Towards A Framework To Make Robots Learn To Dance

by

Ibrahim S. Tholley

A Doctoral Thesis

Submitted in partial fulfillment of the requirements
for the award of

Doctor of Philosophy

Of

Loughborough University

14th May 2012

Loughborough University

CERTIFICATE OF ORIGINALITY

This is to certify that I am responsible for the work submitted in this thesis, that the original work is my own except as specified in acknowledgments or in footnotes, and that neither the thesis nor the original work contained therein has been submitted to this or any other institution for a degree.

…… (Signed)

…… (Date)

Dedicated to Mummy and Daddy

Abstract

A key motive of human-robot interaction is to make robots and humans interact through different aspects of the real world. As robots become more and more realistic in appearance, so has the desire for them to exhibit complex behaviours. A growing area of interest in terms of complex behaviour is robot dancing. Dance is an entertaining activity that is enjoyed either by being the performer or the spectator. Each dance contain fundamental features that make-up a dance. It is the curiosity for some researchers to model such an activity for robots to perform in human social environments. From current research, most dancing robots are pre-programmed with dance motions and few have the ability to generate their own dance or alter their movements according to human responses while dancing.

This thesis explores the question "*Can a robot learn to dance*?". A dancing framework is proposed to address this question. The Sarsa algorithm and the Softmax algorithm from traditional reinforcement learning form part of the dancing framework to enable a virtual robot learn and adapt to appropriate dance behaviours. The robot follows a progressive approach, utilising the knowledge obtained at each stage of its development to improve the dances that it generates.

The proposed framework addresses three stages of development of a robot's dance: learning ability; creative ability of dance motions, and adaptive ability to human preferences. Learning ability is the ability to make a robot gradually perform the desired dance behaviours. Creative ability is the idea of the robot generating its own dance motions, and structuring them into a dance. Adaptive ability is where the robot changes its dance in response to human feedback. A number of experiments have been conducted to explore these challenges, and verified that the quality of the robot dance can be improved through each stage of the robot's development.

Acknowledgement

There are many people that I must thank for achieving this thesis. I would first begin by sincerely thanking my supervisors, Dr Qinggang Meng and Professor Paul Chung for giving me the opportunity to do this PhD and the patience and guidance they provided me throughout the journey. There were many challenges that this PhD faced and both my supervisors were there at every hurdle to help me overcome them. This thesis would have been next to impossible to write without their help and guidance. Second, I am extremely grateful to the Computer Science Department at Loughborough University for taking me on as first an Undergraduate student and second as a research student, and funding this research. I give many thanks also to Professor Chris Hinde for providing interesting discussions on future possible directions of this research. Special thanks go to Dr Eran Edirisinghe, Ms Judith Poulton, Jo McOuat, Dr Iain Phillips and Patrick Holligan for the authorisation and assistance on all funding matters to go to conferences and present my publications. Also, I give thanks to Christine Bagley for being there to deal with all general matters, and Mr Samra, Kip Sahnsi and Richard Mee for providing IT support for my computing needs and the robot used in this research.

There were many others who helped me through this journey by simply being there when I needed research advice. In particular, I give thanks to Sazalinsyah Razali, and Afizan Azman for providing their own insight into research and academia. For those who were there to provide me with some emotional comfort, I give special thanks to all my friends. They played a vital role in giving me strength to continue the research, but the most important people of all who always held on strong for me and provided love and support to rescue and pick me up, every time I fell and keep me motivated are my family. I owe my deepest gratitude to my Mother, my Grandmother, my Father, my sister and my brother for without them, this journey would not have been necessary.

Table of Contents

List of Figures

List of Tables

List of Algorithms

Glossary of Terms

Throughout the content of this thesis, the following terms where used:

Action database (Movement database)
A repertoire of sequences and dance actions that have undergone the process of **pattern matching** or were generated by the robot. These sequences would typically be of varying lengths (number of dance motions) and chosen to generate a different dance.

Buffer (Buffering)
Defined as the process of caching *n* number of **dance motions** from human partners.

Dance action (Dance pattern)
Action selected by the robot based on the action-selection algorithm. There are three different types of dance actions. These are referred to as **dance motion**, **dance phrase** and **dance section** in this research.

Dance motion
Basic and shortest component of **dance actions**. Generated after **gestures** are performed on one time step or two time steps. It is a start gesture and an end gesture.

Dance phrase
A **dance action** generated after **dance motions** have been performed after two time steps. This is achieved by the robot performing one **dance motion** on one time step, followed by moving the same **dance motion** or another **dance motion** on the next time step.

Dance section
The longest **dance action**. Generated after **dance phrases** have been performed after two time steps. This is achieved by the robot performing one **dance phrase** on one time step, followed by moving the same **dance phrase** or another **dance phrase** on the next time step.

Formation
Defined as a "back-and-forth" motion. A human-like movement in which the same joints moved in one direction on one time step are moved again on the next time step in the opposite direction.

Gesture
Joints selected to move in one of their individual specific directions on one time step. The number of joints ranges from 1-15, but their respective directions vary.

Opposite

Defined as the human-like movement in which the same joints mirrored on the robot's body are moved at exactly the same time on one time step regardless of their directions.

Pattern matching

Defined as the process of comparing human preferences and extracting common (the same) combinations between human preferences.

Preference database

A repertoire of sequences and their respective feedback values extracted from the preferences of human partners.

Symmetry

Defined as the human-like movement in which the same joints mirrored on the robot's body are moved on two time steps regardless of their directions. This is achieved by the robot moving a set of joints on one side of the body on one time step and moving the opposite set of joints on the other side of the body on the next time step.

Chapter 1
Introduction

This Chapter introduces the idea of dancing robots as a research project and sets out the aims and objectives of the research presented in this thesis as well as the proposed stages of improving a robot dance as a robot dancing framework.

1.1. Overview

The research in Artificial Intelligence (AI) has enabled the development of machines and robotic systems that are more realistic in appearance in human-robot interaction. It is the aim of many researchers and scientists to continue to push the boundaries that separate man and machine, and artificial learning and art, by developing systems that demonstrate human ability in non-sentient agents in practical and social environments. Dance is one such application and is the topic of this thesis.

The idea of robots learning to dance opens up an area of research for social robots. Getting robots to learn to dance, would help to prepare them for more advanced tasks particularly to do with dynamic non-verbal communication, and help to establish new ways of communicating with them. Also, in terms of entertainment, human dance is already one established form of entertainment. Robot dancing can be then next best thing.

Dance in particular is a complex area of study; therefore, in the field of robotics, it is indeed a grand challenge. But applying it robots would not only provide advancements in robot dancing, but would also help to provide more understanding of dance itself.

What makes this area of study interesting is the fact that dancing robots demonstrate an activity that many humans gain satisfaction from. There is a

psychological stimulation that is obtained through robot dancing as it allows one to either become the entertainer or the spectator. By being the entertainer, the person creates dance motions which can be performed by a robot. By being the spectator, people can observe a dance and provide an assessment of it. Being the entertainer and the spectator are both human roles and it is this application to robots is what makes robot dancing socially interactive.

Great progress has been made in dancing robots. The state-of-the-art in robot dancing tells us that robots can dance and interact with human partners (Aucouturier et al., 2008a). The common approach to robot dancing is to program a robot with pre-programmed dance steps, or to make them imitate visual motions observed from human dancers, which are either randomly selected during the robots dance, or are choreographed accordingly for a particular music signal. These approaches rely on the trainer to initialise a robot with dance steps rather than for the robot to explore for itself the dance steps to perform. This therefore limits their human-like creative ability with dance motions and means that a dance would quickly become uninteresting and predictable. Furthermore, their dancing becomes less interactive as many do not take in feedback from human observers, and ultimately do not learn the appropriate dance motions to perform or adapt their dance to the preferences of human partners. This is therefore not good for social interaction between humans and robots.

Creativity is often expressed through the randomisation of actions. To achieve learning and adaptation, machine learning algorithms form part of the obvious solution. The common technologies employed to make robots learn and adapt their dance motions to the preferences of human partners, are *Interactive Evolutionary Computation (IEC)*

(Takagi, 2001) and *Interactive Reinforcement Learning (IRL)* (Thomaz et al., 2005), which are generally based on genetic algorithms (or neural networks) and reinforcement learning respectively. Other machine learning algorithms are also used such as Hidden Markov Models (HMM). However, their typical use has been for example, for predictive behaviour in mapping dance motions to the changing musical signal and not to achieve learning from human observers.

The general problem between human-robot interactions is the length of time required in order for the robot to learn to perform the desired behaviour and the fatigue that humans experience doing the interaction. Both *IEC* and *IRL* technologies have proven to be affective in demonstrating adaptive behaviour in robots, but little work has been done on them to solve the problems of human-robot interaction, particularly in robot dance, due to the factors of human interaction (Thomaz et al., 2005).

IEC has been the more popular choice to explore dance in robots, whereas *IRL* (and reinforcement learning in general) on the other hand has rarely been used in robot dancing. Many people may consider dance to be innately driven, and perhaps it is for this reason that biologically inspired algorithms (e.g. genetic algorithms) are used in robot dancing. Some, on the other hand would perceive dance to be a conscious activity in which case, it would be psychologically driven.

Reinforcement learning is considered as being inspired from psychology because of the way in which the algorithm learns. A reinforcement learning agent learns through interacting with the environment through trial-and-error and explores its possible actions in order to find gradually the desired behaviour. It can learn through certain (example-specific) solutions and uncertain solutions, and can change (or adapt) what it has learnt as

learning progresses. This is a psychological explanation of how biological systems learn, so reinforcement learning is considered as a psychologically inspired approach.

Similarly, dance can be seen to follow this same process. It involves interacting with the environment which may consist of other dancers, the music and spectators, and it undergoes changes as the music changes or based on the responses provided by others. Dance steps are explored through trial-and-error either mimicking the dance steps of other dancers or through personal exploration, and the dancing improves the more it is carried out. Dancers can therefore be described as reinforcement learners, and the use of reinforcement learning in robots would help to make their dancing more human-like. Making robots dance using *IRL* would make dancing robots dance in a more human-like manner.

There is currently no complete robot system that makes a robot express creativity, as well as learning from human partners and adaptation, to their preferences. This thesis therefore proposes a dancing framework that explores these three stages of development for robot dance. The ambition in this research is to develop a computational dance system that learns and perceptually improves in performance. This research proposes a learning computational framework based on reinforcement learning, to move us beyond these limitations. A virtual robot (Sony AIBO Dog) is programmed to simulate the development of an algorithm that meets these stages of development, in a way that humans may learn to dance.

1.2. Aims & Objectives

The principal aim of this thesis is to *develop an introductory robot dancing framework (robot dance system) that improves a robot's dance at each stage of its development*. This will be achieved by the following additional aims:

2nd Aim:- *Use and justify a suitable learning algorithm from the reinforcement learning approach*.

3rd Aim:- *Demonstrate a robot's improvement based on the proposed robot dancing framework*.

These aims were achieved through specific objectives, which were to:

1. *Identify and propose the stages in a robot's dance for a robot dancing framework based on current state-of-the-art that would improve its dancing (Chapter 2).*
2. *Integrate the tools that would enable a robot to improve its dancing and synchronise its dance motions to the beat of the musical signal (Chapter 3).*
3. *Implement each stage of a robot's improvement based on previously learnt behaviour (Chapters 4, 5 and 6).*
4. *Implement knowledge of human dance (Chapter 5).*
5. *Demonstrate the improvement of a robot dance based on human observation (Chapter 5) and human interaction (Chapter 6).*

The project to develop an introduction to a robot dancing framework is a result of achieving these aims and objectives, which as been achieved in this thesis.

1.3. Research Methodologies

There are at least, three stages of improvement that are critical to make dancing robots improve their dance: *learning*, in terms of learning what dance actions to perform to the music; *creativity*, in terms of generating new movements as opposed to utilising pre-programmed moves, so that robots form their own dance motions and their own dance, and structuring these movements into what can be defined as a "good dance"; and *adaptation*, whereby a dance is changed according to human feedback. The three stages of improvement are described further in the following sections.

1.3.1. Learning To The Beat

Human learning is based on how others judge their dancing and how successful the dancer's movements match the expression of the music. Human learning is continuous and this would be expected of robot dancers. Many current dancing robots do not learn how to dance and cannot respond to human feedback. They rely on the trainers to initialise desirable dance steps for them to perform. With such an approach, everytime a new dance step is required, the robot would have to be re-programmed. Like humans, robot dancers are expected to receive feedback of their dancing and learn from it. Therefore, the first challenge is to give a robot the ability to learn the fundamentals necessary for dance.

Reinforcement learning is the chosen approach to learning in this thesis for two reasons. The first reason is that it can be used to "evaluate" a learning agent's behaviour, as well as provide "instructions" as to what the learning agent should do more often than other actions. Dancing is an activity whereby guidance and feedback is required in order

for it to be accomplished. Therefore dance can be considered both *evaluative* and *instructional* as a dance agent must undergo criticism in order for it to improve its dancing.

Dancers improve their dance as they receive guidance from their trainers. This makes reinforcement learning a potential solution to learning to dance because dancing requires feedback concerning how good the movements are, and not necessarily what are the best moves. Indeed, in using reinforcement learning, many varying rewards can be given to the agent at anytime, to indicate subjective opinions, and the agent is left to make its own judgement on the feedback. This is what makes dancing evaluative. The robot will have to compare the feedback it gets to make valued judgements. What makes dance instructive, is the fact that, upon performing a movement, the dancer experiences a direct response, internally (e.g. pleasure) and/ or externally (e.g. from the audience) as feedback. In other words the dancer is told what moves to do and what moves not to do.

The second reason for choosing reinforcement learning is because with the help of this approach learning can occur through the learning agent's exploration of behaviours i.e. a trial-and-error approach. When dancers first encounter music, they are not informed of what dance steps to take and so they begin to dance by moving and observing their own movements and the movements of other dancers. Dancers create and experiment with new and old movements to the music throughout the dance. It is for these two reasons that reinforcement learning is selected to explore learning in dancing robots.

1.3.2. Generating Dance Motions

Human dancers have the ability to create their own movements by combining joint movements in different ways and different directions, and even when taught dance routines, humans combine different dance steps during their dance. This enables observers to comment on the dance. The pre-programmed or imitated dance steps programmed in current robots do not allow robots to explore or create movements for themselves. Dancing robots require the ability to form their own movements that appear not to be random or pre-sequenced, but autonomous, as pre-programmed routines and movements quickly become uninteresting and predictable.

Current robots are able to initially maintain human interest because their dance already consists of whole motions. That is, their movements are human-like in appearance and the transition between dance motions are logical and also human-like. This is often achieved by pre-programming a robot with primitive human-like behaviours (e.g. moving the head up and down as one dance motion) and connecting each dance motion to the next by terminating each dance motion to the same position or adjusting the dynamics of each motion to balance the robots posture using Zero Moment Point (ZMP) (Vukobratovic, 2004).

Robots can be provided with some initial dance behaviours. However, restricting robots with initial dance steps, limits the robots exploration of dance steps. The important thing is how dancing robots generate their own dance behaviours, and combine them to generate dance patterns to form different dances. Like humans, robot dancers are expected to explore their own movements and learn the fundamental features of dance motions so that the dancing is more autonomous and progressive in appearance.

Therefore, the second stage of development is to give a robot the ability to generate its own dance motions, which can be structured to form a *dance*.

1.3.3. Adapting To Human Preferences

Adaptive behaviour closely follows from learning. Whilst learning in this research focuses on identifying the generic individual desirable dance steps to perform, adaptation in this research is concerned with the ability to make robots change their dance according to human preferences. The adaptive ability of current robot dancers focuses on mimicking audio or visual rhythms, for example the dynamics of the music or the motions of human partners perceived in the environment. Little work has been done in robots being able to adapt their movements based on how human partners judge a robot's dance.

Human preferences are subjective in nature and sometimes conflict, and so the ability to alter a dance to accurately match the preferences of one or more trainers is what would make a robot dance adaptive. The challenge for dancing robots in achieving adaptation is to maintain human interest whilst satisfying and correctly evaluating their preferences. Maintaining human interest can be achieved by being creative with dance motions, but this does not achieve adaptation. For true adaptive behaviour, robot dancers are expected to explore further the human response they receive in order to learn human preferences and adapt to them as this is typically, how humans learn to dance.

As described above, this thesis proposes an introductory framework to robot dancing and aims to address all these three stages of development. These will be further explored in Chapters 4, 5 and 6 respectively.

1.4. Thesis Contributions

This thesis serves its purpose as an introductory framework to improve a robot's dance as one area of social interaction between humans and robots, by implementing learning ability in a robot, as well as the ability generate its own dance motions suitable for human judgement, and the ability to respond to human feedback. It is a stepping stone towards understanding the process of dance mechanics, combining abstract disciplines of art, rhythm and perception, necessary for dance in robots. This thesis addresses three stages of improving a robot's dance. These are: the ability to learn while synchronising dance motions to the music; the ability to generate dance motions autonomously without the need to re-program the robot each time; and the ability to generate an improved dance based on the preferences of human partners. These stages of development are linked to form the proposed dancing framework for robots. This is the first contribution of this thesis.

The second contribution is the learning and development of human-like attributes from dance studies in order to make the robot generate its own dance motions that are suitable for human judgement and help the robot in expressing creativity.

The third contribution is the use of the Sarsa algorithm and Softmax algorithm from reinforcement learning for learning and action-selection respectively in robot dance to help make the robot learn and adapt its dancing based on the direct interaction of human partners. Human partners are able to interact with a dancing robot and simply specify the parts they like and dislike unlike common approaches, which only take into consideration the parts the observer prefers.

The thesis presents some interesting results of dance that were not evident in the current state-of-the-art. It tells us that the increasing number of joints used in the robots dance, increases the quality of the dance up until a certain point, which suggests that the quality of dance may perhaps level off. Furthermore, the research confirms the theory that a robot dance can improve significantly when it incorporates the fundamental mechanics of dance and the preferences of human partners. The research shows that dance must have a structure; dance motions must be sequenced in order to develop a dance and joints must be moved in certain ways to each other to ensure aesthetic appeal.

1.5. Structure of Thesis

Following from this chapter, Chapter 2 is a literature survey of the current state-of-the-art in robot dancing. It focuses on the limitations of current robot dancing approaches and highlights the observable and measurable features of human dance that can be modelled and conceptualised for the robot dancing framework proposed in this thesis.

Chapter 3 provides an overview of the dancing framework and the components for creating the framework necessary for addressing the stages of its development. In particular, it describes the experimental platform; the robot; the beat detection algorithm used in this research and the learning algorithm and action-selection method used in this research.

Chapter 4 describes a simple approach to make the robot learn to be rhythmic to the musical beat with the direct use of traditional reinforcement learning. Initial experiments were conducted to determine what learning coefficient values can be used throughout the further experiments in this research.

In Chapter 5, the learning achieved in Chapter 4 is built upon. Here, the robot learns the desirable movements, by exploring the combination of primitive motions. These primitive motions are performed in different ways to generate autonomous behaviour and the robots own dance motions. These dance motions are then combined to generate different dance patterns and help structure the dance. The robot also learns the structure of dance. Empirical results are obtained to determine what improves the robots dance.

Chapter 6 builds further on the learning achieved in Chapters 4 and 5. In this chapter, the robot adapts its dance in response to the preferences of at most two observers whilst still maintaining creative, by generating new patterns with the preferences received from human partners. Empirical results are obtained to determine whether or not there is a significant difference in satisfaction when different preferences are combined in a robot dance.

Chapter 7 concludes this thesis and identifies future directions for the research.

Chapter 2
Literature Review - Robot Dancing

This chapter provides a literature survey on the current approaches that researchers have used to explore robot dance, focusing on three stages of development that are required for a dancing framework: learning appropriate behaviours, creating dance motions, and adapting to human feedback. Each of these is discussed based on the current state-of-the-art in robot dancing. This chapter also identifies the measurable fundamental attributes of human dance necessary for robot dance.

2.1. What Is Dance?

Dance is a creative and entertaining art form, driven by bodily movement and music, often expressed for social interaction, live performances, cultural practices and sport (for example gymnastics, skating, swimming and martial arts). Dance is a non-verbal communication human attribute that conveys the emotions of dancers. It is a discipline in itself that branches into many areas such as *style* (e.g. Ballet, Ballroom, Waltz, and Tango), *composition* (commonly known as dance choreography), *health* (e.g. for losing weight and exercise) and *therapy* (e.g. for boosting personal confidence). Even animals dance, for example, during mating season.

There are many definitions of dance, of which most still remain vague. Like all human behaviour, it is difficult to clearly define what is actually meant by *dance*, but it can be generally defined from two perspectives – as an art form in relation to music and conceptually as a series of connected patterns of movement. For example, as a general description, the Babylon English dictionary defines dance as a "move [done]

rhythmically to music, often following pre-composed steps and movements; skipping or bouncing about in a dance-like manner".

Van Camp (1981, p. 22) elaborates further by saying the components of human movement are:

> *"...formalized (e.g. by being stylized or performed in certain patterns), with such qualities as grace, elegance, and beauty, to the accompaniment of music or other rhythmic sounds ... [and it is] an art performed by individuals or groups of human beings, in which the human body is the instrument and movement is the medium. The movement is stylized, and the entire dance work is characterized by form".*

From the two definitions, it is possible to obtain an idea of what dance is, however, they do little to help us take practical steps towards teaching robots to dance. For example, using Van Camp's definition, which "formalized patterns" can be used to demonstrate dance? How can qualities such as "grace, beauty, and elegance" be measured? What is a "stylised movement" and how can we compute "form"? Is music necessary in dance? What is "rhythm"?

What appear to be the answers to these questions rely on human judgement and so therefore, the evaluation of whether robots can dance and how well they dance is determined by the results obtained from people. Robots do not understand these definitions and so are typically provided with well connected *dance motion*s in order to express these definitions.

A more conceptual definition on the other hand of dance is a definition proposed by McGreevy-Nichols *et al*. (2005) who say:

> *"Communication through movement should be the goal when building a dance. ... Movements are like words. You put words together to make sentences. In dance, these sentences are called dance phrases. Sentences are put together to make paragraphs in the same way that dance phrases are linked to make sections. Sections, when linked together, make a dance".*

Figure 2.1 shows a more conceptual understanding of this definition.

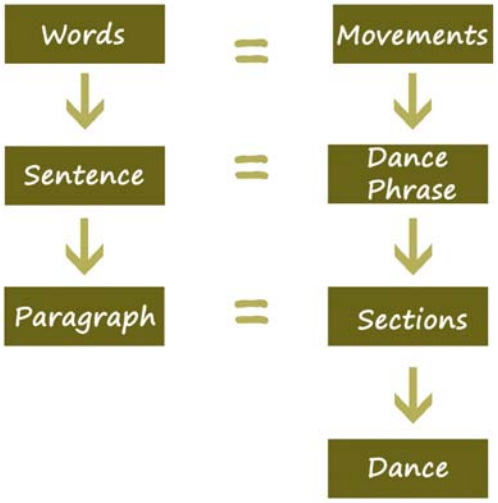


Figure 2.1 – The relationship between language and the structure of dance

In this definition, language is built up of words that are put into phrases that are joined to form meaningful sentences and paragraphs. Dance can similarly be understood as the structure of components that are connected together in human-like ways. The basic

component of dance might be called a *gesture* which is a simple single movement. Generally, in relation to movement, a gesture can be described as a whole motion such as a hand wave as a form of greeting, or a stand alone posture such as a "thumbs up" to indicate approval. In dance, a gesture is considered literal like sign language, making it possible to develop dance notations illustrated as written scripts (The Chicago School of Media Theory, 2004).

In this research, a gesture is interpreted as a stand alone posture of single or multiple joints that are moved in one direction (the same or different) at each moment. Gestures performed on two time steps make *dance motions*, which in turn build up to form sequences. In other words, a *dance motion* can be defined as a start gesture and an end gesture.

Dance motions are chained together to create dance phrases and dance sections, which together create the dance performance. So, combining the definitions above, a dance can be observed as one piece, built up step by step of sequences of dance motions in human-like transit, synchronised to the music (Aucouturier *et al*., 2008a). The next section goes through this in greater detail.

2.2. Current Progress in Robot Dancing

The first documented sign of a dancing robot goes as far back as 60AD created by a Greek engineer by the name of Hero (NewScientist, 2007). He was responsible for constructing a "three-wheeled cart" that could move around on its own accord, powered by "a falling weight that pulled on a string wrapped round the cart's axle" (NewScientist, 2007). This invention was said to be "dancing" because it moved around with no

assistance, except by the shifting weight attached to it. The robot did not move to music and the dancing only continued until the string ran out.

Much progress has been made since the work of Hero. Today, robots are able to dance by detecting the dynamics (intensity and tempo) of the music and mimicking the movements of human partners (for example in Jens *et al*. (2010)). Research into robot dancing has been strongly associated to entertainment and social interaction (Aucouturier *et al*., 2008a). However, the reasons behind robot dance vary between different researchers. For some, robot dance is used to allow robots to have the ability to detect rhythmic movement in humans (for example the work of Oliveira *et al*. (2008) and Tanaka & Suzuki (2004)). For others, it's to preserve and perform dance motions (for example, the work of Nakaoka *et al*. (2004)) so that it can be passed down to future generations; or to help researchers obtain better understanding of human movement (Shiratori *et al*., 2006), Shiratori & Ikeuchi (2008) and animal behaviour (Landgraf *et al*., 2010). Other reasons include the better understanding of human-robot interaction (Tanaka *et al*., 2006); and to become physical dance partners to humans (Jens *et al*. 2010).

The widely accepted approaches to robot dance include pre-programmed motions, which are either randomly selected or sequenced for a chosen music signal (for example, the work of Grunberg *et al*., (2009) and Santiago *et al*. (2011)); motion capture (for example in (Shiratori & Ikeuchi, 2008)) to extract key poses of human movement, which again can be either randomly selected or sequenced to music; dance with human partners (e.g. for example in Jens *et al*., (2010), Takeda *et al*., (2007), and Gentry & Murray-smith (2003)) and by receiving dance motions from stand alone computer applications,

which are controlled by human operators (e.g. the work of Shiratori *et al*., (2006) and Ellenberg *et al*., (2008)).

Whilst these ideas are all aspects of dance and have shown to be successful in robot dance, they limit the exploration of more advanced behaviours in robot dance, such as the ability for learning, generating dance actions and adaptation. For example, most robots do not have any learning ability and so can only demonstrate some form of creative behaviour in their dance, whilst others can adapt their dance motions to the music, but cannot adapt to human feedback. Some researchers suggest some form of framework for dancing robots (e.g. Oliveira *et al*., (2008); Aucouturier *et al*., (2008b) and Kim *et al*., (2007)) that combines aspects of the current approaches to robot dance. However, these frameworks do not address learning, creativity and adaptation in their development. Typically, these advanced behaviours are explored individually and, as of yet, there is no complete system for dancing robots that combines them. This is important as it would help to maintain human interest and interaction in social environments. The following sections describe how researchers have currently addressed these advanced behaviours in robot dance.

2.3. Learning in Robot Dance

The most recent advancement in robot dance is the use of motion caption technology. This works by using special cameras to capture motion data as moving regions of a human partner (Nakaoka *et al*., 2010). These motions are either stored in a database to be reused or are imitated in real time. The primary difficulty in using these data for robot

dance is that robots do not have the same degrees of freedom as humans. Thus, some processing is necessary to convert the human poses into usable robot poses.

The HRP humanoid robot (Nakaoka *et al*., 2004) was one robot which was used to accomplish the task of converting human motion capture data into a form that provided realistic movements in the robot, while maintaining balance. The robot learnt dance motions by observing movements in real time, performed by a human partner, using a method known as *Learning From Observation* (LFO) and a method known as *Zero Moment Point* (ZMP) for dynamic balance control (Nakaoka *et al*., 2004). Nakaoka, *et al*. (2005), on the other hand, analysed motion capture data to build a database of basic motions and instructions that were then used by the robot to perform dance moves.

Although robot dancing based on motion capture is representive of how humans may generally learn to dance from other human dancers, it is typically combined with learning algorithms which are either biologically inspired or probabilistic in nature. For example, the QRIO robot was used to explore a Recurrent Neural Network with Parametric Bias (RNNPB) to keep a record of and learn simple human gestures which were imitated dynamically (e.g. to different speeds) from human partners (Tanaka & Suzuki, 2004). Tanaka *et al*., (2005), on the other hand, used Bayesian inference to update its knowledge base of action sequences whilst in different interactive states. This was achieved by increasing the robot's belief (probabilities) if the robot had observed a response to its action sequences, and decreasing when the reaction from the human partner stopped.

As an alternative to motion capture (and pre-programmed movements), the MIURO robot was made to dance using a neural network based model known as

FitzHug-Nagumo (FHN) in order to generate Chaotic Itinerancy (CI), which is the idea of randomly generating seemingly autonomous behaviour, synchronised to music (Aucouturier *et al*., 2008b). Although a neural network was used to make the robot dance, the robot underwent no learning, but rather the input parameters of the network were altered and updated to encourage varying synchronised and spontaneous behaviours.

Some researchers employed predictive algorithms to achieve robot dancing. For example, the Ms DanceR robot was programmed with knowledge of ballroom dance steps to dance with a human partner, moving in directions estimated by the human partner, using Hidden Markov Models (HMM) (Takeda *et al*., 2006) and Neural Networks (Hirata *et al*., 2005).

The current state-of-the-art for learning in robot dancing, tells us that learning algorithms are used in two ways. Either, to make dancing robots autonomously respond to the dynamics (speed and intensity) of music or, be able to adjust their movements to imitate human partners and synchronise with perceived rhythmic motions. There is no attempt to use these learning algorithms to actually distinguish between "good" and "bad" dance steps judged by a human partner or the appropriate behaviours to perform. Instead, they are used to illustrate the idea of dance in robots and not make robots learn to dance. Nor can they learn the appropriate dance steps that are suitable to develop their own dance.

2.4. Generating Dance Motions in Robot Dance

The most common way to make robots dance is by pre-programming single dance motions that are either choreographed in sequences by a trainer to be performed to

predefined music, or selected at random and altered dynamically to match changing audio or visual rhythms in real time for example, in the work of Michalowski *et al*. (2007), Sinozaki *et al*. (2007), Oliveira *et al*. (2008) and Nakaoka *et al*. (2010), and in robots such as the hexapod dancing robot (Gizmodo, 2008); the Adam Frucci White Guy Dance Robot; the Hasbro iDog Robotic Puppy and Ampbot robots (TrendHunter, 2001).

The problem with pre-programming dance motions is that first of all, it limits the robots ability to generate its own motions and maintain human interest (Tanaka and Suzuki, 2004) and secondly, in order to change or improve a dance, the robot will have to be re-programmed each time, if different dance steps are required for a dance. This approach is neither desirable nor practical and therefore, autonomous behaviour is required of them.

There is no doubt that the above creative robots can be recognised as *dancing*, but the interest is finding the actual aesthetic characteristics that they display and the syntax that dance follows. The question in particular is *what are the characteristics of dance behind the different approaches to creativity in robot motions that make their movements become a dance*?

As described above, dance is a combination of imitation and patterns, consisting of structured rhythmic behaviours and related motions in succession (Michalowski *et al*., 2007). A dance can be detected if the movements are human-like (Sinozaki *et al*., 2007) or at least resemble the way in which humans move. Most non-salient dancing platforms (e.g. robots or animated characters) are humanoid or human-like in appearance (e.g. in the work of Oliveira *et al*. (2008); Tanaka *et al*. (2005), and Solis *et al*. (2005)) excluding

few (e.g. the work of Michalowski *et al*. (2007) and Sinozaki *et al* (2007)), but all demonstrate the aesthetic characteristics of dance motions.

Dance consists of dance motions that are made up of basic motion primitives synchronised to the music. Music itself has patterns and consists of a structure. The correct mapping of movements to music requires that the musical structure and patterns be perceived and therefore, visible in a dance (Nakaoka *et al*., 2007).

From the current state-of-the-art, dance steps containing selected primitive motions were pre-programmed in dance motions and consequently the robot dances appeared human-like and natural. But this made the robots less autonomous in their dancing. Therefore, the logical step to achieve autonomous and creative dance steps would be to first have a set of predefined gestures that can be combined and sequenced to form chains of dance sequences. Shinozoki, *et al*. (2007) explored this idea by developing a system that enabled robots to construct dance routines from basic pre-programmed dance steps. Similarly, Riley *et al*. (2000) used human examples to generate dance movements for the robot.

The idea of having single dance steps in robots that can be grouped together to generate varying sequences and dance patterns is a promising approach as it is a conceptual way to describe human dance. But the science of dance is more than just grouping dance motions in sequences. Dance motions themselves have a syntactic description (Erdem *et al*., 2008), which can be observed in dance in general. There are certain ways in which gestures are related and move together in order to achieve creativity in dance and enhance aesthetic pleasure. The following sections go into this in more detail.

2.4.1. The Structure of Dance

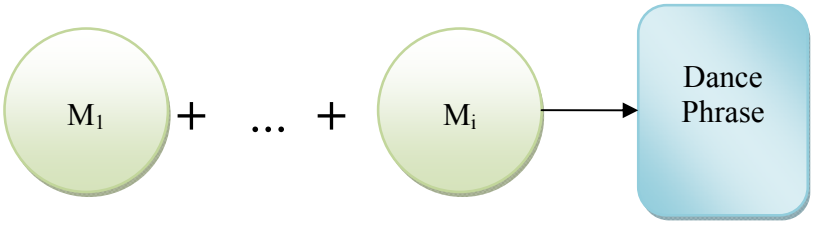
Like music, dance is a combination of related patterns consisting of structured rhythmic behaviours (Michalowski *et al*., 2007). These rhythmic behaviours are made up of primitive motions performed in real time to the music which are sequenced and combined to form a *dance.*

Recall, the definition proposed by McGreevy-Nichols *et al*. (2005) of dance was:

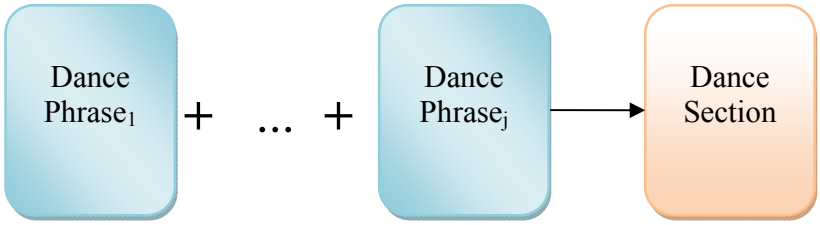
> *"Communication through movement should be the goal when building a dance. ... Movements are like words. You put words together to make sentences. In dance, these sentences are called dance phrases. Sentences are put together to make paragraphs in the same way that dance phrases are linked to make sections. Sections, when linked together, make a dance".*

Conceptually, this can be illustrated as shown below in Figure 2.2:

1. Movements (words) … produce Dance Phrases (sentences)



2. Dance Phrases (sentences) … produce Dance sections (i.e. parts of the dance)



3. Dance sections, linked together … produce a Dance!

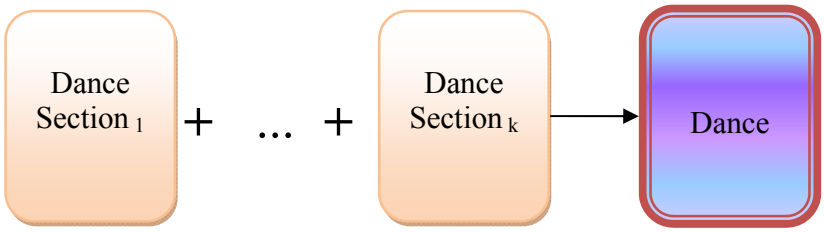


Figure 2.2 – Conceptual design structure of dance proposed by McGreevy-Nichols ***et al***. (2005)

Where i, j and k are the total number of *movements*, *phrases* and *sections* respectively. Movements (or *dance motions*) are the fundamental building blocks of dance, which could be a build up of *skilful* gestures in transition. It is the transition from whole dance motions to sequenced dance motions that gives a dance a structure. This would help to generate interlocking patterns of coordinated movements that are cyclic (repetitive) in behaviour as well as seemingly autonomous.

Many robots today are pre-programmed with whole dance motions so in order to achieve autonomous creativity and *skilful* gestures, dancing robots first need to develop their own dance motions.

2.4.2. The Relationships in Gestures

Whilst some researchers have limited robot dance to specific music and dance genres (e.g. the work of Hirata *et al*. (2005) and Yuuki *et al*. (2009)), the fundamentals of dance are much more generic and can be applied to different dance and music genres. In everyday human movement, there are specific ways in which joints move in order to achieve a task. Some examples include, walking, running, and jumping. All these skilful actions are executed in certain ways, using certain joints and muscular contractions. The joints and muscles used are what provide humans with the fundamentals to perform hierarchical tasks, but it is their relationships that determine their desirability and aesthetic appeal. These basic concepts are what form dance techniques and make the movements appear more *skilled*, because, in their use, they demonstrate an improved performance and can be used to produce more interesting motions.

Human dance is largely made up of perceptually meaningful and skilful gestures. From dance studies, there are at least three ways in which gestures are related for aesthetic appeal, necessary for any human dance and movement in general. They are called *Symmetry, Canon* and *Form*.

Symmetry is defined as the movement of opposite joints (Smith-Autard, 1988) in unison. These specific joints must be moved at the exact same time and can either be symmetric (i.e. moving in the same direction) or asymmetric (i.e. moving in opposite

directions) (Figure 2.3). Symmetric images are perceived as calmer, but not psychologically stimulating, whilst asymmetric images are more interesting and realistic (Humphrey, 1959).

Canon also requires the movement of opposite joints, but these can be moved one after the other (Smith-Autard, 1988). This helps to give the *sequential effect* in dance and looks progressive. Both Symmetry and Canon are largely due to the mechanics of the human body and understanding of body movements. Human bodies are very much symmetrical and everyday movements include the movements of opposite joints, both against each other and working with each other. This is known as *Opposition* and *Succession* (Figure 2.3) respectively in dance studies and can be performed in sequential succession (i.e. one after each other) or simultaneously (i.e. in unison), e.g. standing or walking.

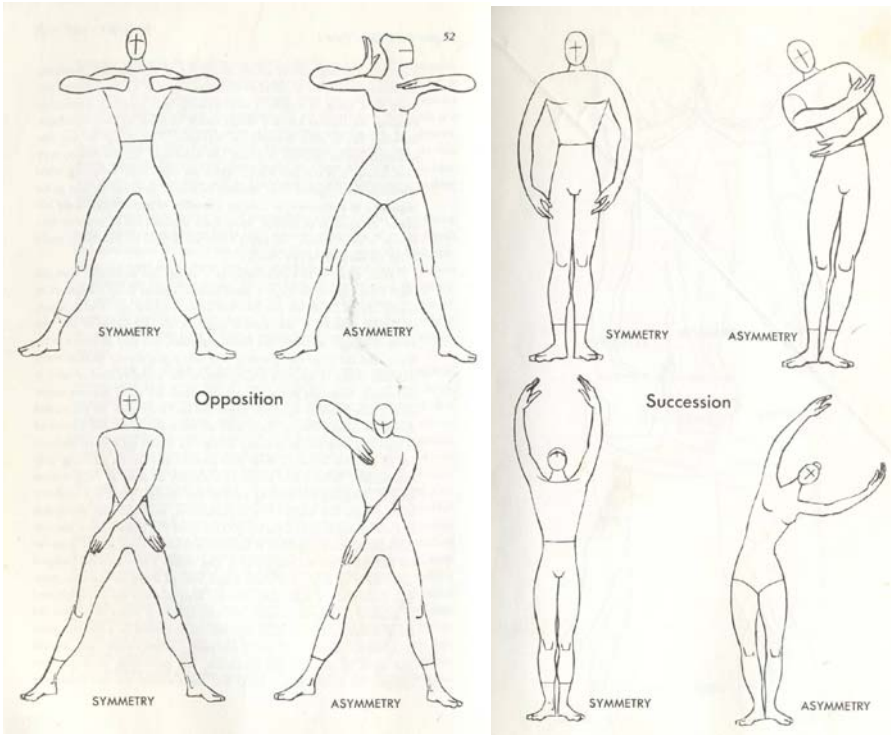


Figure 2.3 – Symmetrical and asymmetric movements that work with each other (opposition) and against each other (succession) (Humphrey, 1959)

Form is the skill that actually gives the dance a "shape" and sense of *style*, (Smith-Autard, 1988). It is what completes a dancers movements and a dance. There are many ways in which form can be achieved, but the most common and basic form is known as *Ternary Form* (Smith-Autard, 1988), which is whereby gestures follow a "back and forth" motion, for example, a set of joints starting from position *A,* which move to position *B* and then back to position *A*. This way of dancing is very common and one of the easier forms to learn for humans. It is a way of movement that is very pleasing to observe as perceptually, it gives the impression of completeness and ensures the satisfaction of an observer's expectations (Smith-Autard, 1988). Forms can be used for the development of dance sequences and dance routines as well as to create different effects with dance steps.

Human gestures are made up of a set of joints which are moved at chosen speeds and directions. These dynamics of movements combined with fundamental ways of relating gestures (i.e. Symmetry, Canon and Form) form basic motions, which are common amongst dancers and can be combined in different ways to give each dancer their own *style* of dancing (Erdem *et al*., 2008). Many robots already dance with given primitive (basic) motions containing already aspects of Symmetry, Canon and Form; for example, in the work of Sinozaki *et al*. (2007) and Shiratori *et al*. (2004). However, autonomous behaviour would be better achieved if these gestured relationships were explored and learnt while dancing, as this will engage human partners as the robot demonstrates improvement in its dancing. This would not only teach the robot what to do, but also, keep a record of how to do specific motions.

2.4.3. The Number Two

The definition proposed by McGreevy-Nichols *et al*. (2005) gives a practical solution to model the structure of dance, but it does very little to specifically quantify each part of the dance structure. For example, how many "movements" make up a dance phrase? How many dance phrases produce a dance section? At what point do dance sections become a dance?

In Western culture (and many others), there is a significant relationship between the number two and dancing. Establishing this simple relationship provides a powerful building block from which more complicated dance moves can be constructed. It is possible to establish this simple relationship by looking at the mechanisms of the human body of which the main expressive components of the body in dance typically include arms and legs. These individual components and their combinations, are based on the power of two, and can immediately be synchronised with rhythmic sound (e.g. most popular music and poetry), both written and oral (Hall, 2005).

Rhythm is the main component of rhythmic sound and dance. Rhythm is an intrinsically mathematical concept. This is because rhythm is measured and counted in distinct, repeating, segments. In speech and music, rhythm is measured by patterns of syllables and sections respectively. Rhythm is subdivided into units known as *measures*, and each measure contains a set number of beats (meter).

Arguably, the concept of the number two in rhythm was used before even music and dance, with its fundamental roots in spoken word and poetry. For example, as early as 200 B.C. a system of rhythm for speech had been devised based on two time units - *short* and *long* (Hall, 2005) which were used and combined in speech and poetry. This

same idea of long and short also became the basis for musical rhythm and dance and structure.

(1) The Number Two in Music

In music, the number two is very prominent, particularly in relation to the time signature of Western music. A time signature is denoted by two numbers written as a fraction (e.g. ¾). The top number in a time signature designates how many beats are used per measure. The bottom number indicates the type of note that is used to count the beat. For the top number, the number two is not so important here. It is in the bottom number that the number two is important. The longest note in musical notation (without adding additional modifiers) is known as the *whole* note. A whole note consists of four beats of equal length. From there, notes are defined that are each one-half the duration. For example, in a time signature of 4/4, there are four beats per measure (top number) and the quarter note (indicated by the bottom number 4 to mean ¼ of a whole note) gets a beat. In other words, each beat within a measure will last for a ¼ of a whole note. Table 2.1 below shows the breakdown of musical notes in a 4/4 time signature.

Note Type	Duration of note in beats	Bottom Number of Time Signature	Power of 2 For note duration
Whole Note	4	1	2^{2}
Half Note	2	2	2^{1}
Quarter Note	1	4	2^{0}
Eighth Note	½	8	2^{-1}
Sixteenth Note	¼	16	2^{-2}

Table 2.1 – The number two in a 4/4 time signature

As can be seen above, all of the fundamental musical notes are related to powers of two. In conventional music, it is not possible to have time signatures with numbers other than two (bottom number) since it determines the length of a beat with reference to the whole note. This, therefore, forms the first mathematical basis for understanding music and rhythm. Of course, this does not mean that all musical rhythms are based on the power of two, but it is certainly the case for most other common musical rhythms such as 2/4, 3/4 and 6/8 time signatures. As a result of this fact, many different types of music and dance have been created based on these musical rhythms. Table 2.2 below shows some common time signatures and the types of music and dance associated with them for which the number two in human movement is important.

Time signature	Music Style (genre)
4 / 4	Folk, blues, rock, jazz, pop, classical
3 / 4	Waltz time
2 / 4	Latin music, marching bands
6 / 8	Strauss Waltzes, Viennese Waltz
2 / 2	Latin samba music

Table 2.2 - Common time signatures and their respective music styles (Craig, 2005)

Table 2.2 reveals that many forms of music and dance are based on time signatures that are fundamentally based on the power of two. The significance of this relationship is that it demonstrates how music and dance are closely related and that it is possible to model this same behaviour on robots.

(2) The Number Two in Dance

In addition to the definition proposed by Van Camp (1981) of dance (Section 2.1), she also stated that dance is a "stylized movement synchronized with music and rhythm". The most significant element of the definition is that the movements are *synchronised* to the rhythm of the music. Since the mathematical basis for most popular music is the number two, it is no surprise that many dances (Table 2.2) also have the same basis as a result of the mechanics of the human body. It is the dual symmetry of the human body that allows dance movements to be tightly tied to the number two found in the rhythm of popular music.

The human body is composed of many pairs e.g. two feet, two arms, two legs. Human joints are typically also made to move in two directions (e.g. same and opposing directions), therefore, making it possible to perform movements such as nodding the head forwards and backwards and side to side; and lifting the arms and legs up then down. Some joints can even move in two ways for example, the forearm can be extended and bent. It is for this reason that logically, popular dance movements can be broken down into movements that alternate between two body positions. Mathematically, this neatly corresponds and synchronises to all meters of rhythm that are based on the number two. Dance instructors take advantage of this simple fact when teaching children how to dance. Table 2.3 shows some common body movements that are taught to children as part of music and dance education.

Movement	Description
The Rocker	The child sits with legs crossed and rocks back and forth.
The Squirrel	The child sits and makes forward and backward circles with his/her arms.
The Owl	The child rocks his/her head forward and back or side to side.
The Starfish	The child lies on the floor and alternates lifting his/her arms and legs up and down.

Table 2.3 - Children's Dance Moves (Kramer, 2010) based on the number two

In much the same way there is a relationship between human gestures using the number two, that is also applied to dancing partners. For example, Table 2.4 shows four ways in which symmetry can be established between dancing pairs.

Symmetry	Description
Translation	Movements are done with all dancers facing the same direction. E.g. if the leader raises his/her right hand the dancers raise his right hand.
Reflection	Movements are mirrored with the dancers mimicking the leader who faces them. If the leader raises his right hand, the dancers raise his/her left hand.
Rotational	The leader faces the dancers. If the leader raises his/her right hand, the dancers raise their right hand.
Glide	Movements are done with all dancers facing the same direction. If the leader raises his right hand the dancers raise his left hand.

Table 2.4 - Symmetrical movements in paired dancing (Schaffer and Stern, 2009)

Again, the combinations of the dance steps, are all based on the number two, and can be combined in multiple combinations that can be synchronised to music and to other

dancers. It is therefore the mechanics of the human body and interaction with other dancers that is used to achieve synchrony with the music.

Typically, many dancing robots are humanoid in appearance and therefore already have these human-like attributes, making their dancing appear natural and human-like, but not all robots are humanoid in appearance and so knowledge of the syntax of dance would benefit human-robot interaction in dance in general, and give dancing robots more autonomous and creative dance behaviours. This can be achieved by making them learn the fundamental ways in which the body and bodily components relate to each other, and use this knowledge to generate dance patterns and sequences that follow the musical rhythm.

2.5. Adaptation in Robot Dance

Adaptation is the process by which a learning agent (e.g. humans) change behaviours through interacting with the environment. Most dancing robots in the current state-of-the-art show adaptation by directly responding to the audio or visual changes in the environment, such as changes in the dynamics of the music (e.g. the work of Shiratori *et al*. (2006) and Solis *et al*. (2005)) or in the rhythmic motions detected of human dancers (e.g. the work of Riley *et al*. (2000) and Tanaka *et al*. (2006)). The primary limitation in these dancing robots is two-fold. Firstly, the robots do not learn from their adapted behaviours, but instead, just mimic changing rhythms. Secondly, these robots cannot receive feedback from human partners on their dancing. Thus, changes in the robots selection of dance motions would have to undergo re-programming. To achieve real time adaptation, without the need to re-program the robot, it is clear that a logical approach to

this would require the robot to have real time interaction with human partners, so that the desired and undesired preferences can be explored and performed. The advantage of this is that the observers can influence the way a robot improves its dancing without having to program the robot each time or even have knowledge of dance or robots. Instead the observer only needs to express their likes and dislikes, to guide the robot's dancing. The most common way in which this has been achieved is though the exploration of two main technologies known as *Interactive Evolutionary Computation* (IEC) and *Interactive Reinforcement Learning* (IRL).

2.5.1. Interactive Evolutionary Computation

Interactive evolutionary computation (IEC) is a biologically inspired interactive learning approach based on evolutionary computations (Suga *et al*., 2005) and works by using genetic algorithms (Graf, 1995) or neural networks (Dozier, 2001) and the evaluation is determined by human feedback. During the process, different variations of generated behaviour are tested to see if a solution closer to the trainer's preferences has been found. Randomness is introduced into the process by allowing a certain percent of generations to occur so that no part of the search space is excluded for exploration. This approach has proved satisfactory in different areas such as 3D modelling (Nishino *et al*., 2001), image processing (Poli and Cagnoni, 1997) and robot dancing (Vircikova and Sincak, 2010).

The advantage of IEC is that it replaces a pre-programmed fitness operation with a human agent. However, the disadvantage is that first, a person cannot make decisions as fast as a computer. Therefore, the use of a human agent reduces the progress of convergence (reaching a "best" dance) and greatly limits the number of possibilities that

can be tested. Second, unlike machines, humans quickly get tired and bored, making it more and more difficult for the human agent to make decisions regarding the fitness of the current generation, after hundreds of generations of samples have been presented (Takagi, 2001).

For many researchers who have explored the idea of human feedback on robotic dance systems, the common experimenting approach is to limit the feedback of dance partners to one dance partner at a time, and limiting human feedback to only positive rewards. For example, in the work of Dozier (2001), the robot interacted with one observer for a limited time (4 seconds). Vircikova and Sincak (2010) implemented an interactive system, in which human agents observed the dance of seven humanoid robots, each with varying dance moves. Each observer was required to rate the overall dance quality from 1 – 5. The work in both papers showed successful results and suggested that the robots could adapt their movements to a human observer, however the results were all based on the feedback of a single observer and participants were limited to give one type of feedback, to express one type of preference. Little was shown to determine the result of the robot responding to preferred and non-prefered dance behaviours. This however, has been demonstrated using *Interactive Reinforcement Learning (IRL)*.

2.5.2. Interactive Reinforcement Learning

Interactive reinforcement learning (IRL) (Thomaz *et al*., 2005) is a psychologically inspired interactive learning approach, based on traditional reinforcement learning algorithms and like IEC, the reward signals are replaced by a human agent as opposed to a pre-programmed model. The human agent can interact with the learning agent (e.g. a robot) at anytime and vary the rewards as they wish, during the robots learning process.

Like IEC, IRL has shown to be successful in areas that require adaptive interactive learning (e.g. the work of Leopold *et al*. (2008) and Liu & Su (2004)), but little has been used in robot dance. Of particular interest is the work by Thomaz and Breazeal (2007). The authors implemented a reinforcement learning approach which allowed humans to interact with a real robot and a computer game. They used the idea of positive and negative rewards in their work to guide learning in two ways: 1) learning by refining a sequence of actions towards a goal; 2) learning as a means to encourage exploration. In their implementation, after the execution of each action, the robot system had a "small delay to allow for human reward".

Thomaz and Breazeal (2007) limited human input to scalar values in the scale of +1 to -1 for preferred and non-preferred actions respectively. Their results showed that participants gave more positive rewards than negative rewards. This same approach of positive and negative rewards for IRL had been explored by others such as Leopold *et al*. (2008); Austermann & Yamada (2008) and Dozier (2001). The use of positive and negative rewards is a simple way for a robot to gain knowledge of the preferred and unwanted dance steps of a dance. However, this has a number of problems. In particular, if the rewards are not consistent then contradictions can occur in the observer's feedback

(Thomaz *et al*., 2005). For example, a "good" dance move in one part of the robot's dance maybe a "bad" dance move in another, which could cause contradiction in the robot's understanding of the true reward. Also, as rewards are given in real time, the wrong movements may be rewarded, in which case, the robot may learn to do the wrong actions. Therefore, an effective way of processing such rewards would have to be adopted. Nevertheless, compared to IEC, the ability to do this in IRL makes it a better approach for representing the likes and dislikes of observers.

2.6. Summary

This chapter summarised the important developments that have been made in the quest to first enable robots to dance and then instil in them the capacity to create autonomous adaptive behaviour. Although progress has been made in this area of robot dancing, the fundamental limitation is that dancing robots dance as a result of pre-programmed human-like examples and the quality of their dancing do not improve with this approach.

Approaches to robot dance have ranged from choreographed dance patterns to mimicking visual motions based on motion capture and predicting dance steps based on human dance partners. Whilst, these approaches have successfully established dance in robots, the creativity and autonomous behaviour of dancing robots is limited because they typically cannot generate their own dance motions and cannot adapt their dance motions to human preferences. For example, most robots do not have any learning ability and so can only demonstrate some form of autonomous behaviour in their dance, whilst others can adapt their dance motions to the music, but cannot adapt to human feedback.

In order to improve a robot dance, it is necessary that dancing robots posses mechanisms that encourage them to learn the appropriate dance behaviours to perform; adapt to the preferences of human partners, and be creative with their dance motions, in order to maintain human interest. These stages in the robots development can be achieved by highlighting the attributes of dance that give dance its aesthetic appeal. This is evident in the ways humans move.

In everyday human movement, apart from the relationship between gestures, it is also clear that dance motions have a numerical commonality as part of their definition. The description of skilful dance motions suggests that they occur in two time steps or factors of two. Using the definition from dance literature, both Canon and Form require two time steps in order to be achieved, whereas Symmetry requires one time step. For this reason, the number two is a good indication of a quantifiable value for each part of the dance structure. To achieve seemingly natural, human-like movement in dancing robots, many researchers automatically provide this knowledge to them as opposed to making them learn them.

The number two is necessary for aesthetic beauty as perceptually this is what humans are familiar with in everyday movement e.g. the mechanics of the human body. Therefore, naturally, if we can break down dance motions into gestures of two time steps then this enables skilled motions (whole motions that contain Canon, Form and Symmetry) to be performed as well. Robots can then form their own primitive dance motions and learn which ones are more desirable through their interaction with human partners.

A learning approach of particular interest in this thesis that can be used to enable human subjects to interact with a robot is Interactive Reinforcement Learning (IRL). This is a promising approach to interactive learning because it is based on an adaptive learning algorithm (reinforcement learning) that makes the learning agent adapt its behaviour as the environment changes, and is capable of evaluating both preferred and non-preferred preferences.

Chapter 3
The Robot Dancing Framework

This chapter provides an overview of the proposed framework and in particular, the robot and the beat detection algorithm that was used in this research and the learning approach.

3.1. General Overview

The fundamental problem with the way in which current robots dance is that there is no system that gives them the ability to learn appropriate dance steps, create their own movements and adapt to human preferences. The current state-of-the-art can do one or the other, but not all. Furthermore, existing implementations require that robots be re-programmed each time a new dance is required and therefore the trainers need to have knowledge of how to program the robots. In this thesis, a robot dancing framework is presented that addresses these key points. The fundamentals explored in this dancing model include:

- Real time analysis and extraction of key music features;
- Development and learning of a collection (repertoire) of desirable dance steps;
- Creative dance steps and sequences that are synchronised to the music;
- Adaptation of dance steps and arrangement of dance steps into different combinations based on human feedback.

Figure 3.1 shows how these fundamentals are combined in this thesis to form a dancing framework.

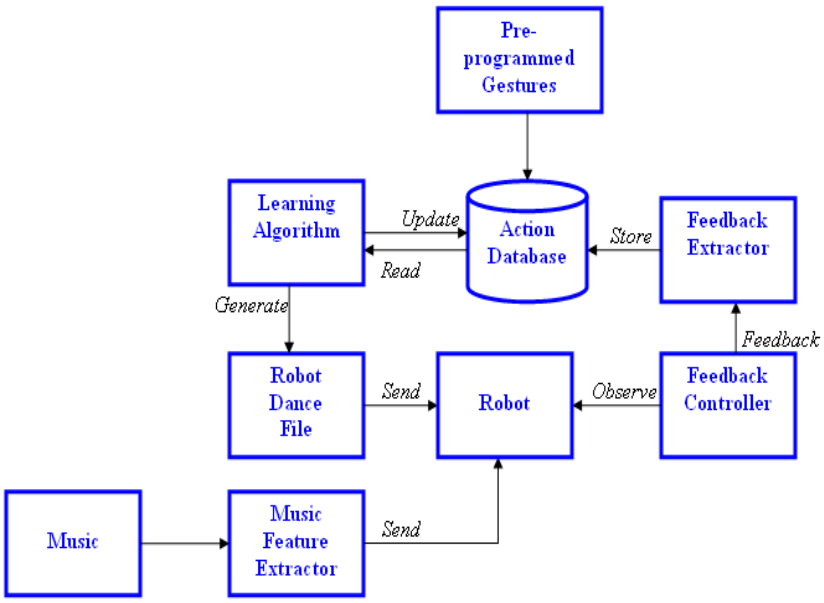


Figure 3.1 – Integrated framework for dancing robots

In Figure 3.1, *Pre-programmed gestures* refers to the initial simple movements available to the robot, which are joint and direction specific e.g. raising an arm, which is different to dropping the arm, or the head rotating to the left, different to its rotation to the right. These gestures are then explored to form primitive and sequenced dance steps (e.g. a dance action that moves the head up and down), which are then stored as a repertoire in the *Action Database* as actions. These are then evaluated using the Sarsa learning algorithm based on rewards defined by the trainer. The actions are then selected by the Softmax action-selection method for the robot to perform.

The *Music* is analysed and the musical beat is extracted in real time, while the learning algorithm is running and the robot performs the chosen actions in time with the beats detected. The music is not stored in the robot's repertoire. This means that the robot can learn the appropriate behaviours whilst different music is being played. The *Feedback Controller* (e.g. the trainer) then observes the robot's movements and provides

feedback, and the *Feedback Extractor* receives the feedback and stores these in the *Action Database* for the learning of which actions are preferred and non-preferred by the *Feedback Controller*. The result is a file containing a sequential list of actions, sent to the *Robot* for it to perform.

The focus of this research is to develop an adaptive dancing model that learns in real time in a real environment (i.e. with real music and real people), without the need to re-program the robot. Like humans, the learning for dancing robots must be achieved in stages of development. In this thesis, the robot gradually improves the quality of its dance as it learns the desired behaviours which are 1) learning to perform movements to the beat, 2) combining basic movements to create its own sequences and 3) learning from the preferences of human partners. These will be explored respectively in Chapters 4, 5 and 6.

The implementation and experiment of the robot's stages of development were conducted using a virtual environment. The robotic test bed was the Sony AIBO dog and its dancing was demonstrated using a simulation package known as Webots (Cyberbotics, 2011). The Sarsa algorithm and Softmax method were implemented for learning and action-selection respectively. The output of the system was the complete dance sequence generated. This was read directly by the robot and the actions were performed on each beat detected by the beat detection algorithm. The beat was extracted in real time, using the BPM Detection Library (adionSoft, 2007) which was integrated into the Webots environment for real time processing. All integrations and processing (i.e. the learning algorithm; the action-selection algorithm; the Webots environment and the beat detection

algorithm) were programmed in C++. The following sections will go into more detail of these components of the dancing framework.

3.2. The Robotic Test Bed

3.2.1. Sony AIBO Robotic Dog

The Sony AIBO dog is a small robotic dog in appearance (Figure 3.2), developed by Sony, whose name is an abbreviation of *Artificial Intelligence Robot* (AIBO). Different models have been developed. The model used in this research was the ERS-7 model. The robot had previously been successfully programmed by Sony to demonstrate realistic dog behaviours as well as emotions using gestured movements and coloured lights. It has the ability to recognise faces and voices using built-in pattern matching and detection algorithms and currently posses predefined dance routines that are performed to predefined music.

Figure 3.2 – Sony AIBO Real Robot Dog (ERS-7 model)

The Sony AIBO dogs (ERS-Model), are equipped with a 576 MHz on board processor with 64 MB RAM. The operating system is Aperios, developed by Sony, built

on the Open-R architecture (Operating System, 2009) and it has a total of 20 degrees of freedom (DOF). These consist of head: 6 DOFs (neck: 3, ears: 2, and mouth: 1), legs: 12 DOFs (3 for each leg), and tail: 2 DOFs. All joints are used in this research except those pertaining to the ears, mouth and the tail as they were not considered to be significant joints necessary for dance.

The AIBO dogs are a popular choice for robotic research because they are relatively inexpensive; have many degrees of freedom and already possess a variety of useful capabilities such as distance and touch sensors, a camera (for image recognition), and microphones for audio input. The Sony AIBO dogs are sufficient enough to demonstrate the stages of dance development explored in this thesis.

3.2.2. Webots

In order to simulate the robots dancing, the Webots software simulator developed by Cyberbotics (Cyberbotics, 2011) was used. Webots is a development software environment that comes with many industrial robots including the Sony AIBO dog (Figure 3.3) and the robots can be programmed using popular languages such as C++, Matlab and Java.

Figure 3.3 – Image of Sony AIBO Dog used (ERS-7 model) in Webots

The Webots simulation package was used to demonstrate the robot's dancing to human partners as well as receive human preferences. The learning algorithm and the action-selection algorithm were not integrated with the Webots simulator, but instead were processed separately after the robot had finished dancing. In other words, the learning and action-selection did not take place while the robot was dancing. Each dance that human partners observed was pre-generated prior to their feedback. This was because processing the learning and action-selection of human feedback while the robot was dancing was a computational expense and resulted in huge delays in synchronising the robot's movements to the music, which in turn affected the appearance of the robot's dance and mixed up the rewards given (i.e. the wrong rewards were given to the wrong behaviours). This also conflicted with the music, causing it to stop and repeat itself at different times. For this reason, the learning algorithm and action-selection algorithm were performed separately from the Webots environment and a file was generated of an entire dance that was directly read by the robot. This not only increased the learning speed, but also ensured that the robot's dance could be observed and human trainer preferences could be recorded, without adding more work to the CPU.

3.3. Synchronising To The Music

In order to detect the beat of the music in real time, the BPM Detection Library (adionSoft, 2007) was used and integrated with the developed system. This enabled the robot to move its joints at the same time to the beats of the music as detected by the algorithm. The BPM Detection Library is an open-source real time rhythm detection program, which can be used to detect the rhythm (beat, tempo and time signature) of music either as real time input from a microphone or decompressed music files on a computer stored in various formats, for example, .mp3 or .wav files. The program is particularly suited for music genres which have steady beats, which is typical of Western music. To perform real time dancing to the music, the robot executed each dance step from the generated dance sequence on each detected beat. When no beats were detected (e.g. the music stopped playing) then the robot would stop dancing at that point.

3.4. Learning Algorithm

The learning approach taken in this thesis is from reinforcement learning. Reinforcement learning is an adaptive computational approach in the field of machine learning which works by applying the same principles of reinforcement in psychology, which are used by animals and humans to learn. The basis of reinforcement learning is a direct feedback system, whereby the learner is either rewarded or punished based on the actions that are performed. Positive reinforcements encourage behaviour to be reproduced, while negative reinforcements discourage behaviour from being repeated (Dayan, 2005).

Research in machine learning use *Supervised Learning* and *Unsupervised Learning*. Supervised Learning is learning from examples provided by the environment

(Sutton and Barto, 1998). The learning approach is example-specific, only learning what it has been instructed to learn in specific environments. The learning agent is specifically told what to do for every example in the training set, and then it simply replicates the correct answers as defined by the trainer. Unsupervised Learning, on the other hand, makes use of unlabelled data. The learning approach is required to categorise and cluster the data as a way of labelling the data. Reinforcement Learning lies somewhere between Supervised Learning and Unsupervised Learning, which works with some level of supervision from the trainer, but the level of supervision is less than in Supervised Learning. That is, the agent is only instructed of the correct behaviours once it has performed actions.

Reinforcement learning has the advantage of applying what it has learnt in related environments, as well as the ability to modify what it has learnt, which is what makes it suitable for adaptive behaviour. A reinforcement learning agent learns through trial-and-error from its own experience. The learning framework works by making the agent adapt through directly interacting with its environment in order for it to learn efficiently (Sutton and Barto, 1998).

3.4.1. How Reinforcement Learning Works

Reinforcement learning works by "defining the interaction of an agent and its environment in terms of states, actions, and rewards" (Sutton and Barto, 1998). It provides a mapping of perceived states to actions known as *policy*. A policy can be defined as "what actions to take in what states" (Sutton and Barto, 1998). The agent keeps a record of the actions performed in each state and associates the link with a *value*

function, used for action selection in future states. The *value function* for actions performed in different states (or state-action pairs) is determined by the rewards the agent receives in doing actions in states. The value function estimates the value of taking an action in a state. It is the expected return (total rewards received over time) of doing actions in different situations (states) in the future. It is therefore the aim of the learning agent to maximise some measure of its future performance (Sutton and Barto, 1998). Actions with greater returns increase the value of state-action pairs, which in turn, encourage the performance of more desirable behaviours.

For reinforcement learning a balance must be made by an agent between choosing more desirable actions and choosing unexplored actions, just in case there are "better" or other desirable actions that have not been discovered yet. The agent's interaction with the environment is the sequential process of selecting actions modelled as a Markov Decision Process (MDP). This works by dividing the learning problem into four elements – a set of states; a set of actions; state transition probabilities and a set of rewards that are associated to an action performed in a state.

Reinforcement learning can take place in two modes – *offline learning* whereby the agent learns by interacting with a simulation (model) of the environment, and *online learning* where the interaction is with the real environment. In both approaches, learning is achieved, but their actual use is problem dependent, for example, learning problems that are too big, complex or dangerous to model in a real environment may be best suited for offline learning as opposed to online learning. The problem of making dancing robots learn to dance is a complex problem and although it is possible for them to learn through online learning, due to the computational costs on the real robot (e.g. the amount of work

that the motors have to undergo or the dangerous positions that joints can be in), it is better for them to learn through offline learning e.g. using a virtual robot that responds to music being played and human feedback in real time.

Typically, the standard application of reinforcement learning is for the learning agent (e.g. a robot) to explore different states and perform behaviours that put it in another (or the same) state. Often, a weighted positive or negative reinforcement (reward/ punishment) is recorded for the state the learning agent ends up in or the actions it takes. This cycle continues until the agent has reached the desired goal. The value function is applied to each state (or state-action pair) for effective action selection and then the cycle is repeated. That is, the value function tells the algorithm what state to go to or action to perform in the next time step that would put the robot in a better situation (i.e. maximise the total expected rewards). After a sufficient number of trials have been run, each state (or state-action pair) in the MDP will have a probability that will drive the agent towards better and better solutions. Once an acceptable performance level is found, the learning algorithm will still continue to run as before but without updating any of the probabilities (Gosavi, 2003).

3.4.2. Comparison Between Q-Learning and Sarsa

As a solution to learning in dancing robots, two of the most popular algorithms, known as Q-Learning and Sarsa, from traditional reinforcement learning are considered. Both algorithms are similar in their procedure and in the updates of value functions of state-action pairs. However, the fundamental difference between them is how the decision policy (the mapping of actions to states) is applied.

As described above, a policy is a rule that determines the choice at each situation. In Q-Learning, a particular policy is chosen (exploited) and applied throughout the learning process. In Sarsa, the policy is evaluated and updated after each action selection. Both algorithms have two learning coefficients referred to as the *learning rate* (α) and *discount factor* (γ). The learning rate is used to determine if new information should be considered (when $0 < \alpha \leq 1$) or not (when $\alpha = 0$). The discount factor weights the value of future rewards after taking an action in the next state. For example, if $\gamma = 0$, then future state-action pairs are not considered, whilst $0 < \gamma < 1$ would mean that future state-action pairs are considered. This is particularly important during the robots transition from one state-action pair to another as it determines to what degree the next state-action pair influences the robot's learning.

Although both Q-Learning and Sarsa have been successfully applied to learning problems, Q-Learning has been widely used in the field of robotics "due to its algorithmic simplicity" (Martinson *et al*., 2002) and the speed at which it finds the optimum policy (Poliscuk 2002). Sarsa on the other hand is more consistent in choosing policies in the learning process than that of Q-Learning, which suggests that it is more suited in areas where consistency and dynamic learning are more important than the speed of learning (Poliscuk, 2002 and Srinivasan, 2005).

Sarsa has also been found to outperform Q-Learning in environments that require learning without previous exploration. For example, in what is called the *cliff-walking task* (Sutton and Barto, 1998), Q-Learning and Sarsa were compared to determine their difference in learning. The task was for each algorithm to reach the goal state, without entering detrimental states. The end result was that, although Q-Learning found the

optimum (shortest) path to the goal state from the initial state, it was more likely to walk the learning agent "off the cliff" (i.e. enter the detrimental states) than Sarsa. The Sarsa algorithm reached the goal state by learning the longer but safer path. This is because it is possible for Q-Learning to exploit sub-optimal actions during the learning process, whereas Sarsa, would evaluate each choice as it is made before taking the next action, making it less likely to walk off the cliff. This suggests that Sarsa is better in situations where there is no prior knowledge (Poliscuk, 2002). This makes Sarsa a better choice for environments that cannot be adequately pre-modelled and simulated. Furthermore, it is suggested that real-world biological systems (i.e. humans and animals) are more likely to learn in a way that is similar to Sarsa rather than Q-Learning (Morris *et al*., 2006).

Learning to dance can be Q-Learning-like or Sarsa-like. For example, in break dancing competitions dancers often perform the same signature moves throughout their dancing to win points, or a belly dancer may continue to drop her hips up and down to please the crowd. These actions are exploited behaviours and so would follow a Q-Learning-like approach. On the other hand a social dancer at a celebration may simply copy the dance moves from other dancers and observe the feedback from the audience. This is more like a Sarsa approach to dance. The difference in these two types of dancing is that in the former the dancer already has knowledge of the "better" dance move (and the response), whereas in the latter, the dancer may not have any prior knowledge of the "correct" dance steps and, therefore, be ready to adapt its dance behaviours according to the feedback. Therefore, a Sarsa implementation would be a logical choice for a dancing robot in terms of learning and adaptation.

A Q-Learning approach to robot dancing would suggest that there are "optimal" ("best") dance steps in which dancing robots must select during their dance, and as a result, is more likely to make them learn the wrong dance steps, because it would be possible for the algorithm to consider sub-optimal dance actions as the optimal dance behaviours. A Sarsa approach would ensure that dancing robots evaluate the feedback they receive from their dance steps and so, therefore, slowly, but carefully learn the appropriate dance steps. Algorithm 3.1 shows the traditional Sarsa algorithm.

Algorithm 3.1: Traditional Sarsa Algorithm (Sutton and Barto, 1998)

1. Initialise parameters
2. Repeat (for each episode)
3. Initialise the starting state (s_t)
4. Choose action (a_t) from state (s_t) using derived policy from Q
5. Repeat (for each episode)
6. Perform action (a_t), receive reward (r)
7. Agent is in next state (s_{t+1})
8. Choose action (a_{t+1}) from s_{t+1} using policy from Q
9. Update value function, current state and action

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha[r + \gamma Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)]$$
$$s_t \leftarrow s_{t+1}$$
$$a_t \leftarrow a_{t+1}$$

10. Until terminal state is reached

In this algorithm, the *parameters* (line 1) would be the value function (otherwise known as the Q-factor, $Q(s,a)$); the set of states ($s \in S$); the set of actions ($a \in A$); the reward function (r) for transiting from one state to another. α is the learning rate; and γ is the discount factor and *t* is the *time step* moment for being in a state (s) or performing an action (a). An *episode* (line 2) is a sequence of actions in the set A that leads the agent from an initial state to the goal state.

The algorithm iteratively explores a possible action in a state until a terminal state (i.e. the goal state) is reached. An action (a) is selected using an action-selection method, of which the most common implementation is to use an approach that always embody exploitation (e.g. an $\varepsilon-$greedy approach). That is, the idea of choosing an action believed to be the "best" most of the time, and on occasion, choosing an action at random as part of the exploration.

The actual processing of the reward signal is carried out in line 9 where the value function ($Q(s_t,a_t)$) for the current state (s_t) and action (a_t) undergoes the update by taking into consideration the value function ($Q(s_{t+1},a_{t+1})$) of the next action (a_{t+1}) performed in the next state (s_{t+1}). The reinforcement learning agent therefore learns which actions it should do in different states that maximise the expected cumulated rewards.

The crucial difference between the traditional Sarsa algorithm and the traditional Q-Learning algorithm is in the update of the value function (line 9). The update equations for traditional Sarsa (equation 3.1) and traditional Q-Learning (equation 3.2) are shown below.

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha[r + \gamma Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)] \quad (3.1)$$

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha[r + \gamma \max Q(s_{t+1}, a) - Q(s_t, a_t)] \quad (3.2)$$

The difference is that traditional Sarsa selects an action based on the next observed policy, whereas traditional Q-Learning selects the action based on the next optimum policy. In computational terms traditional Q-Learning learns by computing the difference between the current Q-factor and the maximum Q-factor observed, whereas traditional Sarsa learns by computing the difference between the current Q-factor and the next observed Q-factor. Traditional Sarsa actually uses the value of the next action to guide its learning whereas, Q-Learning uses the maximum value observed.

3.4.3. Action-Selection Method

During learning, an important concept is choosing between actions that have previously been explored and those which have been unexplored. One of the main challenges of reinforcement learning is balancing the exploration and exploitation of actions. The aim is to determine when best to make use of existing knowledge and when to search for other options, which is known as *exploitation* and *exploration* respectively. ε – greedy is a popular action-selection method (Sutton and Barto, 1998). However, the problem with ε – greedy is that during exploration, each action is equally considered as the next and it does not distinguish between sub-optimal actions and the worse actions. A logical solution to this is to have a weighted value for each action and this is implemented in the *Softmax* action-selection algorithm (Sutton and Barto, 1998).

Softmax works by increasing the probability of selecting an action in a state, if the expected reward for the action is perceived to be higher (Dandurand *et al*., 2007). An example of the Softmax calculation is shown in equation 3.3. In other words, each state-action pair has a weighted probability associated with it and the algorithm works by selecting higher weighted state-action pairs more often, similar to how humans may make decisions (Dandurand *et al*., 2007).

Softmax is based on a set of assumptions, which are represented by different parameter values. Their use simplifies the action-selection process by having parameter values that help a learning agent to make the correct choice. The primary parameter is known as the *temperature* (Sikora, 2005). The temperature is a value greater than 0 and is essentially a weight that is added to the actions of each state-action pair.

Softmax can be used to exploit and explore actions gradually by using the Gibbs or Boltzmann distribution (Gray, 2007). The idea behind the temperature parameter is that at higher temperatures (Sikora, 2005) the learning agent explores more actions and as the temperature decreases exploration decreases also. Therefore, the temperature parameter effectively controls the trade off between exploitation and exploration. Fu & Anderson (in Gray, 2007, p. 168) have shown that Softmax is closer to the way humans and animals make decisions compared to a method like $\varepsilon-\text{greedy}$. Therefore Softmax would provide a more realistic approach for dancing robots to select dance steps.

$$\Pr(s,a) = \frac{e^{Q(s,a)/\tau}}{\sum_{i=1}^{n} e^{Q(s,a_i)/\tau}} \tag{3.3}$$

In equation 3.3 $\Pr(s,a)$ is the probability of selecting an action (a) in state (s). $Q(s,a)$ is the value function (otherwise known as the *Q-value* or *Q-function*) of an action (a) in state (s) and n is the number of actions available in a state (s). τ is the temperature parameter described above.

The temperature (τ) parameter is set by assigning to it a value or a series of different values. For example, Dandurand *et al*. (2007) used values from 1-10, where 1 represented *hardmax* ($\varepsilon-$greedy) and all other numbers represented different Softmax values. This was to determine a suitable Softmax value for balancing exploitation and exploration. Sikora (2005) used values of 5, 50, 500 and a meta learning mechanism to dynamically select a value that appeared to be the best for action-selection. Both researchers concluded that if a fixed temperature value was required (for stationary environments), then a value of five was the best to achieve weighted probabilities between actions. This also coincides with findings from psychological research and appears to be the same value for "estimates of [the] human working memory size" (Dandurand *et al*., 2007).

3.5. Summary

The underlining structure of the dancing model consisted of the integration of a real time beat detection algorithm, as well as a robotic system to demonstrate the dance. C++ is a supporting language of Webots and the beat detection algorithm is written in C++. This means that the entire system can be developed using C++. The Sony AIBO robotic dog can be used as a test bed for this research project and the Webots simulator can be used to

demonstrate the robots dancing and receive human feedback. The Adion beat detection program can be integrated into the system so that the robot would synchronise its movements to the musical beat in real time.

Using knowledge of reinforcement learning, a dancing framework can be developed for dancing robots to learn to dance. Reinforcement learning works by trial-and-error, enabling behavioural adaptation in response to changes in the environment and reward signals. Adaptation is an important part of dance, and dancers typically explore different dance actions in order to improve their performance. Therefore, reinforcement learning is a suitable approach to model dance in robots.

Two main algorithms from reinforcement learning, Q-Learning and Sarsa, were considered in this chapter. Q-Learning is widely used by researchers. However, in relation to learning to dance, it is less appropriate than Sarsa as the Sarsa algorithm has shown to be closer to the way in which humans learn.

For action-selection, the Softmax algorithm is considered better than $\varepsilon-$greedy because $\varepsilon-$greedy is equally likely to pick the "worst" or the "best" action but Softmax distinguishes the suitability of actions by weighting them.

Chapter 4
Learning To Dance To The Beat

This chapter explains a simple approach to make the robot learn to perform actions on the beat in real time through the application of the Sarsa algorithm and the Softmax algorithm from traditional reinforcement learning. Initial experiments were first conducted in order to determine which parameter values can be used to assist the robot in learning.

4.1. Methodology

In this first stage, the research began with the direct implementation of the Sarsa algorithm for learning and the Softmax algorithm for action-selection in order to determine suitable parameter values that can be used to help the robot learn to follow the beat of a musical signal in real time. With the suitable parameters obtained, the robot was then to learn to *bop* its head to the beat, followed by learning to perform predefined dance motions on the strong and weak beats of the music signal.

After each action-selection, the robot received scalar rewards which were defined internally to the system. The states were where the robot was in relation to the music (e.g. on-the-beat or off-the-beat), and the robot's actions were the decisions that the robot could make e.g. what movement to make depending on the strength of the music signal. These are described in more detail below.

Algorithm 4.1 shows the complete algorithm used in these experiments.

Algorithm 4.1: Sarsa Algorithm for Learning To Follow The Beat

1. Initialise parameters
   Sarsa parameters: $Q(s,a)$, r, α, γ
   Softmax parameters: τ
2. Play music
3. **Repeat** (while music is playing)
4. Randomly choose beat state (s_t)
5. Select an action (a_t) to do
6. Check state-action pair (s_t, a_t) in action database
7. **If** in action database **Then**
8. Select (s,a) according to Softmax ($\Pr(s,a)$)
9. Perform action (a) in chosen state (s)
10. **Else**
11. Perform action (a_t)
12. Write (s_t, a_t) in action database
13. Update value-function for state-action (s_t, a_t) pair

$$Q(s_t,a_t) \leftarrow Q(s_t,a_t) + \alpha[r + \gamma Q(s_{t+1},a_{t+1}) - Q(s_t,a_t)]$$

$$s_t \leftarrow s_{t+1}$$

$$a_t \leftarrow a_{t+1}$$

14. **For** all state-action pairs (s,a) Update Softmax

$$\Pr(s_t,a_t) \leftarrow \frac{e^{Q(s_t,a_t)/\tau}}{\sum_{i=1}^{n} e^{Q(s_t,a_i)/\tau}}$$

In the above algorithm, the robot learns to perform dance actions on the beat by randomly selecting dance behaviours and being rewarded (or punished) accordingly. $Q(s_t,a_t)$ is the value of the current decision (a_t) whilst in the current state (s_t); α is the *learning rate*; γ is the *discount factor*. Both the learning rate and the discount factor are of values between the range [0,1]. τ is the *temperature* parameter that controls how far apart are the probability values of state-action pairs; r_t is the immediate reward given for the current decision (a_t) performed in the current state (s_t) and $Q(s_{t+1},a_{t+1})$ is the value of the next decision (a_{t+1}) performed in the next state (s_{t+1}) the robot will be in.

4.2. Experimental Procedure

Learning to dance to the rhythm was divided into three experiments. The first experiment was an initial exploration of the traditional Sarsa algorithm to determine what parameter values to use. The second was for the robot to learn to perform a single head motion in real time with the music to on-the-beat rhythms (i.e. on each time-signature). The third was for the robot to learn to perform correct dance motions to the correct strength of the music signal in real time. The following sections go into these in more detail.

4.2.1. Experiment 1 (Learning Parameters)

In this experiment, the Sarsa algorithm was explored offline (without the music and robot) with the input of different parameter values for the learning rate (α) and the discount factor (γ) to determine the suitable parameters that can be used for all the experiments that will be conducted in this research. Only one state was defined in the system containing four actions (named Dance Action 1, Dance Action 2, Dance Action 3 and Dance Action 4) of which a reward of +1 was associated to one action (Dance Motion 1) and the remaining actions given a reward of 0.

4.2.2. Experiment 2 (Bop To The Beat)

In this experiment, the robot was programmed to learn to bop its head (move its head up and down) as a single dance motion in real time to the beat of the musical signal using the Sarsa algorithm. The music file used was “Any Dream Will Do” by Jason Donovan. This was used because of the simple beat structure. The robot could move its head anytime while the music was being played. The states of the system were defined as *on-the-beat*

and *off-the-beat* which meant "bopping the head on the beat" and "bopping the head off the beat" respectively. These two states were the choices for the robot when deciding to perform the head motion.

The robot could perform the head movement on any state while the music was being played. The states were specifically defined to help with the learning. Although, the robot could have been programmed to move its head anywhere while the music was being played, the actual detection of distinctive states would have proved much harder as the chances of actually performing the head movement on a beat were very slim. The consistent performance of the head movement to off-the-beat rhythms with little attempts (or no attempts) performed on on-the-beat rhythms would cause the robot to learn the wrong behaviour due to the accumulation of the wrong rewards or the lack of exploration of on-the-beat rhythms. For this reason, the robot was programmed to perceive the music as two states defined as *on-the-beat* and *off-the-beat* and were given threshold ranges to categorise them.

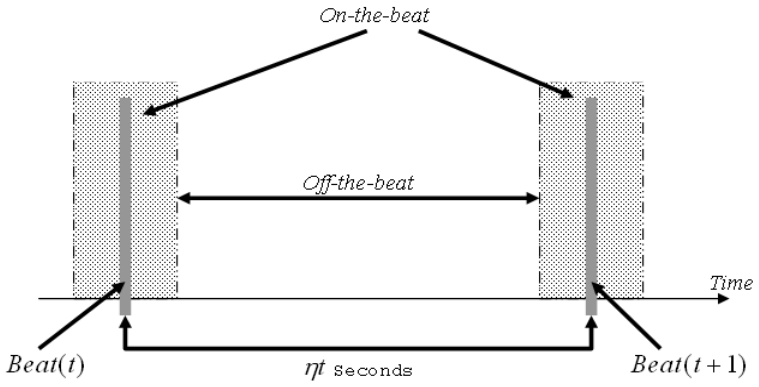


Figure 4.1 – Description of on-the-beat and off-the-beat rhythms where t is the time for each beat occurrence and ηt is the predicted number of seconds between each beat. The shaded region is *on-the-beat state* and outside of this range is *off-the-beat state*.

Figure 4.1 illustrates the states of the music in which the robot can perform the movement. Here, once the beat detection algorithm had made a prediction (ηt) as to when

the next beat (*Beat(t+1)*) would occur, on-the-beat states were defined to be in the range +/-0.1 seconds of the predicted time (ηt). This was so that the robot could perceptually be seen to be in sync with the music if, for example, the beat actually occurred earlier or later than the predicted time. This range is indicated by the shaded area in the diagram above (Figure 4.1). If the head movement occurred outside the on-the-beat state, then the robot was recorded as being in state off-the-beat. Furthermore, the robot was programmed to move its head in one state at a time. For example, if the first state selection was on-the-beat, then the robot would perform the head movement once on the next on-the-beat rhythm. If after this, the next state selection was for off-the-beat, then the robot would perform the head movement once, immediately after or wait for the next off-beat state. In other words, the robot would select what state to perform the head movement and if the robot found itself in the selected state then it would perform the head movement, otherwise it would record the state it found itself in and perform the action the next time the desired state came up. The learning rate was made high ($\alpha = 0.8$) and the discount factor was made low ($\gamma = 0.2$) based on the initial results obtained from Experiment 1.

4.2.3. Experiment 3 (Dance To The Strongest Beat)

In the third experiment, the emphasis was to explore dance using more dance motions. The robot was to learn to perform a dance motion to the strongest beat of the musical signal detected and all other dance motions on the weaker beats, whilst the music is playing. The strongest beat was used as an indication of the *downbeat* in music literature, which is defined as the first count of a music signal's time-signature (i.e. the "1" in the

count of "1-2-3-4" of a 4/4 time-signature or in a "1-2-3" of a 3/4 time-structure). In dance choreography and music conducting for example, it is usually used to indicate when a particular dance movement or instrument in the orchestra should come in to the music. In this experiment, only one dance motion was required to be performed on the strongest beat and all other dance motions performed on weaker beats.

Using the Adion beat detection algorithm (adionSoft, 2007), the strongest beat was determined by determining the strength of beats of the musical signals by isolating the energies (intensity/frequency signals) of each beat detected. This was determined by setting a threshold. If the energy of the next detected beat was greater than the threshold then it was a strong beat, otherwise a weak beat. The threshold was calculated to be a continuous working average of all the previous beat intensities detected.

This approach was based on the assumption that if the human ear detects beats based on the different loud sounds of the musical signal, at varying periodic intervals, and the beats can be classified as either strong or weak, then the loudest sound of a strong beat is a good indication of the beginning of a time signature, i.e. the downbeat. There are other algorithms for detecting the beat and determining the downbeat of the music, however, the approach proposed here worked really well for this experiment. The same music signal as in Experiment 2 was used to test the strong beat, and the robot was programmed to receive immediate rewards using the Sarsa algorithm.

The robot was rewarded after each dance motion performed. Five predefined dance motions were used. The predefined dance motions are described in Figure 4.2, where *A* was defined to be the initial home position of the joints, and *B* was the position moved to by the joints in question.

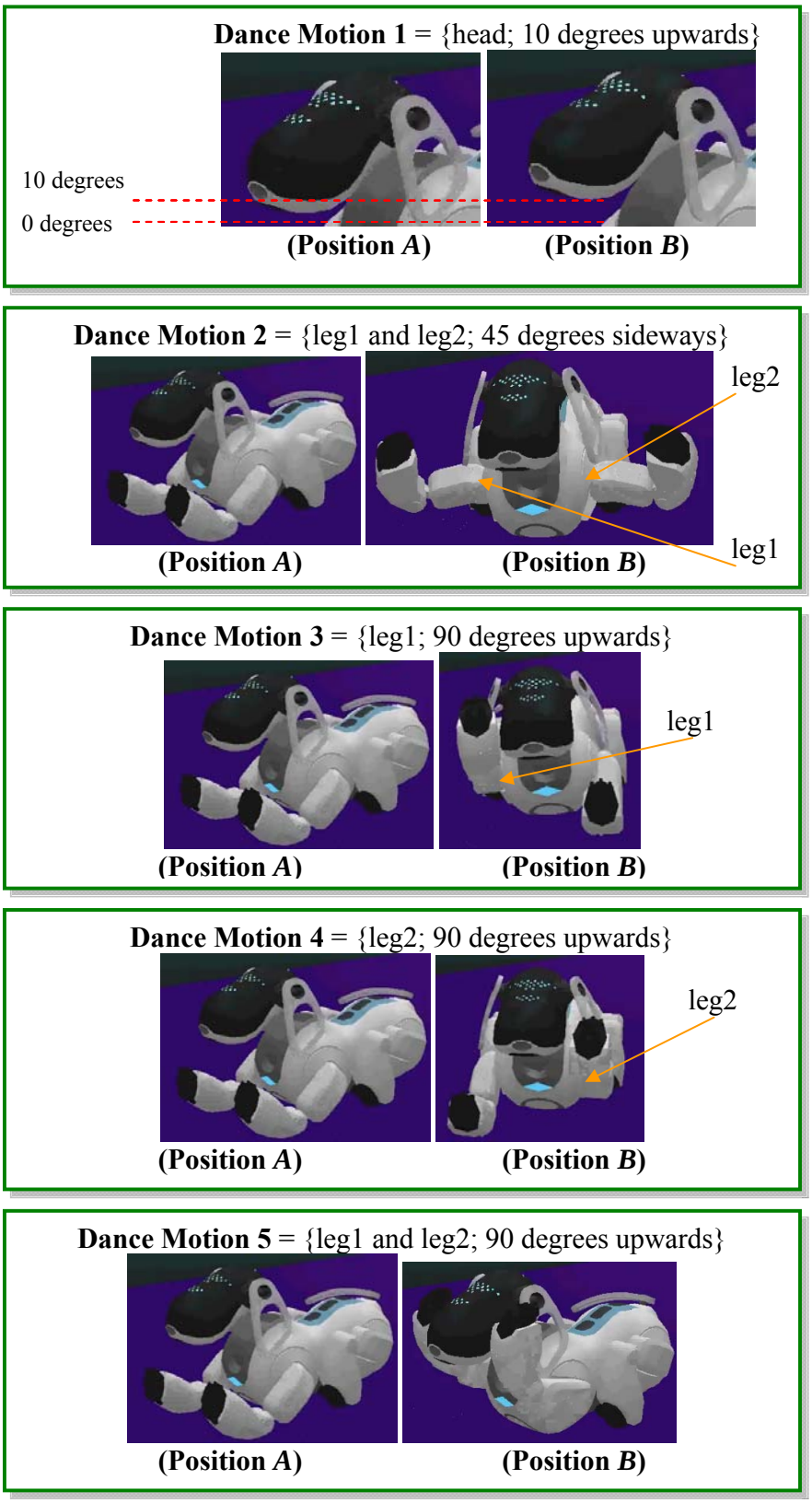


Figure 4.2 – Positions of the different dance motions for Experiment 3

The transition from position *A* to position *B*, and from position *B* back to *A*, formed two single movements (gestures) into one motion. This meant that the robot's joints always started and ended in the same state (i.e. position *A*); ready to undertake the next action.

The robot's rhythmic ability was to perform actions only to on-the-beat rhythms. For this, it was not necessary to define a beat range as was done for Experiment1. Instead, when the beat was detected, the robot would automatically perform the action chosen.

The five dance motions were internally defined in the robot's system prior to learning. The robot selected actions using the Softmax algorithm and learnt to perform the actions on the beats.

Dance Motion 2 was pre-programmed to be the designated dance motion that the robot was to learn to perform on the downbeat (strongest beat), and all the other four dance motions could be performed on any of the other weaker beats. Table 4.1 summaries this expectation.

Action Name	State
Dance Motion 1	Weaker Beat
Dance Motion 2	Strongest Beat
Dance Motion 3	Weaker Beat
Dance Motion 4	Weaker Beat
Dance Motion 5	Weaker Beat

Table 4.1 – Summary of what state to perform actions

A scalar reward of +1 was always given to the desired dance motions performed at the right place. Credit (reward) was distributed according to the previous state-actions pairs that the robot performed prior to receiving the reward. The learning rate was made high ($\alpha = 0.8$) and the discount rate was made low ($\gamma = 0.2$) based on the initial results obtained from Experiment 1.

4.3. Results & Observations

4.3.1. Experiment 1: Results & Analysis

The idea of Experiment 1 was to explore which parameters can be used for the learning rate (α) and the discount factor (γ) in the Sarsa algorithm for the robot to learn the appropriate behaviours while following the music. The temperature (τ) value of the Softmax action-selection method was kept constant at a value of five. The algorithm was defined with one state and four possible dance motions that could be performed in that state. A fixed reward value of +1 was given to Dance Motion 1 and all other dance motions (Dance Motions 2, 3 and 4) were given a reward value of 0. The experiment was run over five trials with 20,000 time steps for each trial. The results can be seen in the figures below.

(1) Results of Exploring the Learning Rate

To begin with, the learning rate was explored with increasing values from zero to one, whilst the discount factor was kept at a fixed low ($\gamma = 0.2$) the first time, and at a fixed high ($\gamma = 0.8$) the second time. A record of the average total occurrence of dance motions over five trials of running the algorithm was kept, with each trial terminating after 20,000

time steps. Figures 4.3 and 4.4 show the results with a low discount factor and Figures 4.5 and 4.6 show the results with a high discount factor.

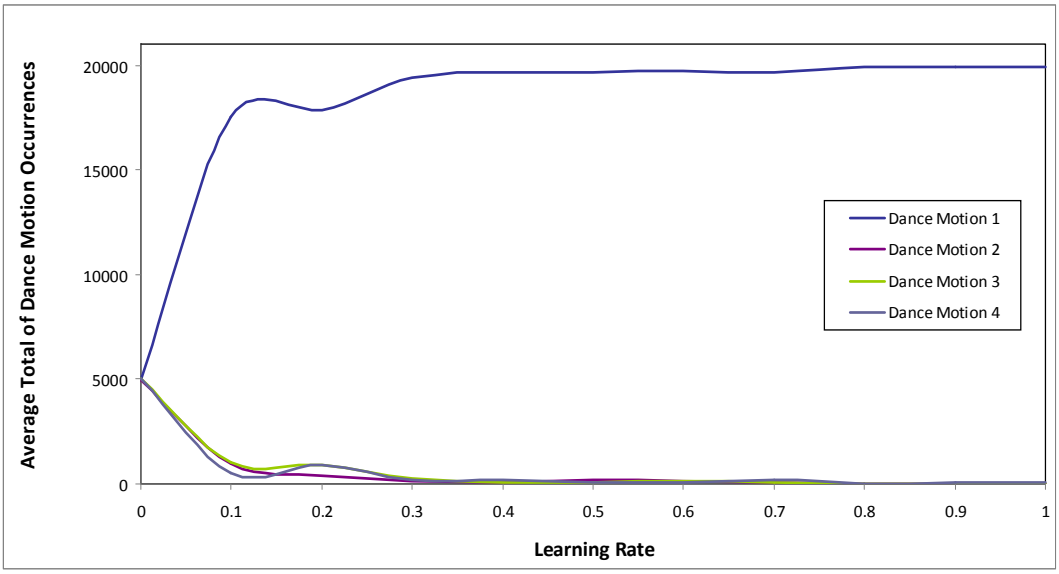


Figure 4.3 – Average occurrence of dance motions with the discount factor made low ($\gamma = 0.2$)

From Figure 4.3, we see that higher values of the learning rate produced the desired motion (Dance Motion 1) to be selected more often, when the discount factor remained fixed at a value of 0.2. This is supported by Figure 4.4 shown below.

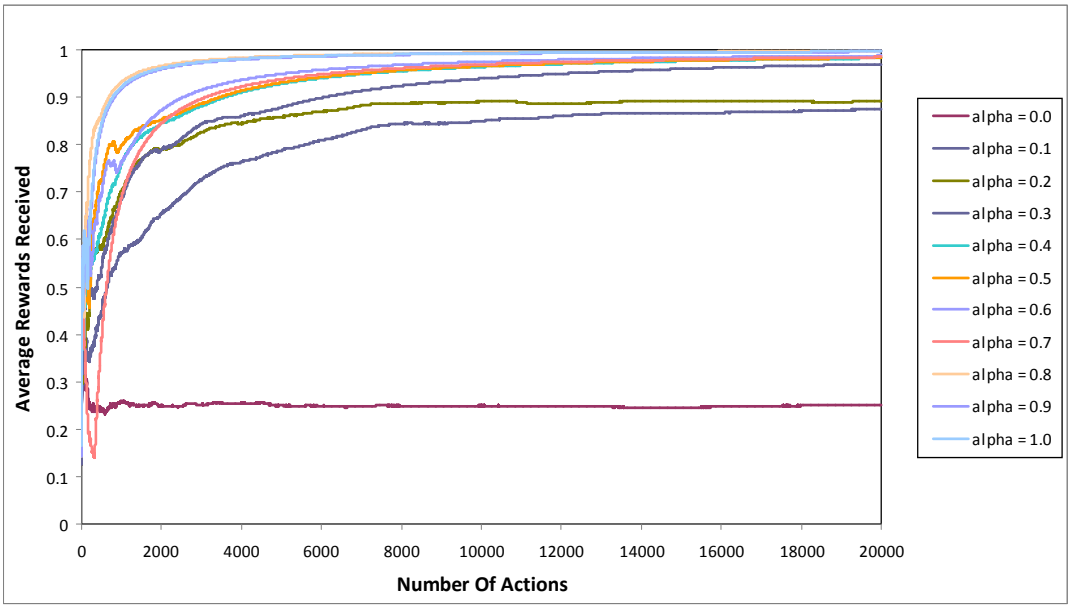


Figure 4.4 – Learning performance of dance motions with the discount factor made low ($\gamma = 0.2$)

From Figure 4.4, we see that when the learning rate (alpha) was zero, there was no learning in the algorithm and the performance of the algorithm improved with time for increasing values of the learning rate.

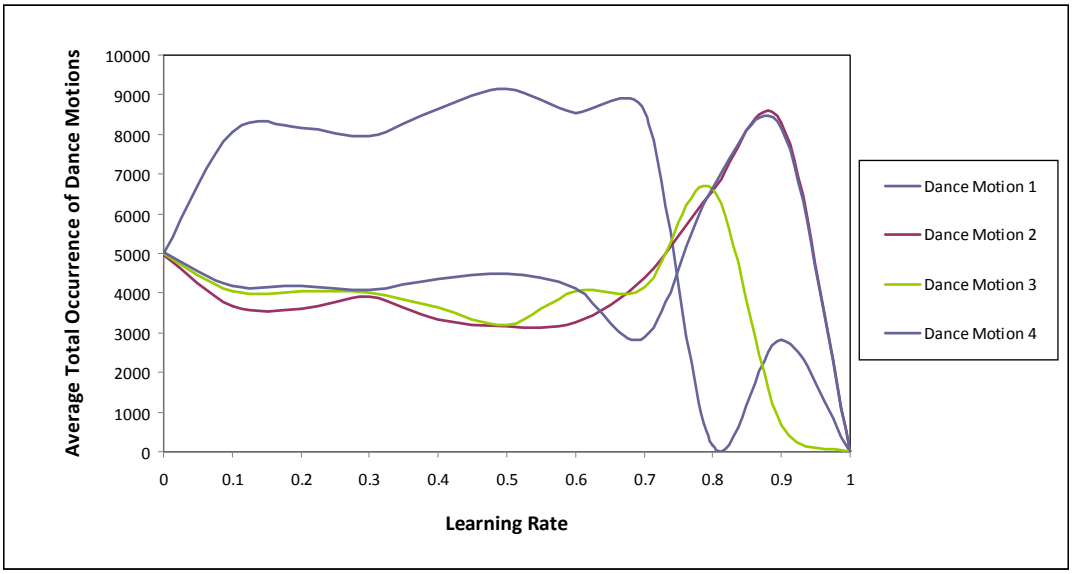


Figure 4.5 – Average occurrence of dance motions with the discount factor made high ($\gamma = 0.8$)

Figure 4.5 shows the result with a high value of the discount factor (γ = 0.8). From Figure 4.5, we see that the increasing values of the learning rate produced the desired motion (Dance Motion 1) to be selected more often for all values of the learning rate up until a value of approximately 0.7, but performed rather unpredictably after that. Generally, the performance appeared to be worse in comparison to the low value of the discount factor explored, with the total occurrence of dance motions being selected more closely together at $\gamma = 0.8$ than at $\gamma = 0.2$. This is supported by the results shown in Figure 4.6 below.

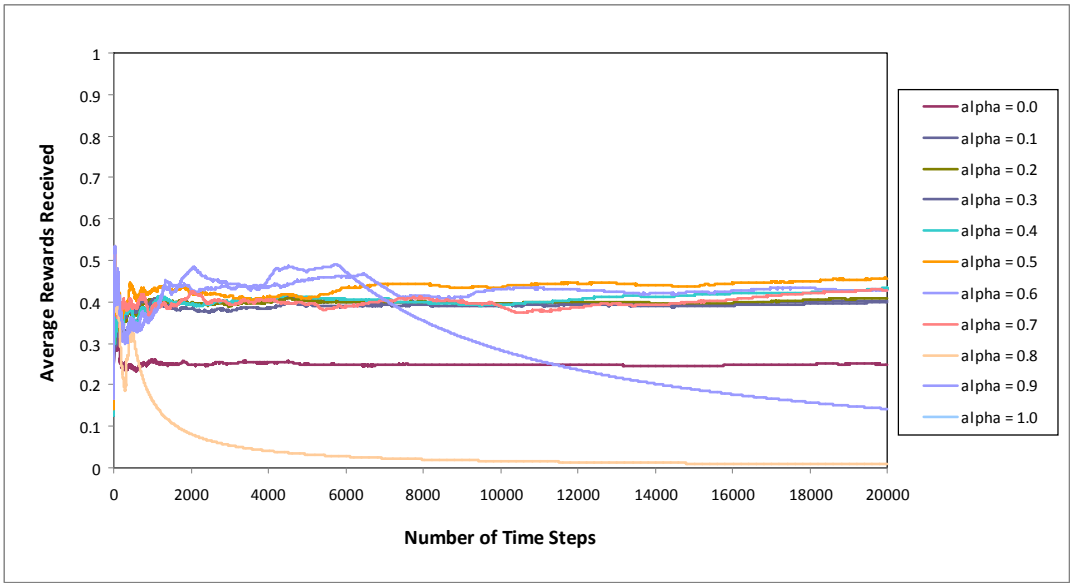


Figure 4.6 – Learning performance of dance motions with the discount factor made high (γ = 0.8).

Figure 4.6 shows that the increase in the learning rate (alpha) did not encourage the algorithm to perform the desired behaviour (Dance Motion 1) with increasing time, when the discount factor was at a value of 0.8. In fact, for some values of the learning rate, the performance worsened. There appears to be no learning taking place. Based on these details, it can be concluded that higher values of the learning rate would perform better with a low discount factor of 0.2 than with a high discount factor of 0.8.

(2) Results of Exploring the Discount Factor

The discount factor was explored in the same way as the learning rate was explored, by testing out increasing values of the discount factor against a low learning rate ($\alpha = 0.2$) and a high learning rate ($\alpha = 0.8$). As was carried out above, a record of the average total occurrence of dance motions over five trials of running the algorithm was kept, with each trial terminating after 20,000 time steps. Figures 4.7 and 4.8 show the results with a low learning rate and Figures 4.9 and 4.10 show the results with a high learning rate.

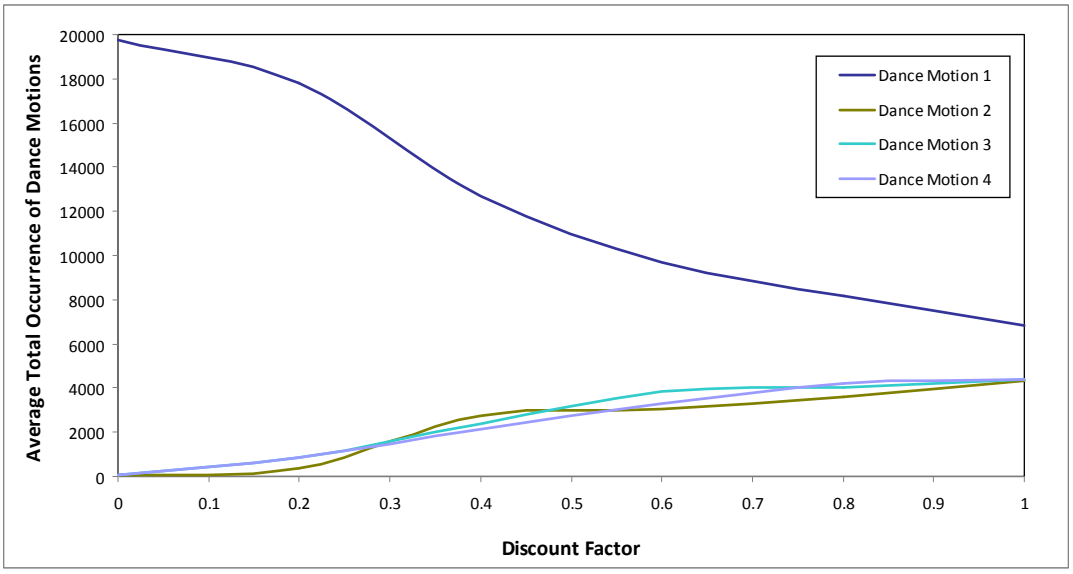


Figure 4.7 – Average occurrence of dance motions with the learning rate made low ($\alpha = 0.2$)

Figure 4.7 shows that an increase in the discount factor, while the learning rate remained low ($\alpha = 0.2$), actually made the total occurrence of the dance motions closer together, decreasing the learning performance of the algorithm. This too can be supported from the results shown in Figure 4.8 below.

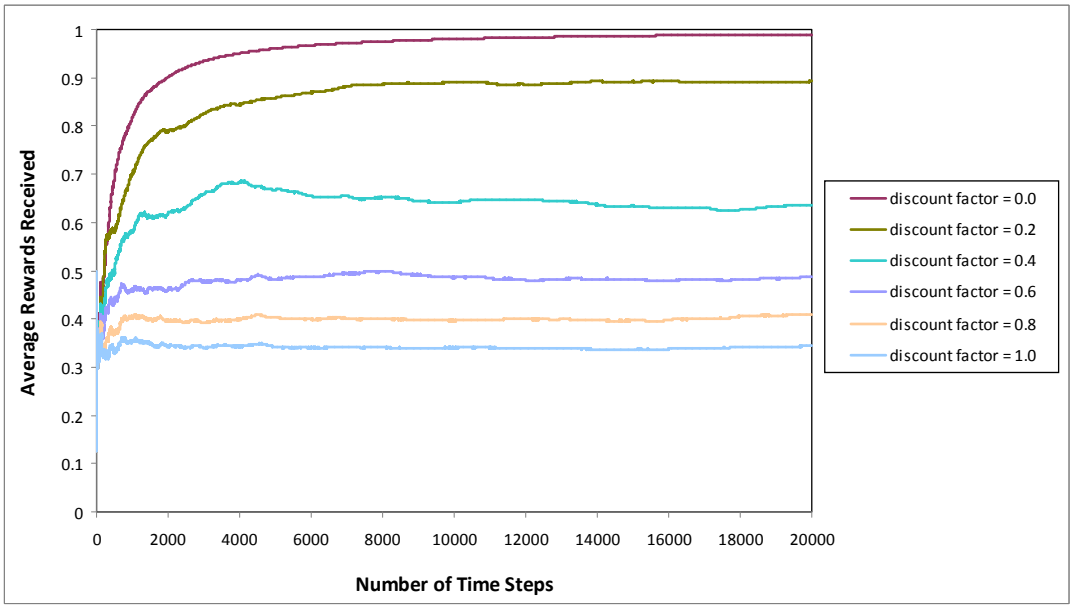


Figure 4.8 – Learning performance of dance motions with the learning rate made low (α = 0.2).

Looking at Figure 4.8, performance seemed to improve with smaller values of the discount factor at a low learning rate of α = 0.2.

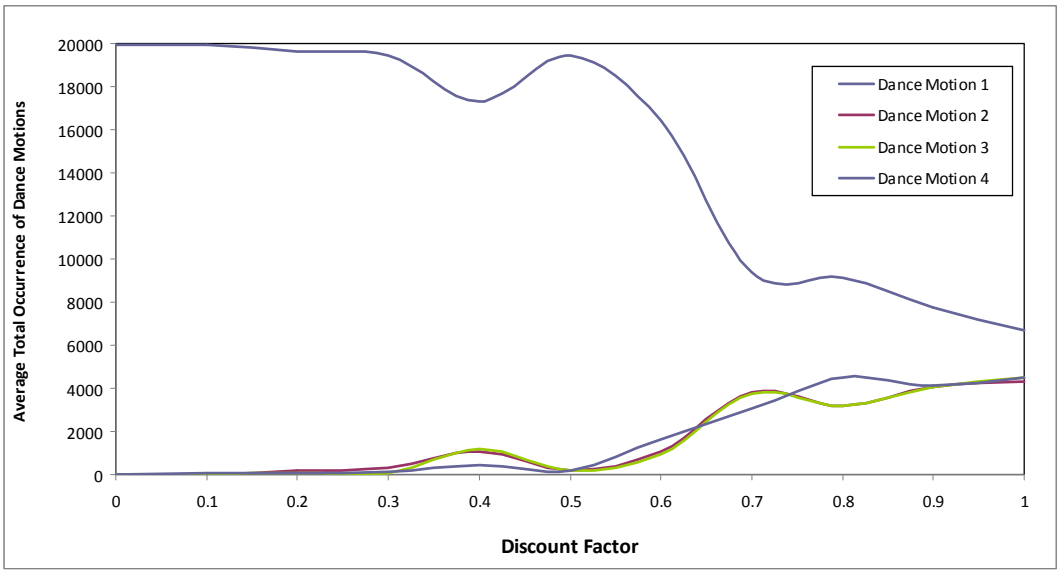


Figure 4.9 – Average occurrence of dance motions with the learning rate made high (α = 0.8).

On the other hand, from Figure 4.9, as the discount factor approached a value of one, the occurrence of the desired dance motion (Dance Motion 1) decreased if the learning rate remained fixed at a high value of 0.8. Figure 4.10 below supports this conclusion, showing that the algorithm performs more optimally over time with decreasing values of the discount factor, and a high value of the learning rate ($\alpha = 0.8$).

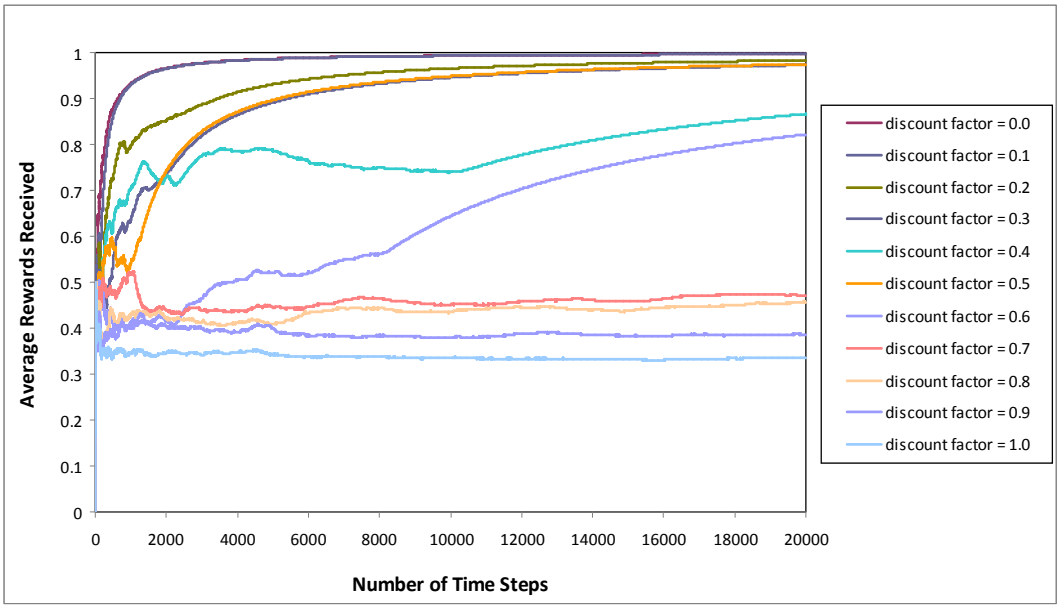


Figure 4.10 – Learning performance of dance motions with the learning rate made high ($\alpha = 0.8$).

In conclusion, the algorithm showed good results when the discount factor was low and poor results when set high. Table 4.2 below summarises these results obtained and shows a ranking of the experiment based on performance, where the best performance is ranked first place, then the next best performance is ranked second place, and so on, down to fourth place.

Learning Rate (α)	Discount Factor (γ)	Result	Performance Ranking
Low	Low	Good	2
Low	High	Bad	3
High	Low	Good	1
High	High	Bad	4

Table 4.2 – Summary results of Experiment 1

These results were obtained with one state defined and a possibility of choosing an action from four actions. Setting the learning rate high ($\alpha = 0.8$) and the discount factor low ($\gamma = 0.2$) seemed to produce a better learning performance compared to the other combinations explored. Therefore, these parameter values will be used for all experiments in this research.

4.3.2. Experiment 2: Results & Analysis

For Experiment 2 (bopping to the beat), the performance of the robot on each attempt to move its head on the beat and the percentage average of selecting the desired behaviour (on-the-beat) over five trials was recorded. Each trial constituted to a complete run of the robot bopping its head to the music and consisted of on average 600 time steps of action selection. Figure 4.11 below shows the result.

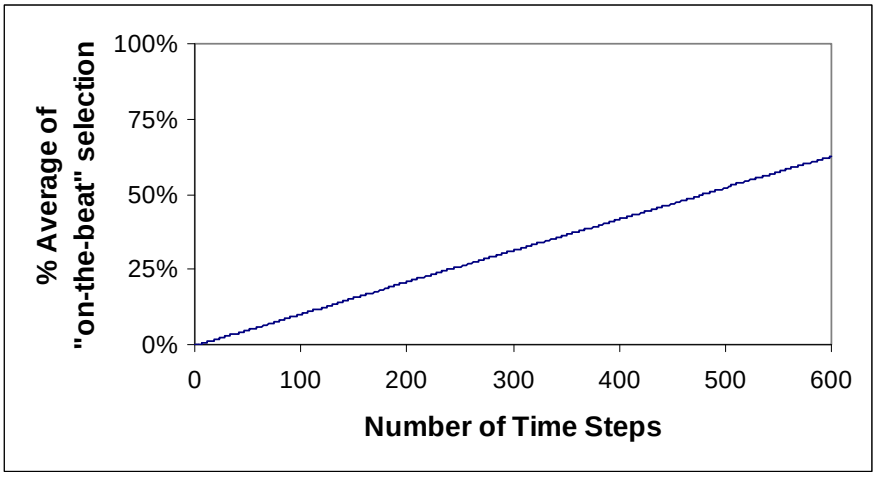


Figure 4.11 – Experiment 2 results of learning to select the head movement in real time on the beat using Sarsa

Looking at Figure 4.11, we see that the robot gradually increased the number of times it selected the optimal action to perform the head movement as the music was playing. Although 100% optimality was not reached, we can see that this would have been achieved given that the music had continued playing.

4.3.3. Experiment 3: Results & Analysis

For Experiment 3, recall that the idea was for the robot to learn to perform Dance Motion 2 on the strongest beat intensity (otherwise known as *downbeat* in this research) detected and all other dance motions (Dance Motions 1, 3, 4 and 5) to be selected on any of the weaker beats.

Figure 4.12 below shows the results of Experiment 3. The graph shows the results of the average rewards received out of the first 600 time steps in five trials (runs) of the robot's dance to the music in real time.

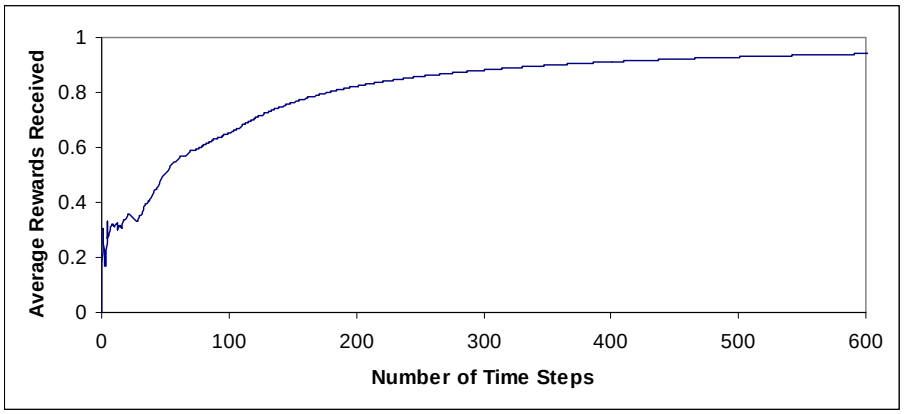


Figure 4.12 – Experiment 3 results of learning to perform actions on the correct music states using Sarsa.

The reward that the algorithm could receive after performing dance motions in the correct states, was one. All other dance motions performed in the wrong states of the system received a reward of zero. Figure 4.12 shows the robot's explorative behaviour in the initial stages leading to a gradual progression in performance as the music continued playing. Here, the graph shows that it would gradually converge to one, given that the music had continued playing and the robot was allowed to continue selecting actions.

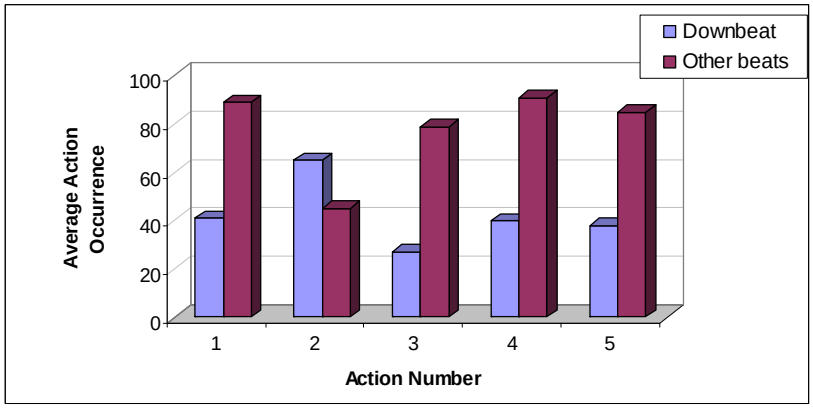


Figure 4.13 – Experiment 3 results of learning to perform actions on the correct states using Sarsa

Figure 4.13 specifically shows the average occurrence of the five dance motions on the detected downbeat and the weaker beats of the music signal. As can be seen from Figure 4.13, on average, dance motions were selected on the correct states.

4.4. Summary

Three simple experiments were carried out to show the working behaviour of reinforcement learning applied to robot dancing. Their main purposes were to explore the fundamentals that are necessary for learning in the dancing framework. These included obtaining the values for learning coefficients, following the beat correctly and learning to do dance steps on the correct beat intensities. The results demonstrate that both the Sarsa algorithm and the Softmax algorithm can be used for the learning framework and action selection respectively.

Whilst this implementation of the reinforcement learning algorithms can be used to learn desirable behaviours, it does not provide a complete dancing model by itself. There is more to dancing than demonstrating rhythmic behaviour by using predefined movements. Dancing requires the formation and coordination of dance steps and combinations and not just the use of the same actions all the time. Although a simple solution was explored in Experiment 3, i.e. by selecting different actions to perform on all beats except the downbeat, the robot was still limited to predefined movements. Furthermore, these experiments did not consider feedback from human observers. The next chapter explores the fundamentals necessary for creative and structured dance steps in robots, followed by learning from human feedback in Chapter 6.

Chapter 5
Generating Dance

The previous chapter shows how it is possible to implement traditional reinforcement learning to enable the robot learn to perform predefined actions to the beat of the music in real time. While the use of reinforcement learning in the previous chapter gave the robot the ability to learn the appropriate dance behaviours, it did not teach the robot how to generate its own actions and be creative with them. This chapter describes how a robot can generate its own dance actions (using internally defined positive and negative reinforcements), which can be grouped together to generate more aesthetically pleasing dances, whilst dancing to music.

5.1. Methodology

5.1.1. Generating Dance Actions

In the previous chapter, the robots initial training was to learn to perform dance actions to on-the-beat rhythms, using the underlying structure of the Sarsa algorithm. Rewards were fixed and pre-assigned throughout the learning process. The approach taken in this stage of the research begins with the robot only moving to the beat (i.e. not off-the-beat rhythms) and generating dance actions based on the knowledge described earlier of the definition of a dance (Chapter 2) and the fundamental movements/ skills (Chapter 2) necessary for aesthetic display. The aim in this chapter is for the robot to learn to perform meaningful (human-like) dance actions, as opposed to unrelated movements, which can be combined to form longer, repetitive sequences in its dancing. To achieve this, the

robot is programmed to randomly choose any number of joint combinations to do on any beat and is rewarded (or punished) for each selection it makes.

There are three skills (fundamental movements) necessary for the robot to learn. They are known as *Opposite*, *Symmetry* and *Formation* in this research and are defined as follows:

- ***Opposite*** skill is the movement of opposite joints that are on opposite sides of the robot's body moved together in unison. For example, the movement of the left and right joints or front and behind joints, at the exact same time on a beat.

- ***Symmetry*** skill is the movement of the same opposite joints moved one after the other. In dance literature, symmetry refers to the exact mirroring of the body at the same time, which is the same definition in this research for *Opposite* skill above. However, in this research, Symmetry is used to mean movements that are mirrored, shortly after each other, for example, on the next beat. In dance literature, this is known as *Canon*.

- ***Formation*** skill is defined as a "back and forth" motion of joints, for example left and right movements or upwards and downwards on a beat. This is known as *Form* in dance studies.

These definitions are based on the literature reviewed in Chapter 2 and are initially predefined in the system for the robot to learn. Each accomplished skill is weighted with fixed reward values (internal to the system) so that after many selections the robot learns to dance with *skilled dance motions* only. The robot retains a repertoire of each newly generated skilled and unskilled dance motion that is created from the

random selection of joints and their directions. This is the basic component of a dance motion, called a *gesture*. A gesture is described in this thesis as "the movement of one or more joints in one direction (the same or different) on one time step". A *time step* is the selection of a dance behaviour typically on a beat. Dance motions are formed when two sets of gestures are performed on two time steps. These dance motions become "skilled" when the gestures that make up the dance motions can be classified as either Opposite, Symmetry or Formation as described above. Opposite skill by definition requires only one time step. Therefore, with the exception of Opposite dance motions, all dance motions (skilled or unskilled) require two time steps.

The complete history of generated dance motions performed by the robot is stored in the *action database*. The robot performs random gestures on every beat. For each gesture that is selected, the action database is scanned for dance motions that exactly match the current and previous gestures selected. If one or more matches of gestures (current and previous) are found in the action database, then an action will be selected according the Softmax rule. If no match is found, then the algorithm considers the selected movement a "new" movement (i.e. because it is not in the action database) and the gestures form a new dance motion, for the robot to perform and is recorded in the action database and rewarded accordingly. The decision to choose (explore/ exploit) a dance motion on each beat is based on the Softmax algorithm, which provides varying probability values for each dance motion. Figure 5.1 below shows a flow diagram of the formation of dance motions. This approach is employed so that the robot is guaranteed to explore new dance behaviours since, on every beat, it randomly chooses gestures to

perform, but would only perform those movements if they were new (i.e. not in the action database) or according to Softmax.

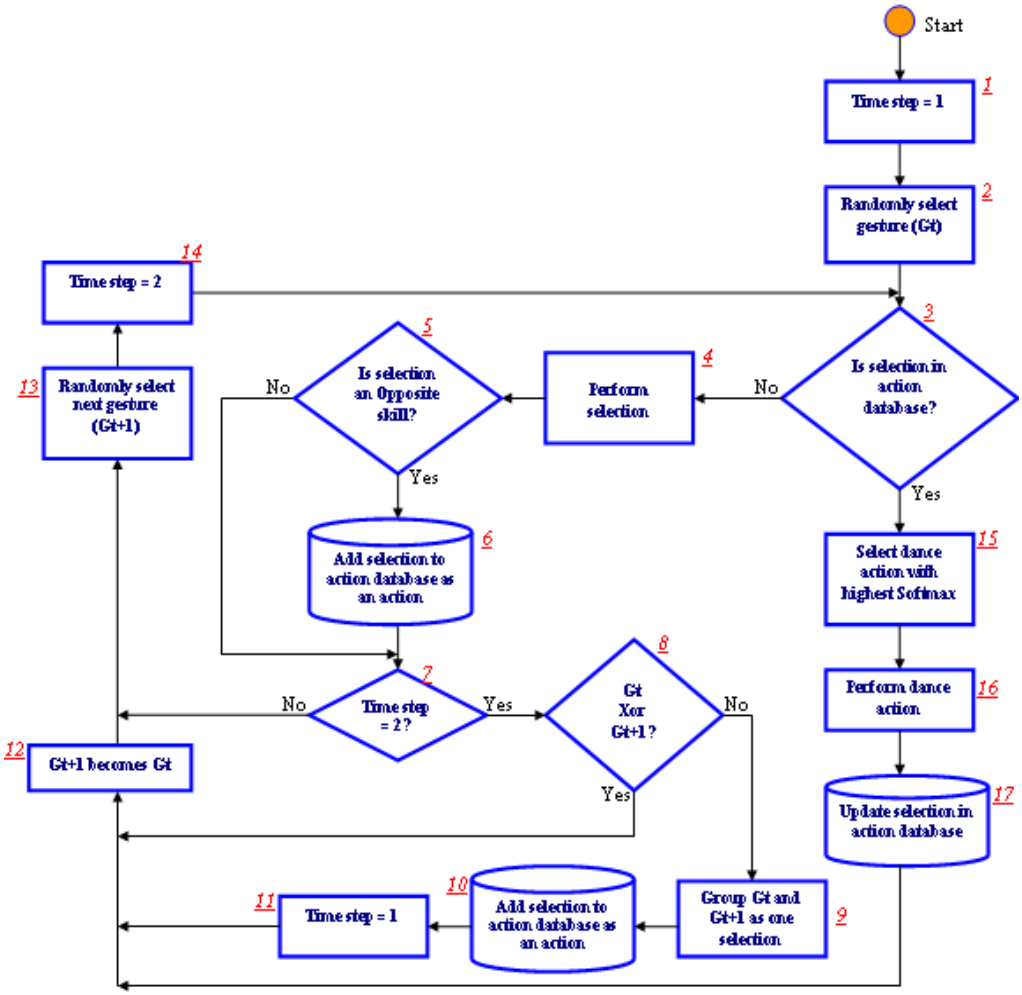


Figure 5.1– Data flow diagram for developing dance motions

Figure 5.1 shows the complex architecture used to generate initial dance motions for the robot. Each component of the architecture is numbered for the purpose of explaining the component parts. The robot begins generating motions as soon as a beat is detected. Once a beat is detected, this is registered as the first time step (process 1). The robot then selects one or more joints to move in random directions for each joint selected (process 2). The robot does not perform these gestures but merely temporarily stores

them as a movement it would "like" to do. In order to decide properly, whether or not this gesture should be performed, the robot must first check its action database to see whether or not it has performed this gesture before (process 3). If the selection is in the action database, then the robot would select and perform the dance action according to Softmax. In other words, there is no guarantee that that particular selection would be performed. On the other hand, if there is no such history for the selected gesture in the action database, then the robot performs the behaviour, in order to obtain immediate feedback on the behaviour and then add the gesture to the action database as an action for future selections. However, before adding the selection to the action database, the robot must first check what type of behaviour it is. As described above, dance behaviours consisting of two gestures performed on two time steps are stored in the action database as dance motions, with the exception of an Opposite dance motion, which is generated after one time step. Dance motions that have been generated after two steps can be either skilled or unskilled in their definition. This is shown in processes 5 – 10 in Figure 5.1.

In process 8, the robot does a check on the current ($t+1$th) and previous (tth) gestures performed to determine their exclusive disjunctions (xor). In other words, a dance motion is only generated (process 9) if both gestures performed on the tth and $t+1$th time steps are either both Opposite skills or not. If the check results to *false* (i.e. "No" in Figure 5.1), then a dance motion is generated and this is stored as an action in the action database (process 10). Both current and previous gestures must be the same type in order for a dance motion to be generated. Table 5.1 shows the possible logical outcomes and the equivalent exclusive disjunctive bitwise values after gestures have been performed on two time steps.

G_t **Gesture Type**	G_{t+1} **Gesture Type**	G_t **Bitwise Value**	G_{t+1} **Bitwise Value**	G_t **xor** G_{t+1} **Bitwise Value**	**Dance Motion Type** (G_t **xor** G_{t+1})	**Stored In Action Database?**
Non-Opposite	Non-Opposite	0	0	0	Dance Motion	Yes
Non-Opposite	Opposite	0	1	1	Not A Dance Motion	No
Opposite	Non-Opposite	1	0	1	Not A Dance Motion	No
Opposite	Opposite	1	1	0	Dance Motion	Yes

Table 5.1 – Exclusive disjunctions of gestures to generate dance motions on two time steps. For the last row, although Opposite gestures form dance motions themselves, when two Opposites have been performed on two time steps, it's possible to generate dance motions that contain more than one skill, producing more interesting movements. See text for details.

It is possible for non-opposite joints performed on two time steps to form a dance motion. The final dance motion (Opposite and Opposite) shown in Table 5.1, would produce very interesting motions. Opposite moves performed after each other does constitute to a dance motion, but would not be an Opposite dance motion, but rather a dance motion with a Symmetry skill or a Formation skill. The movement could consist of opposite joints moved on opposite sides of the body (i.e. Symmetry skill) or the same joints moved in a "back and forth" motion on two time steps (i.e. Formation skill). This differs from an Opposite dance motion in that, to achieve an Opposite dance motion, opposite joints need only be performed on one time step and not two.

This approach generates a series of dance motions that are intertwined and are related, demonstrating more control in the robot's autonomous behaviour. Figure 5.2 shows a conceptual view of two examples of the generation of dance motions (DM) on different time steps (t).

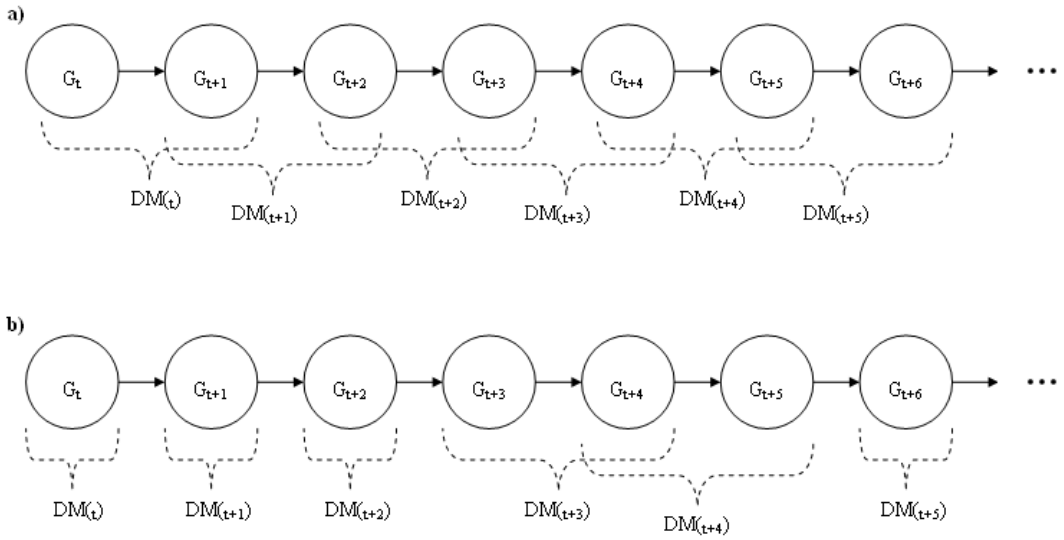


Figure 5.2– State flow diagram for developing dance motions. a) is an example of dance motions generated on two time steps. b) is a an example of a series of dance motions generated on one and two time steps.

Figure 5.2a, shows the generation of intertwined dance motions where the most recent gesture becomes the previous gesture for another dance motion. Figure 5.2b, shows a mixture of dance motions generated on one and two time steps. $DM_{(t)}$, $DM_{(t+1)}$, $DM_{(t+2)}$ and $DM_{(t+5)}$ from Figure 5.2b can be representations of Opposite dance motions generated. Notice here that this type of dance motion is not connected to neighbouring gestures, but becomes a dance motion performed on its own. However, it is possible for Opposite gestures (Opposite dance motions), performed one after the other on two time steps, to form new dance motions (i.e. a dance motion of two Opposites) as described above in Table 5.1. This produces a rather special case dance motion that has more than one skill. For example, if $DM_{(t)}$ and $DM_{(t+1)}$ were the same Opposite gestures performed in opposing directions then this dance motion would be a Formation dance motion consisting of Opposite gestures. On the other hand, if the gestures were exactly opposite

to each other on the body, then this would generate a Symmetry dance motion with Opposite gestures in its definition.

Once a dance motion has been generated, this could be a skilled or an unskilled dance motion depending on what joints were moved on gesture G_t and gesture G_{t+1}. Gestures on their own are not stored in the action database except if they are Opposite skills, in which case, they are known as *Opposite dance motions*. However, gestures are randomly selected (but not necessarily performed) so that new dance combinations can be explored and generated.

Steps 1 to 14 in Figure 5.1 describe the robots exploration of "new" dance behaviours i.e. dance motions that are generated and are not listed in the robot's action database. During these steps, the robot autonomously performs a mixture of movements and generates desirable initial dance motions which are stored in the action database, and rewarded depending on the dance motion generated.

Steps 15 to 17 are used for both the exploration and exploitation of existing actions. This is achieved by the Softmax algorithm (process 15).

5.1.2. Structuring A Dance

Whilst it is necessary for the robot to have a repertoire of basic dance motions, this alone is not sufficient enough for a dance. The dance motions should be arranged in such a way to demonstrate control and more interesting dance patterns. From Chapter 2, McGreevy-Nichols *et al.* (1995) proposed a definition of dance being a build up of basic dance motions which, when combined, form longer sequences known as a *dance phrase* and *dance section*. Their definition was in relation to the way humans structure and develop their dancing. It is therefore a logical approach that can be applied to dancing robots and this chapter makes use of this knowledge.

As described above, a dance motion is generated after two time steps (beats), or in the case of the Opposite skill, this was after one time step. These dance motions were then combined to form dance phrases and dance sections in a similar way to the generation of dance motions. The logic in this research was that, in order for a dance phrase to be generated, two dance motions would have to be performed sequentially, i.e. one dance motion immediately following another dance motion. Similarly, to generate a dance section, dance sections were formed after two dance phrases were performed sequentially i.e. one dance phrase immediately following another dance phrase. Figure 5.3 shows an illustration of this structure.

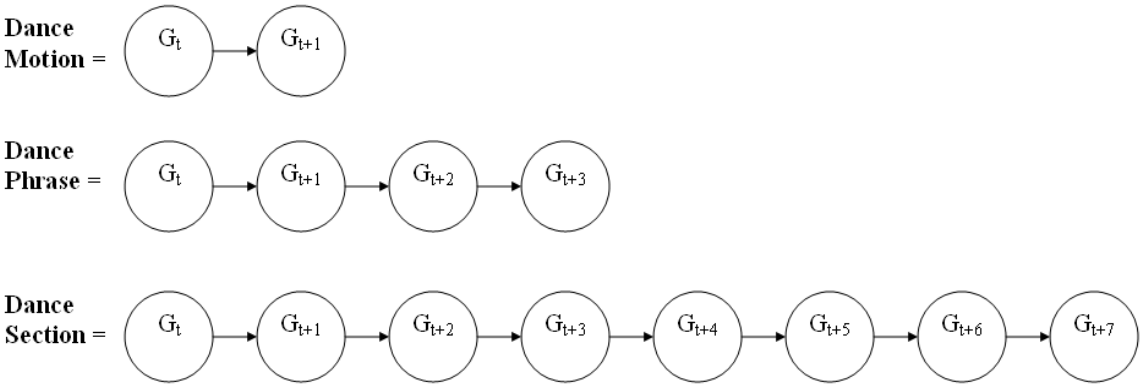


Figure 5.3 – Illustration of dance structure where G_t represents *gestures* performed on time *t*

Figure 5.3 shows the typical structure of generated patterns that use dance motions generated after two time steps. It shows that dance motions are generated after gestures are performed after two steps. Dance phrases require four gestures (or two dance motions) and dance sections require eight gestures (or two dance phrases). Of course, if any of the dance patterns (i.e. dance motions, phrases and sections) contain at least one dance motion that is an Opposite skill, then the length (number of gestures) of each dance pattern would be less and variable, for example, an Opposite dance motion would result in a dance phrase of at least two time steps and a dance section would have at least for time steps.

This build up of dance actions (dance motion; dance phrase or dance section) forms the robot's dance into a long continuous series of movements, where each movement follows from the other. However, each dance motion, dance phrase and dance section is stored individually as actions in the action database and treated as individual actions that the robot can select. Each of these is rewarded accordingly and the robot gradually learns which type of dance pattern (i.e. a dance motion, dance phrase or dance section) it should do more often.

The reasoning to implement this dance structure was two-fold. First, this approach allowed repetitions to occur in the robots dance to a degree similar to human dance. Second, a smooth transition could be perceived in the dancing that resembled the way humans dance and organise their motions.

The complete algorithm integrated with reinforcement learning is shown in Algorithm 5.1 and 5.2 below. Algorithm 5.1 shows the main algorithm used for the generation of gestures, while Algorithm 5.2 shows the approach taken to structure and build sequences of the robot's dance motions in dance phrases and dance sections. With reference to reinforcement learning, the *state* of the robot was always on-the-beat (learnt in Chapter 4) and the *action* was any selection the robot performed e.g. a dance motion, dance phrase or dance section. All rewards were internally defined and the robot was always rewarded immediately after each action.

Algorithm 5.1: Generation Of Dance Actions

Initialise parameters

Sarsa parameters: $Q(s,a)$=0, $r \in \Re$ ={1,2,3}, α=0.8, γ=0.2

Softmax parameters: τ =5

Play music

While music is playing

Choose joints & direction (i.e. gesture, G_t) randomly

Search G_t in action database ($action-KB$)

If G_t is in $action-KB$ **Then**

Select action (a_t) from $action-KB$ using Softmax

Perform a_t

Call Algorithm 5.2

Else

Perform gesture (G_t)

If G_t is an Opposite skill **Then**

Add gesture (G_t) to action database ($action-KB$)

Update $action-KB$

$$Q(s_t,a_t) \leftarrow Q(s_t,a_t) + \alpha\left[r + \gamma Q(s_{t+1},a_{t+1}) - Q(s_t,a_t)\right]$$

$$s_t \leftarrow s_{t+1}$$

$$a_t \leftarrow a_{t+1}$$

For all state-action pairs (s,a) Update Softmax

$$\Pr(s_t,a_t) \leftarrow \frac{e^{Q(s_t,a_t)/\tau}}{\sum_{i=1}^{n} e^{Q(s_t,a_i)/\tau}}$$

Else

Cache gesture (G_t)

Choose joints & direction (G_{t+1}) randomly

Check G_t and G_{t+1} is an action in $action-KB$

If an action is Formation or Symmetry skill **Then**

Select action (a_t) with highest Softmax

Perform a_t

Call Algorithm 5.2

Else

Add G_t and G_{t+1} to $action-KB$

Update action database ($action-KB$)

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha[r + \gamma Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)]$$

$s_t \leftarrow s_{t+1}$

$a_t \leftarrow a_{t+1}$

For all state-action pairs (s, a) Update Softmax

$$\Pr(s_t, a_t) \leftarrow \frac{e^{Q(s_t, a_t)/\tau}}{\sum_{i=1}^{n} e^{Q(s_t, a_i)/\tau}}$$

Algorithm 5.2: Structuring A Dance

1. Check a_t is a Dance Motion (DM) or Dance Phrase (DP) in $action - KB$
2. **If** action a_t is DM or DP **Then**
3. Check previous action (a_{t-1}) in $action - KB$
4. **If** a_{t-1} not in $action - KB$ **Then**
5. $a_{t-1} \leftarrow a_t$
6. Call line 4 of Algorithm 5.1
7. **Else**
8. Check dance action type (i.e. DM or DP) for both a_{t-1} and a_t
9. **If** a_{t-1} and a_t are same dance action type **Then**
10. Check a_{t-1} and a_t are an action (a) in $action - KB$
11. **If** action **Then**
12. Update action database ($action - KB$)
$$Q(s_{t-1}, a_{t-1}) \leftarrow Q(s_{t-1}, a_{t-1}) + \alpha[r + \gamma Q(s_t, a_t) - Q(s_{t-1}, a_{t-1})]$$
13. **Else**
14. Group a_{t-1} and a_t as an action (a_{t+1})
15. Write a_{t+1} to $action - KB$
16. Update action database
$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha[r + \gamma Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)]$$
17. **Else**
18. Update action database ($action - KB$)
$$Q(s_{t-1}, a_{t-1}) \leftarrow Q(s_{t-1}, a_{t-1}) + \alpha[r + \gamma Q(s_t, a_t) - Q(s_{t-1}, a_{t-1})]$$
19. Call line 4 of Algorithm 5.1

Algorithm 5.2, checks to see if actions performed (on one or two time steps) are the same or different. Only actions which are of the same dance action type (i.e. actions that are either both dance motions or dance phrases) are grouped together to form longer sequences. For example, it is not possible to group together a dance motion and a dance section as one action. The actions must be the same in order for them to form whole actions. This definition is described internally in the system. The joints that constitute to a skill are also defined in the system.

In these algorithms, $Q(s,a)$ is the Q-factor (value) of doing an action (dance motion, phrase or section) in a state (on-the-beat) at respective time steps; and α and γ are the learning rate and discount factor learning coefficients respectively.

As can be seen in Algorithm 5.1, the algorithm always randomly picks joints to move, however the actual decision as to which joints to move is based on the Softmax calculated probabilities. Softmax ensures that the algorithm does not always pick what it "thinks" is "best", but those that are not also. However, the random selection of joints is still necessary to allow the robot to explore new joint combinations that had not yet been explored.

For this part of the dancing framework, the aim is only to teach the robot what movements are acceptable, regardless of the music. It is an attempt to allow the robot to create its own initial movements, as opposed to providing it with pre-programmed movements. The following section describes the experimental process. The interest here is in exploring the relationships between joints and the number of joints that would make the dance aesthetically pleasing, and not whether or not the dance matches the music, except dancing on-the-beat (which was achieved in Chapter 4).

5.2. Experimental Setup

The aim of this experiment was to determine the aesthetic beauty of dance and for the robot to generate dance patterns to show autonomous behaviour. An online questionnaire containing various videos of the virtual robot dancing was used to retrieve the results.

5.2.1. Experiment Procedure

The robot created several dances using the algorithm, with immediate rewards after each choice of actions, with values 0 for unskilled movements and 1, 1.5 and 2 for skilled movements that formed dance motions, dance phrases and dance sections respectively. Unskilled movements were movements performed without any skill, whereas skilled movements were movements that utilised at least one skill. All actions were stored in an action database as part of the learning. The system was to not only learn to do skilled actions more often but, learn to do more skilled dance sections.

For the online questionnaire, a collection of robot dancing videos was used for three experiments showing the robots dance after learning the skills and different dance actions. All the videos were the same length, showing the robot dancing to the chosen music, which was “Any Dream Will Do” by Jason Donovan.

The first experiment focused on how the number of joints affected the quality of the dance. The second experiment focused on how the different dance skills affected the quality of the dance. The final experiment focused on how a skilled and structured dance affected the quality of the dance.

5.2.2. Data Gathering

All experiments were conducted online and had to be conducted in sequence i.e. beginning with Experiment 1, then proceeding to Experiment 2 and finally to Experiment 3. Figure 5.4 below shows an illustration. Initially, 40 participants began the experiment (Experiment 1), however, this reduced to 18 participants for Experiment 2 and Experiment 3.

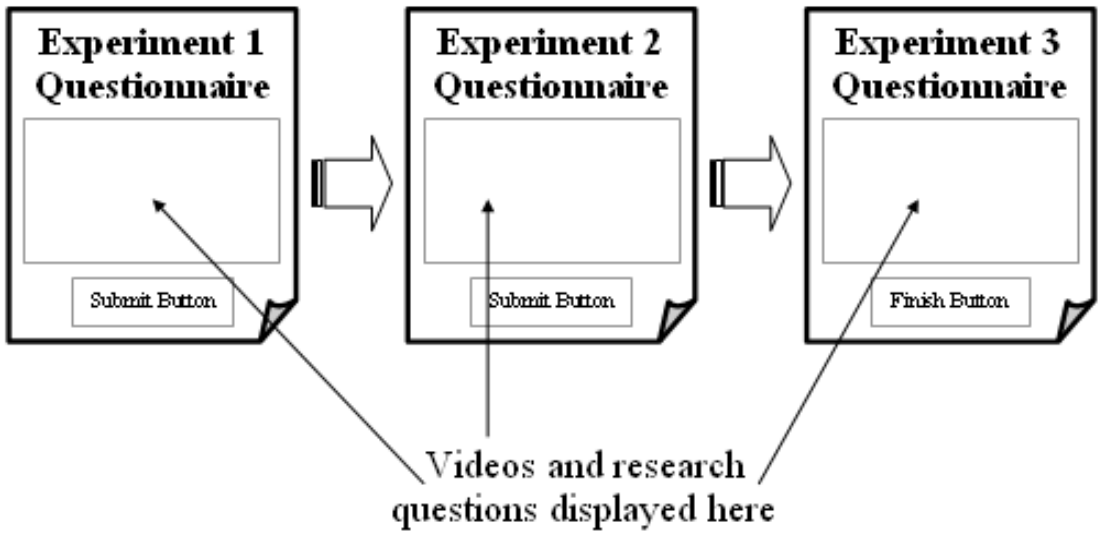


Figure 5.4 – Illustration of online questionnaire

The first experiment studied the relationship between the number of joints and the quality of the dance. In this experiment, the robot's dance movements were constrained by the number of joints that could be used during each beat. Four dances were created using a maximum of 2, 5, 10 and 15 joints for each dance. Participants (40 in total) were asked to rate the quality of each dance using a five point scale, labelled "very poor", "poor", "neutral", "good" and "very good". The quality ratings were then converted into a numerical scale with 1 corresponding to "very poor" and 5 corresponding to "very

good" for the purpose of analysing the results. The participants were also asked to rank the dances giving their favourite dance first place, then second place and so on, down to fourth place. The preference ratings were then converted to scores with 1 being equal to fourth place and 4 being equal to first place. The order of the scale was reversed so that high scores on the performance quality scale could be correlated with high scores on the preference scale.

The second experiment focused on the relationship between different skills used and the quality of the dance. Four dances were created. The robot was programmed to apply a single skill (i.e. a dance containing 100% of each skill – Opposite, Symmetry and Formation) in each of the first three dances. The robot was then programmed to consider equally all three skills in the forth dance. The same participants were asked to rate the quality of the dance and rank their preferences like they did for Experiment 1. Additionally, participants were asked about the difficulty in identifying the skill or skills being demonstrated in the dances using a four point scale labelled: "really easy", "easy", "hard", and "really hard". These ratings were then converted (as above) to a score of 1 to 4 with 1 for "really hard", 2 for "hard", 3 for "easy", and 4 for "very easy". This way the higher rating was given a higher score and used in the analysis.

The third experiment focused on assessing the quality of the dances made up of *random*, *unskilled* and *skilled* movements. In the first dance (random), the robot's movements were performed on the beat, but they did not form skilled motions (i.e. motions did not contain Opposite, Symmetry or Formation). Neither was the robot programmed to form dance motions, phrases nor sections. The robot simply moved joints at random to the beat and no learning was implemented.

In the second dance (unskilled), the robot's movements were synchronised to the beat, but skills were not done. However, there was learning that is, unskilled dance motions formed dance phrases, which in turn, formed unskilled dance sections. In the final dance (skilled), the robot's movements were programmed to use skilled dance motions, which in turn formed skilled dance phrases and skilled dance sections, all synchronised to the music. The participants (18 in total) were again asked to provide preferences and ratings of the quality of the dances as described above. Table 5.2 below summarises the experiments, and the variables that were captured.

Experiment	Dance Type	Dance Performance Rating	Dance Preference Rating	Skill Identifications Rating
1. **Number of Joints** (40 participants)	2 joints	1 = Very Poor 5 = Very Good	1 = Lowest 4 = Highest	N/A
	5 joints	1 = Very Poor 5 = Very Good	1 = Lowest 4 = Highest	N/A
	10 joints	1 = Very Poor 5 = Very Good	1 = Lowest 4 = Highest	N/A
	15 joints	1 = Very Poor 5 = Very Good	1 = Lowest 4 = Highest	N/A
2. **Movement Skills** (18 participants)	100% Opposite	1 = Very Poor 5 = Very Good	1 = Lowest 4 = Highest	1 = Really Hard 4 = Really Easy
	100% Symmetry	1 = Very Poor 5 = Very Good	1 = Lowest 4 = Highest	1 = Really Hard 4 = Really Easy
	100% Formation	1 = Very Poor 5 = Very Good	1 = Lowest 4 = Highest	1 = Really Hard 4 = Really Easy
	33% for each skill	1 = Very Poor 5 = Very Good	1 = Lowest 4 = Highest	1 = Really Hard 4 = Really Easy
3. **Dance Styles** (18 participants)	Random	1 = Very Poor 5 = Very Good	1 = Lowest 3 = Highest	N/A
	Unskilled	1 = Very Poor 5 = Very Good	1 = Lowest 3 = Highest	N/A
	Skilled	1 = Very Poor 5 = Very Good	1 = Lowest 3 = Highest	N/A

Table 5.2 - Experiments and the variables captured

5.3. Results & Observations

5.3.1. Algorithm Analysis

In reference to the algorithm, the average reward over the number of movements (i.e. each selection of dance motions, dance phrases and dance sections) were recorded. Figure 5.5 below shows the results.

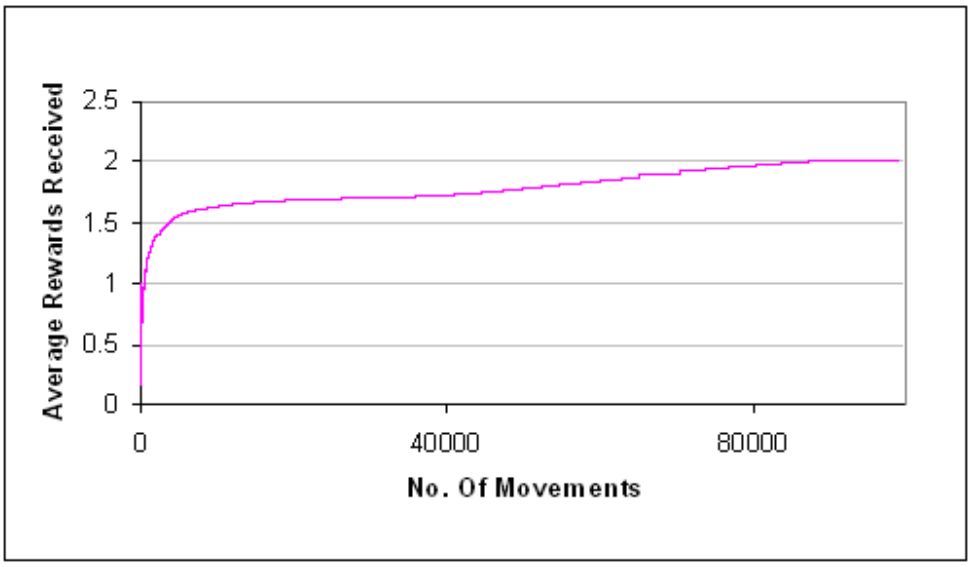


Figure 5.5 – Learning performance of algorithm in five trials (runs)

Recall that after each choice of actions, the robot was rewarded immediately with values 0 for unskilled movements and 1, 1.5 and 2 for skilled movements that formed dance motions, dance phrases and dance sections respectively. From Figure 5.5, we can see that initially, the algorithm explored both skilled and unskilled actions, but then gradually learnt to do skilled actions. The robot progressed to doing skilled dance phrases and dance actions for some time, before eventually exploiting dance sections most of the time. Note that in this experiment, no single action was desired and so the algorithm could select any action to perform, but patterns of skilled dance sections were more desirable. This result shows that the algorithm can learn to do dance sections.

5.3.2. Experiment 1: Results & Analysis

In reference to the questionnaire, for Experiment 1 (how the performance is affected by the number of joints), the ratings for each dance were averaged for both the participant's thoughts on the quality of the dancing and their preferences. Specifically, participants were asked to rate how good each dance was (dance performance) and rank the dances in order (dance preference), Figure 5.6 below shows the results, whereby higher quality values represent a greater satisfaction and preference of the dances.

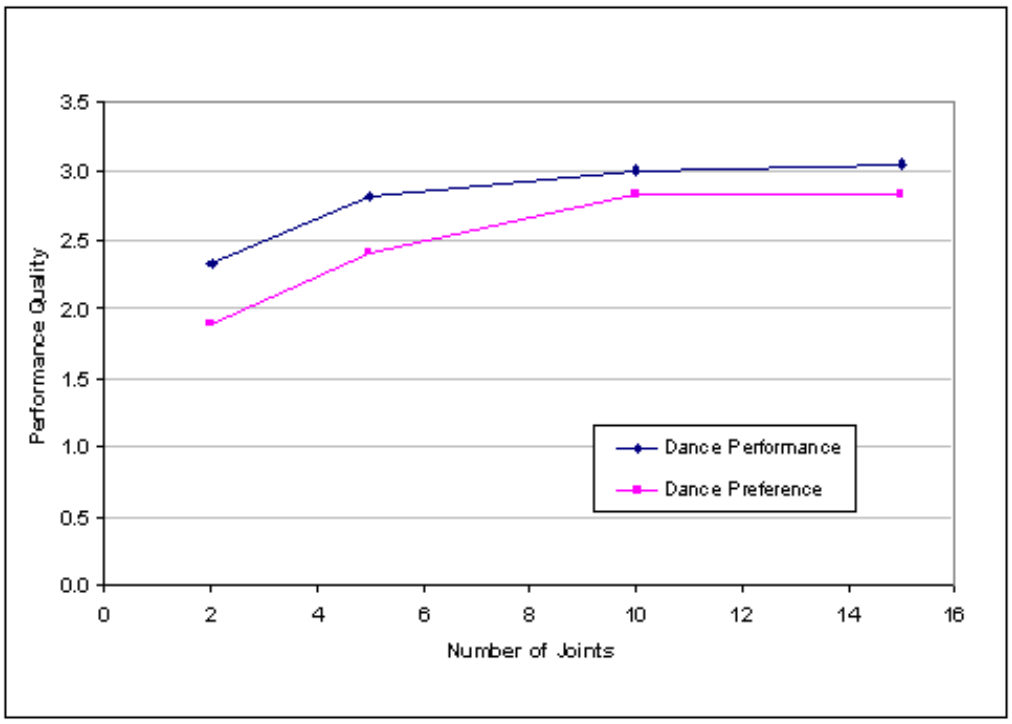


Figure 5.6 – Perceived dance quality vs. number of joints moved

Looking at Figure 5.6, we see that both the perceptions of the dance performance and the dance preferences increases as the number of joints is increased, and so therefore, it appears that increasing the number of joints is correlated to a higher perception of quality. In other words, it appears to be that participants favoured higher joint numbers

than lower ones; although, it is interesting that the maximum rating for dance performances (15 joints) was not that much different from 10 joints, and after 10 joints, the preference of the robots dancing was the same. These facts suggest that the quality of the dance would have levelled up and converged to the same quality if more and more joints were added. For example, even though it is shown that participants preferred the dances with increasing number of joints, there was not much difference in how they perceived the dances. A possible explanation for this could be because participants were looking at how the robot's dancing matched the music, rather than what effect the number of joints had on the dance. In other words, they were looking for dynamics within the movements. Second, the robot was programmed to remain in one position (i.e. 'sleeping' position on the ground) to stop it from falling down and losing its balance, and therefore could not get up and move around. This might have affected their decisions when judging the robots dancing. Nevertheless, it can be concluded that the perceived quality of the dance increased as the number of joints that the robot was allowed to use increased, but possibly up to an unknown limit.

The question, however, is whether or not there is significant improvement in the robots performance between the dances generated. Using the 2-tail sample t-test statistical approach to compare each of the alternative groups, one pair at a time, the hypothesis of each comparison was **there is a significant difference between alternative groups**.

The details of the tests at 95% (p-value < 0.05) confidence are summarised below, with Table 5.3 showing the statistics for participant dance preferences and Table 5.4 showing the statistics for participant dance satisfaction.

		n	Mean	Standard Deviation	Variance	*df*	p-value	Conclusion	
Dance Comparisons	**2**	40	1.895	1.331	1.772	78	0.015	H_1	5 joints is better preferred
	5	40	2.4737	0.6035	0.3642	78			
	2	40	1.895	1.331	1.772	78	<0.001	H_1	10 joints is better preferred
	10	40	2.895	0.689	0.475	78			
	2	40	1.895	1.331	1.772	78	0.001	H_1	15 joints is better preferred
	15	40	2.895	1.331	1.772	78			
	5	40	2.4737	0.6035	0.3642	78	0.005	H_1	10 joints is better preferred
	10	40	2.895	0.689	0.475	78			
	5	40	2.4737	0.6035	0.3642	78	0.073	H_0	No Significant difference
	15	40	2.895	1.331	1.772	78			
	10	40	2.895	0.689	0.475	78	1.000	H_0	No Significant difference
	15	40	2.895	1.331	1.772	78			

Table 5.3 - Experiment 1: Descriptive statistics for participant preferences between comparisons among dances with number of joints alternatives

		n	Mean	Standard Deviation	Variance	*df*	p-value	Conclusion	
Dance Comparisons	**2**	40	2.351	1.230	1.512	78	0.130	H_0	No Significant difference
	5	40	2.811	1.351	1.824	78			
	2	40	2.351	1.230	1.512	78	0.043	H_1	Significant difference
	10	40	3.027	1.572	2.471	78			
	2	40	2.351	1.230	1.512	78	0.052	H_0	No Significant difference
	15	40	3.054	1.794	3.219	78			
	5	40	2.811	1.351	1.824	78	0.531	H_0	No Significant difference
	10	40	3.027	1.572	2.471	78			
	5	40	2.811	1.351	1.824	78	0.511	H_0	No Significant difference
	15	40	3.054	1.794	3.219	78			
	10	40	3.027	1.572	2.471	78	0.944	H_0	No Significant difference
	15	40	3.054	1.794	3.219	78			

Table 5.4 - Experiment 1: Statistics for participant satisfaction ratings between comparisons among dances with number of joints alternatives, where *n* is the number of participants and *df* is the degrees of feedback

The same participants were used to assess the groups (dances). In order for there to be any significance difference between groups, the calculated probability (p-value) would have to be less than the *significant level* (0.05). The results summarised in the above table provides evidence to suggest that the participants generally preferred a dance with more joints although statistically, the level of improvement (or not) between dances was minimal.

5.3.3. Experiment 2: Results & Analysis

For Experiment 2 (how the performance is affected by skills used), the ratings for each dance were also averaged for both the participant's thoughts on the quality of the dancing and their preferences. In this case, the results are shown as a bar chart since the dances were not based on an ordinal scale. Figure 5.7 below shows the results.

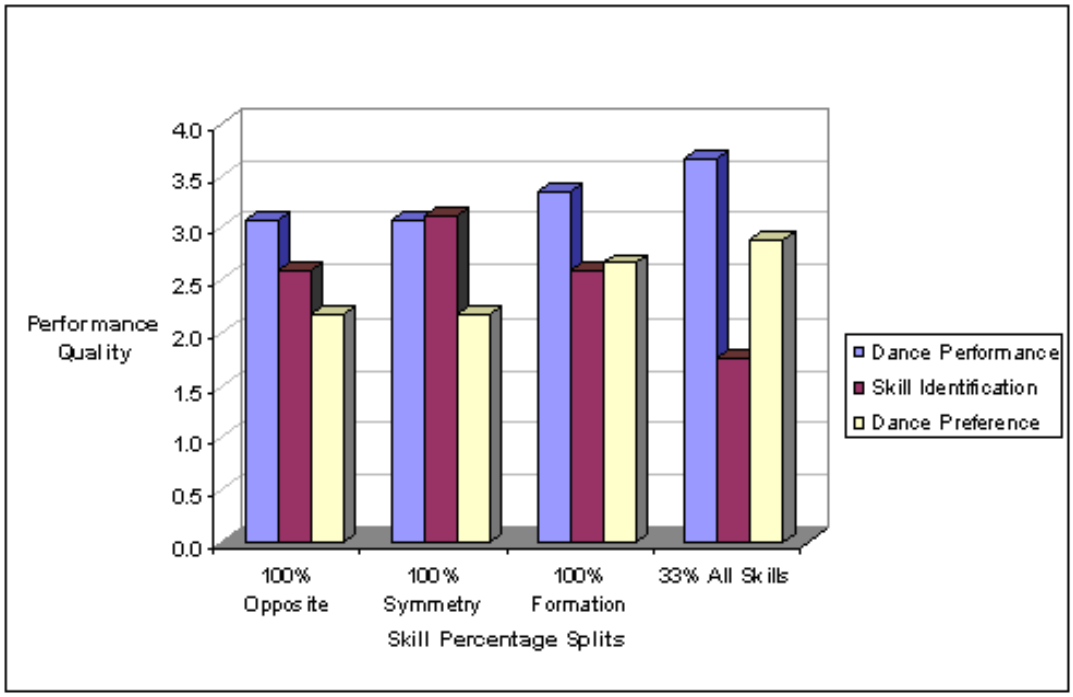


Figure 5.7 – Dance quality vs. skilled dances

Figure 5.7, shows that by integrating all three of the skills (33% of each skill), higher marks are received in both the overall performance and the personal preferences of the participants. However, this skill set received the lowest mark for skill identification. This result makes sense since it would be more difficult for an observer to identify all three skills (Opposite, Symmetry and Formation) of the dance simultaneously in this dance, whereas the other three dances only required the participant to identify a single skill. The average dance performance for each skill independently was slightly varied, suggesting that each of these skills were relevant to determine the quality of the dance, but on their own, the dances were "not that bad". However, the average preference rating slightly changed with Formation being the highest compared to the other individual skilled dances. This suggests that, taken alone, Formation skilled dance movements may be slightly more significant to the quality of the dance, followed by Symmetry and Opposite skills. This again makes sense, since according to dance studies, it is the form (Formation) of a dance that is the most contributing factor of a dance.

This is supported by the statistical results obtained, shown below in Table 5.5 – 5.7. Using the 2-tail sample t-test statistical approach to compare each of the alternative groups the hypothesis of each comparison was again: **There is a significant difference**.

Group Comparisons	***n***	**Mean**	**Standard Deviation**	**Variance**	***df***	**Conclusion**	
100% Formation	18	2.667	1.085	1.176	34	**H_0**	No Significant difference
100% Opposite	18	2.167	1.043	1.088	34		
100% Formation	18	2.667	1.085	1.176	34	**H_0**	No Significant difference
100% Symmetry	18	2.167	0.857	0.735	34		
100% Formation	18	2.667	1.085	1.176	34	**H_0**	No Significant difference
33% for each skill	18	2.889	1.278	1.634	34		
100% Opposite	18	2.167	1.043	1.088	34	**H_0**	No Significant

100% Symmetry	18	2.167	0.857	0.735	34		difference
100% Opposite	18	2.167	1.043	1.088	34	**H_0**	No Significant difference
33% for each skill	18	2.889	1.278	1.634	34		
100% Symmetry	18	2.167	0.857	0.735	34	**H_0**	No Significant difference
33% for each skill	18	2.889	1.278	1.634	34		

Table 5.5 - Experiment 2: Descriptive statistics for participant preferences between comparisons among dances with skills alternatives, where *n* is the number of participants and *df* is the degrees of feedback

Group Comparisons	*n*	Mean	Standard Deviation	Variance	*df*	Conclusion	
100% Formation	18	3.333	0.900	0.810	34	**H_0**	No Significant difference
100% Opposite	18	3.067	0.799	0.638	34		
100% Formation	18	3.333	0.900	0.810	34	**H_0**	No Significant difference
100% Symmetry	18	3.067	0.799	0.638	34		
100% Formation	18	3.333	0.900	0.810	34	**H_0**	No Significant difference
33% for each skill	18	3.667	1.345	1.810	34		
100% Opposite	18	3.067	0.799	0.638	34	**H_0**	No Significant difference
100% Symmetry	18	3.067	0.799	0.638	34		
100% Opposite	18	3.067	0.799	0.638	34	**H_0**	No Significant difference
33% for each skill	18	3.667	1.345	1.810	34		
100% Symmetry	18	3.067	0.799	0.638	34	**H_0**	No Significant difference
33% for each skill	18	3.667	1.345	1.810	34		

Table 5.6 - Experiment 2: Descriptive statistics for participant satisfaction ratings between comparisons among dances with skills alternatives, where *n* is the number of participants and *df* is the degrees of feedback.

Group Comparisons	*n*	Mean	Standard Deviation	Variance	*df*	Conclusion	
100% Formation	18	2.625	0.806	0.650	34	**H_0**	No Significant difference
100% Opposite	18	2.563	0.727	0.529	34		
100% Formation	18	2.625	0.806	0.650	34	**H_0**	No Significant difference
100% Symmetry	18	3.063	0.574	0.329	34		
100% Formation	18	2.625	0.806	0.650	34	**H_1**	Formation is more easier to identify
33% for each skill	18	1.750	0.856	0.733	34		
100% Opposite	18	2.563	0.727	0.529	34	**H_1**	Significant difference
100% Symmetry	18	3.063	0.574	0.329	34		
100% Opposite	18	2.563	0.727	0.529	34	**H_1**	Opposite is

33% for each skill	18	1.750	0.856	0.733	34		more easier to identify
100% Symmetry	18	3.063	0.574	0.329	34	**H_1**	Symmetry is more easier to identify
33% for each skill	18	1.750	0.856	0.733	34		

Table 5.7 - Experiment 2: Descriptive statistics for participant difficulty ratings between comparisons among dances with skills alternatives, where *n* is the number of participants and *df* is the degrees of feedback.

Tables 5.5 – 5.7 show the details of the analysis with confidence levels of 95%. Most of the comparisons support the null hypothesis i.e. no significant difference between alternatives in the robots performance (Table 5.6) and the personal preferences of the participants (Table 5.5). However, it can be concluded that it was harder to identify individual skills in a dance consisting of all the skills combined (Table 5.7), than a dance that only contained the individual skills.

5.3.4. Experiment 3: Results & Analysis

Finally, Figure 5.8 shows the result for experiment 3 (how the performance is affected by skilled and structured dances) and again, the ratings for each dance were averaged for both the participant's thoughts on the quality of the dancing and their preferences.

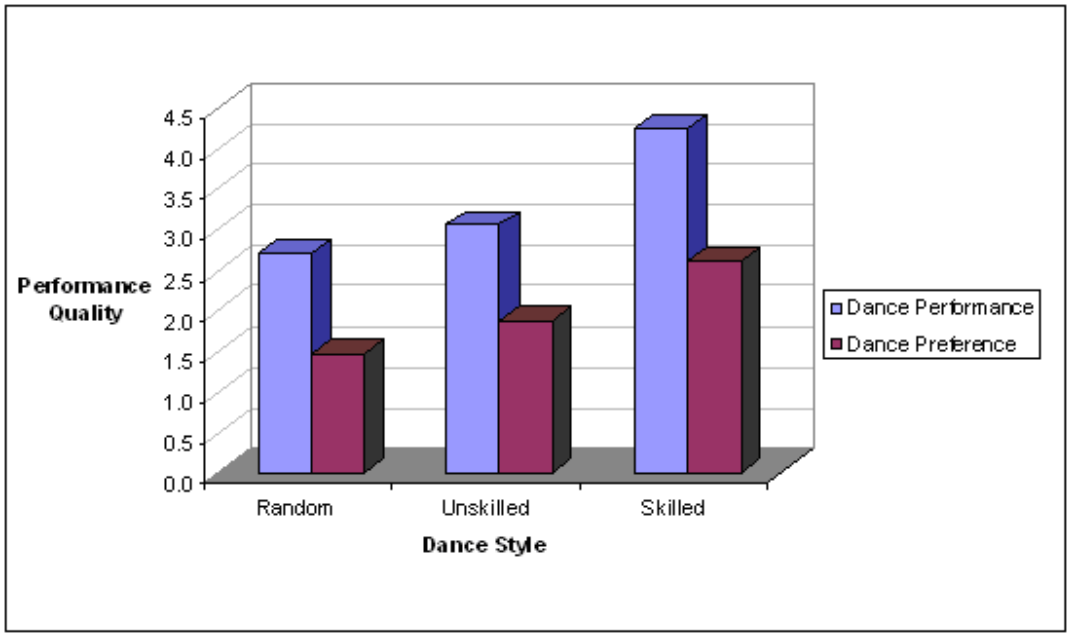


Figure 5.8 – Dance quality vs. dance style

From Figure 5.8, the result was that the skilled dance received the highest marks for both overall performance and personal preference, followed by the unskilled dance and then the random dance. This indicates that learning to dance with skilled movements creates a dance with a higher perceived quality (aesthetic appearance) than learning to dance with no skills and a dance that has no learning and makes use of no skills. It is also interesting that the difference between a random and an unskilled dance was less than the difference between an unskilled and a skilled dance. This suggests that, what is fundamentally required for dance is that dance has a skilled structure (i.e. skilled dance motions that form skilled dance phrases, that in turn form skilled dance sections), and that an unskilled dance was almost as bad as random dancing, but better because it had structure to it. A structured dance was more important than simply picking any action to any beat. Dance must have structure to it. The use of skills only made the dance look "better". This was supported statistically as well. Using the same hypothesis and

confidence level as Experiment 1 and 2, Table 5.8 and Table 5.9 below summaries the results for each comparison between the alternate groups.

Group Comparisons	*n*	Mean	Standard Deviation	Variance	*df*	p-value	Conclusion	
Random	18	1.471	0.624	0.390	34	<0.001	H_1	Skilled is highly preferred
Skilled	18	2.706	0.686	0.471	34			
Random	18	1.471	0.624	0.390	34	0.086	H_0	No significant difference
Unskilled	18	1.824	0.529	0.279	34			
Skilled	18	2.706	0.686	0.471	34	<0.001	H_1	Skilled is highly preferred
Unskilled	18	1.824	0.529	0.279	34			

Table 5.8 - Experiment 3: Statistics for participant preferences between dances types, where *n* is the number of participants and *df* is the degrees of feedback.

Group Comparisons	*n*	Mean	Standard Deviation	Variance	*df*	p-value	Conclusion	
Random	18	2.706	0.920	0.846	34	<0.001	H_1	Skilled is a better dance
Skilled	18	4.235	0.831	0.691	34			
Random	18	2.706	0.920	0.846	34	0.329	H_0	No significant difference
Unskilled	18	3.059	1.144	1.309	34			
Skilled	18	4.235	0.831	0.691	34	0.002	H_1	Skilled is a better dance
Unskilled	18	3.059	1.144	1.309	34			

Table 5.9 - Experiment 3: Statistics for participant performance ratings between dances types, where *n* is the number of participants and *df* is the degrees of feedback.

Looking at the above tables, we see that participants agreed that a skilled dance that had structure to it was “better” than an unskilled or random dance and therefore, had more preference for it.

5.4. Summary

The idea of the robot exploring primitive gestures in order to generate and keep a record of desirable dance motions based on reinforcement learning was explored in this chapter. Gestures combine with previous gestures to form dance motions and a structure of the dance. This approach has lead to satisfactory results without the need for more complex reinforcement learning methods.

To conclude, since dancing is subjective, the final measure of success for this stage of the research has proven subjective. In general, the results validate the hypothesis that applying synchronous, structured movements (skilled and unskilled dancing), produces a higher quality dance than simply creating random moves, even if the random moves are synchronised to the music. Furthermore, skills are needed in order to improve the dance and provide a means to judge the robots dancing. The build-up of dance motions, dance phrases and dance sections not only provided structure to the robot's dancing, but also encouraged the use of repetition of an appreciable value in the dancing, which would have also influenced the opinions of participants.

It is interesting that most of the participants were able to successfully distinguish between what they considered a "poor" dance performance versus a "good" dance performance, even though the appearance of the robot in this stage of the robots development was not humanoid. This implies that it is possible to give robots dance aesthetics that can be identified by human partners, and even though humanoid robots resemble the human body, their kinematical structure are not necessary for people to recognise creativity in a robot dance, even though the dance itself may be rather abstract when compared to how humans dance.

The results in this stage of the dancing model are encouraging and justify the need for skills and a definition (structure) of dance, and the statistical analysis of the results have also supported this. Since the robot is now capable of structuring a dance and producing desirable dance steps, the next logical step would be to have the robot's dance evaluated by humans. This development idea is explored in the next chapter.

Chapter 6
Adapting Dance To Human Preferences

In Chapter 5, the approach explored gave the robot a basic repertoire of dance actions and dance sequences suitable for it to generate its own dance. However, the robot did not learn through direct interaction with human observers. As humans learn to dance based on observing the dances of others or under a dance instructor, it makes sense to teach a robot to dance with the involvement of people.

In this chapter, the idea of adapting a robot's dance steps to match the preferred movements of human subjects is explored. The chapter follows on from and incorporates the behaviours learnt from the previous two chapters, using *interactive reinforcement learning* (IRL) to build a database of appropriate preferred dance moves and improve its dancing based on human preferences. This stage of development determines whether learning from human partners can significantly improve the robots dance.

6.1. Methodology

There are three main considerations for the robot's adaptive behaviour to human preferences. First, human subjects need to inform the robot the parts they liked and disliked in the robot's dance without too many reward inputs. Second, the robot needs to be capable of generating diversity in its movements based on the preferences. Third, the system requires a way of evaluating and making use of the feedback given to influence the generation of new dances. For the first consideration, the Webots interface receives external feedback (via keyboard on a laptop) from participants, while the robot is dancing in real-time to the music. Participants indicate their preferred and non-preferred dance

movements by pressing the appropriate keys. This is then interpreted by the system as values of +1 and -1 for preferred and non-preferred dance movements respectively.

For the second consideration, in order to ensure that the robot would produce diverse movements, the robot is programmed to change its position at random from "sleeping" position to "sitting" position and vice versa, and in each of these positions it is limited to perform certain gestures. This is so that the robot would maintain balance while dancing, but also incorporate varying movements in its dancing. All movements performed are skilled movements learnt in Chapter 5. The robot is programmed to select a maximum of 15 joints at a time (although this can be varied depending on the robot's posture e.g. being in "sleeping" or "sitting" position), and any number of joint combinations (gestures) and skilled movements can be formed. The robot can also select the directions in which joints can be moved.

For the third consideration, the idea of *buffering* and *pattern matching* are explored. These provide a way to capture and compare the preferences based on repeated preferences and their rewarded values. All preferred movements are given a reward of +1 and non-preferred movements given a reward of -1. However during the processing of the feedback, different matched dance sequences are weighed differently based on the total rewards received for each preference. This is to weigh more preferred movements higher than less preferred movements. This will be described in more detail in the next sections.

6.1.1 Buffering

In Chapter 2, two approaches to adaptive robotic dance movements based on human preferences were discussed. These were *interactive evolutionary computation* (IEC) and *interactive reinforcement learning* (IRL). One consideration that was pointed out was how to determine the desirability of preferences. The approach taken in this research is to store all the preferences of each participant and then the robot would only perform those actions that were rewarded by the participant at least twice and had total reward values greater than zero.

When the observer provides feedback to the robot, it is possible that the robot will receive the feedback late. For example, the observer may like a movement and decide to provide a positive feedback but do so just as the robot is performing the next dance movement. In this case, the wrong dance movement will have received the positive feedback causing the robot to learn the wrong behaviour. The same result could be true of negative feedback. Given these conditions, it will be difficult for the robot to determine dance actions which are rewarded (or punished) incorrectly. The observer can be equipped with different inputs to represent different feedback for example, different keys to represent different reward/punishment values for different observations. However, this will increase the number of inputs the observer has to remember. Therefore, the challenge is to reward/ punish actions as required without adding additional load on the observer. As a solution, the idea of *buffering* is used in this research.

Buffering refers to the process of keeping a record of all the dance motions that were performed prior to the most recent feedback given and after the previous feedback.

The buffer is a cache of *n* number of dance actions which constitute a sequence. Each time the robot receives feedback, the last *n* dance actions are saved in the buffer. This is illustrated in figure 6.1 below.

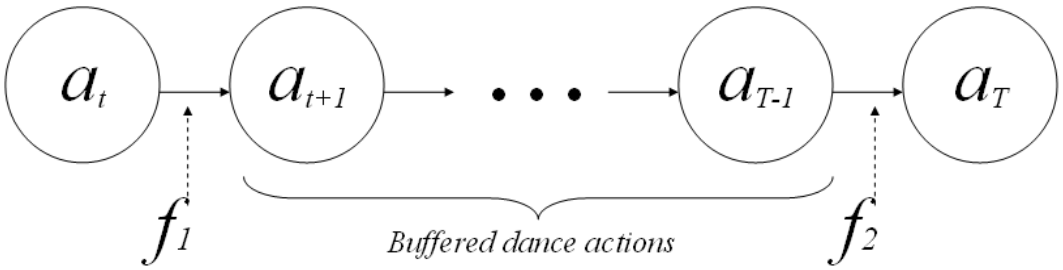


Figure 6.1 – Illustration of buffered sequences, where *a* is a dance motion performed at time *t* up until the most recent time (*T*) and *f* is the feedback given. Here, f_2 denotes the feedback that followed the previous feedback (f_1).

Effectively, the robot associates the entire sequence with the feedback (reward/ punish) given. The number of dance motions of a sequence that can be stored in the buffer can be any value equal to 1 or above. The complete history of buffered sequences performed by the robot is stored in the *preference database* along with the reward given for each buffed sequence. Figure 6.2 shows the overall system and the buffered dance actions being stored into the preference database.

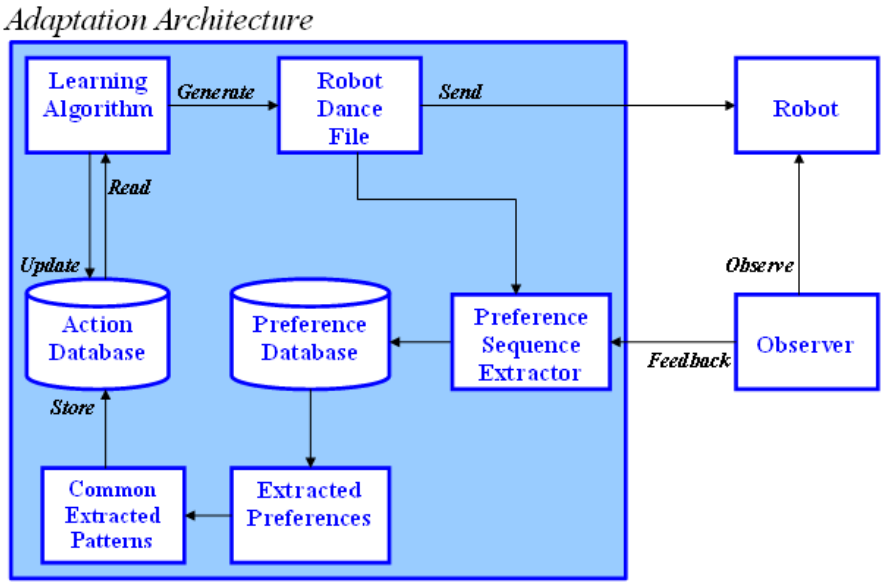


Figure 6.2 – Buffering of dance actions based on human feedback

The process begins by first giving the robot a generated dance (produced autonomously from Chapter 5), which it performs as the music is playing. While the robot is dancing, the observer's chosen preferences are captured by the *preference sequence extractor* and stored temporarily in the *preference database* alongside with the reward given to be further processed within the *adaptation architecture*. The *preference sequence extractor* matches the robot's gestures with the history of gestures sequenced in the *robot dance file* so that the correct arrangement of dance motions are recorded and stored in the *preference database*. Once all preferences have been captured and stored, each stored sequence is reduced to at most, the *n* number of dance motions to cache as predefined by the trainer. This is where the *buffering* takes place. The buffered sequences are then compared to obtain any common patterns (sequence of dance motions) among them. Found matches (common patterns) are then extracted and stored in the action database for the learning algorithm to select and update the action database, generating a new dance file after each selection.

6.1.2 Pattern Matching

The observer's preferences are analysed by comparing buffered sequences in the preference database, to find patterns within each sequence. Specifically, if a dance motion (or series of dance motions) that is part of a buffered sequence is found in another buffered sequence and the total sum of the patterns is greater than zero, then the robot would put those movements in its *action database* so that those dance motions can be selected when the robot generates the new dance. Figure 6.3 shows the dataflow of the pattern matching system.

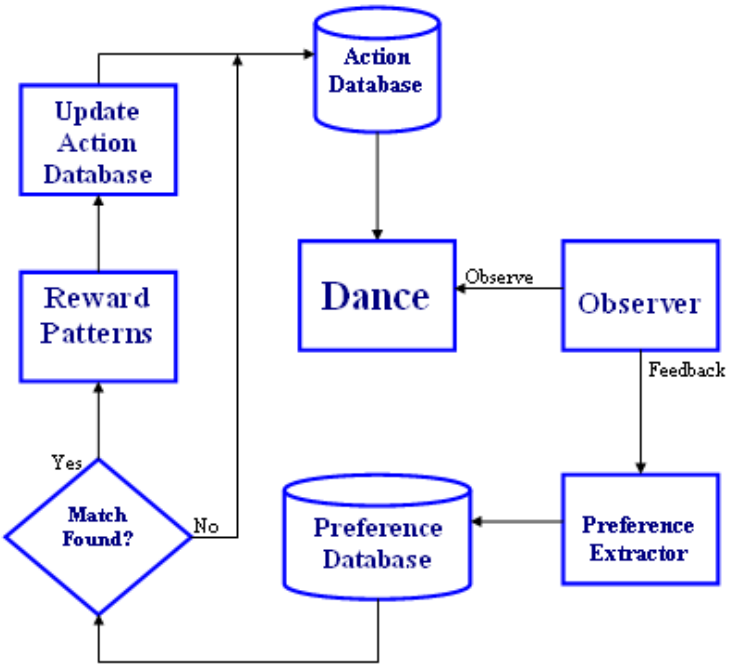


Figure 6.3 - Pattern matching of trainer preferences

The system takes into account the possibility that not all actions would be preferred at the same level by the observer. If a reward had been given, the entire feedback sequence would be stored in the preference database. In the preference database, the robot would search each sequence to see if a dance motion (or series of

dance motions) had appeared more than once. If a match is then found, the action database would be updated to record the matched sequences as complete actions for the robot to perform. For example, if the observer preferred the same pattern of movements twice then the pattern would be promoted for selection for the next dance after being stored in the *action database*. A new dance would be generated based on all the actions in the action database and their rewarded values. On the other hand, if the rewarded dance motions had only received feedback once, i.e. no match was found, then the feedback would be ignored and the movements would not be promoted for selection for the next dance. For example, consider three buffered sequences below. A dance motion is indicated as 'A#' and the transition from one dance motion to another is indicated by an arrow '→':

1. A1→A2→A3→A4
2. A10→A11→A12→A13→A1→A2
3. A2→A3

Assuming the maximum buffer size is 6. Beginning with the third buffered sequence, A2→A3 for example, each sub-sequence in sequence 1 and sequence 2 would be searched to see if they first contained the dance motions A2 and A3, and then contained the pattern A2-A3. The buffering process is shown in Table 6.1.

Sequence No.	Original Pattern	Sequence No.	Searched Buffered Sequences	Match?
3	A2→A3	2	A10→A11→A12→A13→***A1→A2***	No
			A10→A11→***A12→A13***→A1→A2	No
			A10→***A11→A12***→A13→A1→A2	No
			A10→A11→A12→A13→A1→A2	No
		1	A1→A2→***A3→A4***	No
			A1→***A2→A3***→A4	Yes
			A1→A2→A3→A4	No

Table 6.1 - Pattern matching example.

Here, the bold and italic blue colouring indicates the sub-sequence being matched with the original pattern. There is one match in the above example. Therefore the sequence of dance motions in transit (A2→A3) and the individual dance motions themselves that make up the sequence (dance motion A2 and dance motion A3) would be promoted to the action database for selection.

The idea of pattern matching each participant's preferences is so that the movements that the observer(s) did not like are not immediately discarded in case the actions are incorrectly or accidentally rewarded. Patterns which have more than one match are more likely to be selected to generate the robots next dance. However, the reward associated with each pattern (sub-sequence) determines which patterns should be selected more often in the robot's dance. If non-preferred movements appear in the robot's next generated dance, then it would more likely not be selected as more often as preferred moves. Observers can then have another opportunity to further fine-tune the robot's dance in the next generated dance, indicating their dislike for unwanted moves and encouraging preferred dance steps. This approach has two benefits. First, by performing pattern matching, it produces possible preferences because only repeated patterns are selected by the robot, leaving patterns which appear once to be discarded.

Second, by limiting sequences to buffered sequences, the possible preferences are narrowed down.

6.1.3 Learning From Human Partners

Observers can provide feedback on dance moves they do and do not prefer and they are weighted as positive (+1) and negative (-1) reinforcements respectively. So in other words, observers do not have the option to rate their preferences based on a scale of values. This is so that the observers are not overloaded with deciding which value on the scale to use. Instead, the approach taken is to weigh the preferences by summing all subsequent rewards given for matched sequence patterns. For example, if the sequence A2→A3 is matched in three other sequences and for each sequence a positive (+1) reinforcement is given, then the total reward for the action A2→A3 will be +4, i.e. when the sequence is first performed, the robot receives a reward of +1. This pattern is then matched with other sequences each rewarded of +1, which will total the overall reward of the matched sequence to +4. This therefore indicates how much the chosen action is preferred. Consequently, more highly rewarded dance motions (and sequences) encourage the robot to do the dance movements more often than those which are rated not as highly. This means that some movements are exploited more than others. All movements are given a chance to be performed, but in the case where observers prefer not to see particular movements at all, this is only achieved if the total reward value for preferences is negative. The criteria for this was as follows:

- If the sum of matched preferences is greater than or equal to zero, then the dance sequence will be assigned a probability value (i.e. a Softmax value for action-selection) and therefore will have a chance of being performed during the next interactive training session.

- If on the other hand, the sum of matched preferences is less than zero, then a probability (Softmax) value will not be calculated for it, in which case, the movement will no longer be performed or chosen as a movement in future interactive training sessions. The dance sequence is recorded in a form of "blacklist" so that it doesn't get picked.

The reasoning behind this was first to produce dynamic rewards for matched dance motions, so that robots determine the desirability of preferences. Second, even if movements received contradictory feedback, the outcome will be determined by the overall summed value of positive and negative preferences. For example, a movement that received one negative (-1) and two positive (+1) responses (totalling +1) would still have a possibility of being chosen in the future, but not as likely as a movement that received three positive (+1) responses (totalling +3), whereas a movement receiving two negative (-1) and one positive (+1) response would not be performed at all since it would total -1.

The complete algorithm used for this system is illustrated below in Algorithm 6.1. Here, $m(sF_t)$ and $m(sF_{t+1})$ denote the number of dance motions compared in each subsequent sub-sequence respectively.

Algorithm 6.1: Adapting To Human Feedback

1. Initialise parameters: $Q(s_t, a_t) = 0$, state (s_t) = beat; buffer size; time step (k)
2. Play music
3. **While** music is playing
4. Read in predefined dance sequence
5. **If** observer input (r_f) then
6. Cache number of dance actions (n) as a feedback sequence (F_t)
7. Save the sequence (F_t) to preference database
8. **End While**
9. Clear action database
10. **Repeat** (for each feedback sequence (F_t))
11. Initialise parameters:
Number of dance motions in dance sequence ($m(sF_t) = 1$) at time t
Number of dance motions in dance sequence ($m(sF_{t+1}) = 1$) at time t+1
12. **While** $m(sF_t)$ < buffer size + 1
13. Action $a_t \leftarrow sF_t$
14. **Repeat** (for each sub-sequence (sF_{t+1}) in next feedback sequence (F_{t+1}))
15. $m(sF_{t+1}) \leftarrow m(sF_t)$
16. **While** $m(sF_{t+1})$ < buffer size + 1
17. Action $a_{t+1} \leftarrow sF_{t+2}$
18. **If** a_t and a_{t+1} are the same
19. **If** a_t in action database
20. $R(a_t) \leftarrow R(a_t) + R(a_{t+1})$
21. **Else**
22. Store a_t in action database
23. $R(a_t) \leftarrow R(a_t) + R(a_{t+1})$
24. **Else**
25. **If** $m(sF_{t+1}) <=$ buffer size
26. Observe next sub-sequence (F_{t+2})
27. $sF_{t+1} \leftarrow sF_{t+2}$
28. **End Repeat**
29. Increment $m(sF_t)$
30. **End While**
31. Choose action (a_t) from action database (using Softmax)
32. **Repeat** (until k is reached)
33.
34. Take action (a_t) from action database, observe r
35. Add a_t to new dance sequence
36. Choose next action (a_{t+1}) (using Softmax)
37. $Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha[r + \gamma Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)]$
$s_t \leftarrow s_{t+1}; a_t \leftarrow a_{t+1}$
38. **End Repeat**
39. **End Repeat**

6.1.4 Feedback From Multiple Observers

The above approach mimics the idea of learning to dance based on the feedback provided by one observer. This idea has already been discussed in the current state-of-the-art on adaptive robot dance in Chapter 2, but feedback can come from multiple observers as well. The approach taken in this chapter supports not just the feedback from single observers but also multiple observers. This is achieved by storing and performing pattern matching on all the feedbacks provided. The preference values are summed in the same way as described above to give dance sequences different weights based on the feedbacks from all observers.

6.2. Experiment Procedure

The central music piece for all the dances shown to the participants was "Clarity" by John Mayer. This was chosen because of its simple beat structure as well as considered to be an appropriate song for people of different preferred music genres, although this was not important since the purpose of this next stage in the robots development was only to observe the working nature of the implemented adaptive algorithm on human input.

Three experiments were conducted to explore the effects of human feedback on the robot's dance. Each experiment had ten participants take part. The experiments were conducted over two consecutive days, with Experiment 1 and 2 being performed first on day 1 and Experiment 3 being done on day 2. The same participants were used in all three of the experiments.

6.2.1. Experiment 1

For Experiment 1, the robot produced one random skilled dance, which was shown to all participants who then observed the dance and provided their preferences on it. Each participant was then required to observe the newly generated dances which incorporated varying percentages of their own preferences, combined with the varying percentages of another participant who had observed the same initial random dance. No two participant preferences were combined with the same participant. Figure 6.4 shows an illustration.

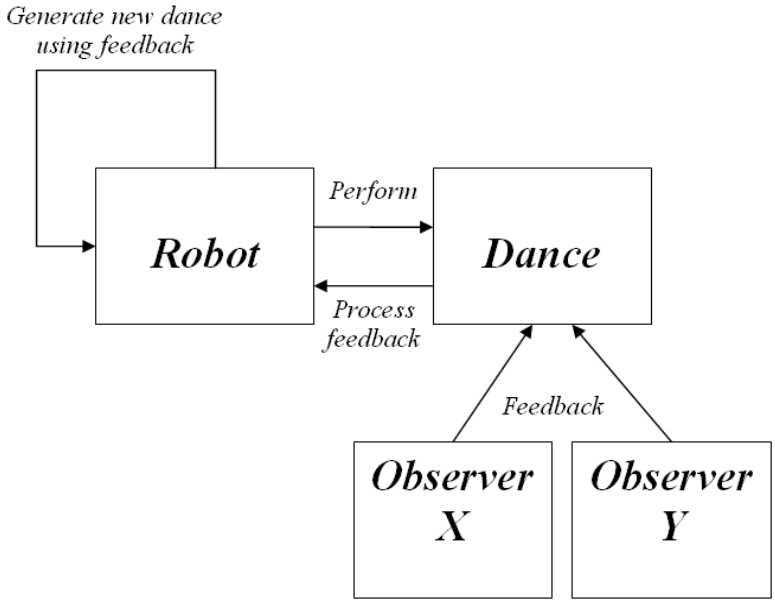


Figure 6.4 – Illustration of Experiment 1 with two different observers. For each participant (Observer X), no two participants were combined with another participant (Observer Y)

6.2.2. Experiment 2

Here, the robot produced a different random skilled dance for each participant. The participants observed their respective random dances and provided their preferences from it. As in Experiment 1, each participant was then required to observe the newly generated dances which incorporated varying percentages of their own preferences, combined with the varying percentages of preferences from another participant who had observed a

different initial random dance. Like Experiment 1, no two participant's preferences were combined with the same participant. Figure 6.5 shows an illustration.

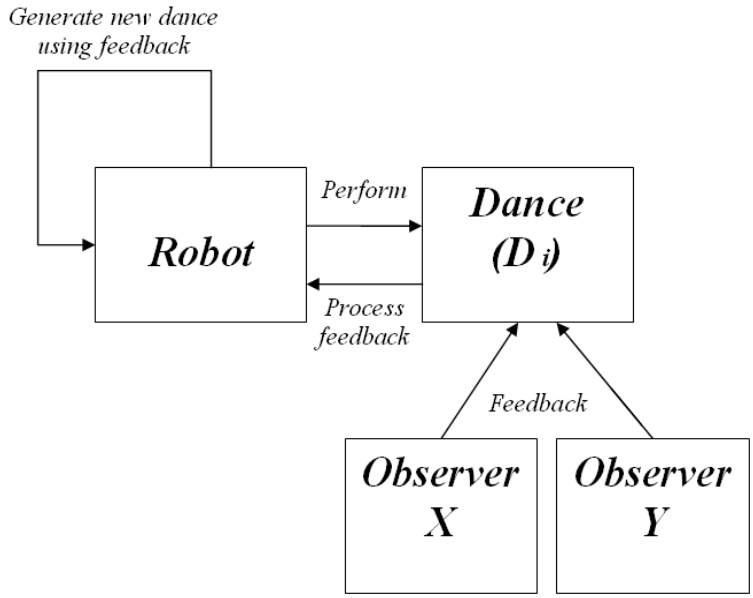


Figure 6.5 – Illustration of Experiment 2 with two different observers. Here, each participant observed a different initial random dance (*Di*) distinguished by *i.*

For both Experiments 1 and 2, the root's dance movements and the participants' choices, were constrained by the number of participant preference combinations. Five different dances were generated which incorporated varying percentages of preferences from the participant taking part in the experiment and the preferences saved from another participant. Participant preferences were divided as follows:

- **100**% - this was a newly generated dance, based entirely on the participant's own preferences.
- **75**% - this was a newly generated dance that contained 75% of the participant's own preferences and 25% of the preferences from another participant.

- **50**% - this was a newly generated dance that contained 50% of the participant's own preferences and 50% of the preferences from another participant.
- **25**% - this was a newly generated dance that contained 25% of the participant's own preferences and 75% of the preferences from another participant.
- **0**% - this was a newly generated dance that contained none of the participant's own preferences and 100% of the preferences from another participant.

These newly generated dances were compared to the initial random dance and observed by the participant to determine how satisfied the participant was with the dances.

Figure 6.6 shows an example illustration of the percentage splits. These combinations were done to determine if any perceived improvements were arbitrary or actually attributed to the preferences of participants. All participants gave a minimum of 15 preferences, therefore, only the first 15 preferences were selected for dance combinations so that the preferences were not biased because different individuals gave different amounts of feedback.

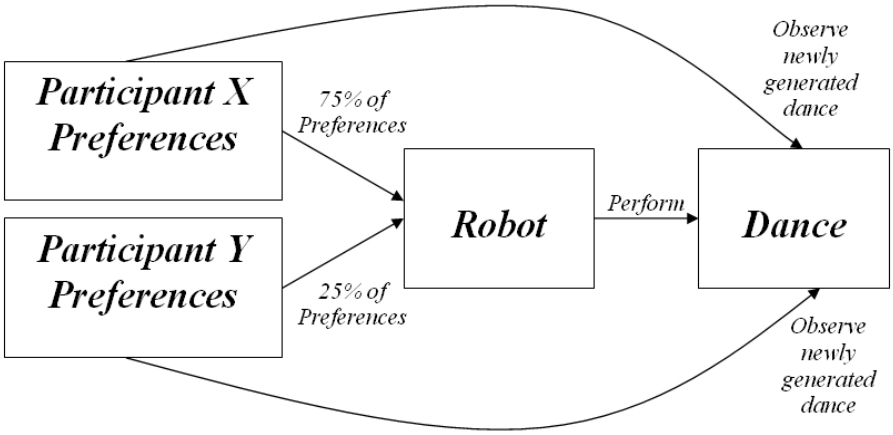


Figure 6.6 – Example illustration of a dance generated with 75% of one participant's preferences and 25% of another participant's preferences.

6.2.3. Experiment 3

In this final experiment, the robot produced one random skilled dance for each participant (as in Experiment 2). For each participant who took part in the experiment, after observing the robot's initial random dance, three other dances were shown to them. The first of the three dances shown was a newly generated random dance and the other two dances were combinations of the preferences of other participants. In all the three dances, none of them contained the participant's own preferences. The robots dance movements were also constrained (as in Experiments 1 and 2) by the number of participant preference combinations. Preferences for the dances were divided as follows:

- **100**% - this was a newly generated dance, based entirely on 100% of another participant's own feedback.
- **50**% - this was a newly generated dance that contained 50% of the preferences of two other participants' feedback.

These dances were compared to the initial random dance observed by the participant to determine how satisfied the participant was with the dances. Figure 6.7 shows an example illustration.

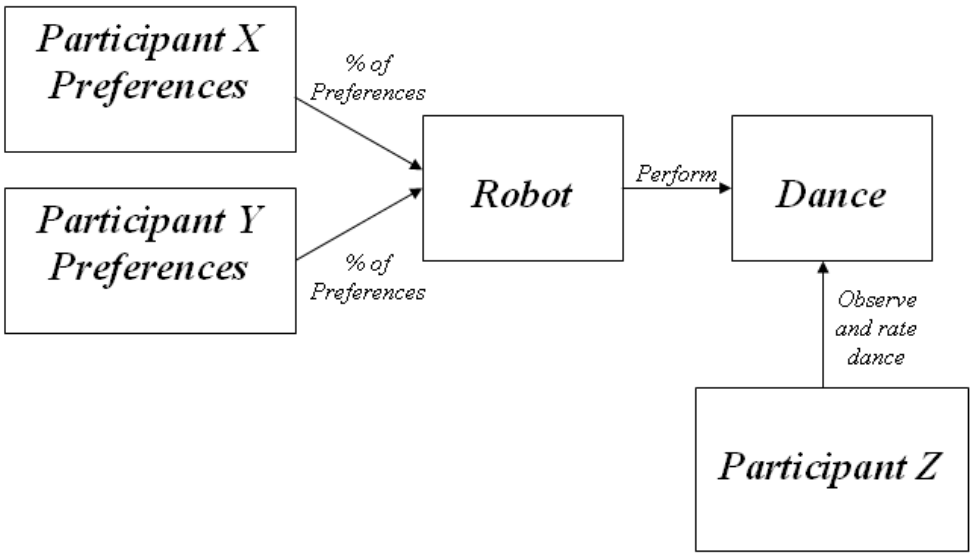


Figure 6.7 – Illustration of experiment 3

Each participant was asked to view each of the generated dances in all three experiments and provide their judgement on how satisfied they were with each dance. The participants were unaware of which of the robot's dances were a result of their own preferences or if the robot had undergone any training at all. This was to determine how well the robot responded to the preferences and improved in its dancing.

6.2.4. Data Gathering

The main research question for these experiments was whether dances got 'better' as a function of training. Another question was whether increasing the number of trainers from zero to two increased the quality of the dances generated. For Experiments 1 and 2, participants were given a questionnaire and were specifically asked three questions for each of the dances they observed:

1. **Dance satisfaction** – how satisfied were you with the robot's dance?
2. **Preferences followed** – how well did the newly generated dance incorporate the preferences you selected from the initial random dance?

3. **Number of newly combined preferred dance moves** – how many newly generated combinations did you prefer?

Figure 6.8 shows an example of one of the dances shown to participants and the above questions shown on the online questionnaire.

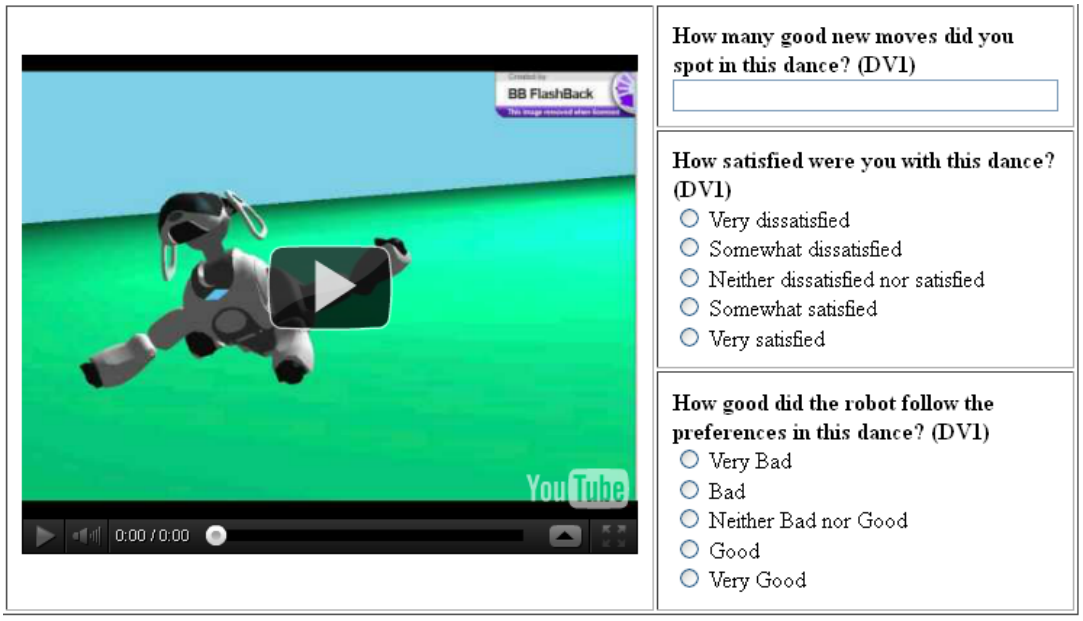


Figure 6.8 – Screenshot of one of the dances shown to participants and the questions asked on the online questionnaire

For question 1, responses were based on a five point ordinal scale with the labels "very dissatisfied", "somewhat dissatisfied", "neither dissatisfied nor satisfied", "somewhat satisfied" and "very satisfied". For question 2 the labels for the five point scale were "very bad", "bad", "neither bad nor good", "good" and "very good". The qualitative ratings were then converted into a numerical scale with 1 corresponding to "very dissatisfied" and "very bad", and 5 corresponding to "very satisfied" and "very good" for the purpose of analysing the results. The participants were also asked to rank the six dances (including the random dance) into order of preference from 1 to 6.

Question 3 was asked in Experiments 1, 2 and 3. A count of the total number of newly preferred dance action combinations that participants observed (compared to the initial random dance) for each dance except in the initial random dance was required. It would have been very difficult for participants to keep count of preferred moves whilst observing the dances, so this was achieved with the help of the experiment controller, who kept a count as each participant, indicated their preference.

Experiment 3 also required that participants kept a count of the total number of newly generated dance actions that were not preferred. Table 6.2 below summarises the experiments showing the number of people participated in each experiment and the variables that were captured.

Experiment	Research Question	Dance Performance Rating
1 & 2 (10 participants)	1. How satisfied were you with the robot's dance?	1 = Very Dissatisfied 5 = Very Satisfied
	2. How well did the newly generated dances incorporate the preferences you selected from the initial random dance?	1 = Very Bad 5 = Very Good
1 & 2 & 3 (10 participants)	3. How many newly generated combinations did you prefer in each dance?	N/A
3 (10 participants)	4. How many newly generated combinations did you not prefer in each dance?	N/A

Table 6.2 - Experiments and the variables captured

Ten participants took part in all three experiments. The participants were from different professions and aged 18 to 40. None of them had any experience of interacting with robots. When asked, "How would you rate your own level of dancing", all described their dancing ability as "casual".

6.3. Results & Observations

The aim of Experiments 1 and 2 were to determine if combining a participant's preferences with another participant's preferences in a dance made a difference in the robot's performance. For Experiment 3, the aim was to determine if it was better for the

robot to learn from other people than through random exploration. For Experiment 1, the hypothesis was that there would not be much difference in the satisfaction rating after observing dances that included dances that resulted from a mix of preferences from another participant. For Experiment 2, the hypothesis was that combined dances (of two participants) would improve to a much better quality (satisfaction) dance after observing different initial dances because of the increase in variation of dance motions in the robot's dancing. For Experiment 3, the hypothesis was that participants would prefer a dance that contained preferences from at least one other participant compared to a random dance from the robot, and that an increase in participant preferences (from no trainers to two trainers) would increase the quality of the dance.

6.3.1. Experiments 1 & 2: Results & Analysis

Experiment 1 and 2 were first compared to determine the robot's performance. The ratings for each dance on both experiments were averaged on how satisfied the participants felt about each dance. Figure 6.9 below shows the average satisfaction ratings for each dance for both experiments.

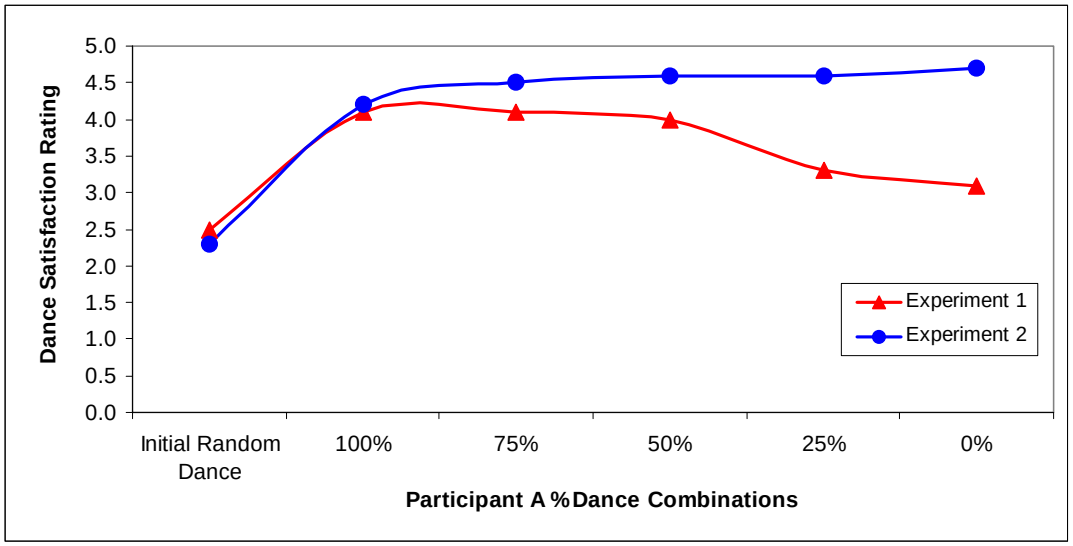


Figure 6.9 – Participant satisfaction rating for Experiments 1 and 2

Figure 6.9 shows that in Experiment 1, the participants rated dances that incorporated 100% of their own preferences higher than the initial random dance they observed, and higher than any of the dances that incorporated less than 100% of their own preferences. In Experiment 2 on the other hand, combining the preferences from another participant improved the level of satisfaction among participants. This suggests that the improvements in the dances were actually caused by the preferences of participants rather than just through random exploration. The results also support the hypothesis that for Experiment 1, participants would prefer to rate their own preferences higher than preferences suggested by another participant, and for Experiment 2 participants would be more satisfied with increasing combinations from another participant.

For Experiments 1 and 2, at the "100%" dance, there was a clear improvement when compared to the initial random dance. However, this improvement was reduced for Experiment 1, but was increased for Experiment 2. The explanation is that in Experiment

1, each participant viewed the same initial (random) dance, so there was no or little diversity in dance steps. In Experiment 2, each participant viewed a different initial dance, which increased the diversity in dance steps.

Each experiment was performed by showing dances to participants in descending order of their preferences. That is, after beginning with a "Random" dance, the next dance shown to participants was a "100%" dance, followed by a "75%" dance, then a "50%" dance, then a "25%" dance and finally a "0%" dance. For this reason in Experiment 1, it could be that participants had too much of their preferences being repeated. After watching dances of repeated dance steps, participants would quickly become bored of seeing the same dance steps, thus making the dances less satisfactory overall compared to Experiment 2, where there was more diversity in dance steps, hence increasing the level of satisfaction.

Figure 6.10 below shows us that, on average, for Experiment 1, the robot had actually incorporated the feedback suggested by each participant observing the dances, whilst in Experiment 2, the robot did not. Recall that *satisfaction* was a measure of how satisfied participants felt on each dance, and *preference* was a measure of how well the observer felt their preferences were followed.

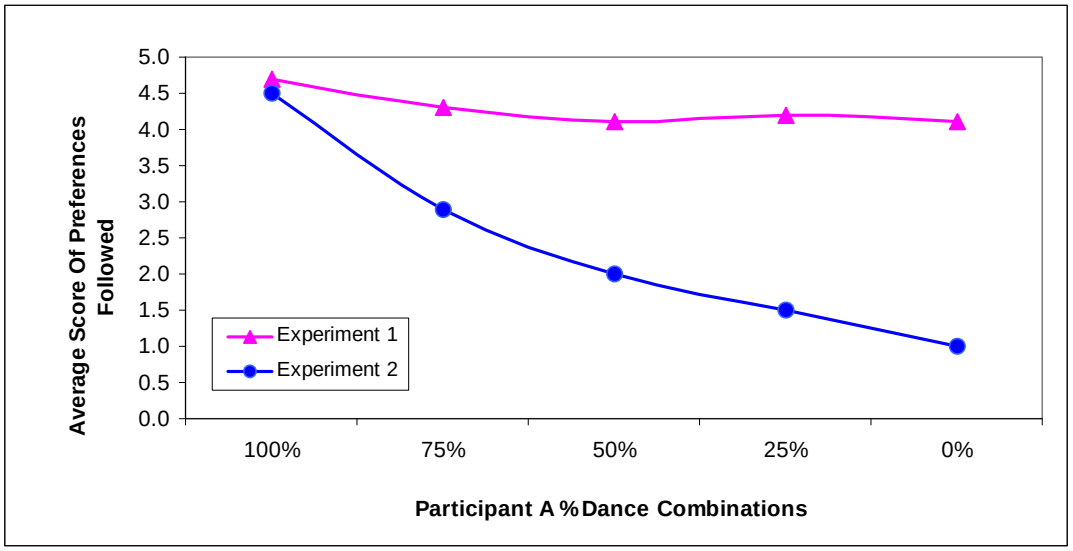


Figure 6.10 – Score of how well participants felt their preferences were followed for Experiments 1 and 2

The y-axis here is a rated measure of how well the participants felt that their preferences were performed in each of the observed dances (x-axis). In both Experiments 1 and 2, the decrease in rating was directly related to the decrease in the participant's own preferences, with Experiment 2 having the biggest decrease in comparison. Like Figure 6.9, this also supported the hypothesis that the participants' increase in rating was as a result of preferences combined in each dance and not a result of random movements.

Participants were also asked to count (with the help of the experiment controller) the number of perceived newly performed dance steps they preferred for each subsequent dance after observing the initial random dance. Figure 6.11 below shows the average number of newly performed dance combinations (to the nearest 10) for each observed dance.

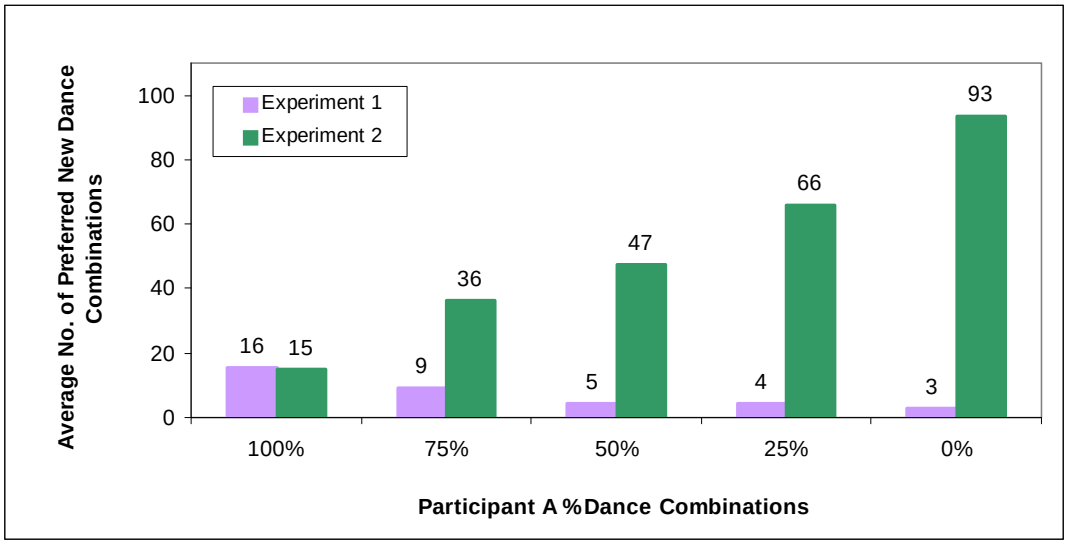


Figure 6.11 – Average score of the number of newly generated preferred combinations for Experiments 1 & 2

As described above, in Experiment 1 there was a clear preference for dances that included 100% of the participant's preferences. In Figure 6.11, as the percentage of that participant's preferences decreased, the perception of preferred newly generated dance combinations decreased. Experiment 2 on the other hand, had a much higher perception of newly performed dance combinations. These results were as expected. This indicates that a dance which combines the preferences of others who have observed different original dances, improves the quality of the dance by producing more favourable dance steps, compared to dances which have combinations from participants who have observed the same original dance.

Finally, participants were asked to rank the six dances that they had viewed in order of preference, with 1 being the best dance and 6 being the worst dance. Table 6.3 below shows the results.

Dance Type		Experiment 1	Experiment 2
Participant A preferences	**Participant B preferences**		
100%	0%	1	5
75%	25%	2	4
50%	50%	3	3
25%	75%	4	1
0%	100%	5	2
Initial Random Dance		6	6

Table 6.3 - Dance preferences rankings for Experiments 1 and 2, where 1 indicates the most preferred dance and 6 indicates the least preferred dance

Taking the average of all participants' preference ranking for the dances, in both Experiment 1 and 2, Table 6.3 tells us that as the inclusion of the participant's own preferences increased in Experiment 1, so did their ranking of the generated dances. Each subsequent dance continued to be ranked higher than the initial random dance. In Experiment 2, the reverse was the case. The inclusion of another participant's own preferences was more preferable. But in generally, any user feedback is better than the initial random dance that the robot began with.

As further analysis, the percentage of each dance group in both Experiment 1 and 2 was compared to determine if the measure of dance quality was significantly different for different levels of participant feedback. In each comparison, the following hypothesis was tested: "**The quality of the dance is significantly different**". A p-test with a critical alpha of 0.05 (95% confidence) was used to determine if the difference in ratings for a particular percentage split was significant.

(1) Experiment 1 Statistical Result

Beginning with Experiment 1, Tables 6.4, 6.5, and 6.6 below summarise the p-test data for the three measures of dance quality - satisfaction rating, number of preferences followed, and number of newly preferred generated moves observed. For each measure of quality, the score for the higher percentage of participant input was compared to the score for quality for the lower percentage. If the p-value was less than 0.05, then the proposed hypothesis was accepted.

Experiment 1 – Satisfaction Comparison							
Group	***n***	**Mean**	***sd***	**Variance**	***df***	**p-value**	**Conclusion**
Random vs.	10	2.5	0.707	5.0	18	<0.001	A dance with 100% of preferences is better than a random dance
100%	10	4.0	0.738	0.544	18		
Random vs.	10	2.5	0.707	5.0	18	<0.001	A dance with 75% of preferences is better than a random dance
75%	10	4.125	0.738	0.544	18		
Random vs.	10	2.5	0.707	5.0	18	<0.001	A dance with 50% of preferences is better than a random dance
50%	10	4.0	0.816	0.667	18		
Random vs.	10	2.5	0.707	5.0	18	0.019	A dance with 25% of preferences is better than a random dance
25%	10	3.375	0.675	0.456	18		
Random vs.	10	2.5	0.707	5.0	18	0.080	No difference
0%	10	3.125	0.738	0.544	18		
100% vs.	10	4.0	0.738	0.544	18	1.0	No difference
75%	10	4.125	0.738	0.544	18		
100% vs.	10	4.0	0.738	0.544	18	0.777	No difference
50%	10	4.0	0.816	0.667	18		
100% vs.	10	4.0	0.738	0.544	18	0.021	A dance with 100% of preferences is better than a dance with 25% of preferences
25%	10	3.375	0.675	0.456	18		
100% vs.	10	4.0	0.738	0.544	18	0.007	A dance with 100% of preferences is better than a dance with no preferences
0%	10	3.125	0.738	0.544	18		
75% vs.	10	4.125	0.738	0.544	18	0.777	No difference
50%	10	4.0	0.816	0.667	18		
75% vs.	10	4.125	0.738	0.544	18	0.021	A dance with 75% of preferences is better than a dance with 25% of preferences
25%	10	3.375	0.675	0.456	18		
75% vs.	10	4.125	0.738	0.544	18	0.007	A dance with 75% of preferences

0%	10	3.125	0.738	0.544	18		is better than a dance with no preferences
50% vs.	10	4.0	0.816	0.667	18	0.051	No difference
25%	10	3.375	0.675	0.456	18		
50% vs.	10	4.0	0.816	0.667	18	0.019	A dance with 50% of preferences is better than a dance with no preferences
0%	10	3.125	0.738	0.544	18		
25% vs.	10	3.375	0.675	0.456	18	0.535	No difference
0%	10	3.125	0.738	0.544	18		

Table 6.4 – Experiment 1 dance satisfaction. Here, *n* is the number of participants; *sd* is the standard deviation and *df* is the degrees of freedom

Experiment 1 – Preferences Followed Comparison							
Group	***n***	**Mean**	***sd***	**Variance**	***df***	**p-value**	**Conclusion**
100% vs.	10	4.875	0.483	0.233	18	0.202	No difference
75%	10	4.250	0.823	0.678	18		
100% vs.	10	4.875	0.483	0.233	18	0.045	Significant difference
50%	10	4.125	0.738	0.544	18		
100% vs.	10	4.875	0.483	0.233	18	0.105	No difference
25%	10	4.0	0.789	0.622	18		
100% vs.	10	4.875	0.483	0.233	18	0.074	No difference
0%	10	3.875	0.876	0.767	18		
75% vs.	10	4.250	0.823	0.678	18	0.574	No difference
50%	10	4.125	0.738	0.544	18		
75% vs.	10	4.250	0.823	0.678	18	0.785	No difference
25%	10	4.0	0.789	0.622	18		
75% vs.	10	4.250	0.823	0.678	18	0.605	No difference
0%	10	3.875	0.876	0.767	18		
50% vs.	10	4.125	0.738	0.544	18	0.773	No difference
25%	10	4.0	0.789	0.622	18		
50% vs.	10	4.125	0.738	0.544	18	1.000	No difference
0%	10	3.875	0.876	0.767	18		
25% vs.	10	4.0	0.789	0.622	18	0.791	No difference
0%	10	3.875	0.876	0.767	18		

Table 6.5 – Experiment 1 dance preferences followed. Here, *n* is the number of participants; *sd* is the standard deviation and *df* is the degrees of freedom

Experiment 1 – Newly Preferred Generated Moves Comparison							
Group	***n***	**Mean**	***sd***	**Variance**	***df***	**p-value**	**Conclusion**
100% vs.	10	15.6	9.430	88.933	18	0.109	No Significant difference
75%	10	9.3	7.119	50.678	18		
100% vs.	10	15.6	9.430	88.933	18	0.003	Significant difference
50%	10	4.5	3.923	15.389	18		
100% vs.	10	15.6	9.430	88.933	18	0.002	Significant difference
25%	10	4.2	3.155	9.956	18		
100% vs.	10	15.6	9.430	88.933	18	0.001	Significant difference
0%	10	3.1	2.470	6.100	18		
75% vs.	10	9.3	7.119	50.678	18	0.078	No Significant difference
50%	10	3.1	3.923	15.389	18		
75% vs.	10	9.3	7.119	50.678	18	0.053	No Significant difference
25%	10	4.2	3.155	9.956	18		
75% vs.	10	9.3	7.119	50.678	18	0.018	Significant difference
0%	10	3.1	2.470	6.100	18		
50% vs.	10	4.5	3.923	15.389	18	0.853	No Significant difference
25%	10	4.2	3.155	9.956	18		
50% vs.	10	4.5	3.923	15.389	18	0.352	No Significant difference
0%	10	3.1	2.470	6.100	18		
25% vs.	10	4.2	3.155	9.956	18	0.397	No Significant difference
0%	10	3.1	2.470	6.100	18		

Table 6.6 – Experiment 1 newly preferred generated moves. Here, *n* is the number of participants; *sd* is the standard deviation and *df* is the degrees of freedom

Note that out of all three measures, "satisfaction" (Table 6.4) was the only measure that allowed a comparison between the rating of the "Random" dance and the next generated dance that incorporated various levels of participant preferences. This was not necessary for the other measures of dance quality i.e. the preferences followed measure and the newly preferred generated moves measure, as it would not be possible to make such comparisons.

(2) Experiment 2 Statistical Result

The same comparisons were made for Experiment 2. Recall that Experiment 2 differed from Experiment 1 in that it required participants to observe different initial dances, as opposed to the same initial dance from the robot. Tables 6.7 – 6.9 below summarise the p-test data for the three measures of dance quality, which were satisfaction rating, number of preferences followed, and the number of newly preferred generated moves observed by the participant.

Experiment 2 – Satisfaction Comparison							
Group	***n***	**Mean**	***sd***	**Variance**	***df***	**p-value**	**Conclusion**
Random vs.	10	2.3	0.823	0.678	18	<0.001	A dance with 100% of preferences is better than a random dance
100%	10	4.2	1.033	1.067	18		
Random vs.	10	2.3	0.823	0.678	18	<0.001	A dance with 75% of preferences is better than a random dance
75%	10	4.5	0.707	0.5	18		
Random vs.	10	2.3	0.823	0.678	18	<0.001	A dance with 50% of preferences is better than a random dance
50%	10	4.6	0.516	0.267	18		
Random vs.	10	2.3	0.823	0.678	18	<0.001	A dance with 25% of preferences is better than a random dance
25%	10	4.6	0.516	0.267	18		
Random vs.	10	2.3	0.823	0.678	18	<0.001	A dance with 0% of preferences is better than a random dance
0%	10	4.7	0.483	0.233	18		
100% vs.	10	4.2	1.033	1.067	18	0.458	No Significant difference
75%	10	4.5	0.707	0.5	18		
100% vs.	10	4.2	1.033	1.067	18	0.288	No Significant difference
50%	10	4.6	0.516	0.267	18		
100% vs.	10	4.2	1.033	1.067	18	0.288	No Significant difference
25%	10	4.6	0.516	0.267	18		
100% vs.	10	4.2	1.033	1.067	18	0.182	No Significant difference
0%	10	4.7	0.483	0.233	18		

75% vs.	10	4.5	0.707	0.5	18	0.722	No Significant difference
50%	10	4.6	0.516	0.267	18		
75% vs.	10	4.5	0.707	0.5	18	0.722	No Significant difference
25%	10	4.6	0.516	0.267	18		
75% vs.	10	4.5	0.707	0.5	18	0.470	No Significant difference
0%	10	4.7	0.483	0.233	18		
50% vs.	10	4.6	0.516	0.267	18	1.00	No Significant difference
25%	10	4.6	0.516	0.267	18		
50% vs.	10	4.6	0.516	0.267	18	0.660	No Significant difference
0%	10	4.7	0.483	0.233	18		
25% vs.	10	4.6	0.516	0.267	18	0.660	No Significant difference
0%	10	4.7	0.483	0.233	18		

Table 6.7 – Experiment 2 dance satisfaction. Here, *n* is the number of participants; *sd* is the standard deviation and *df* is the degrees of freedom

Experiment 2 – Preferences Followed Comparison							
Group	***n***	**Mean**	***sd***	**Variance**	***df***	**p-value**	**Conclusion**
100% vs.	10	4.5	0.527	0.278	18	<0.001	Significant difference
75%	10	2.9	0.316	0.10	18		
100% vs.	10	4.5	0.527	0.278	18	<0.001	Significant difference
50%	10	2.0	0.471	0.222	18		
100% vs.	10	4.5	0.527	0.278	18	<0.001	Significant difference
25%	10	1.5	0.527	0.278	18		
100% vs.	10	4.5	0.527	0.278	18	<0.001	Significant difference
0%	10	1.0	0.0	0.0	18		
75% vs.	10	2.9	0.316	0.10	18	<0.001	Significant difference
50%	10	2.0	0.471	0.222	18		
75% vs.	10	2.9	0.316	0.10	18	<0.001	Significant difference
25%	10	1.5	0.527	0.278	18		
75% vs.	10	2.9	0.316	0.10	18	<0.001	Significant difference
0%	10	1.0	0.0	0.0	18		
50% vs.	10	2.0	0.471	0.222	18	0.038	Significant difference
25%	10	1.5	0.527	0.278	18		
50% vs.	10	2.0	0.471	0.222	18	<0.001	Significant difference
0%	10	1.0	0.0	0.0	18		
25% vs.	10	1.5	0.527	0.278	18	0.008	Significant difference
0%	10	1.0	0.0	0.0	18		

Table 6.8 – Experiment 2 dance preferences followed. Here, *n* is the number of participants; *sd* is the standard deviation and *df* is the degrees of freedom

Experiment 2 – Newly Preferred Generated Moves Comparison							
Group	***n***	**Mean**	***sd***	**Variance**	***df***	**p-value**	**Conclusion**
100% vs.	10	14.9	8.412	70.767	18	0.006	Significant difference
75%	10	36.4	20.173	406.933	18		
100% vs.	10	14.9	8.412	70.767	18	<0.001	Significant difference
50%	10	47.3	22.301	497.344	18		
100% vs.	10	14.9	8.412	70.767	18	<0.001	Significant difference
25%	10	65.90	21.429	459.211	18		
100% vs.	10	14.9	8.412	70.767	18	<0.001	Significant difference
0%	10	93.40	14.630	214.044	18		
75% vs.	10	36.4	20.173	406.933	18	0.267	No Significant difference
50%	10	47.3	22.301	497.344	18		
75% vs.	10	36.4	20.173	406.933	18	0.005	Significant difference
25%	10	65.90	21.429	459.211	18		
75% vs.	10	36.4	20.173	406.933	18	<0.001	Significant difference
0%	10	93.40	14.630	214.044	18		
50% vs.	10	47.3	22.301	497.344	18	0.073	No Significant difference
25%	10	65.90	21.429	459.211	18		
50% vs.	10	47.3	22.301	497.344	18	<0.001	Significant difference
0%	10	93.40	14.630	214.044	18		
25% vs.	10	65.90	21.429	459.211	18	0.004	Significant difference
0%	10	93.40	14.630	214.044	18		

Table 6.9 – Experiment 2 newly preferred generated moves. Here, *n* is the number of participants; *sd* is the standard deviation and *df* is the degrees of freedom

Comparing Experiment 1 and Experiment 2, the results show a clear difference. We see that the implementation of the proposed algorithm does confirm that a dance will improve in terms of satisfaction and newly perceived combinations, when participants observe different dances (Experiment 2) compared to selecting their preferences from the same initial dance (Experiment 1). On the other hand, the results from Experiment 1 show that a dance improves as more of the participant's own feedback is incorporated into the subsequent dances.

6.3.2. Experiment 3: Results & Analysis

In Experiment 3, participants were required to provide the number of newly generated moves they preferred and did not prefer for each dance as a measure to determine the robot's performance. The responses for each dance were averaged and the results can be seen below in Figure 6.12.

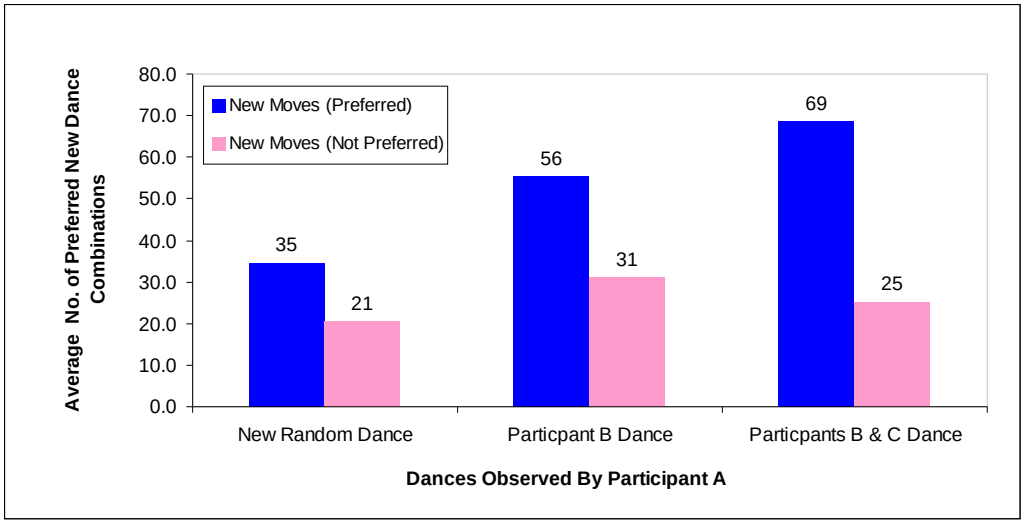


Figure 6.12 – Average score number of newly preferred combinations in Experiment 3

From Figure 6.12, we see that as the number of trainers increased from zero (i.e. a new random dance) to two (i.e. Participants B & C Dance), the number of preferred moves also increased. This suggested that having feedback from two people produced a better dance (that is, a dance with more preferred moves observed by an independent observer) than having no feedback or having feedback from only one person (Participant B dance). Movements that were not preferred on the other hand were almost the same. Although this was initial result showing the increased speed of learning with increasing trainer preferences, the similarity in values of movements that were not preferred, suggested that having observers identify movements that they did not prefer may not have

been as effective a technique as having them identify moves they did prefer. Nevertheless, the result did suggest that given an increase in trainers (from zero to two), the number of preferred newly generated dance combinations would increase.

For Experiment 3, the increase in quality of the robot's dance was as a result of increasing the number of trainers, beginning with no trainers to two trainers. This experiment provides evidence to suggest that a robot's dance performance could be improved when it is adapted to the preferences of either a single participant or at most two participants. To measure this statistically, three research questions were asked for this experiment. The questions and their respective hypotheses were as follows:

1. *Does learning increase with one trainer?*
   Hypothesis: There is a significant difference in learning between no training and 1 trainer.

2. *Does learning increase with two trainers?*
   Hypothesis: There is a significant difference in learning between no training and 2 trainers.

3. *Does learning increase from one trainer to two trainers?*
   Hypothesis: There is a significant difference in learning between 1 trainer and 2 trainers.

Again, the 2-tailed t-test approach was used to compare the dances at a 95% confidence level. Table 6.10 below summarises the statistical results obtained for the above research questions.

Research Question	Dance Type	Preference Category	Mean	Standard Deviation	Variance	p-value	Conclusion
1. ***Does learning increase with 1 trainer?***	Random Dance vs. One Trainer Dance	Preferred moves	34.65	13.901	193.225	0.029	Significant difference
			55.5	23.97	574.5		
		Non-Preferred moves	20.5	18.68	348.94	0.158	NO Significant difference
			31.2	13.36	178.4		
2. ***Does learning increase with 2 trainers?***	Random Dance vs. Two Trainer Dance	Preferred moves	34.65	13.901	193.225	0.005	Significant difference
			68.7	30.247	914.9		
		Non-Preferred moves	20.5	18.68	348.94	0.473	NO Significant difference
			25.2	7.9	62.4		
3. ***Does learning increase from 1 trainer to 2 trainers?***	One Trainer Dance vs. Two Trainer Dance	Preferred moves	55.5	23.97	574.5	0.294	NO Significant difference
			68.7	30.247	914.9		
		Non-Preferred moves	31.2	13.36	178.4	0.23	NO Significant difference
			25.2	7.9	62.4		

Table 6.10 – Research questions and statistical result for Experiment 3. Here, the top row of statistical data for each *preference category* is for the top *dance type* group being compared for each *research question*.

For the first and second research questions, the result for non-preferred moves did not show a significant difference between the untrained (random dance) and trained dances supporting the null hypothesis that there was no significant difference in learning. However, preferred moves did show a significant difference, supporting the hypothesis that a significant difference did occur in learning.

Overall, the statistical questions between the first and second research question suggests that some learning did occur with one trainer and two trainers, but not enough to be considered significant. For the third research question on the other hand, the result for both non-preferred moves and preferred moves did not show a significant difference

between the one trainer dance and the two trainer dance supporting the null hypothesis that there was no significant difference in learning.

What is interesting here is that there was no significant improvement in learning between one and two trainers when one might expect that there would be. There are two reasons to this. The first reason being that, many of the actions performed in the one trainer dance could have been repeated, in the two trainer dance, depending on how favourable they were to the participants, thus, reducing satisfaction levels. Second, the same participants were used in this experiment as for Experiments 1 and 2, and therefore there was a good chance that they had seen some of the combinations in other dances (in Experiments 1 and 2), which would have affected the results in this way. More repeats of this experiment with more (and/ or different) participants are necessary to know for sure.

It can be simply concluded that increasing the number of training improves the results. The optimum number of trainers is to be determined. However, in general, the results suggest that training with at most two trainers produced better learning than no training at all. This also suggests that learning occurred faster if the feedbacks from at most two trainers were combined, although it would be more interesting to know as further research, if by increasing the number of trainers, learning would continue to increase or not.

6.4. Summary

The results in this stage of the robot's development are encouraging and justify the need for human feedback to improve the robot's dancing. From Experiment 1, we saw that the quality of the dance reduced as less of the participant's preferences were incorporated,

and in all subsequent dances observed after the initial random dance, there was improvement. In Experiment 2, the quality of the dance improved as a result of the incorporation of another participants' preferences. In Experiment 3 on the other hand, when an independent observer observed the dances of 1 and 2 trainers, compared to a newly generated random dance, there was improvement in the dances in terms of new dance combinations suggesting that having two trainers produces a more interesting dance than 1 trainer or no trainers at all.

The results suggest that training with single or multiple (at most two) trainers (excluding the preferences of an independent observer) produces improvement in the robot's dancing than no training at all. Furthermore, combining the preferences of participants who have observed different dances also improves the robot's dance. However, it would be interesting to explore dances that have been adapted to the preferences of professional dancers and choreographers. Nevertheless, the learning achieved in this chapter and from the previous chapters in *learning to dance to the beat* and *generating dance*, help to form an introductory framework to make a robot learn to dance.

Chapter 7
Concluding Remarks & Future Work

This thesis presents a framework that addresses three necessary stages of development for generating robot dances. The stages are: learning desirable dance behaviours; adapting dance motions to human preferences, and the ability to demonstrate autonomous and creative behaviour in exploring dance movements. This chapter highlights how these stages were addressed and the results obtained. Suggestions for future research directions in this research are also given.

7.1. Thesis Achievements

The idea of robots dancing is a recent area of study and has been explored by researchers in several ways. The typical approach to robot dance is to pre-program a robot with a collection of dance motions that are either choreographed to specific music signals, or can be randomly selected, synchronised to musical rhythms and dynamics. Other approaches include the direct mimicking of human dance motions, and applications that allow trainers to generate different dance patterns for a robot. All these approaches have proved successful. However, they limit the robot's ability to demonstrate advanced autonomous and creative dance motions, which are necessary to maintain human interest, reducing human input and social interaction. For example, most robots do not have any learning ability and so can only demonstrate pre-defined behaviour in their dance, whilst others can adapt their dance motions to the music, but cannot adapt to human feedback. To address these limitations, the aim of this research was to develop a framework (robot dance system) that would advance a robot's dance to firstly learn the desirable dance

behaviours to perform, and use these behaviours to generate its own dance steps. The robot would then receive and evaluate human feedback and adapt its dance to the observer's preferences. The main contribution of the research is an integrated system that incorporates the robot's previous experience and learnt behaviours, whilst still being able to adapt to human input, synchronised to music, based on traditional reinforcement learning.

The traditional Sarsa algorithm and Softmax algorithm from reinforcement learning, was used as a solution to make a robot learn using immediate and delayed rewards, to perform dance motions to musical beats. The robot was then given the ability to generate its own primitive dance motions. Since the robot did not know which dance motions were desirable, the robot had to undergo further learning. The robot would learn the desired dance motions to perform, based on fundamental attributes of dance described in Chapter 2, and structure the dance motions into chains of sequences, that when combined, actually produce a dance. With this approach, the robot did not have to rely on the trainer to provide it with initial dance steps to perform. This enabled the robot to explore and generate its own dance motions and sequences, which it could perform in seemly autonomous ways. Computational and empirical results were obtained to evaluate the robot's learning and dance improvement respectively.

A number of experiments were conducted to determine the robot's dance improvement. Videos of the robots dances were shown to participants. Participants were requested to rate the dances using a questionnaire. The resulting work showed that increasing the number of joints in the robots dance and incorporating attributes of dance such as *Opposite*, *Symmetry* and *Formation* movements do perceptually improve the

robot's dance. The robot's ability was then further developed to be able to retrieve the preferences of human feedback in real time and change its dance to suit their preferences.

This research has demonstrated that a basic repertoire of dance motions can be combined in a structured way, similar to how humans may organise dance behaviours using reinforcement learning and that it's possible to adapt a robot dance to the preferences of human observers. With this idea, the research has found that when a robot dance incorporates human-like movements that appear to be structured, the dance is considered to significantly improve, but even more so when human preferences are included in the dance.

During the course of this research, the following publications were made:

- Tholley, I.S., Meng, Q. & Chung, P.W.H., (2009). Towards a learning framework for dancing robots. *IEEE International Conference on Control and Automation* (ICCA), Christchurch, New Zealand, 9-11 December 2009, pp.1581-1586.
- Tholley, I.S., Meng, Q. & Chung, P.W.H., (2012). Robot Dancing: What Makes a Dance? Advanced Materials Research, Vol. 403-408, pp. 4901-4909. Available at: *www.scientific.net*

7.2. Future Work

Due to the infancy of this research area, there are a number of potential future directions for dancing robots. Significantly, all the work presented in this thesis can be developed to more advanced ideas. To begin with, the proposed dancing framework only took into consideration the rhythm (beats) of a musical signal, and not the dynamics within the music or the structure of the music. A logical advancement would therefore be to

consider other music features such as tempo and amplitude. This would not only give the robot the ability to follow the music, but also the idea of the robot generating choreographed dance patterns based on the musical structure, or demonstrating emotion in its dance based on the dynamics of the music.

With reference to the experiments carried out in this research, more experiments could be conducted to answer some of the questions derived from this research. For example, it was discovered that it's possible to model fundamental features of dance in the robot that was used in this project, but is this the same for other robots, both humanoid and non-humanoid in appearance? In Chapter 6, the robot's dancing improved as it was trained by one and two trainers, but would this be the same for more than two trainers? These questions are not answered in this thesis and further experiments would be required to answer them.

In this research, traditional reinforcement learning was adopted for learning and knowledge acquisition. Although it has proven successful in this research, it is also worth considering other methods that could perhaps be combined with this learning approach such as function approximators, and learning algorithms such as Neutral Networks and Genetic Algorithms for comparison.

Moving away from the implementations of this research into completely new ideas, an ambitious future consideration is the idea of robots dancing with other dancers (i.e. robots or human partners) to increase social interaction. This thesis focused on humans being the spectator, but social interaction is also concerned with how objects relate to each other. Imitation is currently the most successful implementation of robot dancing.

A limitation of the imitation approach is that currently the dancing robots do not use the dance motions they have perceived to form their own movements. Adding a visual feedback system combined with artificial learning algorithms to make robots learn from other dancers, be it robots or human, could provide a complete novel way for creating robots that dance. Dancing robots could even go one step further and take part in dance competitions and talent shows mimicking TV dance programs such as "Strictly Come Dancing", or artists such as Fred Astaire and Ginger Rodgers to create music with their dancing (like tap dancing), whereby rhythmic sounds and patterns are made with dance movements.

References

adionSoft (2007) *BPM Detection Library* [Online]. Available from: http://adionsoft.net/bpm/

Aucouturier, J.-J. (2008a). Cheek to Chip: Dancing Robots and AI's Future. *IEEE Intelligent Systems*, *23*(2), pp. 74-84.

Aucouturier, J.-J., Ogai, Y., & Ikegami, T. (2008b). Making a Robot Dance to Music Using Chaotic Itinerancy in a Network of FitzHugh-Nagumo Neurons. *Neural Information Processing*, pp. 647-656.

Austermann, A., & Yamada, S. (2008). "Good Robot", "Bad Robot" – Analyzing Users' Feedback in a Human-Robot Teaching Task. *Symposium A Quarterly Journal In Modern Foreign Literatures*, pp. 41-46.

Craig, J. (2005). [Online] *Rhythm.* http://jacmuse.com/041708trc/101706trc/pages/newpage8.htm. [Accessed: September 20, 2010].

Cyberbotics (2011) *Webots 6 for fast prototyping and simulation of mobile robots* [Online] Available from: http://www.cyberbotics.com/

Dandurand, F., Shultz, T. R., & Rivest, F. (2007). *Complex problem solving with reinforcement learning*. In the Proceeding of the 6th IEEE International Conference on Development and Learning (ICDL-2007), pp. 157-162.

Dayan, P. (2005). "*Reinforcement Learning*" in Encyclopaedia of Cognitive Science, pp. 1-14.

Dozier, G. (2001). Evolving robot behavior via interactive evolutionary computation: From real-world to simulation. *Proceedings of the 2001 ACM symposium on Applied computing*, pp. 340-344.

Ellenberg, R., Grunberg, D., Oh, P., & Kim, Y. (2008). Exploring Creativity through Humanoids and Dance. *Proceedings of the 5th International Conference on Ubiquitous Robotics and Ambient Intelligence November 2008*.

Erdem, E., Erdem, A. T., Abaci, T., Ozkan, M. K., & Erdem, C. E. (2008). Unsupervised Dance Figure Analysis From Video For Dancing Avatar Animation, *College of Engineering , Koc. Analysis*, pp. 1484-1487.

Gentry, S., & Murray-Smith, R. (2003). Haptic dancing: human performance at haptic decoding with a vocabulary. *IEEE International Conference on Systems Man and Cybernetics*, pp. 3432-3437.

Gizmodo (2008) *Spooky Hexapod Dancing Robot Scares Ladies Your Way* [Online] Available from: http://www.gizmodo.com [Accessed: 15 July 2010]

Gosavi, A. (2003), *Simulation-Based Optimization: Parametric Optimization Techniques & Reinforcement Learning*, New York: Springer.

Graf, J. and Banzhaf, W. (1995). Interactive evolutionary algorithms in design, Procedures of Artificial Neural Nets and Genetic Algorithms, pp. 227–230.

Gray W. D., (2007) "*Integrated Models of Cognitive Systems*", Oxford University Press, USA

Grunberg, D., Ellenberg, R., Kim, Y., & Oh, P. (2009). Creating an autonomous dancing robot. *Proceedings of the 2009 International Conference on Hybrid Information Technology ICHIT 09*, pp. 221-227. ACM Press.

Hall, R. (2005) *Math for Poets and Drummers: The Mathematics of Rhythm* [Online]. Available from: http://www.sju.edu/~rhall/Rhythms/Poets/arcadia.pdf [Accessed: 22 October 2010]

Hirata, Y., Hayashi, T., Takeda, T., Kosuge, K., & Wang, Z.-D. (2005). Step Estimation Method for Dance Partner Robot " MS DanceR " using Neural Network. *2005 IEEE International Conference on Robotics and Biomimetics ROBIO*, pp. 523-528.

Humphrey, D. (1959) *The Art of Making Dances*, ISBN: 0-903102-39-0

Jens, H., Peer, A. & Buss, M. (2010). Synthesis of an Interactive Haptic Dancing Partner. *Control*, pp. 527-532.

Kim, G., Wang, Y., & Seo, H. (2007). Motion Control of a Dancing Character with Music. *6th IEEE ACIS International Conference on Computer and Information Science ICIS 2007*, pp. 930-936.

Kramer, S. (2010). Rhythm Movement for Toddlers and Preschoolers. Practical Spirituality and Rhytmic Dance for Children, Teens, and Adults. [Online] Available at: http://www.susankramer.com/Toddlers.html [Accessed: 20 September, 2010].

Landgraf, T., Oertel, M., & Rhiel, D. (2010). A biomimetic honeybee robot for the analysis of the honeybee dance communication system. *German Research*, pp. 3097-3102.

Leopold, T.; Kern-Isberner, G. & Peters, G. (2008), Belief revision with reinforcement learning for interactive object recognition, *in* Malik Ghallab; Constantine D. Spyropoulos; Nikos Fakotakis & Nikolaos M. Avouris, ed., 'ECAI' , IOS Press, pp. 65-69.

Liu, F. & Su, J. (2004). Visual learning framework based on reinforcement learning *Proceedings of the World Congress on Intelligent Control and Automation (WCICA),* pp. 4865–4868.

Martinson E., Stoytchev A., Arkin R. (2002), *Robot behavioral selection using Q-learning,* in Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Lausanne, Switzerland

McGreevy-Nichols, S., Scheff, H. and Sprague, M. (2005). Building Dances: A guide to putting dances together. *Human Kinetics, Volume 1, Second Edition.* Peoria: Versa Press.

Michalowski, M.P., Sabanovic, S.**,** & Kozima, H. (2007). A Dancing Robot for Rhythmic Social Interaction. *In Proceedings of the 2nd ACM/IEEE Conference on Human-Robot Interaction (HRI '08)*, Washington, DC.

Morris, G., Nevet, A., Arkadir, D., Vaadia, E., & Bergman, H. (2006). Midbrain dopamine neurons encode decisions for future action. *Nature Neuroscience*, pp. 1057-1063.

Nakamura, J., Nakamura, U., Nakamura, K., Shochiku Kabushiki Kaisha., NHK Sofutowea., Nihon Haiyu Kyokai., & Nihon Hoso Kyokai. (2004). *Shujaku lion dance*.

Nakaoka, S., Kajita, S., & Yokoi, K. (2010). Intuitive and Flexible User Interface for Creating Whole Body Motions of Biped Humanoid Robots. *Science And Technology*, pp. 1675-1682.

Nakaoka, S., Nakazawa, A., Kanehiro, F., Kaneko, K., Morisawa, M., & Ikeuchi, K. (2005). Task Model of Lower Body Motion for a Biped Humanoid Robot to Imitate Human Dances. *2005 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 2769-2774.

Nakaoka, S., Nakazawa, A., Yokoi, K., & Ikeuchi, K. (2004). Leg motion primitives for a dancing humanoid robot. *IEEE International Conference on Robotics and Automation 2004 Proceedings ICRA 04 2004*, pp. 610-615.

NewScientist (2007) *A programmable robot from 60AD* [Online] Available from: http://www.newscientist.com [Accessed: 15 July 2010]

Nishino, H., Takagi, H., & Utsumiya, K. (2001). A 3D modeling system for creative design. *Proceedings 15th International Conference on Information Networking*, pp. 479-486.

Oliveira, J. L., Gouyon, F., & Reis, L. P. (2008). Towards an Interactive Framework for Robot Dancing Applications. *International Conference on Digital Arts*, pp. 52-59.

Retrieved from
http://www.inescporto.pt/~fgouyon/docs/OliveiraGouyonReis_IRobot2008.pdf

Operating System. (2009). *Aperios Operating System.* [Online]. September 2008. Available from: http://www.operating-system.org/betriebssystem/_english/bs-aperios.htm. [Accessed: 10th June 2011]

Poli, R., & Cagoni, S. (1997). Genetic Programming with User-Driven Selection: Experiments on the Evolution of Algorithms for Image Enhancement. *Genetic Programming 1997 Proceedings of the Second Annual Conference*, pp. 269-277. Morgan Kaufmann.

Poliscuk, J. (2002), Adaptive Machine Reinforcement Learning, in *The 15th IEEE International Symposium on Robot and Human Interactive Communication*, Hatfield, pp. 57-74.

Riley, M., Ude, A., & Atkeson, C. G. (2000). Methods for Motion Generation and Interaction with a Humanoid Robot: Case Studies of Dancing and Catching. *Brain*, pp. 35-42.

Santiago, C.B.; Oliveira, J.L.; Reis, L.P.; Sousa, A. (2011). Autonomous robot dancing synchronized to musical rhythmic stimuli. *Information Systems and Technologies (CISTI), 2011 6th Iberian Conference*, pp. 34-47.

Schaffer, K., and Stern, E. (2009). *Sorts of Symmetries. Math Dance*. [Online] Available at: http://www.mathdance.org/symmetrygroups/symmetrygroups.html [Accessed: 19 September, 2010].

Shinozaki, K., Iwatani, A., & Nakatsu, R. (2007). Concept and construction of a dance robot system. *Proceedings of the 2nd international conference on Digital interactive media in entertainment and arts DIMEA 07*, *6*(3), 161. ACM Press.

Shiratori, T. and Ikeuchi, K. (2008). Synthesis of Dance Performance Based on Analyses of Human Motion and Music. *IPSJ Transactions on Computer Vision and Image Media*, Vol. 1, No. 1, pp. 34-47.

Shiratori, T., Kudoh, S., Nakaoka, S., and Ikeuchi, K. (2007). Temporal scaling of upper body motion for Sound feedback system of a dancing humanoid robot. *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp.3251-3257.

Shiratori, T., Nakazawa, A., and Ikeuchi, K. (2006). Dancing-to-Music Character Animation. *Computer Graphics Forum*, Vol. 25, No. 3 (Proc. Eurographics 2006), pp. 449-458.

Shiratori, T., Nakazawa, A., and Ikeuchi, K. (2004). Detecting Dance Motion Structure through Music Analysis. *Sixth IEEE International Conference on Automatic Face and Gesture Recognition (FGR 2004)*, May 2004, pp. 857-862.

Sikora R. T. (2005), *Learning Optimal Parameter Values in Dynamic Environment: An Experiment with Softmax Reinforcement Learning Algorithms*

Smith-Autard, J.M., (1988) *Dance Composition: A Practical Guide for teachers,* A & C Black, London, ISBN: 0-7136-3583-5

Solis, J. Chida, K. Suefuji, K. Takanishi, A. (2005). Improvements of the sound perception processing of the Anthropomorphic Flutist Robot (WF-4R) to effectively interact with humans. *Power*, pp. 450-455.

Srinivasan, B. (2005). Analysis of Memory-Based Learning Schemes for Robot Navigation in Discrete Grid-Worlds with Partial Observability.

Suga, Y., Ikuma, Y., Nagao, D., Ogata, T., Sugano, S. (2005). Interactive Evolution of Human-Robot Communication in Real World; Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Vol. 2, pp. 1592-1597.

Sutton R.S. and Barto A.G (1998), *Reinforcement Learning: An Introduction*, MIT press, Bradford Books, Cambridge, MA.

Takagi, H. (2001). Interactive evolutionary computation: fusion of the capabilities of EC optimization and human evaluation. *Proceedings of the IEEE*, *89*(9), Vol. 89, pp. 1275-1296.

Takeda, T., Hirata, Y., & Kosuge, K. (2007). Dance step estimation method based on HMM for dance partner robot. *IEEE Transactions on Industrial Electronics*, *54*(2), Vol. 54, pp. 699–706.

Takeda, T., Hiarata, Y., Wang, Z., & Kosuge, K. (2006). HMM-based Error Detection of Dance Step Selection for Dance Partner Robot -MS DanceR. *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems,* Vol. 54, pp. 5631-5636.

Tanaka, F., Fortenberry, B., Aisaka, K., & Movellan, J. R. (2005). Plans for Developing Real-time Dance Interaction between QRIO and Toddlers in a Classroom Environment. *4th IEEE International Conference on Development and Learning*, pp. 142-147.

Tanaka, F., Movellan, J. R. Fortenberry, B. and K. Aisaka (2006). Daily HRI evaluation at a classroom environment: Reports from dance interaction experiments. In *Proceedings of the 2006 Conference on Human-Robot Interaction (HRI)*.

Tanaka, F., & Suzuki, H. (2004). Dance interaction with QRIO: a case study for non-boring interaction by using an entrainment ensemble model. *ROMAN 2004 13th IEEE International Workshop on Robot and Human Interactive Communication IEEE Catalogue No 04TH8759,* pp. 419-424.

The Chicago School of Media Theory (2004) *Gesture* [Online]. Available from: http://csmt.uchicago.edu/glossary2004/gesture.htm#_edn2 [Accessed: 05 October 2011]

Thomaz, A. L., & Breazeal, C. (2007). Asymmetric Interpretations of Positive and Negative Human Feedback for a Social Learning Agent. *ROMAN 2007 The 16th IEEE International Symposium on Robot and Human Interactive Communication*, pp. 720-725.

Thomaz, A. L., Hoffman, G., & Breazeal, C. (2005). Real-Time Interactive Reinforcement Learning for Robots. *AAAI 2005 Workshop on Human Comprehensible Machine Learning*.

TrendHunter (2001) *Dancing Robots Pets* [Online]. Available from: http://www.gizmodo.com [Accessed: 15 July 2010]

Yuuki, O., Yamazaki, S., Yamada, K., & Kubota, N. (2009). Dexterous motions of Japanese dance by a miniature humanoid robot. *Bibliothek*, pp. 4427-4432.

Van Camp, J. C (1981). Philosophical Problems Of Dance Criticism. *Ph.D. Dissertation*. Retrieved from http://www.worldcat.org/title/philosophical-problems-of-dance-criticism/oclc/430369696

Virčíková, M., & Sinčák, P. (2010). Dance Choreography Design of Humanoid Robots using Interactive Evolutionary Computation. *Human Friendly Robotics for Young Researchers*.

Vukobratovic, M., & Borovac, B. (2008). Dynamic balance concept and the maintenance of the dynamic balance in humanoid robotics. *2008 6th International Symposium on Intelligent Systems and Informatics*, pp. 1-11.