PROJECT REPORT ON

UNMANNED AQUATIC VEHICLE ADAPTIVE CONTROL SYSTEM WITH APPLICATION FOR UNDERWATER HUMAN DETECTION

in partial fulfilment of the requirements for the award of B.Tech Degree in Electronics and Communication Engineering

Submitted by,

ASHIK. P (TKM15EC030)

M. ARAVIND (TKM15EC070)

ROMAL ABRAHAM (TKM15EC097)

SUDHEESH. K (LTKM15EC132)

under the guidance of

Prof. SHAFI MN

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

T K M COLLEGE OF ENGINEERING

KOLLAM 691005

2019

Thangal Kunju Musaliar College Of Engineering
Kollam, Kerala, 691005

Department Of Electronics & Communication Engineering

CERTIFICATE

This is to certify that the project report entitled

UNMANNED AQUATIC VEHICLE ADAPTIVE CONTROL SYSTEM WITH APPLICATION FOR UNDERWATER HUMAN DETECTION

is a bonafide account of the report of the work by

ASHIK. P (TKM15EC030)
M. ARAVIND (TKM15EC070)
ROMAL ABRAHAM (TKM15EC097)
SUDHEESH. K (LTKM15EC132)

in fulfilment of the requirement for the award of Bachelor of Technology in Electronics and Communication Engineering of APJ Abdul Kalam Technological University during the academic year 2018-2019.

Prof. SHAFI MN	Prof. ABID HUSSAIN M	Dr. SHEEBA O
Assistant Professor,	Associate Professor,	Professor and Head
Dept of ECE	Dept of ECE	Dept of ECE
Project Guide	Project Co-ordinator	

DECLARATON

We hereby declare that the project entitled "**UNMANNED AQUATIC VEHICLE ADAPTIVE CONTROL SYSTEM WITH APPLICATION FOR UNDERWATER HUMAN DETECTION**" was carried out and written by us under the guidance of Prof. SHAFI MN, Professor, Department of Electronics and Communication, T. K. M. College of Engineering, Kollam. This work has not been previously formed the basis for the award of any degree or diploma or certificate nor has been submitted elsewhere for the award of any degree or diploma.

KOLLAM

10-06-2019

ASHIK. P

M. ARAVIND

ROMAL ABRAHAM

SUDHEESH. K

ACKNOWLEDGEMENT

It is with great pleasure and pride that we present this report before you. At this moment of triumph, it would be unfair to neglect all those who helped us in the successful completion of this report.

We place on record and gratefully acknowledge the encouragement, supervision, suggestions and inspired guidance offered by our guide **Prof. SHAFI MN**, Assistant Professor, Department of ECE, T. K. M College of Engineering, Kollam, **Prof. ABID HUSSAIN M**, project coordinator, and **DR. SHEEBA O**, Professor and Head of the Department, in helping bringing this report to a successful completion with their valuable guidance and suggestions. We would also like to thank our friends from Mechanical Engineering Department for helping us with the prototype modelling and underwater testing.

We are thankful to all faculty members of the Department of Electronics and Communication Engineering and our friends for the whole-hearted support.

ASHIK. P (TKM15EC030)

M. ARAVIND (TKM15EC070)

ROMAL ABRAHAM (TKM15EC097)

SUDHEESH. K (LTKM15EC132)

ABSTRACT

Our project discusses the control engineering implementation and testing of an Autonomous Underwater Vehicle (AUV) robot, which travels underwater to detect possible human presence. This robot is expected to be extremely useful in human rescue operations during water accidents. The AUV is deployed above the water surface, near to the approximate target location and it swims underwater, checking for the target (human body), in real time. Once the target is detected, the AUV locks the targets position and move towards the target to verify and make a decision: whether it is a human body or not. After the confirmation, the robot swims back vertically up to the water surface, from where it can be easily noticed by the ground rescue crew. The diver can then dive there, to find the human body. Alternately, an air bag could also be inflated after locking it to the target, which will float back to the surface carrying the target. This technology is very helpful for underwater human detection since we need not have to search the whole water body and hence both time and resources are saved. For more effective operation, many swarm ROV's can work together in real time to cover the whole area in a very short span. Further application of the work can help in maritime and aerial surveillance, underwater archaeology, studying marine species apart from aiding in rescue operations.

Keywords*: Underwater Human detection, Remotely Operated Vehicle (ROV), Autonomous Underwater Vehicle (AUV), Surveillance and Surveying, Safety and Rescue.*

LIST OF ABBREVIATIONS

AUV	Autonomous Underwater Vehicle
ROV	Remotely Operated Vehicle
WHO	World Health Organization
UAAV	Unmanned Aerial and Aquatic Vehicle
GPH	Gallon Per Hour
UAV	Unmanned Aquatic Vehicle
AH	Ampere Hour
PVC	Poly Vinyl Chloride
LAN	Local Area Network
ESC	Electronic Speed Controller
CPU	Central Processing Unit
INR	Indian Rupee
QGC	QGroundControl
MAV	Micro Air Vehicle
CCD	Charge Coupled Device
CMOS	Complementary Metal Oxide Semiconductor
UAPS	Underwater Acoustic Positioning System
ASK	Amplitude Shift Keying
UDP	User Datagram Protocol
API	Application Programming Interface
CNN	Convolutional Neural Network.
USB	Universal Serial Bus
AUV	Autonomous Underwater Vehicle
ROV	Remotely Operated Vehicle

CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1
INTRODUCTION

1.1 MOTIVATION

Underwater human detection is a relatively unexploited field of study under marine technology. According to WHO [1], Drowning is the 3rd leading cause of unintentional injury death worldwide, accounting for 7% of all injury-related deaths, with an estimated 3,60,000 annual drowning deaths worldwide (2018). The main reason for this is that underwater domain is a dangerous and complex environment for human divers. Often, divers have to monitor their own life support systems as they navigate to the work site or while operating dangerous machinery. Military divers have to navigate for extended periods of time without surfacing or without using localization techniques that might give away their positions. Human divers have operated under these harsh conditions for decades with few advancements in technology. In fact, a diver performs the basic task of navigation by aligning the body with a compass and counting leg kicks (i.e., human-oriented dead reckoning).

The scope of the present research work is to improve the existing conventional methods used in search and rescue of drowned human body in water. Traditional procedures incorporating human divers searching underwater for human bodies (surviving or dead) are very inefficient due to visibility issues and the supply oxygen, which increases the delay and complexity in such operations. Hence developing a suitable mechanism to aid this is very much necessary. But research works in this field are advancing at a comparatively low pace, mainly due the difficulty and complexity in imaging and sensing technologies that could be employed under water. One of the main challenges in underwater communication is the inefficiency of Electromagnetic wave communication inside water. Water absorbs electromagnetic radiations and an information could not be easily transmitted across, without suffering from loss. Hence we have to depend on acoustic waves for underwater communication. One such approach is to power a beam of sound energy (ultrasound) at the assumed target location and calculate the absorption of an underwater body by looking at the received power of the signal reflected from the body. The material can then be recognised by suitable training.

1.2 PROBLEM STATEMENT

Traditional procedures incorporating human divers searching underwater for human bodies (surviving or dead) are very inefficient due to visibility issues and the supply oxygen, which increases the delay and complexity in such operations. Hence developing a suitable mechanism to aid this is very much necessary.

But all the existing methods used in this domain have some serious disadvantages. Since the mass of the immersed body is far smaller than the mass of the water body, temperature of the body will level down or up to the water temperature. once the temperature of the ’body’ equals the temperature of the body, infrared does not help. Also, in unstructured environments, current underwater robots are limited in their ability to localize and track human bodies because optical cameras can be rendered useless by the turbidity and murkiness of water. Localizing radio signals also do not propagate well through the water medium, and acoustic positioning systems can be very expensive to deploy. So a technique for underwater human detection must be developed which is economically and technologically feasible.

Due to inefficiency of popular underwater imaging technologies like thermal imaging [2], infrared imaging or laser scanning, the proposed ROV is expected to work using the combined action of an ultrasound detector and an underwater camera. Recognition of human body is achieved with the help of further signal processing and real-time feature extraction using both the underwater camera footage and the reflected sound signal. When the human body is under water, in a few minutes, most of the heat from the body would be dissipated and its nearly impossible to use thermal imaging here [3]. Hence a weighted decision from sound detection and normal imaging is used in the present work.

1.3 OBJECTIVES

This paper proposes the application of Unmanned Aerial and Aqua Vehicles (UAAV) for underwater human rescue operation. The UAAV (a drone) with an Autonomous Underwater Vehicle (AUV) will fly over the water body and land on the water surface. It then deploys the AUV in to the water. AUV swims in to the water searching real time for its target - the human body. Once the target is detected, the AUV locks the targets position and move towards the target to verify and make a decision: whether it is a human body or not. After the confirmation, the robot swims back vertically up to the water surface, from where it can be easily noticed by the ground rescue crew. The diver can then dive there, to find the human body. Alternately, an air bag could also be inflated after locking it to the target, which will float back to the surface carrying the target. This technology is very helpful for underwater human detection since we need not have to search the whole water body and hence both time and resources are saved.

Since thrusters are very much costly, we created thrusters by cutting opening bilge pumps and replacing the impellers with underwater grade propellers, used for making hobby boats. The bilge pumps used are of 1100 GPH capacity and 12V, 3A rating. So we used special Electronic Speed Control Units designed for brushed motors with current rating 30A for controlling the AUV. Power output is taken from a 12V 30AH rickshaw battery. For prototyping weve used Raspberry Pi 3 B+ as the processor which receives camera and sound sensor data and sends the control signals to the ESCs via the relays connected. We have used 4 thrusters, 2 for down motion (We make use of buoyancy to go upwards) and 2 side thrusters for direction change. Additionally 2 more thrusters can be introduced to account for improved stability of the AUV. For chassis, weve used PVC and tightly sealed it to protect the components inside. Since Raspberry Pi 3 is not that much computationally powerful, we connect Pi to a much more powerful computer (a laptop) via a LAN cable (pref. upto 150m length.) for fast processing of input data and faster decision making.

1.4 SCOPE OF THE WORK

The scope of this project is to improve the existing conventional methods used in search and rescue of drowned human body in water. Traditional procedures incorporating human divers searching waterways for human bodies (surviving or dead) are very inefficient due to visibility issues and supply oxygen, which increases the delay and complexity in such operations. Hence developing a suitable mechanism to aid this is necessary. The basic idea is to implement a diver-friendly rescue aiding machine for underwater human detection. The design is targeted at medium sized water bodies like swimming pool or a small lake or a pond. The proposed system uses an AUV mounted with thrusters which can dive into the water to detect the target (human body). The UAAV (drone) with an Autonomous Underwater Vehicle (AUV) will fly over the water body and land on the water surface. It then deploys the AUV in to the water. AUV swims in to the water, while searching for a possible human presence. AUV is supposed to work using the principles of ultrasound imaging, due to inefficiency of other detecting techniques like thermal imaging or laser scanning. Recognition of human body is achieved with the help of signal processing and feature extraction of the received signal after reflection from the target. Optional VR (Virtual Reality) mechanism may be provided to the controller man on the land for further optimized control of the AUV. AUV has a marking mechanism which makes it's location inside water, traceable by the controller on land. AUV then floats back to the water surface, which is detected by the UAAV.

For more effective operation, many swarm AUV's can work together in real time to cover the whole area in a very short span of time using the principles of underwater acoustic communication [4]. Further application of the present research work can aid maritime and aerial surveillance [4], underwater archaeology, study and survey of marine flora and fauna, apart from aiding in human rescue operations.

1.5 EXPECTED RESULTS

But research works in this field are advancing at a comparatively low pace, mainly due the difficulty and complexity in imaging and sensing technologies that could be employed under water. One of the main challenges in underwater communication is the inefficiency of Electromagnetic wave communication inside water. Water absorbs electromagnetic radiations and an information could not be easily transmitted across, without suffering from loss. Hence we have to depend on acoustic waves for underwater communication. One such approach is to power a beam of sound energy (ultrasound) at the assumed target location and calculate the absorption of an underwater body by looking at the received power of the signal reflected from the body. The material can then be recognised by suitable training. Another approach is to focus on a special case of moving object detection under water in which the rate of motion degrades slowly. Since there is also possibility of change in degree of motion for other objects also due to water movement, this method comes with the cost of spending a lot of time.

But all these methods have some serious disadvantages. Since the mass of the immersed body is far smaller than the mass of the water body, temperature of the body will level down or up to the water temperature. once the temperature of the 'body' equals the temperature of the body, infrared does not help. Also, in unstructured environments, current underwater robots are limited in their ability to localize and track human bodies because optical cameras can be rendered useless by the turbidity and murkiness of water. Localizing radio signals also do not propagate well through the water medium, and acoustic positioning systems can be very expensive to deploy. So a technique for underwater human detection must be developed which is economically and technologically feasible.

1.6 THESIS OUTLINE

Introduction section deals with the subject background studies and the motivation that lead us to take up this project work. problem statement, objectives, scope of our work and the expected results are discussed. In literature review, we discuss about the existing research in this domain and what makes our work significant and different. In the system components section, we discuss about the different hardware and software components used in the project in detail. Finally, we present the results and conclusion of our work and it's relevance to the present world. In the Introduction part, we propose the need of this project t some important concepts and requirements behind it. So, in the Literature Review section, we provide our understandings about some of the recent research trends and works done in the field. Then we have the components section, which deals in detail with the different hardware and software sections we've come across, even the ones we could not use in our project, but could benefit future researches.

The challenges and difficulties we faced while carrying out the project work is explained. Future research possibilities based on ongoing research works are given in the future scope section for gaining further insight on the emerging advances in the field of underwater robotics and automation.

CHAPTER 2

LITERATURE REVIEW

2.1 INTRODUCTION

The underwater domain is a dangerous and complex environment for human divers. Often, divers have to monitor their own life support systems as they navigate to the work site or operate dangerous machinery. Military divers have to navigate for extended periods of time without surfacing or without using localization techniques that might give away their positions. Human divers have operated under these harsh conditions for decades with few advancements in technology. In fact, a diver performs the basic task of navigation by aligning the body with a compass and counting leg kicks (i.e., human-oriented dead-reckoning).

The scope of this project is to improve the existing conventional methods used in search and rescue of drowned human body in water. Traditional procedures incorporating human divers searching waterways for human bodies (surviving or dead) are very inefficient due to visibility issues and supply oxygen, which increases the delay and complexity in such operations. Hence developing a suitable mechanism to aid this is necessary.

One approach is to power a beam of sound energy (ultrasound) at the assumed target location and calculate the absorption of an underwater body by looking at the received power of the signal reflected from the body. The material can then be recognised by suitable training.

Another approach is to focus on a special case of moving object detection under water in which the rate of motion degrades slowly. Since there is also possibility of change in degree of motion for other objects also due to water movement, this method comes with the cost of spending a lot of time.

Another approach worth considering is the imaging of the objects under water which may not be always economically feasible. Much image processing and computing is required to extract the required features and make the necessary computing in this case. Imaging could be done either with an underwater camera or with ultrasound imaging, both are prone to high noise.

Since the mass of the immersed body is far smaller than the mass of the water body, temperature of the body will level down or up to the water temperature. once the temperature of the 'body' equals the temperature of the body, infrared does not help.

However, in unstructured environments, underwater robots are limited in their ability to localize and track a human diver at the resolution required to enable diver-robot interactions. Optical cameras can be rendered useless by the turbidity of the water, localizing radio signals do not propagate well through the water medium, and acoustic positioning systems can be expensive to deploy.

Our work in this project aims to find a suitable and foolproof method to devise a system capable of rescuing a drowning human body underwater with accuracy.

2.2 EXISTING SYSTEMS

Fatma et al., [2] is a study to detect and to track the people under the water quickly. The paper focuses on underwater human-robot interaction via biological motion identification. Thresholding, Background Subtraction, Interframe Difference and Foreground Detection methods have been applied to create the silhouette of the people under the water. These methods have been demonstrated on videos. The paper discusses an algorithm for underwater robots to visually detect and track human motion. The objective is to enable human-robot interaction by allowing a robot to follow behind a human moving in (up to) six degrees of freedom. The authors have tried to develop a system to allow a robot to detect, track and follow a scuba diver by using frequency domain detection of biological motion patterns. The motion of the diver relative to the vehicle was then tracked using an Unscented Kalman Filter (UKF), an approach for non-linear estimation.

Sonia et al., [3] deals with the technology for human detection using ultrasonic waves. The approach is based on fuzzy rules that are extracted from the signal features in time and frequency domains. The performance of the human detector system (classifier) is assessed in terms of accuracy, true positive, and false positive rates. Human detection generally refers to the process of differentiating human beings from other animate or inanimate objects through the use of sensory technology. Human detection is a challenging task and forms a vital component of a range of applications which include human-robot interaction, surveillance and monitoring, search and rescue, and smart rooms. The challenge in realizing such a detector arises due to the facts that human beings are non-rigid in structure and are of various sizes and shapes. They are also dressed-up in different types of clothing and are non-stationary. Research on human detection concentrates mainly on vision-based techniques. However, there are many applications where vision-based techniques are not desirable. This situation is very similar to the underwater.

Sameer et al., [4] describes the new developments happening in the area wireless acoustic communications. Normally the underwater vehicles are remained isolated from above water, to preserve the stealth advantage that operations within it provide. Development of robust and stealthy inter-platform communications/network solutions has changed the scenario and has become the critical enabling technology for various future requirements of the navies across the globe. The main challenges of underwater communication as mentioned in this paper as

High transmission loss, Ray bending due to Snells law, High ambient noise, Multipath propagation, propagation delay. Since radio waves are highly attenuated in underwater, acoustic communication methods are preferred. The recent trends in communication like coherent communications, orthogonal frequency domain modulation(OFDM) are also discussed in this paper.

Ted S et al., [5] In order for underwater robots to communicate with land and air-based robots on an equal basis, high speed communications is required. If the robots are not to be tethered then wireless communications is the only possibility. Sonar communications is too slow. Unfortunately, radio waves are rapidly attenuated under water due to phenomena such as skin depth. This paper gives the methods for extending the underwater radio communication. Physical layer communications can be accomplished by a number of technologies, including wire-line, and wireless methods. Wireline communications could be established with all submarines, but the restrictions on mobility such as entanglement suggest wireless communications should be accommodated at some point, whether at the end of a wire, or from a communications device at the surface. Once communication is made to the surface, a surface vessel could relay signals to other swarms outside of the water.

S. Udupa et al., [6] suggested the design for a robot with functionalities for maneuvering, image recognition and depth control. Several simulations were studied and authors proposed a new design with minimal drag and good fluid dynamics.

Y. Jua et al., [7]introduces a new Monostatic-Bistatic Composite system for underwater moving target detection. The data fusion for the monostatic and bistatic was described using the combination processing of imaging method, and the calibration algorithm of this system was discussed. The experiment result showed the observability of the small target in bistatic modes and the better performance of this imaging method used in the composite system. B.

Anwar et al., [8]demonstrated the application of both remotely operated underwater vehicles (ROV) and autonomous underwater vehicles (AUV) equipped with surveillance system in military, scientific research, film making under the water and monitoring underwater industrial structures and underwater network devices.

2.3 PROPOSED SYSTEM

Due to inefficiency of popular underwater imaging technologies like thermal imaging [2], infrared imaging or laser scanning, the proposed ROV is expected to work using the combined action of an ultrasound detector and an underwater camera. Recognition of human body is achieved with the help of further signal processing and real-time feature extraction using both the underwater camera footage and the reflected sound signal. When the human body is under water, in a few minutes, most of the heat from the body would be dissipated and its nearly impossible to use thermal imaging here [3]. Hence a weighted decision from sound detection and normal imaging is used in the present work.

A lot of fluid-dynamic factors must be considered for comparing the existing designs with other design alternatives like cylindrical, elliptical and rectangular, along with Drag and stress calculations. The first ever developed underwater ROV was known as POODLE. The vehicle was mainly used for archaeological research [8]. Since then, till now the ROVs are being widely used both in industry and science for a variety of purposes including exploring hydrothermal vents, inspection of underwater oil derricks, inspection of high radiation area, inspection of sub-sea phenomena, surveying archaeological sites and fixing underwater infrastructure such as cabling and piping. One of their most challenging uses is in deep water search and rescue. The underwater robot system is expensive. As a result,in developing and under developed countries the production and deployment of ROVs are limited. Many of these countries consist of a large number of water bodies and are prone to maritime incidents. Some researchers have been involved in the design and development of low-cost underwater robots [6]. Our primary focus in the work was to reduce the cost to make a suitable ROV to be deployed into developing and under developed countries, for economically feasible underwater human detection. Compared with the work in the developed world, this is very essential as there is no point in developing a system which is not affordable. World class AUV's or ROV's are usually very costly. In 2013 a group of researchers provided a model of a comparatively low cost underwater robot named GUPPIE [8] which costed around 1 lakh INR and we our aim is to develop a system which is much less priced than the existing models.

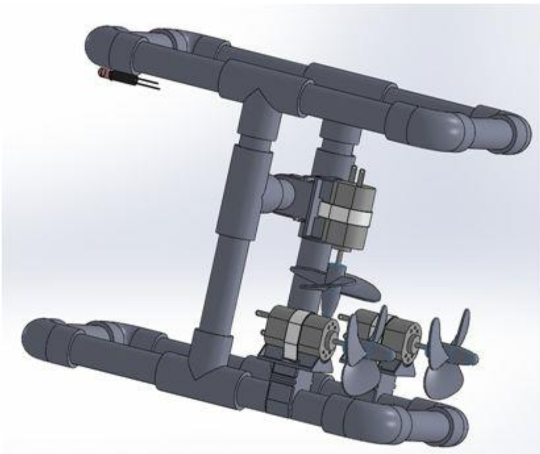
The basic idea is to implement a diver-friendly rescue aiding machine for underwater human detection. The design is targeted at medium sized water bodies like swimming pool or a small lake or a pond. The proposed system uses an AUV mounted with thrusters which can dive into the water to detect the target (human body). The UAAV (drone) with an Autonomous Underwater Vehicle (AUV) will fly over the water body and land on the water surface. It then deploys the AUV in to the water. AUV swims in to the water, while searching for a possible human presence. AUV is supposed to work using the principles of ultrasound imaging, due to inefficiency of other detecting techniques like thermal imaging or laser scanning. Recognition of human body is achieved with the help of signal processing and feature extraction of the received signal after reflection from the target. Optional VR (Virtual Reality) mechanism may be provided to the controller man on the land for further optimized control of the AUV. AUV has a marking mechanism which makes it’s location inside water, traceable by the controller on land. AUV then floats back to the water surface, which is detected by the UAAV. For more effective operation, many swarm AUV’s can work together in real time to cover the whole area in a very short span of time using the principles of underwater acoustic communication [4]. Further application of the present research work can aid maritime and aerial surveillance [4], underwater archaeology, study and survey of marine flora and fauna, apart from aiding in human rescue operations.

Since thrusters are very much costly, we created thrusters by cutting opening bilge pumps and replacing the impellers with underwater grade propellers, used for making hobby boats. The bilge pumps used are of 1100 GPH capacity and 12V, 3A rating. So we used special Electronic Speed Control Units designed for brushed motors with current rating 30A for controlling the AUV. Power output is taken from a 12V 30AH rickshaw battery. For prototyping we’ve used Raspberry Pi 3 B+ as the processor which receives camera and sound sensor data and sends the control signals to the ESCs via the relays connected. We have used 4 thrusters, 2 for down motion (We make use of buoyancy to go upwards) and 2 side thrusters for direction change. Additionally 2 more thrusters can be introduced to account for improved stability of the AUV. For chassis, we’ve used PVC and tightly sealed it to protect the components inside. Since Raspberry Pi 3 is not that much computationally powerful, we connect Pi to a much more powerful computer (a laptop) via a LAN cable (pref. up to 150m length.) for fast processing of input data and faster decision making.

2.4 DEVELOPMENT OF THE MODEL

2.4.1 INITIAL MODEL

PVC sections are connected to each other using L and T joints. After inspection sections are fixed using solvent.



(a)

(b)

Fig 2.1 Prototype of the model

Calculation of Centre Of Mass (COM)

X= (X1+X2+X3)/3

=(1+11+21)/3

=11 cm

Y=(2 X Y1+ Y2)/3

=(2X 5 + 17)/3

= 9 cm

Z=(Z1+ 2 X Z2)/3

=(22+ 2 X33)/3

=29.33 cm

ROV would topple at a distances of 11 cm, 9 cm, 29.33 cm from X,Y and Z axis. During turning even by small radius ROV is not stable. Hence we decided to abandon this design and proceeded to make a new design.

Coordinate	X	Y	Z
Location of centre of mass(cm)	11	9	29.33

Table 2.1 Coordinates of Centre of mass

2.4.2 ANALYSIS USING ANSYS

Simulation was done using ANSYS Version R18.1. Fringe diagrams were obtained by applying pressure from exterior.

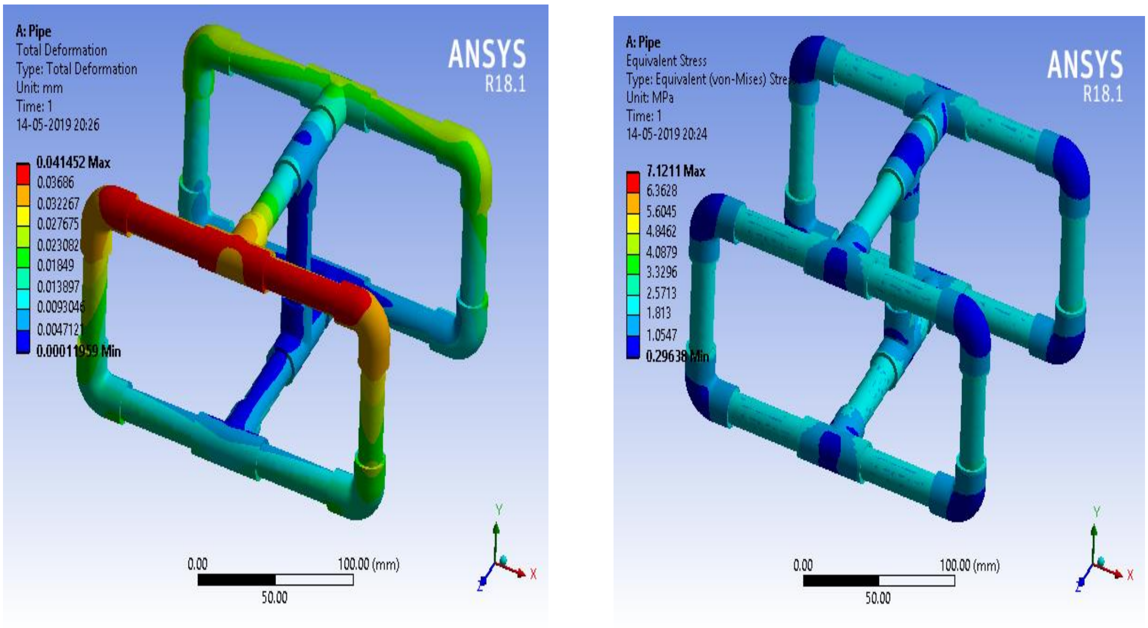
Design is made such a way that ROV can travel upto 40 meters of depth

Pressure at 40m= P0+Pgh

=1*1.01*10^5+ 1000*9.81*40

=4.92*10^5 N/m2

This pressure was applied on ROV frame. Stress and deformation was found out as shown.



No failure is found in simulation. Hence the design is safe.

Fig 2.2 Analysis using Ansys

2.4.3 DRAWBACKS

Even though this model was simple and economically cheap it had a lot of drawbacks. It is unstable as its Centre of Mass (COM) is located high. This caused the ROV to topple upside down. A waterproof chamber was important to keep the electronic components safe. Structurally the surface is weak against impact loads at all ends.

To rectify all these drawbacks and shortcomings a new model was designed.

2.4.4 PRESENT MODEL

This model is made of cylindrical PVC sections of dimensions. Bilge pumps which are majorly used for pumping water was modified and converted to the form of thrusters. The exterior section was cut. On the rotating shaft propeller blades were attached. Two bilge pumps were placed in horizontal direction and two bilge pumps were placed in vertical direction. Directional control is done using vector configuration.

2.4.5 THRUST CALCULATIONS

$F=M(V_2 - V_1)$

M= mass flow rate

F= Thrust in lbF

V_1= Initial velocity

V_2 =Final velocity

ρ = Density of the liquid

Q= Volumetric flow rate = 18.33 GPM =4.163 m^3

D= Diameter of pump= 1.10236 inches =0.02799m

$M= \rho * Q$

$F= \rho * Q (V_2 - V_1)$

$M= \rho * Q* [(Q / (\pi D_2/4))- V_1]$

$V_1= 0$

$F=4* \rho*Q^2/(\pi D_2)$

Substituting Q,D and ρ in the above equation, we get

F = 2.17273N

Since there are two pumps for either motion,

$F= 2*2.17273= 4.3546N$

$V_2=Q/(\pi d^2/4)$, substituting

$V_2=1.866$ m/s

Applying conservation of momentum,

$2* \rho A V_2^2= m*V$

Substituting, we get

$V= 1.072m/s$

This ROV is made of three long cylinders. For a long cylinder coefficient of drag, $C_d=0.82$

$C_d=2*F_d/(\rho u^2 A)$

F_d=drag force

ρ=density of the fluid

u=relative velocity of ROV and fluid (here fluid is assumed to be stationary)

A=area of ROV normal to flow direction

$A= \pi*(.1016)^2/4+2* \pi*(.0762)^2/4$

$=.0171m^2$

$u=1.072m/s$

$\rho=1000kg/m^3$

$C_d=.82$

Substituting in the equation for drag force, we get

$F_d=0.504N$

This is the drag force acting on the ROV

Net force on the ROV=Thrust-Drag

=4.3546-0.504

=3.8506N

Acceleration of the ROV=Thrust/Mass

=3.8506/4

$=0.9626 m/s^2$

Fig 2.3 Prototype of the new model

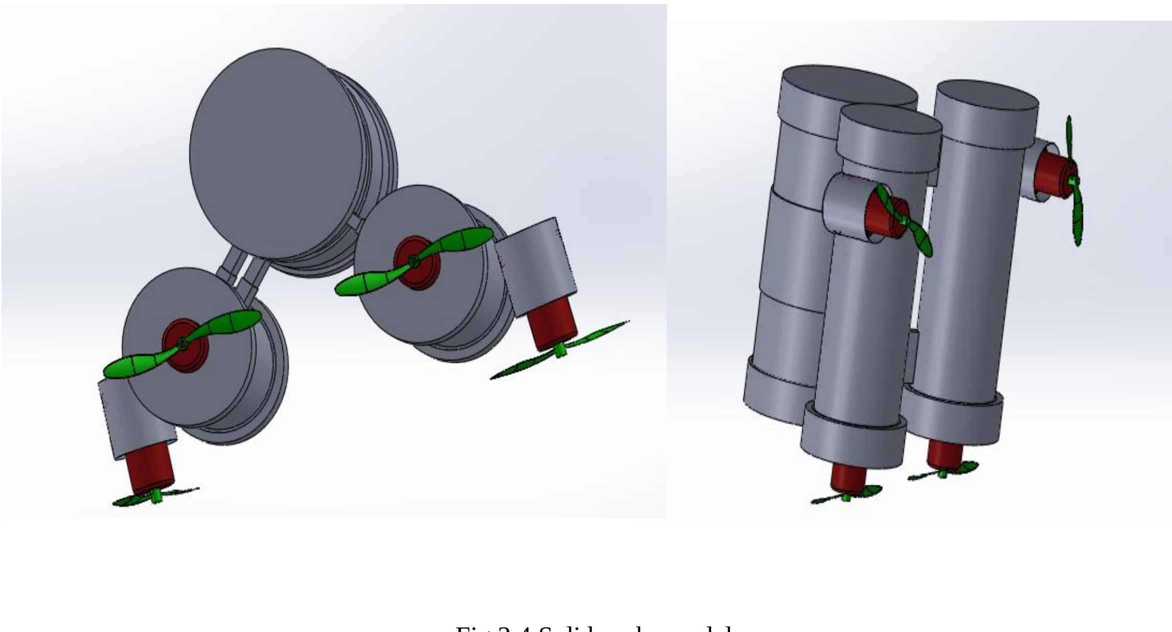


Fig 2.4 Solidworks model

m

2.4.6 ANALYSIS USING ANSYS

Simulation was done using ANSYS Version R18.1. Fringe diagrams were obtained by applying pressure from exterior.

Design is made such a way that ROV can travel upto 40 meters of depth

Pressure at 40m= P_0+Pgh

$=1*1.01*10^5+ 1000*9.81*40$

$=4.92*10^5$ N/m2

This pressure was applied on ROV frame. Stress and deformation was found out as shown.

No failure is found in simulation. Hence the design is safe.

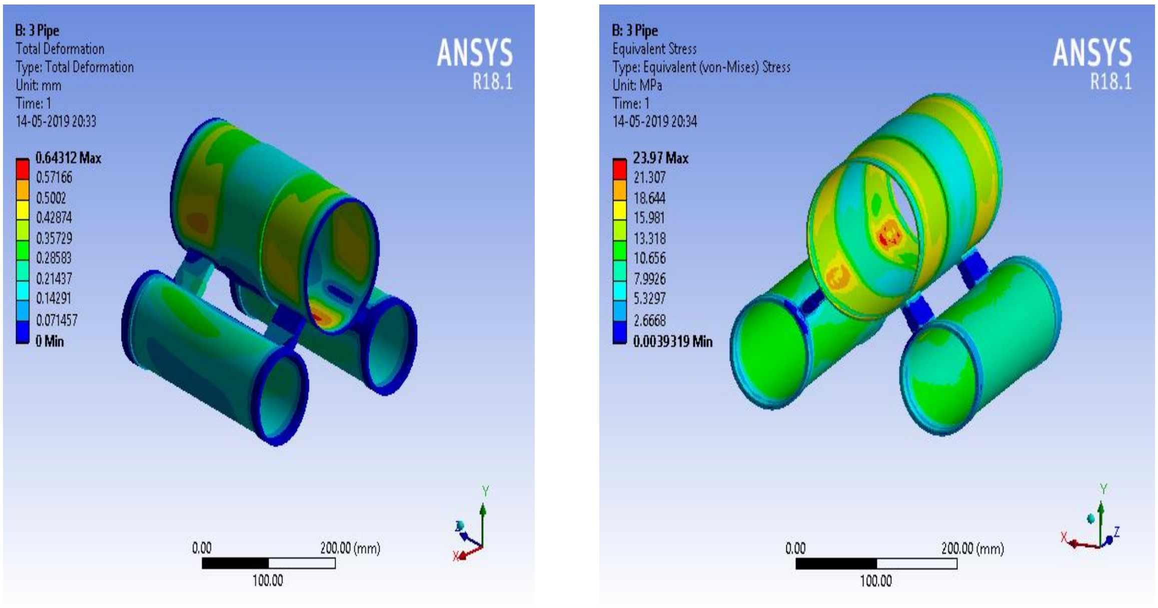


Fig 2.5 Analysis of the model using ANSYS

This model was tested under water to check its functionality. For preparing the model for underwater testing, it was made water tight using M-Seal and silicone. The ends of the PVC pipes were closed using PVC end caps and then sealed using silicone gel. The joints of the three cylinders and the outlets for electrical wires were sealed using M-Seal and then made water tight using silicone.

CHAPTER 3

SYSTEM COMPONENTS

3.1 HARDWARE SECTION

The main hardware components used in the present work are:

1. 1100 GPH bilge pumps, with impeller replaced by underwater propellers (x4)
2. BDESC-S10ERTR Underwater brushed motor ESC. (x4)
3. Suitable PVC for casing
4. LAN cable of suitable operational length
5. Raspberry Pi 3 B+ with ArduSub installed
6. Pixhawk PX4 Autopilot flight controller
7. USB camera (pref. h264 support)
8. Laptop (pref. i5 CPU)
9. Joystick
10. Wires for connection (min 3A current rating)
11. Battery

3.1.1 BLOCK DIAGRAM

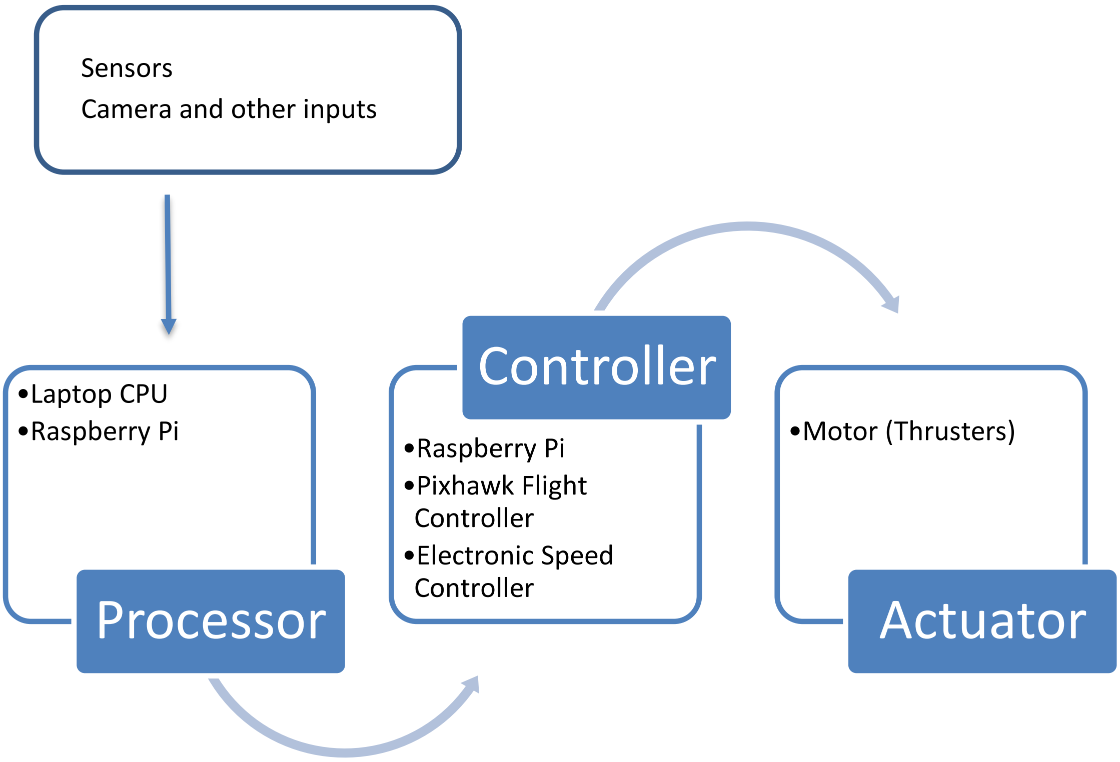


Fig 3.1 Block Diagram

FLOW CHART

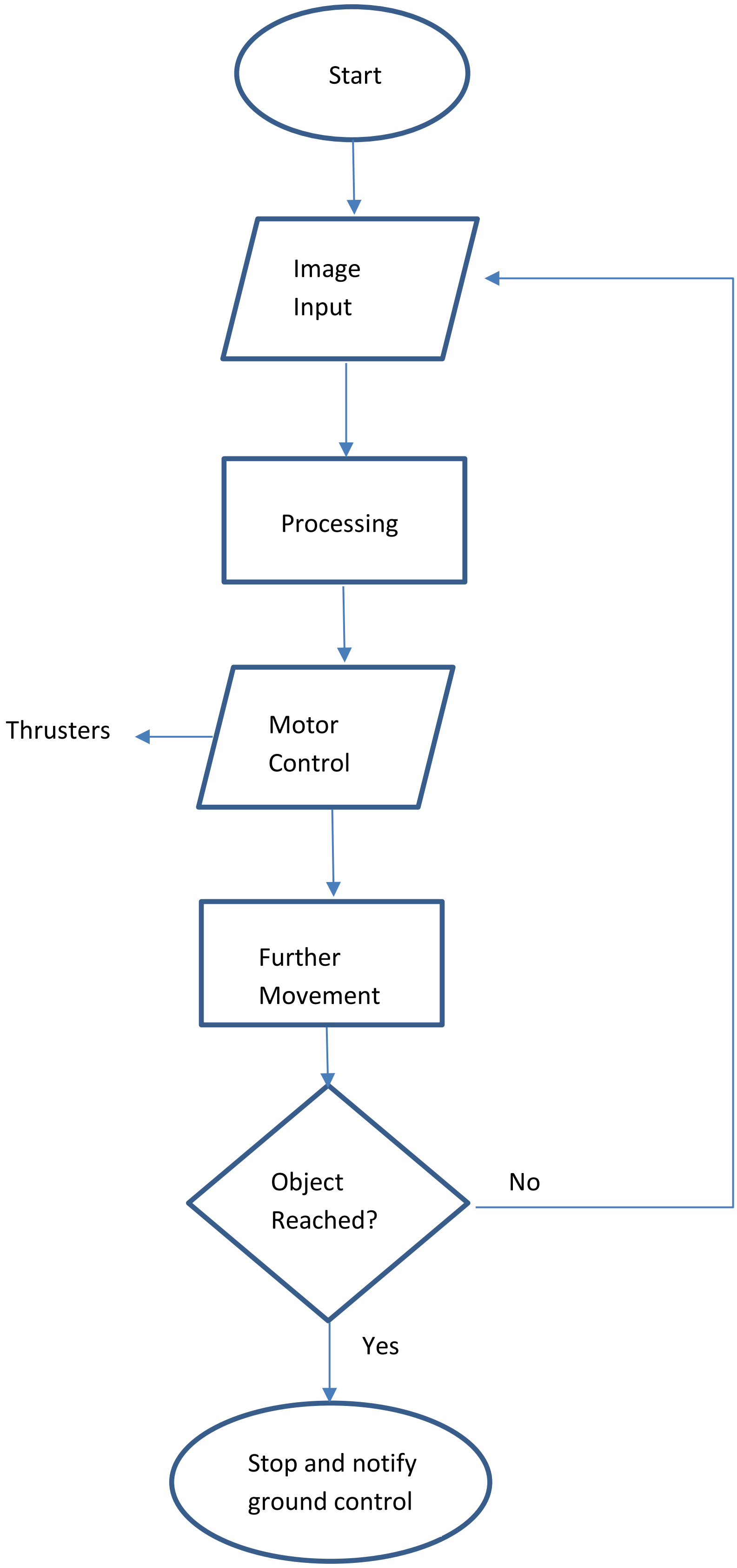


Fig 3.2 Flow Chart

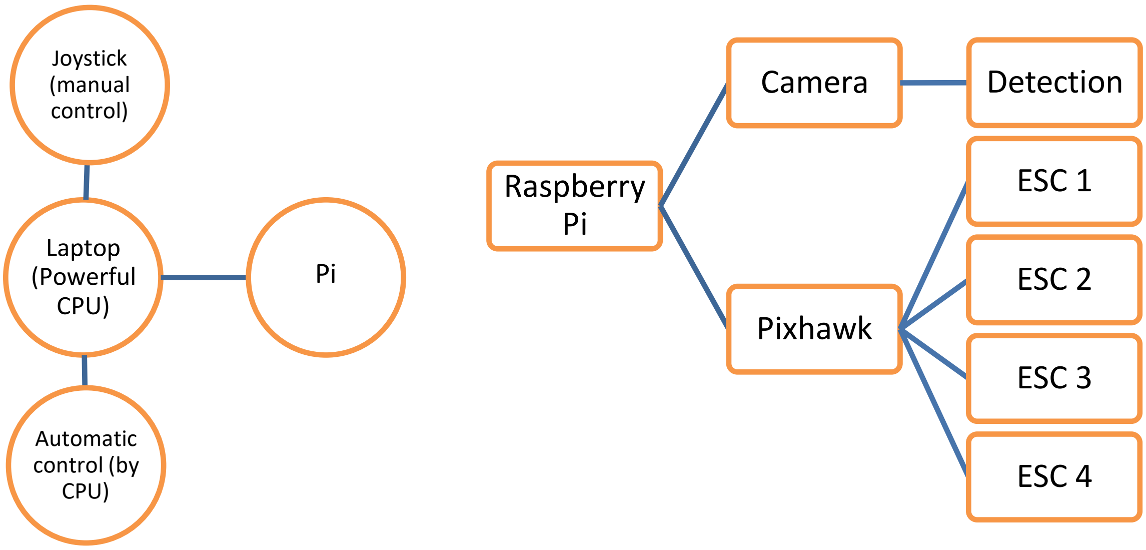


Fig 3.3 Hardware Block Diagram

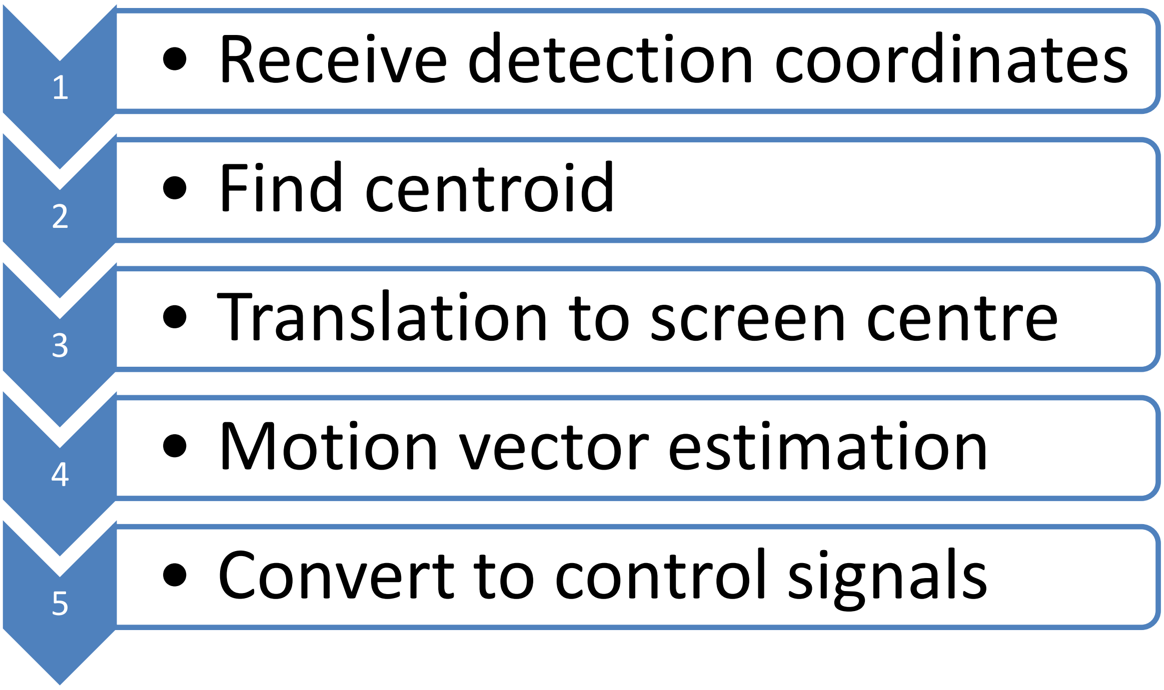


Fig 3.4 Control Processing and Motion Control

3.1.2 COMPONENTS

RASPBERRY PI

Raspberry Pi is the name of a series of single-board computers made by the Raspberry Pi Foundation, a UK charity that aims to educate people in computing and create easier access to computing education. The Raspberry Pi launched in 2012, and there have been several iterations and variations released since then. The original Pi had a single-core 700MHz CPU and just 256MB RAM, and the latest model has a quad-core 1.4GHz CPU with 1GB RAM. The main price point for Raspberry Pi has always been approx. INR 3000 and all models have been the same price or less, including the Pi Zero, which costs just INR 600. All over the world, people use Raspberry Pi’s to learn programming skills, build hardware projects, do home automation, and even use them in industrial applications. The Raspberry Pi is a very cheap computer that runs Linux, but it also provides a set of GPIO (general purpose input/output) pins that allow you to control electronic components for physical computing and explore the Internet of Things (IoT).

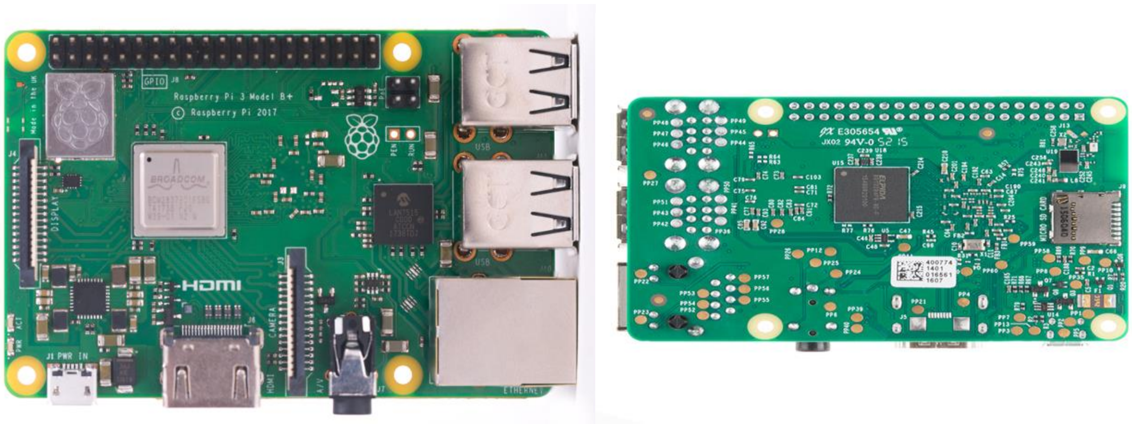
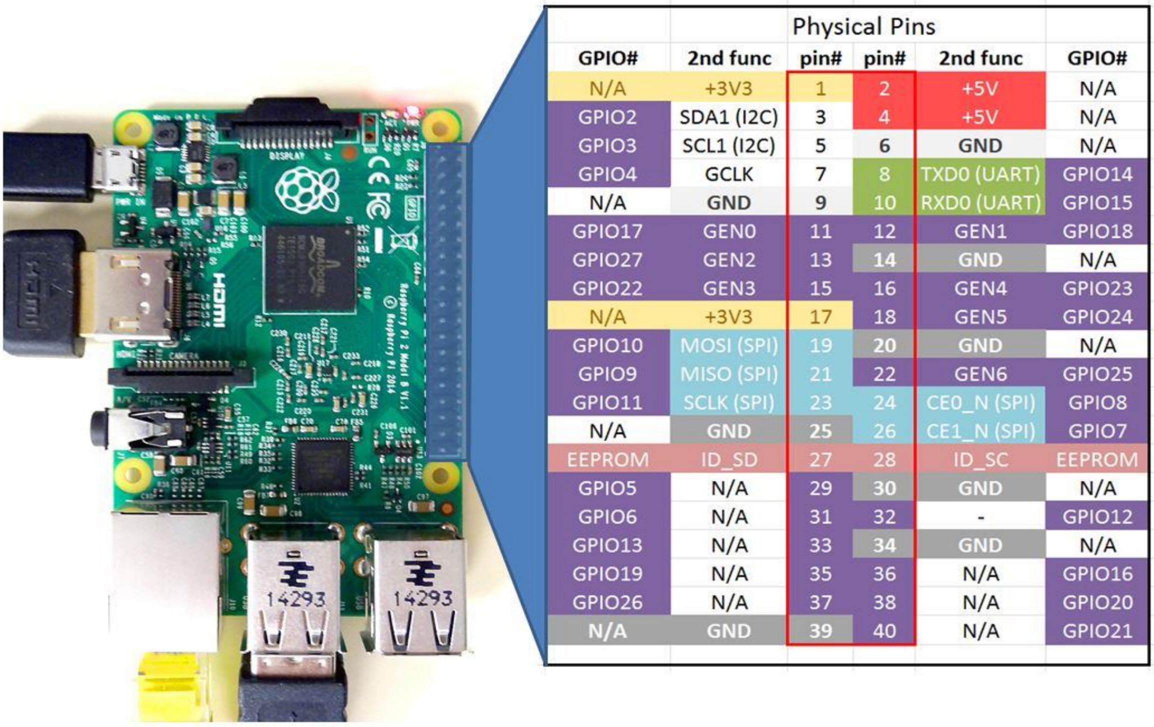


Fig 3.5 Raspberry Pi

The Broadcom BCM2835 SoC used in the first generation Raspberry Pi[20] includes a 700 MHz ARM11 76JZF-S processor, VideoCore IV graphics processing unit (GPU), and RAM. It has a level 1 (L1) cache of 16 KB and a level 2 (L2) cache of 128 KB. The level 2 cache is used primarily by the GPU. The SoC is stacked underneath the RAM chip, so only its edge is visible. The 1176JZ(F)-S is the same CPU used in the original iPhone, although at a higher clock rate, and mated with a much faster GPU.

The Raspberry Pi 3+ uses a Broadcom BCM2837B0 SoC with a 1.4 GHz 64-bit quad-core ARM Cortex-A53 processor, with 512 KB shared L2 cache.[1] The Raspberry Pi Zero and ZeroW use the same Broadcom BCM2835 SoC as the first generation Raspberry Pi, although now running at 1GHz CPU clock speed.



		Physical Pins			
GPIO#	2nd func	pin#	pin#	2nd func	GPIO#
N/A	+3V3	1	2	+5V	N/A
GPIO2	SDA1 (I2C)	3	4	+5V	N/A
GPIO3	SCL1 (I2C)	5	6	GND	N/A
GPIO4	GCLK	7	8	TXD0 (UART)	GPIO14
N/A	GND	9	10	RXD0 (UART)	GPIO15
GPIO17	GEN0	11	12	GEN1	GPIO18
GPIO27	GEN2	13	14	GND	N/A
GPIO22	GEN3	15	16	GEN4	GPIO23
N/A	+3V3	17	18	GEN5	GPIO24
GPIO10	MOSI (SPI)	19	20	GND	N/A
GPIO9	MISO (SPI)	21	22	GEN6	GPIO25
GPIO11	SCLK (SPI)	23	24	CE0_N (SPI)	GPIO8
N/A	GND	25	26	CE1_N (SPI)	GPIO7
EEPROM	ID_SD	27	28	ID_SC	EEPROM
GPIO5	N/A	29	30	GND	N/A
GPIO6	N/A	31	32	-	GPIO12
GPIO13	N/A	33	34	GND	N/A
GPIO19	N/A	35	36	N/A	GPIO16
GPIO26	N/A	37	38	N/A	GPIO20
N/A	GND	39	40	N/A	GPIO21

Fig 3.6 Raspberry Pi pin out

PIXHAWK

Pixhawk PX4 autopilot is an open-source autopilot system oriented toward inexpensive autonomous aircraft. An autopilot allows a remotely piloted aircraft to be flown out of sight. All hardware and software is open-source and freely available to anyone under a BSD license. Free software autopilots provide more flexible hardware and software. Users can modify the autopilot based on their own special requirements. The open-source software suite contains everything to let airborne system fly including **QGroundControl** and **MAVLink** Micro Air Vehicle Communication Protocol, 2D/3D aerial maps (with Google Earth support) and Drag-and-drop waypoints.

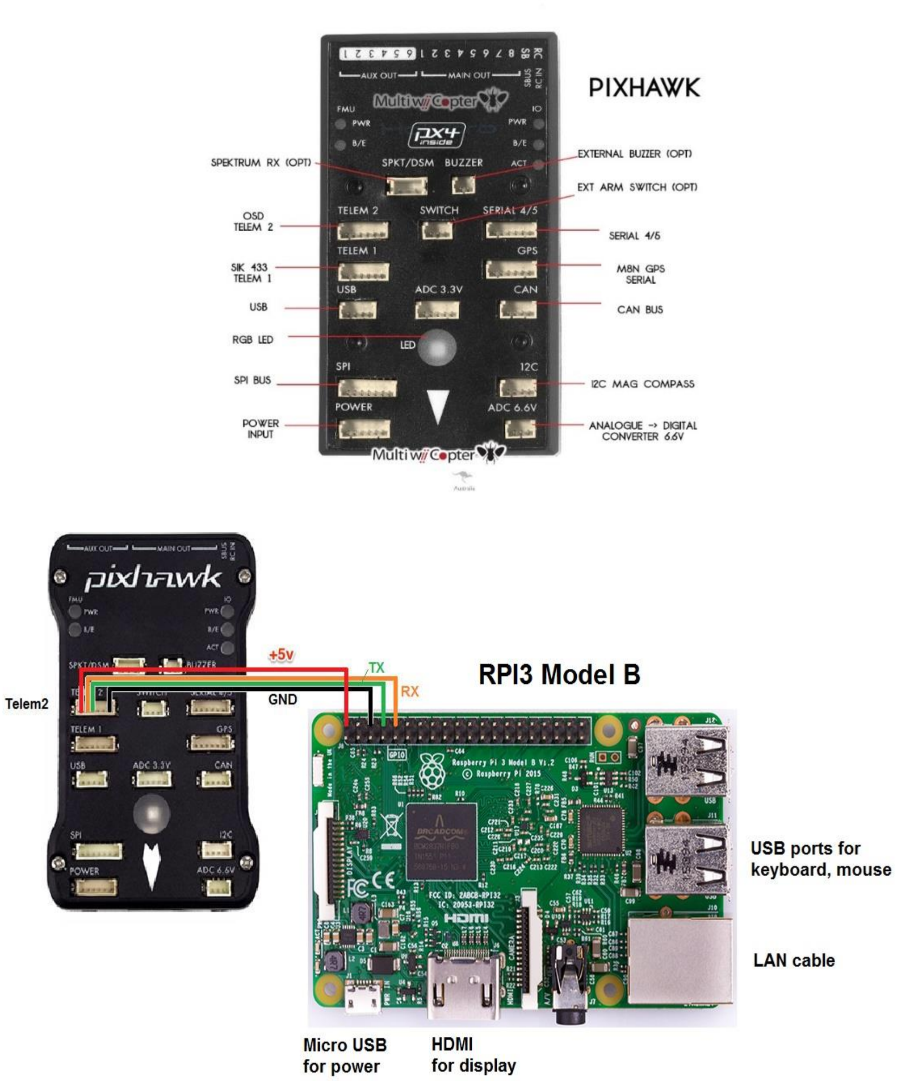


Fig 3.7 Pixhawk PX4 and interfacing with Raspberry Pi

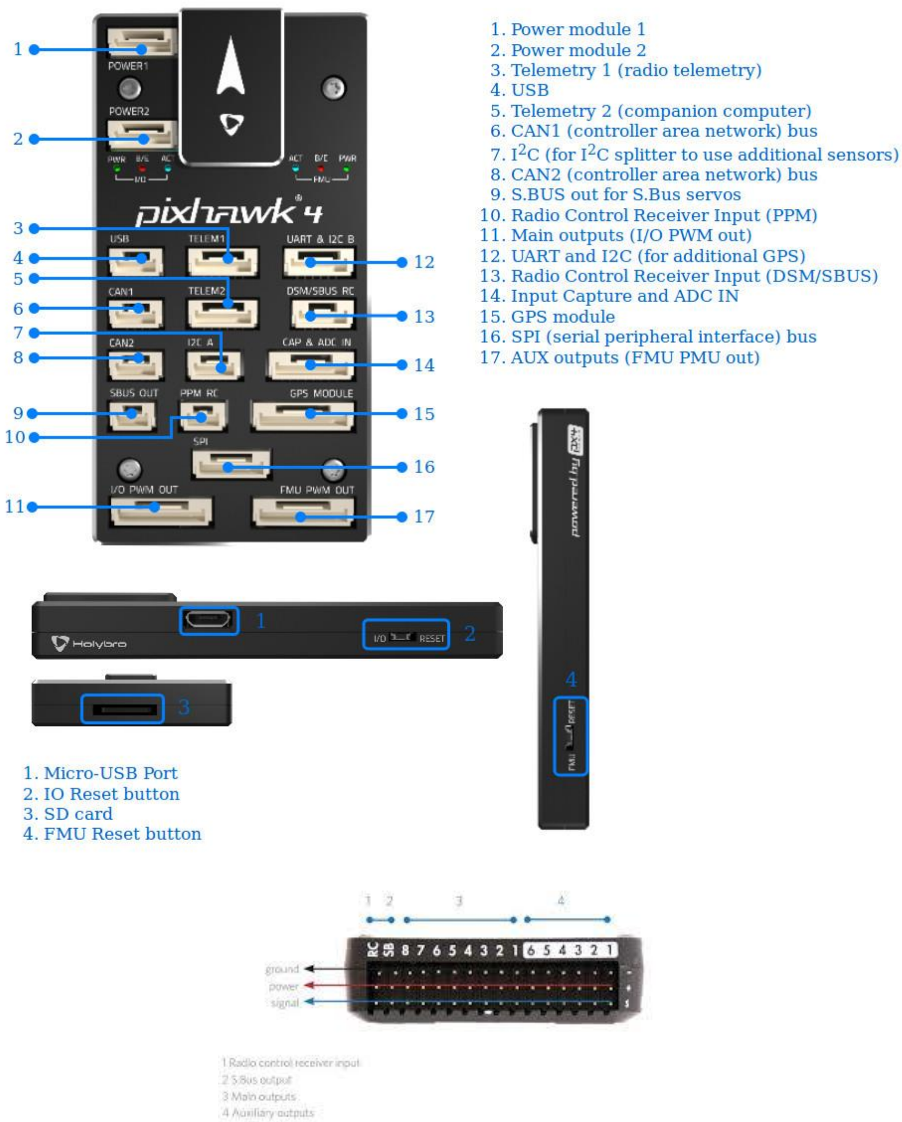


Fig 3.8 Pixhawk ports

Fig 3.9 Interfacing block diagram

USB CAMERA

USB camera typically include a lens, an image sensor, support electronics, and may also include one or even two microphones for sound. Image sensors can be CMOS or CCD, the former being dominant for low-cost cameras, but CCD cameras do not necessarily outperform CMOS-based cameras in the low-price range. Most consumer webcams are capable of providing VGA-resolution video at a frame rate of 30 frames per second. Many newer devices can produce video in multi-megapixel resolutions, and a few can run at high frame rates such as the PlayStation Eye, which can produce 320×240 video at 120 frames per second. The Wii Remote contains an image sensor with a resolution of 1024×768 pixels. As the bayer filter is proprietary, any webcam contains some built-in image processing, separate from compression.

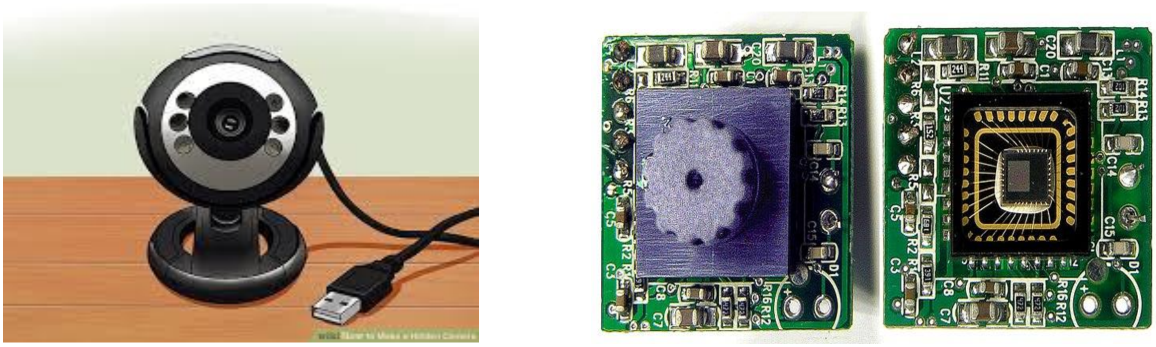


Fig 3.10 USB Camera

The Android platform supports the use of plug-and-play USB cameras (i.e. webcams) using the standard Android Camera2 API and the camera HIDL interface. Webcams generally support USB video class (UVC) drivers and on Linux the standard Video4Linux (V4L) driver is used to control UVC cameras.

The word camera comes from camera obscura, which means "dark chamber" and is the Latin name of the original device for projecting an image of external reality onto a flat surface. The modern photographic camera evolved from the camera obscura. The functioning of the camera is very similar to the functioning of the human eye. An action camera or action-cam is a digital camera designed for recording action while being immersed in it. Action cameras are therefore typically compact and rugged, and waterproof at surface-level.

They typically can take photos in burst mode and time-lapse mode as well as record high-definition video. Image stabilization (both electronic and optical) are essential in action Cameras to produce a reasonably good video. An Internet Protocol camera, or IP camera, is a type of digital video camera that receives control data and sends image data via the Internet. They are commonly used for surveillance. Unlike analog closed-circuit television (CCTV) cameras, they require no local recording device, but only a local area network. Most IP cameras are webcams, but the term IP camera or netcam usually applies only to those used for surveillance that can be directly accessed over a network connection. Some IP cameras require support of a central network video recorder (NVR) to handle the recording, video and alarm management. Others are able to operate in a decentralized manner with no NVR needed, as the camera is able to record directly to any local or remote storage media.

Pool safety cameras are video monitoring or recording systems designed to improve safety, such as by reducing drowning deaths, injuries, or criminal activity, in public and private pools, waterparks, thermal baths, or spa facilities. Manufacturers include Coral Manta, SwimEye, AngelEye, Argusmatik, Poseidon, and Zwembadcamera.

Aquatics video monitoring systems are broken into two categories: Passive Active Passive systems provide lifeguards with views of below water swimmer activity and behaviour. Active systems are designed to further help lifeguards in an attempt to address the physical limitations imposed by the human factor.

A thermographic camera (also called an infrared camera or thermal imaging camera or infrared thermography) is a device that forms a heat zone image using infrared radiation, similar to a common camera that forms an image using visible light. Instead of the 400–700 nanometre range of the visible light camera, infrared cameras operate in wavelengths as long as 14,000 nm (14 µm). Their use is called thermography. An acoustic camera is an imaging device used to locate sound sources and to characterize them. It consists of a group of microphones, also called a microphone array, from which signals are simultaneously collected and processed to form a representation of the location of the sound sources.

An acoustic camera generally consists of a microphone array and optionally an optical camera. The microphones – analog or digital – are acquired simultaneously or with known relative time delays to be able to use the phase difference between the signals. As the sound propagates in the medium (air, water etc.) at a finite known speed, a sound source is perceived by the microphones at different time instants and at different sound intensities that depend on both the sound source location and the microphone location.

One popular method to obtain an acoustic image from the measurement of the microphone is to use beamforming: By delaying each microphone signal relatively and adding them, the signal coming from a specific direction is amplified while signals coming from other directions are cancelled. The power of this resulting signal is then calculated and reported on a power map at a pixel corresponding to the direction. The process is iterated at each direction where the power needs to be computed. While this method has many advantages – robustness, easy to understand, highly parallelizable because each direction can be computed independently, versatile (there exist many types of beamformers to include various types of hypothesis), relatively fast – it also has some drawbacks: the produced acoustic map has artefacts (also called side lobes or ghost sources) and it does not model correctly correlated sound sources. Various methods have been introduced to reduce the artefacts such as DAMAS or to take in account correlated sources such as CLEAN-SC, both at the price of a higher computational cost. When the sound sources are near the acoustic camera, the relative intensity perceived by the different microphones as well as the waves not being any more seen as planar but spherical by the acoustic camera add new information compared to the case of sources being far from the camera. It enables to use more effective methods such as acoustic holography.

Some acoustic cameras use two-dimensional acoustic mapping, which uses a unidirectional microphone array (e.g. a rectangle of microphones, all facing the same direction). Two-dimensional acoustic mapping works best when the surface to be examined is planar and the acoustic camera can be set up facing the surface perpendicularly. However, the surfaces of real-world objects are not often flat, and it is not always possible to optimally position the acoustic camera.

Three-dimensional acoustic cameras fix the errors of two-dimensional cameras by taking into account surface depths, and therefore correctly measuring the distances between the microphone and each spatial point. These cameras produce a more accurate picture, but require a 3-D model of the object or space being analysed.

LAN Cable (RJ45)

Radio Jack 45 cable or commonly known as “LAN cable”. The RJ45S, a standard jack once specified for modem or data interfaces, uses a mechanically-keyed variation of the 8P8C body with an extra tab that prevents it from mating with other connectors; the visual difference from the more-common 8P8C is subtle. The original RJ45S keyed 8P2C modular connector had pins 5 and 4 wired for tip and ring of a single telephone line, and pins 7 and 8 shorting a programming resistor, but is obsolete today. The RJ45S jack must not be confused with the 8P8C eight-pin modular connector. The latter is often incorrectly called RJ45 connector in several fields such as telecommunications and computer networking but it lacks the extra tab. Besides, its pin-out involves some particular schematics as just mentioned.

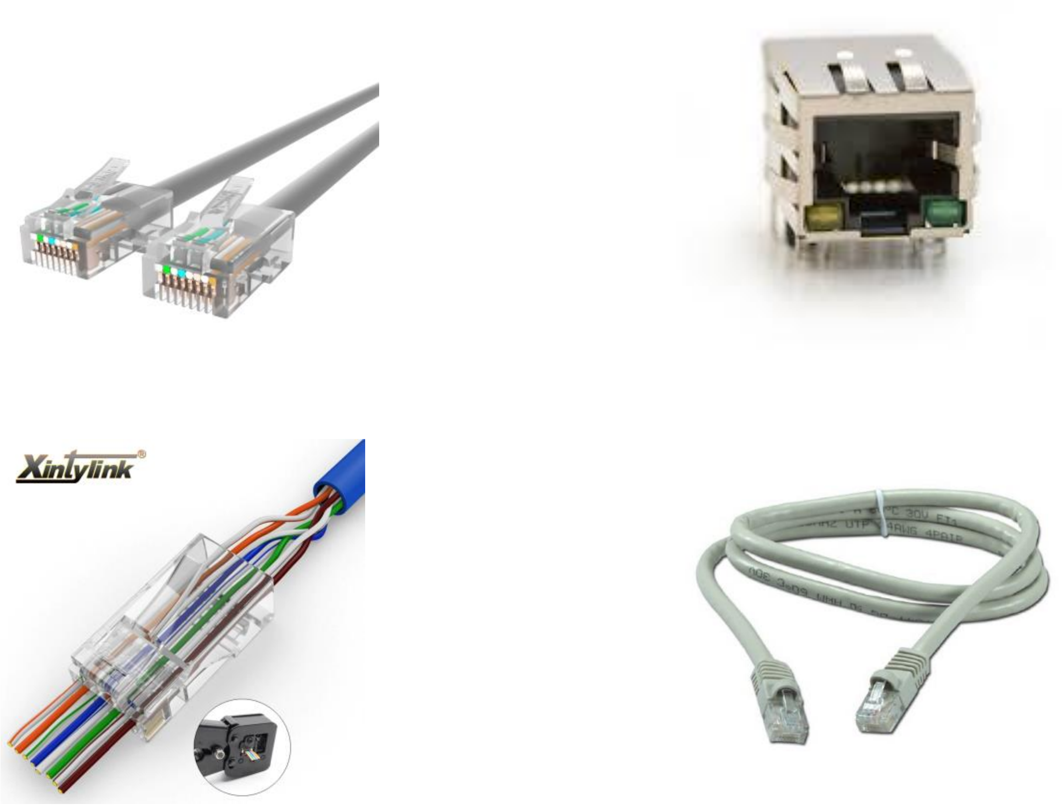


Fig 3.11 LAN cables and ports

WATERPROOF ULTRASONIC TRANSCEIVER

Ultrasonic sensing is one of the best ways to sense proximity and detect levels with high reliability. An ultrasonic sensor is an instrument that measures the distance to an object using ultrasonic sound waves.

Ultrasonic sensors work by emitting sound waves at a frequency too high for humans to hear. They then wait for the sound to be reflected back, calculating distance based on the time required. This is similar to how radar measures the time it takes a radio wave to return after hitting an object. An ultrasonic sensor uses a transducer to send and receive ultrasonic pulses that relay back information about an object's proximity. High-frequency sound waves reflect from boundaries to produce distinct echo patterns.

Ultrasonic sensors are superior to infrared sensors because they aren't affected by smoke or black materials, however, soft materials which don't reflect the sonar (ultrasonic) waves very well may cause issues. It's not a perfect system, but it's good and reliable. Ultrasonic sensors are a reliable, cost-effective solution for distance sensing, level, and obstacle detection.

Water-proofed ultrasonic transceiver modules are available which can be used for depth sensing in water bodies.

If you need to measure the specific distance from your sensor, this can be calculated based on this formula:

Distance = ½ T x C

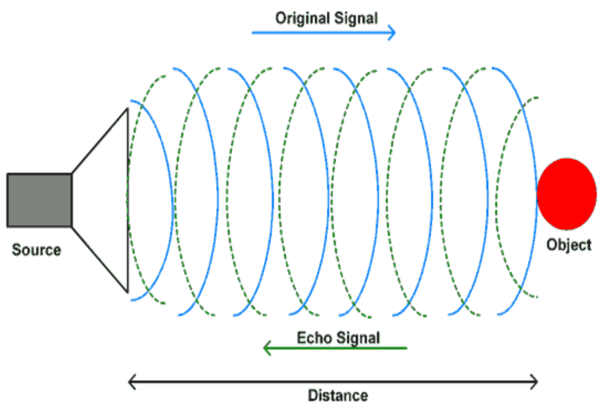


Fig 3.12 Hydrophone and Echo illustration

UAPS

An underwater acoustic positioning system (UAPS) is a system for the tracking and navigation of underwater vehicles or divers by means of acoustic distance and/or direction measurements, and subsequent position triangulation.

Fig 3.13 UAPS or "Underwater GPS"

In the long baseline example (see figure 1), an interrogator (A) is mounted on the ROV that is to be tracked. The interrogator transmits an acoustic signal that is received by the baseline transponders (B, C, D, E). The reply of the baseline transponders is received again at the ROV. The signal time-of-flight or the corresponding distances A-B, A-C, A-D and A-E are transmitted via the ROV umbilical (F) to the surface, where the ROV position is computed and displayed on a tracking screen. The acoustic distance measurements may be augmented by depth sensor data to obtain better positioning accuracy in the three-dimensional underwater space. Acoustic positioning systems can yield an accuracy of a few centimetres to tens of meters and can be used over operating distance from tens of meters to tens of kilometres. Performance depends strongly on the type and model of the positioning system, its configuration for a particular job, and the characteristics of the underwater acoustic environment at the work site.

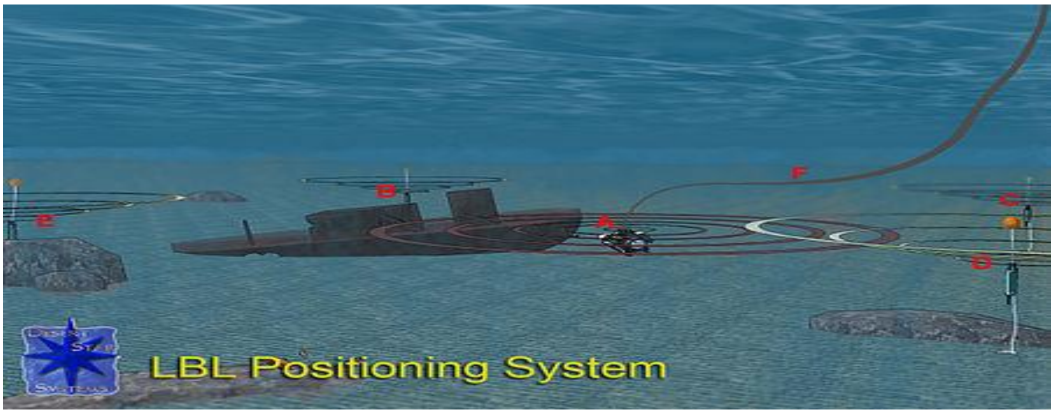


Fig 3.14 Long Base Line Positioning System

ARDUINO

Arduino is an open-source platform used for building electronics projects. Arduino consists of both a physical programmable circuit board often referred to as a microcontroller and a piece of software, or IDE (Integrated Development Environment) that runs on your computer, used to write and upload computer code to the physical board. The Arduino platform has become quite popular with people just starting out with electronics, and for good reason. Unlike most previous programmable circuit boards, the Arduino does not need a separate piece of hardware (called a programmer) in order to load new code onto the board -- you can simply use a USB cable. Additionally, the Arduino IDE uses a simplified version of C++, making it easier to learn to program. Finally, Arduino provides a standard form factor that breaks out the functions of the micro-controller into a more accessible package. Burning the required program is usually done with the help of Arduino IDE software. We initially used the Arduino UNO to control the speed of the bilge pump using the ESC. Sensor reading where also measured using arduino and was send directly to the laptop through LAN. The control signal for the Arduino was given from the raspberry pi.

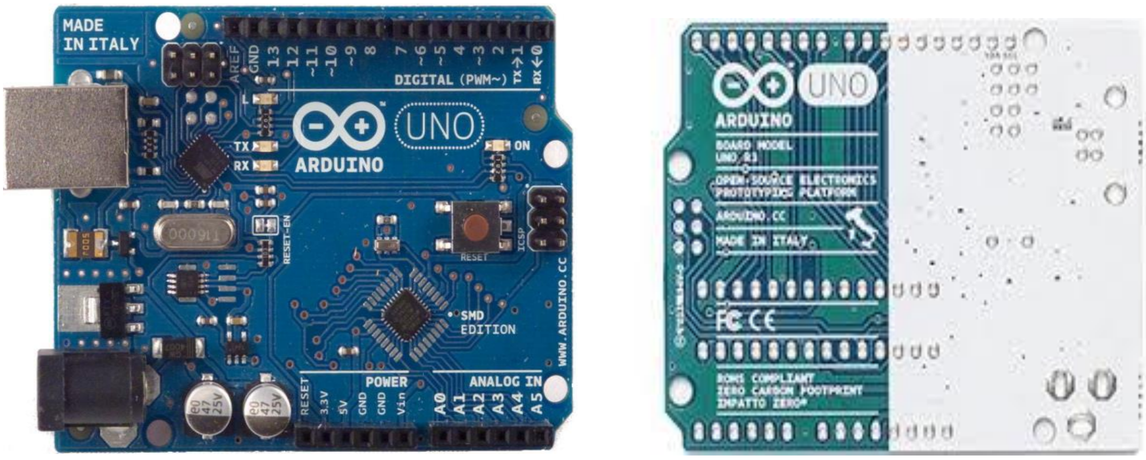


Fig 3.15 Arduino Uno board

BILGE PUMP

For smooth in-water movement, we need thrusters. And thrusters are very costly. So, what we did here was to take a bilge pump (submersible fish tank pump), cut out the top portion and then replace the impeller with a boat propeller. An impeller is a rotor used to increase the pressure and flow of a fluid. A propeller is a type of fan that transmits power by converting rotational motion into thrust. A pressure difference is produced between the forward and rear surfaces of the aerofoil-shaped blade, and a fluid is accelerated behind the blade. We used 4 4-blade fixed pitch propellers .

We used 4 modified bilge pumps instead of 4 thrusters for our project. With this adjustment we could reduce the cost of our project by 1000% (40,000 to 4,000 INR).

Fig 3.16 Bilge pump with impeller replaced with propeller

ESC

An electronic speed control or ESC is an electronic circuit that controls and regulates the speed of an electric motor. It may also provide reversing of the motor and dynamic braking. Miniature electronic speed controls are used in electrically powered radio-controlled models. Full-size electric vehicles also have systems to control the speed of their drive motors.

An electronic speed control follows a speed reference signal (derived from a throttle lever, joystick, or other manual input) and varies the switching rate of a network of field effect transistors (FETs). By adjusting the duty cycle or switching frequency of the transistors, the speed of the motor is changed. The rapid switching of the transistors is what causes the motor itself to emit its characteristic high-pitched whine, especially noticeable at lower speeds.

Different types of speed controls are required for brushed DC motors and brushless DC motors. A brushed motor can have its speed controlled by varying the voltage on its armature. (Industrially, motors with electromagnet field windings instead of permanent magnets can also have their speed controlled by adjusting the strength of the motor field current.) A brushless motor requires a different operating principle. The speed of the motor is varied by adjusting the timing of pulses of current delivered to the several windings of the motor.

Fig 3.17 Waterproof Brushed Motor ESC (S10E-RTR)

POWER SUPPLY AND CONNECTIONS

Component	Rating
1100 GPH Bilge pump	12V 3A (max. load) 900mA (no load)
Raspberry Pi 3 B+	5V 2A
Underwater Brushed motor ESC	5V control signal (very low current) (works with 12V and 320A max. burst current)
Pixhawk PX4	5V (very low current), power able from Raspberry Pi USB port.

Table 3.1 Power consumption/ load requirements

Connections:

1. 4 Bilge pumps are each connected to the output of ESCs.

2. All 4 ESCs are connected in parallel to the 12V 30AH supply with separate grounds (to prevent leakage, back emf., which will affect the working of ESC.)

3. Though rated as 3A under maximum load, it draws only around 2-2.2Amps at max. So total around 10A current is drawn from battery under maximum load (underwater).
(Note: If T100 thrusters are used instead of bilge pump, they use ESCs rated at 90A (or they at least draw more than 30Amps), so they can work with much more load. However they are brushless and hence can rotate in only one direction.)

4. A voltage buck converter is connected from the 12V supply and the output is given to Raspberry Pi, which draws approx. 1-2A current.

5. Pixhawk PX4 has a port for serial connection, which follows USB protocol. We used 4 wires to take connections from the port and connect to the USB port of Pi. Pixhawk uses this USB port for both power and communication with Pi. It works with the 5V supply from the USB port and the current drawn from it. (max 300 mA).

6. 4 ESC’s control pins are connected to Pixhawk’s output port (bottom side, which provide the PWM control signals.

7. Pixhawk has the arming sequence stored inside it, which makes it automatically initialize the ESC connected to it.

8. ESC’s on/off trigger is always set to ON.

9. Relays can be used for restart action and control, if pixhawk is not available and connection has to be made directly with Pi.

10. While operating, in QGC submarine user interface, give gain as 25. If gain is 50 or more, due to back emf generated (due to heavy load underwater), pixhawk may restart. So care has to be given for that.

(a) (b)

Fig 3.18 (a) DC Buck converter 12V-5V (b) 12V Amaron car battery 30 AH

3.1.3 WORKING

CONTROL MECHANISM OF ROV

Underwater thrusters can be divided in two main groups, hydraulic thrusters and electric thrusters. Electric thrusters are mainly used on battery operated underwater robots such as AUVs, submarines and electric ROVs. Hydraulic thrusters are mainly used on work class hydraulic ROVs. Hydraulic thruster technology is older than the electrical one, they are more rugged and their weight-to-thrust ratio is higher than electric thrusters, but maintenance and piping issues cause some dissatisfaction with users. Thanks to developing electric motors, electric thrusters are becoming more popular in newly designed products. Weight-to-thrust ratios are higher for hydraulic thrusters than for electric thrusters, but after taking into account the required hydraulic components including valves, hydraulic power units, pipes joints, etc. hydraulic thruster systems come out heavier than electric thrusters.

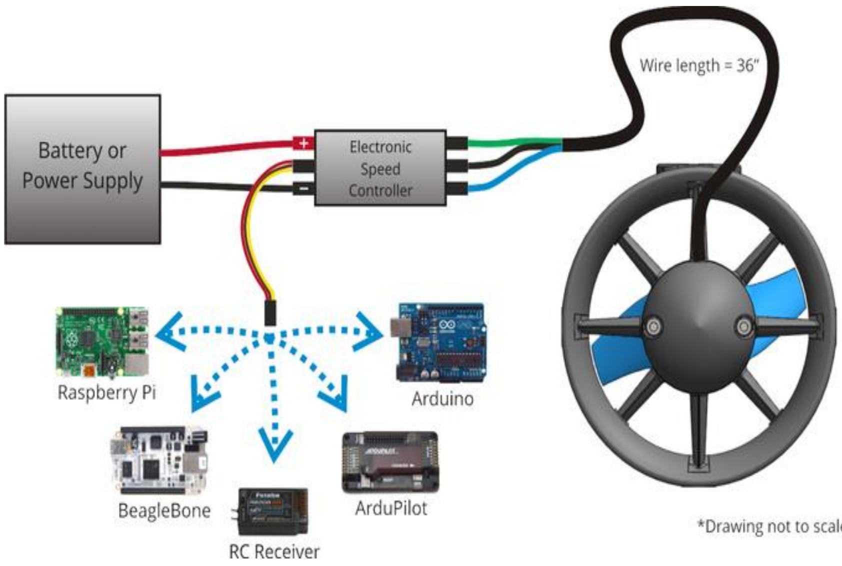


Fig 3.19 Thruster interfacing

The speed and direction can be controlled by applying proper electrical signal to the thruster. A dedicated electronic speed control module is employed to do this task. The control signals for this module is provided by any one of the core controller like Raspberry Pi or Arduino. The signals from the core controller is decided by the movement of ROV. That is the control signals for forward movement is different from control signals for backward movement. Here the power required by the thruster is provided externally.

Communication between Surface and ROV

The ROV is employed for detecting human body underwater. If it detects the presence, it is necessary to pass this information to surface. Hence a communication method is required. An ASK module of 433Mhz can be used for transmitting data from ROV to surface. It is a low power transmitter module. When we connect these modules with HT12 series encoder and Decoder, it will form a 4 bit parallel transmitter and receiver. Since it is a low power transmitter, we can only send low speed data. In the case of ROV, the information that we need to transmit are the presence of human body and the location of ROV. ASK modules are capable of transmitting this information to the surface from underwater ROV. We tested the communication between ROV and surface by using an RF transmitter however the communication was limited only to a few meters. Transmission through EM waves don’t really work underwater, only acoustic communication is possible. Due to these limitations we went on to use wired means of communication through a LAN cable. We get the live video feed from the camera and transmit it to the surface via the LAN cable. Transmission was done by sending these video frames as UDP packets. UDP was faster however lesser reliable as compared to TCP. Each frame of the video was encoded into strings and was serially transmitted through the LAN cable. We implemented this UDP communication by means of socket programming. We started a UDP socket server on the ROV and the client which is running on the processing unit outside gets connected to the server through the wired LAN. This resolved the problem of underwater communication however using a lengthy cable could have resulted in an overall drag of the vehicle reducing its efficiency. Besides sending the video frames the control signals obtained after processing each of the frame was also serially transmitted through the LAN cable.

Algorithm:

1. Camera connected to Raspberry Pi, captures the underwater scene and sends it to the Pi.
2. Pi and Laptop (Processor on the land) are connected to the same network, via a LAN cable.
3. Pi streams the input video received from camera via the LAN to the laptop computer on the land, with the help of UDP protocol.
4. Though UDP is relatively less reliable compared to TCP, we used it here for quick

transmission.

5. The received file from Pi is processed in Laptop computer with the developed program, using Tensorflow and decisions are made.
6. The control decisions made by the program is again sent to Pi using the same network.
7. Pi then forwards the control instructions to Pixhawk flight controller serially via USB.
8. Pixhawk controls the ESC with the received instructions.
9. ESC controls the motor speed and direction.

Working:

1.Camera input

Camera footage from the h.264 camera is received by the Pi serially (USB). Pi transmits the data to the laptop computer (on land surface) as UDP packets. The detection and tracking program in Laptop (main program) performs the detection action and movement calculations and inform the QGC submarine software, which then follows the control action as follows.

2. Control

QGC submarine installed on the laptop computer provides a user interface to issue PWM control signals to the ESC through Pixhawk. We use MAVLink protocol to either simulate (for autonomous control) or forward actual joystick control to ESC. From laptop computer (on land surface), this control signals are sent as UDP packets to Raspberry Pi via LAN cable. Pi acts as a carrier to convert this packet information reached via LAN to USB serial data, which is passed on to Pixhawk PX4. Pixhawk controls the ESCs with these thruster control signals.

Note: Separate UDP ports are used for camera footage transmission and control signal transmission. We used 14450 port for control and 5600 port for camera streaming. Make sure that the Pi and Laptop are in the same network, by verifying the IP address.

Human body Detection

Human body detection can be done using trained neural classifiers. Either we can detect the presence of body using images or by measuring the reflected strength of the acoustic wave. An underwater ultrasonic sensor can be used to detect human body. Acoustic imaging can be done with the help of side scan sonar or acoustic cameras. We can use these images to train a classifier. In our project we used a normal USB camera for the training instead of acoustic imaging. Detection can be also done with the help of acoustic transducers or hydrophone that detect the strength of the reflected sound wave. The basic principle behind the process is that the strength of ultrasonic signals reflected from various objects are different. That means based on the strength and pattern of reflected signals, we can easily classify the objects around the sensor. Since the reflected signal amplitude is very weak, the sensor output is also weak and noisy. For effective classification of signals, any one of the machine learning algorithms is required. Based on the classification result, the presence of human body is determined.

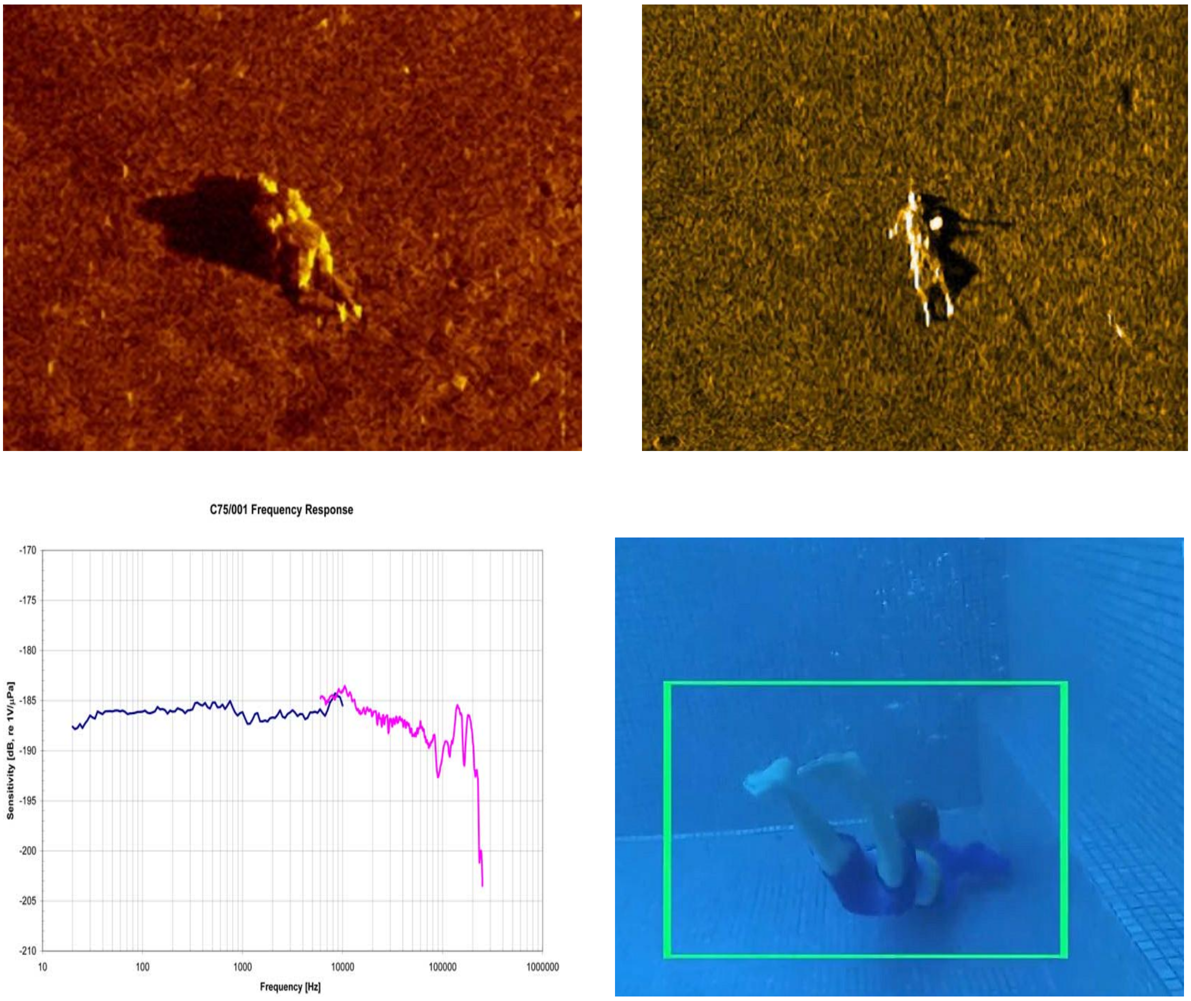


Fig 3.20 Different object detection methods

3.2 SOFTWARE SECTION

3.2.1 FLOWCHART

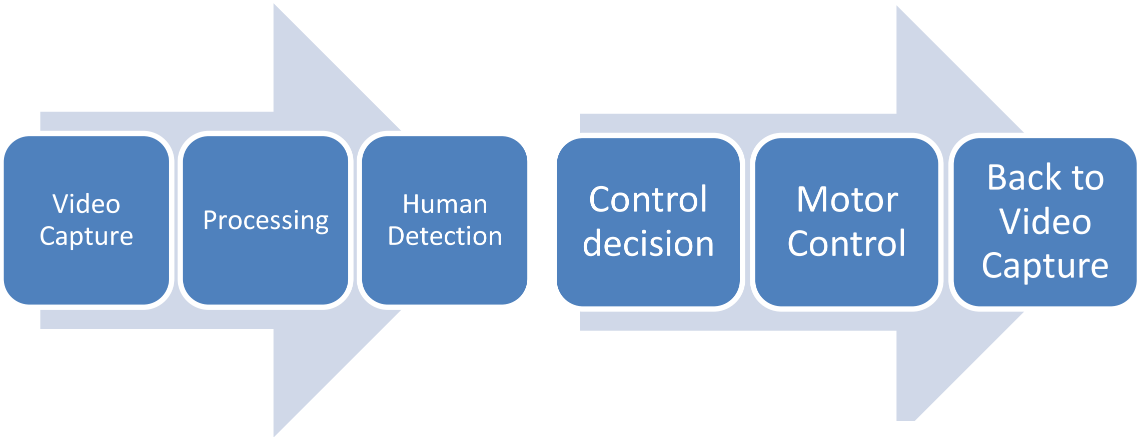


Fig 3.21 Flow chart

3.2.2 ALGORITHM

The principal algorithm has 2 major parts:

1. Underwater Human detection, locking and tracking.
2. Controlling the motors to track.

In underwater human detection part, we detected human body using modified Haar classifiers and edge detection techniques. We then translate the detected body's centre of mass coordinates to the viewing frame's centre. For each frame we do this, and this is how we lock the position to be able to track later.

In Tracking part, we power up the thrusters suitably with different speeds to move suitable so as to lock the detected object in the view plane's centre itself.

We know we have reached near the object, when the object area becomes comparable to the view plane area. So then, we will slow down to maintain a safe distance from the target.

Object detection using deep neural network

Deep learning is a sub-field of machine learning dealing with algorithms inspired by the structure and function of the brain called artificial neural networks. In other words, It mirrors the functioning of our brains. Deep learning algorithms are similar to how nervous system structured where each neuron connected each other and passing information.

Some deep learning frameworks:

- Caffe
- Theano
- Tensorflow
- Lasagne
- Keras
- Pytorch

The Faster Region-based Convolutional Network (Faster R-CNN) is a combination between the RPN and the Fast R-CNN model.

A CNN model takes as input the entire image and produces feature maps. A window of size 3x3 slides all the feature maps and outputs a features vector linked to two fully-connected layers, one for box-regression and one for box-classification. Multiple region proposals are predicted by the fully-connected layers. A maximum of k regions is fixed thus the output of the box-regression layer has a size of 4k (coordinates of the boxes, their height and width) and the output of the box-classification layer a size of 2k ("objectness" scores to detect an object or not in the box). The k region proposals detected by the sliding window are called anchors.

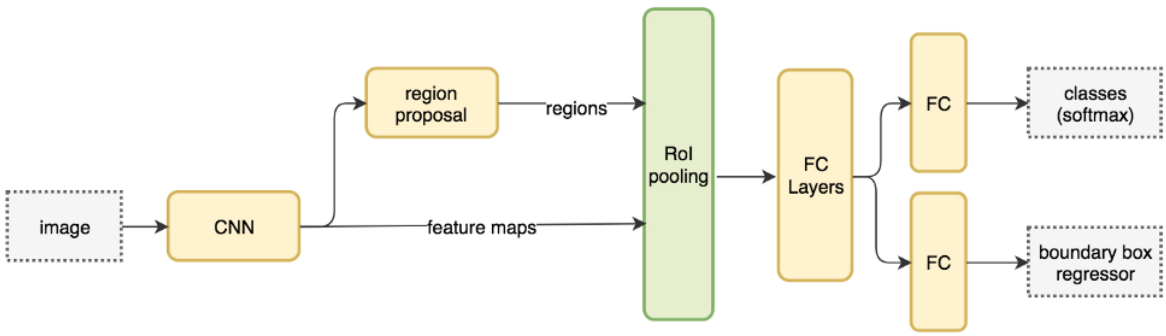


Fig 3.22 Fast R-CNN working model

When the anchor boxes are detected, they are selected by applying a threshold over the “objectness” score to keep only the relevant boxes. These anchor boxes and the feature maps computed by the initial CNN model feeds a Fast R-CNN model.

Faster R-CNN uses RPN to avoid the selective search method, it accelerates the training and testing processes, and improve the performances. The RPN uses a pre-trained model over the ImageNet dataset for classification and it is fine-tuned on the PASCAL VOC dataset. Then the generated region proposals with anchor boxes are used to train the Fast R-CNN. This process is iterative.

The best Faster R-CNNs have obtained mAP scores of 78.8% over the 2007 PASCAL VOC test dataset and 75.9% over the 2012 PASCAL VOC test dataset. They have been trained with PASCAL VOC and COCO datasets. One of these models[2] is 34 times faster than the Fast R-CNN using the selective search method.

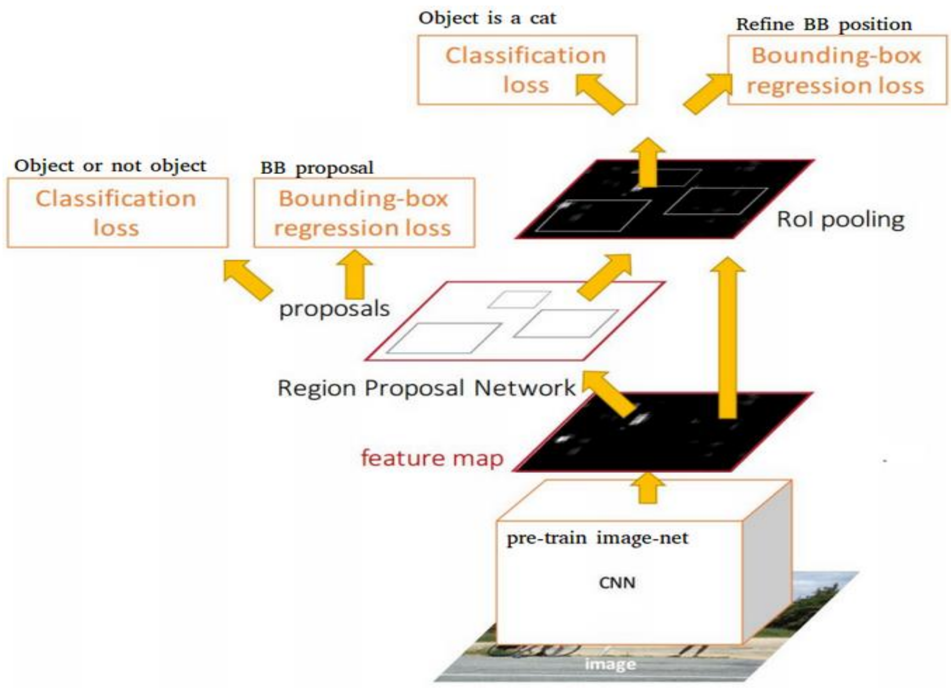


Fig 3.23 Training Fast R-CNN

Object Locking and Tracking

Algorithm:

1. Our main subject (human body) is detected from the camera input frame by Tensorflow and the program finds and returns the coordinates of a rectangle surrounding the detection area.
2. With this coordinates, we find the centre coordinates using the formula: ((x1+x2+x3+x4/4),(y1+y2+y3+y4)/4)
3. We find the vector from these centre coordinates to the screen centre.
4. We apply compensation to reduce this vector to magnitude zero by applying a translation operation to bring the centre coordinates to match with the screen centre coordinates.
5. We do this for all consecutive frames, while moving to maintain the detected object with in the screen centre itself.
6. The movement is adjusted by the whole movement controller setup, comprising of laptop, Pi and Pixhawk so as to do all the above steps correctly. These steps together form object locking and tracking.

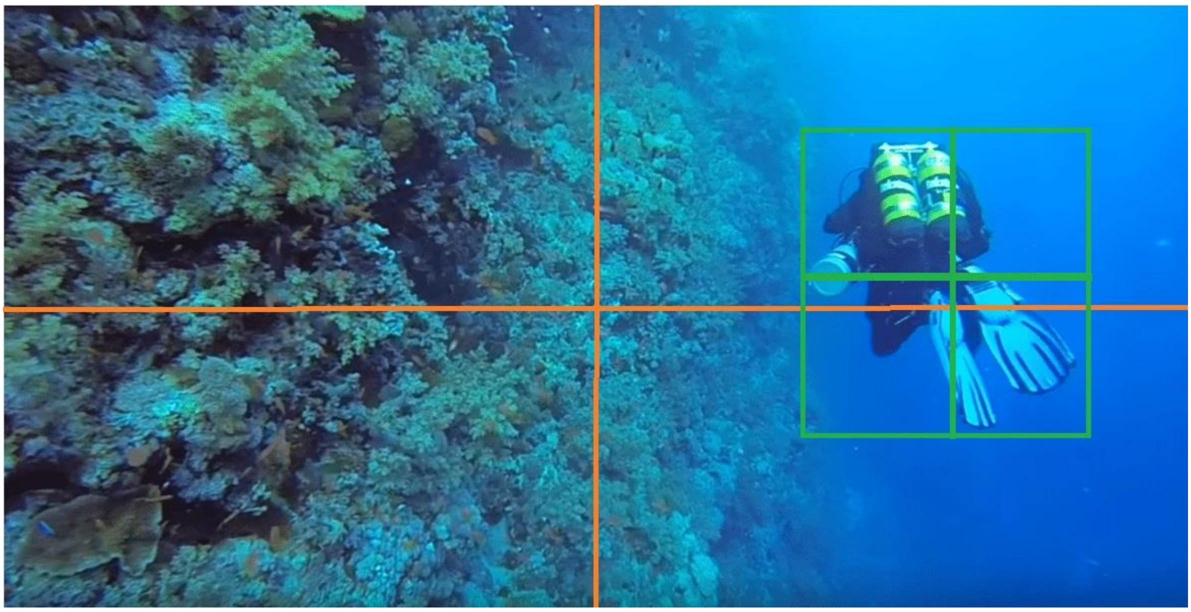


Fig 3.24 Object Locking and Tracking

3.2.3 DESCRIPTION OF SOFTWARE AND PLATFORM

ArduSub

We used ArduSub OS based on Raspbian (which is Debian Linux based) on the Raspberry Pi 3 B+ for this project. We used MAVLink Protocol for passing the manual control signals from the joystick to the Laptop. The laptop takes care of necessary processing (runs windows 10 OS) and sends necessary control signals to the raspberry pi connected to it via the LAN cable. Pi controls the pixhawk flight controller serially. Pixhwak (PX4) takes PI's instructions and control the 4 speed controllers connected to it. Speed Controllers control the motors (thrusters)as per the instruction received.

Pi takes the video camera input from the USB camera connected to it and streams it to the laptop via UDP. We used UDP especially because of its low latency. ArduSub works seamlessly with Ground Control Station software that can monitor vehicle telemetry and perform powerful mission planning activities. It also benefits from other parts of the ArduPilot platform, including simulators, log analysis tools, and higher level APIs for vehicle management and control. ArduSub has extensive capabilities out of the box including feedback stability control, depth and heading hold, and autonomous navigation

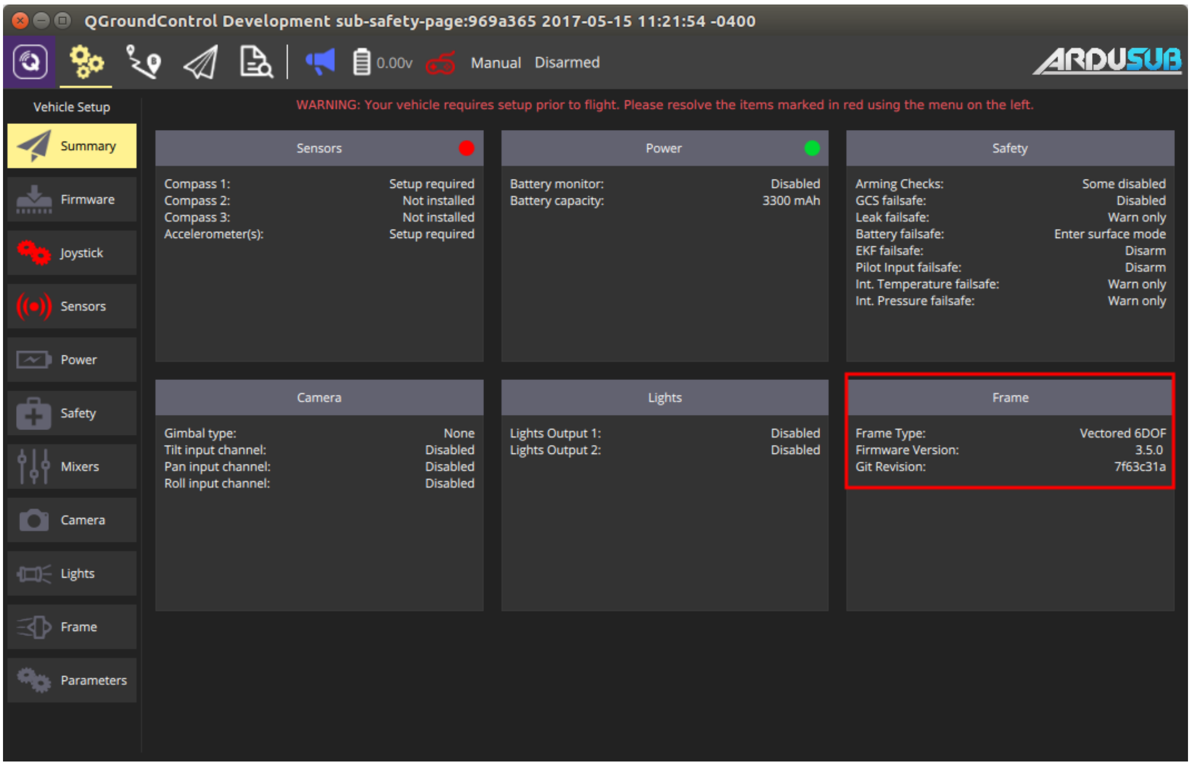


Fig 3.25 QGC UI

UDP

User Datagram Protocol (UDP) is a Transport Layer protocol. UDP is a part of Internet Protocol suite, referred as UDP/IP suite. Unlike TCP, it is unreliable and connectionless protocol**.** So, there is no need to establish connection prior to data transfer.

Though Transmission Control Protocol (TCP) is the dominant transport layer protocol used with most of Internet services; provides assured delivery, reliability and much more but all these services cost us with additional overhead and latency. Here, UDP comes into picture. For the realtime services like computer gaming, voice or video communication, live conferences; we need UDP. Since high performance is needed, UDP permits packets to be dropped instead of processing delayed packets. There is no error checking in UDP, so it also save bandwidth .User Datagram Protocol (UDP) is more efficient in terms of both latency and bandwidth.

UDP Datagram

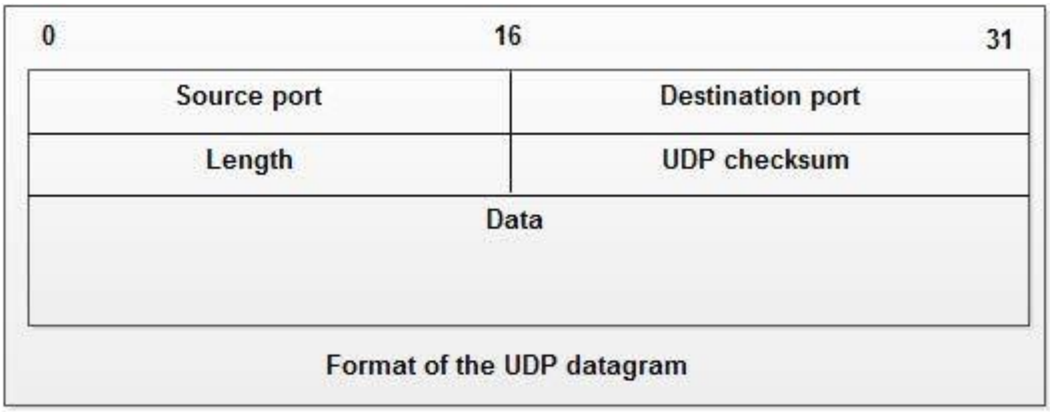


Fig 3.26 UDP Datagram

Source Port This is the port number of the application that is originating the user data.

Destination Port This is the port number pertaining to the destination application.

Length This field describes the total length of the UDP datagram, including both data and header Information.

UDP Checksum Integrity checking is optional under UDP. If turned on, both ends of the communications channel use this field for data integrity checks. The bilge pump, ESC used and the prototype are shown below:

MAVLINK

MAVLink is a very lightweight messaging protocol for communicating with drones (and between onboard drone components). MAVLink or Micro Air Vehicle Link is a protocol for communicating with small unmanned vehicle. The protocol defines a large set of messages which can be found in common.xml and ardupilot.xml. MAVLink messages can be sent over almost any serial connection and does not depend upon the underlying technology (wifi, 900mhz radio, etc). The messages are not guaranteed to be delivered which means ground stations or companion computers must often check the state of the vehicle to determine if a command has been executed.

Anaconda

Anaconda is a free and open-source distribution of the Python and R programming languages for scientific computing (data science, machine learning applications, large-scale data processing, predictive analytics, etc.), that aims to simplify package management and deployment. It provides large selection of packages and commercial support. It is an environment manager, which provides the facility to create different python environments, each with their own settings. We created an environment named “underwater” to manage all the prerequisite libraries and tools for running the body detection program.

Fig 3.27 MAVLINK and ANACONDA

SPYDER IDE

Spyder is an open source cross-platform integrated development environment (IDE) for scientific programming in the Python language. It features a unique combination of the advanced editing, analysis, debugging and profiling functionality of a comprehensive development tool with the data exploration, interactive execution, deep inspection and beautiful visualization capabilities of a scientific package. Furthermore, Spyder offers built-in integration with many popular scientific packages, including NumPy, SciPy, Pandas, IPython, QtConsole, Matplotlib, SymPy, and more. Beyond its many built-in features, Spyder can be extended even further via third-party plugins. Spyder can also be used as a PyQt5 extension library, allowing you to build upon its functionality and embed its components, such as the interactive console or advanced editor, in your own software.

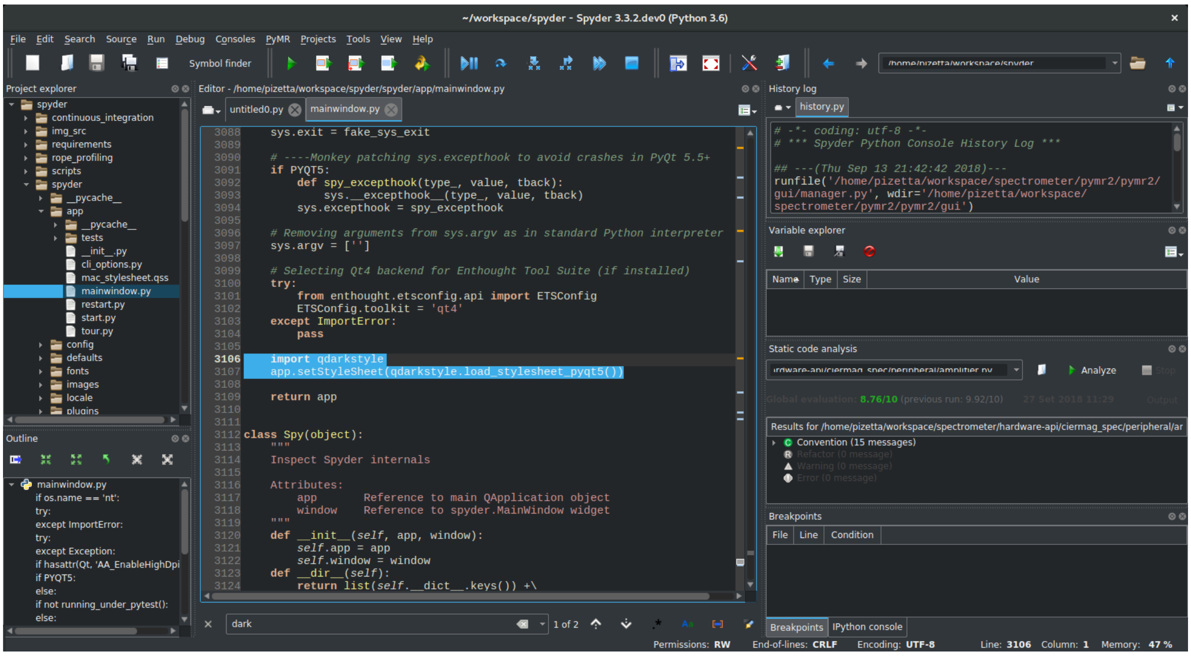


Fig 3.28 Spyder IDE

RASPBIAN OS

Raspbian is a Debian-based computer operating system for Raspberry Pi. There are several versions of Raspbian including Raspbian Stretch and Raspbian Jessie. Since 2015 it has been officially provided by the Raspberry Pi Foundation as the primary operating system for the family of Raspberry Pi single-board computers. Raspbian is a Linux Distribution. Anything that is built on top of the Linux Kernel can be called a Linux Distribution. We used the raspberry pi board to stream real time video, which has the Raspbian OS installed in it.

OPENCV

OpenCV (Open source computer vision) is a library of programming functions mainly aimed at real-time computer vision. Originally developed by Intel, The library is cross-platform and free for use under the open-source BSD license. OpenCV supports the deep learning frameworks like TensorFlow, Torch/ PyTorch and Caffe. OpenCV's application areas include: 2D and 3D feature toolkits, Egomotion estimation, Facial recognition system, Gesture recognition, Human–computer interaction (HCI), Mobile robotics, Motion understanding, Object identification, Segmentation and recognition, Stereopsis stereo vision: depth perception from 2 cameras, Structure from motion (SFM), Motion tracking, Augmented reality etc. To support some of these areas, OpenCV includes a statistical machine learning library that contains: Boosting, Decision tree learning, Gradient boosting trees, Expectation-maximization algorithm, k-nearest neighbour algorithm, Naive Bayes classifier, Artificial neural networks, Random forest, Support vector machine (SVM) etc.

OpenCV is written in C++ and its primary interface is in C++, but it still retains a less comprehensive though extensive older C interface. There are bindings in Python, Java and MATLAB/OCTAVE. The API for these interfaces can be found in the online documentation.[12] Wrappers in other languages such as C#, Perl, Haskell, and Ruby have been developed to encourage adoption by a wider audience. All of the new developments and algorithms in OpenCV are now developed in the C++ interface.

Fig 3.29 OpenCV and Raspbian

SOLIDWORKS

SolidWorks is a solid modelling computer-aided design (CAD) and computer-aided engineering (CAE) computer program that runs on Microsoft Windows. SolidWorks is published by Dassault Systèms. SolidWorks is a solid modeler, and utilizes a parametric feature-based approach which was initially developed by PTC (Creo/Pro-Engineer) to create models and assemblies. The software is written on Parasolid-kernel. Building a model in SolidWorks usually starts with a 2D sketch (although 3D sketches are available for power users). The sketch consists of geometry such as points, lines, arcs, conics (except the hyperbola), and splines. Dimensions are added to the sketch to define the size and location of the geometry. Relations are used to define attributes such as tangency, parallelism, perpendicularity, and concentricity. The parametric nature of SolidWorks means that the dimensions and relations drive the geometry, not the other way around. The dimensions in the sketch can be controlled independently, or by relationships to other parameters inside or outside of the sketch.

ANSYS

Ansys develops and markets finite element analysis software used to simulate engineering problems. The software creates simulated computer models of structures, electronics, or machine components to simulate strength, toughness, elasticity, temperature distribution, electromagnetism, fluid flow, and other attributes.

TENSORFLOW

Currently, the most famous deep learning library in the world is Google's TensorFlow. TensorFlow is a library developed by the Google Brain Team to accelerate machine learning and deep neural network research.

It was built to run on multiple CPUs or GPUs and even mobile operating systems, and it has several wrappers in several languages like Python, C++ or Java.

Tensorflow architecture works in three parts:

- Preprocessing the data
- Build the model

- Train and estimate the model

It is called Tensorflow because it takes input as a multi-dimensional array, also known as **tensors**. You can construct a sort of **flowchart** of operations (called a Graph) that you want to perform on that input. The input goes in at one end, and then it flows through this system of multiple operations and comes out the other end as output. **Faster R-CNN**

Fig 3.30 TensorFlow

Components of TensorFlow:

Tensor

Tensorflow's name is directly derived from its core framework: Tensor. In Tensorflow, all the computations involve tensors. A tensor is a vector or matrix of n-dimensions that represents all types of data. All values in a tensor hold identical data type with a known (or partially known) **s**hape. The shape of the data is the dimensionality of the matrix or array.

A tensor can be originated from the input data or the result of a computation. In TensorFlow, all the operations are conducted inside a graph. The graph is a set of computation that takes place successively. Each operation is called an op node and are connected to each other.

The graph outlines the ops and connections between the nodes. However, it does not display the values. The edge of the nodes is the tensor, i.e., a way to populate the operation with data.

Graphs

TensorFlow makes use of a graph framework. The graph gathers and describes all the series computations done during the training. The graph has lots of advantages:

- It was done to run on multiple CPUs or GPUs and even mobile operating system
- The portability of the graph allows to preserve the computations for immediate or later use. The graph can be saved to be executed in the future.
- All the computations in the graph are done by connecting tensors together
    - A tensor has a node and an edge. The node carries the mathematical operation and produces an endpoint's outputs. The edges the edges explain the input/output relationships between nodes.

Tensorflow Object Detection API

Creating accurate machine learning models capable of localizing and identifying multiple objects in a single image remains a core challenge in computer vision. The TensorFlow Object Detection API is an open source framework built on top of TensorFlow that makes it easy to construct, train and deploy object detection models.

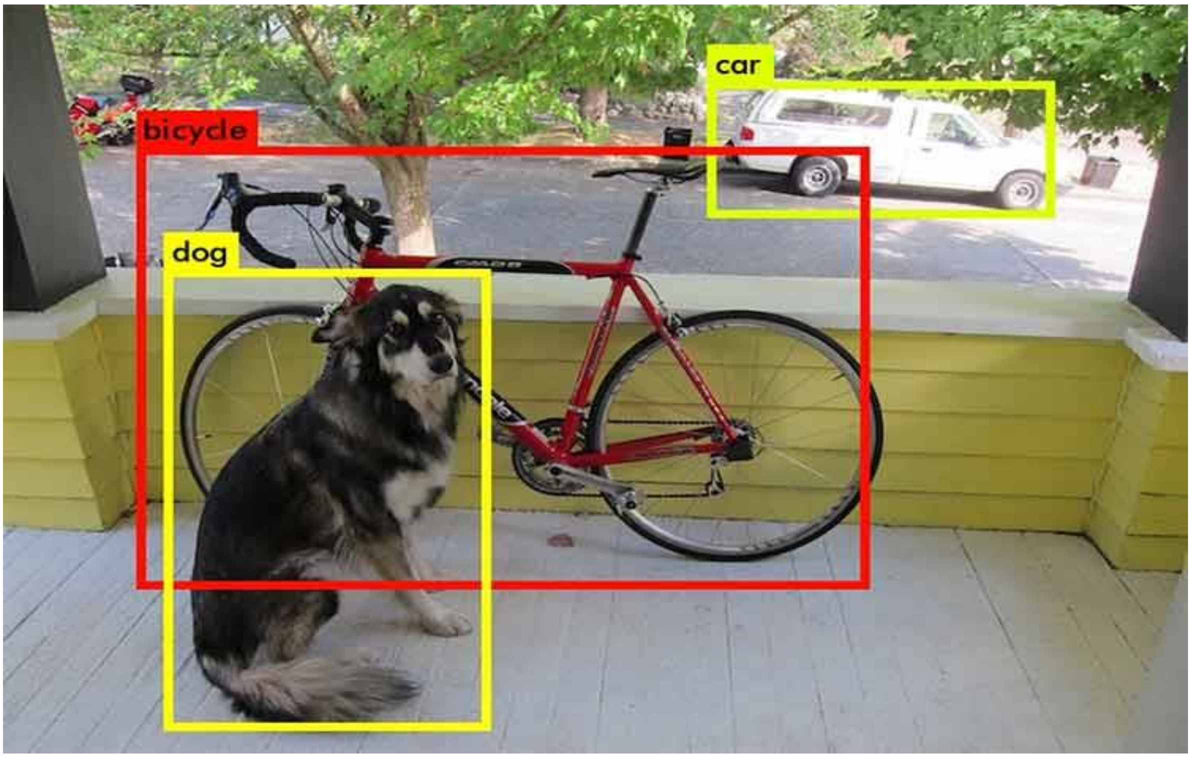


Fig 3.31 Object detection using tensorflow

3.2.4 Programming section

Object detection module

```
# -*- coding: utf-8 -*-
"""
Created on Wed May  1 12:01:48 2019
"""


import numpy as np
import tensorflow as tf
import cv2
import time


class DetectorAPI:
    def __init__(self, path_to_ckpt):
        self.path_to_ckpt = path_to_ckpt

        self.detection_graph = tf.Graph()
        with self.detection_graph.as_default():
            od_graph_def = tf.GraphDef()
            with tf.gfile.GFile(self.path_to_ckpt, 'rb') as fid:
                serialized_graph = fid.read()
                od_graph_def.ParseFromString(serialized_graph)
                tf.import_graph_def(od_graph_def, name='')

        self.default_graph = self.detection_graph.as_default()
        self.sess = tf.Session(graph=self.detection_graph)
```

```
# Definite input and output Tensors for detection_graph
    self.image_tensor = self.detection_graph.get_tensor_by_name('image_tensor:0')
    # Each box represents a part of the image where a particular object was detected.
    self.detection_boxes = self.detection_graph.get_tensor_by_name('detection_boxes:0')
    # Each score represent how level of confidence for each of the objects.
    # Score is shown on the result image, together with the class label.
    self.detection_scores = self.detection_graph.get_tensor_by_name('detection_scores:0')
    self.detection_classes = self.detection_graph.get_tensor_by_name('detection_classes:0')
    self.num_detections = self.detection_graph.get_tensor_by_name('num_detections:0')

  def processFrame(self, image):
    # Expand dimensions since the trained_model expects images to have shape: [1, None, None,
3]
    image_np_expanded = np.expand_dims(image, axis=0)
    # Actual detection.
    start_time = time.time()
    (boxes, scores, classes, num) = self.sess.run(
      [self.detection_boxes, self.detection_scores, self.detection_classes, self.num_detections],
      feed_dict={self.image_tensor: image_np_expanded})
    end_time = time.time()

    print("Elapsed Time:", end_time-start_time)

    im_height, im_width,_ = image.shape
    boxes_list = [None for i in range(boxes.shape[1])]
    for i in range(boxes.shape[1]):
      boxes_list[i] = (int(boxes[0,i,0] * im_height),
            int(boxes[0,i,1]*im_width),
            int(boxes[0,i,2] * im_height),
            int(boxes[0,i,3]*im_width))
```

```
        return boxes_list, scores[0].tolist(), [int(x) for x in classes[0].tolist()], int(num[0])

    def close(self):
        self.sess.close()
        self.default_graph.close()
```

Frame Grabber module

```
# -*- coding: utf-8 -*-
"""
Created on Wed May  1 12:34:41 2019
"""

import socket
import numpy as np
import struct
from threading import Thread, Lock

class ImageGrabber(Thread):
    def __init__(self, host, port):
        Thread.__init__(self)
        self.lock = Lock()
        self.sock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
        self.server_address = (host, port)
        self.sock.settimeout(0.2)
        self.array = None
```

```
    self.running = True

def stopServer(self):
    self.sock.sendto(struct.pack('<L',0), self.server_address)

def imageGrabber(self):
    if self.array is not None:
        self.lock.acquire()
        array = self.array
        self.lock.release()
        return array

def run(self):
    while self.running:
        try:
            self.sock.sendto(struct.pack('<L',1), self.server_address)
            try:
                data_len_packed, server = self.sock.recvfrom(struct.calcsize('<L'))
            except socket.timeout:
                continue
            data_len = struct.unpack('<L',data_len_packed)[0]
            if data_len < 65507:
                print ("byte recv: ", data_len)
            try:
                data, server = self.sock.recvfrom(data_len)
            except socket.timeout:
                continue
            except Exception:
                continue
```

```
        if not len(data) == data_len:
            print ("There was a image packet loss...")
            continue
        if data == 404:
            continue
        self.lock.acquire()
        self.array = np.frombuffer(data, dtype=np.dtype('uint8'))
        self.lock.release()

    except:
        self.running = False
        self.lock.acquire()
        self.image = None
        self.lock.release()
```

Client program (UDP)

```
Created on Tue Apr 30 21:55:41 2019

@author: Ashik P
"""

#!/usr/bin/env python
#pisender side server

import socket
import cv2
from threading import Thread, Lock
import struct
```

```
debug = True

jpeg_quality = 70#30

host = '127.0.0.1'

port = 65534

#host = ''       # Symbolic name meaning all available interfaces

##############socket for reverse result ##### ######### ########################
###########


###################################################################################
#

class VideoGrabber(Thread):

    def __init__(self, jpeg_quality):

        Thread.__init__(self)

        self.encode_param = [int(cv2.IMWRITE_JPEG_QUALITY), jpeg_quality]

        self.cap = cv2.VideoCapture(0)

        self.cap.set(3, 720)#640

        self.cap.set(4, 480)#480

        self.running = True

        self.buffer = None

        self.lock = Lock()


    def stop(self):

        self.running = False


    def get_buffer(self):


        if self.buffer is not None:

            self.lock.acquire()

            cpy = self.buffer.copy()

            self.lock.release()
```

```
            return cpy

    def run(self):
        while self.running:
            success, img = self.cap.read()
            #cv2.imshow("img",img)
            if not success:
                continue
            self.lock.acquire()
            result, self.buffer = cv2.imencode('.jpg', img, self.encode_param)
            self.lock.release()

grabber = VideoGrabber(jpeg_quality)
grabber.daemon = True
grabber.start()

running = True

sock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
##########################################################################################################################################
"""client_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
client_socket.bind(('127.0.0.1', 12345))
client_socket.listen(1)
print("Starting connection for result")"""
##########################################################################################################################################
# Bind the socket to the port
server_address = (host, port)
```

```
print('starting up on %s port %s\n' % server_address)


sock.bind(("127.0.0.1",65534))#sock.bind(server_address,port)
print("Started")
"""connection,client_address=client_socket.accept()
if connection:
   print("Connected to recieve result")"""


while running:
   data_packed, address = sock.recvfrom(struct.calcsize('<L'))
   data = struct.unpack('<L',data_packed)[0]
   if(data == 1):
         buffer = grabber.get_buffer()
         if buffer is None:
               continue
         if len(buffer) > 65507:
            print ("too large sorry")
            sock.sendto(struct.pack('<L',struct.calcsize('<L')), address)
            sock.sendto(struct.pack('L',404), address) #capture error
            continue
         sock.sendto(struct.pack('<L',len(buffer)), address)
         sock.sendto(buffer.tobytes(), address)
   elif(data == 0):
         grabber.stop()
         running = False
   """try:
      data_recieved=connection.recv(16)
      print(data_recieved)
   except:
```

```
    continue"""


print("Quitting..")
grabber.join()
sock.close()
```

Server program

```
# -*- coding: utf-8 -*-
"""
Created on Wed May  1 12:36:52 2019
"""

import grabframe
import detect
import cv2
autoMaxPower = 1.0                  # Maximum output in automatic mode
autoMinPower = 0.2                  # Minimum output in automatic mode
autoMinArea = 10                    # Smallest target to move towards
autoMaxArea = 1000000               # Largest target to move towards
autoFullSpeedArea = 300

if __name__ == '__main__':
    model_path = 'C:/Users/Ashik P/Desktop/project_test1/models/frozen_inference_graph.pb'
    odapi = detect.DetectorAPI(path_to_ckpt=model_path)
    threshold = 0.7
    image = grabframe.ImageGrabber('127.0.0.1', 65534)
```

```
    image.daemon = True
    image.start()
    try:
        while image.running:
            array = image.imageGrabber()
            if array is None:
                continue
            img = cv2.imdecode(array, 1)
            cv2.imshow("original",img)
            found_area=-1
            #r, img = cap.read()
            img = cv2.resize(img, (1280, 720))
        #print(img.shape[0])
        #print(img.shape[1])

            boxes, scores, classes, num = odapi.processFrame(img)

# Visualization of the results of a detection.

            for i in range(len(boxes)):
                if classes[i] == 1 and scores[i] > threshold:
                    box = boxes[i]
                    area=(((box[1]-box[3])*(box[1]-box[3]))-(box[0]-box[2])*(box[0]-box[2]))/2
                    print("%0.2f %0.2f" %(box[1],box[0]))
                    if found_area < area:
                        found_area = area
                        foundX=(box[1]+box[3])/2
                        foundY=(box[0]+box[2])/2
            if(found_area>0):
```

```
        cv2.rectangle(img,(box[1],box[0]),(box[3],box[2]),(255,0,0),2)
        print("%0.2f %0.2f"%((box[1]+box[3])/2,(box[0]+box[2])/2))
        cv2.circle(img,(int((box[1]+box[3])/2),int((box[0]+box[2])/2)),10,(255,255,255),-1)
        ball=[foundX,foundY,found_area]
    else:
        ball=None
    driveLeft  = 0.0
    driveRight = 0.0
    if ball:
        x = ball[0]
        area = ball[2]
        print (x)
        if area < autoMinArea:
            print ('Too small / far')
        elif area > autoMaxArea:
            print ('Close enough')
        else:
            if area < autoFullSpeedArea:
                speed = 1.0
            else:
                speed = 1.0 / (area / autoFullSpeedArea)
                speed *= autoMaxPower - autoMinPower
                speed += autoMinPower
                direction = (x - (img.shape[1]/2.0)) / (img.shape[1]/2.0)
            if direction < 0.0:
                print("Turn right")
                driveLeft  = speed
                driveRight = speed * (1.0 + direction)
            else:
```

```
                print("Turn left")
                driveLeft  = speed * (1.0 - direction)
                driveRight = speed
                print ('%.2f, %.2f' % (driveLeft, driveRight))
        else:
            print("No body")
        cv2.imshow("preview", img)

        if cv2.waitKey(1) & 0xFF == ord('q'):
            break
    except:
        image.stopServer()
image.sock.close()
```

Explanation

Server program runs on the laptop which gets the video steam from the client program running on the Raspberry Pi within the ROV. Transmission of the video frames take place in the form of UDP packets which is implemented through socket program. The Server program starts a server socket and starts listening to one of its port. It imports the" grab frame" and "detect" module. The grab frame module reconstructs each frame from the serially received string from the client socket. Each frame obtained evaluated using the trained tensorflow graph to find the presence of the object. The coordinates of the largest rectangle that encloses the object is obtained and the coordinates are found out. From the coordinates of the centre calculated we measure the relative distance from the X and the Y axis to find the displacement vector. The detect module has all the algorithms and functions for the detection of the object. The client program gets the feed from the camera and serialize the pixel information's into binary string. Once the client gets connected to the server port it starts sending the serial values to the server socket.

CHAPTER 4

IMPLEMENTATION

4.1 BLOCK LEVEL IMPLEMENTATION

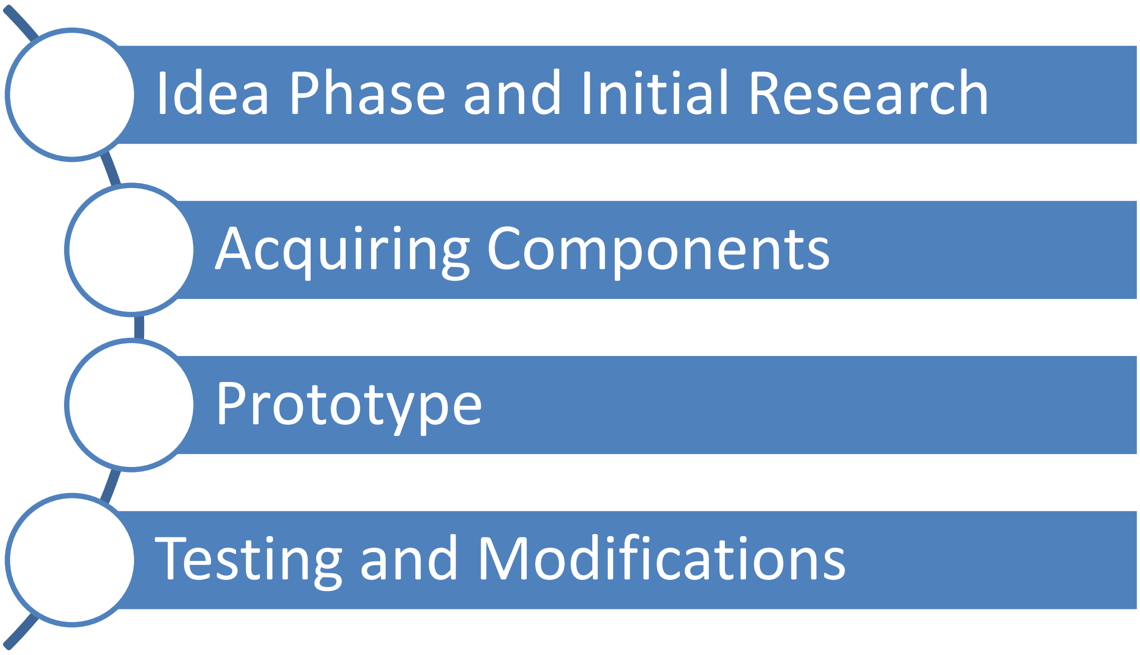


Fig 4.1 Implementation block

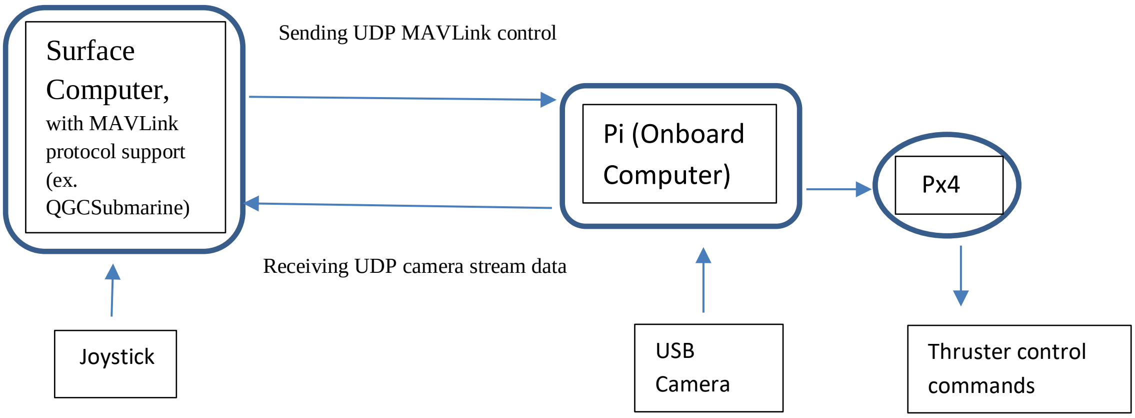


Fig 4.2 Working overview

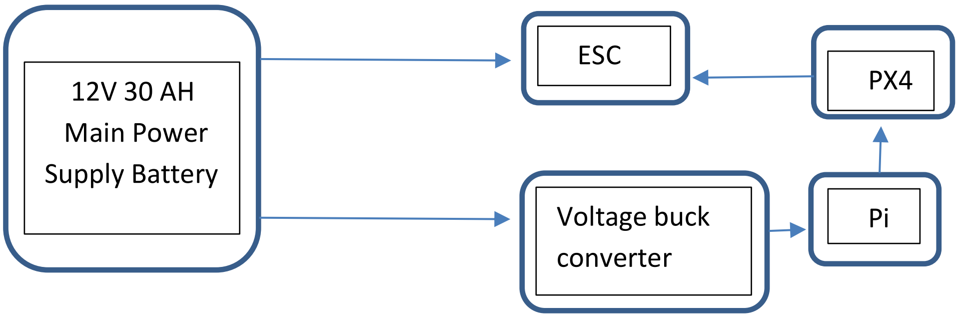


Fig 4.3 Power supply overview

4.2 BLOCK DESCRIPTION

In Idea phase and initial research phase, we conducted research and interviewed people working in related fields like fire force and rescue team. Research and team discussion meetings were conducted to know more about this topic.

We acquired the required components and made the first prototype. The prototype photos were given above in the hardware section 3.1.2 above.

Surface computer is a laptop computer with a decent processor (core i5) connected to the onboard computer (inside the aquatic vehicle) which is a Raspberry Pi 3 B+ board. Surface computer has support for MAVLink protocol communication. For that we've used QGC Submarine and ArduSub (on Pi). USB camera sends video data to Pi, which converts it to UDP packets and then send to surface computer via LAN. Surface computer receives the data, makes the decision and give the control information (MAVLink format) as UDP packets to Pi. Pi communicates with PX4 via USB passing this information. PX4 then issue the received thruster control commands to the connected ESCs.

4.3 WORKING PRINCIPLE

This paper proposes the application of Unmanned Aerial and Aqua Vehicles (UAAV) for underwater human rescue operation. The UAAV (a drone) with an Autonomous Underwater Vehicle (AUV) will fly over the water body and land on the water surface. It then deploys the AUV in to the water. AUV swims in to the water searching real time for its target - the human body. Once the target is detected, the AUV locks the targets position and move towards the target to verify and make a decision: whether it is a human body or not. After the confirmation, the robot swims back vertically up to the water surface, from where it can be easily noticed by the ground rescue crew. The diver can then dive there, to find the human body. Alternately, an air bag could also be inflated after locking it to the target, which will float back to the surface carrying the target. This technology is very helpful for underwater human detection since we need not have to search the whole water body and hence both time and resources are saved.

Traditional procedures incorporating human divers searching underwater for human bodies (surviving or dead) are very inefficient due to visibility issues and the supply oxygen, which increases the delay and complexity in such operations. Hence developing a suitable mechanism to aid this is very much necessary. But research works in this field are advancing at a comparatively low pace, mainly due the difficulty and complexity in imaging and sensing technologies that could be employed under water. One of the main challenges in underwater communication is the inefficiency of Electromagnetic wave communication inside water. Water absorbs electromagnetic radiations and an information could not be easily transmitted across, without suffering from loss. Hence we have to depend on acoustic waves for underwater communication. One such approach is to power a beam of sound energy (ultrasound) at the assumed target location and calculate the absorption of an underwater body by looking at the received power of the signal reflected from the body. The material can then be recognised by suitable training. Another approach is to focus on a special case of moving object detection under water in which the rate of motion degrades slowly. Since there is also possibility of change in degree of motion for other objects also due to water movement, this method comes with the cost of spending a lot of time.

But all these methods have some serious disadvantages. Since the mass of the immersed body is far smaller than the mass of the water body, temperature of the body will level down or up to the water temperature. once the temperature of the 'body' equals the temperature of the body, infrared does not help. Also, in unstructured environments, current underwater robots are limited in their ability to localize and track human bodies because optical cameras can be rendered useless by the turbidity and murkiness of water. Localizing radio signals also do not propagate well through the water medium, and acoustic positioning systems can be very expensive to deploy. So a technique for underwater human detection must be developed which is economically and technologically feasible.

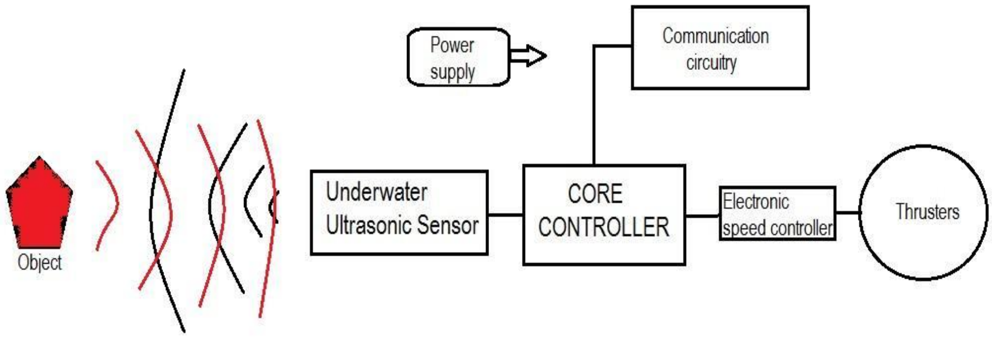


Fig 4.4 Underwater sound sensing and control

Under water ultrasonic sensor is interfaced with the core controller of ROV. Since it requires good computational capacity, Raspberry pi is preferred as the core controller. The power supply section provides the required electric power for various parts of ROV. The ultrasonic sensor will give the analog signal to the controller. For the classification process, the data from sensor to be converted to digital format. Hence this require an ADC which is commonly integrated to the same controller chip. The communication circuitry is employed for transmitting the position and the result (ie. whether body is detected or not) to the surface of water body. Electrical thrusters are controlled by electronic speed controller (ESC) which is similar to the pilot board of a drone. The speed and direction of thrusters are controlled by the signals provided by the core controller to ESC.

4.4 TESTING OF ROV

Testing of ROV was done on 11 June 2019 (2:00 PM- 4:00 PM). Swimming pool situated inside Childrens Park, Kollam was the test venue. Various tests were done on ROV including linear motion, rotation and exposure to pressure. When submerged ROV was submerged to a pressure of 2 bar. Neither the structure nor the sealing of ROV was affected by this pressure

Fig 4.5 Testing of ROV in a pool

6.1 Velocity in forward direction

ROV was moved in forward direction. Time taken for covering one meter distance was measured. This experiment was repeated for 3 times to reduce error.

Attempt Number (no)	Time (sec)
1	4.54
2	4.58
3	4.51

Table 4.1 Forward velocity data

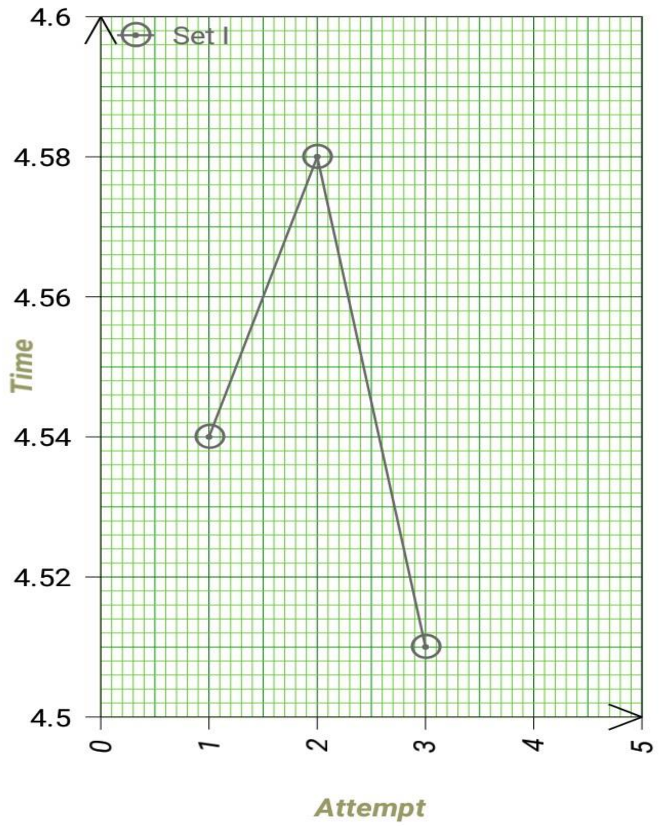


Fig 4.6 Time Attempt graph

Mean time= (4.51+4.54+4.58)/3= 4.543 sec

Speed of ROV in forward direction= Distance/Mean Time

=1/ 4.543

= .2201 m/sec (22.01 cm/sec)

6.2 Velocity in reverse direction

ROV was moved in reverse direction. Time taken for covering one meter distance was measured. This experiment was repeated for 3 times to reduce error.

Attempt Number (no)	Time (sec)
1	4.69
2	4.72
3	4.74

Table 4.2 Reverse Velocity Data

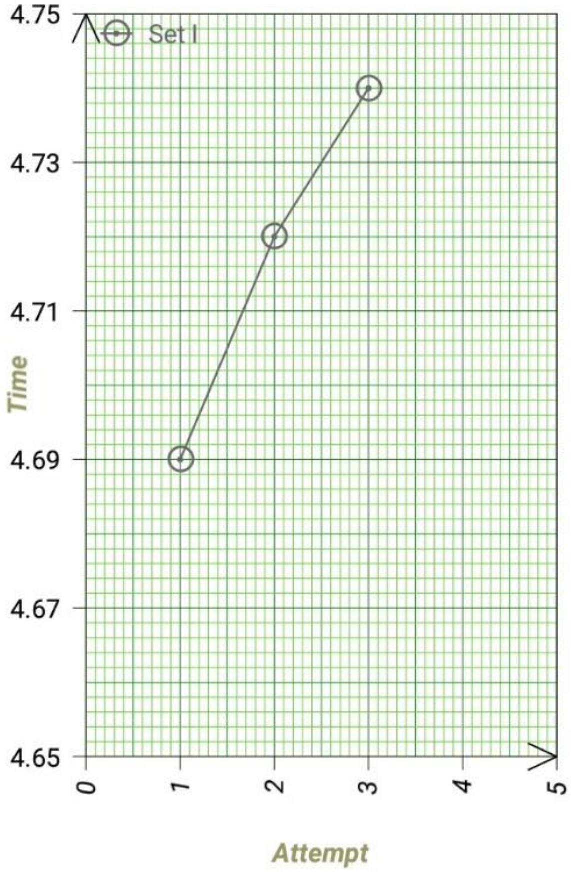


Fig 4.7 Time Attempt graph

Mean time= (4.69+4.72+4.74)/3= 4.716 sec

Speed of ROV in reverse direction= Distance/Mean Time

=1/ 4.716

= .2120 m/sec (21.20 cm/sec)

6.3 Rotation-time graph

In this experiment time taken for 90 degree rotation of ROV is measured.

Attempt Number (no)	Time (sec)
1	3.86
2	4.12
3	3. 97

Table 4.3 Rotation Time data

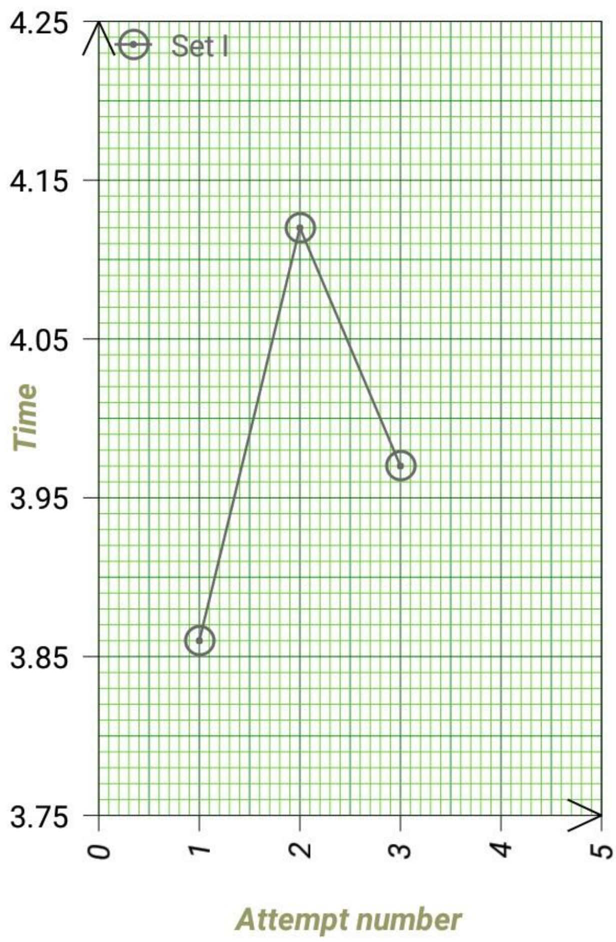


Fig 4.8 Time Attempt graph

Time taken for ROV to take 90 degree turn (Mean time)= (3.86+4.12+3.97)/3

=3.983 sec

Time taken for 1 degree turn= 3.983/90

= .0442 sec

Rotation gradient= .0442 sec/deg

4.5 CONCEPT MODEL

This is the model which we have designed for future. This aerodynamic model has high efficiency due to its low weight. We couldn’t fabricate it due to time and financial constraints. In this model T100 thruster is used for providing motion.

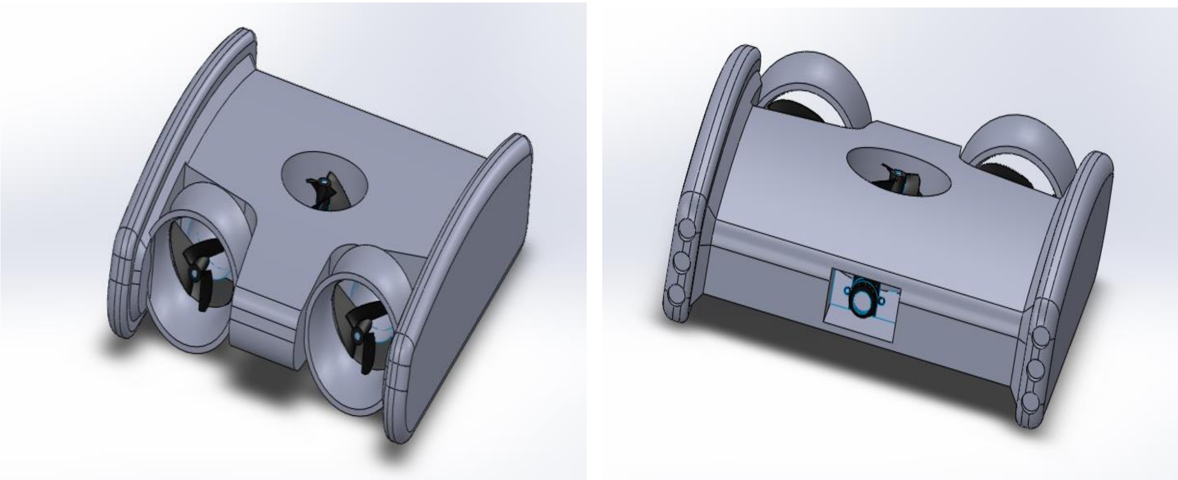


Fig 4.9 Solidworks model

In this design T100 thrusters are used instead of bilge pumps for providing motion

Forward Thrust= 2.36 kgf

There are 2 thrusters in horizontal direction and one thruster in horizontal direction

Mass of ROV is 1.5 kg

Acceleration in horizontal direction= (Forward thrust X 2)/mass

=(2 X 2.36)/1.5

=3.147 m/s2

Acceleration in vertical direction+ Forward thrust/ mass

=2.36/1.5

=1.573 m/s2

CHAPTER 5

RESULTS

1. Human detection was carried out both on pre-recorded videos and live USB camera videos.

2. Successfully controlled brushed motors with their esc with and without Pixhawk.

3. Verified that, our objective can be achieved by following our current methodology with minor modifications.

However we faced difficulties in situations like:
1. High cost of thrusters (brushless DC motors and their ESC's)
(approx. 10k for 1 with esc), so we have to remove cut the body of bilge pump, remove impellers, and connect suitable propellers. Getting the speed controllers for brushed motors was not easy. Had to import from China.
2. Water proof sealed casing for the AUV
3. Underwater human detection processor loading (on Pi)
4. Developing a human body prediction system (using deep learning)
5. Controlling the ESC's using Pixhawk
6. Testing under water.

5.1 EXPENSES INCURRED

Item	Rating/Dimension	Cost(INR)
Raspberry Pi 3 B+	5V 2A	3000
1100 GPH Bilge Pump	12V 3A	4400 (4x1100)
Brushed motor ESC	(6-12)V 320A(burst)	5200 (4x1300)
Amaron Car Battery	12V 30AH	3000
Chassis	50x40x30cm	
And other miscellaneous expenses		4400
Total Expense		20,000 (Approx)

Table 5.1 Expenses incurred

5.2 FUTURE SCOPE

In our model, we have designed an ROV that is capable of scanning the rivers and locating human bodies. This will reduce the time required for triangulating the location of the body. But still, for rescuing the body, an expert diver must dive in and carry the body back. In the future, a hydraulic arm system can be made and attached to the ROV that can carry the body to the water surface. This will be a complete underwater human body rescuing system and will not require any divers. Human intervention required is minimum and hence the risk of divers losing their life is greatly reduced. Another advancement that can be made to this design is using an electromagnetic coupling system to attach the ROV with a drone. In this way, the ROV needs to travel less distance under water. When a particular area has been completely searched, the ROV will come up to the surface and couple with the drone. The drone will carry the ROV to the next area and deploy it under water. Both these alterations will improve the efficiency of underwater human body detection and make it faster.

Apart from human detection ROV will be used as full time surveillance mechanism. Illegal entry of boats to the coast will be detected. In this case coastal guard and navy will be alarmed about the intruders. Swarm technology will be implemented in the near future. Here multiple ROV’s are linked to each other. Search and rescue mission will be more wide and efficient.

To promote eco-friendly measures of various NGO’s, renewable sources of energy will be utilised. Sources such as solar energy will be used for working of ROV. Solar panels harness the energy of sun for the working of thrusters and electronic components. Life of solar panel is high. Hence less maintenance is required for a ROV full time involved in water. Other options worth considering are a completely autonomous aquatic vehicle with improved imaging under low light conditions and improved Fluid Dynamics.

CONCLUSION

Underwater human detection has been a distant dream for the past few decades. But with the advancement of technology and research studies in the past decade in the field of underwater communication, acoustics and signal processing, the goal of underwater human detection is no longer a distant dream. Several very good research works have been published in the last decade which continue to inspire the present research works. We designed, implemented and tested our prototype in pool and pond water and the detection accuracy were satisfactory. Our design is very low cost and is only about 10-20% of the current making costs of an AUV. The overall making cost of an AUV can go nearly up to INR 1,00,000 or way beyond that. We require advanced acoustic imaging setup like side scan SONAR for human detection purposes which are very much costly and make underwater human detection economically unfeasible. So, in the present work, we are trying to develop a low cost AUV for underwater human detection.

We are currently working on developing a low cost acoustic detection solution for integrating with AUV for more accurate underwater human detection. Our work was aimed at finding a suitable and fool-proof instrument to rescue a drowning human body underwater with accuracy. We believe, it will have immense application for fire and rescue team who are currently the authorities for handling this kind of operations. We are currently working on ways to improve the fluid dynamics and control system and also on an improved anomaly detection algorithm for better surveillance, which gives it the potential to be extended for further operations like surveying and under water resource and vegetation analysis.

REFERENCES

1. WHO facts about drowning- url:https://www.who.int/newsroom/factsheets/detail/drowning
2. Underwater human body detection using computer vision algorithm, Fatma Yasar, Huseyin.Kusetogullari, 2018 26th Signal Processing and Communications Applications Conference (SIU), July 2018, Izmir, Turkey, DOI:10.1109/SIU.2018
3. Ultrasonic Sensor-based Human Detector using One-class Classifiers, Sonia, Achyut Mani Tripathi, Rashmi Dutta Baruah and Shivashankar B. Nair, 2015 IEEE International Conference on Evolving and Adaptive Intelligent Systems (EAIS), January 2016, Douai, France, DOI: 10.1109/EAIS.2015.7368797.
4. Underwater Communications, Sameer Babu T.P ,Sunil Kumar S, Naval Physical and Oceanographic Laboratory, Cochin, India, 2015 IEEE Underwater Technology (UT), May 2015, Chennai, India, DOI: 10.1109/UT.2015.7108234
5. RF Communication between Surface and Underwater Robotic Swarms, Ted Shaneyfelt, Matthew A. Joordens, Kranthimanoj Nagothu, Mo Jamshidi, 2008 World Automation Congress, December 2008, Hawaii, USA.
6. Design, analysis and control of an autonomous underwater surveillance robot, S. Udupa, N. Joshi & S. Raman, IEEE International Conference on Consumer Electronics-Asia (ICCE- Asia), October 2017, Banglore, India, DOI: 10.1109/ICCE-ASIA.2017.8307847
7. A Combination Processing by Imaging Method for the Underwater Moving Target Detection, Y. Jua, Xu Feng, Fourth International Symposium on Information Science and Engineering, December 2012, Shanghai, China 10.1109/ISISE.2012.116
8. Remotely operated underwater vehicle with surveillance system, B. Anwar, M. Ajim & S. Alam, International Conference on Advances in Electrical Engineering (ICAEE), Dec 2015, Dhaka, Bangladesh, DOI: 10.1109/ICAEE.2015.7506844
9. Serdar Soylu, Alison A. Proctor, Ron P. Podhorodeski, Colin Bradley, Bradley J. Buckham (2016) Precise trajectory control for an inspection class ROV. Ocean Engineering 2016
10. Mauro Candeloro, Asgeir J. Sørensen, Sauro Longhi, Fredrik Dukan (2012) Observers for Dynamic Positioning of ROVs with Experimental Results IFAC Proceedings 2012
11. Fukushima and Hirose (2011). "Operation of underwater rescue robot anchor diver III during the 2011 Tohoku Earthquake and Tsunami". OCEANS'11
12. Enrico Anderlini, Gordon G. Parker, Giles Thomas (2018) Control of a ROV carrying an object. Ocean Engineering 2018
13. Satja Sivcev , Matja Rossi, Joseph Colemann, Gerrard Dooly, Edin Omerdic, Daniel Toal (2018) Fully automatic visual servoing control for work class marine intervention ROV's Control Engineering Practice 2018
14. Jonathan Teague, Michael J Allen, Tom B Scott(2018) The potential of low cost ROV for use in deep sea mineral or prospecting and monitoring . Ocean Engineering 2018
15. Paul G Thomson ,Peter I Macreadi, Dianne L McLean(2017) Ice in the sea; unlocking the mysteries of the ocean using industrial remotely operated vehicles (ROV's). Science of the total environment 2017