Learning Through Imitation for Mobile Robots

Ioannis N. Gatsoulis

Master of Science
School of Artificial Intelligence
Division of Informatics
University of Edinburgh
2001
Abstract

Imitative learning facilitates skill acquisition in a social environment where one agent learns how to act by perceiving and, in some sense, imitating the actions of another. Most of the research work is concentrated in action–level imitation learning, where the low level motor actions are learned. We are interested in program–level imitation learning, where the learner learns through imitation what behaviour should execute under a perceptual state. Furthermore, most of the research work in imitation learning involves behaviours that do not involve object interaction. There are a few attempts with behaviours that involve object interaction, but these are studied in computer simulated environments. We are also interested in behaviours that involve object interaction, but we wish to study them in real robots. A foraging scenario was chosen as the experimental platform. By robot foraging, we mean that the robot searches for food objects and brings them to its home. There are two ways that the robot could bring the food; it can either pick it up and carry it, or it can push it. In addition, an avoidance behaviour is executed when the food is poisoned. There is a teacher robot, who is an expert in foraging, and intends to teach a learner robot. The learner robot follows the teacher robot and imitates its demonstrated behaviours. The learner then associates the behaviours, it just imitated, with its perception. With this way, the learner will be able to perform the behaviours in a similar situation, without needing the teacher to show it what to do. The experimental results we obtained, show that the learner improves as more training is acquired. However, these results also show that the rate of the learner’s improvement drops beyond a number of training episodes. By observations of the experiments, we also came to the conclusion that some design issues in the recall phase of the learner as well as the variety of the training episodes affect the performance of the learner.
Acknowledgements

First of all, I would like to thank my supervisor Gillian Hayes for her support, ideas and encouragement throughout the project. I must also thank her for reading my manuscripts in such a short notice.

I would also like to thank my second supervisors George Maistro and Yuval Marom for their continuous support, ideas and encouragement. In particular I thank George for reading my manuscripts in such a short notice as well, and Yuval for helping me with any problems I had with the robots and the simulator. I also have to say that it was not my fault but probably God’s will that our meetings had always to take place in crowded cafeterias.

I would like to thank John Hallam for temporarily supervising me when Gillian was away.

I would also like to thank the guys from the workshop, Hugh Cameron, Sandy Colquhoun, Douglas Howie and Robert MacGregor for building the experimental setup and taking care of the robots.

I would like to thank, Tim Colles, one of our system administrators, for configuring the serial ports.

I would also like to thank my best friends here in Edinburgh, Niko and Alexandro, for having such a good time with them during this MSc year.

I would like to thank EPSRC for awarding me a studentship (award number: 00401579) for studying here in Edinburgh University.

I would like to thank my parents, Niko and Kiki, for their moral and financial support throughout the year(s).
Declaration

I declare that this thesis was composed by myself, that the work contained herein is my own except where explicitly stated otherwise in the text, and that this work has not been submitted for any other degree or professional qualification except as specified.

(Ioannis N. Gatsoulis)
To my grandparents, Eleni, Ntina and Yianni.
# Table of Contents

1 **Introduction**  
1.1 Introduction ........................................... 1  
1.2 Motivations ............................................. 2  
1.3 Abstract Description of the Research Scenario ............ 5  
1.4 Objectives ............................................. 6  
1.5 Overview of the Dissertation .................................. 7  

2 **Background**  
2.1 Types, Levels and Terminology of Imitation Learning ........... 8  
2.1.1 Matched Dependent Behaviour: A Type of Imitation Learning 9  
2.1.2 Levels of Imitation Learning ............................. 9  
2.1.3 Terminology ........................................ 10  
2.2 Imitation Learning in Nature and in Ethology ..................... 12  
2.3 Imitation Learning from the Viewpoint of Robotics ............ 13  
2.3.1 Imitation Learning in Mobile Robotics ................... 13  
2.3.2 Imitation Learning in Assembly Robotics ................ 17  
2.4 Where Our Work Fits In .................................... 18  
2.5 Summary ............................................. 19  

3 **System Set-Up** .............................................. 20  
3.1 Environment ............................................ 20  
3.2 The Khepera Robot ..................................... 21  
3.3 The Khepera Simulator .................................. 23  
3.4 The Camera and the Frame Grabber .......................... 24
5.5 Discussion ................................................. 61
5.6 Summary ................................................ 62

6 Conclusions ............................................. 63
6.1 Achievements ......................................... 63
6.2 Conclusions and Further Work .................... 64
6.3 Epilogue ............................................... 65

A Experimental Results .............................. 66

Bibliography ............................................. 74
## List of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>The active imitation’s basic block of behaviour–forward model pairing.</td>
<td>14</td>
</tr>
<tr>
<td></td>
<td>Figure adopted from (Demiris and Hayes, 2001, Figure 2)</td>
<td></td>
</tr>
<tr>
<td>2.2</td>
<td>A schema network: several perceptual–motor pairs send candidate motor</td>
<td>16</td>
</tr>
<tr>
<td></td>
<td>commands to the competition module which gives the output of the network.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Figure adopted from (Maistros and Hayes, 2001, Figure 2)</td>
<td></td>
</tr>
<tr>
<td>2.3</td>
<td>The integrated system: the SOFM handles the stimulus and finds the best</td>
<td>18</td>
</tr>
<tr>
<td></td>
<td>matching node at each time; its hand–wired motor schema calculates the</td>
<td></td>
</tr>
<tr>
<td></td>
<td>motor commands before sending them to the motor system for execution.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Figure adopted from (Maistros et al., 2001, Figure 4).</td>
<td></td>
</tr>
<tr>
<td>3.1</td>
<td>The large arena with the overhead camera.</td>
<td>21</td>
</tr>
<tr>
<td>3.2</td>
<td>The two identical training corridors.</td>
<td>21</td>
</tr>
<tr>
<td>3.3</td>
<td>The objects used to represent the food. They are 35 mm in diameter and</td>
<td>22</td>
</tr>
<tr>
<td></td>
<td>23 mm in height. They are covered by white paper, such that it is also</td>
<td></td>
</tr>
<tr>
<td></td>
<td>easier for the camera tracking program to detect.</td>
<td></td>
</tr>
<tr>
<td>3.4</td>
<td>A picture of a Khepera robot and a drawing showing the positions of the</td>
<td>22</td>
</tr>
<tr>
<td></td>
<td>motors and of the infrared–ambient light sensors.</td>
<td></td>
</tr>
<tr>
<td>3.5</td>
<td>The gripper module and the Khepera robot equipped with it.</td>
<td>23</td>
</tr>
<tr>
<td>3.6</td>
<td>Screenshot of the Khepera simulator.</td>
<td>24</td>
</tr>
<tr>
<td>3.7</td>
<td>Camera view. It can be seen that cables are getting in the way of the</td>
<td>25</td>
</tr>
<tr>
<td></td>
<td>camera and the robot and the objects making it difficult to track them.</td>
<td></td>
</tr>
<tr>
<td>3.8</td>
<td>A 1–hole region. The inner black square is considered to be the hole of</td>
<td>26</td>
</tr>
<tr>
<td></td>
<td>the region. The black frame around the white region is just to make the</td>
<td></td>
</tr>
<tr>
<td></td>
<td>contrast of its intensity more distinct with respect to the background.</td>
<td></td>
</tr>
</tbody>
</table>
3.9 The markers on top of the Khepera robots that the camera software is able to detect. The regions are $120 \times 80 \text{ mm}$. The size of each of the inner black squares is $20 \times 20 \text{ mm}$. 

4.1 Transformation of the raw data to meaningful state vector that determines the state. Note that the classification of a state vector into a state is only used by the teacher. 

4.2 Decision tree of the classification of the state vectors to states. Right branches are true values, left branches are false. 

4.3 A *MultiStep* consists of (number of robots) *Steps*. A MultiStep is like a round in a board game, while a Step is like a player’s turn. 

4.4 Hierarchical imitation: program–level imitation and action–level imitation 

5.1 The training setup. 

5.2 Average scores of the recall phases in the *training–corridor arena*, with respect to the number of training episodes. 

5.3 Number of collected food objects (max 20) *in the training–corridor arena*, with respect to the number of training episodes. 

5.4 When the food lies in Area 2, the robot collects it most of the times. However, this is rarely the case when the food lies in Area 1. 

5.5 Average scores of the recall phases in the *rectangular arena*, with respect to the number of training episodes. 

5.6 Number of collected food objects (max 24) *in the rectangular arena*, with respect to the number of training episodes.
List of Tables

4.1 The states and their corresponding behaviours. . . . . . . . . . . . . . 32
4.2 The attributes of the state vector that determine each state. . . . . . . 46

A.1 Experimental results for the recall phases in the training–corridor arena having the learner go through 1 training episode. Number of food=1, time limit=500 steps, penalty=-20% . . . . . . . . . . . . . . . . . . . . . . . . . . 67
A.2 Experimental results for the recall phases in the training–corridor arena having the learner go through 3 training episode. Number of food=1, time limit=500 steps, penalty=-20% . . . . . . . . . . . . . . . . . . . . . . . . . . 68
A.3 Experimental results for the recall phases in the training–corridor arena having the learner go through 5 training episode. Number of food=1, time limit=500 steps, penalty=-20% . . . . . . . . . . . . . . . . . . . . . . . . . . 69
A.4 Experimental results for the recall phases in the training–corridor arena having the learner go through 10 training episode. Number of food=1, time limit=500 steps, penalty=-20% . . . . . . . . . . . . . . . . . . . . . . . . . . 70
A.5 Experimental results for the recall phases in the rectangular arena having the learner go through 1 training episode. Number of food=2, time limit=700 steps, penalty=-20% for leaving all, food in the arena, -5% for leaving one . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 71
A.6 Experimental results for the recall phases in the rectangular arena having the learner go through 5 training episode. Number of food=2, time limit=700 steps, penalty=-20% for leaving all, food in the arena, -5% for leaving one . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 72
A.7 Experimental results for the recall phases in the rectangular arena having the learner go through 10 training episode. Number of food=2, time limit=700 steps, penalty=-20% for leaving all, food in the arena, -5% for leaving one
Chapter 1

Introduction

In this chapter our inspirations and motivations to the study of imitation learning in robots are firstly described. An abstract description of the project and its objectives are presented next, followed by an overview of the rest of the dissertation.

1.1 Introduction

Most of the intelligent living beings live in groups. Not only because they feel safer and satisfy their needs better, but also because they can gain knowledge or acquire new skills that they do not possess from other members of the group. This type of learning where agents learn from other agents is called social learning.

One form of social learning is imitation learning. It is a powerful means of skill acquisition among humans and a few other animal species. It is so common that quite often we do not even realise that it takes place, as it happens subconsciously (e.g. ordering an espresso just after our company has ordered one). Learning by imitation is not very easy to define and there is still a big chaos in the terminology used to describe imitative phenomena. As Galef points out, “one man’s example of true learning by imitation is another’s paradigmatic case of pseudo-imitation and each can cite historical precedent for treating phenomena as he does” (Galef, 1988, page 4). However, imitation learning can be considered as the learning process where one agent learns how to act by observing another agent demonstrating a skill, and then in some sense
imitate its actions, so as to be able to perform the skill demonstrated in the absence of the expert agent.

The next section describes our motivations and inspirations towards studying and using imitation learning in the programming of autonomous robots.

1.2 Motivations

In robotics, it is quite unlikely for two robots to be ‘identical’. Even if they are supposed to be, there are minimal differences in their morphologies and variance in the configuration of their sensors and their corresponding readings that can lead to great differences in their behaviours. This gives rise the problem of program incompatibility; since a mere copying of a program to another robot usually fails. Even worse, if the morphologies (gear wearing, axle alignments, etc.) of the robots are very different from each other, then it is pretty obvious that the same program will not work for both of them.

The main reason for this problem is in the implementation of the program. It is usually ‘custom-implemented’ for a particular robot, bearing in mind the robot’s morphology, sensor and motor configurations. On the other hand, trying to build a more general program may result in poor performance of the task.

A solution to this kind of problem is to acquire the new skill through learning by imitation; one robot learns how to perform a task from its own perceptions and capabilities by observing and possibly ‘talking’ with a demonstrator robot, which could be the robot with the custom-implemented program. Learning by imitation has drawn serious attention from roboticists in the last few years. The most important reason for this interest has already been described in the sense of the major drawback of program compatibility. It could allow robots of different morphologies to perform the same task without the need of reprogramming them. For example, the robots could use different devices for navigating and obstacle avoidance, like laser-range sensors or cameras; or they could have different sensors for detecting object presence on their grippers, such as optical-barrier sensors or infra-red ones. To a greater extent, it could make it possible not to have to program any robot at all in the first place, rather a human could
play the role of the teacher, demonstrating the task to a robot; opening a new era in the programming of agents.

Furthermore, imitation learning is also interesting and worth researching from the industrial point of view, as it is a step towards fully-automated systems\(^1\). “A robot which starts when the sun rises could teach another to start work when it hears the birds singing” (Hayes and Demiris, 1994a, page 6); saving money since a robot can work and teach without the need for resting, or at least a small amount of time to charge its batteries.

Hayes and Demiris (1994a) have investigated imitation learning in a task where a robot must learn to navigate through a maze. However, we are interested in behaviours involving interactions with objects, such as foraging. The motivation is that behaviours that involve interaction with objects are much more complicated in general. For example picking up an object requires correct orientation of the robot with respect to the object, which in turn can be considered as a whole behaviour since it includes a whole set of basic actions. On the other hand tasks like navigating through a maze can be dealt by simple actions like moving forward and turning left or right. The complexity of programming behaviours with object interaction is also reflected in the teaching process. Firstly, from the teacher’s point of view, as it must be really successful on its demonstration; as a failure can result in a whole series of mistakes in the demonstration, and thus in the learning of the learner. Of course, it can be argued that this is the case with every task, but it is only that these behaviours are quite complex, as mentioned, and naturally they are much more difficult to implement as well as any possible recovery routines. From the learner’s perspective it might be tricky to judge if it has learned to imitate correctly. For instance, consider the scenario where the learner has correctly imitated orienting with an object and is expected to pick it up successfully; however, because its grippers are not as powerful as the teacher’s, it fails to do so. How can the learner understand that there was no mistake in its learning process then? In the scenario of navigating through a maze\(^2\), when the learner finds a wall, let’s say

\(^1\)If automated systems are desirable is an issue which I am not going to discuss in this dissertation though.

\(^2\)We assume that there are no other obstacles on the path of the robot except from the walls of the maze; as these obstacles could be then considered as objects, resulting in behaviours that involve object interaction.
its right, then the action of turning left will always make it turn away from the wall it was facing. Even if its sensors say that there is still a wall on the right, the robot it can assume that this wall is not the same as the one encountered previously, since it is in fact not. In other words, if the learner imitates the behaviours of its teacher ‘correctly’, these behaviours will always be successful, something that is not always the case with behaviours involving interactions with objects, as explained above.

At this point what is meant by saying above ‘the learner imitates the behaviours of its teacher ‘correctly’ ’ should be explained. When agents learn by imitation it is not necessary to perform the skill optimally at this stage; and this is most of the times the case. They just need to capture the general concept of the acquired skill. For instance an agent that has learned how to orient with an object, is not required to do it in the least possible moves. The optimisation of the skill is a further step after the learning and constitutes another area of research. The strength of imitation learning lies in the fact that the learner firstly learns quickly, and secondly it learns from its own perceptions of the environment and using its own capabilities to imitate an acquired skill; making the need for reprogramming the robots obsolete; or being more precise ‘cutting it down’ to the need for programming the imitation learning mechanisms for acquiring new skills and possibly some optimisation algorithms (like reinforcement learning) for improving them over time.

Lastly, as already mentioned, there are big disagreements among the ethologists on the terminology used to explain imitative phenomena; Galef (1988) mentions 22 instances of imitation learning. The reason for this chaos may lie in the fact that these complex mechanisms of imitation learning are not very well understood yet. Studies on human infants (Meltzoff and Moore, 1989) and on animals have been applied, in order to explain these mechanisms and bring some light on the field, not only from the ethological and psychological perspective (Galef, 1988; Miklosi, 1999; Tomasello et al., 1993; Skoyles, 1999), but from the neurophysiological one as well (Rizzolatti et al., 1996). However, it is difficult for an ethologist to test his theories on behaviours; firstly because it is hard to include all the factors that influence a behaviour in the model, and secondly the experiments are difficult to replicate since their repetition depends upon the ‘mood’ of the animals. A solution to this, is to test these theories on
robots, making possible the replication of the experiments. Someone may argue that the robots’ hardware cannot capture all the physical characteristics of the animals. But, this is not the case. Hallam and Hayes (1992) show that there are a lot of correlations between robots and animals, which allow us to simulate animal characteristics with robotic ones. Another argument against using robots to explore ethological concerns and ideas is the time required to run an experiment in comparison to the time needed to run it on a computer simulation. We also agree that the run time is much longer, but on the other hand a computer simulation cannot capture all the characteristics of the environment and all the features of the studied model; also assumptions made for the simulation do not apply in the real world (Hallam and Hayes, 1992; Brooks, 1992).

Finally, the motivations and the inspirations for this project can be summarised to:

1. Imitation learning cuts down the time required to program robots.

2. It allows cross-modal learning; a human can demonstrate a skill to a robot, which in turn can teach it to another robot, and so on.

3. From an industrial point of view, it is a step towards fully-automated systems.

4. From an ethological perspective it provides ethologists with a platform where they can repeat their experiments in a similar environment. Furthermore, researchers will develop computational models that could be used to explain the complex mechanisms of imitation. Finally, all these experiments could join the gap that exists in the terminology of describing imitative phenomena.

1.3 Abstract Description of the Research Scenario

In this project we are interested in learning by imitation behaviours that involve object interaction in a high level of abstraction, which was inspired by Byrne and Russon (1998)’s hierarchical approach. By high level of abstraction we mean that the learner robot learns to imitate and then associate hand-coded behaviours, which are demonstrated by a teacher robot, with its environmental perceptions, i.e. it learns what it should do in every case, knowing, however, *a priori* how to do it. This is achieved by
having the teacher robot leading the learner through a series of experiences where the learner will have the opportunity to form these associations.

Furthermore, there are a lot of types of imitation learning. We are interested in one, called ‘matched dependent behaviour’, in which an agent tries to match its behaviours to a demonstrator’s behaviours as best as it can.

The demonstrated scenario we chose is a foraging task since it involves behaviours with object interaction. A robot moves in an arena, trying to find white cylindrical pieces of wood, which represent food, and either collects them and brings them to a specified area of the arena considered to be its home (by carrying or pushing), or alternatively it avoids them if they are poisonous.

### 1.4 Objectives

The principal goal of this project is to implement the above learning scenario and experiment on how well the learner robot performs foraging in the absence of the teacher, after it has been trained.

The implementation of this high level of abstraction scenario will provide a platform for conducting more experiments. For instance, we said that the behaviours, which the learner learns to associate with its perception, have been implemented by us. A further project would be to learn these behaviours by imitation. Then, by combining each one of them with this work, it would result in a complete imitation learning system.

Along the way of the project another goal appeared. The software used for controlling the robots did not support control for more than a single one. Hence, extending it to allow us to control multiple robots became a necessity.

The objectives of this project can be highlighted to:

- Study imitation learning in behaviours that involve object interaction at a high level of abstraction.

- Provide a platform for further research, where more experiments can be conducted.
• Extend the robot control software to support control over multiple robots.

### 1.5 Overview of the Dissertation

In this chapter the motivations for studying imitation learning, along with a brief description of the research scenario and our main objectives have been presented.

Chapter 2 presents the background information. In the first section we describe some types and levels of imitation, as well as a few important terms that must be explained. The following section briefly presents studies of imitation learning in cognitive science and in nature, where roboticists often draw their inspirations from, in order to design robots capable of dealing with real world conditions. A description of robotic research in imitation learning is then presented, followed by a highlighting of the research that is closely related to ours.

Chapter 3 presents the physical set up, describing the environment, the robot and the software used.

Chapter 4 discusses in detail the designed framework. Both the training scenario and method are described, as well as how the learner is able to perform foraging in its recall phase on its own. The use of facilitation (such as communication between the teacher and the learner) provided by the social environment in order to enhance imitation learning is also discussed.

Chapter 5 presents the experimental design and discusses these experimental results.

Chapter 6 concludes over this dissertation. Our achievements are described. Our conclusions and suggestions for further work follow next.

Appendix 1 shows our experimental data.
Chapter 2

Background

The chapter starts by presenting some of the types and models of imitation learning trying to explain the underlying mechanisms; paying close attention to the types and the models that we investigate. Roboticists are inspired from human and animal behaviours, which cope so easily with the complexities of real environments. This is why studies of imitation learning in humans and animals are mentioned next. The chapter concludes by presenting work done in imitation learning on robots and explaining where our work fits in.

2.1 Types, Levels and Terminology of Imitation Learning

Imitation could be defined as trying to match our behaviours to the behaviours of an observed agent. In the same sense, imitation learning could be defined as the learning process where a learner learns to perform a task by matching his behaviours to the behaviours of a demonstrator/teacher, such that the learner is able to perform the task in the absence of the teacher.
2.1.1 Matched Dependent Behaviour: A Type of Imitation Learning

However, these definitions for imitation and imitation learning are quite wide. Ethologists use different terminology to describe imitative phenomena depending on how each one of them perceives these phenomena. Galef (1988) mentions and clarifies 22 types of imitation learning that have recently been developed to describe imitative learning instances.

From those 22 types of imitation learning, we are interested in one, called ‘matched dependent behaviour’. According to Galef it “refers to situations in which the application of external reinforcement leads organisms to match their own behaviour to that of conspecifics” (Galef, 1988, page 19). Hayes and Demiris interpret that as “the initial stimulus for the learner is the behaviour of the teacher which indicates ‘the right thing to do’ to gain some reward. The learner matches its behaviour to the teacher’s and eventually associates the other stimuli in the environment at the time (presumably those which stimulated the teacher to do the right thing in the first place) with the appropriate action, so that it can later carry out that action in the absence of the teacher” (Hayes and Demiris, 1994b, page 2). Matched dependent behaviour instances are quite common in nature. A frequent example is of a human teenager who is likely to be reinforced to start smoking by observing his friends smoking.

Presenting the remaining 21 types of imitation learning is beyond the scope of this dissertation. These are explained in detail in (Galef, 1988).

2.1.2 Levels of Imitation Learning

Apart from the various types of imitation learning that exist, both ethologists and roboticists have distinguished between different levels in imitation learning.

Demiris (1999) distinguishes between three levels of imitation\(^1\):

**Basic imitation:** which can be seen as reproduction of the perceived stimulus, for example, imitation of body movements or speech sounds.

**Functional imitation:** for example picking up an object, moving towards a door, making a sound to scare off a predator, etc. Essentially, in

\(^1\)Adopted from (Demiris, 1999, page 10)
this level it is not the exact stimulus that is being imitated but rather the function or effect that it has.

**Internal state imitation:** i.e. imitation not of the external action but of the presumed internal state of the partner (for example, making a sad face when another one is crying, smiling when others laugh). This can also be thought as empathy, or social attunement.

Byrne and Russon (1998) distinguish a hierarchy of levels of imitation; the low level ‘action–level’ imitation and the higher level ‘program–level’ imitation. In the former the learner imitates basic actions of a behaviour while in the latter the learner imitates ‘novel behavioural strategies’ observed from the teacher. Hence, an agent could use action-level imitation to learn each behaviour individually and then program-level imitation to develop an organisational framework for achieving a goal.

Even if action–level and program–level imitation levels are similar to Demiris’ distinctions of basic and functional imitation, Demiris and Hayes (1997) clearly distinguish in the same sense between ‘learning to imitate’, where the robot learns what to do with its motor system in order to perform the same behaviour as another robot; and ‘learning through imitation’, where “the robot learns by imitating the other agent, and associating perceptual experiences with this motor act” (Demiris and Hayes, 1997, page 2).

Our work lies exactly at this program level, the learning through imitation form of imitation learning. A teacher robot leads a learner robot through a series of perceptual experiences and demonstrates to the learner the corresponding behaviours. The learner robot imitates the teacher’s behaviours and at the same time it associates these behaviours with its perceptions (the perceptual experiences the teacher is leading him to). As a result, the learner will be able to perform these behaviours on its own next time.

### 2.1.3 Terminology

Some of the terminology keywords must be explained or clarified in order to avoid any confusion.

Firstly, ‘raw data’ are the raw readings of the sensors. Sometimes these raw data are not really meaningful, and thus not useful at all. This is why they are transformed to
meaningful values. For example, knowing our position and the position of an object is not really useful, by converting these coordinates into the euclidean distance between us and the object we only then understand how away we are. A *state vector* consists of a set of these meaningful transformations needed to determine the state. A *state*, is a situation at which the robot is. We label these states with symbolic names, called *state labels*, describing the situation, like AT_HOME_WITH_FOOD.

A necessary clarification must be made between what is an action and what is a behaviour. An *action* is a simple change to the motor system of the robot, like turning left or closing the gripper to grasp an object. Usually, actions result in a change of a single attribute in a state vector. A *behaviour*, on the other hand, is more complicated. It is a higher level of abstraction in the organisational structure of a program according to the ‘subsumption architecture’ (Brooks, 1986) and consists of a set of actions and maybe other simpler behaviours as well. For example, a behaviour that searches for a food object would consist of an obstacle avoidance behaviour and simple movements towards the object. Hence, a behaviour changes more than one attribute value in a state vector.

In our project we are dealing with teaching a robot the *state–behaviour associations*, which are a mapping of the state vectors to their corresponding behaviours. During the learning there are *imitation phases* (also the terms *training phase* or *learning phase* are used interchangeably), during which the learner robot imitates the behaviours demonstrated by the teacher, and *following phases* during which the learner tries to maintain the same position as the teacher, in order to be at the same perceptual state with the teacher. The series of these imitation and following phases compose a *training episode*. It must also be highlighted that we are dealing with *immediate imitation* (in contrast to *delayed imitation* and *simultaneous imitation*), where the learner imitates the behaviours just after their demonstration by the teacher.

In the next section the studies of imitation learning for animals as well as for the human culture and for the development of language are briefly mentioned, in order to highlight the significance of imitation learning in nature.
2.2 Imitation Learning in Nature and in Ethology

Imitation learning is not only considered to be a very strong mechanism for acquiring new skills but it is seen as a whole important form of cultural learning (Tomasello et al., 1993). Furthermore, imitation is an important process in the learning of speaking a language. Skoyles (1999) points out that “language itself would not exist without the ability to imitate the sound of words”.

In addition, experiments on orangutans (Miles et al., 1996), chimpanzees (Custance et al., 1997), monkeys (Visabergi and Fragaszy, 1990), rats (Heyes et al., 1992), parrots (Galef et al., 1986) and dolphins (Tillery et al., 1972) have shown that these animals have shown evidences of being able to learn by imitation; although there is a disagreement as to whether all the animals are capable of learning by imitation, however this disagreement is not concerned with the matched dependent behaviour type we are interested in. On the other hand, humans’ ability to learn by imitation is beyond doubt. Meltzoff and Moore (1989) have shown that humans start learning by imitation from a very early age, when we are just infants.

As a result, more and more ethologists are trying lately to explicate the imitative instances, as well as understand their underlying mechanisms, while roboticists are also helping by building computational models inspired by these instances of imitation learning. Overviews of imitation learning from the ethological perspective include (Galef, 1988) and (Miklosi, 1999).

From the viewpoint of neuroscience it is interesting to study the neurophysiology of imitation. Studies on macaque monkeys’ brains discovered two classes of neurons in area F5, called canonical neurons and mirror neurons. The canonical neurons are triggered when the monkey observes an object, while the mirror neurons are triggered when the monkey sees itself performing a behaviour involving interaction with the object (di Pellegrino et al., 1992; Rizzolatti et al., 1996), i.e. the mirror neurons fire when the monkey observes its hand grasping a glass for example (canonical neurons still fire from the sight of the glass). It seems then that there could be a close coupling of these neurons with learning by imitation behaviours that require interaction with objects, and therefore a framework simulating the function of the area F5 could be built based on that (Maistros and Hayes, 2000).
From the studies and the literature that were briefly presented in this section, it appears that imitation learning is a very important ability of humans and animals. Obviously, this could not escape from the interest of behaviour-based roboticists, whose work is presented in the next section.

2.3 Imitation Learning from the Viewpoint of Robotics

This section presents research on imitation learning mainly applied to mobile robots, as well as two cases of imitation learning in assembly robotics.

2.3.1 Imitation Learning in Mobile Robotics

One of the earliest studies involved a robot that was learning to negotiate a maze by following another robot that was an expert on navigating through one (Hayes and Demiris, 1994a,b). The learner robot was following the teacher (following phase) until a significant event occurred at which point the learner remembers the position of the teacher and ‘switches’ to the learning phase. A change in the direction of the movement of the teacher or a change in the direction at which the teacher is pointing at are considered significant events, and can also be signified by ‘watch me’ signals sent by the teacher. During the learning phase the learner moves to the recalled position where the teacher was, perceives the environment and at that point it tries to imitate the action of the teacher by (trying to keep on) following it, while it associates this executed action with its perception, hence being able to repeat the action in the absence of the teacher.

Later, Demiris and Hayes (1999, 1998) introduced two architectures for implementing imitation learning on robots, passive imitation and active imitation. Passive imitation is characterised by a three stage process ('perceive–recognise–reproduce' cycle) during which the imitator perceives the environment as well as the action demonstrated, recognises the demonstrated action and finally tries to imitate it. Although Hayes and Demiris (1994a,b)’s previous work on a robot that learned to negotiate a maze falls in this case of passive imitation, their approach did not involve a recognition stage. The imitator was not recognising the demonstrator’s actions, but it was reproducing them by trying to satisfy a goal, which was to keep its distance from the demonstrator.
constant. As Demiris and Hayes say “... the imitator is not imitating because it is understanding what the demonstrator is showing, but rather, it is understanding it because it is imitating” (Demiris and Hayes, 2001, page 2).

Active imitation, on the other hand, is characterised by the fact that the learner internally generates and tests candidate matching actions while the teacher demonstrates an action, hence taking advantage of the demonstration time by not waiting to classify the demonstrated action until it is completed. The basis for the active imitation is the use of behaviours with forward models\(^2\) (Figure 2.1). A forward model is a function that, given the current state of the model and a control action, returns the predicted next state which is fed back to the behaviour, allowing it to adjust any of its parameters. The importance of this behaviour–forward model pairing is that it can be used by the learner to match a visually perceived demonstrated behaviour with the equivalent motor one (Demiris and Hayes, 1999, 1998).

One difference between passive and active imitation is that in passive imitation, the motor system of the robot is involved only at the final reproduction stage, while in active imitation it is actively involved from the beginning (i.e. during the perception stage as well). Although the active imitation architecture seems to be better than the passive one, there is a drawback. The learner can only imitate demonstrated move-

\(^2\)In the control systems literature they are known as ‘plants’.
ments that are already present in its repertoire. As a result Demiris and Hayes (2001) combined the two architectures into one, called ‘dual–route process’, taking advantage of the benefits from both. The imitator uses active imitation to imitate the actions that already exist in its repertoire, while it uses passive imitation to imitate new ones, which are then included in its repertoire. The computational model and the experimental results are described in Demiris and Hayes (2001) and in more detail in Demiris (1999).

At about the same time that active and passive imitation were investigated, Billard and Hayes (1998) were studying how to transmit a language from one robot to another by using learning by imitation. In the experimental scenario a learner robot follows a teacher robot from behind. The teacher takes the learner through a series of observations and teaches the learner a vocabulary to describe its perceptions. This vocabulary consists of transmitted radio signals that represent single words, which the learner ‘grounds’ by associating them with its own perceptions in terms of the environment and of its motor states. As a result, in the end, the learner and the teacher robots get a common vocabulary for describing their states and actions.

Billard and Mataric (2000) have used two simulated humanoids where one of them learns from the other repetitive patterns of arm and leg movements, oscillatory movements of the shoulders and the elbows, and precise movements for reaching and grasping imaginary objects. The learning model used was the DRAMA neural architecture introduced in (Billard and Hayes, 1999). Adonis, a simulated humanoid torso, showed the ability to learn how to dance the macarena from a verbal description of the dance movements (Mataric et al., 1998). The approaches used to teach the humanoid to learn macarena were, firstly, using force–fields, where the hands were driven to the required positions by applied forces; and, secondly, using a joint–space controller with torque actuators, where these controllers were choosing a set of torques for all actuated joints of the humanoid. A summary of the two works on the humanoid avatars and on Adonis can be found in (Mataric, 2000).

Maistros and Hayes (2000) inspired by Rizzolatti and colleagues’ work on canonical and mirror neurons in pre–motor area F5 of the monkeys’ brains (Rizzolatti et al., 1996; Jeannerod et al., 1995; di Pellegrino et al., 1992), use Arbib’s ‘Schema Theory’ (Arbib and Cobas, 1990) as a framework to simulate the function of these neu-
The Schema Theory suggests a network of schemas (Figure 2.2), which schemas can be either perceptual schemas dealing with the perceptual structures of an action (e.g. joint-angles of an observed demonstrator), or motor schemas dealing with the motor skills for that action (in terms of target positions of the motors, e.g. joint-angle targets). These schemas network controls the passing of parameters between the perceptual schemas and the motor schemas. The experimental platform consists of a simulated humanoid torso that tries to imitate a demonstrator drinking a glass of beer with different behaviour-schemas\(^3\) (Maistros and Hayes, 2000).

One of the research issues in imitation learning is knowing when to imitate. Hayes and Demiris (1994a) used changes in the orientation of the teacher to signify the learning phase to the learner (when it should start paying attention to what the teacher does). Marom and Hayes (1999, 2000) proposed two statistical mechanisms to model attention in this sense. The first one uses *short-term memory* where new information is compared with the immediate previous one. The second mechanism uses *long-term memory* as well, where the new information is again compared to previous stored experience. The two mechanisms have been tested using two simulated Khepera robots.

---

\(^3\)Like drinking ‘politely’, less ‘politely’, passing the glass to the other hand, etc.
Chapter 2. Background

in a photo–taxis behaviour (finding and approaching light sources) and in an obstacle avoidance behaviour. Marom and Hayes (2001) have also investigated an alternative attention system which is based on the dynamics of the memory in response to stimulus and long–term memory. Stimuli (state vectors) that occur frequently are ‘better remembered’, while ones that rarely appear are slowly ‘forgotten’. The system was implemented by using a variation of the Self Organising Feature Map, where nodes grow from experience, rather than being specified a priori. Each node keeps a numerical value, called ‘habituation value’, to signify how frequently a stimulus is perceived. The system has been tested both in a simulated and a real environment using a wall–following scenario.

The Schema network for imitation proposed by Maistros and Hayes (2001) and the attention system using the Self Organising Feature Map by Marom and Hayes (2001) have been joined into one system (Maistros et al., 2001). The attention system takes the place of the perceptual schemas in the Schema network. The SOFM receives the raw readings from the sensors and classifies them appropriately. Each motor schema in the Schema network is now connected to one node in the SOFM instead of a perceptual schema (Figure 2.3). The experimental results on a simulated humanoid torso learning to drink a glass of beer and on a wall–following scenario with two simulated Khepera robots have shown that the imitation component takes full advantage of the attention one.

2.3.2 Imitation Learning in Assembly Robotics

Although the related work presented so far lies in the field of mobile robotics, one of the earliest studies of imitation learning was applied in assembly robotic systems. Kuniyoshi et al. (1994) taught a robot to do block assembly tasks from visual data taken from a human demonstration.

Another application of imitation learning in assembly robots, in the context of reinforcement learning, was by (Schaal, 1997). An anthropomorphic robot arm managed to balance a pole after observing for 30 seconds a demonstration given by a human.
Figure 2.3: The integrated system: the SOFM handles the stimulus and finds the best matching node at each time; its hand–wired motor schema calculates the motor commands before sending them to the motor system for execution. Figure adopted from (Maistros et al., 2001, Figure 4).

2.4 Where Our Work Fits In

Most of the work in imitation learning for mobile robots presented, either involves behaviours without any interaction with objects, or if it does, the experimental platforms are simulated robots. Our work concentrates on learning by imitation behaviours that involve object interaction. We are particularly interested in testing our work in real robots, rather than in simulated robots and environments which are “doomed to succeed”.

Furthermore, most of the related work focuses on the low action–level imitation, i.e. in imitating the motor actions of a demonstrator. On the other hand, we are interested in the higher program–level imitation, in which a robot tries to imitate the organisational structure of the behaviours of a demonstrator. This approach is a modular approach, meaning that changes to a behaviour will not affect this organisational structure, keeping any problems of a behaviour within its scope.
Chapter 2. Background

2.5 Summary

In this chapter the levels and types of imitation with the terminology used in this dissertation were described, followed by illustrating some research from the view of ethology and neuroscience. The chapter concluded by presenting related work on imitation learning mainly applied to mobile robots and humanoids as well as a couple of studies on assembly systems. An overview of imitation learning on robots can also be found in (Schaal, 1999). In the next chapter the system setup for our experiments is described.
Chapter 3

System Set-Up

In this chapter the physical implementation of the experiment is introduced. Firstly, the arena in which the robot moves, and the objects it collects, which are supposed to be the food, are described. The robot and its control software are then described, followed by a description of the camera and the robot tracking software.

3.1 Environment

In the mobile robotics lab there is a large arena ($2.4 \times 2.4 \text{ m}$) with an overhead black and white camera above the centre of the arena, for tracking Khepera robots used in the department’s experiments. Within, this arena we built smaller training and evaluation arenas for the purposes of our project.

The training arenas are two corridors of $230 \text{ mm}$ wide by $650 \text{ mm}$ long (Figure 3.1). Each arena is occupied by one robot, i.e. one arena is occupied by the teacher robot and the other by the learner. One arena is at an offset from the other, in the sense of where the robots start, where the food lies and the location of their homes.

The evaluation arenas consist of one corridor like the training one and one larger rectangular arena, of $600 \text{ mm}$ in width by $650 \text{ mm}$ in length.

The floor of the arena and of the walls are of a medium intensity colour such that it is easier for the camera tracking program to detect the robot white (high intensity) markers in contrast to the intensity of the arena.
The objects (Figure 3.3), used as the food, are cylindric objects made of wood such that they are light enough for a Khepera to lift. They are 35 mm in diameter and 23 mm in height, and are covered by white tape to be easier for the camera tracking program to detect.

3.2 The Khepera Robot

In our experiments we used two K–Team™Khepera™ robots (Figure 3.4). Khepera is a miniature robot of just 55 mm in diameter, 30 mm in height and weighs just 70 grams. It has 8 infrared, ambient light sensors, 6 in front and 2 behind, and 2 DC
Chapter 3. System Set-Up

Figure 3.3: The objects used to represent the food. They are 35 mm in diameter and 23 mm in height. They are covered by white paper, such that it is also easier for the camera tracking program to detect.

Figure 3.4: A picture of a Khepera robot and a drawing showing the positions of the motors and of the infrared–ambient light sensors.

motors with incremental encoders. For more information on the Khepera robot refer to (K-Team, 1999b).

The Khepera is equipped with a gripper module (Figure 3.5). The size of the gripper module is $65 \times 90 \times 10$ mm. An optical barrier is attached to the fingers of the gripper to detect object presence. The grippers are also capable of measuring the electrical resistivity of a gripped object. More information on the grippers module can be found in (K-Team, 1999a). The electrical resistivity measurement was not going to be used as a sensor reading, giving us the opportunity to increase the reliability of grasping an object by covering the two grippers with sand paper (thus, losing this
Chapter 3. System Set-Up

3.3 The Khepera Simulator

In order to control the Khepera robots a free Khepera simulator\(^1\) was used (Figure 3.6), which provides functions for controlling a *single* real one. However, these functions are restricted to controlling the motors and the readings of the infrared, ambient light sensors. The gripper module routines are not implemented. However, it is still possible to control the gripper module by sending the low level commands to the Khepera. The software supports multiple Kheperas in simulation, but just a single real one. Therefore, we needed to extend the simulator, which is written in C, for controlling multiple real Kheperas\(^2\). The robots’ programs were also implemented in C.

The simulator was used mainly because it provided a convenient graphical interface for observing the sensor readings, making the programming of the behaviours much easier. Furthermore, Yuval Marom in our department had already worked with the simulator, and thus getting familiar with programming the Khepera with the simulator was easier than trying to do it from scratch. On the other hand, the unsupported routines for the gripper module and the necessity to implement the simulator for controlling more than one real robot could have proved a drawback, which fortunately they did not.

---

\(^1\)The simulator can be downloaded from [http://diwww.epfl.ch/lami/team/michel/khep-sim](http://diwww.epfl.ch/lami/team/michel/khep-sim)

\(^2\)The extended simulator can be downloaded from [http://www.dai.ed.ac.uk/groups/mrg/MRG.html](http://www.dai.ed.ac.uk/groups/mrg/MRG.html)
3.4 The Camera and the Frame Grabber

The CCD camera used by the tracking system is a black and white one. It is placed above the centre of a $2.4 \times 2.4$ m square arena (Figure 3.1). A view of the arena acquired by the camera is shown in Figure 3.7.

It must be pointed out that the camera is probably the most important sensor of the robot. Both robots use the camera to know where they are, their orientation and where the food objects lie. Furthermore, when the robot is pushing a food home, its gripper is in front of the infrareds. The camera is used in this case for obstacle avoidance.

The frame grabber is a Matrox Meteor Card, capable of sampling the camera at a maximum resolution of $640 \times 480$ at a maximum rate of 50 frames per second, i.e. every 20 msec.

3.5 The Camera Tracking Software

The camera tracking software was written by Quek (2000). It detects regions, which represent objects, and is able to keep track of them. A region is an area in an image, whose intensity is either much higher than a threshold value, (e.g. white areas), or
Chapter 3. System Set-Up

Figure 3.7: Camera view. It can be seen that cables are getting in the way of the camera and the robot and the objects making it difficult to track them.

where there is a significant contrast (again determined by a threshold value) between the region and the background (e.g. white areas in black background). Any other distinct regions within the region are considered to be holes. A region with one hole is shown in Figure 3.8. The hole is considered to be the black square within the white region. The black frame around the white region is just to make the contrast of its intensity more distinct with respect to the background.

The tracking program uses featureless or feature–based tracking in order to retrack regions that have been lost. Features of a region are considered to be its area, its perimeter, its compactness and the number of holes that it has. However, only the number of holes is used in the tracking software for the feature–based tracking, since
Figure 3.8: A 1–hole region. The inner black square is considered to be the hole of the region. The black frame around the white region is just to make the contrast of its intensity more distinct with respect to the background.

the rest of the features proved to be unreliable as they varied with the position and the orientation of the region as well as with changes in light illumination (Quek, 2000, pages 35–44). In feature–based tracking, when a region is lost, the tracking program remembers the number of holes of each detected region, such that upon redetection it is able retrack it. In featureless tracking, it is difficult to retrack a lost region, since there is no evidence that a redetected region corresponds to a previous lost one. Therefore, in our project as well, feature–based tracking was preferred over featureless, since it is more reliable. In order to distinguish between the teacher and the learner we concluded that the markers shown in Figure 3.9 provided the best reliability in tracking, after experimenting with various types of markers, in the context of size and colour of the region and the holes. From experiments with both markers, the 2–hole region is lost more frequently than the 1–hole one.

The program scans the captured frames and stores in a reserved shared memory the x–y coordinates of the first pixels of the region, the x–y coordinates of the centre of the region, the number of holes, the orientation, the area, the perimeter, a status indicating if the region is found or lost and an edge–flag indicating if the region is on the edge of the track window. From those, the x–y centre coordinates of the region, the number of holes, the orientation and status of being tracked or lost are useful in our experiments for tracking the robots and the food objects.
Figure 3.9: The markers on top of the Khepera robots that the camera software is able to detect. The regions are $120 \times 80$ mm. The size of each of the inner black squares is $20 \times 20$ mm.

### 3.6 Minimum Requirements

The Khepera simulator, the camera frame grabber and the tracking software are all installed on a PC with:

- A Pentium 75 MHz CPU
- 16 MB RAM
- Running Linux operating system

### 3.7 Constraints Imposed by the System

The camera program, although it is reliable at keeping track of the objects, occasionally still loses the robots or the food objects sometimes. As it is shown these losses become more frequent as cables get in the way of the camera and the robot regions (Figure 3.7), making it impossible for the tracking program to see them. These losses did not have any serious implications on the evaluation of the learner robot when trying to perform foraging after it has been trained. However, training was seriously compromised, since one of the key elements in the training is the learner to follow the teacher, which is impossible when the learner is ‘blind’.

A secondary constraint was imposed by the black and white camera, which did not allow us to use different colours for distinguishing between different types of food.
Furthermore, since the area, the perimeter and the compactness of the tracked regions varied from place to place and with different orientations (Quek, 2000, pages 35–44), it was impossible to use different sizes for the objects in order to distinguish again between different types of food. However, we resolved this constraint by assigning a random value, representing the food type when the robot needed to identify the food.

### 3.8 Summary

In this chapter the system set up has been described as well as any constraints that it imposes. The rest of the dissertation is concerned with discussing our framework, the experimental results and the conclusions derived.
Chapter 4

The Foraging Task

In this chapter our work is presented. A foraging task was chosen as the scenario. The reasons for choosing it and a description of it are firstly described. The method of how the teacher does foraging, how the learner learns to associate its perceptions with their corresponding behaviours and finally how it performs foraging in the absence of the teacher, after it has been trained, are discussed in detail. Our approach is based on program–level imitation. The chapter concludes by showing how imitation learning could benefit from facilitation, like communication between the teacher and the learner, which are provided from the social environment.

4.1 Reasons for Choosing Foraging as the Experimental Scenario

Foraging, i.e. searching for food and bringing it home, has been chosen to be our experimental scenario mainly due to the fact that it undoubtedly involves behaviours of interaction with objects. Although programming a robot to perform foraging is not a very straightforward task as it first appears, it is still much simpler than other scenarios like trying to program a robot to paint a canvas using various paintbrushes and a palette of different colours. This relative simplicity was desired because we wanted to spend as little time as possible on programming the teacher robot and concentrate on the learner and on the more important issues of imitation learning. However, the teacher
Chapter 4. The Foraging Task

should really be an expert in foraging. This meant that considerable effort and time had to be spent to program the teacher to perform as good as possible.

Furthermore, foraging can be a collective task where more than one agent is involved. Imitation learning is a learning mechanism that performs its best within a social environment (see section 4.7). Therefore, it seemed reasonable to choose the foraging scenario to study imitation learning. Although the learning and the investigation of collective behaviours were not principal aims of this project, the setup and the implementation of it would provide a platform for studying these behaviours in the future.

In addition, we are interested in behaviours for mobile robots which are fully autonomous and need to cope with an unconstrained world environment rather than with an assembly environment; this is another reason for preferring foraging to the peg–in–hole problem for instance.

Lastly, some ethologists and animal biologists have looked into animal foraging, feeding and interaction with food in general, not only to explain their behaviours but also to study the underlying mechanisms of imitation (Byrne and Russon, 1998).

4.2 Description of the Foraging Scenario

The scenario could be described in one sentence as: a robot searches for food objects and carries them to its home. The way that the robot transfers the food home depends on the type of the food (see section 4.2.3).

In more detail, a typical run of the robot would be as follows; it starts from a position distant from the food and not at its home, through the overhead camera it can see where the food lies and thus it starts approaching it at a fast speed; when it is close to a food object it cuts down its speed a bit in order to be more maneuverable; when it is next to a food object it starts orienting; upon successful orientation the type of food is identified and depending on its type the appropriate behaviour associated with it is executed; for instance, let’s assume that the food can be lifted up, then the robot would pick up the object and it would start moving towards its home; when the robot reaches its home it would leave the food object and then it will get out of its home and will
start looking for new food objects. In order to execute the appropriate behaviours, the robot executes the following routines at every ‘step’:

1. It perceives its environment.

2. The raw data is preprocessed to form a state vector.

3. The state vector

   • is classified into one of the possible states for the teacher robot;
   
   • or is best matched to one of the state vectors, the learner has learned, for the learner robot.

4. The corresponding behaviour is executed.

### 4.2.1 States and Behaviours

We have divided the foraging task into 10 main states and 10 corresponding hand coded behaviours. The names of the states and of the corresponding behaviours are self-evident. These names are called *state labels* in the sense that we give a label to a set of state vectors (section 4.3).

The states are mutually exclusive (i.e. two states cannot occur simultaneously), because there must be a one-to-one correspondence between the states and the behaviours. If that was not the case then under one state there would be more than one behaviour that could be executed, and a conjunction of states would be needed to determine the correct behaviour to execute. For example, consider the robot being home, and a state that classifies this but which does not determine if a food object is carried or not, then there are two possible behaviours that could be executed, leaving the object if it carries one and getting out of its home if it does not. Another state that determines if a food object is carried or not should be checked in order to decide which behaviour should be executed.

Furthermore, there is a logical sequence in the order that the states occur and hence also on the executed behaviours. This could seem that the robot follows this order in order to achieve the task of foraging. However, this is not the case. The robot
Table 4.1: The states and their corresponding behaviours.

<table>
<thead>
<tr>
<th>STATE</th>
<th>BEHAVIOUR</th>
</tr>
</thead>
<tbody>
<tr>
<td>SEARCH</td>
<td>SEARCH_FOR_FOOD</td>
</tr>
<tr>
<td>CLOSE_TO_FOOD</td>
<td>MOVE_SLOWLY</td>
</tr>
<tr>
<td>NEXT_TO_FOOD</td>
<td>ORIENT_WITH_FOOD</td>
</tr>
<tr>
<td>ORIENTED_WITH_FOOD</td>
<td>EXPLORE_FOOD</td>
</tr>
<tr>
<td>READY_TO_PICK_UP_FOOD</td>
<td>PICK_UP_FOOD</td>
</tr>
<tr>
<td>READY_TO_PUSH_FOOD</td>
<td>PUSH_FOOD</td>
</tr>
<tr>
<td>MUST_AVOID_FOOD</td>
<td>AVOID_FOOD</td>
</tr>
<tr>
<td>FOOD_IN_HAND</td>
<td>GO_HOME</td>
</tr>
<tr>
<td>AT_HOME_NO_FOOD</td>
<td>AVOID_HOME</td>
</tr>
<tr>
<td>AT_HOME_WITH_FOOD</td>
<td>RELEASE_FOOD</td>
</tr>
</tbody>
</table>

follows the procedure described in section 4.2 to decide on what to do. It does not have any memory of what has happened so far nor any knowledge of what comes next. This means that the robot does not have any knowledge of what the ‘global’ goal is, which is to collect food objects and take them home, or the ‘local’ goals, which are the goals of each behaviour (for instance it orients with a food object in order to be able to detect its type and be able to pick it up or push it). In fact our first approach involved knowledge of the local goals. The behaviours were returning the next state upon successful completion or an ‘unknown state’ upon failure. However, this is a monolithic approach, firstly because the use of states would have no sense, since then every behaviour could directly return the next behaviour to be executed and secondly there would be a problem if there were more than one state that requires a particular behaviour. Moreover, the implementation of the behaviours would become more complicated, since all the processing of the perception of the environment and any emergency routines should be implemented in every behaviour. Furthermore, in this approach the robot was perceiving the environment once and then it was executing the behaviour until it was completed (successfully or unsuccessfully). For example if the robot perceived that it was away from food, it would execute the ‘search for food’ behaviour until it is close to a food object. This means that the learner will only acquire
one perceptual experience for every behaviour and hence it should be trained explicitly at every possible perceptual state.

In our latter approach though, the robot continuously (at every step) decides on what to do based only on its current perception. It could be thought as if–then rules rather to a repeat–until loop used by our formal approach. In our latter approach it is the perception that determines the behaviour to be executed, and this is important as the robot responses to its stimuli; while in our formal approach it was the behaviours that determined the next behaviour to be executed.

4.2.2 Description of the Raw Data

Raw data is the raw readings of the sensors. The sensors used from the robot are the 6 front infrareds\(^1\), the sensors of the gripper module and the overhead camera. Thus, the raw data is:

- The six front infrared readings. These are integer values from 0–1023.
- The position of the arm motors of the gripper module. This is a value from 255 (touching the ground in front) to 0 (touching the ground behind). Actually only the range of 255 to approximately 215 (gripper raised) is used.
- The position of the gripper motors. This is an integer value from 200 (fingers open) to 0 (fingers completely closed).
- The object presence given by the optical barrier of the gripper module. It can take the value 0 or 255 and hence is considered as a boolean.
- The x–y coordinates of the robot in pixels from the overhead camera. These are float numbers.
- The orientation of the robot in degrees from the overhead camera. This is an integer number. Note that sometimes the tracking software gives the opposite direction from the actual one.

\(^{1}\)The back 2 infrareds and the ambient light sensors are not necessary.
• The x–y coordinates of the nearest food object in pixels from the overhead camera. These are float numbers. It should be noted that when the robot is next to the food and the camera needs to redetect the objects it is unable to detect the food object next to it, since it is covered by the robot region.

• The type of the food. This is an enumerated type that can have one of the values PICKABLE, PUSHABLE, POISONOUS and UNKNOWN.

• The x–y coordinates of the home and its radius are known to the robot from the beginning.

Note that the camera in conjunction with the tracking software return data about all the tracked regions. It is our program that finds from all these regions which ones correspond to the robot regions (regions with holes) and which to the food regions (regions without holes).

### 4.2.3 Types of Food and Behaviours Involving Object Interaction

The type of the food determines the object interaction behaviour. The food can be one of the following types:

**Pickable:** These food objects must be lifted by the robot and carried home.

**Pushable:** These food objects must be pushed home. Although all the object are of the same size and weight this food type represents objects that would be too heavy for the robot to lift, yet light enough to push.

**Poisonous:** These food objects are dangerous and must be avoided.

The type of food is determined randomly upon successful orientation of the robot with it.

### 4.3 From Raw Data to State Vectors

Some of the raw data, in this form, is not really helpful for the robot to determine in which state it is. For example, the x–y coordinates of the robot and of the nearest
food object do not really have meaning in the context of the foraging task; it is the (euclidean) distance between them that would help to classify the state vector. In addition, not all the raw data is needed to determine the state. For instance the infrared readings do not play any part in this classification.

Hence, we have defined the prototype of a state vector to consist of 5 attributes, which are:

1. **ObjectPresence**  The object presence from the optical barrier. It is a boolean type.

2. **RobotHomeDistance**  The Euclidean distance between the robot and the centre of its home.

3. **FoodType**  The type of the food. This is of enumerated type and can have one of PICKABLE, PUSHABLE, POISONOUS or UNKNOWN values (like in the raw data).

4. **ArmsHeight**  The height of the arms of the gripper. It is an integer between 255–215 (like in the raw data).

5. **RobotFoodDistance**  The Euclidean distance between the robot and the nearest food object.

### 4.4 How the Teacher Classifies the State Vectors to States

In section 4.3 the interpretation of the raw data into state vectors has been presented. The next step is, given a state vector to determine which state it corresponds to (Figure 4.1). Every state has its own subset of attributes from the state vector that makes it distinct, being independent of the values of the rest. The attributes that affect each state are shown in Table 4.2.

The simplest way to classify the state vectors would then be with *IF–THEN* statements. However, we preferred to use a decision tree instead (Figure 4.2), because it is faster for a large number of states. Of course in our case the number of states is quite small (just 11) and it does not really make a difference. The attributes of the state vector that would give the highest information gain\(^2\) have been chosen as the top level nodes.

\(^2\)Due to the time constraints we have calculated the information gain empirically and not through formal mathematical methods.
Figure 4.1: Transformation of the raw data to meaningful state vector that determines the state. Note that the classification of a state vector into a state is only used by the teacher.

(ObjectPresence and RobotHomeDistance). Note that there is an UNKNOWN STATE in one of the leaf nodes of the tree. This state was used during the design and the implementation of the decision tree, and it never occurs in practice.

It is important to point out that the above classification method is for the hand-coded program of the teacher. The learner, on the other hand uses another method to classify the state vectors based on its experience (section 4.6).

4.4.1 **IF–THEN** Rules: A Means of Associating Behaviours to States

The small number of states and behaviours allowed us to use simple *IF–THEN associations* between the states and their corresponding behaviours, as oppose to the decision tree used to classify the state vectors to states. If this number was larger a more complicated approach, such as an artificial neural network, would be desired in order to reduce the processing time. Thus a prototype of a general *IF–THEN* association is:

\[
IF(\text{STATE\_LABEL}_X) \quad \text{THEN} \quad (execute \ \text{BEHAVIOUR}_X)
\]  

At this point we should note that these hand-coded *IF–THEN* rules (the associations between the states and the behaviours) together with the classification of the state vectors to states (using the decision tree), are known *only* to the teacher and *not* to the learner who associates directly state vectors with behaviours (sections 4.5, 4.6).
4.5 Description of the Training Scenario

In the training scenario there are two separated arenas that are identical, each one occupied by one of the two robots as described in section 3.1. The program, due to the way the simulator works, consists of ‘MultiSteps’ and ‘Steps’. A MultiStep is like one round in a board game while a Step is a player’s turn (Figure 4.3). Note that the teacher robot always plays before the learner.

The teacher on its step performs foraging as it would do if it were on its own, with the following additions to its routine:

- It ‘shouts’ the behaviour that it executes.
- It stops moving and does nothing when the learner has been separated from him.

Hence, in every step the teacher firstly sees if the learner is separated in which case it pauses. Otherwise it:

1. obtains the raw data and transforms them into a state vector;
2. classifies the state vector into a state (using the decision tree);
3. retrieves the corresponding behaviour to the state (using the IF–THEN associations);
4. ‘shouts’ the retrieved behaviour for the learner to hear;
5. executes the behaviour.

The learner, on the other hand, performs the following processes on its every step:

1. It obtains the raw data and transforms them into a state vector.
2. It listens to the behaviour shouted by the teacher.
3. It records (into its ‘experience’ file) its state vector and the behaviour it just heard. We call these vectors ‘experience vectors’. In this way the state vector–behaviour correlations are drawn.

---

3 As a reminder, the dimensions of the two arenas, the starting position of the robots, the position of the food object and the position of the home are identical.
4. It then executes the behaviour.

5. Finally, it follows the teacher in an attempt to maintain identical spatial positions.

### 4.5.1 The Logic Behind the Learning Method

The learner, by recording its state vector and the corresponding behaviour (‘experience vector’) in its experience file, learns which behaviour should perform in a similar situation. This behaviour should be the correct one since the learner is in a similar situation (identical position) as the teacher who knows how to classify its state vectors correctly and hence execute the appropriate behaviour. The learner then executes the behaviour it has just been told, resulting in imitating the teacher. *This imitation is necessary for the learner in order to be in the same perceptual state as the teacher.* For example, if the learner did not imitate picking up the food, it would not know ‘how it feels’ to hold the food.

However, due to likely different configurations of the two robots and noise from the environment (like different frictions) the position of the learner may not be the same as the teacher’s. In fact, we observed that although the behaviour functions are the same, the two robots execute them slightly differently. For example, the two robots do not approach the food objects along the same path, although they should. This occurs due to additional noise on their sensor readings. On the other hand, of course we do not expect the two robots to execute behaviours such as orienting with food exactly in the same way. This is why the last thing that the learner does is to follow the teacher in order to reach the same spatial position.

It is worth mentioning that in our preliminary approach, the teacher was shouting both the state it was in and its corresponding behaviour. The learner was listening to the shouted state and behaviour and was recording these as a pair in its experience file. The learner also knew how to classify the state vectors to states and the only thing it did not know was the associations between the states and the behaviours, which were built up in its experience file. Obviously, this method could be considered as a mere copying of the teacher program to the learner. Furthermore, the learner does not really associate its direct perceptions with behaviours rather it needs to classify them to states.
first. This classification was also hand-coded, which is not the case in the final version of our learning method, since there is no need to ‘put labels’ to the perceptions; they are directly associated with the corresponding behaviours. This state (label)–behaviour associations could also be considered as program–level imitation, but in a very trivial form.

4.5.2 On Following and Identical Arenas

It now becomes clear why we need the two separated arenas and setups to be identical. The teacher and the learner must be at the same perceptual state in order for the learning to make sense.

However there are other, more natural ways of following with the robots in the same arena, but these would not work in our case. For instance, the learner could follow the teacher from behind. However, the learner behind would not be able to ‘do’ what the teacher is doing. When the teacher would execute a behaviour such as picking up the food; then there would be no food for the learner to pick up. Furthermore, the learner should have a delay on when to imitate, as it would have to move to the same perceptual state as the teacher. The teacher should also ‘leave space’ for the learner. All these things make our scenario impossible to implement.

An alternative way of following would be side–by–side, each robot having its own piece of food to collect. However, by having the two robots side–by–side in the same arena, ‘evasive’ behaviours, for not bumping into each other, would have to be implemented. These evasive behaviours should also take care of the side–by–side movements, making their implementation even more difficult.

Ideally, we would like the learner to follow the teacher by being ‘inside’ of it, but perceiving the environment with its own capabilities. Although, this could be possible in a simulator, in the real world it is physically impossible.

This is why we simulated it by offsetting the two arenas and setups. A disadvantage of it, is that it is also physically impossible to set up the two arenas to be exactly identical. However, with a bit of effort a small displacement did not pose any problems.

The following behaviour we implemented is a simple one. The learner tries to reach the position of the teacher by changing its direction of movement while moving, i.e.
the learner moves in arcs to approach the teacher. It does not alter its speed according to its position to the teacher (i.e. slow down if it is ahead or speed up if it is behind). The learner does not follow the teacher if they are in about the same position or in the same direction. If the two robots get significantly apart, the teacher robot stops and waits for the learner to match its position. The learner uses the same behaviour for recovery as well.

### 4.6 The Recall Phase of the Learner

In the recall phase the learner robot should be able to negotiate a foraging task in the absence of the teacher robot, without any help at all. Ideally, it should be able to perform as well as the teacher does.

The learner goes through the following processes at its every step:

1. It perceives the environment.
2. It transforms the raw data into a state vector, its *input state vector*.
3. The input state vector is then compared with all the state vectors from the experience vectors written in its experience file, in order to find the most similar ones with the input one.
4. From these experience vectors the associated behaviour is finally executed.

#### 4.6.1 The Nearest–Neighbour Algorithm

The state vectors most similar to the input state vector are found by using a form of a *nearest–neighbour* algorithm. Recall from section 4.3 that a state vector consists of the boolean attribute ObjectPresence, the enumerated FoodType and the integers ArmsHeight, RobotFoodDistance and RobotHomeDistance. The similarity of the input state vector with the state vectors of the experience vectors is determined by a float number representing the *distance* between them, i.e. the smaller the distance the more similar they are. The distance is calculated in the following way:

- For the boolean attribute, ObjectPresence, add one if it is different.
• For the enumerated attribute, FoodType, add one if it is different.

• For the integer attribute, ArmsHeight, calculate the difference between the input value and the value of the experience vector and normalise it between 0 and 1 by dividing over the maximum difference. This value is then added to the distance (Equation 4.2). Note that although the maximum difference is 255 in our case we only raise the arms up to a certain height. Thus, the maximum difference is 55. In fact we are using the range 255–215 but the maximum difference is chosen to be bigger for safety reasons. This, however means, that the attribute distance cannot exceed one, which does not though have any impact on the process of finding the most similar state vector.

\[
distance = distance + \frac{|X_{in} - X_{exp}|}{X_{max}}, \text{ where } X_{max} = 55. \quad (4.2)
\]

• The attribute distance of the integer attribute RobotFoodDistance, is calculated in the same way as the ArmsHeight one. The difference between the attribute value of the input state vector and of the value in the experience vector is divided by the maximum distance, which is the length of the diagonal of an arena, in order to normalise them between 0 and 1 (Equation 4.3). The normalised attribute distance is added to the distance.

\[
distance = distance + \frac{|X_{in} - X_{exp}|}{X_{max}}, \text{ where } X_{max} = \text{length of the arena diagonal}. \quad (4.3)
\]

• The attribute distance of the integer attribute RobotFoodDistance is calculated exactly in the same way as the RobotHomeDistance attribute (Equation 4.3).

Thus, the maximum distance that the input state vector and an experience state vector can have is 5 (in fact less than 5, due to the ArmsHeight difference). In our nearest neighbour algorithm the perceived state vector is compared in turn with all the state vectors in the experience file. A state vector (in the file) will always replace the previous best matched one, if its distance is smaller than the best. An alternative would be to have some kind of threshold similarity, where vectors with small differences are included in the list of best matched ones, instead of replacing them. We have
done a couple of experiments with this kind of threshold, but with poor results. The reason is that the distances of the state vectors may differ a little, and thus irrelevant ones were also appended in the list of the best matched state vectors. Finding an appropriate threshold would be time consuming. A solution, could be to scale the attribute distances, such that there is a distinct distance between them. However, this would rely on even more heuristics that are possibly time consuming to determine the correct parameters.

Thus, extracting only the vectors with exactly the same minimum distance is equivalent to using a threshold of 0. These state vectors do not always correspond to a single behaviour. This is why we used a confidence level to select the most ‘likely’ behaviour.

### 4.6.2 Confidence Level

We shall attempt to explain the confidence level through an example. Consider that there are 5 experience vectors which have the same minimum distance from the input state vector. From these 5, 4 of them are associated with behaviour x and 1 is associated with behaviour y. Then we select to execute with a probability of \( \frac{4}{5} = 0.8 \) behaviour x and with a probability of \( \frac{1}{5} = 0.2 \) behaviour y. This probability is what we call confidence level. Hence, the confidence level signifies how likely it is to choose each behaviour.

According to our training logic, correct experience vectors should dominate over the noisy vectors in the long run. This means that behaviours with high confidence levels are more likely to be the correct corresponding behaviours. However, we preferred to use probabilities for choosing among different extracted behaviours than always executing the one with the highest confidence level. It should be also noted that if there is only one behaviour, then this behaviour has a confidence level of one.
4.7 Teacher–Learner Communication as Social Facilitation

Someone may argue that by having the teacher telling the learner what it should do in every step we miss the concept of imitation learning; in the sense of observing someone’s actions/behaviours and then trying to imitate these actions/behaviours. This might be true when the teacher is not really a teacher, i.e. he is not there to teach us, rather ‘it is doing its own business’. However, in our case we are dealing with ‘intended teaching’, where the teacher is there to explicitly help us learn. Therefore, we should take full advantage of the social facilitation provided, one of which is communicating with the teacher. Demiris and Hayes also point out that “if we want our robots to achieve the degree of imitation displayed in natural societies, we have to provide the robots with comparable levels of social support” (Demiris and Hayes, 1997, page 3).

For example, consider learning the shooting technique\textsuperscript{4} in basketball. Correct shooting technique in basketball consists of a series of movements that take place too fast for someone to observe all the details. Just to point out that shooting in basketball is not trivial at all the method is: the ball should be grabbed by the fingers and not by the inner palm, the two thumbs should form a 90 degree angle between them, the ball should pass from in front of the eyes, the elbow breaks when the ball is just above the forehead and just after the elbow breaks, the wrist breaks too to shoot the ball. As it can be seen the shooting technique in basketball is quite complicated and some important details have also been left out, since they can only be described with a figure or by demonstration. A human, of course, could learn the technique on his own, by watching a game of basketball on television or watching an education video tape. However, some of these actions or details would slip from the attention of the learner. On the other hand by joining a basketball club the learner has the opportunity of having an expert teacher that would show him how to shoot, draw his attention into important points and errors that the learner could possibly do when imitating. Even further, the teacher can provide feedback to the learner as an expert and as an outside

\textsuperscript{4}Note the difference between learning the shooting technique and learning to shoot in order to score. The former can be acquired through imitation, since it is not requested to score the ball. The latter can only be learned by a lot of practising (trial and error).
observer, since the learner cannot see himself. Undoubtedly therefore, communication in natural societies is a very important social facilitation.

In our case, the teacher robot is also there to help the learner to learn how to forage. This can be seen by the fact that the teacher stops doing anything and waits for the learner, when the learner is separated from the teacher. The teacher shouts to the learner what it should do at every step. The teacher appears to be not a very good one in teaching, since it does not provide any feedback to the learner. However, this is not always the case, as the feedback provided could prove to be harmful than beneficial. Tan (1993) raises this question (among his other conclusions) of up to which point information exchange is beneficial and rather than noisy and harmful, driven by a series of experiments he performed in independent and cooperative learning in a simulated hunting scenario. In our case, except from the teacher telling the learner what to do, we did not provide any more information exchange between them, since this would be quite complicated to solve in the time we had.

Furthermore, having the teacher telling the learner what it should do was the only way for the learner to identify what the teacher is doing. The size of the Khepera in conjunction with the long zoom of the camera, did not permit much visual information.

Thus, even this minimum in our case communication also shows that imitation learning performs its best in a social environment by making use of the provided facilitation, overcoming any constraints imposed by the environment.

4.8 Program–Level Imitation

The scenario discussed was inspired by the hierarchical approach of imitation (Byrne and Russon, 1998), in particular to the higher level of program–level imitation (Figure 4.4). The learner learns to associate its perceptions with a behaviour that should be executed. Although the behaviours are hand-coded the same approach could also be used to learn these actions (action level imitation). The result would be a complete imitation learning system.
4.9 Summary

In this chapter the foraging scenario was explained. The proposed training method for program–level imitation was then discussed. The implications of this scenario appear to be threefold. Firstly, there is no need to program the learner from scratch. Secondly, the learner should be able to generalise beyond the scope of its training arena, being able to negotiate foraging in any arena. Finally, we also highlighted how imitation learning could benefit from social facilitation, like communication between the teacher and the learner agents.

The experimental results and their discussion follow in the next chapter.
<table>
<thead>
<tr>
<th>STATE</th>
<th>ATTRIBUTES</th>
</tr>
</thead>
<tbody>
<tr>
<td>SEARCH</td>
<td>ObjectPresence == false, RobotHomeDistance &gt; HomeRadius, RobotFoodDistance &gt; RobotCloseToFoodThreshold</td>
</tr>
<tr>
<td>CLOSE_TO_FOOD</td>
<td>ObjectPresence == false, RobotHomeDistance &gt; HomeRadius, RobotFoodDistance &gt; RobotNextToFoodThreshold, RobotFoodDistance &lt; RobotCloseToFoodThreshold</td>
</tr>
<tr>
<td>NEXT_TO_FOOD</td>
<td>ObjectPresence == false, RobotHomeDistance &gt; HomeRadius, RobotFoodDistance &lt; RobotNextToFoodThreshold, FoodType == UNKNOWN</td>
</tr>
<tr>
<td>ORIENTED_WITH_FOOD</td>
<td>ObjectPresence == true, RobotHomeDistance &gt; HomeRadius, FoodType == UNKNOWN</td>
</tr>
<tr>
<td>READY_TO_PICK_UP_FOOD</td>
<td>ObjectPresence == true, RobotHomeDistance &gt; HomeRadius, FoodType == PICKABLE</td>
</tr>
<tr>
<td>READY_TO_PUSH_FOOD</td>
<td>ObjectPresence == true, RobotHomeDistance &gt; HomeRadius, FoodType == PUSHABLE</td>
</tr>
<tr>
<td>MUST_AVOID_FOOD</td>
<td>ObjectPresence == true, RobotHomeDistance &gt; HomeRadius, FoodType == POISONOUS</td>
</tr>
<tr>
<td>FOOD_IN_HAND</td>
<td>ObjectPresence == true, RobotHomeDistance &gt; HomeRadius, FoodType == PICKABLE, ArmsHeight == CARRY_HEIGHT</td>
</tr>
<tr>
<td>AT_HOME_NO_FOOD</td>
<td>ObjectPresence == false, RobotHomeDistance &lt;= HomeRadius</td>
</tr>
<tr>
<td>AT_HOME_WITH_FOOD</td>
<td>ObjectPresence == true, RobotHomeDistance &lt;= HomeRadius</td>
</tr>
</tbody>
</table>

Table 4.2: The attributes of the state vector that determine each state.
Figure 4.2: Decision tree of the classification of the state vectors to states. Right branches are true values, left branches are false.
Figure 4.3: A MultiStep consists of (number of robots) Steps. A MultiStep is like a round in a board game, while a Step is like a player’s turn.

Figure 4.4: Hierarchical imitation: program–level imitation and action–level imitation
Chapter 5

Evaluation

In this chapter the evaluation of the program–level imitation that we implemented is illustrated. The chapter starts by explaining the experimental setup followed by a discussion on the experimental results.

5.1 Why Evaluating Imitation Learning Does Not Have A Definite Answer

We spend some time thinking on how we should evaluate our work. Obviously, one evaluation design would be to train the learner and then leave it to forage. If it manages to deliver a food object at its home, or even better as many as the teacher does, then we could conclude that imitation learning is successful. On the other hand if the learner failed to deliver a single piece of food, then we could argue about the performance of imitation learning.

But is that an appropriate evaluation method? For example, the robot could have learned to approach a food object, orient with it and pick it up but does not know what to do with it. This could be the case because maybe during its learning phase the teacher failed to demonstrate this behaviour, or the teacher has not led the learner trough this experience (yet). It would be a mistake then to say that imitation learning is not useful. Furthermore, the case of imitating correctly complex behaviours on the first attempt is rare even for humans. This is why teachers repeat their demonstrations, such
that the students can catch any missed details. Moreover, expressions like ‘students, forget what I’ve shown’ are quite common among human teachers, when they see the students failing to imitate the demonstrated behaviour, because they have perceived it completely wrong. The teacher then demonstrates the behaviour again, maybe in a simpler manner, while the students are observing it as if it was the first time.

What we are trying to point out is that evaluating imitation learning systems is a complex task that does not involve a definite answer. One needs to consider the individual steps and the subgoals.

### 5.2 Experimental Design

All these fit in the evaluation of our work. When the learner fails to bring a food object home after it has picked it up, it might be because it needs more training, *since correct correlations would dominate over noisy ones in the long run*. How much training is necessary before the learner performs well is an experimental issue, which determines the performance of imitation learning.

One parameter that we use to evaluate our work is by measuring how many times the teacher has to demonstrate to the learner the task like ‘if it was the first time’ until a complete episode is achieved. This is based on the ‘forget what you I’ve shown you’ expressions when a training episode goes completely wrong, mainly because the two robots have been so much separated that recovery takes a long time.

For the training of the robot we used two separated corridor arenas as described in sections 3.1. By the use of corridor arenas we are ‘biasing’ the learner to follow the teacher. This would make the coding of the follow behaviour easier.

We did that to demonstrate that the training environment could be independent from the evaluation one, thus giving us the opportunity to take advantage of the facilitation provided by the environment in order to make the training easier. In our case, by using the corridor arenas we are ‘biasing’ the learner to follow the teacher up to a point, making the implementation of the follow behaviour easier and less time consuming. The behaviours involving interaction with objects (pick-up, push and avoid) are taught
in separate training episodes due to hardware restrictions. The set-up is shown in Figure 5.1. Unfortunately, due to time constraints we only trained the learner for the ‘pick up’ behaviour.

### 5.3 Learning Phase

A complete training episode is considered to be when both the teacher and the learner deliver their food objects on their homes and get out of it. The training setup is shown in Figure 5.1. This training setup has been run until we obtained 10 complete training episodes for the pick up behaviour.

#### 5.3.1 How Much Time For a Complete Training Episode

The completeness of a training episode is mainly affected from the tracking of the robots. Although the camera is quite reliable, it happens to lose the robot(s) once in a while for the reasons explained in section 3.7. During the training phase, it is

---

1The cable of the second Khepera is not long enough to allow a longer corridor.
important that the robots are tracked all the time. Firstly, because behaviours that involve approaching the food objects are determined by the positions of the robots and the food. Losing the robot(s) means that the robots may be approaching the food in different paths, and hence they be in different perceptual states. Secondly, it is impossible to follow someone ‘blindfolded’.

Although, our recovery technique should take care of situations like that, it usually takes a long time. The narrow arena may ‘bias’ the robots to maintain same paths. However, it does not allow flexible maneuverability. As a result, the recovery behaviour takes a long time, and thus it we find it faster to stop the episode and start from the beginning. Having more sophisticated follow or recovery behaviours might be desirable, but time consuming as well.

However, complete episodes are achieved quite frequently, provided that there are no tracking losses. Furthermore, we can break down the whole episode into smaller ones. When we see that an episode is about to go wrong, we can stop the robots and place the learner approximately in the same position as the teacher, continuing from that point onwards (keeping all the previous experience). This does not compromise our aim; it is merely a pause and resume technique.

5.4 Recall Phase Evaluation

In the recall phase we are asking the learner to forage by using the experience files obtained during its training (section 5.3). The learner is given limited amount of time (i.e. steps) to collect the food objects. This period depends on the size of the arena as well as on the number of the available foods. This period is only an empirical estimate, based on our observations of how much time the learner should have. A more formal approach would be to measure how many steps the teacher robot needs on average. However, due to time constraints we did not do any experiments of this sort.

When all the food objects are carried home or time runs out the evaluation episode ends. The evaluation measurement is a percentage score, which is calculated in a two step process.

1. Firstly, we count the number of ‘correct’ behaviours (i.e. the ones the teacher
would execute) and we normalise them to a percentage by dividing with the total number of executed behaviours (steps).

2. A penalty is then applied, depending on how many food objects the learner did not collect and are still lying in the arena. There is no bonus for delivering a food object at its home.

Hence, the score calculation can be expressed with the following equation:

\[
\text{Score} = \frac{\text{CorrectBehaviours}}{\text{TotalNumberOfSteps}} - \text{Penalty for Uncollected Food} \quad (5.1)
\]

The number of correct behaviours of the learner is a measurement of how well it has learned to forage. However, this is not an objective criterion as it does not capture the whole scope of the foraging process. For instance, the learner could spend most of its time ‘correctly’ searching for food. Therefore, the learner could achieve a high score of correct behaviours without bringing any food to its home or doing anything other than searching.

This is why we decided to give a penalty for any uncollected foods left in the arena in addition to the time limit. The value of the penalty depends on the number of the available for collection food objects. Note that this penalty is only an empirical estimate, based on common sense rather than experiments.

5.4.1 In the Corridor Training Arena

5.4.1.1 Experimental Setup

Firstly, we evaluated the performance of the learner in its training arena. Due to the small size of the arena we used only one available food object, which is placed in various positions in the arena. The time that the learner was allowed to forage was 500 steps. The penalty for failing to collect the food was defined to be -20%. The starting position of the robot and the position of the food was random in every recall episode.

We want to investigate how the performance of the learner is affected as its experience increases, i.e. with respect to the number of the training episodes it has gone through. Naturally, we are expecting the performance to increase as the learner gains more experience. However, there is the question of how this performance increases,
i.e. there might be a point where the performance does not increase significantly as more experience is gained.

For our experiments, firstly, we had the learner go through 1 training episode (1 experience file) and then using its experiences from that we did 4 recall episodes. We repeated this 5 times, thus giving us 20 recall episodes (attempts to forage). Next, we did the same but we had the learner go through 3 training episodes this time. Similarly, for 5 training episodes. Finally, we had the learner go through 10 training episodes and using its experiences we did 10 recall phases (instead of the 20 for 1, 3 and 5 training episodes).

5.4.1.2 Experimental Results

The average scores of using 1, 3, 5 and 10 training episodes are shown in Figure 5.2. As it is shown, the performance is quite poor when the learner uses 1 training episode (mean value 52.7%). The reason for this poor performance is due to the small number of experience vectors, which means that not all the perceptual states have been encountered. Furthermore, some of these experience vectors can be noisy. As a result of the small number of experience and the noise, the input state vector is often matched to wrong experience vectors. As the training episodes are increased to 3, we can see that there is a significant increase in the score (mean value 66.75%). The learner has encountered more perceptual states and the correct experience vectors are dominating over the noisy ones. Hence, the matched experience vector is now closer to the input state vector. Although the score increases as the number of the training episodes increases, it does not increase significantly as for 1 and 3 training episodes. This implies that the experience of 3 training episodes were enough to give a reasonable performance. Any further training will improve the performance for the price of the time spent for these extra training episodes.

This score, though, gives primary information about the percentage of the correct behaviours the learner executed, and not if the task was complete, i.e. the learner carried food objects to its home. The penalty applied gives some indication about this, but not in a very clear manner. The histogram in Figure 5.3 shows how many foods were carried home for each number of training episodes. Bear in mind that the
Figure 5.2: Average scores of the recall phases in the training–corridor arena, with respect to the number of training episodes.

maximum number of food objects that could be brought home is 20. As it is shown, for 1 training episode only 8 food objects were collected. As the number of training episodes increases to 3, the number of foods that were collected increases to 11. Any further increase in the training episodes does not result in significant changes, 12 for 5 and 7 for 10, which is doubled to 14 in the histogram since the number of recall episodes was half than the others. This also implies that after a point, more training episodes do not benefit the learner as much as the first few ones.

Furthermore, from observations of the experimental runs we noticed that the learner most of the times is able to carry the food to its home, when this is lying in Area 2, as shown in Figure 5.4 (note that the initial position of the robot is random). The few
times that it failed was because it did not execute the ‘orient with the food’ behaviour. It appears that the experience vectors of this particular behaviour are quite close to others, in terms of their Euclidean distances used in the classification. When the ‘orient with food’ behaviour was successfully executed, all the consequent behaviours are also correctly executed. Hence, this closeness can be interpreted as difficulty on learning the correlation of this particular behaviour.

Another observation made was that the learner failed to collect the food most of the times when this was lying in Area 1. In addition, it was executing the ‘orient with food’ behaviour in Area 2 (even if there was no food there). This means that the learner had not generalise enough to this kind of situations. This could probably be solved by
Figure 5.4: When the food lies in Area 2, the robot collects it most of the times. However, this is rarely the case when the food lies in Area 1.

a different design of the prototype of the state vector. Furthermore, in the current state vector all the attributes have the same weight contribution in the calculation of the Euclidean distance. It seems from these observations that some of them are more significant than others. In particular, the distance of the robot from the food appears to be more significant than the distance of the robot from its home, something which our Euclidean distance does seem to encompass. However, experiments must be conducted in order to see if that is truly the case.

Lastly, note that the learner was trained 10 times in the same identical setup (Figure 5.1). The learner has not encountered an experience vector where the food lies in Area 1. A solution to this of course is to train the learner for this setup. We are not implying and do not believe that the learner must be trained explicitly in every perceptual experience. If that was the case we would miss the concept of generalising beyond the training. What we are saying is that the learner should probably have the experience
of what to do when the food is close to its home as well as when it is far away.

### 5.4.2 In a Rectangular Arena

In order to show that the learner can also cope with comparable performance in other arenas, we tested its performance in a similar experimental setup in a larger rectangular arena.

#### 5.4.2.1 Experimental Setup

This time we used two available food objects as the arena was bigger. However, due to the larger size and the two food objects, the time limit for foraging was 700 steps. This time limit might be a bit short, but it was preferred due to time restrictions. However, within this time limit the learner had enough chances to bring the food to its home. The penalties were also different. If the robot did not bring any food then a penalty of -20% was applied. However, if collected one of the two food objects, failing to bring the second back as well, a penalty of -5% was applied. This penalty was reduced due to the likely short time limit given to the learner. Also due to the time restrictions, we did not make as many experiments as we did for the corridor arena.

For these experiments, we had the learner go through 1 training episodes (3 experience files) and then we asked it to do perform foraging 4 times in the rectangular arena. We repeated this (with 1 training file) another two times, thus giving us 12 recall episodes. Then we did the same having the learner go through 5 training episodes. Finally we did 12 recall phases after having the learner go through 10 training episodes.

#### 5.4.2.2 Experimental Results

The experimental results for the rectangular arena are similar to the ones for the corridor arena. As it can be seen in Figure 5.5, the score is quite low when using 1 training episode (44.25%). The score improves significantly when using 5 training episodes (60.33%).Doubling the number of training episodes to 10 does not affect accordingly the increase of the score (67%). This means that the first few training episodes improve the performance of the learner quite significantly. After a point, though, any additional
training episodes do not pay back the time spent for the training, as much as the first ones do.

Furthermore, the histogram of Figure 5.6 shows how many food objects were collected in each case. As it can be seen, the number of food collected was a poor 7 out of 24 which was the possible maximum, when the learner had gone through 1 training episode. This improved again significantly to 14 collected food objects (out of 24), when 5 training episodes were used. Beyond this point, it appears that there is no improvement on how many food objects are collected. As it is shown for the 10 training episodes experience file, the number of collected food was 13 (out of 24).

Furthermore, the same observations, as the ones in the corridor arena, are also
Figure 5.6: Number of collected food objects (max 24) in the rectangular arena, with respect to the number of training episodes.

noticed on the experimental runs in the rectangular arena. There are areas in the rectangular arena similar to Area 2 (Figure 5.4) of the corridor arena, that when food lies there, the learner performs particularly well. On the other hand there are also areas, similar to Area 1 of the corridor arena, that the robot performs rather poorly. In addition, the phenomenon of executing the ‘orient with food’ behaviour when being in an area where there are no food objects was also observed (similar to Area 2 of the corridor arena). The reasons for these observed behaviours are the same to the ones explained for the corridor arena (see section 5.4.1.2).
5.5 Discussion

Firstly, the question of how to evaluate an imitation learning system is asked. Giving a definite answer (like useful or useless) is not applicable in these systems. There may be a lot of criteria to determine the value of imitation learning, but they are not easily designed or measured.

From the experiments we did we showed that the learner improves as more experience is gained. However, there is the question of up to which point the experience gained makes a significant difference. In our experiments, it seems that the learner improves in the first training episodes quite significantly. It still keeps on improving as more experience is gained, but with a smaller rate. The cost for more training is, primary, the time spend to train the learner.

Furthermore, from the observation of our experiments, we showed that not all the attributes of the state vector have the same significance in the classification. This means that either the more significant attributes should be weighted more; or a redefine of the state vector prototype is necessary. In addition, there is also the question of the ‘spatial variety’ of the training episodes. By no means, we do not wish to lead the learner through every possible perceptual experience. What we are implying is that, the learner, maybe, should also be exposed to some other setups.

The experiments on the rectangular arena showed evidence of generalisation in the learning. The learner was able to cope in an arena of a different size. However, the same observations with the training arena were also noticed. Hence, if these problems were solved for the training arena, they should also be solved for any other arena as well. However, experiments need to be done in order to see if that is the case. If this hypothesis is true, then this means that the learning could take place in an arena where ‘training facilitation’ are provided for helping the learning process, while the learner is able to negotiate any kind of environment.

Lastly, although we did not have time to train the learner in the ‘push’ and the ‘avoid food’ behaviours, we believe that the same conclusions would have been drawn. However, these experiments should be done in order to verify our belief.
5.6 Summary

In this chapter, we firstly presented how imitation learning systems could be evaluated. The experimental design and setup for our experiments were then described. Our experimental results and the discussion follow last. In the next chapter the achievements of the project are described. A description of the conclusions of this project and suggestions for further work conclude the next chapter.
Chapter 6

Conclusions

In this chapter the achievements of this project are described, followed by the conclusions derived and suggestions for further research work.

6.1 Achievements

In section 1.4 the objectives of this project have been defined. We believe that we achieved all of them.

Firstly, we wanted to study imitation learning in behaviours that involved interaction with objects, in a high level of abstraction. This high level of abstraction, was the program level imitation learning, where a learner robot imitates the demonstrated behaviours of a teacher and associates them with its perceptual experiences. We designed a computational framework of program level imitation learning, and we believe we came to some interesting conclusions.

Furthermore, through this project a platform was built, where more experiments and further research work can be conducted.

Lastly, the Khepera simulator was successfully extended to support multiple real Khepera robots.
6.2 Conclusions and Further Work

In this dissertation, a program level imitation learning framework was presented. In program level imitation learning, the learner learns to associate its perceptual experiences with their corresponding behaviours. The behaviours consist of basic motor actions. This implies that each behaviour could also be learned by imitation (action level imitation learning). The conjunction of the program level system and the action level system would result in a complete imitation learning system. The action level system consists part of further work.

During the design of the framework we have also discussed that imitation learning should take advantage of the social facilitation provided by the environment. One of these facilitation is communication among the members of the community.

For our experiments we used a foraging scenario, where a robot searches for food objects, collects the food and brings it to its home, either by carrying it or by pushing it. There is a case that a food is poisoned, in which the robot avoids it. In our experiments we trained a learner robot to forage by picking up the objects and carrying them home. From the experimental results we showed that the learner improves as more training episodes are acquired. However, the rate of improvement drops beyond a number of training episodes. We also showed that the learner is able to generalise beyond the scope of its training arena, by experiments in a different arena. This means that we could design training arenas that provide facilitation in order to make the training easier.

From the observations of our experiments, we concluded that there might a need to redefine some of the design issues of the recall phase. From the same observations, we also believe that the variety of the training episodes is as important as their quantity. However, these conclusion were drawn from observations rather than from experimental results, which is another extension to this project.

Due to time constraints we did not have time to train the learner for the other behaviours that involve object interaction (push and avoid). We believe though that the same conclusions apply to these behaviours as well. However, this must be proved with experimental results.
6.3 Epilogue

Our primary motivation was driven by the fact that mere copy of a program from one robot to another usually requires quite a bit of time and effort from the programmers. This is the case because even two robots that are supposed to be identical, will have differences in their configurations that affect their behaviour.

This problem can be avoided by learning the actions or the behaviours by imitating a demonstrator agent. The learner robot ignores the perceptions and capabilities of the demonstrator. It learns what to do or when it should do by using its own perceptions and capabilities, i.e. it learns using its own terms. Imitation learning then could lead into a new form of programming robots. A human demonstrator, who may not have any programming knowledge, could demonstrate a task to a robot which robot after learning the task by imitating the human teacher, could teach another robot in the same way.
Appendix A

Experimental Results
<table>
<thead>
<tr>
<th>File ID</th>
<th>Correct Behaviours</th>
<th>Penalty</th>
<th>Score</th>
<th>Average Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>g1</td>
<td>195/249 (78%)</td>
<td>-</td>
<td>78</td>
<td></td>
</tr>
<tr>
<td>g1</td>
<td>196/249 (78%)</td>
<td>-20</td>
<td>50</td>
<td></td>
</tr>
<tr>
<td>g1</td>
<td>197/249 (78%)</td>
<td>-</td>
<td>90</td>
<td></td>
</tr>
<tr>
<td>g1</td>
<td>198/249 (78%)</td>
<td>-20</td>
<td>45</td>
<td></td>
</tr>
<tr>
<td>g3</td>
<td>100/123 (81%)</td>
<td>-</td>
<td>81</td>
<td></td>
</tr>
<tr>
<td>g3</td>
<td>159/490 (32%)</td>
<td>-</td>
<td>32</td>
<td></td>
</tr>
<tr>
<td>g3</td>
<td>160/490 (32%)</td>
<td>-20</td>
<td>27</td>
<td></td>
</tr>
<tr>
<td>g3</td>
<td>161/490 (32%)</td>
<td>-</td>
<td>82</td>
<td></td>
</tr>
<tr>
<td>g6</td>
<td>415/500 (83%)</td>
<td>-20</td>
<td>63</td>
<td></td>
</tr>
<tr>
<td>g6</td>
<td>280/500 (56%)</td>
<td>-20</td>
<td>36</td>
<td></td>
</tr>
<tr>
<td>g6</td>
<td>194/500 (38%)</td>
<td>-20</td>
<td>18</td>
<td></td>
</tr>
<tr>
<td>g6</td>
<td>170/274 (62%)</td>
<td>-</td>
<td>62</td>
<td></td>
</tr>
<tr>
<td>g7</td>
<td>71/93 (76%)</td>
<td>-</td>
<td>76</td>
<td></td>
</tr>
<tr>
<td>g7</td>
<td>303/500 (60%)</td>
<td>-20</td>
<td>40</td>
<td></td>
</tr>
<tr>
<td>g7</td>
<td>290/500 (58%)</td>
<td>-20</td>
<td>38</td>
<td></td>
</tr>
<tr>
<td>g7</td>
<td>124/218 (56%)</td>
<td>-</td>
<td>56</td>
<td></td>
</tr>
<tr>
<td>g8</td>
<td>290/500 (58%)</td>
<td>-20</td>
<td>38</td>
<td></td>
</tr>
<tr>
<td>g8</td>
<td>379/500 (75%)</td>
<td>-20</td>
<td>55</td>
<td></td>
</tr>
<tr>
<td>g8</td>
<td>346/500 (69%)</td>
<td>-20</td>
<td>49</td>
<td></td>
</tr>
<tr>
<td>g8</td>
<td>290/500 (58%)</td>
<td>-20</td>
<td>38</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Total Average 52.7</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Food Collected 08/20</td>
</tr>
</tbody>
</table>

Table A.1: Experimental results for the recall phases in the training–corridor arena having the learner go through 1 training episode. Number of food=1, time limit=500 steps, penalty=-20%
<table>
<thead>
<tr>
<th>File ID</th>
<th>Correct Behaviours</th>
<th>Penalty</th>
<th>Score (%)</th>
<th>Average</th>
</tr>
</thead>
<tbody>
<tr>
<td>g123</td>
<td>382/477 (80%)</td>
<td>-</td>
<td>80</td>
<td>63.5</td>
</tr>
<tr>
<td>g123</td>
<td>393/500 (78%)</td>
<td>-20</td>
<td>58</td>
<td>63.5</td>
</tr>
<tr>
<td>g123</td>
<td>366/500 (73%)</td>
<td>-20</td>
<td>53</td>
<td>63.5</td>
</tr>
<tr>
<td>g123</td>
<td>416/500 (83%)</td>
<td>-20</td>
<td>63</td>
<td>63.5</td>
</tr>
<tr>
<td>g135</td>
<td>235/282 (83%)</td>
<td>-</td>
<td>83</td>
<td>64.5</td>
</tr>
<tr>
<td>g135</td>
<td>411/500 (82%)</td>
<td>-20</td>
<td>62</td>
<td>64.5</td>
</tr>
<tr>
<td>g135</td>
<td>293/500 (58%)</td>
<td>-20</td>
<td>38</td>
<td>64.5</td>
</tr>
<tr>
<td>g135</td>
<td>146/194 (75%)</td>
<td>-</td>
<td>75</td>
<td>64.5</td>
</tr>
<tr>
<td>g134</td>
<td>306/472 (65%)</td>
<td>-</td>
<td>65</td>
<td>64.5</td>
</tr>
<tr>
<td>g134</td>
<td>365/415 (87%)</td>
<td>-</td>
<td>87</td>
<td>64.5</td>
</tr>
<tr>
<td>g134</td>
<td>287/500 (57%)</td>
<td>-20</td>
<td>37</td>
<td>64.5</td>
</tr>
<tr>
<td>g134</td>
<td>56/72 (77%)</td>
<td>-</td>
<td>77</td>
<td>64.5</td>
</tr>
<tr>
<td>g158</td>
<td>91/100 (91%)</td>
<td>-</td>
<td>91</td>
<td>66.5</td>
</tr>
<tr>
<td>g158</td>
<td>273/409 (66%)</td>
<td>-</td>
<td>66</td>
<td>66.5</td>
</tr>
<tr>
<td>g158</td>
<td>441/500 (88%)</td>
<td>-20</td>
<td>68</td>
<td>66.5</td>
</tr>
<tr>
<td>g158</td>
<td>136/170 (80%)</td>
<td>-</td>
<td>80</td>
<td>66.5</td>
</tr>
<tr>
<td>g369</td>
<td>132/154 (85%)</td>
<td>-</td>
<td>85</td>
<td>76.25</td>
</tr>
<tr>
<td>g369</td>
<td>61/65 (93%)</td>
<td>-</td>
<td>93</td>
<td>76.25</td>
</tr>
<tr>
<td>g369</td>
<td>280/500 (56%)</td>
<td>-20</td>
<td>36</td>
<td>76.25</td>
</tr>
<tr>
<td>g369</td>
<td>371/500 (74%)</td>
<td>-20</td>
<td>54</td>
<td>76.25</td>
</tr>
<tr>
<td></td>
<td>Total Average</td>
<td></td>
<td>66.75</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Food Collected</td>
<td></td>
<td>11/20</td>
<td></td>
</tr>
</tbody>
</table>

Table A.2: Experimental results for the recall phases in the training–corridor arena having the learner go through 3 training episode. Number of food=1, time limit=500 steps, penalty=-20%
<table>
<thead>
<tr>
<th>File ID</th>
<th>Correct Behaviours</th>
<th>Penalty</th>
<th>Score (%)</th>
<th>Average Score (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>g12345</td>
<td>57/81 (70%)</td>
<td>-</td>
<td>70</td>
<td></td>
</tr>
<tr>
<td>g12345</td>
<td>122/45 (84%)</td>
<td>-</td>
<td>84</td>
<td></td>
</tr>
<tr>
<td>g12345</td>
<td>370/500 (74%)</td>
<td>-20</td>
<td>54</td>
<td></td>
</tr>
<tr>
<td>g12345</td>
<td>350/500 (70%)</td>
<td>-20</td>
<td>50</td>
<td></td>
</tr>
<tr>
<td>g678910</td>
<td>74/80 (92%)</td>
<td>-</td>
<td>92</td>
<td>64.5</td>
</tr>
<tr>
<td>g678910</td>
<td>349/407 (85%)</td>
<td>-</td>
<td>85</td>
<td></td>
</tr>
<tr>
<td>g678910</td>
<td>90/93 (96%)</td>
<td>-</td>
<td>96</td>
<td></td>
</tr>
<tr>
<td>g678910</td>
<td>415/500 (83%)</td>
<td>-20</td>
<td>63</td>
<td></td>
</tr>
<tr>
<td>g13579</td>
<td>116/146 (79%)</td>
<td>-</td>
<td>79</td>
<td>84</td>
</tr>
<tr>
<td>g13579</td>
<td>176/185 (95%)</td>
<td>-</td>
<td>95</td>
<td></td>
</tr>
<tr>
<td>g13579</td>
<td>393/500 (78%)</td>
<td>-20</td>
<td>58</td>
<td></td>
</tr>
<tr>
<td>g13579</td>
<td>360/500 (72%)</td>
<td>-20</td>
<td>52</td>
<td></td>
</tr>
<tr>
<td>g246810</td>
<td>356/469 (75%)</td>
<td>-</td>
<td>75</td>
<td>71</td>
</tr>
<tr>
<td>g246810</td>
<td>392/500 (78%)</td>
<td>-20</td>
<td>50</td>
<td></td>
</tr>
<tr>
<td>g246810</td>
<td>125/162 (77%)</td>
<td>-</td>
<td>77</td>
<td></td>
</tr>
<tr>
<td>g246810</td>
<td>271/500 (54%)</td>
<td>-20</td>
<td>34</td>
<td></td>
</tr>
<tr>
<td>g346810</td>
<td>97/106 (91%)</td>
<td>-</td>
<td>91</td>
<td>59</td>
</tr>
<tr>
<td>g346810</td>
<td>109/186 (58%)</td>
<td>-</td>
<td>58</td>
<td></td>
</tr>
<tr>
<td>g346810</td>
<td>315/500 (63%)</td>
<td>-20</td>
<td>43</td>
<td></td>
</tr>
<tr>
<td>g346810</td>
<td>206/352 (58%)</td>
<td>-</td>
<td>58</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>62.5</td>
</tr>
</tbody>
</table>

Table A.3: Experimental results for the recall phases in the training–corridor arena having the learner go through 5 training episode. Number of food=1, time limit=500 steps, penalty=-20%
<table>
<thead>
<tr>
<th>File ID</th>
<th>Correct Behaviours /Total (%)</th>
<th>Penalty</th>
<th>Score (%)</th>
<th>Average Score (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>g12345678910</td>
<td>107/123 (86%)</td>
<td>-</td>
<td>86</td>
<td></td>
</tr>
<tr>
<td>g12345678910</td>
<td>239/283 (84%)</td>
<td>-</td>
<td>84</td>
<td></td>
</tr>
<tr>
<td>g12345678910</td>
<td>309/354 (87%)</td>
<td>-</td>
<td>87</td>
<td></td>
</tr>
<tr>
<td>g12345678910</td>
<td>131/141 (92%)</td>
<td>-</td>
<td>92</td>
<td></td>
</tr>
<tr>
<td>g12345678910</td>
<td>65/74 (87%)</td>
<td>-</td>
<td>87</td>
<td></td>
</tr>
<tr>
<td>g12345678910</td>
<td>332/421 (78%)</td>
<td>-</td>
<td>78</td>
<td></td>
</tr>
<tr>
<td>g12345678910</td>
<td>300/500 (60%)</td>
<td>-20</td>
<td>40</td>
<td></td>
</tr>
<tr>
<td>g12345678910</td>
<td>383/500 (76%)</td>
<td>-20</td>
<td>56</td>
<td></td>
</tr>
<tr>
<td>g12345678910</td>
<td>248/500 (75%)</td>
<td>-</td>
<td>75</td>
<td></td>
</tr>
<tr>
<td>g12345678910</td>
<td>365/500 (73%)</td>
<td>-20</td>
<td>53</td>
<td></td>
</tr>
</tbody>
</table>

Table A.4: Experimental results for the recall phases in the training–corridor arena having the learner go through 10 training episode. Number of food=1, time limit=500 steps, penalty=-20%
<table>
<thead>
<tr>
<th>File ID</th>
<th>Correct Behaviours</th>
<th>Penalty</th>
<th>Score (%)</th>
<th>Average</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>/Total (%)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>g1</td>
<td>286/334 (85%)</td>
<td>85</td>
<td></td>
<td></td>
</tr>
<tr>
<td>g1</td>
<td>372/700 (53%)</td>
<td>-20</td>
<td>33</td>
<td></td>
</tr>
<tr>
<td>g1</td>
<td>477/700 (86%)</td>
<td>-20</td>
<td>48</td>
<td></td>
</tr>
<tr>
<td>g1</td>
<td>364/700 (52%)</td>
<td>-20</td>
<td>32</td>
<td>49.5</td>
</tr>
<tr>
<td>g7</td>
<td>340/700 (48%)</td>
<td>-20</td>
<td>28</td>
<td></td>
</tr>
<tr>
<td>g7</td>
<td>378/687 (54%)</td>
<td></td>
<td>54</td>
<td></td>
</tr>
<tr>
<td>g7</td>
<td>321/700 (45%)</td>
<td>-20</td>
<td>25</td>
<td></td>
</tr>
<tr>
<td>g7</td>
<td>366/700 (52%)</td>
<td>-20</td>
<td>32</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>34.75</td>
<td></td>
</tr>
<tr>
<td>g8</td>
<td>460/698 (66%)</td>
<td></td>
<td>66</td>
<td></td>
</tr>
<tr>
<td>g8</td>
<td>539/700 (77%)</td>
<td>-5</td>
<td>72</td>
<td></td>
</tr>
<tr>
<td>g8</td>
<td>278/700 (39%)</td>
<td>-20</td>
<td>19</td>
<td></td>
</tr>
<tr>
<td>g8</td>
<td>400/700 (57%)</td>
<td>-20</td>
<td>37</td>
<td>48.5</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Total Average</td>
<td></td>
<td>44.5</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Food Collected</td>
<td></td>
<td>07/24</td>
<td></td>
</tr>
</tbody>
</table>

Table A.5: Experimental results for the recall phases in the rectangular arena having the learner go through 1 training episode. Number of food=2, time limit=700 steps, penalty=-20% for leaving all, food in the arena, -5% for leaving one
<table>
<thead>
<tr>
<th>File ID</th>
<th>Correct Behaviours</th>
<th>Penalty</th>
<th>Score (%)</th>
<th>Average</th>
</tr>
</thead>
<tbody>
<tr>
<td>g12345</td>
<td>271/398 (68%)</td>
<td>-</td>
<td>68</td>
<td>65.5</td>
</tr>
<tr>
<td>g12345</td>
<td>408/623 (65%)</td>
<td>-</td>
<td>65</td>
<td></td>
</tr>
<tr>
<td>g12345</td>
<td>506/700 (72%)</td>
<td>-20</td>
<td>52</td>
<td></td>
</tr>
<tr>
<td>g12345</td>
<td>464/598 (77%)</td>
<td>-</td>
<td>77</td>
<td></td>
</tr>
<tr>
<td>g678910</td>
<td>455/700 (65%)</td>
<td>-5</td>
<td>60</td>
<td></td>
</tr>
<tr>
<td>g678910</td>
<td>433/700 (61%)</td>
<td>-20</td>
<td>41</td>
<td></td>
</tr>
<tr>
<td>g678910</td>
<td>247/316 (78%)</td>
<td>-</td>
<td>78</td>
<td></td>
</tr>
<tr>
<td>g678910</td>
<td>504/700 (72%)</td>
<td>-5</td>
<td>67</td>
<td></td>
</tr>
<tr>
<td>g246810</td>
<td>403/700 (57%)</td>
<td>-5</td>
<td>52</td>
<td>61.5</td>
</tr>
<tr>
<td>g246810</td>
<td>393/700 (56%)</td>
<td>-20</td>
<td>36</td>
<td></td>
</tr>
<tr>
<td>g246810</td>
<td>204/293 (69%)</td>
<td>-</td>
<td>69</td>
<td></td>
</tr>
<tr>
<td>g246810</td>
<td>448/700 (64%)</td>
<td>-5</td>
<td>59</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>54</td>
</tr>
</tbody>
</table>

Table A.6: Experimental results for the recall phases in the rectangular arena having the learner go through 5 training episode. Number of food=2, time limit=700 steps, penalty=-20% for leaving all, food in the arena, -5% for leaving one.
### Table A.7: Experimental results for the recall phases in the rectangular arena having the learner go through 10 training episode. Number of food=2, time limit=700 steps, penalty=-20% for leaving all, food in the arena, -5% for leaving one

<table>
<thead>
<tr>
<th>File ID</th>
<th>Correct Behaviours /Total (%)</th>
<th>Penalty</th>
<th>Score (%)</th>
<th>Average Score (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>g12345678910</td>
<td>560/700 (80%)</td>
<td>-5%</td>
<td>75</td>
<td></td>
</tr>
<tr>
<td>g12345678910</td>
<td>354/495 (71%)</td>
<td>-</td>
<td>71</td>
<td></td>
</tr>
<tr>
<td>g12345678910</td>
<td>270/373 (72%)</td>
<td>-</td>
<td>72</td>
<td></td>
</tr>
<tr>
<td>g12345678910</td>
<td>335/397 (84%)</td>
<td></td>
<td>84</td>
<td></td>
</tr>
<tr>
<td>g12345678910</td>
<td>550/700 (78%)</td>
<td>-5%</td>
<td>73</td>
<td></td>
</tr>
<tr>
<td>g12345678910</td>
<td>535/700 (75%)</td>
<td>-5%</td>
<td>71</td>
<td></td>
</tr>
<tr>
<td>g12345678910</td>
<td>503/700 (54%)</td>
<td>-20%</td>
<td>51</td>
<td></td>
</tr>
<tr>
<td>g12345678910</td>
<td>454/700 (64%)</td>
<td>-20%</td>
<td>44</td>
<td></td>
</tr>
<tr>
<td>g12345678910</td>
<td>557/700 (79%)</td>
<td>-5%</td>
<td>72</td>
<td></td>
</tr>
<tr>
<td>g12345678910</td>
<td>457/700 (65%)</td>
<td>-20%</td>
<td>45</td>
<td></td>
</tr>
<tr>
<td>g12345678910</td>
<td>144/205 (70%)</td>
<td>-</td>
<td>70</td>
<td></td>
</tr>
<tr>
<td>g12345678910</td>
<td>580/700 (82%)</td>
<td>-5%</td>
<td>77</td>
<td></td>
</tr>
</tbody>
</table>

|                           | Total Average | 67     |           |                   |
|                           | Food Collected| 13/24  |           |                   |
Bibliography


