DOCUMENTATION

for

UNDERWATER IMAGE PROCESSING

Project under Robotics Club, IIT Kanpur

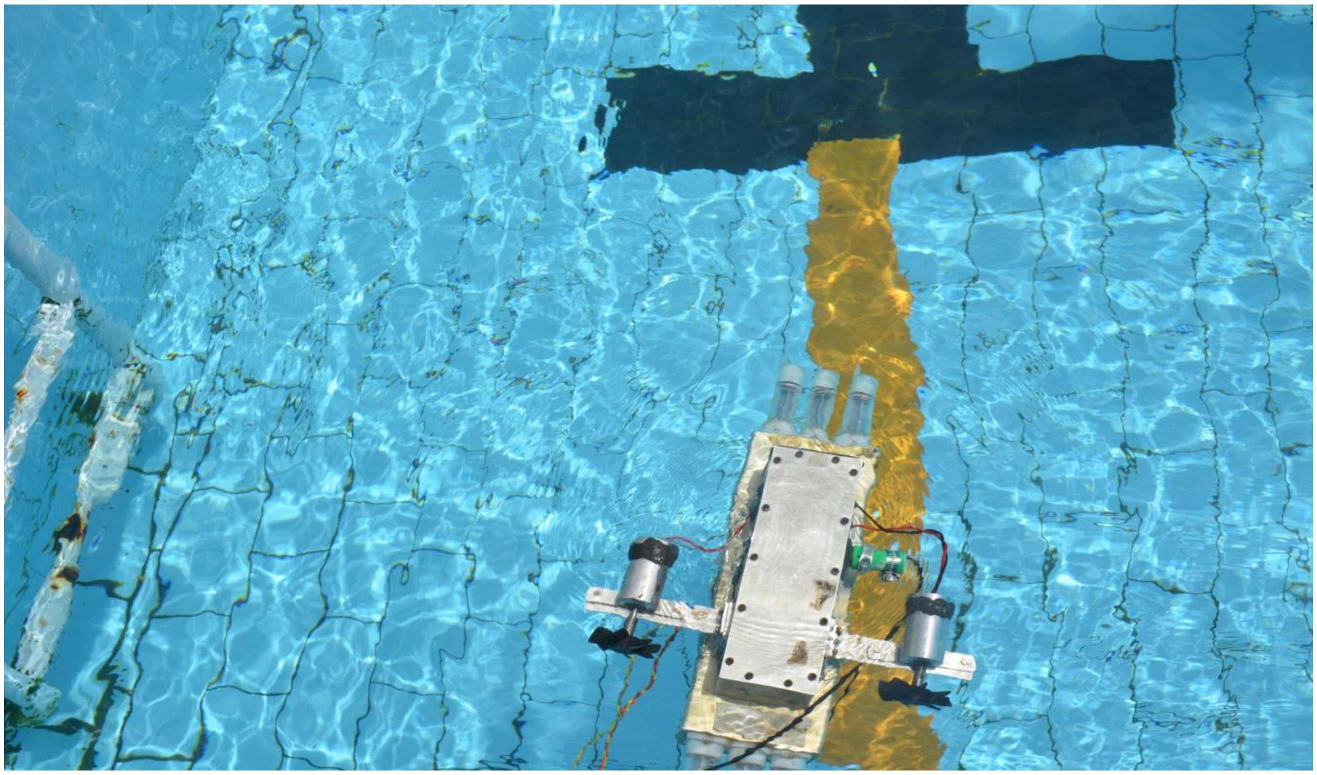
TEAM

ArchitTripathi ***Arpit Agarwal*** ***Harkirat Singh Behl*** ***Himanshu Shukla***

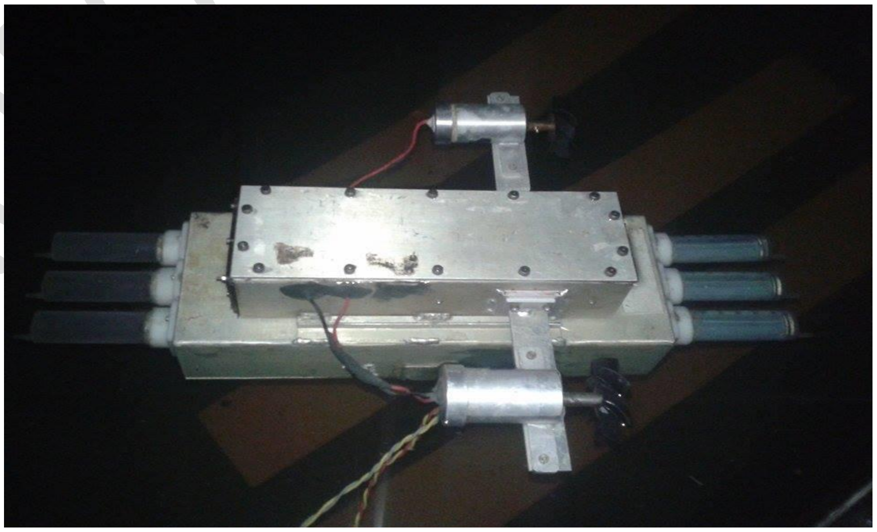
KapilSethi ***NamanSogani*** ***Sumit Kumar*** ***VigneshMuthukumar***

Indian Institute of Technology, Kanpur, India

*students.**iitk**.ac.in/robo**club***



CONTENTS



Abstract

This paper contains the description of the project done under Indian Institute of Technology Kanpur, Robotics Club.

Background: **The project was started with a prior motivation of participating in the AUVSI RoboSub Annual Meet held at San Diego, U.S.A. Another basic aim was to bring Robotics Club IITK at another level in the field of underwater automation and control.**

Methods and Result: **We started on with learning of various methods of processing the image and detection of lines like histogram equalizations, blurs and other various operations using Opencv Library. This further describes how we mended the machine developed by our seniors a year back according to our needs and the problems faced and solutions to them. Also there is a description of electronics portion such as the micro-processors, processors, motor drivers power backup and convertors etc. Along with this the programming part and the algorithms used and the experimental analysis of the machine has been briefly described. This machine perfectly follows the line of the blaze orange colour.**

Introduction

The competition is a platform for students to display their skills in underwater robotics and build a connection with industries working along similar verticals. Since the competition demanded for designing and manufacturing of an unmanned autonomous underwater vehicle that can perform predefined tasks. This draws upon expertise of the areas of engineering provided by multifaceted team. So, the preparation was initiated with a summer project .Two teams were made –

- AUV IP
- AUV HULL DESIGN AND TORPEDO FIRING.

AUV IP team focusses on the image processing, electronics and software part of the entire machine.
The aim of this summer project is underwater blaze orange coloured line following. We modified a machine built a year ago in summer project and attached an extra section camera casing containing a logitech C270 webcam facing downward for recording video. We then process the incoming video and extract information from it, that is, the angle of deviation of machine with respect to the blaze orange coloured strip. Then after calculations, as command was sent to the arduino which accordingly directed the motors attached to the propellers.

The paper describes various design features including mechanical, electrical and programming aspects.

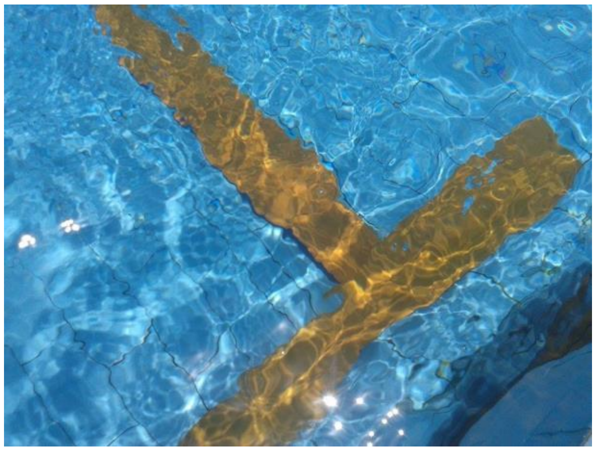


Fig. 1: Orange blaze colored strips underwater

1 - MECHANICAL SYSTEMS

1.1- BASIC STRUCTURE

The machine consists of 3 separable parts -

- The upper chamber containing all the electronic components of machine.
- The middle chamber containing motors for driving the two thrusters (ballast mechanism).
- The lowest part, that is, the camera casing containing logitech webcam C270.

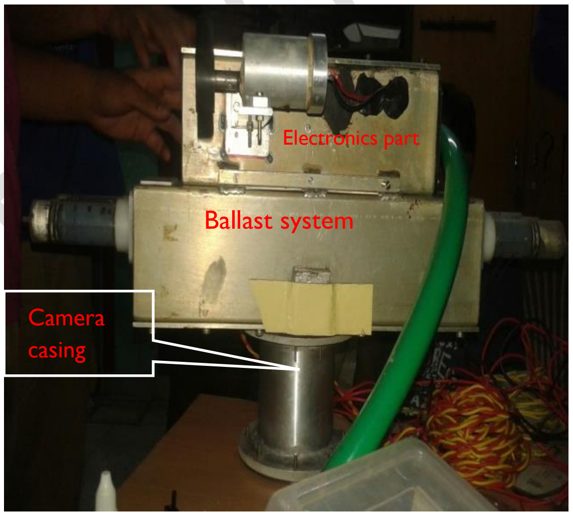


Fig. 2: Basic structure

The upper and lower chambers are cuboidal boxes made up of 4mm thick chromium plated aluminium metal sheet in such a way that they can be attached to each other. Both boxes have rectangular lids. The upper box has it on its top surface while the other box has it on the bottom surface. The camera casing is attached with the lid of middle chamber. The middle chamber has six holes with plastic syringes coming out of them. The syringes have plastic flanges around them to give them support to handle the large pressures underwater. The upper box has a polycarbonate window in the front and one clamp on either sides to attach the propellers.

In order to keep machine in stable equilibrium the centre of buoyancy and centre of mass need to be on the same vertical line. Any tilt of the bot produces sideward shift of the centre of buoyancy and this produces counter torque which stabilise the bot again. So, while designing the body and adding trimming weights it has been tried to increase the separation of centre of buoyancy and centre of mass to the maximum and also to keep them in same vertical line by maintaining a symmetrical design as much as possible .

1.2- BALLAST MECHANISM

The principle of buoyancy comes here. The ballast system provides a variable buoyancy to the machine by sucking in and releasing out water. Here a total of six, 60 ml syringes have been used, three on the front and three at the back. The pistons of the triplets are connected to each other inside the box and move together by virtue of acme screw - drive mechanism. It consists of a screwed rod attached to the rod connecting the three pistons of a triplet. The screwed rod is moved linearly using a cylinder which has screw threads on the inner surface and gear teeth on the outer surface. The cylinder is bound between two clamps and free two move. A 100 rpm motor rotates this cylinder which in turn moves the pistons linearly. We have to replace one of the motor because the screw connecting the motor shaft and gear broke suddenly during a testing of machine.

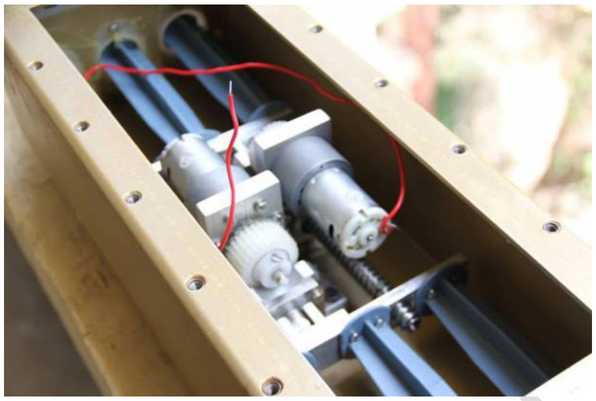


Fig. 3: Ballast system

Opening the piston triplets equally would cause the AUV to sink in straight, while variably opening the triplets can be used to change the pitch of the AUV. These motors were controlled with a 12V lead-acid battery.

Fig. 4: High torque motor 100 rpm motor used in ballistic mechanism

1.3- PROPELLERS

We changed both the propellers of the machine along with the motors driving them. Large surface area of propellers provides large torque. But at the same time it increases the drag force. So thinner blades are better for higher top speeds while larger, flatter blades are essential for better acceleration. As the objective was not to maximize the speed but to improve handling, we have used 7 blade propeller fans (3 inches diameter) that are used in CPU exhaust systems. We have used 600 rpm high torque motors instead of 300 rpm high torque motors due to higher mass of the system. Using the propellers the bot can achieve forward backward (surge motion) and yaw motion.

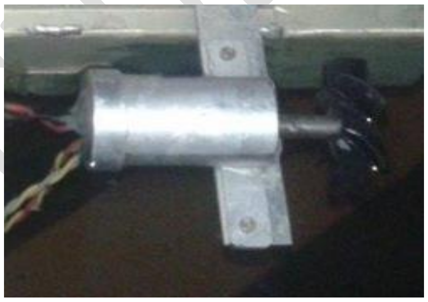


Fig. 5: propeller with its waterproof casing

1.4-CAMERA CASING

To make the camera water proof we prepared its casing using aluminium and acrylic sheet. .
The camera casing has a cylindrical structure with inner diameter of 90 mm, outer of 96 mm and height of 10 cm as the dimension of our logitech camera is itself 70*15*30 cubic mm. One sided of it was welded with an aluminium disk of diameter 146 mm and thickness 8 mm with a groove made in it of diameter slightly less than 90 mm done on a lathe machine in tinkering lab so as to properly fit the cylindrical part. The other side consisted of an aluminium disk of

I.D=90mm and O.D.=146mm along with an acrylic disk fitted over it of diameter 146 mm but thickness 3 mm.
An outlet was made in the cylindrical part in order to eject camera cable. The outlet was an aluminium pipe of O.D. 1 inch, thickness of 2 mm and length of 5cm.

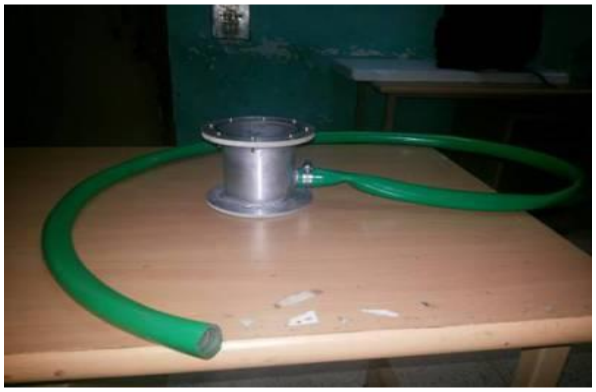


Fig. 6: Camera Casing

1.5- WATER-PROOFING

Water-proofing of machine is a big task for an underwater machine and the machine we started with was too in bad condition. We made some more holes in the original machine for the tethering mechanism i.e., for taking out different cables of the camera, arduino and other electronic components. The leakages from the lid, holes and polycarbonate window of the upper (electronics chamber) box were ceased by applying a layer of m-seal on its periphery.

The leakage through lid was stopped by replacing the o-ring of electronics chamber. For this we had cut o-ring symmetrically and carefully ensuring no gaps in between. The water proofing around these holes and around plastic flanges has been done using m-seal, araldite, feviquick and hot glue. The propeller motors have been encased in aluminium casings especially made for them for waterproofing purposes. Proper sealing was done using m-seal, feviquick and bondtite so that there were no gaps left.

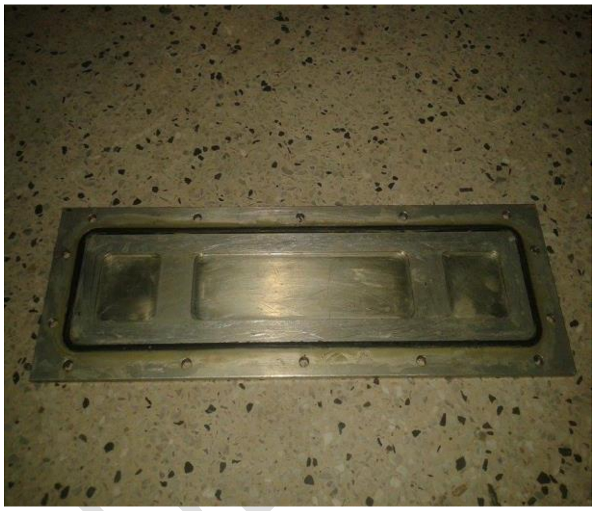


Fig. 7: O-ring in the lid of the electronics box

1.6- WEIGHT BALANCING

We have to place dead weights in order to make the machine float in water. Since , we used the machine in inverted position as it was used previously, so all the calculations of dead weight were made again emphasizing that Centre of Buoyancy(COB) remains below Centre of Gravity(COG) that too in same vertical line. We placed dead weights on the inner walls of the middle chamber and also on its lid. The total amount of these dead weights was 3 kg. We also added dead weights of 225 g in the camera casing in order to make it neutrally buoyant. These dead weights were attached using double sided tapes (DST).

2 – ELECTRONICS

Electronics is a very important part in any autonomous system as it forms the nervous system hence it was important to use the best thing as per the availability for the machine.

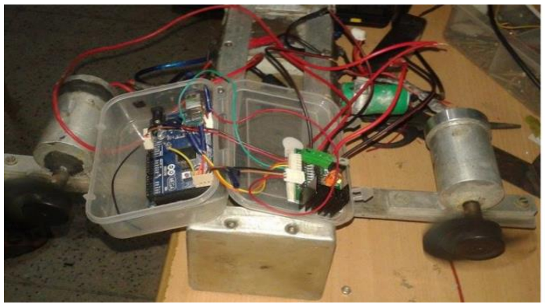


Fig. 8: View of the Electronic Box

2.1- MICROCONTROLLER

We used Arduino Uno as per the availability conditions.
Specifications of Arduino Uno:

Microcontroller	**ATmega328**
Operating Voltage	5V
Input Voltage(recommended)	7-12V
Input Voltage(limits)	6-20V
Digital I/O Pins	14(of which 6 provide PWM Output)
Analog Input Pins	6
DC Current per I/O Pins	40mA
DC Current for 3.3v Pin	50mA
Flash Memory	32 KB(ATmega328) of which 0.5 KB used by boot loader
SRAM	2 KB(ATmega328)
EEPROM	1 KB(ATmega328)
Clock Speed	16MHz

Table 1

Fig. 9: Arduino MEGA 2560

2.2- MOTOR DRIVER

We decide to use a single motor driver as we have to control only two motors attached with propels with Arduino. The motor driver has a current rating of 20 A. This takes into account the current rating of the two motors which control the propellers (7.5 A. each).
The motor driver was not working properly in the beginning due to some problems with the soldering of its PCB but at the end it was corrected.

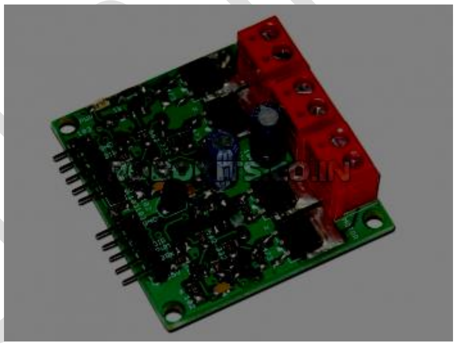


Fig. 10: Motor Driver RKI 1341

Features

• Simple connectivity to IO pins of any MCU.
• Compatible with motors rated up to 18V
• Can easily deliver 20A of current during normal operation
• Braking feature included without affecting the performance of an MCU

Applications

• Simple DC motor applications that require forward and backward driving of motors
• DC motor applications requiring speed control via PWM input
• Halting or braking a DC motor during operation

Electrical Characteristics

Input Voltage: 7V minimum to 18V maximum

Continuous Current (< 1seconds) ~ 20A
Continuous Current (< 10seconds) ~ 10A
Continuous Current (> 10seconds) ~ 5A (without heat sink on MOSFETS)
Absolute Maximum Peak Current ~ 50A
No short circuit protection on output of the driver.

2.3- ODROID AND CONVERTOR

At first it was decided to make a completely autonomous machine using a single-board processor Odroid X2+. We bought a convertor to convert 12V DC from battery to 5V DC required to power the Odroid. But all of a sudden the jack of Odroid through which power is supplied went out of order. It couldn't be corrected and we didn't have that much time left so as to order a new one. Finally, we have to use laptop in place of Odroid.

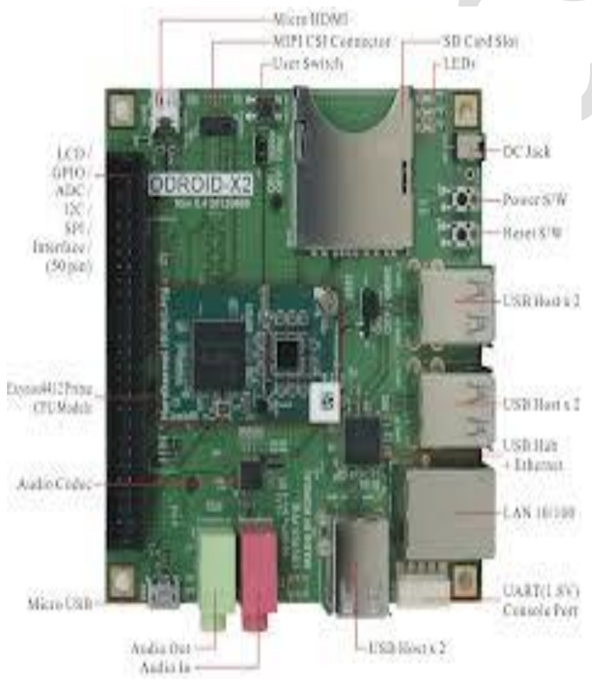


Fig. 11: Odroid X2+ Processor

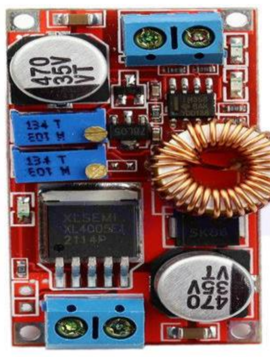
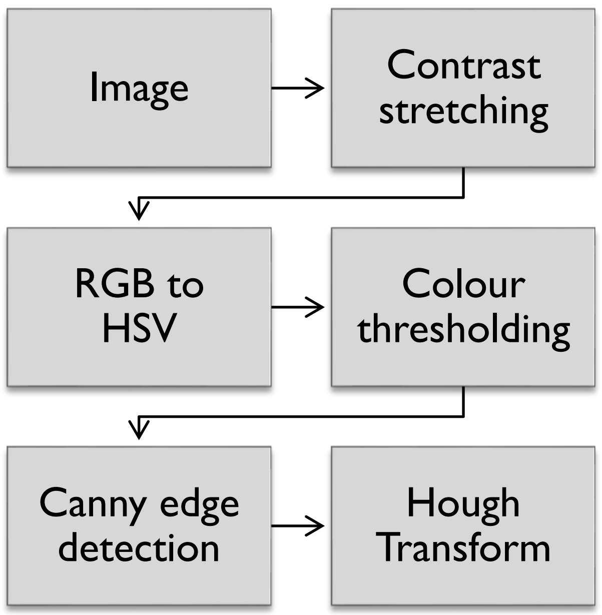


Fig 12: 12V to 5V variable current Convertor

2.4- POWER

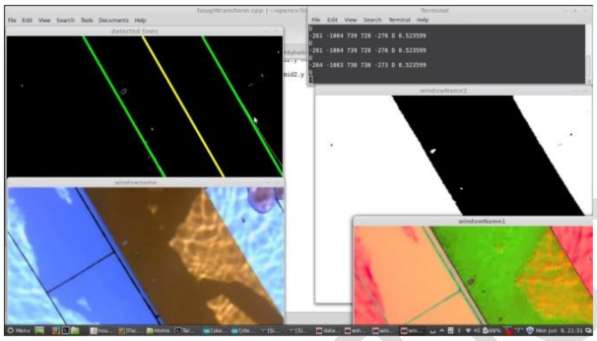
We have to power two thrusters and two propellers. Since, the motors used for propellers were 600 rpm high torque motors, so we decided to use 11.1V 5000 mAh lithium polymer battery. For driving the two thrusters a single 12V lead-acid battery was used .The arduino board was given constant 5V power by laptop.



Flowchart 1

Fig. 13: Li ion battery 3 Cell, 11.1V, 5000mAh

3 – PROGRAMMING



```
U
-261 -1004 739 728 -276 D 0.523599
U
-261 -1004 739 728 -276 D 0.523599
U
-264 -1003 736 738 -273 D 0.523599
U
```

Fig. 14: Execution of Codes

3.1- OPENCV CODE

Image processing was done in UBUNTU 14.04 LTS operating system in our laptops in gedit / emacs text editor using opencv library. Ubuntu is a free-source operating system software

We used UBUNTU rather than WINDOWS because code developing is more compatible and easy in UBUNTU. Also Odroid has KUBUNTU (an official derivative of UBUNTU) installed in it as the OS. We used opencv library for image processing rather than other options like MATLAB, SCILAB, etc. because these are quiet heavy softwares which will not be effective in Odroid. Also, the functions in Opencv are quiet basic and ideal for beginners.

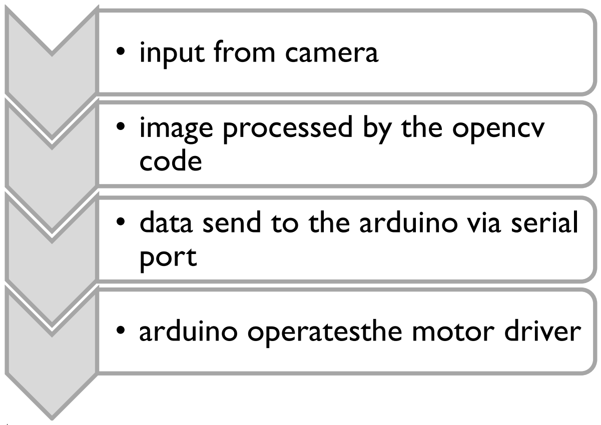
We learnt and used codes of

- image conversion from one form to another
- image thresholding
- Colour thresholding
- Contrast stretching
- canny edge detection
- morphological operations like erosion, dilation, opening and closing
- Hough line transformation

We are now able to calculate the angle of deviation of bot w.r.t. its path, i.e., the blaze orange strip.

3.2- SERIAL COMMUNICATION

The arduino will take input from the code of line detection and then accordingly vary the PWM (Pulse Width Modulation) of the two motors attached with propellers. So, it was important to establish real-time communication between them. It was made possible by using a serial communication library in the code which is available on github.


input from camera
image processed by the opencv code
data send to the arduino via serial port
arduino operatesthe motor driver

Flowchart 2

3.3- PID CONTROLLER

PID (Proportional, Integral, and Derivative) control is a widely-used method to achieve and maintain a process set point. The PID control equation may be expressed in various ways, but a general formulation is:

$$Drive = kP * Error + kI * \sum Error + kD * dP/dT$$

where Error is the difference between the current value of the process variable (temperature, speed, position) and the desired set point, usually written as

$$Error = (Value - Setpoint)$$

∑Error is the summation of previous Error values;

dP/dT is the time rate of change of the process variable being controlled, or of the error itself.

The proportional coefficient kP, the integral coefficient kI, and the derivative coefficient kD are *gain* coefficients which *tune* the PID equation to the particular process being controlled. Drive is the total control effort (often a voltage or current) applied to actuators (heater, motor, and valve) to achieve and hold the set point.

After successfully establishing serial communication, the next task was of PID computation.

The algorithm of PID computation used was-

PID:

$$Error = Setpoint - Actual$$

$$Integral = Integral + (Error * dt)$$

$$Derivative = \frac{Error - Previous\ error}{dt}$$

$$Drive = (Error * kP) + (Integral * kI) + (Derivative * kD)$$

$$Previous\ Error = Error\ wait(dt)$$

$$GOTO\ PID$$

A flow chart for the PID algorithm has been given below:

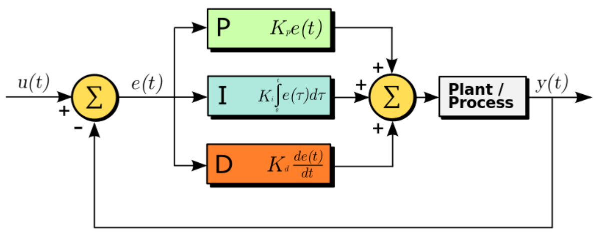
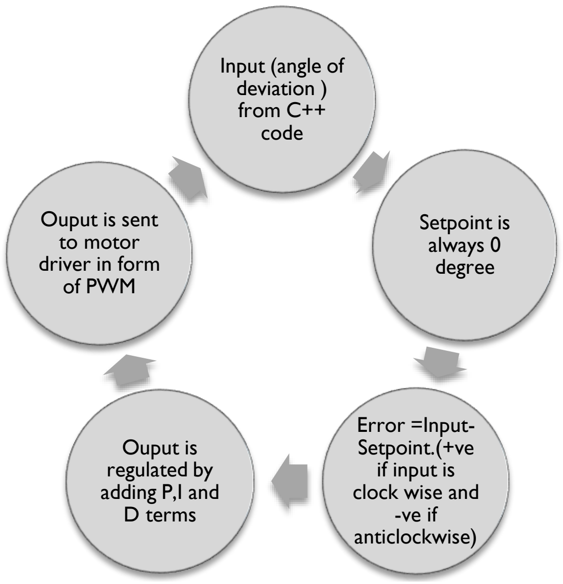


Fig. 15: Flowchart of PID Algorithm

Initially, we incorporated our PID algorithm in arduino code and tried to plot a graph of angle of deviation or error v/s time in PROCESSING software but since arduino was communicating with our code of line detection, its serial port was already busy and this communication couldn't be established. Then, we decided to incorporate the pid algorithm in our line detection code.

Then we sent the drive value to arduino which then accordingly control the motors. Also, we saved a list of 3 values, i.e., time, error/angle of deviation and drive value in a txt file.

This txt file was then fed into a PID tuner named as DOTX PID TUNER. It was used to calculate the required values of kp, ki and kd of our PID algorithm. Once we got those values, our code was final.

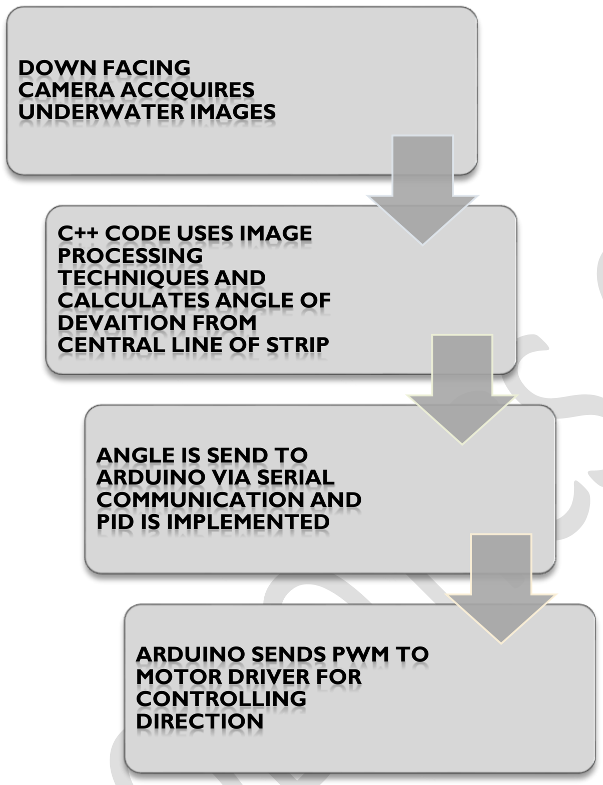


Input (angle of deviation) from C++ code
Setpoint is always 0 degree
Error =Input-Setpoint.(+ve if input is clock wise and -ve if anticlockwise)
Ouput is regulated by adding P,I and D terms
Ouput is sent to motor driver in form of PWM

Flowchart 3

3.4- ARDUINO CODE

An arduino code was written to control the two motors attached to the propellers via a motor driver.
It took drive value from line detection code and then accordingly varies the PWM of the motors such that the net effect is to turn the robot in such a way to decrease the angle of deviation.


DOWN FACING CAMERA ACCQUIRES UNDERWATER IMAGES
C++ CODE USES IMAGE PROCESSING TECHNIQUES AND CALCULATES ANGLE OF DEVAITION FROM CENTRAL LINE OF STRIP
ANGLE IS SEND TO ARDUINO VIA SERIAL COMMUNICATION AND PID IS IMPLEMENTED
ARDUINO SENDS PWM TO MOTOR DRIVER FOR CONTROLLING DIRECTION

Flowchart 4

4 -FUTURE STRATEGIES

- Presently the testing hours for the bot have been very less. So focus would lie on making the bot's motions more robust and calculated. Experiments need to be done on how the bot varies its depth at various levels of water in the syringes and also on how the bot responds to various adjustments of propellers.
- Secondly, implementing a robust sensory system would be the aim which might include IMU, SONAR module, depth sensors, DVL, hydrophones and water sensors. This would enable much accurate obstacle detection and avoidance.
- Then split line following can be done in which there will be more than one stripe that too in a random manner.

5 -ACKNOWLEDGEMENTS

First of all the team would like to thank itself for wholehearted dedication and hard work. It was the team's collaborative effort that made all this possible.
Next to be thanked are the coordinators of the robotics club for providing us such an opportunity and helping us with all the difficulties.
The team also thanks ***Arpit Agrawal*** and ***Abhishek Attal*** for being close and supporting the team in hard times.
Special thanks to professor, swimming pool authorities and pool staff who all made real testing of the AUV in swimming pool possible.
At last the team thanks IIT Kanpur for providing such an excellent environment for doing enjoyable research.

6 -REFERENCES

- Learning Opencv by Gary Bradski and Adrian Kaehler
- Practical Opencv by Samarth Bramhbhatt

- Fundamentals of computer vision by Mubarak Shah
- http://en.wikipedia.org/wiki/PID_controller
- https://www.youtube.com/watch?v=XfAt6hNV8XM
- http://www.arduino.cc/
- http://opencv.org/
- http://www.straightlinecontrol.com/pid_algorithms.html
- http://szeliski.org/book/