Building an Interactive Robotics Control Laboratory with Python

Bin Wang
School of Electrical and Computer Engineering
RMIT University, Melbourne, Australia
b.wang@rmit.edu.au

Andrew Jennings
School of Electrical and Computer Engineering
RMIT University, Melbourne, Australia
andrew.jennings@rmit.edu.au

***Abstract**: This paper details the development of an interactive robotics control laboratory at the RMIT University. A general-purpose programming language, Python is applied to Khepera II robots in the laboratory. The progress of students working in the robotics laboratory who take the course of Robotics Control at the RMIT University is presented. It is hoped that this paper will be of use to others developing robotics control laboratory.*

Introduction

There is always a big gap between the control theory and control applications. Many students take control theory courses, but few retain the details of the skills learned. The quest for a new way of teaching control theory concepts, or other concepts such as artificial intelligence, has led educators to explore methods involving hands-on and creative design work involving robotics (e.g. Bekey 2005).

Due to the high cost of buying and maintaining robots, the study of robotics has been limited to graduate level classes at universities. Recently, the advent of smaller less expensive robots has made it feasible for educators to integrate robotics into undergraduate curriculum (e.g. Harlanetal, Levine and McClarigan 2001, Bushnell and Crick 2003).

The subject of Computer Robotics Control in RMIT University is a course for fourth year and postgraduate students at the School of Electrical and Computer Engineering. The purpose of this subject is to let students know how to construct a robot control system. Students were assumed to have basic knowledge of robot system architectures, mathematical knowledge to analyse robot behaviour and analytical knowledge of simple control systems. This is a one semester, high-level robotics control subject. Most of the work needs to be done in the lab. The weighting of the lab assessment is 70\%. To get students a better understanding of what they've been taught in the lecture and the lab, a real robot, Khepera II, a product from K-TEAM SA, Switzerland is used (see Figure 1).

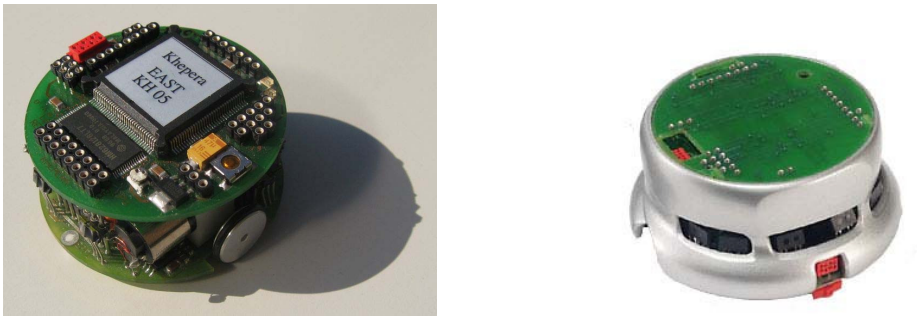

Figure 1: Khepera II robot

In this paper, a Python-written robot interface is developed for the Khepera II robot which makes it an excellent platform for introducing robotics to students. The interface simplifies the design and testing of control algorithms by hiding the low-level communication between the controlling program and the robot. Furthermore, it can be used to build derived classes that encapsulate related base behaviours so that high-level tasks can be implemented.

Overview of the Khepera II Robot

Khepera II robot has a circular shape, a diameter of 55 mm, a height of 30 nm, and a weight of 70g. It possesses on-board power supply and two motors which can be independently controlled by a PID controller. It is equipped with eight IR sensors and a Motorola 68331 25MHz 32 bit microcontroller. The robot can be controlled in two ways. The controlling program may be run on a host computer. Communications can be realized by sending and receiving data and commands through a serial line. Alternatively the controlling program is cross-compiled on the host computer and downloaded to the robot which in turn executes the program on board. The micro-controller has 512 KB of RAM and a ROM containing a small operating system named "BIOS" (Basic I/Os system). Considering the capacity of the battery can only support 1.5 hours, the serial mode control was chosen in the lab. In the lab room, eight workstations have been built up which are connected to robots through serial cables (Figure 2). The protocol between Khepera II and PC is RS232 which is a standard protocol inasynchronous serial communication (Figure 3).

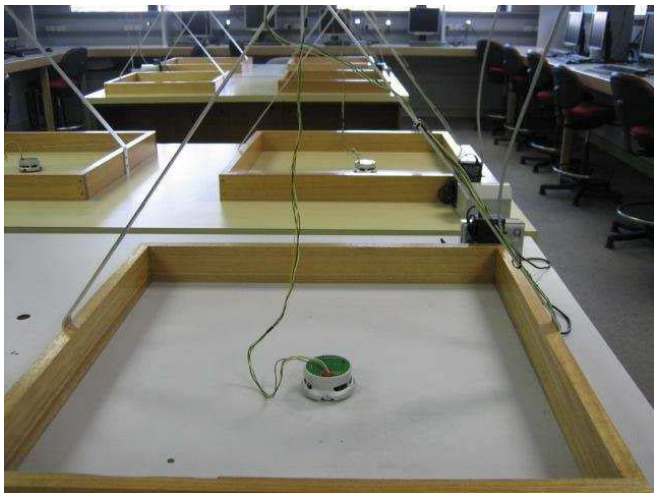

Figure 2: A picture of the lab room

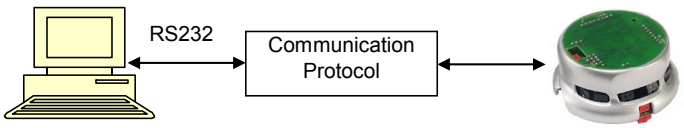

Figure 3: Serial communication between robot and PC

Motor controller is the core of mobile robots. In Khepera II robots, an incremental encoder is placed on the motor axis and gives 24 pulses per revolution of the motor. The Khepera II main processor has the direct control on the motor power supply and can read the pulses of the incremental encoder. An interrupt routine detects every pulse of the incremental encoder and updates a wheel position counter. To let robots move a required distance or turn with a specific angle, the 32 bit position counters of the two motors need to set as 0 before they move. The structure of the DC motor controller is shown in Figure 4.

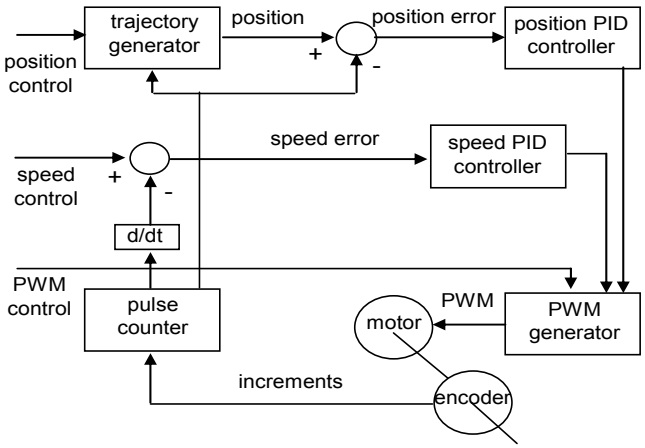

Figure 4: Structure of the DC motor controller

Development of the Robot Class

Each semester in RMIT University has only 12 weeks for teaching with only two hours lab time each week. Thus, it is necessary to find out a way to teach students how to apply what they have learned in the lecture to mobile robots efficiently and give students a quick start so that they could dip into this subject immediately. The Khepera II robot uses the SerCom protocol (K-Team SA 2002) to communicate with the host computer. The protocol is based on ASCII commands and responses. These commands are those embedded ``modules'' in the Khepera of sensors and motors. So the first step is to get students familiar with these commands which are used to set controls and request sensors value form the robot such as set the motor speeds, read proximity sensor values. A programming language needs to be picked to let students see how these commands work. Among those popular programming languages, Python is chosen because its interactive mode could save much time comparing with other languages such as C/C++ and Java. It allows students to focus less on details such as types, compilers, and writing boilerplate code and more on the algorithms and data structures relevant to performing their intended tasks (Miller and Ranum 2005). It is believed that the syntax of Python is so clear that writing in Python is so enjoyable for all of one's programming work (Lutz 2007). To check out every protocol of Khepera II robots, simply import the serial package and type the following lines:

```
import serial

ser=serial.Serial(port=0,stopbits=2,baudrate=9600)

ser.write("D,5,5\n")

ser.realine()
```

In this way, students could be familiar with these protocols within two hours with the simple but powerful language.

To give students a better support, an interface for the Khepera II robot that hides the low-level communication between the robot and the controlling program is preferred so that students could focus on developing behaviour control algorithms. To achieve this goal, a robot class was produced in Python which supports object-oriented programming (OOP). Inside the robot class, there are a few methods which include opening and closing the serial port connection to the host computer, moving the robot and monitoring the light and proximity sensors. Below is part of the code of the robot class.

```
import serial,math,string,time,msvcrt

#Set the initial speed of the motor

wheelSpeeds=[5,5]
```

```
class Myrobot:

    def __init__(self,speed = 9600,port=0,stopbits=2):

        self.speed = speed

        self.port = port

        self.stopbits = stopbits

#connect the robot

    def connect(self):

        #connects the serial port

        self.serial = serial.Serial(baudrate=self.speed,port=self.port,stopbits=self.stopbits)

        self.serial.open()

        print self.serial.isOpen()

        print self.serial

        print self.serial.parity

        self.serial.write("G,0,0\n")

        print self.serial.readline()

#disconnect the robot

    def disconnect(self):

        self.serial.close()

#Reset the position counter

    def reset(self):

        self.serial.write("G,0,0\n")

        print self.serial.readline()

#Read the position counter

    def readPos(self):

        self.serial.write("H\n")

        print self.serial.readline()

         … … …
```

Introducing the Pyrobot Simulator

Simulators allow users to test their control programs in the host computer. In this way, they can greatly reduce development time and cost. Simulators are especially preferred when using time-consuming algorithm for learning or evolution of intelligent controllers. After a thorough view of current robotics software platforms, I chose Pyrobot (Python Robotics) as the simulator for this subject. Pyrobot is an open source Python-based robot programming environment. It comes with a nice curriculum, so it can be used for classroom robotics. It has great potentials for applications development of Khepera robots. Current applications include computer vision, artificial intelligence, robot soccer, etc. It also supports the command line mode in its graphical user interface. Below is an example of running Pyrobot simulator (Figure 5, 6).

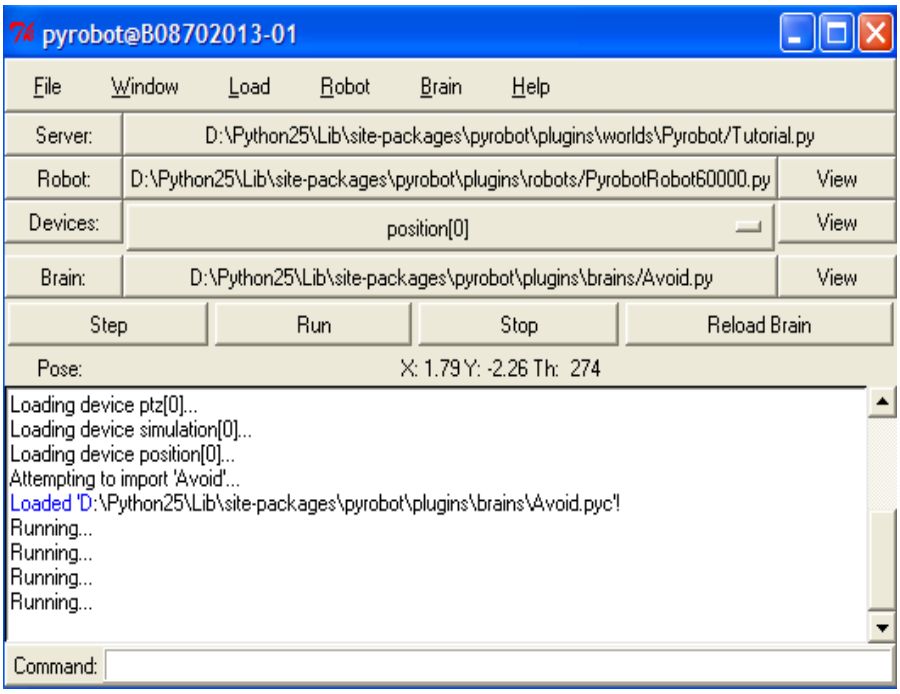

Figure 5: Pyrobot simulator control window

Using Pyrobot simulator has been strongly encouraged during the teaching semester. Students are getting comfortable to work on it. Another merit of this simulator is it is easy to modify the code from simulator to control the real Khpera II robot. Different environments are easy to build up in the simulator as well.

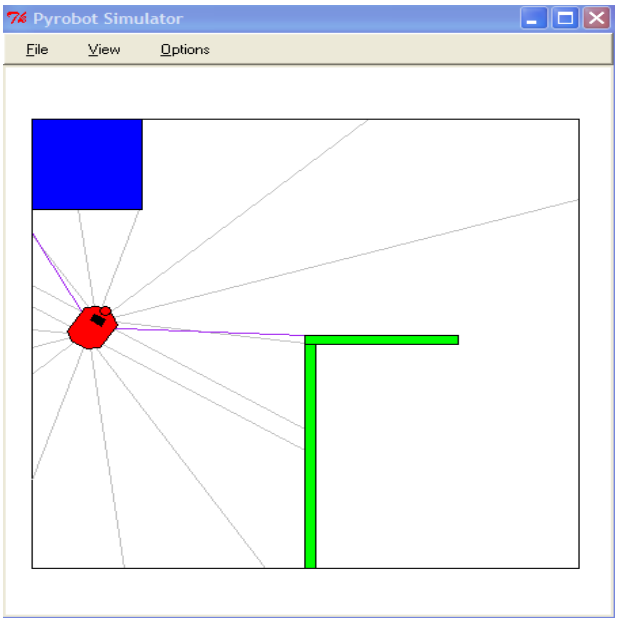

Figure 6: Pyrobot simulator running window

Assignments Design

The purpose of assignments is to make students be able to achieve the following targets at the completion of this course:

- Analyse different robot motion systems.
- Design localisation systems and measure their performance.
- Analyse communication systems for robot teams.

- Construct cooperation schemes for robot teams.
- Program a robot to perform sophisticated manoeuvres.

There are two main lab assignments in this subject, the first one is wall following and obstacle avoiding. Robots are required to find the wall first, once a wall has been found, it needs to follow it without either running into it or drifting too far away from it. The purpose of this one is get students familiar with how the sensors and motors work with Khepera II robots. The second one is a maze mapping task where a maze table is being used. The maze is designed to reflect a "rescue" scenario: first the robot has to find the target (a small infrared source). This is the first run through the maze. It then is placed at the start again and has to find the target again with the shortest route. Robots need to be able to map the maze and request the x-y position of the target. This task requires students to apply what they have learned about the localisation methods to robots. Throughout these two assignments, the "time" module in Python has been found useful. The "time.sleep()" method in the module allows robots complete one action before having another movement. In the second assignment, the white light source was initially used to make the target in the maze mapping task. However, after realising it could confuse the IR sensors on Khepera II robots, we switched to use a directional infrared emitter as the target and robots can find it successfully.

Discussion

The introduction of Khepera II robot to the subject Computer Robotics Control gives students a boost of learning robots at the School of Electrical and Computer Engineering, RMIT University. Python was chosen as the programming language for this subject because of its advantages over other languages as the teaching tool in the lab. Students felt satisfied with Python through the whole semester although many of them didn't know much of it before they took this subject. Without the tedious debugging and compilation, they found programming with Python quite enjoyable.

Besides, students got many ideas on how to build a real robot from Khepera II. After finishing this subject, they couldn't wait to get started to build up their own robots. The manuals for DC motors, sensors and BIOS are quite helpful. Many of them followed Khepera II to make the SerCom protocol for their robots so that their robots can also be controlled by Python. The control theory concepts such as PID control are also emphasized in this subject. Further developments on the laboratory will be focusing on multi-robot systems where Khepera II robots' extensions to turrets which have wireless communication function with other robots and PCs will be applied. The multi-robot system will benefit some other courses taught in RMIT University.

References

G. A. Bekey (2005). *Autonomous Robots: from Biological Inspiration to Implementation and Control*. The MIT Press.

R. M. Harlan, D. B. Levine and S. McClarigan (2001). The Khepera robot and the kRobot class: a platform for introducing robotics in the undergraduate curriculum. *ACM SIGCSE Bulletin, 33 (1), 105—109.*

L. G. Bushnell and A. P. Crick (2003). Control education via autonomous robotics. *42nd IEEE Conference on Decision and Control, Hawaii, USA.*

K-Team S. A. (2002). *Khepera II User Manual, version 1.1.* K-Team, Switzerland.

B. N. Miller and D. L. Ranum (2005). *Problem Solving with Algorithms and Data Structures Using Python.* Franklin Beedle & Associates.

M. Lutz (2007). *Learning Python, 3rd edition*. O'Reilly Media, Inc.