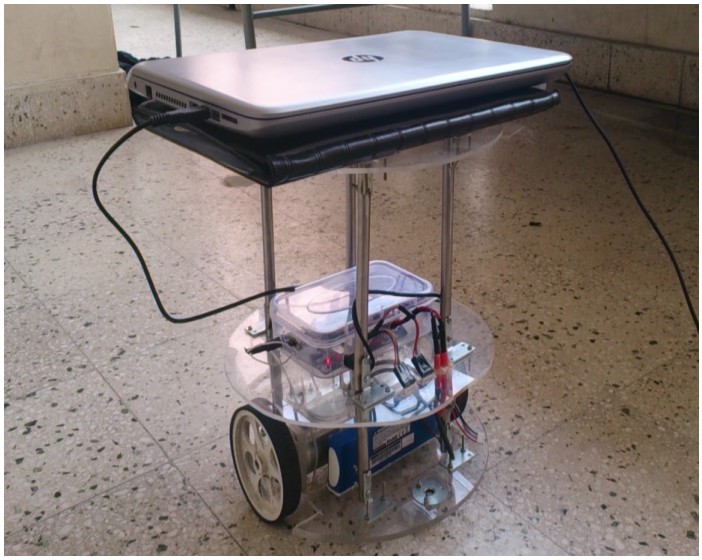
ROBOTICS
IIT GUWAHATI

Technical Board
IIT Guwahati
Kriti 2016

Plane Segmentation on FAIR. (450 points)

Note – There are 3 subparts to this problem (each of 150 points) no partial marks shall be awarded, points shall only be awarded if the subpart is completed.

FAIR or First Autonomous Intelligent Robot is an indoor robot capable of Autonomous Indoor navigation. Our team at the Robotics Club has been working on FAIR since December'14 and we were able to finish the first phase of FAIR in September'15.



This year's task includes working on the FAIR platform with the final aim of doing plane segmentation on the point cloud data from Kinect sensor.
Right now all this might sound very intimidating to you, but there is nothing to worry, we will guide you smoothly through all these tasks through lectures and workshops.

Let's first look at some hardware details of FAIR-

1) Visual senses (Microsoft Kinect version 2.0) – details later.

2) Motors (RMCS 2205) -

HIGH TORQUE ENCODER DC SERVO MOTOR 300RPM W/ UART/I2C/PPM DRIVE

The motor details can be found here
The motor manual can be found here.

The motors can be controlled through I2C communication using an arduino/induino. Every motor has a unique 8 bit hexadecimal address which can be changed as per our needs. Currently the addresses are as follows-
Left Motor - 0x0F
Right Motor - 0x1E

3) Micro-Controller (Induino R4) – runs on same software as an Arduino.

More details can be found here.
Tutorials on arduino programming can be found here.

Objective 1 (150 points) : Designing a controller to drive FAIR (Arduino Task).

This task is meant to be done without using ROS, but you will get full points even if you do it using ROS. The motors are controlled using I2C communication. Check out the motor manual and details given above for more information.
A good tutorial on I2C communication can be found here.

The task is to write your own code to convert the joystick inputs into motor commands and control the motion of the robot. To manually operate FAIR using a joystick/remote control and walk it through a simple maze.

A good tutorial on connecting an analog joystick can be found here.

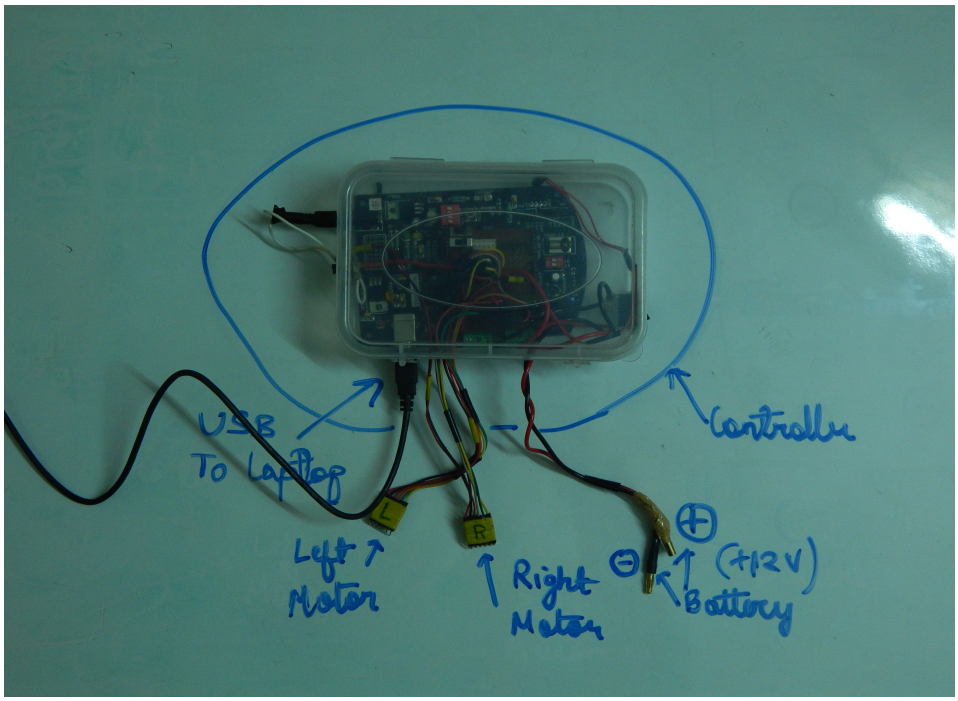
Some of the important functions for I2C with these motors can be found in the images-

```
44  void SpeedRun(byte Address, int Speed)
45  {
46    Wire.beginTransmission(Address);
47    Wire.write(1);
48    Wire.write(Speed         & 0xff);
49    Wire.write((Speed >>  8) & 0xff);
50    Wire.endTransmission();    // stop transmitting
51  }
52
53  void SetMaxSpeed(byte Address, byte Speed)
54  {
55     Wire.beginTransmission(Address);
56     Wire.write(0);
57     Wire.write(Speed);
58     Wire.write(0);
59     Wire.endTransmission();
60  }
61
62  void SetDamping(byte Address, byte Damping)
63  {
64     Wire.beginTransmission(Address);
65     Wire.write(2);
66     Wire.write(Damping);
67     Wire.write(0);
68     Wire.endTransmission();
69     delay(500);    //Required to store settings
70  }
```

```
72  void MoveAbs(byte Address, long Position)
73  {
74     Wire.beginTransmission(Address);
75     Wire.write(4);
76     Wire.write(Position         & 0xff);
77     Wire.write((Position >>  8) & 0xff);
78     Wire.write((Position >> 32) & 0xff);
79     Wire.write((Position >> 24) & 0xff);
80     Wire.endTransmission();
81  }
82
83  void MoveRel(byte Address, long Position)
84  {
85     Wire.beginTransmission(Address);
86     Wire.write(8);
87     Wire.write(Position         & 0xff);
88     Wire.write((Position >>  8) & 0xff);
89     Wire.write((Position >> 32) & 0xff);
90     Wire.write((Position >> 24) & 0xff);
91     Wire.endTransmission();
92  }
```

Controller Specifications:-

- The controller must have an input port for power supply(12V LiPo battery).
- The micro-controller (Induino/arduiono or anything else you may choose) should be powered on-board from the same 12V power supply.
- There must be 2 ports for the left and the right motors to attach to the controller.



Arena :- will be disclosed at the time of the event.

Objective 2 (150 points) : Interfacing your controller to your laptop using ROS and using a joystick attached to your laptop to do the above mentioned task.
(You will be awarded full marks for objective 1 even if you completed objective 1 using ROS and a joystick).

To make a robot run through your laptop, you need to interface your controller (arduino/induino) with some software in your computer, for that purpose we are using ROS.

Installing Ubuntu 14.04 and ROS Indigo:-

- ROS only works with Ubuntu, thus first you need to install Ubuntu as an operating system in your laptop. Take the help of your seniors for this. You can find the Ubuntu 14.04 ISO file in the IITG software repository.
- After installing ubuntu you need to set up proxy in your system as shown here.
- Then you should install ROS following this document.

Some of you might face problems in installing ROS, thus there will be a session on this in early January, dates shall be given later.

After installing ROS, it is of utmost importance to read and do the intial 8 ROS tutorials

Interfacing a joystick:-

- Install the joy package by typing 'sudo apt-get install ros-indigo-joy' in your terminal.
- Then you need to configure the joystick, details are here.

After you get your joystick up and running, you will be using this input data from the joystick to maneuver FAIR around the maze. This kind of manual control is called Teleoperation.

Interfacing the Arduino/Induino:-

- Install the package rosserial-arduino by typing ' sudo apt-get install ros-indigo-rosserial-arduino'.
- Then follow this document for further steps.
- After you have successfully configured arduino, do the initial 3 tutorials from here.

You need to write your own code to map the input joystick data into commands that can be given to the arduino to drive the motors.

At the end the task is similar to the above task, you just need to walk FAIR through a maze as shown above. But this time using a joystick configured through ROS.

Objective 3 (150 points) : Interfacing Kinect 2.0 with ROS to get Point Cloud data. Your task is to display the normal vector arrows of the planes in the same colors as points belonging to that plane.

Steps to interface the Kinect-

- Install Ubuntu 14.04 and ROS indigo (done in the previous steps).
- Install the drivers for Kinect version 2.0.
- A detailed description of the steps involved can be found here.

After successfully installing the drivers power up the kinect and connect to your laptop using the USB cable. Now type ' rostopic list ' in your terminal and you must see a list of topics that are being published by the kinect. Basically the kinect sensor publishes, 2D images (as done by a typical camera), Disparity/depth images (you might want to read more about these here and here), and our software then creates a Point Cloud from this data.

By now you must have done the initial 8 tutorials from ROS tutorials page. If not there is no point in going forward, please do the tutorials first.

You would need a software to visualize this incoming data, RVIZ does that for you, this software would already be installed in your system as it is a part of the ROS system.
To familiarize yourself with the ROS visualization tool (rviz), please follow the given tutorial. We use Plane Segmentation of a 3D point cloud as a motivational example to get you familiar with how ROS infrastructure might be used in real world applications.

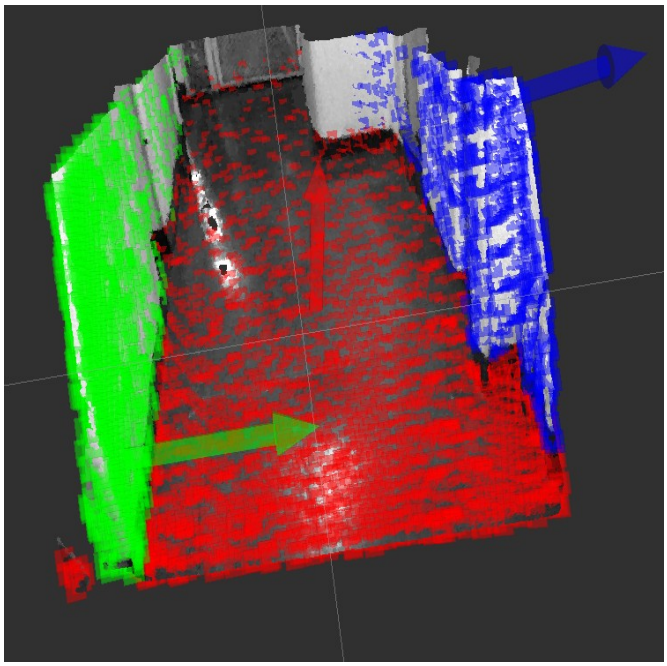
Please find and download the plane segmentation tutorial package here. Download the package in your workspace's src directory. To run the demo:

- Run 'catkin make' in a terminal, with your new workspace as your working directory, and make sure the package compiles without any errors. Run 'source ./devel/setup.sh'
- Type 'roscore' to run a ROS master in the background.
- Now type ' roslaunch plane_segmentation_tutorial sample.launch' to launch the rviz and plane segmentation binary.
- In another terminal, run ' rosbag play -l -r 0.01 2013-02-21-17-59-01.bag ' after changing the working directory to [yourpath]/plane_segmentation_tutorial/data. This plays a pre-recorded data file ("bag file") which provides the point cloud to the plane segmentation.
  A bag file is a collection of sensor data, which allows us to test our softwares even without an actual sensor, you might want to call bag files as vistual sensors.

The demo should show a grayscale pointcloud and overlaid colored points (red, green

or blue) indicating which points belong to which of the detected planes. This is displayed inside the rviz window. You may need to adjust the viewpoint.

Now your task will be to create normal vectors to these planes shown in red, green and blue and display them in the visualizer in the same colour as the plane.



The source file you will be updating is in plane segmentation tutorial/src directory. This code is pretty easy to implement, data points on a plane are given to you, you just need to find a normal vector to the planes and display it. Remember the vector and 3D geometry lessons from JEE times?

Team :-

- Each team can have maximum of 4 members.
- All the members of the single team must be from same hostel.
- All the team members must be bonafide student of IIT Guwahati.
- Only 1 team from each hostel is allowed to participate for this event.

General Rules :-

- If team performs any act against the spirit of gameplay, team will be disqualified immediately.
- All the dimensions of the arena can have +-5% margin.
- Organizers will provide the robot platform and power supply of 12V
- Any changes in the above mentioned rule will be notified through mail.
- The decision given by the organizer will be final. However teams are encouraged to ask questions.

Contact Details:-

Rishabh Jangir
Secretary, Robotics Club
+91-8011036734
r.jangir@iitg.ac.in

Raghuram Krishnaswami
Project Manager, Robotics Club
+91-8011034469
k.raghuram@iitg.ac.in