DESIGN AND DEVELOPMENT OF ROBOTIC CONTROL INTERFACE FOR IROBOT CREATE USING MATLAB

Aaron R. Rababaah
University of Maryland Eastern Shore, Princess Anne, Maryland, USA
arrababaah@umes.edu, bgjohnson@umes.edu

ABSTRACT

In this paper we will present the development of a software control interface for iRobot Create robotic platform, a system we call RISC (Robotic Interface Software for Create Robot). The purpose of the interface is to be able to control and command the robot using friendly GUI (Graphical User Interface) tools that interact with the robot to do different tasks including: initialization, setting parameters of the robot, reading states of the robot such as different sensors, mobility control of the robot, activating built-in demo behaviors, etc. RISC is based on ISIDE (Intelligent Systems Integrated Development Environment) which is built in MatLab software. RISC consists of number of components including: the GUI and its handlers, communication via USB (Universal Serial Bus) and the command protocol between the robot RISC. RISC is run on a computer that is connected to Create robot using USB cable. The user of RISC can issue commands using the GUI components to task the robot to move, turn, slowdown, speedup, forward, backward, stop, display sensor readings and more. RISC will be tested after it has been developed in a lab environment where the Create robot is designed to be used in an indoor environment.

1.0 INTRODUCTION

Robotics technology is rapidly growing and is recognized as one of the dominant technologies of the century [1, 2]. At the department of Math and Computer Science at UMES are interested to enrich our curricula with robotics and get our students good exposure to the fundamentals and advancements in this technology. We had some experience with software platforms to program robots via linux operating system and other third party development software [9]. Through these experiences we had some success but we felt that our students need friendlier development environment that allow them to easily interact with the robot. Based on our previous experience, we choose MatLab [10] as our development environment along with ISDIE [11] software as an intermediate development and testing environment.

Although we may find number of previous work in this area related to our work but never the less we are interested to develop the software from scratch just to give our students the academic and research exercise and experience taking on this interesting and engaging project. We have engaged undergraduate and graduate students in this project so we can generate senior design and master level work under the same environment.

2.0 SYSTEM CONCEPT AND DESIGN

In this section we will present and discus the different components of the software and hardware used to design and implement the RISC systems. As shown in Figure 1, the system architecture illustrates how the different components are integrated together and their interconnections and communications that are required for the proper operation of the system. The different components in Figure 1 are explained as follows:

2.1 GUI Controller Software

The graphical user interface (GUI) is the main way of interaction with the robot which includes number of components each has its own functionality, these are illustrated in Figure 2 and explained as follows:

- *Drop-down list*: includes commands to initialize and configure the software and the robot e.g. set communication link and configure sensor sampling delay/frequency.

- *Command Message Box*: used to support input from the drop-down list.
- *Mobility panel*: several buttons created to control robot mobility: forward, backward, turn left, turn right, stop, etc.
- *Speed control*: a slider that sets the robot speed from 0 – 2000mm/s.
- *Sensor readings*: checkboxes that periodically posts sensor readings of bumpers, wheel drop, cliff and wall sensors.
- *Message textbox*: for any customized message to the robot, this text box is used to communicate via the RS232 link.
- *Drop-down list01*: includes commands to initialize and configure the software and the robot e.g. set communication link and configure sensor sampling delay/frequency.
- *Drop-down list02*: includes commands to that instruct the robot to watch for a particular event and then to react according to what the user requested. These events include: bumper, wall sensor, cliff sensor, wheel drops, robot buttons, etc.

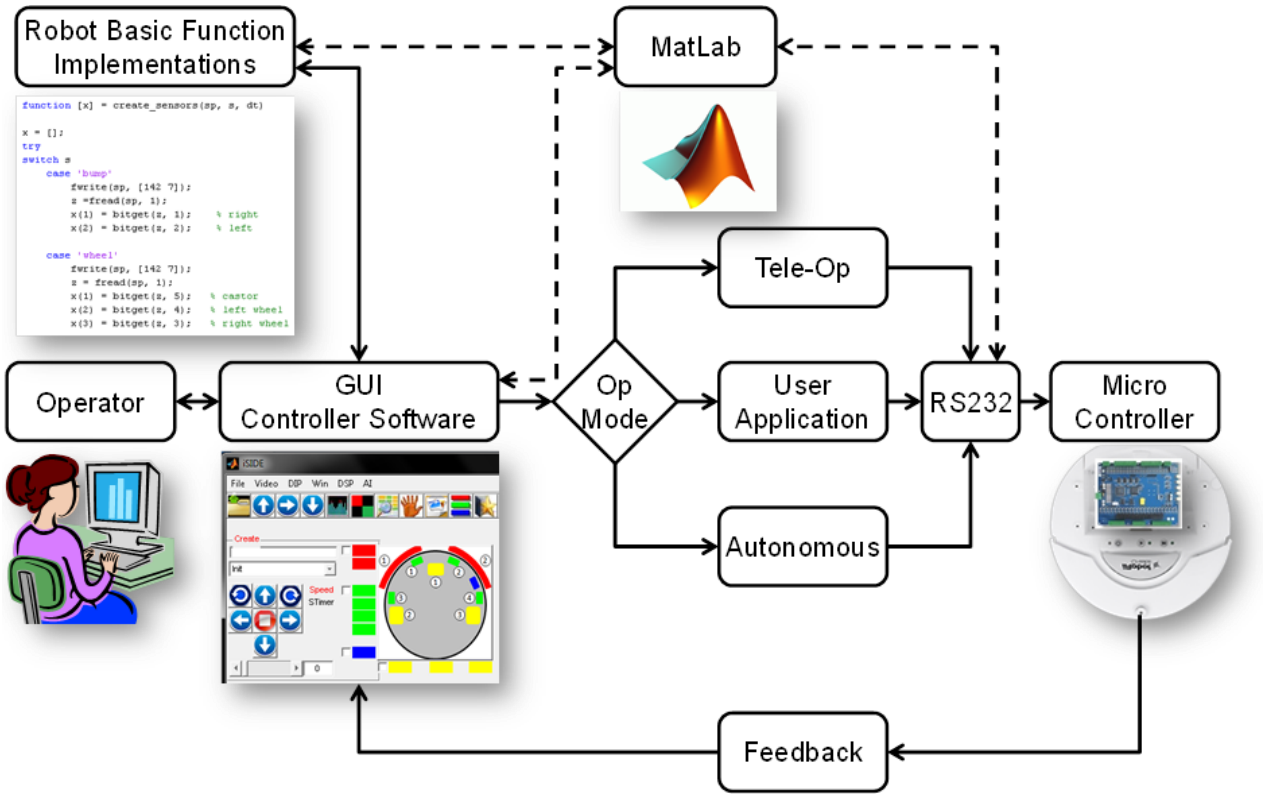

Figure 1: System Architecture

2.2 Robot Basic Functions

This component is the MatLab implementation of the basic functions of the robot that are necessary to control the robot, some of these are depicted in Figure 3 and explained here:

- *create_backDist*: given serial port object, speed (mm/s) and distance (mm), this function drives the robot forward the requested distance and then stops.

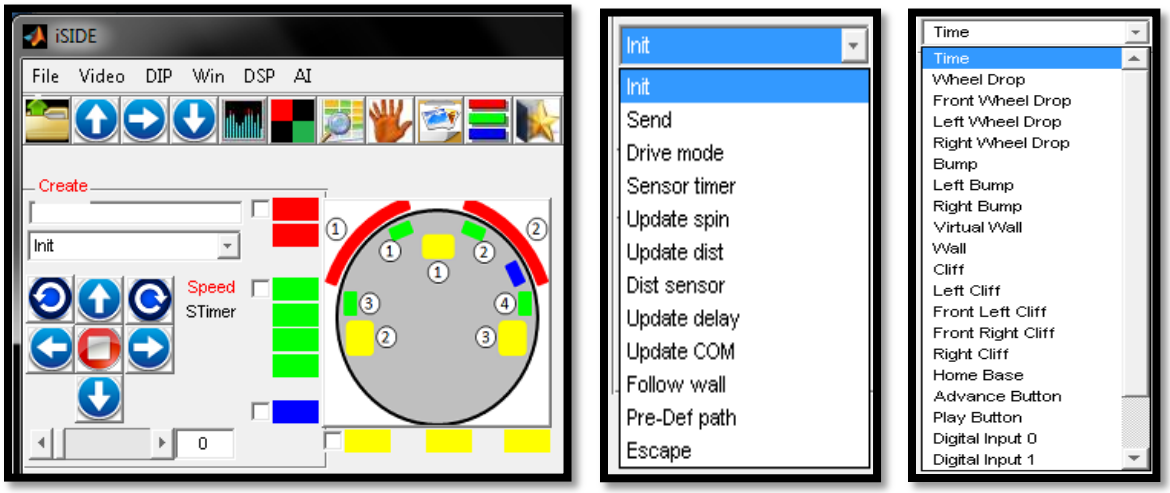

Figure 2: The GUI Software Different Controls

- *create_backward*: given serial port object and speed (mm/s) this function drives the robot at the requested speed indefinitely.
- *create_forDist*: given serial port object, speed (mm/s) and distance (mm), this function drives the robot backward the requested distance and then stops.
- *create_forward*: given serial port object and speed (mm/s) this function drives the robot at the requested speed indefinitely.
- *create_sensors*: given serial port object, sensor option and delay (sec), this function returns the current readings of the requested sensors.
- *create_init*: given the COM port# option, this function sends the initialization commands to the robot to get it ready before any task.
- *create_leds*: given serial port object and the state (0/1) of each LED of the robot, the robot sets the LEDs to these states on/off.
- *create_spinCCW*: given serial port object, rotational angle and the rotational speed, this function rotates the robot CCW the specified angle and stops.
- *create_spinCW*: given serial port object, rotational angle and the rotational speed, this function rotates the robot CW the specified angle and stops.
- *create_stop*: given SP, this function sets the speed of the robot to "0" and stops the robot.

2.3 Operational Mode:

Three modes of operation can be activated according to the designed RISC system, these are:

- *Tele-Operation*: the operator interacts with the robot remotely and manually sends commands to the robot to perform simple task or request a status. {go forward, turn left, speed up, stop, read ultrasound sensor, etc.}

- *User Application*: the operator activates a program at the user side that has a certain goal via a series of commands and constructs. (not limited to the robot OS language). {Follow wall, find an object, etc.}
- *Autonomous*: the operator activates a predefined behavior in the robot micro-controller. {sweep space, home to base, etc.}

```
function [x] = create_sensors(sp, s, dt)

x = [];
try
switch s
    case 'bump'
        fwrite(sp, [142 7]);
        z =fread(sp, 1);
        x(1) = bitget(z, 1);    % right
        x(2) = bitget(z, 2);    % left
```

```
function create_spinCCW(sp, a, s)

if s < 0
    disp('ERR: speed < 0 ...');
    return;
end

if a < 0
    disp('ERR: angle < 0 ...');
    return;
end

fwrite(sp, [137 bin22sdecBytes(dec22s(s,16))  0 1]);
fwrite(sp, [157 bin22sdecBytes(dec22s(a,16))]);
fwrite(sp, [137 0 0 0 0]);
```

```
function create_forDist(sp, d, s)
if s < 0 | s > 2000
    disp('ERR: speed [0, 2000] ...');
    return;
end

if d < 0
    disp('ERR: distance < 0 ...');
    return;
end

fwrite(sp, [137 bin22sdecBytes(dec22s(s,16))  128 0]);
fwrite(sp, [156 bin22sdecBytes(dec22s(d,16))]);
fwrite(sp, [137 0 0 0 0]);
```

```
function [sp] = create_init(usrCOM);
global td
td = 0;
comm = strcat('COM', num2str(usrCOM));
sp = serial(comm,'BaudRate', 57600);
set(sp,'Terminator','LF');    % LF = lin
set(sp,'InputBufferSize',100)% 100 byte
set(sp, 'Timeout', 1);          %connecti
set(sp, 'ByteOrder','bigEndian');    % b
set(sp, 'Tag', 'Create');
fopen(sp);
fwrite(sp,[128]);  % start byte code
pause(td)
fwrite(sp,[132]);  % set operational mo
pause(td)
```

Figure 3: Sample Code Implementation of Basic Functions in MatLab

2.4 Serial Port (RS232):

The communication link between the computer and the robot in this project is the serial port (RS232) device. Where, sequence of binary of bytes can be transmitted in two-way communication protocol. The robot is already equipped for this protocol and the computer uses the USB ports along with MatLab software interface for serial communication library. Figure 4 shows the pinout of the RS232 as well as the signal profile for transmitting a byte.

The following is an example of writing (sending command) and reading (receiving data) using the RS232 port.

```
% set speed to 0 => stop
% (137)10 = (10001001)2
fwrite(serial_port_obj, [137 0 0 0 0]);
x = fread(serial_port_obj, 2);
```

The ***serial_port_obj*** is created and initialized/configured with appropriate settings first then.

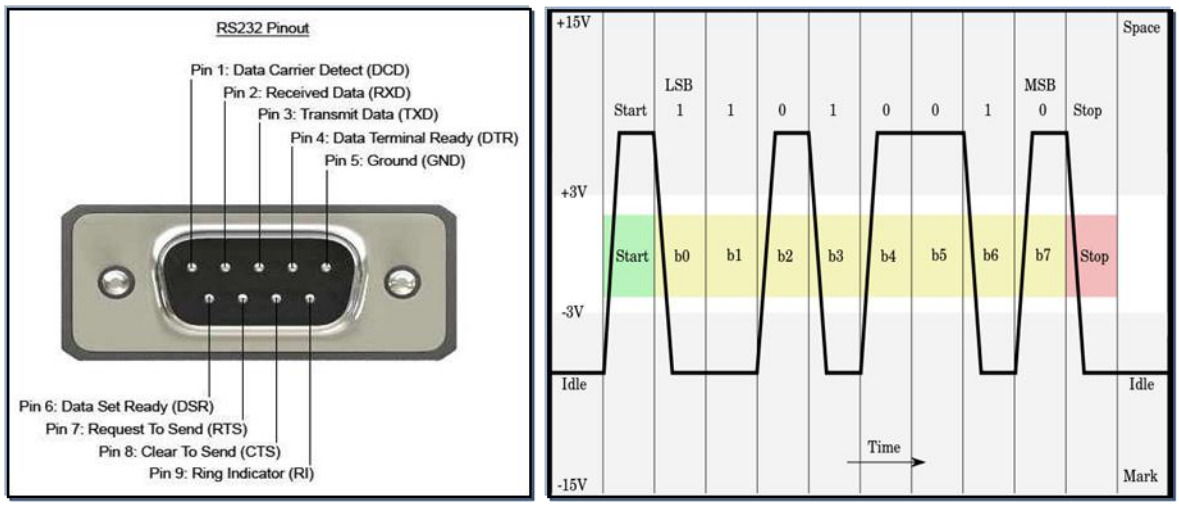

Figure 4: LEFT: Pinout of the RS232 , RIGHT: signal profile for transmitting a byte

2.5 Micro Controller:

The micro controller is the brain of the robot where all the hardware computing devices and needed software that runs on these devices are provided. Every micro controller has an operating system that is light weight compared to regular computer since the type of load and interaction at much lower level. It is equipped with the main CPU chip, memory, storage, wired/wireless communication capabilities. Also, it has A/D converters for sensing and I/O connections Figure 5 a typical picture of a microcontroller and Table 1 illustrates a comparison between a micro controller and a regular PC specifications.

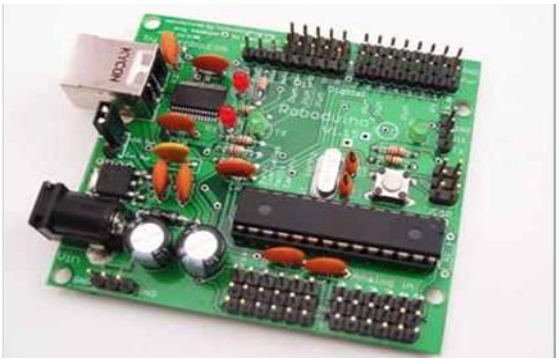

Figure 5: A typical picture of a microcontroller.

Table 1: Comparison between a micro controller and a regular PC specifications as well as

Feature	Micro Controller	PC
RAM	1KB	4GB
Storage	15KB	1TB
Power	0.1W	600W
Input/Output	Pins, USB, RS232	PC peripherals
Wireless	Bluetooth, infrared	Bluetooth, infrared, RF
Video	None	VGA, DVI, HDMI
Internet	WiFi or Ethernet	WiFi or Ethernet
Price	$4 to $300	$400 to $2000

2.6 Robotic Platform:

The mobile robotic platform is the iRobot® Create robot [4]. Create is depicted in Figure 6 below. Create in its basic structure has a mobility base, several sensors, and interface ports for other components extensions such as: cameras, robotics arms, etc. as it shown in Figure 6 (right).

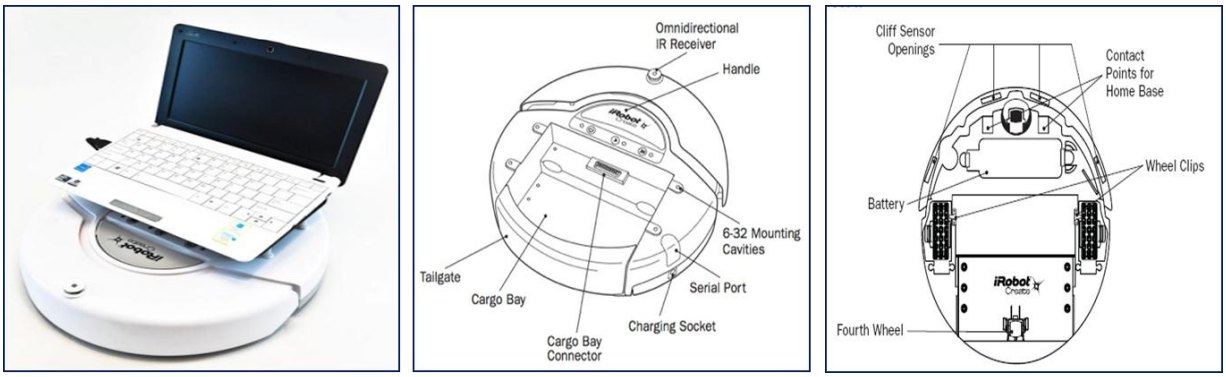

Figure 6: iRobot Create. Left: the real robot with a laptop, Middle: Top view of the robot schematic diagram, Right: Bottom view of the robot schematic diagram

The available sensors on board of the robot are: Wheel Drop, Front Wheel Drop, Left Wheel Drop, Right Wheel, Drop, Bump, Left Bump, Right Bump, Virtual Wall, Wall, Cliff, Front Left Cliff, Front Right Cliff, Right Cliff, Home Base, Advance Button, Play Button, Digital Input 0, Digital Input 1, Digital Input 2, Digital Input 3, and Omni IR. All listed sensors come with the package, but these: virtual wall, home base which need to be acquired separately. A schematic diagram of the different components and sensors of Create is shown in Figure 6. The robot connects to a laptop on board through a serial port (RS232) as the communication link between the user program and the robot command suit. For more details on the robot hardware, the reader is advised to review the user manual at [4].

CONCLUSIONS

We believe that this effort in making robotic technology readily available to our students by providing a friendly software development environment is a good addition to our educational and research environment. Also, establishing a working set of tools that can be used in the lab to train students on basic functionality of the robot and train them how to build complex behavior on top of them is a great help. We intend to keep the project alive and dynamic by attracting more students to pursue their projects at the undergraduate and graduate level in this area of robotics. In the near future, we will attempt to develop intelligent behaviors utilizing the work established in this paper.

REFERENCES

[1] David Hall and Sonya McMullen, "*Mathematical Techniques in Multi-Sensor Data Fusion*", 2004, Artech House, Inc. 685 Canton Street, Norwood, MA 02062.

[2] http://www.scientificamerican.com/article.cfm?id=a-robot-in-every-home

[3] Jennifer S. Kay, "*Robots as Recruitment Tools in Computer Science: The New Frontier or Simply Bait and Switch?*", Association for the Advancement of Artificial Intelligence (www.aaai.org), 2010.

[4] http://www.irobot.com/filelibrary/create/Create%20Manual_Final.pdf

[5] James Wolfer1 Aaron R. A. Rababaah, "Creating a Hands-On Robot Environment for Teaching Assembly Language Programming", Global Congress on Engineering and Technology Education, March, 2005, São Paulo, BRAZIL.

[6] James Wolfer1 Aaron R. A. Rababaah, "An Integrated Khepera And Sumo-Bot Development Environment For Assembly Language Programming", Global Congress on Engineering and Technology Education, November, 2005.

[7] Saed Amer, Amir Shirkhodaie, and Aaron Rababaah, UXO detection, characterization, and remediation using intelligent robotic systems, Proc. SPIE 6953, 69530P (2008).

[8] Aaron R. Rababaah, Emin Kuscu, Amir Shirkhodaie, Indoor Mobile Robot Localization Using IPS Cricket Technology, 2010 MTMI-NIT INTERNATIONAL CONFERENCE ON Global Issues in Business & Technology, (December 22 – December 24, 2010).

[9] http://wiki.tekkotsu.org/index.php/Main_Page

[10] www.mathworks.com

[11] Aaron R. Rababaah, "*Intelligent Systems Integrated Development Environment*", a software for machine intelligence algorithms development, University of Maryland Eastern Shore, 2011.

[12] Dr. Joel M. Esposito:http://www.usna.edu/Users/weapsys/esposito/