ISSN:0975-9646

Saswat KUmar Jena et al, / (IJCSIT) International Journal of Computer Science and Information Technologies, Vol. 2 (5) , 2011, 2052-2054

Development of a Robot-Car in LEGO Mindstorms NXT tool kit for Obstacle Avoidance using JAVA

[#1]Saswat Kumar Jena, [#2]Prof. (Dr.) Srikant Patnaik

[#1] Department of Computer Applications, I.T.E.R, Siksha "O" Anusandhan University Jagamohan Nagar, Bhubaneswar, India

[#2]Department of Computer Science & Engineering, I.T.E.R, Siksha "O" Anusandhan University, Jagamohan Nagar, Bhubaneswar, India

***Abstract*-A robot-car has been designed using LEGO Mindstorms NXT tool kit which can detect obstacles if any by use of various sensors e.g. Touch Sensor, Light Sensor, Ultrasonic Sensor etc. One Lego Mindstorms NXT brick which is the brain of the robot is mounted upon the robot-car and is having LEJOS as the operating system. The sensors act as input and the servo motors are responsible for the motion of the robot-car. Java programs were developed, compiled and executed and were uploaded to the NXT brick which converts the java programs with *.java* extension to *.nxj* extension.**

***Keywords*- LEGO Mindstorms NXT, LEJOS, Touch Sensor, Light Sensor, Ultrasonic Sensor, Sound Sensor**

I. INTRODUCTION

Nowadays robotics system is being applied in Medical Surgery, Factory automation, military, security systems, space research and many more. Now many vendors are interested in engaging robots in human's daily life as service robot. The commonest thing about the above mentioned type of robot is that they can move from one location to another. Unlike Human beings robots cannot take decision of their own. Robots can sense the presence of obstacles through various sensors (In case of LEGO Mindstorms NXT tool kit various sensors are Touch Sensor, Ultrasonic Sensor, Light Sensor, Infrared Sensor etc.) and process accordingly and move further. The processing is done by the processor the specification of which in details is given later in this paper.

A robot-car is been designed which has a touch sensor attached to the front for detecting any obstacle which is present on the front side of the robot-car. There is a button on top of the touch sensor which gets pressed when an obstacle (say wall) touches the sensor. The pressing of the tip of the touch sensor returns a Boolean value true to the program when is queried through a method "isPressed ()" and this runs the code written for obstacle avoidance.

This part of the paper describes how to design the robot car with the available spare parts of LEGO Mindstorms NXT.

II. ROBOT-CAR ARCHITECTURE AND DESIGN

As shown in Fig-1 the robot car consists of programmable unit known as Lego Mindstorms NXT brick which is essentially a microcomputer.

Fig-1: The robot-car with Lego Mindstorms NXT & Touch Sensor

III. SPECIFICATION OF LEGO MINDSTORMS NXT

LEGO MINDSTORMS NXT is a robotics toolset that provides limitless opportunities for robotics fanatics and LEGO builders of all ages to build and program robots. New technologies and expanded sensor capabilities challenge more experienced robot creators. The programming language has been revamped, as have the sensors, motors, and I/O ports. New features include Bluetooth, a 12Mbit/sec USB port, a 100x64 LCD panel, a speaker. As a result, Mindstorms NXT robots look and act far more realistic than their predecessors. The kit includes three interactive servo motors with inbuilt rotation sensors to align speed for precise control. A new ultrasonic sensor makes robots see by responding to movement. The sound sensor enables robots to react to sound commands, including sound pattern and tone recognition and improved light sensor detects different colors and light intensity. The touch sensor mechanism is essentially made up of a basic sensor pad that detects pressure.

The heart of the new system is the NXT brick: a 32-bit ARM7 microprocessor, with 256KBytes of flash memory and 64KBytes RAM that can be programmed using a PC or a Mac. After building their robots, users create a program within easy-to-use and feature-rich software, powered by LabVIEW from National Instruments - programming system made up of intuitive icons. You can also program it using java or NXT_Python.

Users can download programs wirelessly using Bluetooth-enabled computer hardware or they can use the included USB 2.0 cable. The robot then takes on a life of its own, fully autonomous from the computer. The inclusion of Bluetooth technology also extends possibilities for controlling robots remotely, for example, from a mobile phone or PDA.

Step-by-step instructions help acclimate users to the new system to create robots ranging from humanoids and machinery to animals and vehicles and in this case this is a robot-car. The firmware and developer kits for Lego Mindstorms NXT are open source.

Fig-3 Disassembled parts of Lego Mindstorms Kit

Fig-4: Lego Mindstorms NXT brick with I/O devices

1-Lego Mindstorms NXT controller, 2-Touch Sensor, 3-Sound Sensor, 4-Light Sensor, 5-Ultrasonic Sensor, 6-Servo Motors.

IV. FUNCTIONS OF VARIOUS PARTS OF LEGO MINDSTORMS NXT

NXT Controller Technical specification-It consists of a 32-bit ARM7 microcontroller with 256 Kb flash memory and 64 Kb RAM and another 8-bit AVR microcontroller with 4 Kb flash memory and 512 Byte RAM. The NXT controller also bears a Bluetooth device for wireless communication (Bluetooth Class II v2.0 compliant), and for wired communication it uses USB port (12Mbits/s). To connect various I/O devices it contains 4 input ports and 3 output ports.

1. ***Touch Sensor***-Enable the robot to feel and react to its environment.
2. ***Sound Senor***-Enables the robot car to hear and react to it.
3. ***Light Sensor***-Enables the robot car to detect light and color.
4. ***Ultrasonic Sensor***-Enables the robot to see, measure distance to an object and react to movement.
5. ***Servo Motors***-Ensures that the robot moves with precision.

V. LOCALIZATION PROBLEM:

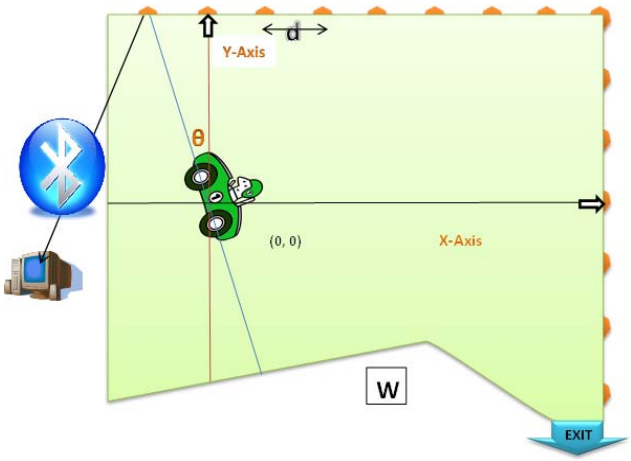


Fig-5: **Workspace W where robot-car starts moving from the origin (0, 0). The surrounding 15 equidistant points are those where the robot touches the wall during the traversal and finally escapes the workspace through the exit gate. 'd' is the distance between two neighboring points on the wall and 'θ' is the angle between initial direction and the direction after deviation.**

As depicted in the above figure W is a workspace and has hour walls and only one exit gate. The robot-car starts from the origin (0, 0) of the co-ordinate axis at time t_1 and moves forward till touches the first orange point among 15 marked points on the wall. The robot-car bears a touch sensor in front of it and when it touches any obstacle may it be wall, fingertip etc. there is a button like element on top of the touch sensor which gets pressed and once it gets pressed it returns a true value to the Java Program when queried through *isPressed()* method. The movement of the robot-car is due to the mechanical force given by two servo motors *Motor A* and *Motor C* connected to *A port and C port* of the NXT controller respectively. For forward movement both the servo motors are set to move in forward direction with a desired speed till it touches wall by writing the following Java code

```
Motor.A.setSpeed (400);
Motor.A.forward ();
Motor.C.setSpeed (400);
Motor.C.forward ();
```

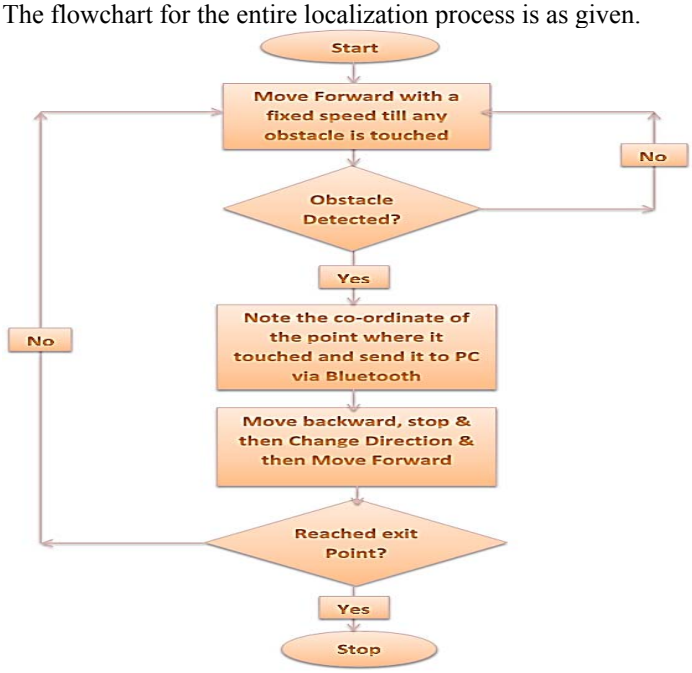
The next task is to move the robot backward and then turn right deviating an angle θ from the initial direction. This is accomplished by the following Java code given below:

```
Motor.A.stop ();    //Stops Motor A
Motor.C.stop ();    //Stops Motor C
Motor.A.backward (); //Motor A moves reversely
Motor.C.backward();  //Motor A moves reversely
Thread.sleep (2000); //Halt for two seconds
Motor.A.stop (); /*Rotate with angle by making*/
Thread.sleep (300); /*only one wheel to stop*/
Motor.C.stop (); /*for 300 milliseconds*/
```

The next job is to keep track of the co-ordinate points of all the 15 orange marked equidistant (distance d) points of the wall where the robot-car is supposed to touch theoretically. Later on this report can be sent to a computer through the built-in Bluetooth device of the NXT controller for analysis. The analysis part is left for the future work.

VI. FLOW CHART

The flowchart for the entire localization process is as given.



VII. ADVANTAGES AND LIMITATIONS

The main advantage of the robot-car that we have designed lies in the implementation part where we have used Java as the programming language which can be used in any operating system due to the platform independent nature of Java. Apart from that it is more robust, reliable and efficient than other models designed in other languages like C, NQC (Not Quite 'C') etc. The main disadvantage is that it is costly and the car cannot sustain much load as it would demand high power consumption.

VIII. CONCLUSION

Today the world is getting mechanized where human thinks and machine does work .In this present scenario the robot-car can be used to serve different purposes where human need not go. The best example can be the destroyed buildings in earthquake where robot-car can move and can perform rescue operation with little modifications made to it. In addition to the touch sensor, Ultrasonic Sensors and Light Sensors along with night vision camera can be attached to the robot-car to make it more efficient. This can also be used in space research and many more.

IX. REFERENCES

[1] Borenstein J, Koren Y, "Real-time obstacle avoidance for fast mobile robots," IEEE Int. Conf. Systems, Man and Cybernetics, September 1989, pp. 1179 - 1187

[2] Koren Y, Borenstein J, "Potential field methods and their inherent limitations for mobile robot navigation," IEEE Int. Conf. Robotics and Automation, April 1991, pp. 1398 - 1404

[3] Thongchai S, Kawamura K, "Application of fuzzy control to a sonar-based obstacle avoidance mobile robot," IEEE Int. Conf. Control Applications, September 2000, pp. 425 – 430

[4] Borenstein J, Koren Y, "The vector field histogram-fast obstacle avoidance for mobile robots," IEEE Int. Conf. Robotics and Automation, June 1991, pp. 278 - 288

[5] Mbede J B, Huang X, Wang M, "Robust fuzzy and recurrent neural network motion control among dynamic obstacles for robot manipulators," IEEE Int. Conf. Robotics and Automation, April 2000, pp. 2136 - 2145

[6] Moravec H, Elfes A, "High resolution maps from wide angle sonar," IEEE Int. Conf. Robotics and Automation, May 1985, pp. 116 – 121