while the ethical agent asks 13, only 1 more if we consider the 12 questions in the survey (remember Table 3.1). This is due to two causes: firstly, the unethical agent has prompted 4 questions in the first menu, before the player even started playing, where the probabilities of invalid answers are much higher (remember Table 3.2); secondly, the unethical agent has performed a total of 4 questions in-game, where the simulated user also has a higher probability of providing invalid answers.

In this example, the ethical agent has managed not only to avoid disturbing engagement, by asking during menus, but also reduced the amount of invalid answers by avoiding asking too early, prompting the user when the game has been played at least once.

5.4 Conclusions

In summary, results highlight how the ethical agent is improving greatly the results in most areas over the counterpart. The ethical agent is able to increase the amount of valid answers, asking mostly in menus. We consider the fact that the ethical agent tries to minimise questions in game a reflection on how it has learned to avoid disturbing the user engagement. These results are satisfactory, as our agent is asking questions in more appropriate situations, fulfilling the ethical objective while still pursuing the individual

one.

Results indicate that the behaviour of the simulated user allows the unethical agent to ask during the game, as well as to ask early in the experience, despite the probability of obtaining invalid answers. The model for the simulated user is allowing us to build this first iteration of our agent, corroborating the impact of introducing *respect*, but it should be revisited when data from human users is available.

We also notice how our codification of the moral value does not promote to completely avoid asking in-game, which could be caused by the behaviour of the simulated user (again, reinforcing the idea of the need for human data), but also due to the penalisation of not asking on low user input. We should remind the reader that the *ethical embedding* algorithm ensured that a moral value is fulfilled, it is the second step of the value alignment solution, being the first the reward specification. As in any Reinforcement Learning method, this specification relies in the adequate translation of our moral knowledge into a set of rewards, both positive and negative. With the *ethical embedding* algorithm being originally applied to toy problems, we can see how applying it into a more complex scenario introduces difficulties in this translation step. This difficulty is naturally inherited from the use of RL methods. The fact that the simulated user considers the probability of the user deciding to provide valid answers in-game, combined with our specification of the ethical value, leads to finding some situations in which the ethical agent deems adequate to ask in-game, although very few.

Chapter 6

Conclusions and future work

This final chapter is dedicated to recalling the concepts showcased through the document and highlighting the main conclusions extracted, as well as the direction in which our work could be carried on. Therefore, Section 6.1 summarizes the document and collects the conclusions extracted in each chapter, while Section 6.2 suggests the steps to take in the future in order to continue the advancements and contributions of our thesis.

6.1 Conclusions

We initiated this document providing a review on the state-of-the-art of several lines of research, where great advances are being made in different directions. This master thesis aimed at closing the gap between these advances, taking advantage and combining them into a novel application, to implement a conversational agent able to carry out a game user experience survey in-game.

This review highlighted how the use of Reinforcement Learning in games, serious games and Virtual Reality experiences has taken an interest in personalizing the content of a user's experience, as well as provide support, by means of introducing an agent that accompanies the user throughout the game. In our case, this agent is to ask questions of a survey, adapting to the users behaviour in order to avoid disturbing the engagement. Conversational agents in the literature have been applied before in order to perform inter-

views, but focusing mainly on the advantages of using natural language to interact, taking the survey at the end of an experience. Our approach aims at using the flexibility offered by conversational agents, but focuses on distributing the survey throughout the gameplay in a respectful manner. The survey should be integrated in the experience taking into consideration the different levels of engagement depending on the situation. This engagement is embedded in our agent by means of a moral value, with machine ethics studying how to do this. The field of machine ethics has gained attention recently, with multiple morality systems proposed in order to allow agents to gain a notion of our values. We aimed at introducing the concept of *respect* into our agent, guiding it to learn when to interrupt the user's experience. In summary, this review introduces the reader to the different lines of research that are combined in our thesis, showcasing the novelty of our approach. This novelty is twofold: firstly, the application of *ethical embedding* outside moral dilemmas in an application that interacts with a user; and secondly, the enrichment with this moral value of a conversational agent integrated in a game, an agent tasked with carrying out surveys during the experience.

In our approach, we first implemented the environment in which our agent was built. Due to the originality of the application, several simplifications are to be performed in the different lines of work, aiding at closing the gap between them. There was a need for, first, a game from which the agent needs to learn in which situations the user's engagement is lower. Pong was selected as, being as simple as it is, we can assume when the different periods of engagement are, and confirm that our agent behaves ethically. Then, we built an interface for the agent to communicate with the user, a WIMP interface, not based on natural language to avoid introducing new ethical concerns in the problem at hand. Finally, for the environment to be complete, we needed a user that our agent can observe in order to learn to behave. For this, a simulated user was built based on the literature and some simple heuristics, allowing to develop the base of our agent without the need for investing into human testing. The simulated user has also been useful in order not to start testing our agent directly with humans, allowing us to learn some initial behaviour before interaction with real players.

Having a game, a simulated user and an interface to communicate, we have selected a state-of-the-art approach to embed our moral value into the agent. Specifying *respect* as a set of rewards and punishments, the *ethical embedding* algorithm ensures that the agent learns to pursue its individual goal while fulfilling the ethical objective of behaving respectfully during the interaction with the user. This approach was originally tested in the lit-

erature on a toy problem, this master thesis serves as first steps towards the assimilation of such morality systems into actual computer applications. Moreover, our ethical conversational agent paves the way towards integrating user experience surveys into games and VR experiences.

Results obtained are promising, with the ethical agent improving the behaviour over an agent with no notion of *respect*. The agent learns to find appropriate situations to prompt the user with the survey. It manages to get a high proportion of valid answers while minimising interactions in-game. Overall, the introduction of the moral value improves the behaviour, having the agent a lower impact on the user's experience, by mostly avoiding the disturbance of the engagement.

6.2 Future work

In the future, we should study the generality of the approach followed in this thesis on new games, in order to confirm that the proposed MDP and interaction model can be translated into new problems without the need to completely redesign the environment. We should confirm how complex it is to apply our agent in new games, which will, at the same time, depend on the availability of data, in order to develop a simulated user or synthetic dataset, as information is in fact dependent on the game used.

As mentioned when introducing simulated users, and as reflected in the results, it would be advisable to perform testing of the resulting behaviour with actual human users. This would allow us to capture the data we are currently missing and corroborate the results. The simulated user has proven useful to build this first iteration of our ethical agent and confirm the advantages of introducing our moral value of *respect*, allowing us to, in the future, start testing on humans with an initial ethical behaviour, not needing to learn from scratch directly with human users.

We should also research different approaches for specifying rewards, as translating our ethical knowledge into actual rewards is a key point when considering morality systems. The approach taken to embed a moral value specifies an algorithm that allows to ensure an additional objective is fulfilled while pursuing the individual objective, but the proper codification of this objective is itself a challenge.

The difficulty in translating ethical knowledge is only exacerbated if we take into account the multiple perspectives or values held by different people or societies, highlighting the need to guarantee that our values are correctly translated into rewards. In would be interesting in the future to study how multiple ethical objectives, or perspectives, can be introduced in an agent.

Researchers at University of Barcelona are developing a Virtual Reality serious game that introduces programming concepts and computational thinking to tweens [7, 8, 9]. The advancements in this thesis are to be used to integrate a conversational agent as a virtual character in this game, being the agent capable of performing a user experience questionnaire in-world while avoiding disturbing the user's engagement and immersion. Our work serves as first steps for this goal, introducing such novel application and setting the pavement for the development of these agents.

In order to incorporate such agent into new games and Virtual Reality experiences, it would be interesting to follow the examples in the literature and use a conversational interface, which allows the user to express in natural language. To do this, we would need to map responses in natural language to the responses recognised by our agent, as either the valid and invalid answers defined in Section 3.3. This would, at the same time, open the door at using language cues to identify survey fatigue, which could be used, combined with the obtained results, to know not only when to start an interaction with the user but also when to end it appropriately, introducing additional moral values.

Appendices

Appendix A

States and transitions

Last valid	Res. time	Input	In-game	Level	State
0	0	0	0	0	0
0	0	0	0	1	1
0	0	0	0	2	2
0	0	0	1	0	3
0	0	0	1	1	4
0	0	0	1	2	5
0	0	1	0	0	6
0	0	1	0	1	7
0	0	1	0	2	8
0	0	1	1	0	9
0	0	1	1	1	10
0	0	1	1	2	11
0	1	0	0	0	12
0	1	0	0	1	13
0	1	0	0	2	14
0	1	0	1	0	15
0	1	0	1	1	16
0	1	0	1	2	17
0	1	1	0	0	18
0	1	1	0	1	19
0	1	1	0	2	20
0	1	1	1	0	21
0	1	1	1	1	22
0	1	1	1	2	23

Last valid	Res. time	Input	In-game	Level	State
1	0	0	0	0	24
1	0	0	0	1	25
1	0	0	0	2	26
1	0	0	1	0	27
1	0	0	1	1	28
1	0	0	1	2	29
1	0	1	0	0	30
1	0	1	0	1	31
1	0	1	0	2	32
1	0	1	1	0	33
1	0	1	1	1	34
1	0	1	1	2	35
1	1	0	0	0	36
1	1	0	0	1	37
1	1	0	0	2	38
1	1	0	1	0	39
1	1	0	1	1	40
1	1	0	1	2	41
1	1	1	0	0	42
1	1	1	0	1	43
1	1	1	0	2	44
1	1	1	1	0	45
1	1	1	1	1	46
1	1	1	1	2	47

Table A.1: Encoding of the states.

	0	1	2	3	4	5	6	7	8	9
0	0.000	0.000	0.000	0.000	0.000	0.000	0.994	0.000	0.000	0.000
1	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.814	0.000	0.000
2	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.880	0.000
3	0.015	0.000	0.000	0.755	0.000	0.000	0.008	0.000	0.000	0.128
4	0.000	0.031	0.000	0.000	0.784	0.000	0.000	0.008	0.000	0.000
5	0.000	0.000	0.032	0.000	0.000	0.795	0.000	0.000	0.006	0.000
6	0.000	0.000	0.000	0.098	0.048	0.000	0.783	0.005	0.000	0.003
7	0.000	0.000	0.000	0.000	0.035	0.255	0.000	0.506	0.008	0.000
8	0.000	0.000	0.000	0.000	0.000	0.025	0.000	0.000	0.817	0.000
9	0.020	0.000	0.000	0.683	0.000	0.000	0.003	0.000	0.000	0.119
10	0.000	0.056	0.000	0.000	0.608	0.000	0.000	0.008	0.000	0.000
11	0.000	0.000	0.023	0.000	0.000	0.628	0.000	0.000	0.000	0.000
12	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
13	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
14	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
15	0.000	0.000	0.000	0.500	0.000	0.000	0.000	0.000	0.000	0.000
16	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
17	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
18	0.000	0.000	0.000	0.000	0.000	0.000	0.821	0.000	0.000	0.000
19	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.645	0.000	0.000
20	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.706	0.000
21	0.053	0.000	0.000	0.489	0.000	0.000	0.030	0.000	0.000	0.060
22	0.000	0.073	0.000	0.000	0.561	0.000	0.000	0.000	0.000	0.000
23	0.000	0.000	0.115	0.000	0.000	0.577	0.000	0.000	0.038	0.000
24	0.000	0.000	0.000	0.000	0.000	0.000	0.111	0.000	0.000	0.000
25	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
26	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
27	0.000	0.000	0.000	0.000	0.000	0.000	0.008	0.000	0.000	0.027
28	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.003	0.000	0.000
29	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.008	0.000
30	0.000	0.000	0.000	0.000	0.000	0.000	0.160	0.000	0.000	0.000
31	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.089	0.000	0.000
32	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.038	0.000
33	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
34	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
35	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
36	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
37	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
38	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
39	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
40	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
41	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
42	0.000	0.000	0.000	0.000	0.000	0.000	0.136	0.000	0.000	0.000
43	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.029	0.000	0.000
44	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
45	0.000	0.000	0.000	0.000	0.000	0.000	0.038	0.000	0.000	0.000
46	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
47	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000

Table A.2: Transition table $T(s, a, s')$ for $a = 0$ (part 1).

	10	11	12	13	14	15	16	17	18	19
0	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
1	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
2	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
3	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.010	0.000
4	0.068	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.008
5	0.000	0.058	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
6	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.008	0.000
7	0.000	0.004	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.004
8	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
9	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.025	0.000
10	0.072	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.040
11	0.000	0.140	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
12	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
13	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
14	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
15	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
16	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
17	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
18	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.011	0.000
19	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
20	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
21	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.105	0.000
22	0.024	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.098
23	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
24	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
25	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
26	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
27	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.032	0.000
28	0.053	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.013
29	0.000	0.029	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
30	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.006	0.000
31	0.003	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.002
32	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
33	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.045	0.000
34	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.018
35	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
36	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
37	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
38	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
39	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
40	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
41	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
42	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
43	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
44	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
45	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.151	0.000
46	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.022
47	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000

Table A.3: Transition table $T(s, a, s')$ for $a = 0$ (part 2).

	20	21	22	23	24	25	26	27	28	29
0	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
1	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
2	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
3	0.000	0.022	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
4	0.001	0.000	0.009	0.000	0.000	0.000	0.000	0.000	0.000	0.000
5	0.014	0.000	0.000	0.012	0.000	0.000	0.000	0.000	0.000	0.000
6	0.002	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
7	0.004	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
8	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
9	0.000	0.082	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
10	0.000	0.000	0.056	0.000	0.000	0.000	0.000	0.000	0.000	0.000
11	0.058	0.000	0.000	0.023	0.000	0.000	0.000	0.000	0.000	0.000
12	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
13	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
14	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
15	0.000	0.500	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
16	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
17	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
18	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
19	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
20	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
21	0.000	0.120	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
22	0.000	0.000	0.073	0.000	0.000	0.000	0.000	0.000	0.000	0.000
23	0.077	0.000	0.000	0.038	0.000	0.000	0.000	0.000	0.000	0.000
24	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
25	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
26	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
27	0.003	0.021	0.000	0.000	0.011	0.000	0.000	0.752	0.000	0.000
28	0.001	0.000	0.013	0.000	0.000	0.029	0.000	0.000	0.751	0.000
29	0.012	0.000	0.000	0.005	0.000	0.000	0.030	0.000	0.000	0.766
30	0.003	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.142	0.000
31	0.002	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.005	0.178
32	0.003	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.011
33	0.000	0.063	0.000	0.000	0.009	0.000	0.000	0.667	0.000	0.000
34	0.000	0.000	0.035	0.000	0.000	0.035	0.000	0.000	0.752	0.000
35	0.008	0.000	0.000	0.068	0.000	0.000	0.015	0.000	0.000	0.644
36	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
37	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
38	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
39	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
40	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
41	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
42	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
43	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
44	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
45	0.000	0.226	0.000	0.000	0.057	0.000	0.000	0.434	0.000	0.000
46	0.022	0.000	0.000	0.000	0.000	0.065	0.000	0.000	0.478	0.000
47	0.000	0.000	0.000	0.000	0.000	0.000	0.175	0.000	0.000	0.525

Table A.4: Transition table $T(s, a, s')$ for $a = 0$ (part 3).

	30	31	32	33	34	35	36	37	38	39
0	0.006	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
1	0.000	0.186	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
2	0.000	0.000	0.120	0.000	0.000	0.000	0.000	0.000	0.000	0.000
3	0.003	0.000	0.000	0.037	0.000	0.000	0.000	0.000	0.000	0.000
4	0.000	0.008	0.000	0.000	0.051	0.000	0.000	0.000	0.000	0.000
5	0.000	0.000	0.008	0.000	0.000	0.041	0.000	0.000	0.000	0.000
6	0.048	0.000	0.000	0.001	0.000	0.000	0.000	0.000	0.000	0.000
7	0.000	0.157	0.000	0.000	0.008	0.000	0.000	0.000	0.000	0.000
8	0.000	0.000	0.142	0.000	0.000	0.008	0.000	0.000	0.000	0.000
9	0.003	0.000	0.000	0.003	0.000	0.000	0.000	0.000	0.000	0.000
10	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
11	0.000	0.000	0.012	0.000	0.000	0.000	0.000	0.000	0.000	0.000
12	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
13	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
14	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
15	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
16	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
17	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
18	0.168	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
19	0.000	0.323	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
20	0.000	0.000	0.294	0.000	0.000	0.000	0.000	0.000	0.000	0.000
21	0.023	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
22	0.000	0.049	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
23	0.000	0.000	0.038	0.000	0.000	0.000	0.000	0.000	0.000	0.000
24	0.889	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
25	0.000	1.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
26	0.000	0.000	1.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
27	0.013	0.000	0.000	0.085	0.000	0.000	0.000	0.000	0.000	0.000
28	0.000	0.010	0.000	0.000	0.086	0.000	0.000	0.000	0.000	0.000
29	0.000	0.000	0.015	0.000	0.000	0.082	0.000	0.000	0.000	0.000
30	0.654	0.010	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
31	0.000	0.657	0.012	0.000	0.002	0.002	0.000	0.000	0.000	0.000
32	0.000	0.000	0.888	0.000	0.000	0.008	0.000	0.000	0.000	0.000
33	0.018	0.000	0.000	0.099	0.000	0.000	0.000	0.000	0.000	0.000
34	0.000	0.009	0.000	0.000	0.071	0.000	0.000	0.000	0.000	0.000
35	0.000	0.000	0.015	0.000	0.000	0.091	0.000	0.000	0.000	0.000
36	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
37	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
38	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
39	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
40	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
41	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
42	0.831	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
43	0.000	0.957	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
44	0.000	0.000	0.878	0.000	0.000	0.000	0.000	0.000	0.000	0.000
45	0.019	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
46	0.000	0.087	0.000	0.000	0.087	0.000	0.000	0.000	0.000	0.000
47	0.000	0.000	0.025	0.000	0.000	0.050	0.000	0.000	0.000	0.000

Table A.5: Transition table $T(s, a, s')$ for $a = 0$ (part 4).

	40	41	42	43	44	45	46	47
0	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
1	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
2	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
3	0.000	0.000	0.016	0.000	0.000	0.006	0.000	0.000
4	0.000	0.000	0.000	0.014	0.002	0.000	0.014	0.000
5	0.000	0.002	0.000	0.000	0.024	0.000	0.000	0.009
6	0.000	0.000	0.001	0.000	0.003	0.000	0.000	0.000
7	0.000	0.000	0.000	0.004	0.016	0.000	0.000	0.000
8	0.000	0.000	0.000	0.000	0.008	0.000	0.000	0.000
9	0.000	0.000	0.031	0.000	0.000	0.031	0.000	0.000
10	0.000	0.000	0.000	0.064	0.000	0.000	0.096	0.000
11	0.000	0.000	0.000	0.000	0.047	0.000	0.000	0.070
12	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
13	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
14	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
15	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
16	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
17	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
18	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
19	0.000	0.000	0.000	0.000	0.032	0.000	0.000	0.000
20	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
21	0.000	0.000	0.045	0.000	0.000	0.075	0.000	0.000
22	0.000	0.000	0.000	0.098	0.000	0.000	0.024	0.000
23	0.000	0.000	0.000	0.000	0.077	0.000	0.000	0.038
24	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
25	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
26	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
27	0.000	0.000	0.019	0.000	0.000	0.029	0.000	0.000
28	0.000	0.000	0.000	0.021	0.000	0.000	0.021	0.000
29	0.000	0.000	0.000	0.000	0.039	0.000	0.000	0.013
30	0.000	0.000	0.006	0.000	0.019	0.000	0.000	0.000
31	0.000	0.000	0.000	0.015	0.034	0.000	0.000	0.000
32	0.000	0.000	0.000	0.000	0.052	0.000	0.000	0.000
33	0.000	0.000	0.027	0.000	0.000	0.072	0.000	0.000
34	0.000	0.000	0.000	0.062	0.000	0.000	0.018	0.000
35	0.000	0.000	0.000	0.000	0.068	0.000	0.000	0.091
36	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
37	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
38	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
39	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
40	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
41	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
42	0.000	0.000	0.000	0.000	0.034	0.000	0.000	0.000
43	0.000	0.000	0.000	0.000	0.014	0.000	0.000	0.000
44	0.000	0.000	0.000	0.000	0.122	0.000	0.000	0.000
45	0.000	0.000	0.057	0.000	0.000	0.019	0.000	0.000
46	0.000	0.000	0.000	0.174	0.000	0.000	0.065	0.000
47	0.000	0.025	0.000	0.000	0.075	0.000	0.000	0.125

Table A.6: Transition table $T(s, a, s')$ for $a = 0$ (part 5).

	0	1	2	3	4	5	6	7	8	9
0	0.000	0.000	0.000	0.000	0.000	0.000	0.883	0.000	0.000	0.000
1	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.822	0.000	0.000
2	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.839	0.000
3	0.022	0.000	0.000	0.737	0.000	0.000	0.012	0.000	0.000	0.139
4	0.000	0.037	0.000	0.000	0.765	0.000	0.000	0.011	0.000	0.000
5	0.000	0.000	0.043	0.000	0.000	0.774	0.000	0.000	0.005	0.000
6	0.000	0.000	0.000	0.092	0.045	0.000	0.789	0.004	0.000	0.004
7	0.000	0.000	0.000	0.000	0.037	0.266	0.000	0.494	0.022	0.000
8	0.000	0.000	0.000	0.000	0.000	0.036	0.000	0.000	0.768	0.000
9	0.011	0.000	0.000	0.645	0.000	0.000	0.005	0.000	0.000	0.140
10	0.000	0.016	0.000	0.000	0.720	0.000	0.000	0.000	0.000	0.000
11	0.000	0.000	0.012	0.000	0.000	0.647	0.000	0.000	0.012	0.000
12	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
13	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
14	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
15	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
16	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
17	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
18	0.000	0.000	0.000	0.009	0.000	0.000	0.804	0.000	0.000	0.009
19	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.676	0.029	0.000
20	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.714	0.000
21	0.022	0.000	0.000	0.434	0.000	0.000	0.007	0.000	0.000	0.088
22	0.000	0.175	0.000	0.000	0.325	0.000	0.000	0.025	0.000	0.000
23	0.000	0.000	0.048	0.000	0.000	0.714	0.000	0.000	0.048	0.000
24	0.000	0.000	0.000	0.000	0.000	0.000	0.083	0.000	0.000	0.000
25	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.034	0.000	0.000
26	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
27	0.000	0.000	0.000	0.000	0.000	0.000	0.011	0.000	0.000	0.047
28	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.008	0.000	0.000
29	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.006	0.000
30	0.000	0.000	0.000	0.000	0.000	0.000	0.174	0.000	0.000	0.000
31	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.070	0.000	0.000
32	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.036	0.000
33	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.017
34	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
35	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.010	0.000
36	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
37	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
38	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
39	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
40	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
41	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
42	0.000	0.000	0.000	0.000	0.000	0.000	0.175	0.000	0.000	0.000
43	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.081	0.000	0.000
44	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.040	0.000
45	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
46	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
47	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000

Table A.7: Transition table $T(s, a, s')$ for $a = 1$ (part 1).

	10	11	12	13	14	15	16	17	18	19
0	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.017	0.000
1	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
2	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
3	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.014	0.000
4	0.085	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.005
5	0.000	0.061	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
6	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.011	0.000
7	0.004	0.004	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.004
8	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
9	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.032	0.000
10	0.056	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.024
11	0.000	0.129	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
12	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
13	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
14	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
15	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
16	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
17	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
18	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.009	0.000
19	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
20	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
21	0.000	0.000	0.000	0.000	0.000	0.007	0.000	0.000	0.125	0.000
22	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.050
23	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
24	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
25	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
26	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
27	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.019	0.000
28	0.047	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.017
29	0.000	0.046	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
30	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.005	0.000
31	0.002	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.004
32	0.000	0.003	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
33	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.025	0.000
34	0.009	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.043
35	0.000	0.019	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
36	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
37	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
38	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
39	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
40	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
41	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
42	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.013	0.000
43	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
44	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
45	0.000	0.000	0.000	0.000	0.000	0.019	0.000	0.000	0.094	0.000
46	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.068

Table A.8: Transition table $T(s, a, s')$ for $a = 1$ (part 2).

	20	21	22	23	24	25	26	27	28	29
0	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
1	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
2	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
3	0.002	0.020	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
4	0.001	0.000	0.009	0.000	0.000	0.000	0.000	0.000	0.000	0.000
5	0.006	0.000	0.000	0.012	0.000	0.000	0.000	0.000	0.000	0.000
6	0.002	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
7	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
8	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
9	0.000	0.089	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
10	0.008	0.000	0.064	0.000	0.000	0.000	0.000	0.000	0.000	0.000
11	0.035	0.000	0.000	0.024	0.000	0.000	0.000	0.000	0.000	0.000
12	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
13	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
14	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
15	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
16	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
17	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
18	0.019	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
19	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
20	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
21	0.007	0.184	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
22	0.000	0.000	0.075	0.000	0.000	0.000	0.000	0.000	0.000	0.000
23	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
24	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
25	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
26	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
27	0.000	0.055	0.000	0.000	0.030	0.000	0.000	0.721	0.000	0.000
28	0.001	0.000	0.017	0.000	0.000	0.042	0.000	0.000	0.738	0.000
29	0.011	0.000	0.000	0.010	0.000	0.000	0.027	0.000	0.000	0.757
30	0.003	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.113	0.000
31	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.010	0.191
32	0.006	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.017
33	0.000	0.176	0.000	0.000	0.008	0.000	0.000	0.597	0.000	0.000
34	0.000	0.000	0.103	0.000	0.000	0.017	0.000	0.000	0.684	0.000
35	0.029	0.000	0.000	0.029	0.000	0.000	0.019	0.000	0.000	0.680
36	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
37	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
38	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
39	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
40	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
41	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
42	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
43	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
44	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
45	0.000	0.189	0.000	0.000	0.019	0.000	0.000	0.509	0.000	0.000
46	0.000	0.000	0.091	0.000	0.000	0.136	0.000	0.000	0.455	0.000
47	0.029	0.000	0.000	0.057	0.000	0.000	0.086	0.000	0.000	0.629

Table A.9: Transition table $T(s, a, s')$ for $a = 1$ (part 3).

	30	31	32	33	34	35	36	37	38	39
0	0.100	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
1	0.000	0.178	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
2	0.000	0.000	0.161	0.000	0.000	0.000	0.000	0.000	0.000	0.000
3	0.004	0.000	0.000	0.028	0.000	0.000	0.000	0.000	0.000	0.000
4	0.000	0.006	0.000	0.000	0.038	0.000	0.000	0.000	0.000	0.000
5	0.000	0.000	0.011	0.000	0.000	0.056	0.000	0.000	0.000	0.000
6	0.049	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
7	0.000	0.155	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
8	0.000	0.000	0.188	0.000	0.000	0.000	0.000	0.000	0.000	0.000
9	0.000	0.000	0.000	0.013	0.000	0.000	0.000	0.000	0.000	0.000
10	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
11	0.000	0.000	0.000	0.000	0.000	0.012	0.000	0.000	0.000	0.000
12	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
13	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
14	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
15	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
16	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
17	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
18	0.150	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
19	0.000	0.294	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
20	0.000	0.000	0.238	0.000	0.000	0.000	0.000	0.000	0.000	0.000
21	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
22	0.000	0.025	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
23	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
24	0.917	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
25	0.000	0.966	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
26	0.000	0.000	0.969	0.000	0.000	0.000	0.000	0.000	0.000	0.000
27	0.014	0.000	0.000	0.074	0.000	0.000	0.000	0.000	0.000	0.000
28	0.000	0.010	0.000	0.000	0.078	0.000	0.000	0.000	0.000	0.000
29	0.000	0.000	0.012	0.000	0.000	0.080	0.000	0.000	0.000	0.000
30	0.661	0.010	0.000	0.000	0.002	0.000	0.000	0.000	0.000	0.000
31	0.000	0.665	0.022	0.000	0.002	0.000	0.000	0.000	0.000	0.000
32	0.000	0.000	0.881	0.000	0.000	0.006	0.000	0.000	0.000	0.000
33	0.008	0.000	0.000	0.084	0.000	0.000	0.000	0.000	0.000	0.000
34	0.000	0.009	0.000	0.000	0.017	0.000	0.000	0.000	0.000	0.000
35	0.000	0.000	0.019	0.000	0.000	0.097	0.000	0.000	0.000	0.000
36	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
37	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
38	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
39	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
40	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
41	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
42	0.800	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
43	0.000	0.905	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
44	0.000	0.000	0.900	0.000	0.000	0.000	0.000	0.000	0.000	0.000
45	0.019	0.000	0.000	0.057	0.000	0.000	0.000	0.000	0.000	0.000
46	0.000	0.023	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
47	0.000	0.000	0.114	0.000	0.000	0.000	0.000	0.000	0.000	0.000

Table A.10: Transition table $T(s, a, s')$ for $a = 1$ (part 4).

	40	41	42	43	44	45	46	47
0	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
1	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
2	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
3	0.000	0.000	0.013	0.000	0.000	0.007	0.000	0.000
4	0.000	0.000	0.000	0.025	0.001	0.000	0.017	0.000
5	0.000	0.000	0.000	0.000	0.020	0.000	0.000	0.012
6	0.000	0.000	0.003	0.000	0.001	0.000	0.000	0.000
7	0.000	0.000	0.000	0.004	0.011	0.000	0.000	0.000
8	0.000	0.000	0.000	0.000	0.009	0.000	0.000	0.000
9	0.000	0.000	0.022	0.000	0.000	0.043	0.000	0.000
10	0.000	0.000	0.000	0.072	0.008	0.000	0.032	0.000
11	0.000	0.000	0.000	0.000	0.059	0.000	0.000	0.071
12	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
13	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
14	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
15	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
16	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
17	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
18	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
19	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
20	0.000	0.000	0.000	0.000	0.048	0.000	0.000	0.000
21	0.000	0.000	0.074	0.000	0.000	0.051	0.000	0.000
22	0.000	0.000	0.000	0.175	0.075	0.000	0.075	0.000
23	0.000	0.000	0.000	0.000	0.095	0.000	0.000	0.095
24	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
25	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
26	0.000	0.000	0.000	0.000	0.031	0.000	0.000	0.000
27	0.000	0.000	0.011	0.000	0.003	0.016	0.000	0.000
28	0.000	0.000	0.000	0.025	0.004	0.000	0.014	0.000
29	0.000	0.000	0.000	0.000	0.038	0.000	0.000	0.011
30	0.000	0.000	0.010	0.002	0.021	0.000	0.000	0.000
31	0.000	0.000	0.000	0.004	0.030	0.000	0.000	0.000
32	0.000	0.000	0.000	0.000	0.052	0.000	0.000	0.000
33	0.000	0.000	0.034	0.000	0.000	0.050	0.000	0.000
34	0.000	0.000	0.000	0.051	0.017	0.000	0.051	0.000
35	0.000	0.000	0.000	0.000	0.029	0.000	0.000	0.068
36	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
37	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
38	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
39	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
40	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
41	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
42	0.000	0.000	0.013	0.000	0.000	0.000	0.000	0.000
43	0.000	0.000	0.000	0.000	0.014	0.000	0.000	0.000
44	0.000	0.000	0.000	0.000	0.060	0.000	0.000	0.000
45	0.000	0.000	0.038	0.000	0.000	0.057	0.000	0.000
46	0.000	0.000	0.000	0.159	0.023	0.000	0.045	0.000
47	0.000	0.000	0.000	0.000	0.057	0.000	0.000	0.029

Table A.11: Transition table $T(s, a, s')$ for $a = 1$ (part 5).

Bibliography

[1] H. L. O'Brien and E. G. Toms, "What is user engagement? a conceptual framework for defining user engagement with technology," *Journal of the American society for Information Science and Technology*, vol. 59, no. 6, pp. 938–955, 2008.

[2] M. Rodriguez-Soto, M. Lopez-Sanchez, and J. A. Rodriguez-Aguilar, "Guaranteeing the learning of ethical behaviour through multi-objective reinforcement learning," 2021.

[3] J.-A. Cervantes, S. López, L.-F. Rodríguez, S. Cervantes, F. Cervantes, and F. Ramos, "Artificial moral agents: A survey of the current status," *Science and Engineering Ethics*, vol. 26, no. 2, pp. 501–532, 2020.

[4] J. R. Keebler, W. J. Shelstad, D. C. Smith, B. S. Chaparro, and M. H. Phan, "Validation of the guess-18: a short version of the game user experience satisfaction scale (guess)," *Journal of Usability Studies*, vol. 16, no. 1, p. 49, 2020.

[5] A. Beck, B. Stevens, K. A. Bard, and L. Cañamero, "Emotional body language displayed by artificial agents," *ACM Transactions on Interactive Intelligent Systems (TiiS)*, vol. 2, no. 1, pp. 1–29, 2012.

[6] M. Anderson, S. L. Anderson, and C. Armen, "An approach to computing ethics," *IEEE Intelligent Systems*, vol. 21, no. 4, pp. 56–63, 2006.

[7] I. Rodríguez and A. Puig, "Open the microphone, please! conversational ux evaluation in virtual reality," in *Workshop 'Evaluating user experiences in mixed reality' in CHI'21*, 2021.

[8] I. Rodríguez, A. Puig, and À. Rodríguez, "Towards adaptive gamification: A method using dynamic player profile and a case study," *Applied Sciences*, vol. 12, no. 1, p. 486, 2022.

[9] A. Puig, I. Rodríguez, J. Baldeón, and S. Múria, "Children building and having fun while they learn geometry," *Computer Applications in Engineering Education*, 2021.

[10] M. Frutos-Pascual and B. G. Zapirain, "Review of the use of ai techniques in serious games: Decision making and machine learning," *IEEE Transactions on Computational Intelligence and AI in Games*, vol. 9, no. 2, pp. 133–152, 2015.

[11] D. Tellols, M. Lopez-Sanchez, I. Rodríguez, P. Almajano, and A. Puig, "Enhancing sentient embodied conversational agents with machine learning," *Pattern Recognition Letters*, vol. 129, pp. 317–323, 2020.

[12] A. Steinmaurer, M. Sackl, and C. Gütl, "Engagement in in-game questionnaires-perspectives from users and experts," in *2021 7th International Conference of the Immersive Learning Research Network (iLRN)*. IEEE, 2021, pp. 1–7.

[13] P. Van Rosmalen, J. Eikelboom, E. Bloemers, K. Van Winzum, and P. Spronck, "Towards a game-chatbot: Extending the interaction in serious games," 2012.

[14] P. Almajano, M. L. Sanchez, I. Rodriguez, and E. Mayas, "Including conversational agents into structured hybrid 3d virtual environments," *IEEE Latin America Transactions*, vol. 13, no. 2, pp. 523–531, 2015.

[15] J. Casas-Roma and J. Conesa, "Towards the design of ethically-aware pedagogical conversational agents," in *International Conference on P2P, Parallel, Grid, Cloud and Internet Computing*. Springer, 2020, pp. 188–198.

[16] R. Arrabales, "Perla: a conversational agent for depression screening in digital ecosystems. design, implementation and validation," *arXiv preprint arXiv:2008.12875*, 2020.

[17] K. Shao, Z. Tang, Y. Zhu, N. Li, and D. Zhao, "A survey of deep reinforcement learning in video games," *arXiv preprint arXiv:1912.10944*, 2019.

[18] R. Costa, I. Mendonça, and D. Souza, "Exploring the intelligent agents for controlling user navigation in 3d games for cognitive stimulation," in *8th International Conference on Disability, Virtual Reality and Associated Technologies*, vol. 1, 2010, pp. 1–6.

[19] F. E. Santos, A. P. Bastos, L. C. Andrade, K. Revoredo, and P. Mattos, "Assessment of adhd through a computer game: an experiment with a sample of students," in *2011 Third International Conference on Games and Virtual Worlds for Serious Applications*. IEEE, 2011, pp. 104–111.

[20] F. M. de Oliveira, R. S. Lanzillotti, R. M. M. da Costa, R. Gonçalves, P. Ventura, and L. A. V. de Carvalho, "Arvet and saptept: A virtual environment and a system supported by fuzzy logic in virtual reality exposure therapy for ptsd patients," in *2012 12th International Conference on Computational Science and Its Applications*. IEEE, 2012, pp. 103–107.

[21] A. Raffert, M. Zaharia, and T. Griffiths, "Optimally designing games for cognitive science research," in *Proceedings of the Annual Meeting of the Cognitive Science Society*, vol. 34, no. 34, 2012.

[22] P. Jepp, M. Fradinho, and J. M. Pereira, "An agent framework for a modular serious game," in *2010 Second International Conference on Games and Virtual Worlds for Serious Applications*. IEEE, 2010, pp. 19–26.

[23] A. Ward, M. McKeown, C. Utay, O. Medvedeva, and R. Crowley, "Interactive stories and motivation to read in the raft dyslexia fluency tutor," in *International Conference on Intelligent Virtual Agents*. Springer, 2012, pp. 260–267.

[24] C. T. Tan, A. Johnston, K. Ballard, S. Ferguson, and D. Perera-Schulz, "speak-man: towards popular gameplay for speech therapy," in *Proceedings of The 9th Australasian Conference on Interactive Entertainment: Matters of Life and Death*, 2013, pp. 1–4.

[25] V. Wattanasoontorn, I. Boada, C. Blavi, and M. Sbert, "The framework of a life support simulation application," *Procedia Computer Science*, vol. 15, pp. 293–294, 2012.

[26] J. Feldman, A. Monteserin, and A. Amandi, "Detecting students' perception style by using games," *Computers & Education*, vol. 71, pp. 14–22, 2014.

[27] L. Zhang, J. W. Wade, D. Bian, A. Swanson, Z. Warren, and N. Sarkar, "Data fusion for difficulty adjustment in an adaptive virtual reality game system for autism intervention," in *International conference on human-computer interaction*. Springer, 2014, pp. 648–652.

[28] D. Puzenat and I. Verlut, "Behavior analysis through games using artificial neural networks," in *2010 Third International Conference on Advances in Computer-Human Interactions*. IEEE, 2010, pp. 134–138.

[29] G. Rebolledo-Mendez, S. de Freitas, and A. R. G. Gaona, "A model of motivation based on empathy for ai-driven avatars in virtual worlds," in *2009 Conference in Games and Virtual Worlds for Serious Applications*. IEEE, 2009, pp. 5–11.

[30] I. Hulpuş, C. Hayes, and M. O. Fradinho, "A framework for personalised learning-plan recommendations in game-based learning," in *Recommender Systems for Technology Enhanced Learning*. Springer, 2014, pp. 99–122.

[31] S. Bernardini, K. Porayska-Pomsta, and H. Sampath, "Designing an intelligent virtual agent for social communication in autism," in *Ninth Artificial Intelligence and Interactive Digital Entertainment Conference*, 2013.

[32] A. Singla, A. N. Rafferty, G. Radanovic, and N. T. Heffernan, "Reinforcement learning for education: Opportunities and challenges," *arXiv preprint arXiv:2107.08828*, 2021.

[33] J. He-Yueya and A. Singla, "Quizzing policy using reinforcement learning for inferring the student knowledge state." *International Educational Data Mining Society*, 2021.

[34] S. Doroudi, V. Aleven, E. Brunskill, S. Kimball, J. Long, S. Ludovise *et al.*, "Where's the reward? a review of reinforcement learning for instructional sequencing." in *ITS Workshops*, 2018, p. 147.

[35] A. Efremov, A. Ghosh, and A. Singla, "Zero-shot learning of hint policy via reinforcement learning and program synthesis." *International Educational Data Mining Society*, 2020.

[36] A. N. Rafferty, H. Ying, and J. J. Williams, "Bandit assignment for educational experiments: Benefits to students versus statistical power," in *International Conference on Artificial Intelligence in Education*. Springer, 2018, pp. 286–290.

[37] X. Yang, G. Zhou, M. Taub, R. Azevedo, and M. Chi, "Student subtyping via em-inverse reinforcement learning." *International Educational Data Mining Society*, 2020.

[38] X. Zhu, A. Singla, S. Zilles, and A. N. Rafferty, "An overview of machine teaching," *arXiv preprint arXiv:1801.05927*, 2018.

[39] L. Gisslén, A. Eakins, C. Gordillo, J. Bergdahl, and K. Tollmar, "Adversarial reinforcement learning for procedural content generation," in *2021 IEEE Conference on Games (CoG)*. IEEE, 2021, pp. 1–8.

[40] T. Barnes and J. Stamper, "Toward automatic hint generation for logic proof tutoring using historical student data," in *International conference on intelligent tutoring systems*. Springer, 2008, pp. 373–382.

[41] T. Mandel, Y.-E. Liu, S. Levine, E. Brunskill, and Z. Popovic, "Offline policy evaluation across representations with applications to educational games." in *AAMAS*, vol. 1077, 2014.

[42] D. A. Deza Veliz, "Policies selection for pedagogical agent based on the roulette wheel algorithm," 2021.

[43] G. Veletsianos and G. S. Russell, "Pedagogical agents," in *Handbook of research on educational communications and technology*. Springer, 2014, pp. 759–769.

[44] J. Weizenbaum, "Eliza—a computer program for the study of natural language communication between man and machine," *Communications of the ACM*, vol. 9, no. 1, pp. 36–45, 1966.

[45] T. Strohmann, "From assistance to companionship-designing virtual companions," Ph.D. dissertation, 2021.

[46] K. Mo, Y. Zhang, S. Li, J. Li, and Q. Yang, "Personalizing a dialogue system with transfer reinforcement learning," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 32, no. 1, 2018.

[47] S. Singh, D. Litman, M. Kearns, and M. Walker, "Optimizing dialogue management with reinforcement learning: Experiments with the njfun system," *Journal of Artificial Intelligence Research*, vol. 16, pp. 105–133, 2002.

[48] M. S. Satu, M. H. Parvez *et al.*, "Review of integrated applications with aiml based chatbot," in *2015 International Conference on Computer and Information Engineering (ICCIE)*. IEEE, 2015, pp. 87–90.

[49] M. A. Walker, "An application of reinforcement learning to dialogue strategy selection in a spoken dialogue system for email," *Journal of Artificial Intelligence Research*, vol. 12, pp. 387–416, 2000.

[50] Z. Xiao, M. X. Zhou, Q. V. Liao, G. Mark, C. Chi, W. Chen, and H. Yang, "Tell me about yourself: Using an ai-powered chatbot to conduct conversational surveys with open-ended questions," *ACM Transactions on Computer-Human Interaction (TOCHI)*, vol. 27, no. 3, pp. 1–37, 2020.

[51] X. Han, M. Zhou, M. J. Turner, and T. Yeh, "Designing effective interview chatbots: Automatic chatbot profiling and design suggestion generation for chatbot debugging," in *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*, 2021, pp. 1–15.

[52] D. Alexandrovsky, S. Putze, M. Bonfert, S. Höffner, P. Michelmann, D. Wenig, R. Malaka, and J. D. Smeddinck, "Examining design choices of questionnaires in vr user studies," in *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, 2020, pp. 1–21.

[53] V. Scotti, R. Tedesco, and L. Sbattella, "A modular data-driven architecture for empathetic conversational agents," in *2021 IEEE International Conference on Big Data and Smart Computing (BigComp)*. IEEE, 2021, pp. 365–368.

[54] V. J. Shute, "Stealth assessment in computer-based games to support learning," *Computer games and instruction*, vol. 55, no. 2, pp. 503–524, 2011.

[55] I. Gabriel, "Artificial intelligence, values, and alignment," *Minds and machines*, vol. 30, no. 3, pp. 411–437, 2020.

[56] C. Sierra, N. Osman, P. Noriega, J. Sabater-Mir, and A. Perelló, "Value alignment: a formal approach," *arXiv preprint arXiv:2110.09240*, 2021.

[57] J. H. Moor, "The nature, importance, and difficulty of machine ethics," *IEEE intelligent systems*, vol. 21, no. 4, pp. 18–21, 2006.

[58] M. Rodriguez-Soto, M. Lopez-Sanchez, and J. A. Rodriguez-Aguilar, "A structural solution to sequential moral dilemmas," in *Proceedings of the 19th International Conference on Autonomous Agents and MultiAgent Systems*, 2020, pp. 1152–1160.

[59] ——, "Multi-objective reinforcement learning for designing ethical environments," in *Proceedings of the 30th International Joint Conference on Artificial Intelligence*, 2021, pp. 1–7.

[60] K. Doherty and G. Doherty, "Engagement in hci: conception, theory and measurement," *ACM Computing Surveys (CSUR)*, vol. 51, no. 5, pp. 1–39, 2018.

[61] C. Peters, G. Castellano, and S. De Freitas, "An exploration of user engagement in hci," in *Proceedings of the International Workshop on Affective-Aware Virtual Agents and Social Robots*, 2009, pp. 1–3.

[62] R. W. Picard, *Affective computing*. MIT press, 2000.

[63] S. Attfield, G. Kazai, M. Lalmas, and B. Piwowarski, "Towards a science of user engagement (position paper)," in *WSDM workshop on user modelling for Web applications*, 2011, pp. 9–12.

[64] F. Dobrian, V. Sekar, A. Awan, I. Stoica, D. Joseph, A. Ganjam, J. Zhan, and H. Zhang, "Understanding the impact of video quality on user engagement," *ACM SIGCOMM computer communication review*, vol. 41, no. 4, pp. 362–373, 2011.

[65] C. L. Sidner and M. Dzikovska, "Human-robot interaction: Engagement between humans and robots for hosting activities," in *Proceedings. Fourth IEEE International Conference on Multimodal Interfaces*. IEEE, 2002, pp. 123–128.

[66] E. Brown and P. Cairns, "A grounded investigation of game immersion," in *CHI'04 extended abstracts on Human factors in computing systems*, 2004, pp. 1297–1300.

[67] B. Cowley, D. Charles, M. Black, and R. Hickey, "Toward an understanding of flow in video games," *Computers in Entertainment (CIE)*, vol. 6, no. 2, pp. 1–27, 2008.

[68] H. Lowood, "Videogames in computer space: The complex history of pong," *IEEE Annals of the History of Computing*, vol. 31, no. 3, pp. 5–19, 2009.

[69] K. Poels, Y. de Kort, and W. IJsselsteijn, *D3.3 : Game Experience Questionnaire: development of a self-report measure to assess the psychological impact of digital games*. Technische Universiteit Eindhoven, 2007.

[70] E. L.-C. Law, F. Brühlmann, and E. D. Mekler, "Systematic review and validation of the game experience questionnaire (geq)-implications for citation and reporting practice," in *Proceedings of the 2018 Annual Symposium on Computer-Human Interaction in Play*, 2018, pp. 257–270.

[71] M. H. Phan, J. R. Keebler, and B. S. Chaparro, "The development and validation of the game user experience satisfaction scale (guess)," *Human factors*, vol. 58, no. 8, pp. 1217–1247, 2016.

[72] F. Brühlmann and G.-M. Schmid, "How to measure the game experience? analysis of the factor structure of two questionnaires," in *Proceedings of the 33rd annual acm conference extended abstracts on human factors in computing systems*, 2015, pp. 1181–1186.

[73] D. Johnson, L. E. Nacke, and P. Wyeth, "All about that base: differing player experiences in video game genres and the unique case of moba games," in *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, 2015, pp. 2265–2274.

[74] D. Johnson, M. J. Gardner, and R. Perry, "Validation of two game experience scales: the player experience of need satisfaction (pens) and game experience questionnaire (geq)," *International Journal of Human-Computer Studies*, vol. 118, pp. 38–46, 2018.

[75] P. Cairns, A. L. Cox, M. Day, H. Martin, and T. Perryman, "Who but not where: The effect of social play on immersion in digital games," *International Journal of Human-Computer Studies*, vol. 71, no. 11, pp. 1069–1077, 2013.

[76] H. Candello, C. Pinhanez, and F. Figueiredo, "Typefaces and the perception of humanness in natural language chatbots," in *Proceedings of the 2017 chi conference on human factors in computing systems*, 2017, pp. 3476–3487.

[77] F. Cruz, G. I. Parisi, and S. Wermter, "Multi-modal feedback for affordance-driven interactive reinforcement learning," in *2018 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2018, pp. 1–8.

[78] F. Cruz, S. Magg, Y. Nagai, and S. Wermter, "Improving interactive reinforcement learning: What makes a good teacher?" *Connection Science*, vol. 30, no. 3, pp. 306–325, 2018.

[79] A. Bignold, F. Cruz, R. Dazeley, P. Vamplew, and C. Foale, "An evaluation methodology for interactive reinforcement learning with simulated users," *Biomimetics*, vol. 6, no. 1, p. 13, 2021.

[80] J. Schatzmann, K. Weilhammer, M. Stuttle, and S. Young, "A survey of statistical user simulation techniques for reinforcement-learning

of dialogue management strategies," *The knowledge engineering review*, vol. 21, no. 2, pp. 97–126, 2006.

[81] E. Levin, R. Pieraccini, and W. Eckert, "A stochastic model of human-machine interaction for learning dialog strategies," *IEEE Transactions on speech and audio processing*, vol. 8, no. 1, pp. 11–23, 2000.

[82] P. Compton, P. Preston, and B. Kang, "The use of simulated experts in evaluating knowledge acquisition," *University of Calgary*, 1995.

[83] K. Scheffler and S. Young, "Corpus-based dialogue simulation for automatic strategy learning and evaluation," in *Proc. NAACL Workshop on Adaptation in Dialogue Systems*, 2001, pp. 64–70.

[84] H. Cuayáhuitl, S. Renals, O. Lemon, and H. Shimodaira, "Reinforcement learning of dialogue strategies with hierarchical abstract machines," in *2006 IEEE Spoken Language Technology Workshop*. IEEE, 2006, pp. 182–185.

[85] L. A. Celiberto, C. H. Ribeiro, A. H. Costa, and R. A. Bianchi, "Heuristic reinforcement learning applied to robocup simulation agents," in *Robot Soccer World Cup*. Springer, 2007, pp. 220–227.

[86] A. Bignold, F. Cruz, R. Dazeley, P. Vamplew, and C. Foale, "Persistent rule-based interactive reinforcement learning," *Neural Computing and Applications*, pp. 1–18, 2021.

[87] K. Komatani, S. Ueno, T. Kawahara, and H. G. Okuno, "Flexible guidance generation using user model in spoken dialogue systems," in *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, 2003, pp. 256–263.

[88] K. Scheffler and S. Young, "Automatic learning of dialogue strategy using dialogue simulation and reinforcement learning," in *Proceedings of HLT*, vol. 2, 2002.

[89] D. Abel, J. MacGlashan, and M. L. Littman, "Reinforcement learning as a framework for ethical decision making," in *Workshops at the thirtieth AAAI conference on artificial intelligence*, 2016.

[90] T. M. Mitchell *et al.*, "Machine learning."

[91] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

[92] D. M. Roijers and S. Whiteson, "Multi-objective decision making," *Synthesis Lectures on Artificial Intelligence and Machine Learning*, vol. 11, no. 1, pp. 1–129, 2017.

[93] L. Barrett and S. Narayanan, "Learning all optimal policies with multiple criteria," in *Proceedings of the 25th international conference on Machine learning*, 2008, pp. 41–47.