COURSE PROJECT DOCUMENTATION

CS101 EMBEDDED SYSTEMS PROJECT

# TURTLESIM BOT/Robot to draw shapes


TEAM ID 394

TEAM MEMBERS

DIPTI RANJAN SAHU: 140070038

ABHAY VIKRAM: 140040113

ROHAN VORA: 14D170003

SURYA TEJA: 140070048

# TABLE OF CONTENTS

## INTRODUCTION

The purpose in this project was to design a bot that could imitate the Turtle in the TurtleSim function of the simplecpp library and draw various shapes on paper upon the execution of a program by the user. This bot has been designed for learners in C++ who are using the simplecpp library to write their first programs. The bot will only execute the program if it is properly written.

Here, the bot uses a marker pen to draw the shape. Drawing is initialized by the compilation of the code. The laptop communicates the appropriate commands to the bot using a Xbee Module.

Some major limitations that our bot currently has our: (1) the user cannot use data types while writing his code. (2) Use of loops and functions, is also not permissible.
(3) The size of the program that the user can write is limited. However, despite these drawbacks, our bot has a wide variety of teaching and designing applications, as we will point out later.

## PROBLEM STATEMENT

The problem at hand is to make a hardware implementation of a small number of commands of the function TurtleSim in simplecpp library, which would enable a user to draw basic shapes on paper.

Our most basic goal was to design a program, which would read the code written by the user, and give instructions to the bot accordingly. The bot, would then trace out the path of the shape that would have been constructed virtually, had the program been compiled on a computer. A pen would be attached to the bot and controlled by a servomotor, allowing the pen to draw a shape as required

## REQUIREMENTS

### A) HARDWARE

1. Firebird: It would serve as the TurtleSim bot
2. Servo Motor: Used for holding the marker and implementing the PenUp(), PenDown() commands.
3. Marker: For drawing the required figure
4. XBee Module: For communication of commands from the laptop to the bot.

### B) SOFTWARE

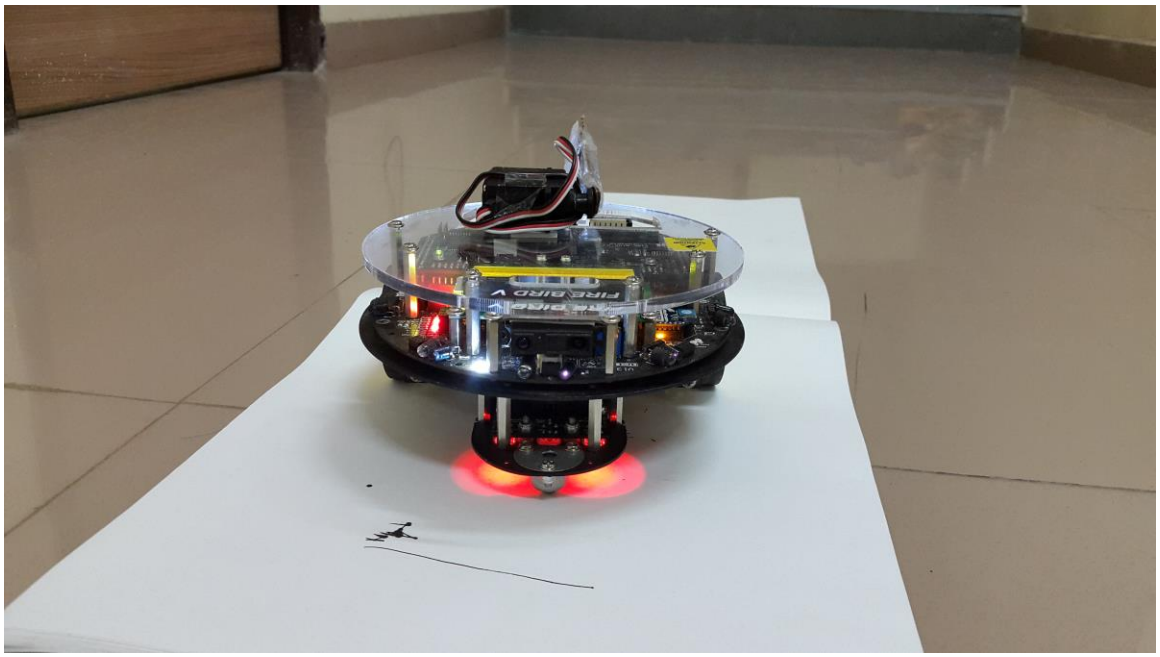
1. Notepad: It would serve as the interface for the user in which the code is written and saved.
2. AVR Studio: To program instructions onto givenbot
3. CodeBlocks: Used for writing the code for our project.

## IMPLEMENTATION

### A) FUNCTIONALITY

The user will write his/her code in Notepad and save it as a file “program.txt”. Our code, will read this file and search for keywords like forward, left, right etc. It will also check for proper syntax within its limitations. Once a keyword is detected, its argument is read as a string, this string is converted into an integer, which is fed into the appropriate function in our code.

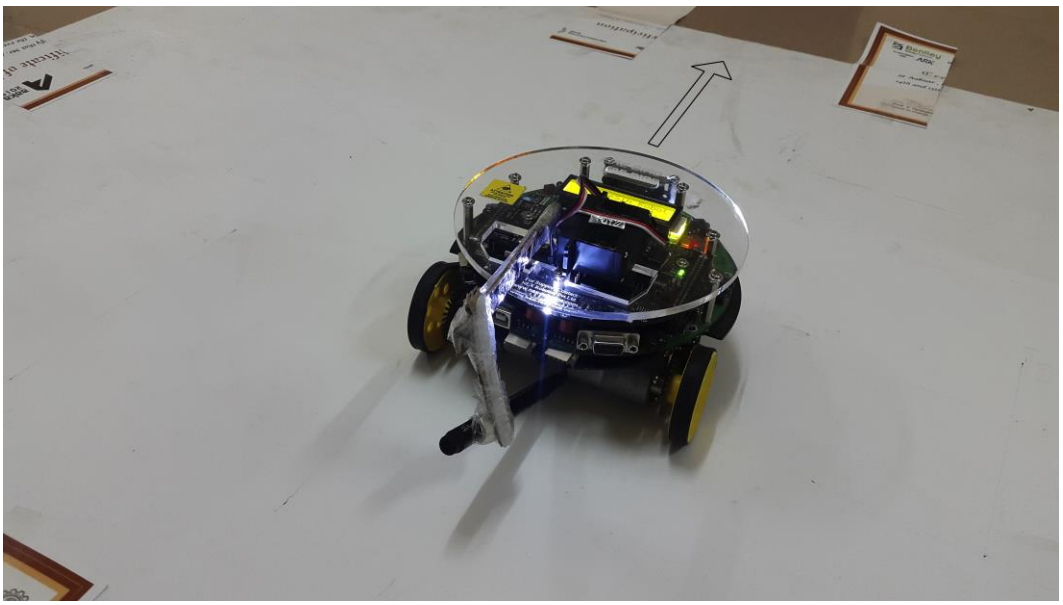
This code, when executed now relays the commands to the bot, which draws the required shape. The marker tip is attached, as close to the axle as possible, which minimizes error in the shape upon turning of the bot. Attached are some views of our bot, to give you an idea what our bot looks like from various angles.



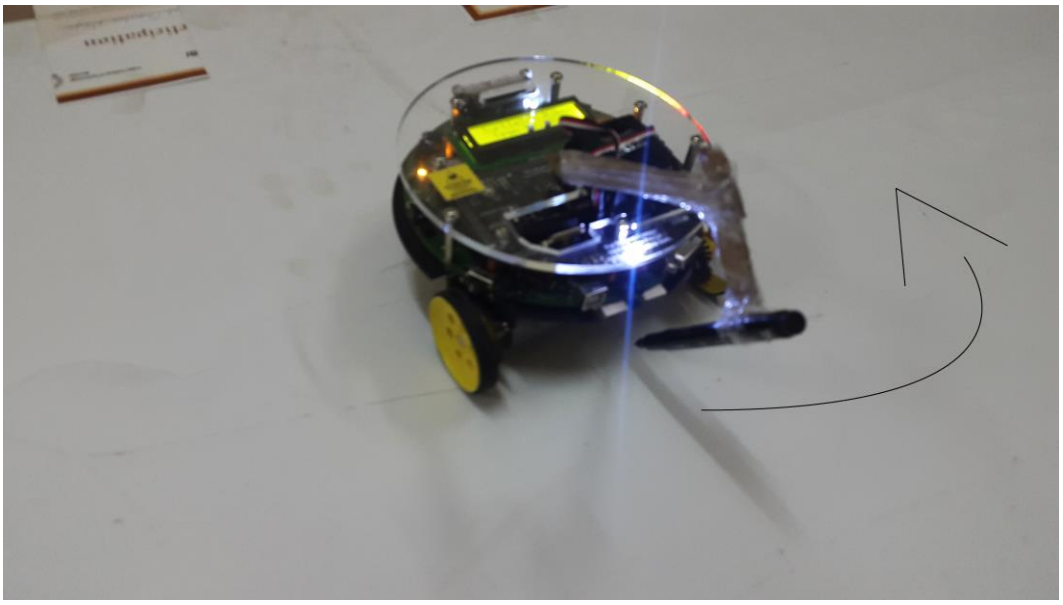
## TESTING STRATEGY

The user enters his code in Notepad. Our program reads and implements his code line by line. Once it encounters a word like "forward" it reads its argument as a string, converts it into a integer and passes the appropriate command to the bot.
A series images depicts the movement of the bot for some commands (as indicated by the arrows)

**1. Forward**



**2. Left**



## 3. PenUp

4. **In case the user enters incorrect code, the following message will be displayed**

## CHALLENGES

We encountered the following challenges during this project.

- **'Translating' commands from simplecpp to Embedded C**: We converted the code for the program into a .txt file and read it using input functions in simplecpp. We searched for particular strings like "forward", "left", "right", etc. and upon finding them, read their arguments as strings, converted them into integers, and passed these numbers as arguments into our drawing functions.

- **Location of the servomotor**: Initially, the servomotor was placed on the sides. However, this led to the writing of unnecessary amounts of code to prevent a discontinuous/deformed shape being formed. This was resolved by placing the servomotor such that the tip of the pen was as close to the axle as possible.

- **Xbee Communication:** Originally, we were thinking to send the entire code via Xbee to the bot, which would then process it accordingly and give the required output. Afterwards, we realised that we could only send characters one-by-one and therefore we dropped this idea and evaluated the code in simplecpp, and sent arguments character by character.

## DISCUSSION OF THE SYSTEM

### A) What worked as per our plans?

1. Placing pen close to the axle: We had hoped to reduce the amount of coding required by at least a 100 lines by placing the pen near the axle. Originally, we had placed the pen on one side of the bot. This meant that at corners, we were forced to move the pen up, move our bot and move our bot into an appropriate position, so that drawing of the next line in our shape could begin. By placing, the pen close to the axle, we ensured that the pen tip did not move while turning thus increasing the efficiency of our bot.
2. Xbee Communication: The Xbee was crucial to our plans for "translating" C++ to Embedded C. Instead of writing a program, which would write the equivalent Embedded C code of the C++ program, we reduced our job to simply searching for keywords within the original program while maintaining correct syntax, to run the bot.

### B) What did we add beyond what was mentioned in the SRS

1. We had written in our SRS that communication with the bot will be done via a *FT232 USB to Serial Converter (built in the Firebot) to connect the bot to the computer via a USB chord.* However, we went one step further and made communication wireless, which increased the flexibility of our bot. This also increased user ease with the system.

### C) Changes made in plan

1. Originally we had thought that we would make the user write the code in CodeBlocks and upon compilation of the code, the program would run. Unfortunately, we realized during the course of the project that this was not a feasible idea, as the bot would only run when the code we had already programmed in the bot would be compiled. That code was only able to access the user-entered code as required if it was saved as a .txt file rather than a .cpp file.
   So now the user enters the Code in Notepad.

## FUTURE WORK

Some modifications possible for our bot are

- Further features of C++ library like data-types are not read in our code. You can try to improve it and add identification of data-types like int, float, etc. in our code.
- Functions like cout in C++ could be displayed on the bot's LCD screen.
- Attaching another servomotor to the bot would enable it to erase any patterns as well (given that they are not permanent). A brush/duster could be attached to the bot through the servo.

## CONCLUSIONS

The beauty of this project is that the code is highly reusable and extendable. Almost any shape drawing, writing application can use this code. Further, if you want to be able to draw more complicated shapes like 5 pointed stars etc. using the bot, just a few extra lines of code need to be added. Our project has design, artistic and teaching applications. Some of them are:

- It can be extended as a drawing bot, where you design the drawing/sketch on your computer and the bot makes it for you.
- Just like TurtleSim was a learning tool for beginners to C++. This bot can also be a fun and educative way to begin learning the language.

## REFERENCES


1. Xbee Reference Manual
2. Firebird V Hardware and Software Manuals
3. https://github.com/E-yantra
4. https://msdn.microsoft.com/en-us/windows/desktop/
5. https://www.digi.com/products/wireless-wired-embedded-solutions/zigbee-rf-modules/xctu


Github Repository:

https://github.com/diptisahu/TurtleSim_Robot/tree/9f83be38bc418c838039176aa137f7b3b9a09910

Youtube Video Links: (For Reference)

1. https://youtu.be/U_PFMgSixfY
2. http://youtu.be/mnUAcpKo-sA