Course Project Documentation

CS-101 Project

EMBEDDED SYSTEMS

ARTISTIC BOT

TEAM ID : 489

TEAM MEMBERS ::

SHUBHAM YADAV :: 140070028
SHUBHAM AGRAWAL :: 140040083
RASHISH RAJENDRA SHINGI :: 14D070058
RAGHAV DAGA :: 140040013

1. PROJECT DESCRIPTION

This project was basically designed with the intention that, as manual drawing of a sketch on a large scale leads to a significant amount of inaccuracy, it would be better, if a bot replicates a similar sketch drawn on a user interface by the user.

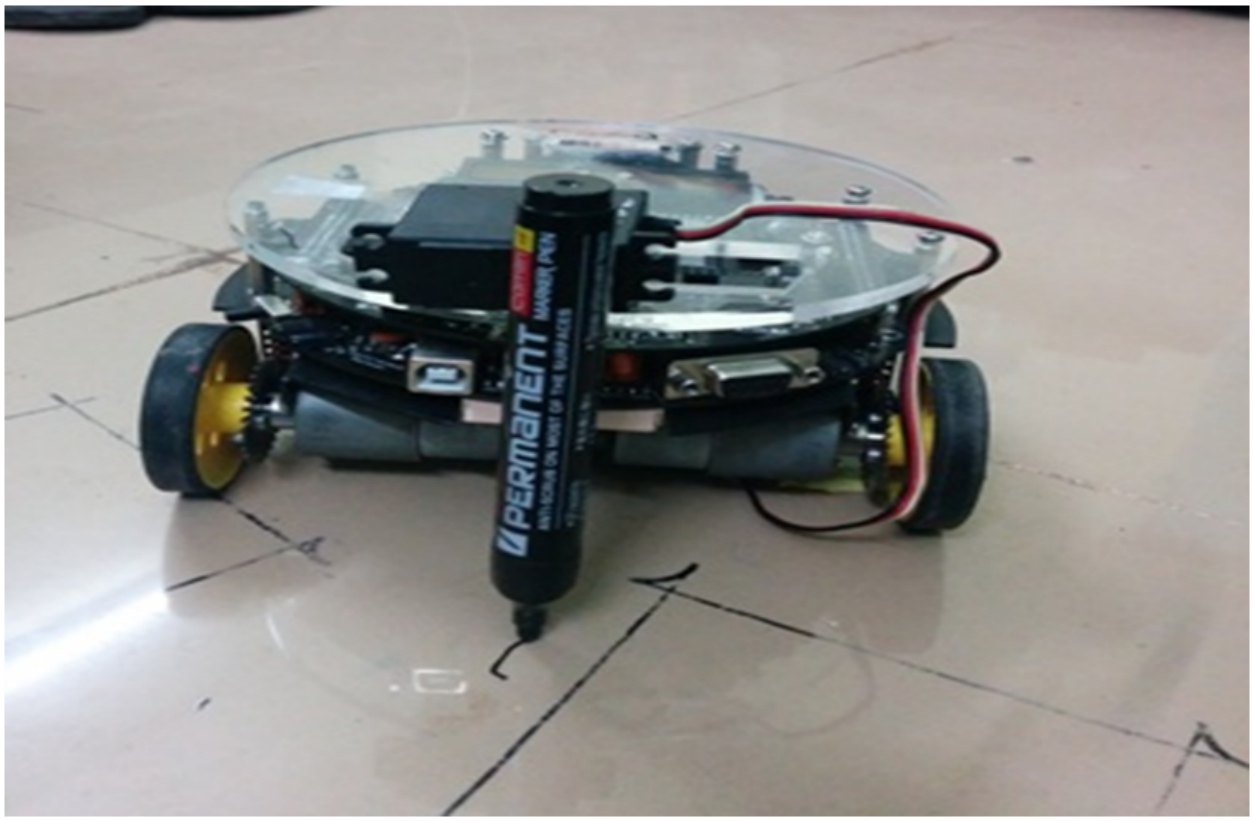
This was the main contention of our project which instilled the inspiration to work towards acheiving our goal.

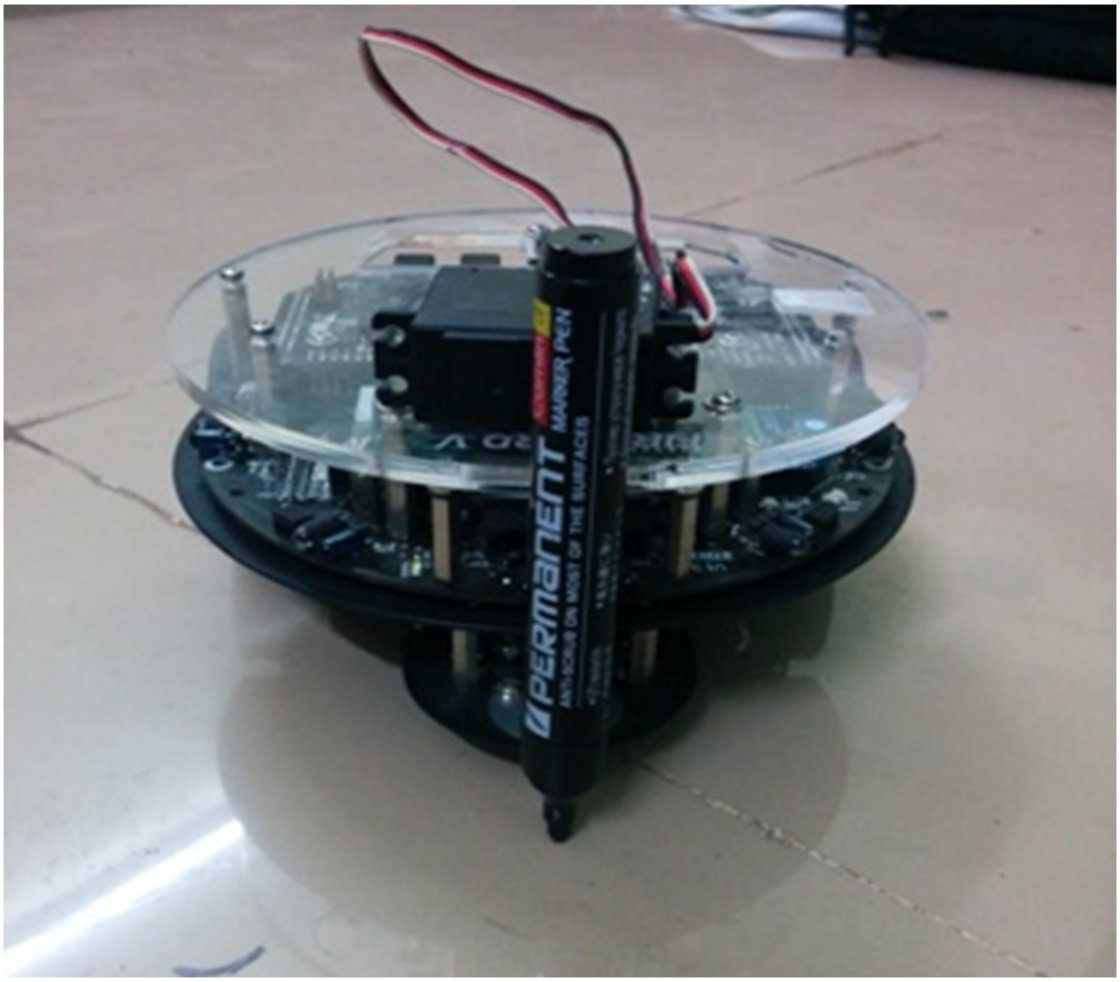
Upon drawing the sketch onto the simplecpp interface, the user compiles the other file and if the x-bee's are properly connected and fully functional (they are in proper range), the bot starts drawing the required sketch on the paper with considerably reduced inaccuracy.

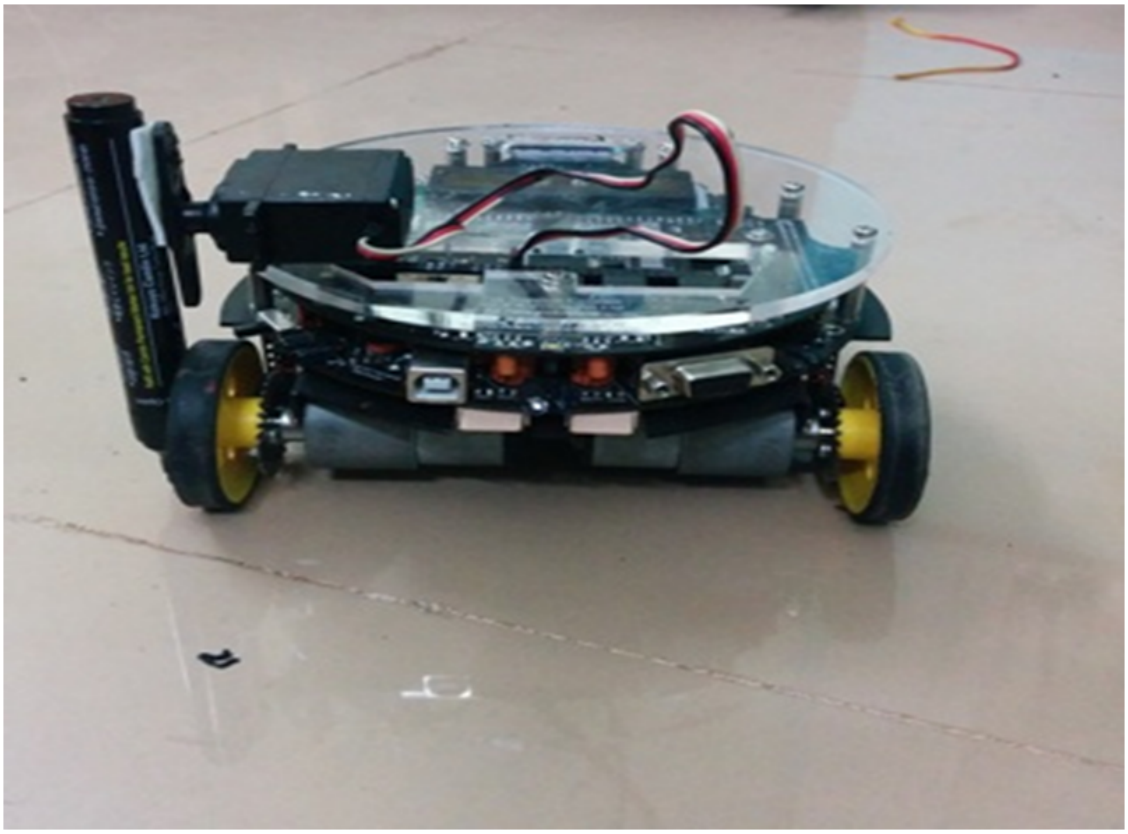
2. PROBLEM STATEMENT

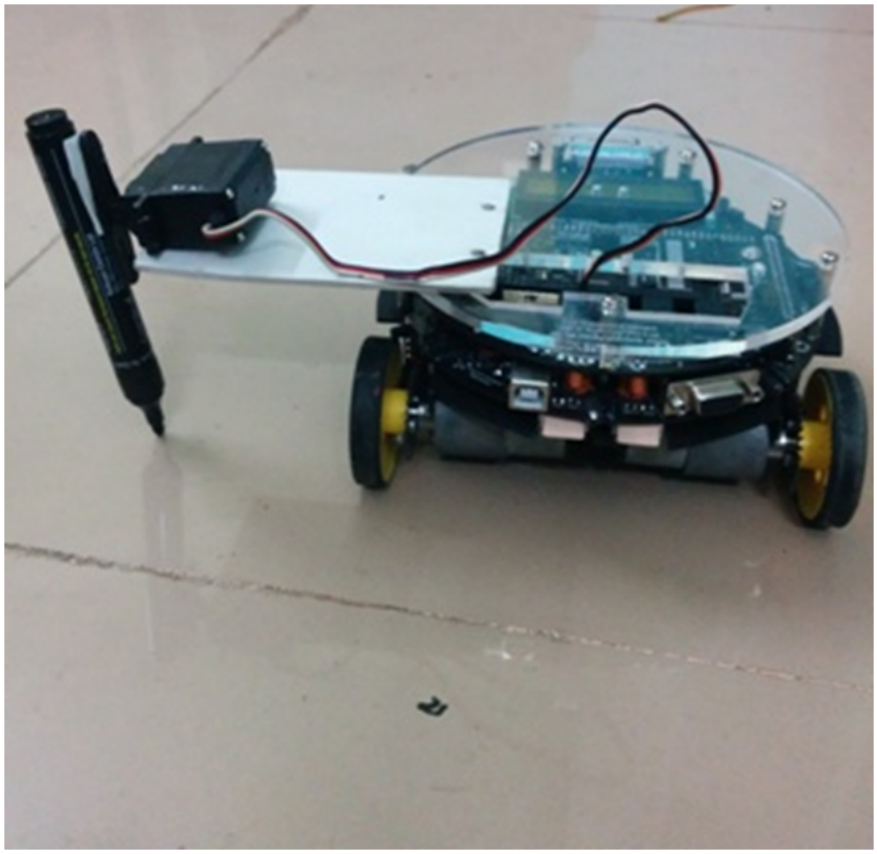
Stage 1 : Gripper pen and error correction & Motion of Bot…

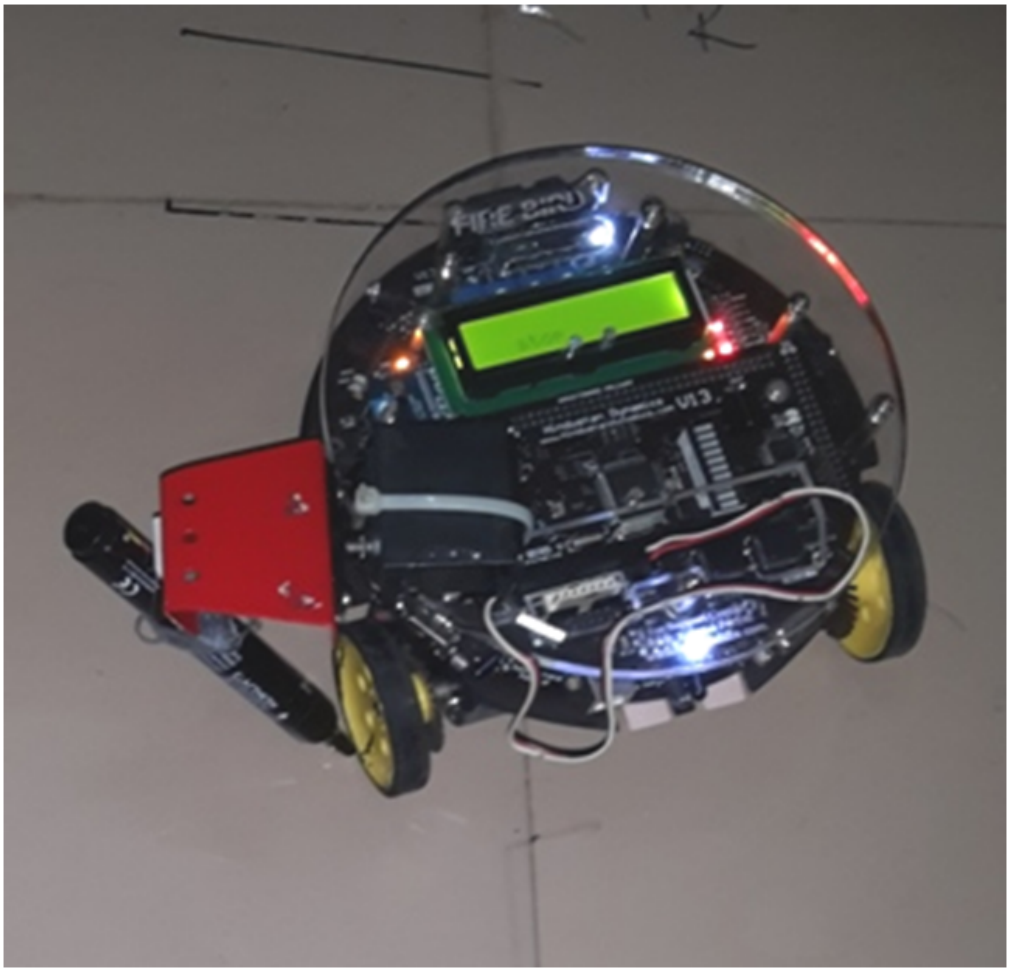
The initial step of our project was to have a gripping mechanism controlled by servo motor for proper positioning of pen and minimum error. Proper positioning of servo motor was required for a perfect sketch.




PERMANENT
MARKER, PEN





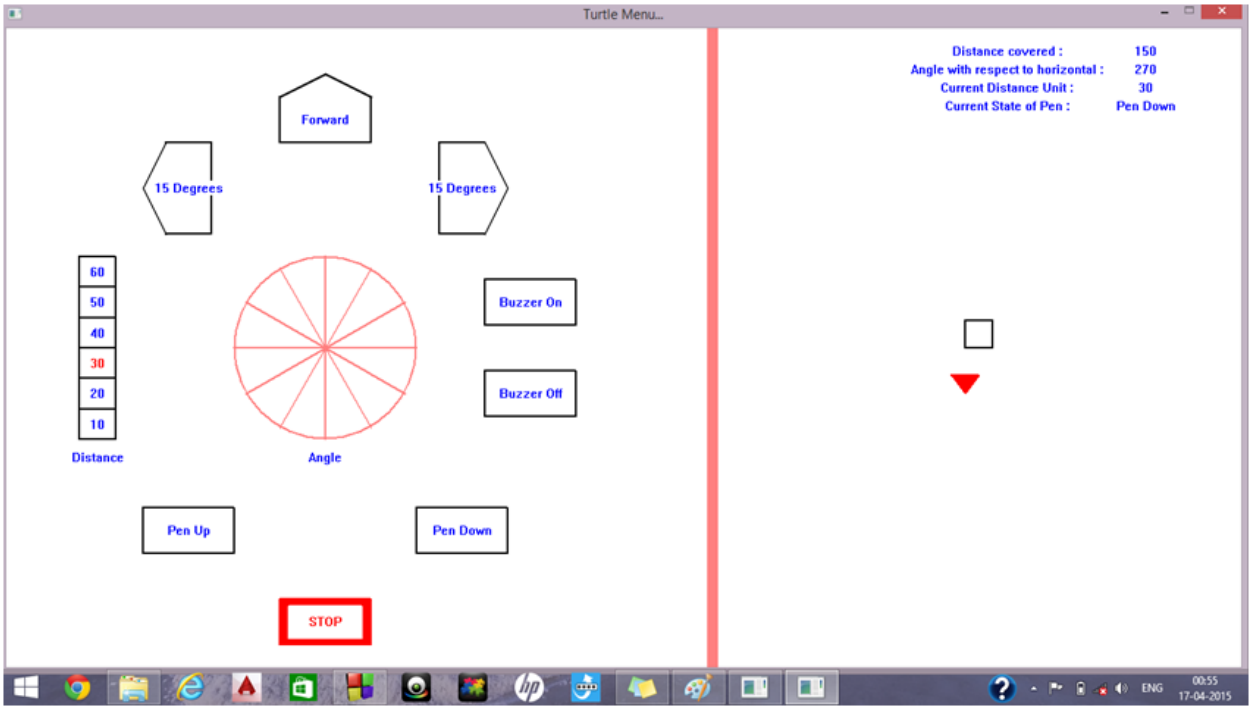


Stage 2 : User Friendly Simplecpp Interface…

This was certainly a very important part of the project because this the user interface in simplecpp was an entity, where the user would draw the sketch which he/she wants the bot to replicate.

The major steps that are accomplished in this stage will be:

- **Providing different options of length, angle rotation, buzzer options, pen up/down for user for a perfect drawing on simplecpp.**
- **Proper storage of data through file handling so that it is not misplaced and the array is sent to the bot.**


Turtle Menu...
Forward
15 Degrees
15 Degrees
60
50
40
30
20
10
Distance
Angle
Buzzer On
Buzzer Off
Pen Up
Pen Down
STOP
Distance covered : 150
Angle with respect to horizontal : 270
Current Distance Unit : 30
Current State of Pen : Pen Down
ENG
00:55
17-04-2015

Stage 3(A) : Wireless Communication…

After making the interface for user and the bot moving mechanism, next task was to send the commands from interface to bot.

Delivered:

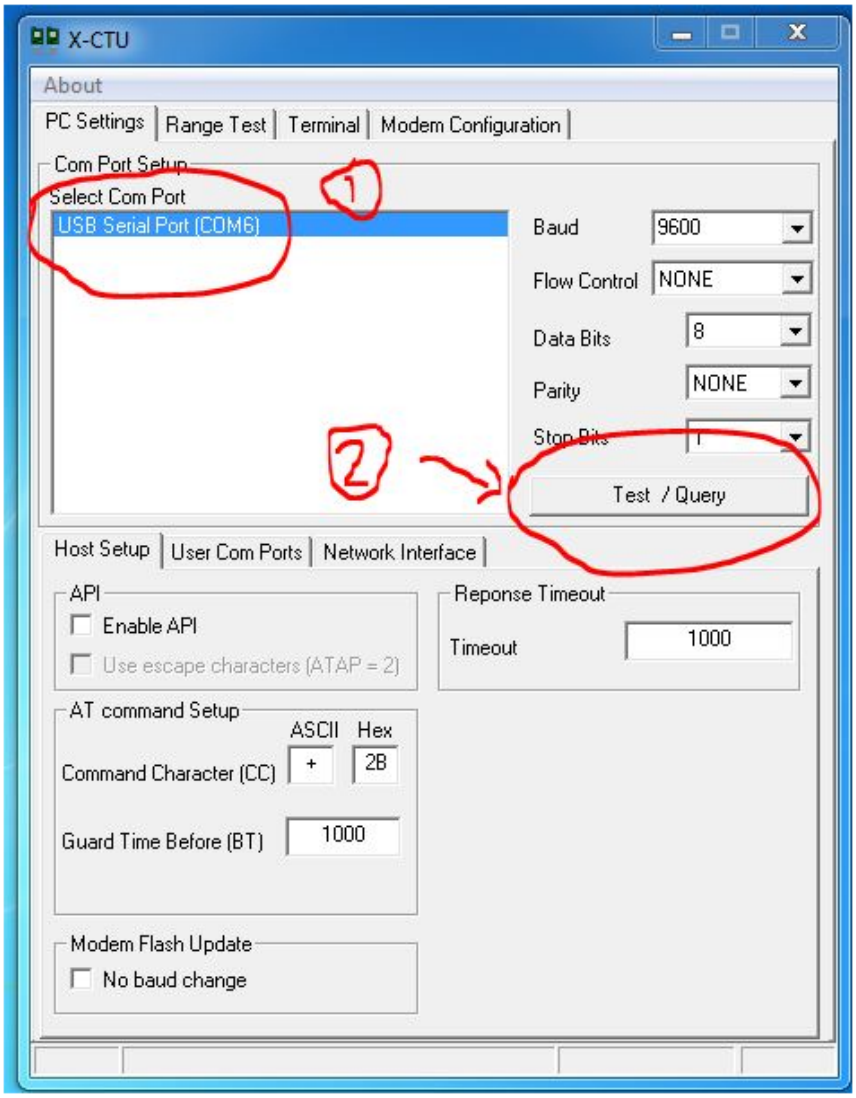
- **We were successful at sending basic commands like move forward, back, etc and buzzer on, off. This was done using Xbee and sending data packets through X-CTU terminal .**

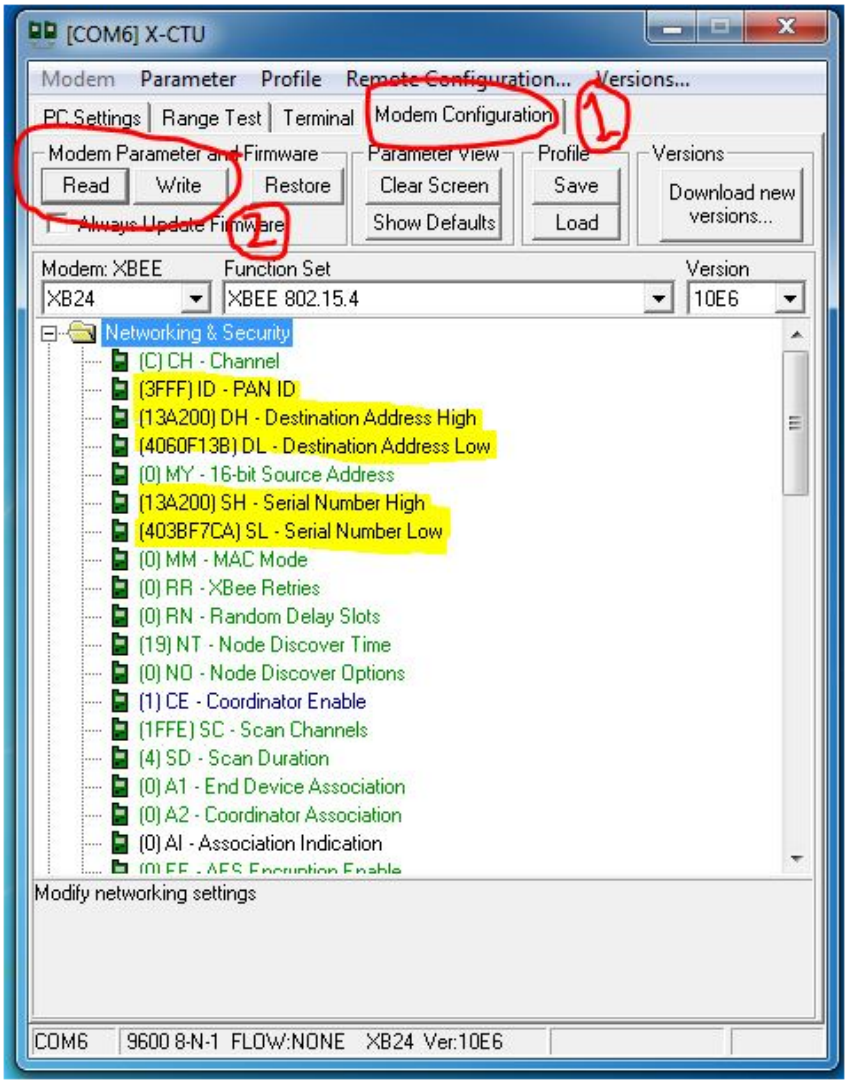
Stage :3(B) Wireless Communication Using SPC…

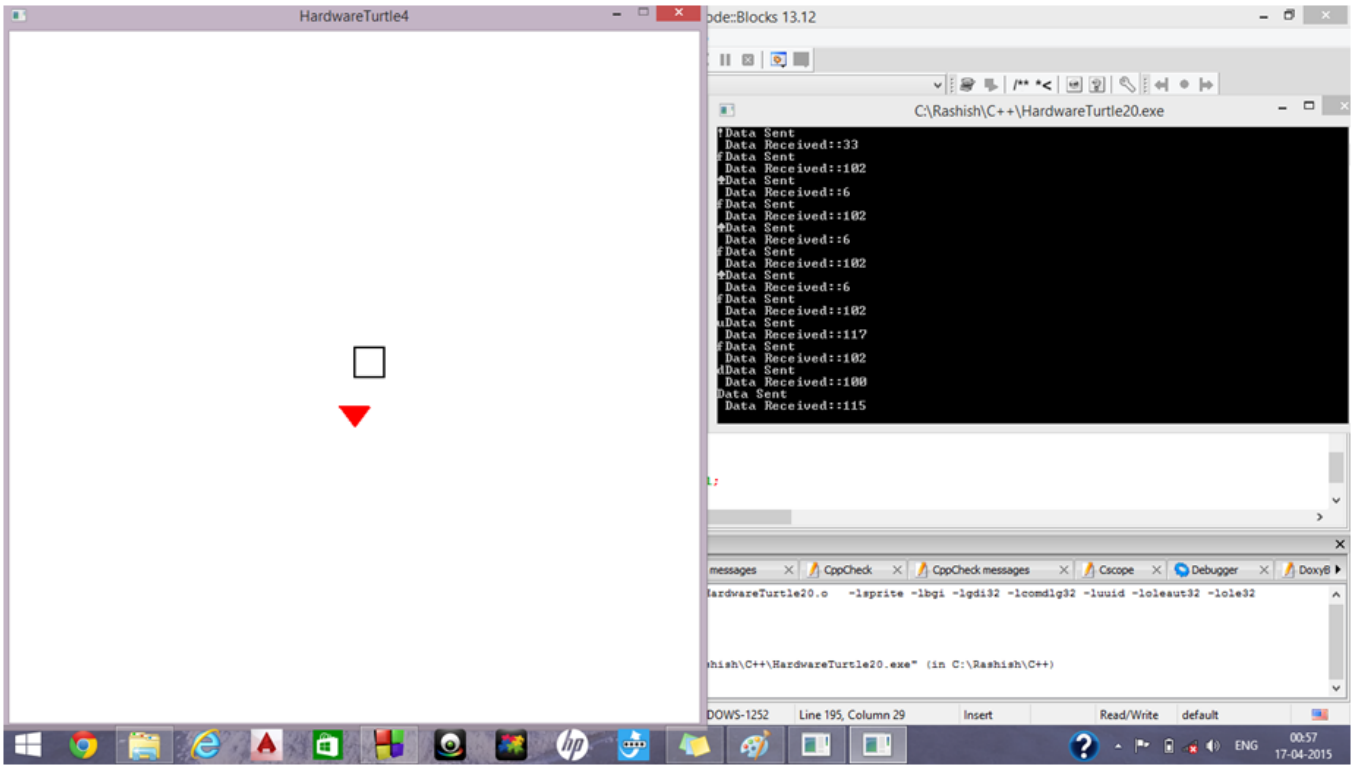
Only Wireless communication was not sufficient. We required to send an array of commands (file handling) rather than data packets (different from the former) to make the bot autonomous.

Delivered:

- **This was indeed a tough task (it consumed a lot of time). Nonetheless, we finally succeeded .We did this including a special library (got after 8+5+3 hrs of research) in our Code::Blocks code, and thus, we are now able to send commands through Serial Port communication (command prompt in windows and terminal in linux) , no longer using X-CTU.**




[COM6] X-CTU
Modem
Parameter
Profile
Remote Configuration...
Versions...
PC Settings
Range Test
Terminal
Modem Configuration
1
Modem Parameter and Firmware
Read
Write
Restore
Always Update Firmware
2
Parameter View
Clear Screen
Show Defaults
Profile
Save
Load
Versions
Download new versions...
Modem: XBEE
XB24
Function Set
XBEE 802.15.4
Version
10E6
Networking & Security
(C) CH - Channel
(3FFF) ID - PAN ID
(13A200) DH - Destination Address High
(4060F13B) DL - Destination Address Low
(0) MY - 16-bit Source Address
(13A200) SH - Serial Number High
(403BF7CA) SL - Serial Number Low
(0) MM - MAC Mode
(0) RR - XBee Retries
(0) RN - Random Delay Slots
(19) NT - Node Discover Time
(0) NO - Node Discover Options
(1) CE - Coordinator Enable
(1FFE) SC - Scan Channels
(4) SD - Scan Duration
(0) A1 - End Device Association
(0) A2 - Coordinator Association
(0) AI - Association Indication
Modify networking settings
COM6
9600 8-N-1 FLOW:NONE XB24 Ver:10E6



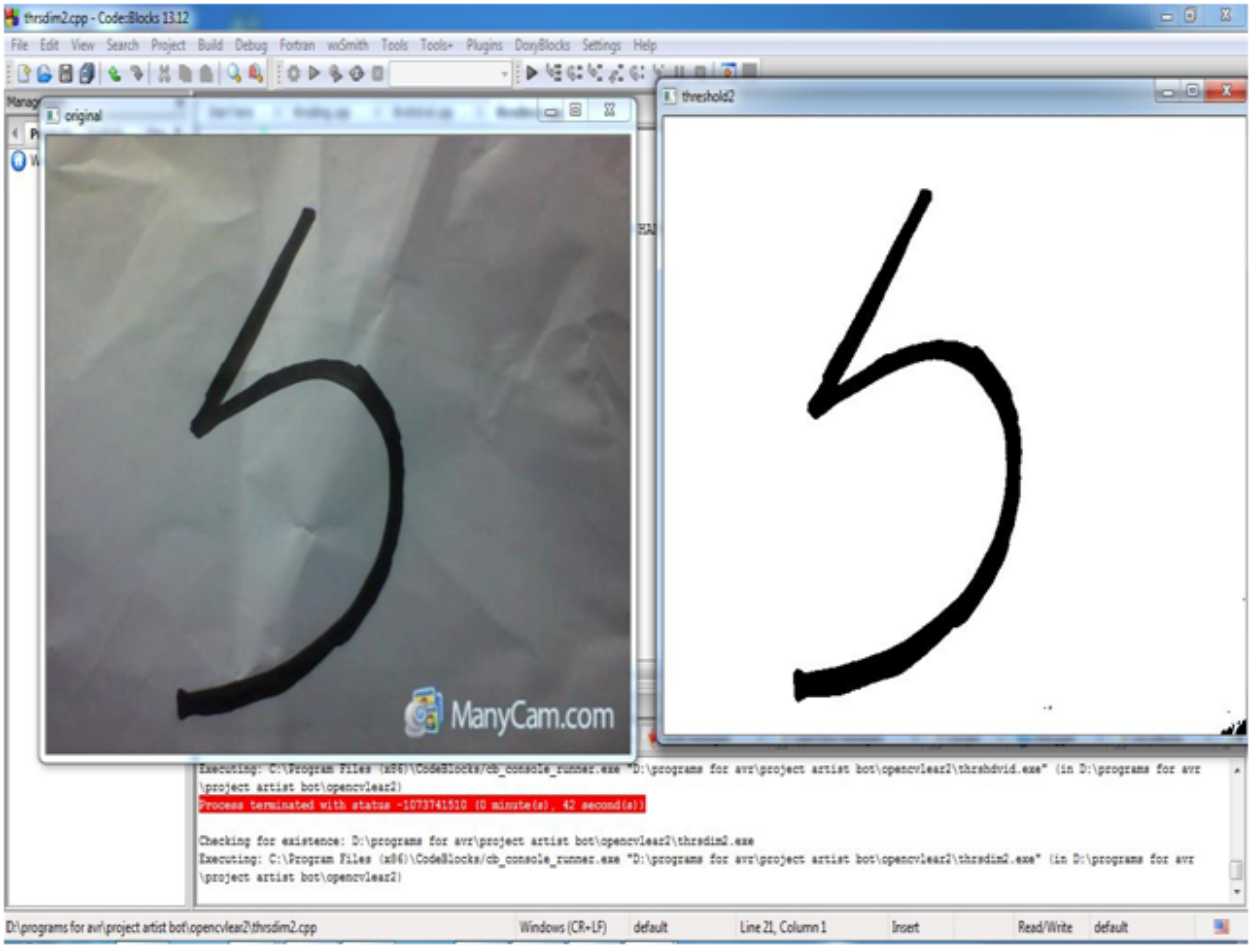
Stage :4 IMAGE PROCESSING AND THRESHOLDING

In this part we were to write a program which can capture and threshold the sketch drawn by the user on a white board and black pen.

Delivered :

- **We are capturing the sketch using webcam of computer rather than a separate webcam on Bot, as it will increase it`s overall cost and also weight of bot hindering its speed and motion.**

Then, through OpenCV we are successful in converting the captured sketch to binary form using threshold (black and white].



3. Requirements:

A) Hardware Requirements

1. FireBird V : Requires 1 bot

2. Gripper : To hold the pen in correct position

3. Marker : To draw the image

4. Xbee : To maintain communication between Bot and Code::Blocks

B) Software Requirements

1. Atmel Studio : To write the program for the bot in Embedded C language

2. Code::Blocks : with Simplecpp

3. AVR studio : To program instruction onto a given bot (acts as a bootloader).

4. OpenCV : For processing images sent by camera.

4. CHALLENGES:

Challenge : Gripper mechanism

Solution: Several positions of gripper were tried out as shown in previous slides with error circles , and angle correction.

The most efficient position of gripper was near the turning wheel which gave least error circle.

Challenge : Storage in the file “moves.txt”.

Some of the ASCII values were non-responding, for e.g. characters corresponding to the ASCII values 10, 12, 13, 32, which were either whitespaces or newlines, which resulted in bugs either in codeblocks or in the bot.

Solution: We used different ASCII values which were not blankspaces and thus, again made our code functional.

Challenge : The configuration of Xbee with X-CTU was a overwhelming challenge for us.

Solution: We overcame this, by trying a set of permutations of configurations, and a lot of research on Google, YouTube and so. Finally, we succeeded at this and were able to do range test, send data packets, commands through terminal.

Challenge : Transferring array of commands to bot with codeblocks

Solution: We overcame this, by trying header files (like libxbee, , serialcomm.h, windows.h) .
And finally "windows.h" worked for us. We were able to send array through file handling, without any signal loss , with configuring with X-CTU

Challenge : Using multiple interrupts –

1. **SPC**
2. **Movement of the bot (position-velocity interrupt)**

Solution:

The data was obtained from the file "moves.txt", which was completely sent in consecutive packets and processed in the bot.

5. FUTURE PATH:

1. The future aspect of this project is to maximize the usage of the hardware on the Bot such as proximity sensors, and using different color pens and Paint bucket so that it can be used in wall paintings, street paintings, Rangoli making, etc.

2. Also using bot with close to zero angular error and perfect movements which can minimize the error in the painting and make it more artistic and beautiful.

3. Wall Sketching

4. Street Sketching

5. Rangoli

6. Including Paint or CAD features in our interface, or communicating the bot directly with AutoCAD or Paint.

6. REFERENCES:

1.https://github.com/eyantra?tab=repositories

2.http://en.wikipedia.org/wiki/Image_processing

3. http://opencv-srf.blogspot.in/

4.http://www.cplusplus.com/forum/windows/76250/

5.https://learn.sparkfun.com/tutorials/exploring-xbees-and-xctu

6.http://www.tutorialspoint.com/cplusplus/cpp_files_streams.htm

THANK YOU !!!