AFFECTIVE DECISION-MAKING ENGINE FOR SOLVING COMPLEX PROBLEMS IN COMPUTER GAMES

HAFIZ BIN MOHD SARIM

UNIVERSITI KEBANGSAAN MALAYSIA

AFFECTIVE DECISION-MAKING ENGINE FOR SOLVING COMPLEX PROBLEMS IN COMPUTER GAMES

HAFIZ BIN MOHD SARIM

THESIS SUBMITTED IN FULFILMENT FOR THE DEGREE OF DOCTOR OF PHILOSOPHY

FACULTY OF INFORMATION SCIENCE AND TECHNOLOGY
UNIVERSITI KEBANGSAAN MALAYSIA
BANGI

2018

ENJIN PEMBUATAN KEPUTUSAN AFEKTIF UNTUK PENYELESAIAN MASALAH KOMPLEKS DALAM PERMAINAN KOMPUTER

HAFIZ BIN MOHD SARIM

TESIS YANG DIKEMUKAKAN UNTUK MEMPEROLEH
IJAZAH DOKTOR FALSAFAH

FAKULTI TEKNOLOGI DAN SAINS MAKLUMAT
UNIVERSITI KEBANGSAAN MALAYSIA
BANGI

2018

DECLARATION

I hereby declare that the work in this thesis is my own except for quotations and summaries which have been duly acknowledged.

06 August 2018

HAFIZ BIN MOHD SARIM
P51821

ACKNOWLEDGEMENTS

In the name of Allah, the Most Gracious, and Most Merciful.

I shower my praise upon you, oh Almighty God, for all the blessings of health, strength, patience, and wisdom, that you have saw fit to bestowed upon me, your humble servant. That I may complete the labours you have tasked me to accomplish, and overcome the challenges you have commanded me to endure, in this journey to complete my PhD research.

I would like to thank my research supervisors, Professor Dr Abdul Razak Hamdan, Professor Dr Azuraliza Abu Bakar, and Associate Professor Dr Zulaiha for all the knowledge they have shared, and the guidance they have provided and the help they have given me throughout the length and breadth of this research. I am most of all thankful for your patience and belief in me, and for care you have shown.

I am grateful for the facilities provided by Fakulti Teknologi dan Sains Maklumat, Universiti Kebangsaan Malaysia where this research was conducted. My gratitude goes out to the administration of FTSM UKM for accommodating my requests. I would also like to thank the staff and academicians of the Centre for Artificial Intelligence for your support. I am also most thankful to the people of the Data Mining and Optimization research group of FTSM UKM for being my friends and motivating me to complete my research.

I gratefully acknowledge the most generous financial support provided by the Ministry of Higher Education (KPT), the Public Service Department (JPA) and Universiti Kebangsaan Malaysia for providing the SLAI doctoral scholarship. I would also like to thank the Ministry of Science and Technology (MOSTI) for sponsoring this research under the FRGS Grant Scheme by providing me the UKM-TT-030FRGS0134-2010 project grant.

To my dear Mother, Hjh Zaharah Binti Ishak, I am forever grateful for the love and care you have given me and that I sorely needed to complete my work. To my brothers Iqbal and Zahrim, and to my sister Adlin, I thank you for caring and being with me. And finally, I dedicate this research to the memory of my late father, Allahyarham Hj Mohd Sarim Bin Mustajab, my pillar of strength and my guiding light, your love and care has brought me here. I dedicate this work to you, my dear father.

ABSTRACT

Affective agents are autonomous intelligent software agents that are programmed to achieve a set of goals by deciding upon action choices within a particular problem environment through the emulation of psychological affect, i.e. valenced 'feelings' elicited toward the current problem environment. The Affective Decision-Making Engine (ADME) used by affective agents is designed to allow it to cope with complex problem environments such as computer games. For this research, open, interactive, and context-sensitive environments found in games are models for complex problems. Current adaptive agents and classification agents face difficulty in coping with these environments: the problem presents neither a consistent dataset to optimize against, nor a stable population of elements to train behaviour. These factors further exacerbate their ability to fulfil all set goals. This research is motivated by the ability of living organisms to adapt and summarily classify completely new and unfamiliar situations similar to these complex problems. In organisms, emotions act as an internal cognitive mechanism to evaluate how good or bad any situation is to the organism. The organism is positively or negatively affected, and is driven to perform actions that mitigate this affect. To emulate this process of affect, ADME assigns an Affect Value (AV) for every feature possessed by observed elements or objects in the problem environment. The AV calculates the correlation coefficient of a feature against the change in each goal outcome value, in a table called an Affect Matrix (AM). Through the sum correlation, the AM summarizes each element's overall affect to each goal, to discover significant goals to prioritize. Later, a homeostat balances the significant goals with the best action for mitigating the affect. By using correlation coefficients, this research has found that the ADME is able to discover features that affect the agent's goal outcome values. Experimental results show that affective agents surpass state-transition and adaptive agents when converging towards best actions for significant goals in complex problems, though it requires greater memory usage. These research findings show that affective agents have great utility as single pass multi-goal reinforcement tools for unfamiliar and highly dynamic environments.

ABSTRAK

Agen afektif merupakan perisian agen cerdas berautonomi yang diaturcara untuk memenuhi suatu set matlamat dengan menentukan pilihan tindakan untuk sesuatu persekitaran masalah secara menteladani afek psikologi, iaitu valens perasaan yang diterbitkan dalam persekitaran masalah semasa. Enjin Pembuatan Keputusan Afektif (ADME) yang digunakan oleh agen afektif di rekabentuk untuk membolehkannya tangani persekitaran masalah kompleks seperti permainan komputer. Dalam penyelidikan ini, persekitaran yang terbuka, berinteraksi, dan peka-konteks yang ditemui dalam permainan komputer dijadikan model bagi masalah kompleks. Agen penyesuaian dan agen pengelasan semasa menghadapi kerumitan dalam menangani persekitaran sebegini: tiada set data tekal yang boleh dibekalkan untuk kerja pengoptimuman, dan tiada juga populasi unsur yang stabil yang boleh digunakan untuk melatih kelakuan. Faktor-faktor ini juga menyukarkan kerja untuk memenuhi seluruh set matlamat yang ditetapkan. Penyelidikan ini didorongkan oleh keupayaan organisma hidup menyesuaikan dan mengkelaskan secara ringkas situasi yang baru dan belum dikenali, seperti juga masalah kompleks sebegini. Dalam setiap organisma, emosi berperanan sebagai alat kognitif untuk menilai berapa baik atau burukkah situasi tersebut kepada organisma berkenaan. Organisma tersebut akan terkesan secara baik atau buruk, dan ia akan tergerak untuk melaksanakan tindakan yang mampu mengatasi kesan tersebut. Untuk menteladani proses afek ini, ADME meletakkan Nilai Afek (AV) pada setiap ciri yang dimiliki oleh setiap elemen atau objek yang kelihatan dalam persekitaran masalah. Nilai AV mengira pekali korelasi sesuatu ciri bardasarkan perubahan pada nilai matlamat semasa, dalam sebuah jadual yang dipanggil Matriks Afek (AM). Dengan menjumlahkan korelasi, AM meringkaskan afek keseluruhan setiap unsur terhadap setiap matlamat, untuk menemui matlamat penting yang perlu diutamakan. Kemudian, sebuah homeostat akan mengimbangi matlamat penting dengan tindakan terbaik untuk menangani kesan afeknya. Dengan menggunakan pekali korelasi, penyelidikan ini mendapati bahawa ADME mampu menemui ciri-ciri yang akan memberi-kesan pada nilai-niai matlamat agen. Keputusan ujikaji menunjukkan yang agen afektif mendahului agen peralihan keadaan dan penyesuaian semasa proses mencari tindakan terbaik untuk setiap matlamat penting dalam masalah kompleks. Namum begitu, ia memerlukan penggunaan ingatan utama yang lebih. Penemuan penyelidikan ini menunjukkan yang agen afektif mempunyai kebergunaan tinggi sebagai alat pengkukuhan matlamat-berbilang sekali-lalu, untuk persekitaran yang sentiasa berubah dan tidak dikenali.

TABLE OF CONTENTS

CHAPTER II LITERATURE REVIEW

LIST OF TABLES

LIST OF ILLUSTRATIONS

LIST OF ABBREVIATIONS

3GL	Third-Generation Programming Language
A*	A-Star Search
A.I.	Artificial Intelligence
AAAI	Association For The Advancement Of Artificial Intelligence
ADME	Affective Decision-Making Engine
AF	ADME-based Affective Agent
AH	Affect Homeostat
AI	Artificial Intelligence
ALEC	Asynchronous Learning By Emotion And Cognition Architecture
AM	Affect Matrix
API	Application Programming Interface
AV	Affect Value
BDI	Belief-Desire-Intention
COOP	Cooperate
DARE	Emotion-Based Robotic Agent Development Architecture
DEF	Defect
DLC	Downloadable Content
EB	Emotion-Based Architecture
EF	External Features
EMAI	Emotionally Motivated Artificial Intelligence
FPS	First-Person Shooters
FRGS	Fundamental Research Grant Scheme
FSM	Finite State Machine

FTSM	Fakulti Teknologi Dan Sains Maklumat
GDL	Game Description Language
GGP	General Game Playing
HCI	Human-Computer Interface
HUMAN	Human Players
IF	Internal Features
IPD	Iterated Prisoner's Dilemma
IPOMDP	Infinite Partially Observable Markov Decision Process
KM	Knowledge Model
KMA	Knowledge Model Approximator
KMM	Knowledge Model Matrix Manager
MDP	Markov Decision Process
MMORPG	Massively-Multiplayer Online Role-Playing Games
MSE	Mean Square Error
NPC	Non-Player Characters
NSMD	N-State Markov Decision State-Transition Agent
OCC	Ortony-Clore-Collins Model Of Emotions
ORIENT	Overcoming Refugee Integration With Empathic Novel Technology
POMDP	Partially Observable Markov Decision Process
QL	Q-Learning Adaptive Agent
RL	Reinforcement Learning
RPG	Role-Playing Games
SDK	Software Development Kit
SHMUPS	Shoot-Em-Up Games

SP	Simple Probabilistic State-Transition Agent
SST	Sum Of Squared Totals
TnHnRn	Test N, Hidden Rule N, Round N
TD	Temporal Difference
TDL	Temporal-Distance Learning Agent
UKM	Universiti Kebangsaan Malaysia
UML	Unified Modelling Language

CHAPTER I

INTRODUCTION

1.1 INTRODUCTION

This research aims to improve the adaptability of artificially intelligent virtual players, which are called 'Non-Player Characters' or NPCs, in computer games. Modern computer games frequently expand their game environment, either by adding new content or changing existing content. Programming an NPC to make decisions in such *scalable game environments* is difficult because it is impossible to know what future content or change will be introduced in the game. The problem that arises from this is the lack of fixed external states and known deterministic rules, which are required by the NPC to act as the basis of A.I decision making. The problem that needs to be solved is how to program a decision making algorithm for NPCs that does not require fixed external states, so that it can adapt to any future change to the game environment. The solution chosen by this research is to create a new agent framework for NPCs that emulates the biological process of emotional *affect*, in order to internalize the problem states and adapt its own internal rules using emotions. An affective agent framework based on these internalized states and rules will allow the NPC to identify what is important, regardless of how the game environment may change in the future.

1.2 RESEARCH BACKGROUND

The computer and video games industry has reached a market share of USD $137.9 billion (RM 569.87 billion) globally in April 2018 (Ell 2018), which represents a 112.15% increase since the first video game industry report was first released in

June 2011 (Bilton 2011). The sheer size of the industry has catalyzed the need for the application of new technology to make games more entertaining, engaging, and immersive to the human player. Artificial intelligence is one such technology that has been applied to computer and video games with greater frequency since the turn of the century, (Yannakakis & Togelius. 2018).

Artificial inteligence and machine learning techniques have been used in computer and video games as early as the 1950s (Shannon 1950, Rosales-Pulido 2016), to provide entertainment and challenge to the people who play these games. Modern computer games implement artificial intelligence for two primary reasons (Laird and van Lent 2001; Petrović 2018):

1. **Realism** - To simulate real-world phenomenon in games in order to make the game more realistic and enhance the illusion of reality.

2. **Intelligence** - To control the behaviour or computer generated virtual players, by emulating cognitive processes, in order provide challenge to the human player in the game environment.

'Non-Player Characters', or NPCs, are artificially intelligent software agents that embody these A,I. techniques, particularly in the form of computer-controlled game characters that interact with human players in the game (Millington & Funge 2009; Yannakakis & Togelius. 2018). These NPCs are programmed to exhibit the properties of intelligent agents, which are Persistence, Autonomy, Social Ability, and Reactivity (Woolridge 2002; Dignum et al. 2009). In a computer game, NPCs are the A.I players in games that acts as opponents, supporting characters, bystanders, animals, vehicles, and others. The use of A.I. driven NPCs in modern. The most common A.I algorithms for programming NPCs are summarized in Table 1.1 (Millington and Funge 2009):

Table 1.1 Most common A.I techniques for NPCs

Algorithm	General Technique	Earliest Examples	Modern Games	Requirements
Finite State Machines (FSM)	Evaluate the problem state condition before transitioning to a new behavioural state.	• Moore machines (Moore 1956, Staddon 2016) • Mealy machines (Mealy 1955, Vaandrager 2017)	• Doom (2016) • Destiny 2 (2017)	• Fully observable states (i.e. complete information) • Determinism
Reinforcement Learning	Apply reward or penalties to action choices, according to the degree of goal fulfilment.	• Q-learning (Watkins 1989, Zaremba et al. 2016) • TD Learning (Sutton 1988, Hertz et al. 2018)	• Sid Meier's Civilization VI (2016) • FIFA 18 (2017)	• Fixed choices and elements (e.g. the same actions or objects are always available)
Decision Trees	Simulate all possible move combinations up to n-ply and evaluate outcomes of leaf nodes.	• ID3 (Quinlan 1986, Wu et al. 2016) • C4.5 (Quinlan 1993, Witten et al. 2016)	• Warhammer 40k: Space Wolf (2014) • XCOM 2 (2016)	• Fully observable states (i.e. complete information) • Determinism
Search	Simulate path movement and propagate reward or error to paths that reach closer to goals.	• A* (Hart et al. 1968, Bast et al. 2016) • Minimax (von Neumann 1928, Aumann 2017)	• Ashes of the Singularity (2016) • Warhammer 40k: Dawn of War III (2017)	• Fully observable states (i.e. complete information) • Determinism

Source: Millington and Funge 2009

The A.I techniques listed Table 1.1 are the most preferred by the majority of game publishers for practical reasons: these A.I techniques are easy to program by NPC programmers, are subsequently easy to change and edit should the need arise (Millington and Funge 2009). It is also very simple to explain the decisions made by NPCs using these A.I algorithms. Therefore, if the behaviour of a particular NPC needs adjustment, then it is very easy for the NPC programmer to reconfigure the parameters of these A.I techniques to produce the desired behaviour.

Table 1.1 also shows the requirements for the effective use of these A.I. algorithms. In order to program the NPC to behave as intended, these A.I techniques all universally require the game environment to be fully observable, fixed and unchanging, and that the rules governing the behaviour of objects and interaction in the game to be fully known and deterministic.

1.3 THE PROBLEM DOMAIN: SCALABLE GAME ENVIRONMENTS

The difficulty that is faced by programmers that create and program NPCs for computer games is that modern computer games frequently expand beyond its initial design parameters, often for the purpose adding new content to the game in order to keep the human player entertained (Yannakakis & Togelius. 2018). For example. Figure 1.1 shows how one of the most popular games in ever created, *World of Warcraft* (2004), has been expanded at least six times over the span of 14 years in order to keep its peak player base of 12 million concurrent players entertained with new content.

Figure 1.1: Expanding video games with new content

Source: World of Warcraft, 2004

The common A.I algorithms used in current NPC design also face problems if the game arena dynamically changes over time, either through randomization, or due to some unknown rule that governs the change, which is hidden from NPCs view. This research coins the term "*Scalable Game Environments*" to describe these type of games. Scalable game environments are games that change its game play 'arena' over time by introducing new play areas, new game rules, new objects, new challenges, and new opponents, or by being highly dynamic.

1.3.1 The Research Problem: Lack of Fixed States and Deterministic Rules in Scalable Game Environments

Whenever the objects in a game's environment change, the game can be said to experience a 'state-change'. If the number of all possible state-changes are finite, and if the change in the game state is governed by some known permanent rule, the game can be said to have 'fixed states and deterministic rules' (Russell and Norvig 2015). The common A.I techniques used for NPCs can operate perfectly well under these conditions as they meet the requirements stated in Table 1.1.

However, if a game expands by constantly adding new content (i.e. objects) to the game environment, then the number of all possible state-changes increase, until they become seemingly infinite. Furthermore, if the rules that govern changes in game states are not known, or hidden from view, then the game can be said to 'lack fixed states and deterministic rules'. This is main problem introduced by scalable game environments, that needs to be solved in this research.

The common A.I. techniques used in NPCs all requires the game environment to have fixed observable (i.e. external) state. Each game state is used as a basis for iterative reinforcement and iterative classification by the A.I. techniques. If there is an unlimited number of possible game states, however, then the A.I techniques cannot repeat or experience the same game state more than once. This inhibits the algorithms ability to iteratively reinforce or iteratively classify better action choices based on these game states.

All scalable game environments lack fixed external states and deterministic rules. This problem will effectively prevent the common A.I techniques from reinforcing, classifying, or modelling the game environment. This characteristic of scalable game environments will effectively cause the NPCs to 'break' and fail due to the lack of information needed by its algorithm to function properly.

1.3.2 Justification for a New Agent Framework

In order to make decisions in a scalable game environments that are constantly changing with new content and dynamic rules, an NPC would have to rely on an internalized state evaluation system in order make decisions. This internalized state evaluation system would have to do the following:

1. The NPC will need a set of persistent internal features, which are part of its agent framework, which will become the basis of a state-evaluation and performance measurement.

2. The NPC would need to self-generate a set of rules to discover how to 'translate' the utility values of the non-persistent external features from the game, to utility values of its own persistent internal features.

3. Upon receiving any external stimuli, the NPC would need to predict the change to expected future utility of its own persistent internal features, and make a decision on an action choice.

At present, there does not yet exist an intelligent agent framework, which can be used to build NPCs, that relies solely on internalized feature states for making decisions. Therefore existing agent frameworks for NPCs that rely on the common A.I algorithms listed in Table 1.1 is bound to perform poorly in scalable game environments.

1.3.3 The Research Gap: Inability to Translate Varied External Feature States to Internal Feature States

The values of each feature that is entered as input into an artificial intelligence algorithm, as part of its decision-making evaluation function, represents the utility value that the feature holds (Haykin 1999; Zhang & Suganthan 2016). Two different feature types will hold different utility ranges, each with a different semantic value meaning. On their own, two separate feature types do not have a common utility

denomination. This is not usually a concern if the total number of feature types are fixed and known in a problem environment, like a game. The agent designer would manually provide the common utility denomination for the A.I algorithm used in the agent's framework, in the form of a weighted utility evaluation function (Haykin 1999; Zhang & Suganthan 2016).

However, in a scalable game environment these different feature types change over time, often leaving no opportunity for the agent designer to manually set feature weights, or no time for the agent to learn new weights through iterative reinforcements. The main cause of this is that the scalable game environment changes too quickly for the agent to experience the feature long enough for reinforcement to occur. Therefore, a common utility denomination in the form of a weight utility evaluation function, cannot be adequately constructed. This is the research gap that this research intends to fill.

The approach that is taken by this research to fill this research gap is to use *affective computing* techniques (Picard 1997; Fairclough 2017) in the agent's framework design. The intention is to give an NPC an internalized body state based on emotional affect, which can be used by the NPC as the basis and common denomination for making decisions in a highly dynamic scalable game environments. By fulfilling this research gap, an agent based on affective computing techniques, i.e. an affective agent, can be thrown in any untrained situation and still make rational decisions based on how it feels internally.

As it is not the aim of this research to discover the exact emotional affect phenomenon that contributes to better decisions, the research will model the agent’s decision-making algorithm based on an existing model and theory of emotional affect. The Somatic Marker Hypothesis (Damasio 1996; Kanbara & Fukunaga 2016), which is the emotional affect model used for this research, posits that emotional affect contributes to better decisions by:

1. Recording a feeling of what happens to the body upon experiencing stimuli

2. Invoking the feeling of what happens to the body in anticipation of the experienced stimuli

3. Altering the value of decisions that either promote or avoid the recurrence of the feeling.

A more detailed treatise of the Somatic Marker Hypothesis can be found in Chapter 2. These characteristics of emotional affect can be directly translated to a set of functional component for discorporate game agents (i.e. agents that lack bodies) as follows:

1. Identify how features in the game environment determine goal outcomes.

2. Predict how goals outcomes will change from the game environment features

3. Evaluate actions by determining which goals are important.

1.4 RESEARCH QUESTIONS

In the pursuit of an agent framework that will allow agents to adapt to new elements introduced in scalable game environments, while at the same time being able to make good decisions, the primary question this research now asks is as follows:

RQ1: "What are the necessary agent components that are required to emulate the benefits of emotional affect when making decisions, which will allow game AI agents to operate well in scalable game environments?"

From this question, this research aims to discover the simplest possible design for an agent framework that can mimic the benefits of emotional affect in decision making. This question focuses on the functional characteristics observed from emotional affect that must be mirrored in the components of an agent decision making

algorithm. The resulting components must be implementable in an agent framework that will be used to generate NPC agents that operate in scalable game environments.

To properly simulate a scalable game environment, it would be intractable to use an actual full-fledged computer game to test the new agent framework for NPCs, due to the sheer number of variables that are involved a commercial computer game. It would also me completely impractical to use the standard game experiments for machine learning (e.g. Chess, Go, Backgammon, Blackjack, Checkers, Poker and others) as there are other more efficient methods such search, classification, and reinforcement learning that already perform admirably in these tests. Instead a proper experiment must be small enough, with the fewest number of variables to simulate a scalable game environment. But at the same time it must be sufficiently complex to to be able to observe an agent's adaptability in a highly dynamic environment. From this, a second research question is derived:

RQ2: "What is the smallest experiment that can be used to simulate a scalable game environment, and measure an agent's adaptability to dynamism?"

This research does not assume that an agent framework based on affect will be suitable for all possible eventualities in a scalable game environment. As mentioned in Section 1.2 simpler AI methods have proven to be preferable when the game is deterministic (Millington & Funge 2009; Yannakakis & Togelius. 2018). Therefore, a third research question is posed here:

RQ3: “In what types of scalable game environments would the use of emotional affect for decision making allow agents to make better decisions compared to using other decision making methods?”

This question focuses on the exact operational conditions which would necessitate the use of emotional affect for making decisions. By uncovering the specific operational conditions, the potential applications of the affective agent framework or the affective decision making algorithm, beyond entertainment and computer games, can be revealed.

1.5 RESEARCH OBJECTIVES

The aim of this research is to answer the research questions posed in Section 1.4. The specific objectives of this research are:

1. To propose an affective agent framework with components that emulate emotional affect for decision making in scalable game environments.

2. To develop a principal experiment that simulates the complexities of scalable game environments for benchmarking agent performance.

3. To establish the ideal operating environment where the affective agent framework has greater utility compared to other agent types.

1.6 METHODOLOGY FOR ACHIEVING THE RESEARCH OBJECTIVES

The first objective is undertaken through the development of an object-oriented agent class in the form of an application programming interface (API) for the generation of software agents. This research focuses on the implementation of the affective agent class API for use in computer games written in third-generation programming languages (3GL). However, the design of the affective agent framework in this research is expected to be applicable in other class-type programming structures available in any object-oriented programming language. Furthermore, the affective agent framework is intended to be potentially applicable to other problem-solving domains, beyond computer games, that meet the criteria of the ideal operating environment outlined for the third objective.

To achieve the first objective, the affective agent framework must be built around a core decision making algorithm that emulates the process of emotional affect during problem solving. The specific functional characteristics that must be exhibited by this core decision-making algorithm involves the undertaking of the following activities:

i. Design and develop a component in affective agent's core decision-making algorithm that allows the agent to 'feel' how the present game environment affects the agent in real-time. The component should accomplish this by discovering how features of objects in the game environment affect the agent's goals.

ii. Design and develop a component in the affective agent's core decision-making algorithm that allows the agent to 'emotionally override' its perception on what is important in real-time. The component should accomplish this by managing the priority of the agent's goals and altering its evaluation of the game environment based on goal priorities.

The end result of achieving these functional characteristics, and ultimately the first objective, will be the development of an algorithm that emulates the process of emotional affect in decision making. This research intends the affective decision-making algorithm to be used as part of the affective agent framework, and only explores the utility of the algorithm in relation to the implementation of the affective agent framework. Even so, the algorithm can be used independently from the affective agent framework and applied as an additional decision-making component or sub-procedure other software agent frameworks or even physical mechanisms like robots, to make motivated decisions in unfamiliar and untrained situations.

The allowance for the portability of the affective decision-making algorithm is provided since there will be situations in which simpler and less-complex decision making algorithms are sufficient and preferable. Therefore the affective decision making algorithm is designed to complement and work alongside other decision making algorithms in more advanced agent designs. In terms of implementation, the design of the affective decision-making algorithm must be simple enough that it can be integrated easily as a subcomponent for decision making in other game playing agents types. The problem domain in which the affective decision making algorithm will supersede other algorithms is explored in the third objective.

To achieve the second objective, modern computer games that expand in content cannot be used as they are too massive in content to be practical for observing an agent's adaptability to new dynamic situations. Therefore the only practical way to achieve the second objective is to build a new 'minimal' game that is small enough fo agent benchmarking and analysis. At the same time, the dynamic component of the game must also involve the fewest variables in order to properly attribute the ability of the agent to adapt to scalable game environments to the agent framework's underlying A.I. algorithm, rather than some coincidental external factor.

The third objective is concerning the identification of the type or class of problem where affective decision-making has the most use. To achieve this third objective, the operation of the affective decision-making algorithm under empirical testing is scrutinized. Specifically, the questions that must be answered by the third objective are as follows:

i. What types of problem would require the use of affective decision making?

ii. When should affective decision making not be used?

It must be noted, however, that it is not the objective of this research to produce new theories on how emotions can become rational or even theories on how emotions can influence rational behaviour. Neither is it to objective of this research to either prove or disprove particular psychological or cognitive theories on emotions and how it influences behaviour. Instead, both objectives of this research focus on the emulation of reported emotional theories in AI heuristics, such that an alternative to rational or utilitarian algorithms can be produced.

1.7 RESEARCH SCOPE

The scope of this research is to create an agent framework for NPCs based on affect and test the affective agent framework in a simulated scalable game environment. The performance of affective agent will be benchmarked against the

performance of agent frameworks that use the common A.I. algorithms shown in Table 1.1.

As specified in research objectives outlined in Section 1.5, the research will be primarily concerned with the development of an affective agent framework, and the specification of the exact problem domain in which the affective decision making algorithm will have greater utility compared to traditional game AI algorithms. An existing model of emotional affect from neuroscience, as it pertains to the role of affect in reasoning, will be employed as the primary guideline for the construction of the affective agent framework. Specifically, the emotional affect model adopted for this purpose is the 'As-If Body Loop Model' developed to support the 'Somatic Marker Hypothesis' theorised by Damasio (1994)(Lopez-Franco et al. 2018), and is described in further detail in Chapter 2 of this thesis. This model is chosen as it is outlines the specific roles of emotional affect from the perspective of the physiological function of emotions in affecting decisions. The physiological functions of affect described by the Somatic Marker Hypothesis can be readily mirrored and translated into the exact computational functions that need to be achieved in the affective agent framework. It is the emulation of these functions of affect in an agent, and the evaluation of its utility in decision making in games that is the focus of this research.

It is not the role of this research to debate whether the emotional affect model employed in the affective agent framework is the best among other existing models of emotions available in academic literature. In other words, this research will not undertake a comparison of how well the different models of emotional affect fare amongst each other in a competition of performance. Rather, this research hopes to answer how emotional affect can be emulated by AI agents, and in what types of problem should affective decision making be used by AI agents.

The scope for the first objective covers the following activities:

1. The identification of the specific function of emotional affect that can be instrumental for the formation of decisions.

2. The design of the agent framework components that duplicate the specified functions of emotional affect in decision-making.

3. The specification of the minimal prerequisite inputs that need to be provided by the agent's operating environment in order to allow the affective agent to function properly.

4. The construction of actual agent framework in programming code for use during empirical testing in the problem domain.

The scope of the second objective the covers the evaluation of the affective decision-making algorithm used by the agent framework in the problem domain. The problem domain which will be used as the primary test platform, for the execution of empirical experiments on agents using the affective agent framework, is 'Scalable Game Environments'. For this research, scalable game environments are defined as computer games which possess the properties summarised here:

1. Open game objects: The agent will have to adapt to new objects that enter the game with undefined behaviour.

2. Non-symbolic interaction: The agent only selects actions as a result of the presence of the game object. The agent does not communicate with the user or with the game objects.

3. Non-competitive performance comparison: There is no 'winner' in the game, only a benchmark comparison of end-game performance between tested agent frameworks

4. Incomplete information: The tree of all possible game states and game outcomes cannot be constructed *apriori*.

5. Continuous: The dataset used for benchmarking agent framework performance is presented gradually in real-time.

These properties are further elaborated in Chapter 2: Literature Review and Chapter 5: Principal Experiments on Affective Decision Making.

The scope of the third objective is to control the assessment of the specific role of affective decision making in games. Currently available commercial computer games will not be used due to their complexity in design. The complexity of current commercial games, whether in game rules, game play, or even programmatic integration of the affective agent's framework class in game's program code, will incur heightened difficulty for the purpose of assessing the affective decision-making algorithm's advantages and weaknesses. Toy games, such as software versions of traditional board games will also not be used for assessing the affective agent framework as they lack the properties of scalable game environments defined here. Instead a new scalable game will be purposely constructed for the purpose of testing and benchmarking between the agent frameworks. The specification of the scalable game built specifically for this research is provided in Chapter 5.

Finally, this research will benchmark the affective decision-making algorithm only in comparison with the most common game AI algorithms that have been popularly used for the construction of agents in computer games. The common game AI algorithms types that will be used for the purpose of benchmarking the affective decision-making algorithm are as follows:

1. Finite state machines.

2. Decision trees.

3. Reinforcement learning.

The justification for using these three agent frameworks for benchmarking and comparison is because they are the three most common types of agent frameworks that are used for creating NPCs for existing computer games, as listed in Table 1.1 (Millington & Funge 2009; Yannakakis & Togelius. 2018).

1.8 CHAPTER OUTLINE

This thesis is divided into seven chapters, which are outlined as follows:

Chapter 1: Introduction

This chapter provides an introduction on the use of artificial intelligence in games, and specifies the type of game, 'open interactive games' that will serve as the problem domain. Emotional and affective agents are introduced as a proposed solution which will allow for intelligent decision making in the problem domain. The research question is posed here. The research objectives, scope, and methodology for proving answering the research questions are outlined. A summary of the affective agent framework, and its core component, the affective decision making engine, is introduced.

Chapter 2: Literature Review

This chapter contains a broad survey of research literature on the role of emotional affect in problem solving. Of particular focus is the Somatic Marker Hypothesis, the model of emotional affect adopted in this research as a guideline for the construction of an affective agent framework. The specific contribution of emotion theory for the formulation of a decision-making algorithm that emulates emotional affect is outlined. A comparison is made of the types of emotional and affective agents currently explored in the field of artificial intelligence, as a means for classifying the affective agent framework that will be developed of this research. Past and existing similar work in affective computing that employs emotional affect models as AI in games is listed. The chapter concludes by highlighting the difference between this research and these other works.

Chapter 3: Research Methodology

This chapter reiterates the research question as the primary motivation of the study. It elaborates on all the major phases that have been taken during this research in order to provide a holistic coverage of the research question. The exact component activities performed for each phase of research are described, along with methods used to undertake each stage of activity. The associated results discovered for each activity, which has led to the final research direction, are presented.

Chapter 4: An Affective Agent Framework For Scalable Game Environments

This chapter describes the 'Affective Decision Making Engine' (ADME) in detail as the core algorithm that emulates the roles of emotional affect during decision making, for use by software agents. The logic and calculation of the two components of ADME, namely the Affect Matrix and the Affect Homeostat are provided in detail. The flow of information and details on data processing in ADME is demonstrated. Finally, the parameters used to fine-tune the intelligence of ADME are described.

Chapter 5: Principal Experiments on Affective Decision Making

This chapter specifies the necessary properties of the experiments that simulate the problem domain, i.e. open interactive environments. The game developed for this research, called 'Scalable Whack-A-Mole', is introduced and the parameters which control complexity in the game are provided and described. Design of existing agent frameworks that will be used for benchmarking are also presented here. The common datasets utilised for producing the level playing-field needed for benchmark testing of all agent frameworks in multi-stage experimentation are presented. The purpose of each stage of testing is also presented and justified.

Chapter 6: Analysis of Results for Establishing The Ideal Operating Environment For The Affective Agent Framework

This chapter shows the results of the multi-stage empirical experiments performed on the affective agent framework. The exact configuration of each stage of the multi stage experimentation is detailed. The performance of each agent framework benchmarked in the multi-stage experiments are compared, and the differences in affective agent's performance are highlighted. The raw data discovered in this chapter is presented to understand the conditions in which emotional affect will be useful. The chapter concludes by fulfilling the third research objective: i.e. establishing the ideal operating environment where affect will have greater utility during decision-making.

Chapter 7: Conclusion

This chapter repeats the research question, and highlights once again how the first, second and third research objectives have been fulfilled. This is done by highlighting the primary research contribution, i.e. the affective agent framework, and the details of the operating environment where affective agents will be needed. Possible future directions for other researchers who follow this work are presented. The thesis concludes with an insight on the future of affective computing mechanisms in real-world applications.

CHAPTER II

LITERATURE REVIEW

2.1 INTRODUCTION

This chapter begins by looking at the computer and video games industry and the current state of artificial intelligence in games, particularly in the development of believable and interactive virtual players for computer games, called 'Non-Player Characters' or NPCs. The challenges faced in developing decision making A.I for NPCs are highlighted, especially in getting NPCs to adapt to scalable game environments.

The core literature that provides the solution that is proposed by this research is explored next. In order to create A.I for NPCs that can adapt to scalable game environments, this chapter looks at research on how humans and other living organisms use emotional *affect* to manage, or cope, in dynamic and unfamiliar situations that are similar to scalable game environments. The theories on emotions covered in this treatise focus will focus more on ‘affect’: the feelings which motivate or bias decisions. The lessons learned from this investigation allows us to model a framework for agents that emulate the benefits of affect when making decisions in these situations.

Figure 2.1 summarizes the the most important topics that will be covered in this literature review. The prevailing theories on the influence of emotions-on decision making will be outlined first. Next, a coverage of existing applications of emotion theory in artificially intelligent software agents will be provided. Finally, some of the issues that confront any application of emotion theory as a means for

rational decision-making will be discussed. The shaded area in Figure 2.1 indicates the topics that are most relevant, and are directly used, in the proposal of an affective agent framework for decision-making in games.

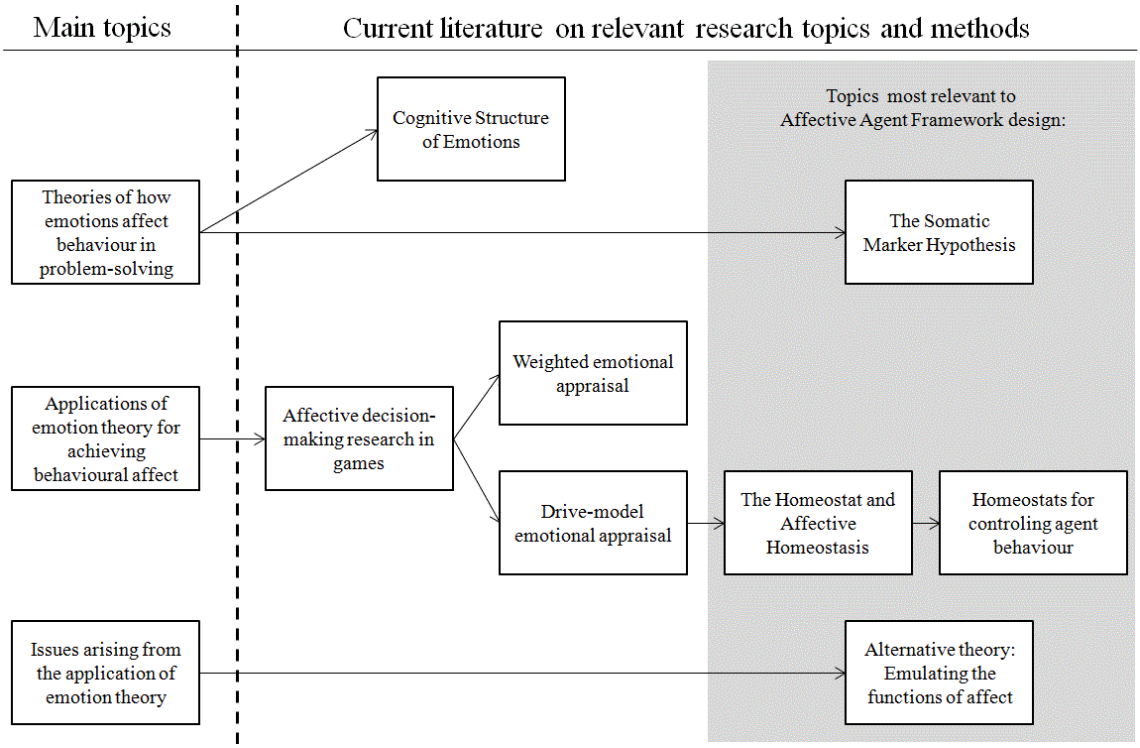


Figure 2.1 Literature Review Topics

The final part of this literature review will discuss the research gap by summarizing the functional requirements needed to emulate the affective decision making process in software agents. Existing methods and technologies that can be used to fill this research gap by fulfilling each functional requirement are presented.

2.2 THE COMPUTER AND VIDEO GAMES INDUSTRY

The computer and video games, once thought of as electronic toys to distract children and teenagers, have become a global economic force by the sheer volume of revenue it creates. A report by DFS Intelligence estimates sales of computer and video game hardware and software from January to June 2011 generated a gross revenue of USD $65 billion (RM 207.18 billion) globally (Baker 2011). Based on current purchasing trends, Gartner Intelligence projects that this figure will increase to

USD $74 billion (RM 235.87 billion) globally by the end of 2011 (Bilton 2011; Ell 2018). The same report predicts that the momentum is expected to continue, with annual earnings estimates of USD $112 billion (RM 357 billion) globally for the year 2015 (Gaudiosi 2011). In terms of employment, the Entertainment Software Association, or ESA, states in 2011 that the computer and video games industry provides around 120,000 jobs for people in the U.S.A alone, with an average annual salary of USD $90,000 (RM286,875) per employee (Entertainment Software Association 2011). The wealth generated by computer and video games justifies the development of methods for making games more engaging, as the appeal of computer games lies in the level of immersion they provide. To that end, serious research in artificial intelligence is increasingly being integrated into computer games to make them more entertaining. Computer games are no longer just a distraction, but are now a platform for good science.

This dissertation is about how to make computer games more entertaining: by giving ‘feelings’ to the virtual players we play against, to make them more intelligent and challenging in the game. The approach taken is not to give these virtual players expressive ‘emotions’ like ‘happiness’, ‘anger’ or ‘sadness’. Instead, the aim is to provide them with a means to be affected by their environment, so that they may understand how to live in a world they have never seen.

2.3 ARTIFICIAL INTELLIGENCE IN GAMES

Computer and video games are software designed to provide entertainment through the simulation of events and automation of interactive game objects or players, thereby allowing people to experience roles far different from their daily lives. Many computer games entertain players, i.e. people who interact with the game, by providing scenarios in which to perform various tasks, such as solving puzzles, performing a sequence of actions, or by reaching goals. Players become entertained either by the completion of these set tasks, or by the challenges and obstacles presented by the game that prevents players from completing these tasks. Players can also be immersed in the game the closer the game simulates reality, either from the visual realism displayed by the game graphics, or by behavioural realism exhibited by

the game objects. Behavioural realism is provided through the inclusion of artificial intelligence algorithms, or AI, as part of the game objects control code. The following is a non-exhaustive list of the game genres where AI is used:

1. *Shooters*: Includes SHMUPS (*shoot-em-up* games), FPS (first-person shooters), and other shooting type games. AI is used here to simulate game opponent movement, targeting, and firing.

2. *Sports games*: Team-based AI is used to coordinate computer player strategy and positioning as a multi-agent system. AI is also used to calculate the trajectory of game objects, such as a football.

3. *Board games*: Strategic AI is used here to search through game-state trees and find the optimum sequence of moves that maximizes game outcome.

4. *Role-playing games (RPG)*: The realism of RPG game environments, such as gravity and particle collision, is controlled by AI. The game characters that can interact with the player, or challenge the player, are controlled by behavioural and decision-making AI. Player interaction with game characters may also include conversations, scripted with chatterbot-type AI.

5. *Online games*: Includes massively-multiplayer online role-playing games (MMORPG), which are extremely large scale versions of RPG games. As with RPGs, online games employ AI for environmental control, behavioural control and interaction, but to a much larger degree. Multiple human players simultaneously interact with game objects, therefore AI is also used as part of the game infrastructure, for load balancing and process optimization.

2.3.1 Game AI research directions: Realism vs. Intelligence

There are two general directions taken in the application of AI in computer games, which are:

1. **Realism** - This direction involves using AI inside game elements, both the game environment and game objects, to enhance the illusion of reality in the game as a whole, by mirroring actual observable real-world phenomena.

2. **Intelligence** - This direction involves using AI inside game objects in particular, in order to control their actions and provide either the guise and semblance of cognition, or some mimicry of actual rational thought.

For each direction stated previously, there are multiple specific roles that can be taken by AI in games. Laird and van Lent (2001)(Petrović 2018) provides an partial list of game types, the AI roles in each type of game, AI problem domains in gaming, and the specific research areas in AI that is of commercial interest to computer game developers. This list can be seen in Table 2.1.

Table 2.1 Game genres and AI roles, problems and research

Game Genres	AI Entity Roles	AI Research Problems	AI Research Areas
Action	Tactical enemies	Interact with environment	High-level perception
Role playing	Partners	Fast response	Commonsense reasoning
Adventure	Support characters	Realistic sensing	Natural language
Strategy games	Story directors	Adapt to environment	Speech processing
God games	Strategic opponents	Interact with humans	Gesture processing
Team sports	Units	Adapt to human player	Planning & counterplanning
Individual sports	Commentators	Difficulty adaptation	Cognitive modeling
		Strategic adaptation	Plan recognition
		Interact with other AIs	Soft real-time response
		Coordinate behavior	Reactive behavior
		Navigation	Teamwork
		Use tactics and strategies	Scheduling
		Allocate resources	Path planning
		Understand game flow	Spatial reasoning
		Humanlike responses	Temporal reasoning
		Reaction times	Opponent modeling
		Realistic movement	Learning
		Emotions	Knowledge acquisition
		Personalities	
		Low computational overhead	
		Low development overhead	

Source: Laird & van Lent 2001

Millington and Funge (2009)(Yannakakis & Togelius. 2018) further classifies the most common AI roles into five distinct models, which are Movement, Decision-Making, Strategy, Infrastructure, and Agent AI. Table 2.2 consolidates Laird and van Lent's (2001)(Petrović 2018) list of AI research areas in games, against Millington and Funge's (2009)(Yannakakis & Togelius. 2018) classification of AI roles.

Table 2.2 Common roles of AI in games and associated research

Game AI Direction	Game AI Roles	AI Research Areas
Realism	Infrastructure	Physics and particle control Perception control Collision detection AI programming languages
	Communication	Dialog control Automated story direction
Intelligence	Movement	Path finding Goal directed behaviour
	Strategy	Tree search State evaluation
	Decision Making	Behavioural state transition Adaptation Reinforcement learning Classification

Sources: Laird & van Lent 2001; Millington & Funge 2009.

The roles taken by AI in each research direction is elaborated here. In the game realism research direction, the AI roles are typically as follows:

1. **Infrastructure** – AI is used here to control environmental dynamics to provide the illusion of real-world reaction, which is done through various programmed techniques. For example, physics and gravity in games can be simulated using physics subsystems or engines, such as Ageia and NVIDIA's 'PhysX Engine' (PhysX System Software 2011) and Havok's 'Havok Physics' engine (Havok Physics SDK 2011). In another example, game information provided to the player can be limited through perception control mechanisms that emulate line-of-sight (Terzopoulos et al. 1994; Elmalech et al. 2016). Collision detection algorithms have been integrated with rule-based algorithms to trigger environmental reaction to game events (Lin & Gottschalk 1998; Kim et al. 2018), for example: controlling crowd dispersal when a game car is about to crash into a game wall, and simultaneously sequences the necessary

animation during this interaction. AI research is also used in the creation of specialized rule-based languages for AI control in games, such as RC++ (Wright, I. & Marshall. J. 2000).

2. **Communication** – Refers to AI that is used to control dialog choices between game characters and the game story board. Weizenbaum's (1976; Ferrara et al. 2016) ELIZA, a chatterbot programmed with pre-scripted responses mimicking a psychotherapist, was historically one of the earliest examples of AI simulating direct conversation with users. Story directors, such as Young's (2001) MIMESIS architecture, instead use AI to plan the narrative of the game's story.

In the game intelligence, the roles taken by AI are classified as follows:

1. **Movement** – AI is used in this role for path finding and path planning. Classical AI search methods are most commonly used here. A* search (Hart et al. 1968; Bast et al. 2016) is perhaps the most popular algorithm for path finding ever used in games. Integration of movement plans and action plans requires more than just general search, and is achieved using goal-oriented planning tools such as the GOAP system (Orkin 2004).

2. **Strategy** – Refers to AI that lays out a plan for a series or sequence of actions that is expected to reach highly preferable game outcomes. Monte-Carlo tree search (Chaslot et al. 2008) has found its place as a strategy search mechanism for deterministic board games with massive search trees such as Go. More complex AI such as Dynamic procedural tactics (Straatman et al. 2005) have been used in games such as Killzone (2004) to tactically pick positions.

3. **Decision-Making** – AI for classification, adaptation and learning have been found to be of great use for this role in games. Finite state machines (FSM), governed by pre-scripted rule bases, have been extremely popular due to its simplicity. Games such as F.E.A.R. (Orkin 2006) use FSM due to the ease in which state transition mechanisms such as these can be used to alternate

between different sets of behaviours. Other forms of FSM, such as Abstract State Machines (Schwab, 2008) integrate FSM with decision-tree mechanisms to allow for iterative search of best decisions. Reinforcement learning mechanisms such as Q-learning (Watkins 1989; Zaremba et al. 2016) excel at adapting the correct action to the game input in real-time games without the need for training. TD-Learning (Sutton 1988; Hertz et al. 2018) is another adaptive learning mechanism that is able to adapt the ideal weights to position evaluation functions for use in real-time stochastic games such as Backgammon (Tesauro 1995; Sutton & Barto 2018).

2.3.2 Agents: The embodiment of autonomous AI in games

The AI game intelligence methods listed here are most commonly integrated as algorithmic components that control the actions and decisions of game objects in the form of software agents (van Lent et al. 1999; Gaudl 2016). In computer games, agents are usually object-oriented program code classes that behave autonomously inside the game environment. Passive agents, i.e. agents that do not display active movement selection, strategic planning, decision making, or any form of interactivity, generally become part of the game's smart-background. Passive agents are largely equipped with algorithms that play the infrastructural role of AI in games. Active agents, i.e. agents that do endeavour to make decisions, strategize, plan and interact, usually play some or all of the AI intelligence roles in games listed in Section 2.3.1, including the conversational role.

The specific types of agent which human players can see as 'people', 'animals', or 'opponents' in games, are called 'Non-Player Characters' (NPC) (Dignum et al. 2009). NPC are intelligent agents that use AI for game character cognition. In simple terms, NPC are computer controlled characters. They display the four necessary characteristics of intelligent software agents (Wooldridge 2002), which are:

1. **Persistence**: The agent is always in the program memory, waiting to execute behaviour under pre-programmed conditions.

2. **Autonomy**: The agent does not require direct program invocation or user input to execute behaviour, but instead self-determines when behaviour should be executed.

3. **Social ability**: The agent can communicate and collaborate with program objects or even with other agents

4. **Reactivity**: When the correct conditions are observed, the agent executes behaviour that it determines to be appropriate for that condition.

Table 2.3 Most frequently used decision-making algorithms for A.I game agents.

Type	Examples	General Technique	Requirements	Issues
Finite State Machines (FSM) Algorithms	• Moore machines (Moore 1956, Staddon 2016) • Mealy machines (Mealy 1955, Vaandrager 2017)	Evaluate the problem state condition before transitioning to a new behavioural state.	• Fully observable states (i.e. complete information) • Determinism	Knowledge rules need to be fully manually determined and customized for the game before agent can operate.
Decision Tree Algorithms	• ID3 (Quinlan 1986, Wu et al. 2016) • C4.5 (Quinlan 1993, Witten et al. 2016)	Simulate all possible move combinations up to *n*-ply and evaluate outcomes of leaf nodes.	• Fully observable states (i.e. complete information) • Determinism	Tree can branch exponentially if the possibilities that occur for each ply increases.
Search Algorithms	• A* (Hart et al. 1968, Bast et al. 2016) • Minimax (von Neumann 1928, Aumann 2017)	Simulate path movement and propagate reward or error to paths that reach closer to goals.	• Fully observable states (i.e. complete information) • Determinism	Stagnation (e.g. going back and forth) may occur if changes along paths alternately propagate conflicting rewards.
Adaptation Algorithms	• Q-learning (Watkins 1989, Zaremba et al. 2016) • TD Learning (Sutton 1988, Hertz et al. 2018)	Apply reward or penalties to action choices, according to the degree of goal fulfilment.	• Fixed choices and elements (e.g. the same actions or objects are always available)	Preference for heavily reinforced actions can be difficult to change in a short period.

Artificially intelligent agent frameworks used for current computer games mostly assumes that the game environment is fully observable and deterministic. The algorithms used are tailored to cater for every eventuality within the game

environment. Table 2.3 summarizes the major types of decision-making algorithms used by game agents in current computer games.

The most common decision-making algorithms used to control agent behaviour in commercial computer games today are as follows (Sweetser & Wiles 2002):

1. **Finite state machines**: Finite state machines (FSM), such as Moore machines (Moore 1956; Staddon 2016) and Mealy machines (Mealy 1955; Vaandrager 2017) uses state transition methods to determine the agent's current state in a game. Each state is manually associated to a particular output or action that can be taken by the agent. The state transition diagrams or truth tables used to determine the state and appropriate action must be pre-optimized for the game, prior to game play, for FSM to work properly (Holzmann 1991; Cassel et al. 2016).

2. **Decision trees**: Decision tree algorithms make use of the classification results from decision tree learning methods such as ID3 (Quinlan 1986; Wu et al. 2016) or C4.5 (Quinlan 1993; Witten et al. 2016) to determine the current state of the game. Similar to finite state machines, each classified state or node of the decision tree is mapped to the appropriate action. For the same reason, the decision trees must be trained on and optimised for the game, prior to game play (Murthy 1998; Suthaharan 2016).

3. **Search**: Search algorithms, such as minimax (von Neumann 1928; Aumann 2017) or A-star (A*) search (Hart et al. 1968; Bast et al. 2016), projects all possible future states for each action, either until the game's conclusion (i.e. endgame) or up to a limited number of game rounds, or 'ply'. The outcomes of all end nodes at the endgame or final-ply is compared, and the action mapped to the path leading to the most desirable outcome is chosen. In order to assign the proper value to end nodes, search algorithms require that the game be fully observable and deterministic. This is so that the sequence of

future actions can be planned and the outcome of each future action step can be predicted (Russell & Norvig 2015).

4. **Adaptation**: Adaptive algorithms use reinforcement learning methods, such as Q-learning (Watkins 1989; Zaremba et al. 2016) or temporal difference (TD) learning (Sutton 1988; Hertz et al. 2018), to compare the change in outcome received after trying different actions. Actions with higher outcomes are rewarded (i.e. reinforced), and the action with the highest reward is chosen in subsequent iterations in the game (Tesauro 1995; Sutton & Barto 2018). Adaptive algorithms require the game to be deterministic. The current actions reinforced through the algorithm must be available in future iterations for agent to benefit from adaptation (Yen & Hickey 2004).

This research focuses on the role of AI for decision-making in games as the primary area of study. Agents such as NPCs need not be complex and are usually not programmed with the full range of AI roles in order to play games. Seemingly complex NPCs like the AI agents in the computer game 'F.E.A.R.' only uses the simplest form of FSM, which has just three states for behavioural selection, and an A* algorithm for path finding (Orkin 2006). Yet to the human user playing the game, the illusion of intelligence is near perfect, evident through the range of pre-scripted behaviour for performing long sequences of actions. For the purpose of providing the user with interactive entertainment, simple algorithms such as these are sufficient (Butcher & Griesemer 2002). Furthermore, specialized and hardcoded AI solutions to problem solving in games are easier to develop and require less time to modify due to their simplicity. The lack of the need for NPCs to be scalable beyond the roles that they play in close-ended computer game negated the need for using more advanced AI techniques (Millington & Funge 2009; Yannakakis & Togelius. 2018). This algorithmic simplicity is why FSM and adaptive reinforcement learning algorithms, such as Q-learning, remains the preferred decision-making component in agent frameworks for NPC development in games.

It has been established in this section that very simple AI algorithms are sufficient to provide a convincing illusion of intelligent behaviour for adding

entertainment value in games – with the caveat that the game does not increase in size or complexity. Since the widespread acceptance of online games and online delivery of game content, however, there is an increasing trend to make games expandable beyond their original retail release format, in order to increase the commercial value of computer games (Toivonen & Sotamaa 2010). For example, an MMORPG such as World of Warcraft (2004) has increased its game environment three times since its original release date, implemented through game expansion packs, with a fourth game expansion currently planned (Blizzard Entertainment 2011). Even non-online games such as Fallout 3 (2008), has expanded five times through the distribution of downloadable content packs (DLC) that can be purchased from its in-game application distribution store.

The growth of game environments is now the emerging trend. The subscription model, where players pay to keep playing the same game, has proven to be a more lucrative business model for computer game developers, since the process of core game engine development only needs to occur once (Stenros & Sotamaa 2009). Game developers have found that it is easier to make money from games that already have an established fan base. To preserve subscriptions, new game content that works with the existing game engines must be built.

2.4 SCALABLE GAME ENVIRONMENTS

This research coins the term “Scalable Game Environments” , the problem domain for this research outlined in Chapter 1, to refer to the any type of game where the task environment or operational environment, in which players will play, can be expanded. The term differs from “scalable game architecture” (Cai et al. 2002) which actually refers to the scalability, distribution, and balancing of game processing load throughout a game’s server and network infrastructure (Glinka et al. 2007). Instead, scalable game environments refer to the ‘playing field’ or ‘arena’ of the game itself which can grow in size, complexity, and population. This growth is commonly a result of additions or modifications to the game program, implemented through software updates, software expansion packs, or downloadable content. For example:

1. A software patch to a game may change the value of the attributes, or ‘features’, of current NPC agents.

2. A game expansion pack may introduce a new playing area and new classes of opponent NPC agents which have different features and abilities.

3. A downloadable content pack may introduce a new puzzle and provide new rules for playing the puzzle.

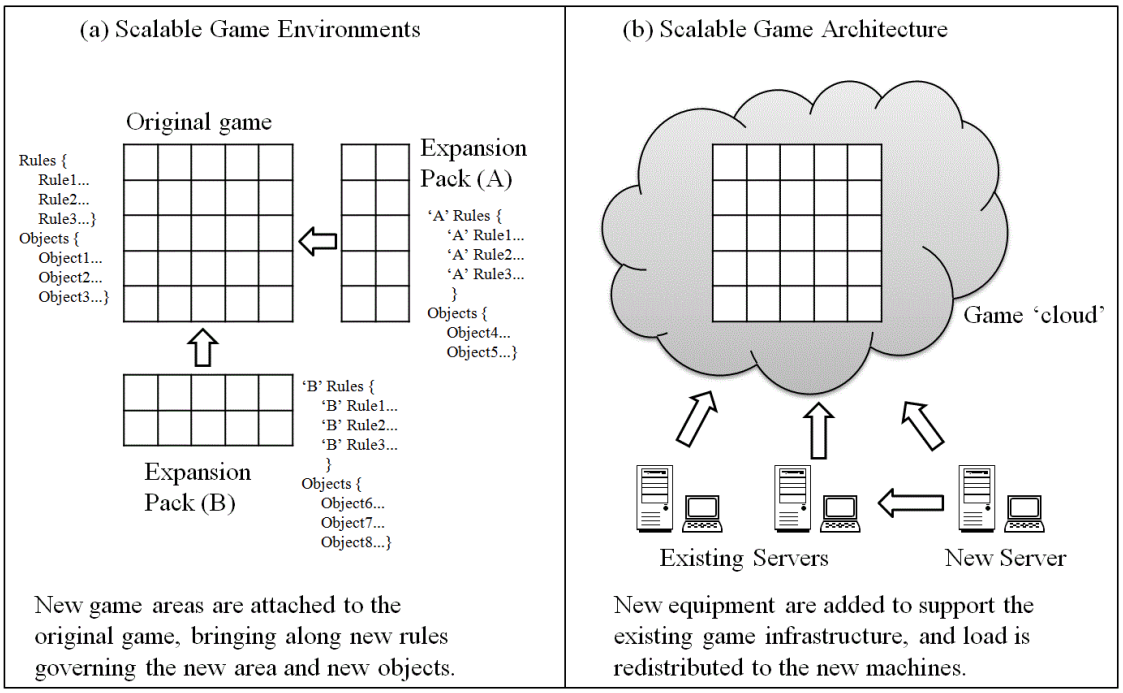
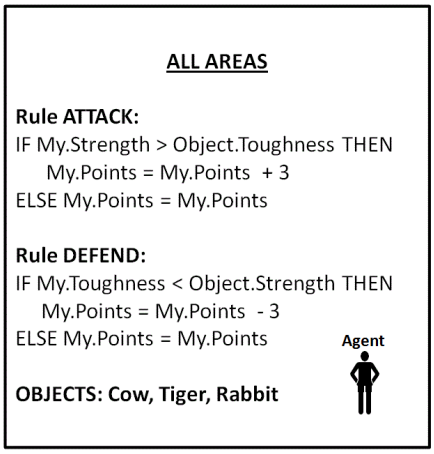


Figure 2.2 Scalable Game Environments and Scalable Game Architectures.

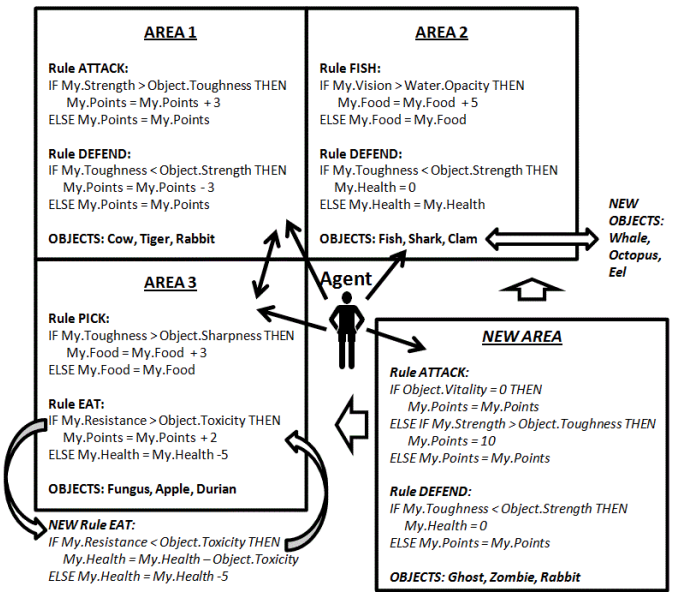
Figure 2.2 highlights the difference between scalable game environments and scalable game architectures to avoid confusion between the two. In Figure 2.2(a), whenever an existing game can be expanded by adding new playing areas, new game rules, or new game objects, then this refers to a “Scalable Game Environment”. In Figure 2.2(b), when new servers or network nodes can be added to the existing infrastructure supporting the game, then this refers to “Scalable Game Architectures”. This research only focuses on scalable game environments, and uses a custom built ‘toy’ game simulating a minimal scalable game environment to test the affective agent framework. . Figure 2.3 illustrates scalable game environments and how it differs from current static game environments.

(a) Static Game Environment



(a) Static game environments have uniform rules and permanent object types for all areas in the game. The agent can use a fixed algorithm, such as FSM, to manually program behaviour for every possible encounter with object that exists in the game. This is currently the preferred method for creating artificially intelligent agent for games (Millington & Funge 2009).

(b) Scalable Game Environment



(b) Scalable game environments have different rules and objects for different areas. Furthermore the environment is scalable, in that new game expansions can add new areas with new rules, update existing rules and replace objects. Manual programming of behaviour to cater for unknown future areas and rules is infeasible.

Figure 2.3 Static game environments vs. Scalable game environments.

Scalable game environments shares some characteristics of real-world environments that make online or real-time classification and search of optimal actions difficult. The general characteristics that are shared are as follows:

1. **The environment is partially observable –** There is a limit to the objects can be perceived by agents operating in a real-world environment. This limit is either imposed by the agent's own perceptual range, or is obscured by obstacles, or simply because the object has not come in to view in the environment. In the same way, scalable game environments are naturally partially observable because it is impossible to know beforehand what objects will exist for a game extension that has not been created. These limits mean that, although it is conceivable to train an agent to classify optimal actions (i.e. output) for all current inputs, retraining for reclassification is necessary for all new input observed in the future, which is intractable. Adaptation is the only feasible method for online learning in this case.

2. **Outcomes of interactions are stochastic** – In a real-world environment, the rules governing cause and effect can only be conclusively known through deep inspection of the physical laws controlling the interaction. In the same way, the outcomes of interactions in scalable game environments are also stochastic, because the rules governing game interactions are not known to agents operating in the environment. As a result, decision-making algorithms for game NPC agents, which have been optimally configured for a set of rules in one game environment, will not behave optimal in another game environment because the rules may be different. The agent will have to determine what rules are currently enforced before it is able to make intelligible decisions.

2.4.1 Complexity from Uncertainty in Scalable Game Environments

In intelligent agent research, Russell and Norvig (2015) have identified that uncertainty will occur when the agent's environment is as follows:

1. **Partially observable**: The agent either has limited perception or cannot feasibly observe all changes in the state of its environment. Uncertainty in this case is a result of the failure of the agent to account for all input/output changes, resulting in missing information required for analytical modelling.

2. **Stochastic**: The agent's environment is non-deterministic, i.e. the agent does not have full control over the outcomes of its actions. For instance, the final outcome may depend not only on how the environment reacts to the agent's action, but also on the actions of a second agent within the same environment. The existence of multiple factors, in this case separate and non-linear evaluation functions, brings about uncertainty when determining outcomes.

Therefore, in terms of intelligent agent operation, the problem domain for this research covers any agent interaction in an environment that is both partially observable and stochastic. The purpose of this research is now to reduce agent decision-making problems due to uncertainty within such environments. One type of

environment, which has been chosen as the experimental scope of this research, is computer games.

This research finds that computer games are a useful experimental platform to simulate the effects of uncertainty to the agent, when the agent's operating environment is both partially observable and stochastic. First and foremost, computer games are a platform where the exact evaluation function used to determine outcomes can be known conclusively. The evaluation function is known by the game designer, yet at the same time is unknown to the agent. For this reason, any knowledge model produced by the agent can be validated for accuracy by comparing it against an outcome graph produced by the actual evaluation function. Secondly, uncertainty can be simulated by controlling the amount of information that can be observed by the agent, and the degree in which agent interactions are deterministic or stochastic. This is accomplished by periodically expanding the game environment to either include new game areas, or to include new game objects. This is the reason why research coins the term "Scalable Game Environments" to refer to such games

This research argues that each of the common decision-making algorithms used in current computer games, as listed in Table 2.3, would suffer the effects of uncertainty if the game environment were scalable. The effects of uncertainty are described for each type of algorithm:

a. Finite state machines and decision trees require that the game environment be deterministic and fully observable, so optimal state-transition diagrams or decision trees can be constructed to suit the game. The argument here is that if new game areas or objects were introduced in the game, the decisions made using these state-transition diagrams or decision trees would no longer be optimal as it is not optimized for the new input. The effect of uncertainty caused by the new input would be the introduction of a new state and a new set of non-weighted state-transition paths. This new set of state-transition paths may potentially outweigh all existing learned state-transitions, effectively causing to become the preferred state-transition path over all others. If new

states continuously come into the finite state machine, it would cease to be finite, and may never properly weigh any new state-transition path.

b. Search techniques also require the game environment to be deterministic, so that a complete path for reaching the best expected outcomes can be projected. Yet, if the game were to constantly scale and expand to include new areas and objects, such search paths will need to be recomputed. This may cause the path taken to be less optimal in the long run. The effect of uncertainty to search algorithms therefore would be to cause the algorithm to explore and test all possible new paths. If new paths continuously become available, then the search algorithm would never stop exploring.

c. Adaptive techniques can be expected to work well in scalable game environments. It can operate in partially observable and stochastic environments, since it can reinforce decisions against any available action or object it encounters. However, because the reinforcement is made against past actions or objects, it can be argued that the efficiency of adaptive techniques will degrade when the game changes if it constantly discards past actions or objects, or replaces them with new ones. This will force adaptive mechanisms to readapt to the changes. The effect of uncertainty to adaptive techniques is that as new objects become available, it will spend more time iteratively reinforcing the new objects or actions, rather than exploiting its knowledge of past action choices. Furthermore, if new objects or actions are continuously introduced in the game, the adaptive algorithm may not experience the new object long enough to iterative reinforcement to adequately occur.

It should be pointed out that the computer game development community does not see these shortcomings as a problem (Butcher & Griesemer 2002; Straatman et al. 2005; Orkin 2006). This is because the focus of commercial game development is entertainment, rather than optimal efficiency of agent operation (Miikkulainen et al. 2006). The current solution, preferred by almost all game developers, is to simply create different agent A.I. algorithms for each different eventuality or environmental change that may be encountered when the game expands (Millington & Funge 2009;

Yannakakis & Togelius. 2018). For this reason, computer games are only used as an experimental platform, to test new agent frameworks in this research.

2.4.2 Complexity in Scaling Game AI

To cater for the growth in computer game environments, programming elements such as the game AI needs to be scalable beyond their initial design. Herein lies the inherent flaw with the initial preference for using simple AI algorithms in agent frameworks for generating NPCs. FSM, for example is a type of specialized AI, in that the state transition nodes of agents using FSM are tailored specifically for the immediate game environment. The same is true for NPCs developed using rule-based agent frameworks. Figure 2.4 provides a minimal scenario where an agent using FSM would fall back to a default action, when the rules for the specific encounter with game objects are absent.

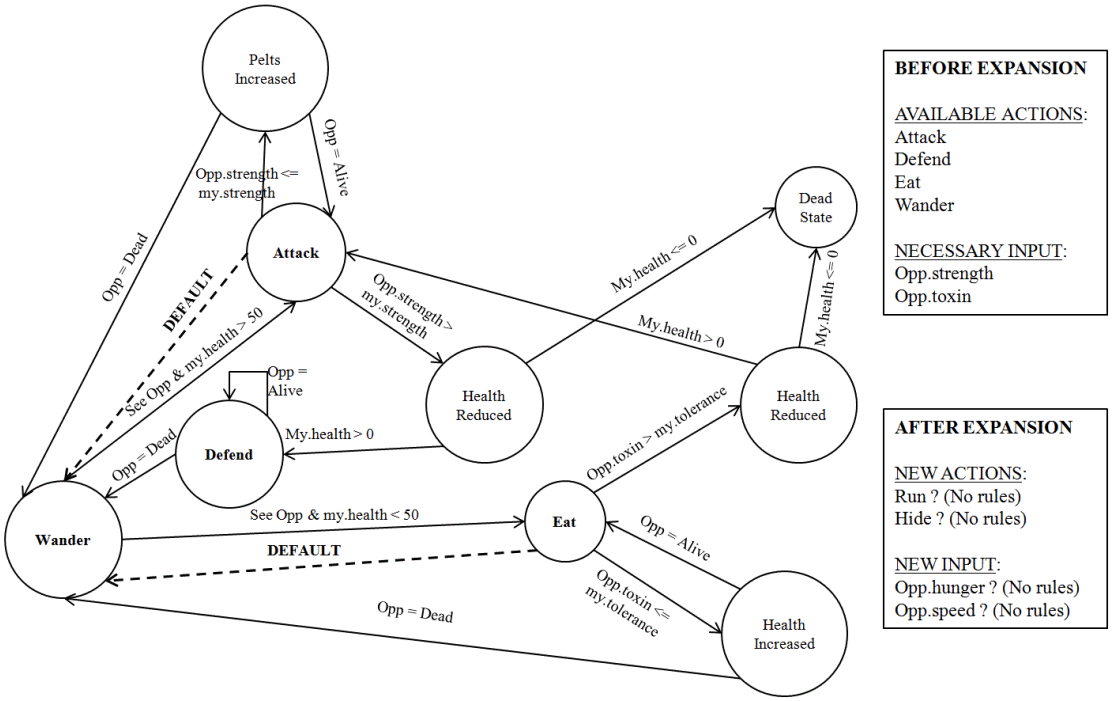


Figure 2.4 FSM-based agent responses to game environment encounters.

In Figure 2.4, the agent can find the correct action to take with the FSM, when the object being encountered ("Opp" for 'Opponent') has the necessary properties, or features types ('Opp.strength' and 'Opp.toxin') for transition between the many agent states in the FSM. In this case, the rules for subsequent actions are provided to the

agent. The rules are hardcoded to cover every eventuality in a deterministic game environment. However, after a software expansion, the game is no longer deterministic, but is now stochastic. New actions ('Run' and 'Hide') may become available to the agent, which are not represented in the agent's current FSM. Furthermore, the object faced in the game may possess features that are not represented as transitions in the FSM either. This means that after the software expansion, the agent will only remain aware of its old actions and the old features that it needs to look out for. If these features are not available, the agent would instead fall back to a default action, when the FSM is not provided with the input it needs.

Since the agent framework that the NPC is based on cannot learn new rules without the game programmer's intervention, new states and state-transition rules will have to be added to the FSM (Millington & Funge 2009; Yannakakis & Togelius. 2018). For every new area of the expanded game environment, the game developer has to take either of the following approaches when programming NPC agents:

1. Develop new NPCs with new state-transition nodes or rules for the new game environment

2. Modify existing NPC by integrating new state-transition nodes or rules to existing nodes and rules, to allow it to work in new game areas.

The former approach is the easier of the two, since it would not alter the tried and tested behaviour of the NPCs in old areas. The illusion of continuity of the NPC, in different areas of the game, is preserved by regenerating the NPC object container into a new agent framework (Sweetser & Wiles 2002). In other words, replacing the AI code, while retaining the old NPC 'face' for each new area would give the illusion of a continuous yet behaviourally complex agent. Figure 2.5 demonstrates how the agent framework for the same NPC can be changed in different game areas.

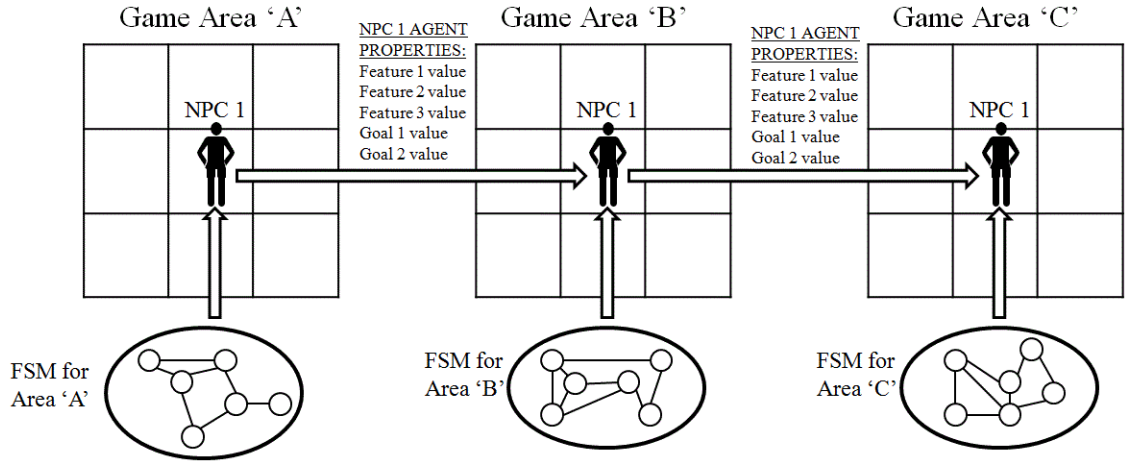


Figure 2.5 An NPC travelling from one game area to another is reinitialized using a new agent framework, while preserving the NPC's properties.

The NPCs alteration from using one agent framework to another is necessary in a commercial game, due the differences inherent within the agent's operational environment in each area. Russell and Norvig (2015) provides a list of six environmental properties that can differ between game environments:

1. **Fully observable vs. partially observable**: Either some or all game objects can be perceived by NPCs.

2. **Deterministic vs. stochastic**: Future NPC state can either be completely controlled by NPC, or is subject to an element of chance.

3. **Episodic vs. sequential**: NPCs decisions either have a momentary or persistent impact on future outcomes.

4. **Static vs. dynamic**: The game environment may change independently of NPC actions.

5. **Discrete vs. continuous**: The game environment may change due to the passage of time

6. **Single vs. multi-agent**: Other agents may exist in the game to interact with NPC or to alter the game environment.

Due to the differences listed above, game developers find it easier use different agent frameworks for different areas of the game, rather than program the NPC based on a single agent framework for all game areas. On the other hand, this creates a problem in that for every new game expansion released, the process for creating new agent frameworks will have to be repeated, thereby adding significant development cost to the game.

For the moment, adaptive machine learning methods, such as reinforcement learning, are employed to allow scalability of NPC agent behaviour in games. Dynamic scripting (Spronck 2006) bridges rule-based AI with reinforcement learning mechanisms, by using a weight update function to determine what the best rule-base to use for a particular game environment is. Stop-gap measures adaptive methods such as these are however still tied closely to particular game environments and require extensive pre-scripting of the potential rule-bases. A different adaptive machine learning method, call Q-learning, takes a more general approach (Watkins 1989; Zaremba et al. 2016). For any game, the Q-learning algorithm will initially explore all possible actions to find the action that provides the best delayed reward for the provided game stimuli. This action will be recalled whenever the same game input stimulus is experienced.

2.4.3 General Game Playing (GGP) for Scalable AI Agents

Since 2005 that has been a movement to create AI agents that can be used to play any game, without the need to tailor or optimize the underlying agent framework to the game being played. The challenge presented was to build a framework for AI agents that can play many different, undisclosed games, using only a single agent framework. These types of agent are called General Game Players (GGP) (Genesereth & Love 2005; Perez-Liebana et al. 2016). General game players are designed to play games that they have never played, by only being provided the rules of the game, minimally abstracted using the Game Description Language (GDL) (Love et al. 2008), which includes an abstraction of the current game state and a list of actions that the agent can take. The GGP agents are able to use this abstract

description and determine the next move to take. This determination is done after having processed the GDL description using first-order logic, fuzzy logic, and theorem proving to find potential best actions. Machine learning methods such as reinforcement learning have also been implemented as GGP agents (Banerjee & Stone 2007). Fluxplayer (Schiffel & Thielscher 2007) for example, won the 2006 GGP Competition, organized by the Association for the Advancement of Artificial Intelligence (AAAI), after having played through 70 different matches of different games. Fluxplayer's success is due to the automated generation of heuristic functions for games defined in GDL.

This class of agents, i.e. agents that exhibit human-level AI and flexibility, is exactly what is needed in order to have game AI agent framework that can scale automatically to match any expansion of the game environment (Laird & van Lent 2001; Petrović 2018). Unfortunately, at this time, current GGP is mainly suited to play board-type games. This limitation is due to the necessity of abstracting the game rules and complete game states to GDL prior to playing the game. The dynamic, sequential, and continuous properties of modern computer games are often prohibitively difficult to properly abstract in GDL, although efforts to make the language more flexible have been made (Thielscher 2010).

2.4.4 What is needed: AI Agent Frameworks That Can Operate in Scalable Game Environments

In order to allow agents to operate in game environments that scale (i.e. expand beyond its initial parameters) without having to recode the game AI to cater for the new environment, the agent framework used to generate NPCs must allow the NPC to be able to scale up in intelligence as well. Similar to GGP agents, the NPC agent must learn to play the game by itself, after having been provided with minimal information such as the game state. However, unlike the fully-observable deterministic board games played by GGP agents, the game rules may not be provided. This research declares that scalable game environments naturally exhibit the following game properties:

1. Partially observable: The complete range of game objects cannot possibly be known because game developers are free to implement any game object class for future game expansions.

2. Stochastic: The rules governing game outcomes cannot possibly be known because game developers are free to tailor new game rules for future game expansions.

Since new game areas may bring with it new rules, which are not known to existing game NPCs, the agent framework must be able to supply the NPC with insight on how to react within these new game areas, or how to behave in encounters with new game objects. In order to test the effectiveness of decisions made by such scalable AI agent frameworks, and in order to control the analysis of the agent's performance in a scalable game environment, this research has developed a toy game that simulates a minimal scalable game environment as its problem domain. Chapter 5 describes this problem domain test platform in greater detail. The types of changes and expansions that will occur in this miniature scalable game environment are:

1. New game objects will be introduced randomly in the game.

2. The new game objects may or may not possess new attributes or features types.

3. New game rules, unknown to the agent, will govern the evaluation of game encounter outcomes with new game objects.

2.5 INSIGHT TO A SOLUTION: INTERNALIZING VALUES

The roots of this research began with the examination of curious problem sets in Game Theory, where irrational sub-optimal behaviours lead to superior outcomes compared to rational utility driven behaviours. In Game Theory, any interaction involving decisions made between two or more entities (e.g. people, objects, and nature) can be reduced to an economic problem that can be expressed as a single or a

series of two dimensional utility matrices (Straffin 1996; Colman 2016). One such problem discovered by Flood and Dresher (1952)(Embrey et al. 2017), which invoked curiosity towards non-rational behaviour, was the Prisoner's Dilemma.

A summary of the Prisoner's Dilemma problem is as follows as described by Axelrod (1984)(Leary & Baumeister 2017): Two prisoners are interrogated separately by the prison warden regarding a committed crime. Each prisoner is given the opportunity to accuse the other prisoner (i.e. in Game Theory terms called 'Defect' or DEF) or feign ignorance (i.e. in Game Theory terms called 'Cooperate' or COOP). A prisoner that accuses gets a reward from the prison warden, while a prisoner that is accused gets a penalty from the prison warden. However, if both prisoners accuse each other, then both prisoners gain nothing for confusing the prison warden. As a twist, if both prisoners feign ignorance, then both prisoners are given a small reward by the other prisoner for keeping silent. Neither prisoner knows what the other prisoner will do. This interaction is expressed in a two dimensional matrix shown in Figure 2.6 (a).

(a)

Prisoner 1 \ Prisoner 2	COOP	DEF
COOP	3 / 3	5 / -2
DEF	-2 / 5	0 / 0

(a) The Prisoner's Dilemma game matrix. Each prisoner chooses either COOP or DEF, which will realize an outcome represented by the cell intersections. For each cell intersection, the lower left number is the outcome realized by Prisoner 1, while the upper right number is the outcome realized by Prisoner 2.

(b)

Prisoner 1 \ Prisoner 2	COOP →	**DEF**
COOP	3 / 3	**5** / -2
↓ **DEF**	-2 / **5**	**0** / **0**

(b) The shaded cells show Pareto-optimal outcomes, i.e. preferred outcomes that are not exceeded completely by any other outcome. The arrows show which action has higher utility under any eventuality. The vertical arrows show Prisoner 1's dominant action choice (in bold), and the horizontal arrows show Prisoner 2's dominant action choice (in bold).

(c)

Prisoner 1 \ Prisoner 2	COOP	DEF
COOP	3 / 3	5 / -2
DEF	-2 / 5	0 / 0

(c) Both prisoners choose their dominant actions (DEF) but end up realizing outcome (0, 0) which is not Pareto-optimal. This is because there exists an equilibrium outcome that will always be realized if both prisoners rigidly follow a single measure of rational utility.

Figure 2.6 The Prisoner's Dilemma Game-Theory Matrix

Source: Axelrod 1984

The dilemma arises because both prisoners have an optimally dominant action, 'Defect', that will lead to an inferior equilibrium outcome, where both prisoners get '0' reward (Figure 2.6 (c)). Furthermore, the superior Pareto-optimal outcome (i.e. mutually beneficial outcome), where both prisoners get '3' rewards (Figure 2.6 (b)),

can never be realized because the action leading to the reward is sub-optimally non-dominant (Poundstone 1992; Zeng et al. 2016).

Under uncertainty, if both prisoners assume that the other prisoner will always decide on behaviour purely through rational utility, then only the inferior equilibrium outcome will ever be realized. Rapoport and Chammah (1965)(Sandoval et al. 2016) conclude that this is the inescapable consequence of the pursuit for optimal decision-making through a singular measurement of utility. Consider, however, what would occur if both prisoners can do either of the following:

a. Associate some degree of attachment (e.g. affection, fear) to the other prisoner, or

b. Be privy to alternate measurements of utility, for instance, which considers subsequent consequences a decision against the other prisoner.

In the two situations above, both prisoners would now be in a position to realize the superior Pareto-optimal outcome. Gerald Edelman (1987)(Barrett 2017), in his theory of neuronal group selection, states that the measure of utility can be adjusted by internalized value systems (Edelman 1987; Barrett 2017). The most recognizable internal value system uniform to most living organisms is the use of emotions (LeDoux 1996; Hart 2018).

The Prisoner's Dilemma teaches us that blind regard to singular evaluation functions under uncertainty can potentially lead to poor sub-optimal decisions. Real-life parallels of the Prisoner's Dilemma, where real prisoners are made to tell on their fellow prisoners, reveal that fear of future consequences override any promise of personal benefit (Poundstone 1992; Zeng et al. 2016). Even though the fear cannot be quantified in the same utility terms provided by monetary rewards, its existence is enough to cause the prisoner to consider alternate measures of value, for example, the prisoner's own survival. The same can be seen in a software competition, called the Iterated Prisoner's Dilemma (IPD), a version of the game that is played repeatedly by two artificially intelligent agents. Competition results of the IPD has proven that

agents that adhere to rigid evaluation functions perform weakly compared to agents that endeavour to identify with, or profile, opposing agents (Voth and Alfonsi 2005).

The Prisoner's Dilemma is not used as the problem domain to be solved by this research. However, it motivates this research by showing the following:

1. The criticality of properly determining which measures of value (i.e. evaluation function) to use under uncertainty or in situations of incomplete information, prior to making a choice of the course of action to take by the chosen evaluation function.

2. The attractiveness of internalizing a uniform unit of value when the problem presents multiple irreconcilable measures of utility for each choice (e.g. monetary value versus the value of survival).

By integrating the two lessons above in the design of a decision making agent, action choices which are previously seen to be irrational and sub-optimal from a strictly external value perspective can now be seen as being rational and optimal under an internal value perspective. The realization that emotions and affect similarly alter a living organism's value perspective warrants further investigation for its use in agent decision-making. This 'emotional affect' would enable the execution of non-dominant actions for the pursuit of superior outcomes. This insight provides the motivation for the creation of an affective agent framework for decision-making.

2.6 DISCOVERING WHAT IS NEEDED: THEORIES ON HOW EMOTIONS AFFECT BEHAVIOUR DURING PROBLEM-SOLVING

In order to find a model for developing a behavioural affect mechanism required to solve the intended problem domain, this research looks toward the natural world for initial insights on how organisms (humans, animals, insects and others) manage incomplete information and uncertainty when making decisions. As a reminder to the reader, this research aims to develop a mechanism that achieves successful behavioural affect for a game playing NPC agent, which will allow the

agent to cope with new objects and game rules that the agent has not been designed for. As previously specified, the problem domain for this research is scalable game environments, where the outcome of the game is made uncertain by the unknown rules that dictate outcomes of interactions in new game areas. Because of these undisclosed rules, the inherent patterns that would guide agents to conclusively predict expected outcome is obscured. Finally, the game is made dynamic and constantly changing due to the introduction of new and never-before encountered objects that react to these hidden rules in different ways. Some of the desired abilities of the behavioural affect mechanism needed for this problem domain are as follows:

1. To be able to compensate for the lack of complete information, by selectively utilizing other source of information to rationalize.

2. To be able to adapt to the changing conditions of the game.

3. To be able to prepare for future anticipated conditions through facts learnt about the opponent(s) within the game.

4. To be able to approximate the most appropriate learnt information when an unfamiliar state of the game is encountered.

One area of cognitive science that sheds some light into how organisms decide intelligently under limited or incomplete information is in studies on emotion. Oatley (1987)(Reeck et al. 2016) highlighted the need to consider emotions as being an integral tool for managing uncertainty after analysis on how mammals are able to cope with and solve cognitive design problems successfully, despite being deliberately given insufficient cues for solving these problems. De Sousa (1987)(Morag 2016) theorised that emotions provided this capability by "supplying the insufficiency of reason", allowing the organism to make inferences and proceed with making choices. LeDoux (1996)(Hart 2018) agrees along similar lines when he posited that organisms depend on emotions as a mechanism for self-preservation by cutting down on time for deliberation, or to compensate for insufficient information. Perhaps the best summary for the function of emotions, as it relates to decision-making under uncertainty, is

presented by Fellous (2005)(Moser et al. 2016): that emotion communicates information in a simplified yet high-impact package to the organism in multiple levels (physically, hormonally, and mentally).

The question this research asks is this: if emotions were to be a model for achieving advantageous behavioural affect under uncertainty in problem solving, what exactly is it about emotions that provide these functions?

The most prevalent opinion on what emotions are, and also the most recognizable opinion to the man on the street, is that emotions are distinct mental states that describe the current disposition of the organism. This opinion is the mainstay of the 'palette theory' of emotions; so-called due to its premise that there are a limited number of basic emotions (or palette emotions) through which all other emotions are derived. These basic emotions are recognizable in that it has a non-complex linguistic representation in emotional words (also called emotional tokens) such as 'fear', 'happiness', 'love', 'disgust' and many others. The idea behind palette emotions is that behaviour is controlled through the transition and flow between one emotional state to another. Researchers such as Scherer (1984)(Frijda 2016), Ekman (1994)(Coppin & Sander 2016), and Fellous (1999)(Wang et al. 2016) subscribe to the existence of palette emotions and have modelled cognitive structures for appraisal using palette theory to explain how coping works.

Unfortunately, a palette theory of emotions is unsuitable for the purposes of this research for the following reasons:

1. It fails to describe how cognitive structures based on a limited number of basic emotional states can supplement incomplete information and eventually lead to successful behavioural affect.

2. The premise that there are a few basic emotions is arguable as this is a linguistic constraint: some languages have a larger set of emotional representations than others.

3. Most importantly, the disposition implied by particular emotional tokens have no place or function in discorporate software agents dealing specifically with problem solving. There is no need for a software agent playing chess to feel 'love', for example. Likewise, if this chess-playing agent were designed to feel 'anger', does this mean that the moment anger is elicited the agent must suspend its rational processes and chance a poorly reasoned move?

For the reasons stated above, palette theory provides little in the way of insight into how emotion could be used to develop a behavioural affect mechanism for handling uncertainty. Even so, this did not prevent attempts to create such mechanisms in other research (El-Nasr et al. 2000; Gmytrasiewicz & Lisetti 2000; Vallverdú et al. 2016). The palette theory of emotions is mentioned in this section to acknowledge that this is what most people outside the field of cognitive science would immediately recognize as emotions. This research's decision to not employ palette theory is one of the reasons why the results of this research would be 'affective agent' framework rather than an actual 'emotional agent'. This distinction must be made because it is the purpose of this research to emulate the benefits of emotions in problem solving. It is not the purpose of this research to develop an agent that is recognisably emotional.

There are emotion theories that do not depend of the existence of palette emotions or the use of linguistic emotional tokens to explain how emotions can be beneficial to problem solving:

1. The cognitive structure of emotions provided by the OCC model (Ortony et al. 1988; Frijda 2017) describes how emotions are able to point out what is important in a particular problem situation.

2. The Somatic Marker Hypothesis (Damasio 1994; Lopez-Franco et al. 2018) reveals how the disposition an organism feels as emotions are really alterations of what are the important facts in a problem. The hypothesis also states that these dispositions are recalled later in the future, either as a response to a similar contextual stimulus or in anticipation of such stimulus.

The subsequent sections of this chapter will describe these emotion theories in more detail and also show how they are currently applied in existing artificial intelligence research.

2.6.1 Elements From The Cognitive Structure Of Emotions That Contribute Towards Behavioural Affect

Some designers of artificial intelligent systems that utilize emotions as a model for the decision-making process (Gratch 2000; Bazzan & Bordini 2001; Davidsson et al. 2017; Kowalczuk & Czubenko 2016) chose to define these processes around the OCC model of emotions. The OCC model of emotions is a cognitive structure of emotions produced by psychologists Andrew Ortony, Gerald L Clore, and Allan Collins (1998)(Frijda 2017) as a system for predicting the emotions that are experienced by a person, and for understanding the eliciting conditions responsible for invoking the experienced emotional state. The OCC model defines emotions as 'valenced' responses, which are defined as positive or negative responses directed towards three perceivable situational elements: events, agents, and objects. The polarity and intensity of these valenced responses serve to highlight aspects of the situation that are of personal significance to the perceiver. The medium for conveying this highlighted information is through distinctive physiological feelings and cognitions. In other words, the function of emotions under the OCC model is to provide information on what is important in the present situation.

The OCC model of emotions is of particular interest to this research due to its attempts to divorce the linguistic connotations of emotional tokens from the understanding of spirit of emotional function in decision-making[1]. The authors of the OCC model themselves reject the vagaries of palette emotions theory as being insufficient to define the relationship between emotions and behavioural affect. To achieve this, the cognitive structure of emotions presented by the OCC model groups

[1] It must be noted however that throughout their treatise, the authors still find it necessary to employ these emotional tokens to explain their model. This is done out of necessity to communicate within a relatable framework, and does not connote tacit support for palette emotions.

physiological states by the eliciting conditions that they share. These physiological states define emotions through the positive or negative affect they incur on the perceiver. Finally, intensity variables measure the degree of affect experienced by the perceiver[2].

- **Eliciting Condition** - Each eliciting condition centres around one of the three perceivable situational elements:

  1) Consequences of events

  2) Actions of agents

  3) Properties of objects

  This grouping of eliciting conditions is further refined through particular bindings related to each perceivable element. An example of further bindings for events is whether consequences of a particular event affects the perceiver (an internal binding) or someone else (an external binding).

- **Valence** - The key element for defining an emotion and the emotion's behavioural affect is the valence of the physiological states within each group. For example, '*like*' and '*dislike*' are respectively positive and negative valenced physiological states of an emotional group centred on objects.

- **Intensity Variable** - The magnitude of affect an emotion has is determined by intensity variables such as desirability (for events), praiseworthiness (for agents), and appeal (for objects). For example, the magnitude to which an event facilitates the realization of a particular consequence is defined by its 'desirability', whereas 'undesirability' expresses the magnitude in which the event hinders the realization of the consequence.

[2] The term 'perceiver' is used in this section only to distinguish the agent perceiving these elements from the perceived 'agent' element in the OCC model that is the subject of the valenced reaction.

Figure 2.7 exhibits the cognitive structure of emotions as presented in the original OCC model. The OCC model of emotions groups valenced emotions (opposite terms in parentheses) according to their eliciting conditions (squares) and intensity variables (bold). By defining emotions as physiological states that occupy each emotional group, any number of distinguishable physiological states can be recognized as an emotion without needing a corresponding emotional token to represent it linguistically. More importantly, this cognitive structure of emotions provided by the OCC model provides a quantifiable means for examining how emotions motivate behavioural affect.

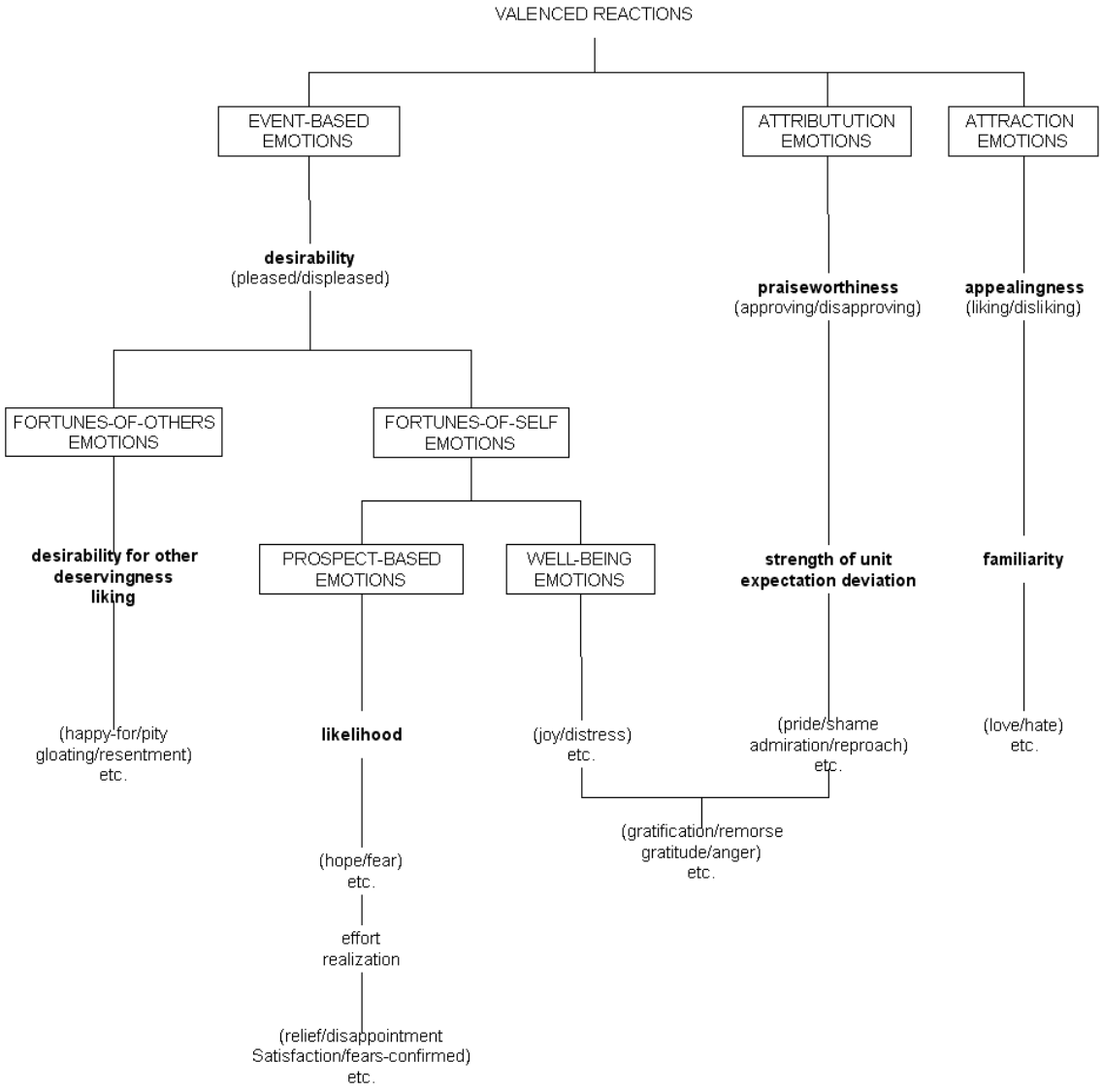


Figure 2.7 The OCC model of emotions.

(Source: Ortony, A., Clore, G., L., and Collins, A (1988)., "The Cognitive Structure of Emotions", Cambridge University Press, p. 69)

In the OCC model, emotional states can be abstracted as a value of the intensity variable for a particular valenced emotional state at a particular time. This abstraction specifies the importance of a perceivable situational element to the perceiver in terms of a specific emotional state. An example of this abstraction is as follows:

- *DESIRE* (*p, e, t*) = the measure of how much a person, *p,* desires a particular event, *e,* to occur at time, *t*.

Abstractions of emotional state are intended to provide a basis for behavioural affect, through the maintenance of emotional state values within some acceptable range. Ortony, Clore, and Collins (1988)(Frijda 2017) posit the use of thresholds for each emotional state abstraction in order to define when behavioural affect should occur. Going over or under this threshold would either:

a. Trigger the adjustment of other emotional state values according to the relationship between emotional states in the OCC model, or

b. Trigger a behavioural response from the perceiver in order to mitigate the increase or decrease of emotional value and restore emotional value to within acceptable ranges.

An example of the former, as provided by Ortony, Clore, and Collins (1988)(Frijda 2017) themselves, and repeated here for illustrative purposes only, is shown below:

IF JOY-POTENTIAL(p, e, t) > JOY-THRESHOLD(p, t) THEN
set JOY-INTENSITY(p, e, t) = JOY-POTENTIAL (p, e, t) – JOY-THRESHOLD(p, t)
ELSE
set JOY-INTENSITY(p, e, t) = 0

An extended set of similar IF-THEN rules can potentially be used to establish relations between each and every emotional state within the cognitive model more vividly. Such extensions have led to the construction of some types of emotional agents, which use the dynamic interplay of values change within the emotional states to motivate behavioural affect (Bazzan & Bordini 2001; Davidsson et al. 2017). Behavioural affect mechanisms, or simply coping mechanisms (Gratch 2000; Kowalczuk & Czubenko 2016), for maintaining emotional values have also been a product of the abstraction of emotional states within the OCC model[3].

It is important to note that the development of coping mechanisms using the OCC model goes beyond the authors' intended purpose of defining a cognitive structure of emotions in the first place (Ortony et al. 1988; Frijda 2017). To repeat, the intended purpose of the OCC model is to predict the emotions that are experienced by a person, and for understanding the eliciting conditions responsible for invoking the experienced emotional state. The main contribution of the OCC model specific to this research is the revelation that emotions are not dispositions invoked in ether. Rather, emotions are measurable valenced reactions elicited toward perceivable situational elements (events, agents, objects). As mentioned in the beginning of this section, these emotional reactions are elicited for the purpose of providing information on what is important in the present situation. This revelation suggests an important lesson relevant to this research: that the flexibility exhibited by living organisms in managing problems under uncertainty with emotions is the result of a process of identifying what is important in the problem through valenced assessments.

2.6.2 Elements From The Somatic Marker Hypothesis That Contribute Towards Behavioural Affect

Research in neuroscience provides some additional insight into how emotions contribute towards flexible behavioural affect under uncertainty during problem solving. Antonio Damasio (1994)(Lopez-Franco et al. 2018) presented a comprehensive neurological analysis that supports the criticality of emotions during

[3] The example systems mentioned here do not exclusively use the OCC model in their design, as they often include concepts from palette theory and other emotional theories. However, the reliance on the OCC model in these designs are prominent.

decision-making and problem solving in his Somatic Marker Hypothesis. Similar to the OCC model of emotions, the Somatic Marker Hypothesis avoids reference to linguistic emotional tokens and palette emotions. Instead, a systemic view of emotions rooted on the physiological and mental state of the perceiver is taken. The hypothesis asserts that the brain produces 'marker' signals, which are representative of the body state, born as a result of the bio-regulatory processes triggered in response to experienced stimulus. The somatic markers would later covertly re-emerge from the brain's representation of the *soma* (i.e. Latin for '*body*') in anticipation of the stimuli, and manifests itself in the decision-making process by forming selection biases (Damasio 1996; Kanbara & Fukunaga 2016). Damasio recognizes these body state representations, or "somatic markers", to be synonymous to what organisms experience as emotions, which suggests to this research two additional contributions emotions make towards decision-making:

1. That emotions are learnt body states invoked upon experience or anticipation of similar contextual stimuli, and

2. This invocation of emotion results in the recollection of decisional bias from past body states, thereby leading towards behavioural affect for present decisions.

The Somatic Marker Hypothesis was conceived by Damasio as a result of examination of patients with brain lesions within the prefrontal cortex, specifically the ventromedial sector. This area of the brain is recognized to be a repository for associations between the situational context (i.e. facts that make up the context of a particular situation) and the disposition to recall an emotion as a result of that context form the amygdala, the emotional centre of the brain (Damasio 1996; Bechara et al. 2000; John et al. 2016; Kanbara & Fukunaga 2016). Patients with damage to this area preserve normal levels of learning, intelligence, and perception, but fail to experience or express emotion within complex or social situations. As a result of this failure, these patients are commonly observed to have lost the capacity for making advantageous decisions. This observation led Damasio to hypothesize the function of

emotions as a '*body-loop*' that alters the person's physical and mental state in preparation for the pertinent facts of the situation.

Control patients with none of the associated brain damage positively show physiological changes, in the form of skin conductance responses, as a result of emotional invocation, but the target patients do not. The lack of somatic change is followed by a lack of insight regarding the situational context, which explains the patients' inability to make proper action choices in tests under uncertainty. This led to the conclusion that emotions are responsible for the recollection of pertinent facts regarding a particular context, and that these recalled facts are instrumental for making advantageous decisions within the context (Damasio 1994; Lopez-Franco et al. 2018).

The theory formulated to describe this observation of emotional affect is called the '*As-if body-loop*'. When a person experiences new stimuli for the first time, the physiological change experienced by the body generates a valenced (i.e. positive or negative) emotional response of attraction or repulsion in reference to the resulting body state. This is illustrated in Figure 2.8(a) and is generally known as a normal '*body-loop*' (Bechara 2004; Nelson et al. 2016). Upon continuous observation of the environment, the same emotional response, known as 'affect', is propagated by the ventromedial pre-frontal cortex to the amygdala, and other associated areas of the brain. This occurs in anticipation of the same stimuli recurring, but most crucially, without the stimuli actually happening. This influence of emotional affect creates a gut-feeling that either urges the person to realize the stimuli once again, or repulses the person enough to prevent the stimuli from being realized. This spontaneous and continuous influence of affect is called the '*As-if body-loop*'. An illustration of the '*As-if body-loop*' is shown in Figure 2.8(b). The entire operation of the '*As-if body-loop*' describes, from a neurological perspective, what we recognize to be the influence of emotional affect on decision-making.

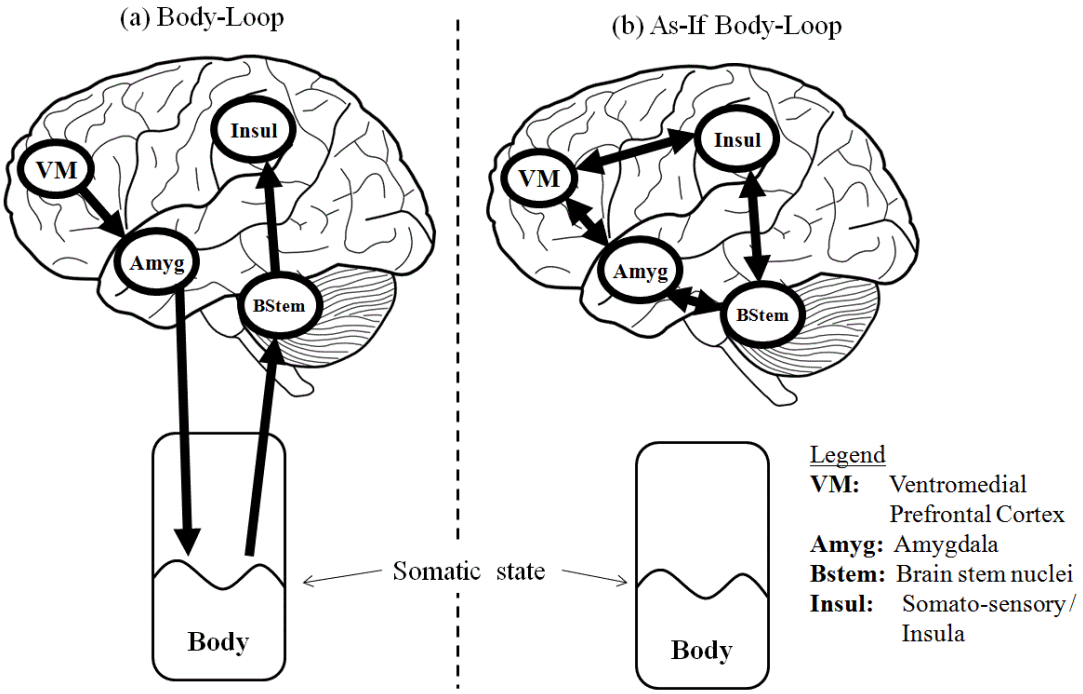


Figure 2.8 The '*Body-loop*' and '*As-if body-loop*'

Source: Bechara, A. & Damasio, A. R. 2005. The somatic marker hypothesis: A neural theory of economic decision. *Games and Economic Behavior*. 52: 336–372.

Apart from revelation that one of the functions of emotions is to recall facts for making advantageous decisions in problem, this research is further interested in the Somatic Marker Hypothesis due to the manner in which the hypothesis reveals how emotions redefine problems. A series of experiments, called the Gambling Test, were performed by Antoine Bechara et al. (1994)(Hiser & Koenigs 2018) to prove the Somatic Marker Hypothesis. These tests provide clues to the process in which emotions establish the pertinent facts for problem solving. The purpose of the Gambling Test was to see whether target patients, with impairments in the ventromedial prefrontal cortex but with intact cognitive capabilities, would be sensitive to the future consequences of their actions when handling situations of uncertainty. During the test, each patient was required to make a profit from a loan by selecting cards from different decks. Some decks had uniformly high rewards but disproportionately higher hidden penalties; other decks had low rewards but negligible penalties. The hidden penalties occur frequently in some decks, and less so in others. While all patients were observed to initially prefer the higher rewarding decks, the control patients tended to veer away from these decks after learning of the high

penalties they suffered as a consequence. Observations from the tests showed that target patients constantly failed to anticipate the magnitude of unknown future penalties, which resulted from their continual favouring of the apparent high rewards that reinforced their decisions. It was later surmised that the lack of emotional capacity caused by the target patients' impairments resulted in less sensitivity to prospects of success or failure (Damasio 1994; Lopez-Franco et al. 2018).

The results of the Gambling Test suggest that the use of these hypothetical somatic markers signalled the appropriate reaction in response to the prospect of outcomes. Specifically, somatic markers reinforced the emotions associated to outcomes, and subsequently motivates behavioural affect. What was of greater interest to this research regarding emotional problem-solving however, was that the collective association of somatic markers to the outcomes of all decks was able to cause a redefinition of the problem which then suggested the form of desirable outcomes. The task assigned to the patients at the start of the test was to increase profit, but as the test progressed, the patients' emphasis shifted more towards avoiding catastrophic loss in their card selections without any instructions from the tester. Since the new task did not obstruct the accomplishment of the original task, this realization justified the control patients' selection of decks with lower payoffs.

The first important lesson that can be taken from the Somatic Marker Hypothesis for this research is that it reveals an additional function of emotion that provides behavioural affect advantages to organisms when dealing with uncertainty in problems. Specifically, the Somatic Marker Hypothesis shows that emotions are used to recall the pertinent facts of the problem to be solved in order to make an advantageous choice. The second lesson important to this research is the Somatic Marker Hypothesis reveals the manner in which emotional problem-solving works when there is uncertainty in the problem. The gambling tests have shown that emotions are important for periodically redefining what are the important and noteworthy facts that need to be paid attention to as the problem changes, so that advantageous choices can be continually made. These two additional items about emotions are used in this research for the creation of a behavioural affect mechanism

that can handle uncertainty introduced by interactive complexity when solving problems.

2.7 CURRENT APPLICATION OF EMOTION THEORY FOR ACHIEVING BEHAVIOURAL AFFECT DURING PROBLEM SOLVING

Both the OCC model of emotions and the Somatic Marker Hypothesis have been used for the creation of intelligent agents, be it agents for problem-solving, communication and expression, or simulation. These agents are loosely categorised as either ‘emotional agents’ or ‘affective agents’, i.e. agents that are motivated by the impact of experiencing emotions. Wehrle (1998)(Soleimani & Kobti 2016), provides a more functional categorisation of emotional agents, separating them into either scientific emotional agents, engineering emotional agents, or HCI emotional agents. It must be noted that regardless of what the final agent will be called or categorised as, almost all types of emotional agents make use of some mixture OCC model of emotions, the Somatic Marker Hypothesis, and even palette emotions as suggested by Scherer (1984)(Frijda 2016) and Ekman (1994)(Coppin & Sander 2016) and rarely are their designs motivated exclusively by a singular emotional model. Four distinct areas of agent research which directly employ emotion theory are:

1. Emotional elicitation mechanisms

2. Appraisal and coping mechanisms

3. Emotional state transition mechanisms

4. Homeostatic mechanisms

It is important to acknowledge these different ways in which emotional theory has been used to create existing agents in order to understand the reasons for this research’s hesitance in calling the final agent design an emotional agent, even though

strict adherence of the lessons of emotional theory to problem-solving has been maintained.

Cognitive models of emotions, such as the one provided by the OCC model, has been the basis for the emotional elicitation mechanisms. In particular, the cognitive model provides some structure for the flow between emotional states to govern behavioural affect or emotional expression. The Oz project by Reilly (1996)(Smith 2017) and Bates (1994)(Kaptein et al. 2016) employs the OCC model in the design of their 'Em' emotional model, which governs the behaviour of agents in their simulated world. An artful use of emotional elicitation is the assessment of the emotional message within hand-drawn images (Sengers et al. 2002). As a more direct adoption of the OCC model, Bazzan and Bordini (2001)(Davidsson et al. 2017) use similar abstractions of emotional state previously shown in Section 2.5.1 for the creation of extensive IF-THEN production rules to achieve behavioural affect. A shared characteristic of agents that use cognitive structures of emotions for the purpose of emotional elicitation is that specific agent behaviours are attached to each valenced pair of emotions within the cognitive structure. The compound values of each valenced pair at any moment determine the type of behaviour, or emotional expression, to be exhibited by the agent. The cognitive structure itself provides the connections or hierarchy for the emotions to be elicited.

Systems that simulate appraisal and coping, such as those created by Gratch and Marsella (2001)(McDuff & Czerwinski 2018), employ the concept of emotions as dispositions as used in the Somatic Marker Hypothesis combined with the concept of emotions as a reaction to perceivable situational contexts in the OCC model in order to simulate emotional cause and effect. In appraisal and coping mechanisms, particular events within the simulation cause a shift in the valence of opposing emotional states, thereby changing the emotional disposition of the simulated character towards the player. This effectively expands or limits the range of interaction choices that can be made between the player and simulated agent. Such applications of emotional theory are useful for the purpose of training players to cope in emotionally charged situations and teach them how to bring the situation down to a more emotionally manageable level. EMA, the appraisal and coping simulation

created by Gratch and Marsella (2001)(McDuff & Czerwinski 2018), for instance, is used to train U.S. Army soldiers what effect their interaction choices may have in a simulated battlefield.

Emotional state transition mechanisms follow a similar vein to emotional elicitation mechanisms, in that it uses the cognitive structure of emotions to cause behavioural affect or exhibit emotional states. The main difference that emotional state transition mechanisms have is that behavioural affect is the result of the movement between one emotional state to another. In this sense, emotional state transition mechanisms recognize the importance of palette emotional states for providing the hierarchy of emotional affect. Gmytrasiewicz and Lisetti (2002)(Adam et al. 2016) directly employs an emotional state transition mechanism in order to control the decision-making bias of agents that play the spatial version of the Iterated Prisoner's Dilemma effectively. FLAME, a fuzzy logic adaptive model of emotions created by El-Nasr et. al. (2000) uses emotional state transition mechanisms to govern how emotions are expressed, to be ultimately used as a computational model of emotions within different user-interactive programs and software interfaces.

The concepts of valence, intensity variables, and somatic recall of emotions are used to create homeostatic mechanisms for balancing emotional states. In homeostatic mechanisms, valenced weights are attached to a set of emotional states, whereby particular sets of weight ranges exhibit a desirable state for the agent to be in (Gadanho 1999; Kirandziska & Ackovska 2017). Actions performed by the agent, and also the current status of the agent, alters the weights in each emotional state in different ways. Homeostatic imbalance occurs when the weights of particular emotional states exceed acceptable thresholds. As a result, the agent would modify behaviour in order to restore the homeostasis of its emotions. Homeostatic mechanisms have been successfully demonstrated to affect the behaviour of robots in obstacle avoidance tests by Gadanho and Hallam (2001)(Moerland et al. 2018) and in similar robotic control research (Maçãs et al. 2001b; Morgado & Gaspar 2005).

Although these four areas of agent research employ the emotion theories afforded by both the OCC model and the Somatic Marker Hypothesis, one departure

from the spirit of these theories is the extensive use of linguistic emotional tokens to represent agent or decisional state. The reader is reminded that both the OCC model and the Somatic Marker Hypothesis is presented as language neutral alternatives of emotions in order to understand how the emotional process benefits decision-making directly (see Section 2.6.1 and 2.6.2). Therefore the use of linguistic emotional tokens is epiphenomenal[4] to demonstrating how behavioural effect under uncertainty is successfully achieved with emotions. Even so, the usage of these emotional tokens under mechanisms inspired by the OCC model of emotions, the Somatic Marker Hypothesis, and palette emotions does have merit in that it legitimises the use of the label 'emotional agent' for these systems.

2.8 OVERVIEW OF AFFECTIVE COMPUTING

Affective computing is a relatively new branch of computer science, spearheaded by Rosalind Picard (1995)(McStay 2018), that studies how emotions can be emulated or mimicked in software, particularly in artificial intelligence (AI) systems. This emulation of emotions is done either in order to enhance human-computer interaction (HCI) or to improve the decision-making processes of software agents (Picard 1997, Fairclough 2017). Current research in emotional agents encompasses many different areas, such as the understanding of the nature and implications of emotion, the development of artefacts based on emotional agent theory for task completion, and use of emotions in human-computer interaction (Wehrle 1998; Soleimani & Kobti 2016). However, we can generally categorize emotional agent research into two distinct areas, which are:

1. Emotional characters, and

2. Emotional problem-solvers

[4] 'Epiphenomena' used in the context of this research refers to implementations of emotional artefacts that only have a cosmetic effect for qualifying the use of the label 'emotional' or 'affective' in describing agent behaviour or agent algorithms. Epiphenomena can be considered 'window-dressing' that has no real bearing on the process or outcomes of decision-making. This means that generic non-emotional tokens can take the place of epiphenomena, and still display the same result.

2.8.1 Emotional Characters

Emotional characters are intelligent agents that emulate emotions through computational mechanisms to produce behaviour and perform actions that match the agent's current emotional state. Emotions are used in these types of agents as a means for expression which will aid in the development of believable artificial personalities that can be embedded in AI characters, robots, and human-computer interface systems. One of the earliest and most cited projects in emotional agents of this type is the Oz project (Reilly & Bates 1992; Bates et al. 1992; Broekens et al. 2016; Hudlicka 2016). The Oz project involved the development of an extensive virtual world populated by artificial life (ALife) characters that use emotional computation mechanisms in multi-agent interaction. The purpose of the project was to observe how emotions evolve the characters' behaviour so that the users playing in the Oz world would have a rich and realistic interactive experience (Reilly 1996; Smith 2017). Another commonly cited project is the Émile project (Gratch 2000; Gratch & Marsella 2004; Kowalczuk & Czubenko 2016; Alfonso et. al. 2017). The Émile project focused on the development of synthetic human characters for the use in the U.S Army's civilian interaction training programs, so that real soldiers can practice making strategic decisions involving civilian life during emergencies before actually going into the field.

In recent years, this type of affective computing research has also been extremely valuable in the computer and electronic entertainment industry. Numerous popular toys, robots, and computer games can attribute their multi-million dollar success to the application of emotional character principles. Some well known examples are like the Tamagotchi (1997) bird egg toy, the Aibo (1999) pet robot dog, and The Sims (2000) life simulation computer game.

2.8.2 Emotional Problem-Solvers

Emotional problem-solvers are intelligent agents that use emotional computation mechanisms for more practical tasks such as motivation, exploration, discovery, learning, and decision-making. In this type of agent, emotions are used as

the agent's core AI algorithm that governs not only the agent's operation but also the rationalization of action responses to its percepts. One commonly cited research explores the use of emotional computation mechanisms to improve the autonomous behaviour of a learning agent (Gadanho 1999; Gadanho & Hallam 2001; Gadanho 2003; Ghayoumi & Bansal 2016; Kirandziska & Ackovska 2017; Moerland et al. 2018). These researchers explored the use of emotions to influence the agent's perceptions, provide reinforcement value from objects in the agent's environment, and in deciding whether to re-evaluate a previously made decision. A different approach taken in emotional problem solving research explored how emotion can affect behaviour in multi-agent interactions (Bazzan et al. 1997; Bazzan & Bordini 2001; Anantsuksomsri & Tontisirin 2016; Davidsson et al. 2017). By analyzing the performance of emotional agents in the spatial version of the Iterative Prisoner's Dilemma (IPD) game (Axelrod 1984; Leary & Baumeister 2017), the researchers observed that the use of emotion was able to affect cooperative behaviour among other agent types, thereby guaranteeing the success of emotional agents in such games.

Despite its potential in providing new insights in the more practical uses of emotion in artificial intelligence, research in emotional problem-solvers have been slower to develop than that of emotional characters. One of the reasons is that industrial demand for emotional characters far outweigh that of emotional problem-solvers (Bates 1994; Kaptein et al. 2016). Another reason, which is elaborated in detail in Section 2.7 is the problem faced in validating the utility of decisions made through emotional affect in models that rely on characterized sets of emotions. Nevertheless, the rarity of research work in the area of emotional problem-solvers means that there is ample opportunity for new research to contribute to more practical applications of emotional agents.

2.8.3 Types of Agents in Affective Computing Research

To avoid confusion, this research distinguishes artificially intelligent software agents that embody affective computing methods and mechanisms into two distinct types of agents:

1. **Emotional Agents** - Agents that are involved with the detection and expression of emotions to enrich HCI are generally referred to as emotional agents.

2. **Affective Agents** - Agents that imitate ‘affect’, i.e. the emotional processes and influence which motivate or bias behavioural change, are aptly called affective agents.

This research does not delve into expressive emotional agents, nor does it attempt to integrate characterised sets of emotions into any solution for the problem domain. Rather this research will focus on emulating emotional affect in affective agents. Research in affective agents focuses on the development of algorithms and heuristics that provide the same flexibility in software agent behaviour, as that which is afforded by intuition, instinct, or ‘gut feelings’ in living organisms. This is done for the purpose of improving affective agent performance in decision-making or problem solving. Affective agents are distinct from emotional agents in that affective agents do not communicate emotion through emotional expression (Wehrle 1998; Soleimani & Kobti 2016).

2.8.4 Using Affective Agents to Play in Scalable Game Environments

One of the roles of emotions during decision-making, as postulated by philosopher de Sousa (de Sousa 1997; Morag 2016), is to supply the insufficiency of reason. Specifically, in situations where a pre-trained rational judgement may fail, either due to complete unfamiliarity with the problem, or as a result of being encumbered by the mass of features to be processed *apriori*, the emotional state becomes the basis for identifying or summarizing what is important, so that rational judgement may commence. This phenomenon of affect is of particular interests, as it describes a mechanism that is exactly needed in order make intelligible decisions in the problem domain explored for this research, i.e. scalable game environments.

Some evidence of this phenomenon, i.e. demonstrating of the usefulness of emotional affect for making (paradoxically) rational decisions, have been provided by (Bechara et al. 1994; Hiser & Koenigs 2018). An experiment now known as the Iowa Gambling Test is a card-flipping game developed to analyze the impact on rational behaviour in patients that have physical damage to the 'emotion' areas of the brain. The experiment fulfils the 'partial information' and 'stochastic' property of scalable game environments. The experiment discovered that emotional affect contributes towards the association of feature values to predicted game outcome, and towards a re-evaluation of important goals through affect. Bechara et al. (1994) describes the Iowa Gambling Test in detail. The results of the Iowa Gambling Test appear to suggest that emotional affect can tell a person how to play a game well, when game has never been played by the person. In different terms, the test suggests that emotional affect may be useful when playing partially observable stochastic games, when the game rules are poorly described. In relation to the problem domain for this research, the test suggests that a mechanism emulating emotional affect may allow a game AI agent to cope with new game objects and unknown game rules that the agent may have to face with in scalable game environments.

2.8.5 Current Affective Decision Making Research in Games

To avoid dismissals on the reliability of decisions by emotional agents, which by design are expected to make justifiably 'correct' decisions, there have been some research that explicitly depart from the use epiphenomenal emotional language tokens in their design. Since these mechanisms do not have anything that can be pointed out as being expressively emotional, i.e. communicating the emotional state, they are called 'affective' mechanisms instead. The aim of affective decision-making mechanisms is to try to emulate the specific functions of emotions when making decisions. The approach taken to achieve the required functionality can be a mixture of rational, statistical, adaptive, and even emotional methods. This section looks at some affective methods that endeavour to retain the justifiability of their actions, particularly when making decisions in computer games.

1. **Weighted emotional appraisal and behavioural association** – This method of affective decision making uses emotional states as affective placeholders for the elicitation of pre-programmed behaviours upon the activation of the emotional state. The methods usually begin by determining which emotional state the agent is in, a process which is called 'appraisal'. The activated emotional state will dictate which behaviour the agent is to use. Figure 2.9 shows the general framework of game-playing agents that used this method:

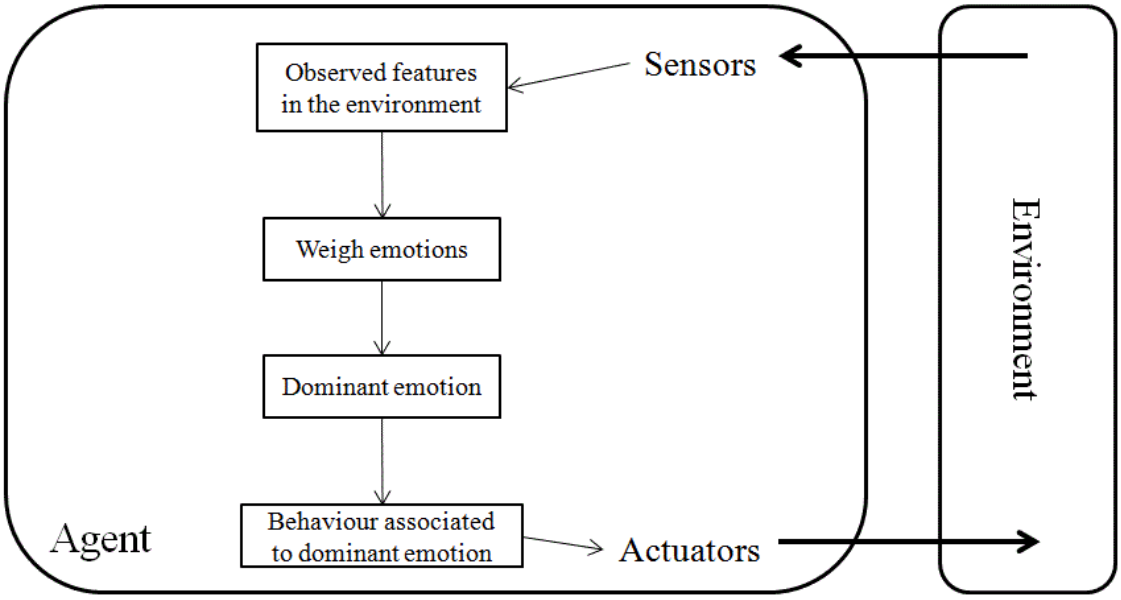


Figure 2.9 General architecture of weighted emotional appraisal and behavioural association methods of affective decision-making.

The following are some existing affective agent frameworks that use some form of weighted emotional elicitation for playing games:

- Burghouts et al. (2003) provided an algorithm for game character agents in a game called Gridworld. Each agent is equipped with an Emotion Process Module, which is centred on a few primitive emotional states. For every event faced by the game characters, whether they are encounters with other game agents or the occurrence of game environment changes, the Emotion Process Module appraises the event using three parameters: Potency, Impact, and Intensity. The event appraisal specifies which emotions are elicited, and the corresponding behavioural associated with the elicited emotion is executed.

- A similar method of emotional appraisal is also done by Baillie-de Byl (2003) in her EMAI (Emotionally Motivated Artificial Intelligence) architecture. The EMAI architecture is used to elicit sets of behaviours of a virtual pet dog, where each set of behaviour is distinctly characteristic of a particular emotional state. 15 emotional states are used: ranging from 'happiness' to 'guilt'. A distinct behaviour set is associated with each state. The EMAI architecture works by appraising events in the game environment using a weighted sum of six orthogonal appraisal dimensions (i.e. 'pleasantness', 'anticipated effort', 'certainty', 'attentional activity', 'responsibility' and 'control') in relation to each emotional state. This determines which emotion is elicited, and hence, which behaviour set is used by the virtual dog.

- The method of affect appraisal across various emotional parameters as shown in EMAI is also used by Johansson and Acqua (2009) in the Emotional Behaviour Network, or EmoBN. EmoBN is a affect-activated behavioural state transition mechanism. It uses 4 elicitation parameters to determine the strength and influence of each emotional state. The change in the emotional states subsequently causes the parameters of behaviour networks attached to each emotion state.

2. **Drive-model emotional appraisal** – This class of affective decision-making methods takes a different approach by not appraising the agent's emotional states directly. Instead, intermediate states that serves as the agent's motivations or 'drives' are first used to appraise the agent's overall performance. For game-playing agent, the various game-related attributes of the agent in the game, such as 'health', 'damage' and 'gold' are used as the intermediate state. For general software agents, however, each of the agent's goals acts as the intermediate state. Depending on the state of the attributes or goals, different emotional states will be elicited. For example, if the goals of an agent have not been successfully reached past a set threshold, the agent would transit to a specific emotional state. As with the weighted emotional appraisal, each emotion is associated with a particular behavioural set. Figure

2.10 shows the general framework of game-playing agents that used this method.

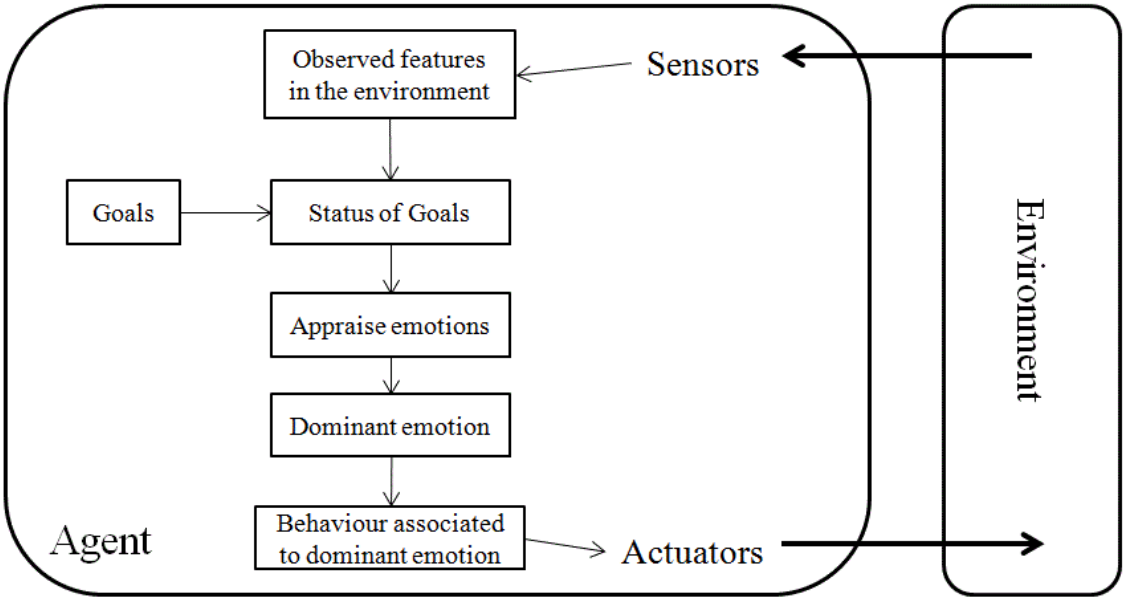


Figure 2.10 General architecture of drive-model emotional appraisal and behavioural association methods of affective decision-making.

The following are some existing affective agent frameworks that are appraise emotional states using the drive-model:

- In CLIPS (Chaplin & El Rhalibi 2004), a rule-base system is used to evaluate the agent's performance in the game. The rules are tailored specifically for the game environment. The rule-base act as the emotional drive model that governs emotional state transition in a finite-state machine (FSM) decision structure. This method has been used to control agent behaviour in game-theoric situations such as the Iterated Prisoner's Dilemma (IPD). Depending on the emotional state elicited as a result of the rule-base, the agent would be motivated adopt a strategy that it feels will benefit it in the long run with any opponent it faces in the IPD game.

- A combination of the weighted and drive-model emotional appraisal methods has been used by Lim et al. (2010). An NPC agent playing an educational role-playing game called ORIENT (Overcoming Refugee Integration with

Empathic Novel Technology) is equipped with a cognitive appraisal-based method called the FAtiMA architecture. The FAtiMA architecture used continuous planning to actively appraise 22 emotion types. The NPC agent is also equipped with a drive-model for goal appraisal called the PSI model. The PSI model sets thresholds to the various goals of the agent. The appraised emotions in FAtiMA alter the levels of the goal-status in the PSI model. When the goal-levels exceed particular thresholds, various behaviour sets are elicited. Distinct behaviour sets are used for different permutations of the goal-levels.

3. **Other methods** - Apart from the weighted emotion appraisal method and the drive-model emotion appraisal approach, other methods of affective decision-making in games have been used. Among them are adaptive learning, logical, and statistical methods. The following affective-decision making methods are some of the alternate approaches currently available:

- Shihab (2009) has produced an agent framework called the Emotional Decision Making Model that utilizes adaptive learning for emotional appraisal. In the Emotional Decision Making Model, Q-learning (Watkins 1989; Zaremba et al. 2016) is used to assess the agent's current performance in a game. The reinforced 'Q' values would control the transition of the agent's emotion states, specifically a 'normal' state and 'anxiety'. This method has been used to control agent navigation in a game which involves multi-agent communication (Shihab & Chalabi 2009).

- First-order logical evaluation and fuzzy-logical evaluation have methods have also been used in affective decision-making methods. Sardo (2011) has created a BDI (*Belief-Desire-Intention*) agent framework that utilizes the OCC model of emotions (Ortony et al. 1988; Frijda 2017) for playing the 'Trust Game', another game-theoric type game. The probability of success or failure of the agent is evaluated to alter the values of emotional 'triggers'. When the triggers are activated, first-order logical evaluation of triggered emotions would activate subsequent emotional states. The state that the agent ends up in after the logical evaluation dictates the behaviour set that will be used.

These methods represent only a sample of the available affective decision-making methods currently being developed today. Although the approaches used are different, they all share a commonality in that 'emotional states' are still present as the basis for behaviour selection. However, the emotional states, state-transitions, and associated behaviour set for each state still has to be hardcoded by the agent designer, in order to tailor it to the immediate game environment being played. For this reason, the reliance on emotional states and state transitions, much like FSM, would be of limited use to this research in particular, since it will not be suitable for scalable game environments.

Even so, a lesson from these methods that is of immediate use to this research is that the both the 'antecedents' for affective decision-making (i.e. appraisal) and 'consequents' for affective decision making (i.e. behavioural elicitation) can be devoid of emotional labels, while still delivering the benefits of emotional phenomenon. This revelation allows this research to develop a mechanism that mirrors the physiological functions of emotional affect, such as the Somatic Marker Hypothesis (Damasio 1994; Lopez-Franco et al. 2018), without needing to refer to epiphenomenal emotional labels to qualify calling the results of this research and 'affective agent'.

2.8.6 The Problem with Existing Affective Agent Frameworks

The main problem with the weighted emotional appraisal agent framework and the drive-model emotional appraisal agent frameworks is that both these methods still rely on the use of emotional tokens as intermediary states for making the decisions. Table 2.4 compares these existing agent types to the agent framework that will be built by this research: the Affective Decision Making Engine.

Table 2.4 Comparison of existing affective agent frameworks.

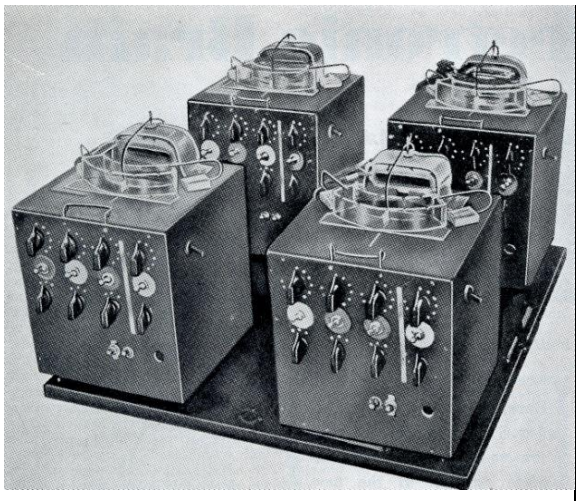
	Weighted emotional appraisal framework	**Drive-model emotional appraisal framework**	**Affective Decision Making Engine (ADME) framework**
Uses emotional tokens	Yes	Yes	No
Is based on an arbitrary measure of utility	Yes	No	No
Requires pre-conversion of external feature values to internal state values	Yes	Yes	No
Can handle untrained features	No	No	Yes

Both existing agent frameworks rely on the use of epiphenomenal emotional 'words' as a tike placeholder representing states. Each of these emotional tokens are encoded manually by their designers to make the agent react a particular way upon reaching a particular emotional state. This behaviour is unnecessary for a software agent that is more concerned about game performance. The value of each of these emotional tokens must also be pre-defined by the agent designer in order to allow the conversion of external utility values to internal state values. Because of this limitation, the existing agent frameworks are still constrained to static game environments in order to function properly, as they will not be able to adapt if they are placed in a scalable game environment. The ADME agent framework that will be built by this research will not suffer this limitation because it will rely on its own performance measures as a basis for emotional affect and decision making.

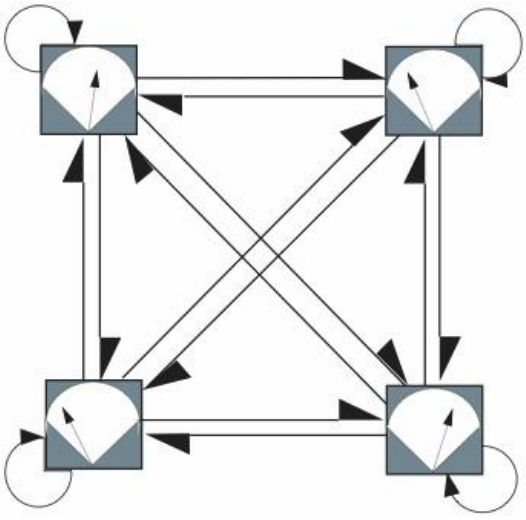
2.8.7 Homeostats and Affective Homeostasis

One particular device that emulates some of the somatic (i.e. body) functions of emotions is a homeostat. A homeostat is an ultrastable self-organizing system, which was first introduced by cybernetician W. Ross Ashby (1952) (Seth & Friston 2016) to emulate the self regulating activities of the human body in a mechanical device.

Homeostats operate by striving to maintain a state of equilibrium in the potentiometers that make up the homeostat's 'body'. Each potentiometer is attached to an element that can be influenced, or 'perturbed', by environmental conditions. For example, in the original homeostat, four magnets were used as the elements, and the magnet's angular deviations in a trough of water became the influence by the environment (Ashby 1956, DeGroff & Neelakanta 2018). The homeostat is initialized in a state of equilibrium, where each magnet is positioned so that it resists displacement by the magnetic fields of all other magnets. However, perturbations in the environment (i.e. water) displace the magnets from equilibrium, realigning their angular deviations. The potentiometers detect this deviation and coordinate the repositioning of all magnets such that the system returns to equilibrium. A self-organising system that has returned to equilibrium is said to have achieved 'homeostasis' (Ashby 1962, Ye, Zhang & Vasilakos 2017). Figure 2.11(a) provides an illustration of the original homeostat, consisting of feedback from four magnets, while Figure 2.11(b) shows the change in feedback from each magnet when perturbed, and process of adaptation to regain equilibrium in the homeostat.



(a) Photograph of the original Homeostat. Four sets of magnets in a trough of water sit atop four potentiometers that measure their angular deviations



(b) When the homeostat loses equilibrium, the angular deviations will be feedback to every potentiometer. Random perturbation alter the magnets position to bring the homeostat back to equilibrium

Figure 2.11 Illustration of W. Ross Ashby's Homeostat

Source: Ashby, W. R. 1952. *Design for a brain: The origin of adaptive behaviour*. London: Chapman & Hall.

In its simplest form, when only one element is being regulated, the operation of the homeostat is similar to that of the thermostat in an automatic kettle. The thermostat of the kettle triggers the heating coil when it detects the water temperature dipping below a set threshold. When the water has been reheated above the threshold, the heating coil is switched off, and the process repeats. What makes a homeostat different from a thermostat is that the homeostat tries to achieve equilibrium in multiple diametrically opposed elements, where perturbations in one element will propagate to the other elements (Eldridge 2002; Stovold 2016). Therefore, each permutation of the element's perturbations must be mapped to different actions that will bring the entire system to equilibrium. In other words, the action performed must ensure that all elements are returned to their desired states above the threshold. In a closed system, all permutations of the elements can be known beforehand; therefore the resolving actions for each permutation can be hardcoded into the homeostat. In an open system, however, is not possible to know what all the permutations are, therefore the homeostat has to learn the correct resolving action to take on the fly. In Ashby's original homeostat, upon perturbation of a single magnet, a randomizing circuit alters the angular deviations of all remaining magnets, until such a combination is found that brings the system back to equilibrium (Ashby 1956, DeGroff & Neelakanta 2018). In a living organism, it is likely that homeostasis is achieved as a result of some learned physiological corrective state.

The method of adaptation through ultrastable internal states in the homeostat has been implemented as a control mechanism in system design. Homeostats are useful as an internal regulation mechanism for both linear and non-linear systems, and has been implemented in the fields of engineering and computer science (Terry & Capehart 1968; Yoshida 2017). Di Paolo (2000) introduced the concept of homeostatic neurons integrated to an artificial neural network to control the locomotion of autonomous robots. Each homeostatic neuron is trained to facilitate the robot to adapt to severe distortions in its visual field (i.e. looking at the world upside down). The homeostatic neurons allow the robot to reorient its navigation behaviour, so that it may travel properly when given inverted visual input. A similar work by IIzuka and Di Paolo (2008) follows the method by implementing the homeostat as a plastic neural controller. The approach caters for situations where perturbations are