Chapter 3

Application Description

3.1 Introduction

This chapter describes the anatomy of the robot (or *robot shell*) that will be used throughout the work. This chapter also gives a description of the initial environment in which the robot will work and provides an informal statement of the requirements on the target behaviour. Finally, this chapter provides description of the Khepera/Matlab interface. Although this last item is not a specific requirement of this stage, it was deemed appropriate to discuss it in detail in this chapter since it provides the primary means of interacting with the *robot shell*.

3.2 Description of the Robot Shell

The hardware shell that will be used in this work is the Khepera miniature robot (Figure 3-1). The basic configuration consists of a sensory/motor board and a CPU board. These two boards combined are 55mm in diameter and 30mm high. The CPU board is based on the MC68331 microcontroller and has 256 Kbytes of user RAM and 256-512 Kbytes of ROM. The CPU board has all the hardware required to interface with a computer through an asynchronous serial port. The basic sensory/motor board includes two DC motors, two incremental sensors, eight analog infra-red (IR) proximity/ambient light sensors, and an on-board power supply. See or the *Khepera User's Manual* or [Mondada93] for a complete technical description of the robot

and its capabilities.

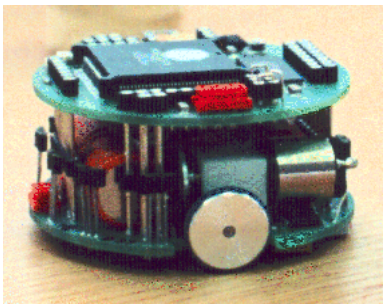

Figure 3-1
A View of Khepera[1]

The Khepera base unit can accommodate additional turrets for the modular application of functionality. For example, there is a **General I/O** module which makes simple I/O extensions possible. A **Gripper** turret gives the Khepera an object manipulation capability. Other extensions currently available include the **K213 Vision** turret, the **K2D Video** turret, the **Radio Modem** turret and the **IR Communications** turret.

None of the additional turrets were used during the course of the present work although they may be considered for future work. As a result, the only remote sensing possible will be through the eight ambient light/IR range sensors equipped on the sensory/motor board. These sensors constitute the robot's entire sensory interface and are, in general, not receptive to

[1]Figures 3-1 and 3-3 courtesy of ftp://lamiftp.epfl.ch/khepera

modification. Although the ability to modify the sensorimotor interface is an assumption of the BAT methodology, it is not an issue here since the effectiveness of the Khepera in its basic configuration is being assessed as part of this work.

A functional block diagram of the Khepera is shown in Figure 3-2. Locomotion is provided by two motors placed on the outer perimiter of the base unit. Sensors are provided in the locations shown. The angle in the figure refers to the left or right deviation from the vertical of the position of each sensor. Note that each sensor provides *dual functionality*. In one mode, they provide proximity data by transmitting in the infra-red band and measuring the reflected return. In another mode, they can be used to determine the level of ambient light present.

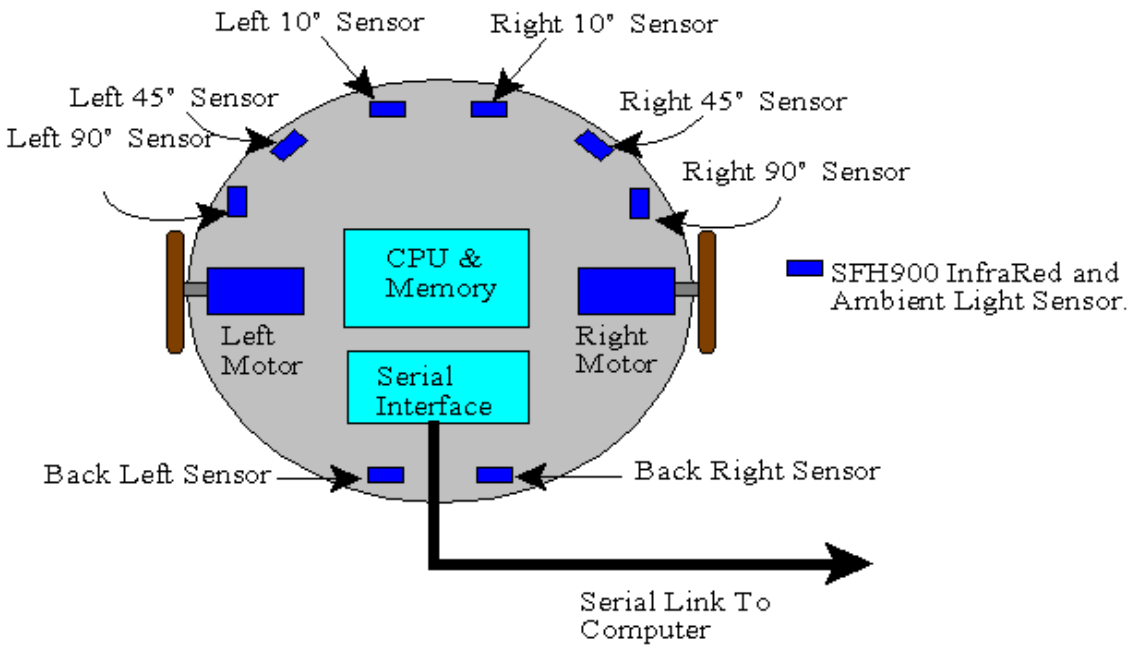

Figure 3-2
Block Diagram of Khepera Robot

3.3 Description of the Initial Environment

The initial environment consisted of a small arena (or *playpen*) whose maximum dimensions were 55cm by 60 cm with 4 cm high aluminum covered walls around the outer perimeter. The playpen was easily transportable and had to be used in rooms that had limited or controlled light sources.

The playpen walls were lined with coloured Lego blocks connected in a random fashion. The playpen itself will contained obstacles and walls in various configurations also constructed from randomly coloured Lego blocks. The random colours of the Lego blocks provided a noisier environment since different colours absorb and reflect visible and infra-red light differently.

3.4 Requirements on the Target Behaviour

In this work, it was desired to have the robot turn towards and approach a light source placed at some location either inside the playpen or suspended at some point outside of it. In approaching the light, the robot was to avoid obstacles, walls, and deadlock positions.

3.5 The Programming Environment

3.5.1 Introduction

Given its extensibility, ease of use, and graphical presentation capabilities, Matlab was selected as the programming platform for the Khepera robot. For this work, a toolbox of functions and interfaces was developed for programming and operating the control algorithms of the Khepera from within the Matlab environment. This toolbox is known as the *Khepera*

Toolbox and is included as Attachment 1.

The interface between the Khepera and the Matlab environment is discussed in the following sections. The physical interface between the Khepera and a host PC is described first, followed by an examination of the core interface functions which allow communications between the robot and Matlab. The section is concluded with a description of additional functionality provided by the toolbox, which includes a Khepera robot simulator.

3.5.2 Robot/PC Interface

The Khepera can be operated in real time through a serial port on a PC when configured as shown in Figure 3-3. There are a fixed set of serial commands that can be issued through the serial port to make the robot move and for reading its sensors. This command set can be found in the *Khepera User's Manual*.

Connected in this manner, it is possible to make use of the host machine memory and programming resources to help develop, understand, and document autonomous control algorithms. Although a physical link exists, the robot is no less an autonomous device than a vehicle operating via a radio link.

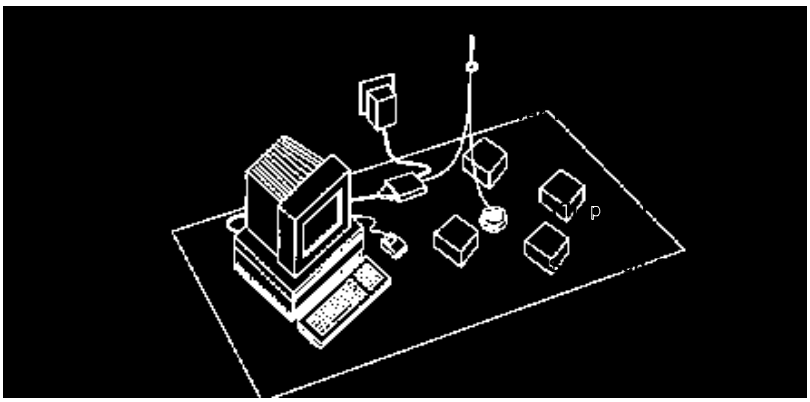

Figure 3-3
Typical Tethered Experimental Configuration

3.5.3 Khepera Toolbox Core Functionality

Communications between the Khepera robot and the Matlab environment is provided by thirteen Matlab MEX functions shown in Table 3-1. A MEX (or Matlab EXecutable) function is a Matlab-callable C or Fortran function. Each MEX function in Table 3-1 implements an equivalent Khepera serial command from the within the Matlab environment and therefore provide the core functionality of the Matlab-Khepera Toolbox. It is thus possible to create, test, and document real time control algorithms using all of the programming and graphics tools provided by the Matlab Environment.

Khepera Serial Commands Implemented as MEX Functions		
Khepera Serial Command	Command Definition	MEX Equivalent
A	**Con**figure PID parameters of speed regulator	con
C	**s**et **k**hepera **p**osition	skp
D	**S**et **m**otor **s**peed	sms
E	**R**ead **m**otor **s**peed	rms
F	**C**onfigure **PID** parameters of position regulator	cpid
G	**S**et **p**osition **c**ounter	spc
H	**R**ead **p**osition **c**ounter	rpc

Khepera Serial Commands Implemented as MEX Functions		
I	**R**ead **A/D** input	rad
J	**C**onfigure **s**peed **p**rofiler **c**ontroller	cspc
K	**R**ead **m**otion **c**ontroller	rmc
L	**C**hange **L**ED **s**tatus	cls
N	**R**ead **p**roximity **s**ensors	rps
O	**R**ead **l**ight **s**ensors	rls

Table 3-1
Khepera Serial Commands Executable from within Matlab

Each MEX function in the table passes the required serial command character (and any required arguments) through a serial port to the Khepera. Return data is passed back into the Matlab environment by the MEX function as well. The functions in Table 3-1 were written in C and the code for each is provided in Annex A of Attachment 2, the *Khepera Toolbox for Matlab*. A detailed description of the serial communications protocol (which uses BIOS INT14 routines instead of byte read/write routines) is also provided in that annex.

3.5.4 Additional Functionality and Graphical User Interfaces's

The toolbox provides more than just access to the Khepera serial commands from within the Matlab environment. A number of Graphical User Interfaces (GUI's) have been developed for algorithm automation, sensor data collection, and data presentation. A complete description of the application of each GUI and associated miscellaneous commands is given in Attachment 2,

the *Khepera Toolbox for Matlab*. The Matlab code for all of the additional functionality is provided as Annex B of this Attachment.

3.5.5 The Khepera Simulator

In addition to the GUI's developed for use with a real Khepera robot configured as in Figure 3-3, a separate simulation environment has been developed for the Khepera base unit. The BAT methodology calls attention to the problem of training a real robot: it can be a time and computationally expensive task. It suggests that simulations can be used to develop first approximations of the desired control architecture which can then be implemented on real robots. In several papers, ([Columbetti96], [Meeden93]), transplanting an architecture trained in a simulated environment into a real robot provides the target system with a level of 'initial knowledge' that allows for faster training and often superior performance. This improvement in training time and performance is usually purchased at very low cost in terms of time and resources since a simulator may run at many orders of magnitude faster than the actual robot and can be executed on many workstations at once.

Although simulations can only be an approximate model of the real world, the environment takes into account issues brought up by [Harvey 93] with regards to ensuring that a simulator stays in close step with reality. Additionally, utilities have been developed for calibrating the simulation sensor responses to those produced by a real Khepera in a given environment. Attachment 1 contains usage details of the simulation environment for the Khepera robot.