Development of Collaborative Autonomous Underwater Systems

Final Report

Team C:
Peter Roberts
Rochak Chadha
Suraj Joseph
Suresh Gopalakrishnan
Abstract

This paper provides a description of the development of the Submarine Tracking Interacting Group of Autonomous Intelligent Systems project, also known as STINGRAIS.

STINGRAIS has been conceptualized, designed and developed by Team C for the MRSD Project Course 2011-12.

The primary objective was to design an autonomous system of mobile underwater robots which could collaboratively search for, detect and track a submarine. The system consists of one Autonomous Underwater Vehicle (AUV), a stationary Underwater Detection System (UDS) and a Ground Control Station (GCS). The UDS and the GCS communicate with each other using a Universal Serial Bus (USB) connection, and the AUV communicates with the GCS using a wireless link.

This report gives a detailed description of the full system requirements, functional architecture, physical architecture, and subsystems developed and concludes with the results obtained and lessons learnt by the team over the course of this project.
VI. Components 33

VII. Results and conclusions 34
   7.1. Challenges faced 34
   7.2. System performance evaluation 34
      7.2.1. Strengths 36
      7.2.2. Weaknesses 36
   7.3. Conclusions and Lessons learnt 36
   7.4. Future work 37

References 38

Websites consulted 39

Appendices
   A: Identification of resources and requirements of the functional analysis. 40
   B: Hull design. 44
   C: Details of all electrical subsystems. 50
List of Acronyms

ASW: Anti Submarine Warfare.
CAD: Computer-Aided Design.
CG: Centre of Gravity.
CMU: Carnegie Mellon University.
CPU: Central Processing Unit.
DC: Direct Current.
DIY: Do it yourself.
DOF: Degrees of Freedom.
EKF: Extended Kalman Filter.
FTDI: Future Technology Devices International.
GCS: Ground Control Station.
GUI: Graphical User Interface.
IMU: Inertial Measurement Unit.
LOS: Line of sight.
OpenCV: Open Source Computer Vision.
PoE: Power over Ethernet.
PCB: Printed Circuit Board.
ROV: Remotely Operated Vehicle.
PWM: Pulse-width modulation.
SBC: Single Board Computer.
STINGRAIS: Submarine Tracking Interacting Group of Autonomous Intelligent Systems.
TTL: Transistor-Transistor Logic.
UDS: Underwater Detection System.
USB: Universal Serial Bus.
## List of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Graphical representation of the system.</td>
<td>10</td>
</tr>
<tr>
<td>2</td>
<td>Swimming pool zones for use case scenario.</td>
<td>11</td>
</tr>
<tr>
<td>3</td>
<td>System deployment in pool.</td>
<td>11</td>
</tr>
<tr>
<td>4</td>
<td>Human submarine.</td>
<td>11</td>
</tr>
<tr>
<td>5</td>
<td>Functional architecture of the system.</td>
<td>15</td>
</tr>
<tr>
<td>6</td>
<td>System’s high-level physical architecture.</td>
<td>18</td>
</tr>
<tr>
<td>7</td>
<td>Hydrodynamic model for AUV.</td>
<td>19</td>
</tr>
<tr>
<td>8</td>
<td>Motor testing results.</td>
<td>20</td>
</tr>
<tr>
<td>9</td>
<td>AUV CAD design.</td>
<td>20</td>
</tr>
<tr>
<td>10</td>
<td>AUV.</td>
<td>21</td>
</tr>
<tr>
<td>11</td>
<td>Internal Electrical system diagram.</td>
<td>21</td>
</tr>
<tr>
<td>12</td>
<td>Power distribution board schematic.</td>
<td>23</td>
</tr>
<tr>
<td>13</td>
<td>The software architecture of the AUV software, with 5 threads executing continuously.</td>
<td>24</td>
</tr>
<tr>
<td>14</td>
<td>AUV Localization.</td>
<td>25</td>
</tr>
<tr>
<td>15</td>
<td>Stationary Underwater Detection System</td>
<td>26</td>
</tr>
<tr>
<td>16</td>
<td>Colour detection.</td>
<td>27</td>
</tr>
<tr>
<td>17</td>
<td>Background subtraction.</td>
<td>27</td>
</tr>
<tr>
<td>18</td>
<td>Contour detection.</td>
<td>28</td>
</tr>
<tr>
<td>19</td>
<td>Range finding algorithm running in UDS</td>
<td>28</td>
</tr>
<tr>
<td>20</td>
<td>The GCS setup – laptop, joystick, Wi-Fi router and antenna.</td>
<td>29</td>
</tr>
<tr>
<td>21</td>
<td>The AUV GUI.</td>
<td>30</td>
</tr>
<tr>
<td>22</td>
<td>The UDS GUI.</td>
<td>30</td>
</tr>
<tr>
<td>23</td>
<td>Screenshot from one simulation.</td>
<td>31</td>
</tr>
<tr>
<td>24</td>
<td>Results from twenty consecutive simulations.</td>
<td>32</td>
</tr>
<tr>
<td>25</td>
<td>Path followed by the AUV that detects the submarine.</td>
<td>32</td>
</tr>
<tr>
<td>26</td>
<td>Transport cart loaded with system’s components.</td>
<td>36</td>
</tr>
</tbody>
</table>
Figure A.1: AUV State measurement requirements analysis. 40
Figure A.2: Submarine detection requirements analysis. 40
Figure A.3: Submarine tracking requirements analysis. 41
Figure A.4: Motion control requirements analysis. 41
Figure A.5: AUV Communications requirements analysis. 42
Figure A.6: GUI communications requirements analysis. 42
Figure A.7: GUI Displaying requirements analysis. 43
Figure B.1: Drag Coefficient at different L/d ratio. 45
Figure B.2: Drag Force at different speeds. 45
Figure B.3: Thrust Power required for different speeds. 46
Figure B.4: Lift force at different L/d ratio. 46
Figure B.5: AUV design evolution. 47
Figure B.6: Ballast weight. 48
Figure B.7: Internal components on acrylic board. 48
Figure B.8: AUV interior. 49
Figure C.1: Beagle Board SBC. 50
Figure C.2: Arduino Mega Development Board. 51
Figure C.3: Logitech Webcam. 51
Figure C.4: Sparkfun 9 DOF IMU. 52
Figure C.5: Pressure Sensor. 52
Figure C.6: Motor Driver 53
List of Tables

Table 1: System Requirements and Key Performance Metrics. 13
Table 2: Power System Requirements Estimation. 22
Table 3: Electronic Components Rationale. 23
Table 4: AUV Performance. 25
Table 5: UDS Performance. 29
Table 6: Components developed and acquired for the system. 33
Table 7: System evaluation. 34
I. Introduction

Submarines are one of the biggest threats in naval warfare. Hence there exists a need to improve Anti-Submarine Warfare (ASW) by incorporating technologies from other fields, or designing and developing specific elements to make the detection of submarines easier.

In recent years, submarines have also improved their performance, getting faster, quieter, diving deeper and staying submerged for longer durations.

These factors make searching for, detecting and tracking a submarine a long, tedious and dangerous process, which means robots are ideal candidates for this task. For this reason, many nations’ naval forces are showing a growing interest in using unmanned systems to aid ASW.

The aim of project STINGRAIS is to provide a system of autonomous underwater robots whose sole function is to search, detect and track an enemy submarine in a collaborative way.
II. Project Description

Team STINGRAIS’ goal is to provide a system that can search for, detect and track a submarine using autonomous underwater robots.

To fulfil this objective, the system will be composed of a stationary detection subsystem, whose mission is to detect the submarine when it crosses into the detection zone; a tracking AUV, whose function is to track the submarine while it travels through the surveillance area; and finally a user interface, which alerts the ground control station once a submarine has been detected and shows the target submarine while it is being tracked by the system. Figure 1 shows a graphical representation of the system.

![Graphical representation of the system.](image)

The system’s testing and demonstration was held at the Carnegie Mellon University (CMU) swimming pool, which was divided into 4 different areas, as shown in Figure 2. The submarine deployment zone is the region from which the submarine was deployed. The detection zone is where the detection subsystem was stationed at the edge of the pool, with its camera facing the other end. Both zones are separated by an imaginary line called the Patrol Border Line. The tracking zone is the region where the submarine is tracked by the AUV, which can move freely inside this zone. While no submarine is detected, the AUV remains on the surface, and it dives only once a submarine is detected, in order to track it underwater. The touch-down zone is at the opposite end of the pool from the submarine deployment zone, and the submarine’s goal is to enter this zone without being detected. Figure 3 shows a graphical representation of the system deployed in the pool.

Due to budget and time constraints, the target for system testing and demonstration was a “human submarine”. One of the team members dressed up in a green suit and swam across the pool, from the deployment zone to the touch-down zone. Figure 4 shows the human submarine in action, which will be referred to as the submarine for the purposes of this report.
Figure 2: Swimming pool zones for use case scenario.

Figure 3: System deployment in pool.

Figure 4: Human submarine.
The system’s first goal is to detect and identify the submarine, with the detection subsystem. The submarine enters the detection zone from an unknown point along the patrol border line, within the detection system’s range. Once the submarine is detected by the detection system, an alert is launched at the GCS, and the AUV is informed of the submarine’s presence. The AUV then goes into tracking mode and autonomously dives to a suitable depth to track the submarine. Once the AUV detects the submarine, it will follow it, using visual servoing techniques, along the whole tracking zone.

During the course of the mission, the direction of motion of the submarine is at all times towards the touch-down zone and it is unaware of the AUV’s position.
III. System Requirements

The requirements and performance metrics for STINGRAIS are shown in Table 1.

Table 1: System Requirements and Key Performance Metrics.

<table>
<thead>
<tr>
<th>Requirements/Key Performance Metrics</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Area of operation</td>
<td>300 m² (15 m x 20 m)</td>
</tr>
<tr>
<td>Patrol Border length</td>
<td>15 m</td>
</tr>
<tr>
<td>Operating Depth</td>
<td>1 m to 3 m</td>
</tr>
<tr>
<td>Maximum Depth</td>
<td>4 m</td>
</tr>
<tr>
<td>Operating Temperature</td>
<td>26.7°C to 30.5°C</td>
</tr>
<tr>
<td>Number of submarines to be detected</td>
<td>1</td>
</tr>
<tr>
<td>Speed of the submarine</td>
<td>0.6 m/s</td>
</tr>
<tr>
<td>Maximum possible single mission duration</td>
<td>2 min</td>
</tr>
<tr>
<td>Turn Cycle Time (assuming no failures)</td>
<td>Under 30 minutes</td>
</tr>
<tr>
<td>GCS equipment weight</td>
<td>Less than 100 lbs.</td>
</tr>
<tr>
<td>Detection success rate (in an event of submarine entering system’s detection range)</td>
<td>90%</td>
</tr>
<tr>
<td>Individual subsystem’s detection range</td>
<td>5 m.</td>
</tr>
<tr>
<td>Position of the submarine to be reported</td>
<td>Cartesian with respect to pool’s top corner</td>
</tr>
<tr>
<td>Accuracy of AUV and submarine position</td>
<td>Inside a 2 m radius circle of the measured position</td>
</tr>
</tbody>
</table>

Besides these, other requirements which also need to be met are:

- System should be able to move within area of operation
- Mission begins as soon as all the underwater systems and surface systems are deployed and fully operational.
- System needs to be portable by men without the need of any special lifting machines.
- Display the last measured position and past known trajectory of the submarine on the Graphical User Interface (GUI).
- Display the position of the AUVs on the GUI.
- Submarine must be tracked while in tracking zone

3.1. Subsystems Requirements

Based on all these requirements, the subsystem requirements can be determined.
3.1.1. Detection subsystem requirements

- Detection range: 0-5 m.
- Detection accuracy: 90%.
- Maximum time available for detection: 7 seconds.
- Report the submarine’s position to the GCS.
- Report the detection of the submarine to the GCS.
- Operating depth: 1 m.

3.1.2. Tracking subsystem requirements

- Subsystem must be on board an AUV, with a maximum speed of 0.16 m/s.
- Continuously track the submarine for at least 30 seconds after the initial detection by the detection subsystem.
- Not losing the contact for more than 3 seconds.
- Report the submarine’s position to the GCS.
- Report the tracking AUV position to the GCS.

3.1.3. User interface subsystem requirements

- Display the position of the submarine.
- Display the trajectory of the submarine, as a path of the known previous locations.
- Display the position of the AUV.
- Send emergency commands to the AUV. Ex: Surface immediately.
- Display internal status of the AUV. Ex: Temperature, leakages, battery level.
- Maximum communication range to AUV, Line of Sight (LOS): 35 m.
IV. Functional Architecture

The first level functions of the system are:

1. Detection.
2. Tracking.
3. Displaying the data.

These functions can be further divided into sub-functions, and then each of these can be divided as well, as shown in Figure 5.

Figure 5: Functional Architecture of the system.

The detection subsystem is composed of 2 main blocks:

1.1. Submarine Detection (Looking for the submarine)
1.2. Communications

The tracking subsystem is comprised of 5 main blocks:

2.1. Motion Control
2.2. AUV State Measurement
2.3. Submarine Tracking
2.4. Communications

The user interface is comprised of 2 main blocks
Details for each block are provided below. Each sub-function block (X,Y), has been evaluated by defining the necessary inputs, outputs, resources and constraints, according to [1]. This analysis is shown in Appendix “A”.

4.1. **AUV State Measurement Block (2.2)**

This block is responsible for the basic state estimation of the AUV and determines parameters such as the actual depth of the AUV using a pressure sensor, and the status of internal systems such as the battery voltage, leakage and temperature.

4.2. **Motion Control Block (2.1)**

This block is responsible for guiding the AUV to the desired position as commanded by the communication block. It utilizes the position estimated by the state measurement block and then gives the desired commands to the thrusters so that the AUV reaches the desired position as well as depth in the swimming pool.

4.3. **Looking for the Submarine Block (1.1)**

This block is responsible for detecting, identifying and locating the submarine in the swimming pool. It uses the camera in the UDS for the above tasks. A computer vision algorithm runs on the image captured by the camera. The algorithm detects objects in the image which can potentially be submarines. It then identifies the submarine from them. Once a submarine is identified, the GCS is notified. The algorithm also determines the position of the submarine in the swimming pool based on the image of the submarine.

4.4. **Communications Block (1.2 and 2.5)**

This block is responsible for the communication between the UDS and the GCS, and between the AUV and the GCS to pass key actionable intelligence to the end-user. The state of the AUV and position of the submarine are the key data elements exchanged via this block. Data between the AUV and GCS is exchanged using wireless routers (802.11-based) with an antenna floating on the surface of the swimming pool, connected to the AUV via a tether. The UDS and the GCS are connected using a USB tethered cable.

4.5. **Submarine Tracking Block (2.3)**

This block is responsible for continuously determining the position of the submarine in the swimming pool. Once the submarine has been identified and located by the Submarine Identification block (2.3.1), its position is passed to the Continuously Locate Submarine block (2.3.2), which continuously tracks the submarine’s position and passes it to the communications block, which reports it to the GCS.

4.6. **Communications Block (User Interface, 3.1)**

The communications block on the user interface end is responsible for enabling communications between the UDS, the AUV, and the GCS. It receives and transmits data such as the state of the AUV, operator commands and the position of the submarine in real-time between the various subsystems.
4.7. Display Block (3.2)

This block displays key actionable intelligence to the end user such as the position of the AUV and the position of the submarine. It also displays the AUV internal state parameters from block (2.2).
V. Physical Architecture and subsystems

5.1. Physical Architecture

The physical architecture of the system can be divided into three main elements, which are the detection subsystem, the tracking AUV and the GCS, as shown in Figure 6.

Each of these three main components corresponds to one subsystem of the whole system, which will be described in the following sections.

5.2. The AUV

The system has to work underwater and it is required to move within the area of operation. These two requirements involve the use of an underwater vehicle and, as the system is required to be autonomous, this vehicle must be unmanned and able to govern itself.

5.2.1. Mechanical design

Due to the speed requirements and the manoeuvrability necessary to keep tracking the target, the AUV needed a good hydrodynamic shape.

With this in mind, we used previous studies done on different AUVs ([2] to [7]) to develop a Matlab code for torpedo-shaped underwater vehicles, which can be used to estimate the key hydrodynamic parameters of a torpedo-shaped AUV based on its physical qualities like length, diameter, weight and speed. Next, we determined the best length-
diameter relationship (L/D ratio) that would give the vehicle a low drag. Figure 7 shows the GUI of the Matlab code and the results obtained for the AUV.

![Figure 7: Hydrodynamic model for AUV.](image)

The hull of the AUV was made with a 17 cm external diameter PVC pipe with a total length of 80 cm.

As the detection system will be based on a vision algorithm, the bow of the AUV was made with transparent acrylic in the shape of a dome. The back end cap was made out of polyethylene, two o-rings made a watertight seal so that it can be opened in order to insert and extract the internal components (electronics and batteries).

The internal electronic components were mounted on an acrylic board that slides inside the AUV through two aluminium rails. This allows an easy access to all the components in the AUV for debugging and future maintenance. The batteries were mounted on a steel bar that slides in the lower part of the AUV. This helps to lower the position of the centre of gravity (CG) inside the AUV, due to the batteries’ weight.

As for the propulsion, the vehicle has two motors fixed at the back, which give the AUV forward motion as well as steering ability. For depth control, two vertical motors were mounted aligned with the longitudinal component of the CG of the vehicle, which allow the AUV to make changes to its depth, without affecting its pitch angle. For these four motors, bilge pump motors were used, which are already waterproof, so we made shafts and couplings to attach the propellers to them and shrouds were made for the propellers.

The shrouds increase the efficiency of the propellers, in 10% and reduce the power consumption by a 20%. This was tested in the laboratory, and Figure 8 shows the comparison between motors with shrouds and without. The shrouds also serve as mounting for the motors and provide a protection for the blades of the propellers.
To mount the motors and also to increase the stability of the AUV, we made an aluminium frame and attached it to the lower part of the AUV. Ballast weight was added to the bottom part of the frame in order to lower the CG of the AUV, increasing its stability.

Figure 9 shows the Computer-Aided Design (CAD) design of the AUV and Figure 10 shows the finished AUV. A more detailed description of the hull design is described in Appendix “B”.

Figure 8: Motor testing results.
5.2.2. Internal Electrical System

The electrical system on the AUV is graphically described in Figure 11.
A detailed description of each of the above systems, how they are organized and their internal system layout in the AUV is described in Appendix “C”.

### 5.2.2.1. Power System

The system uses on-board power for autonomous operations. At the same time, it will be possible to power the system using off board power via a waterproof tether. A power distribution board is used to provide regulated power to all subsystems.

Table 2 shows an estimate for the power consumed by all the major subsystems on the AUV.

<table>
<thead>
<tr>
<th>Subsystem</th>
<th>Voltage Requirement (V)</th>
<th>Estimated Peak Current Consumption (A)</th>
<th>Power Consumed (W)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Single Board Computer</td>
<td>5</td>
<td>2.25</td>
<td>11.25</td>
</tr>
<tr>
<td>Inertial Measurement Unit</td>
<td>5</td>
<td>0.5</td>
<td>2.5</td>
</tr>
<tr>
<td>Arduino</td>
<td>5</td>
<td>1</td>
<td>5</td>
</tr>
<tr>
<td>Cameras</td>
<td>5</td>
<td>0.5</td>
<td>2.5</td>
</tr>
<tr>
<td>Wireless Routers</td>
<td>12</td>
<td>0.34</td>
<td>4.08</td>
</tr>
<tr>
<td>Pressure Sensor</td>
<td>5</td>
<td>0.1</td>
<td>0.5</td>
</tr>
<tr>
<td>Temperature Sensor</td>
<td>5</td>
<td>0.01</td>
<td>0.05</td>
</tr>
<tr>
<td>Leakage Sensor</td>
<td>5</td>
<td>0.01</td>
<td>0.05</td>
</tr>
<tr>
<td>Motors (each)</td>
<td>12</td>
<td>5</td>
<td>240</td>
</tr>
<tr>
<td><strong>Total Power Consumed (approx)</strong></td>
<td></td>
<td></td>
<td><strong>266 W</strong></td>
</tr>
</tbody>
</table>

Separate batteries are used to power the electronic subsystems and the motors on the AUV. Based on the above table, two 12V 7 Ah Sealed Lead acid batteries will be used to power the AUV, one for electronics and the other for the motors.

Figure 12 shows the schematic of the power distribution board used to distribute power to all the systems on the AUV. Further, the power distribution board allows implementation of an underwater switch on the hull of the AUV. The switch controls the power to all the systems on the AUV. When the switch is turned on, the circuit with the electronics battery is completed and the SBC, the Arduino, the webcam, the IMU and wireless routers are turned on. It further powers a 10V regulator which is used to drive the depth sensor. The 5V regulator, besides powering the SBC, provides a switching signal to the relays which complete the circuit with the motor battery. As only 10A relays were available at the time the circuit was designed while the motors can draw up to 20A current, two relays were used.

Fuses are extensively used on the power distribution board. A 25A fuse prevents a short circuit of the motor battery, while a 5A fuse prevents a short circuit of the electronics battery. Further, 7A fuses are connected in line to each motor for isolation in case any motor burns out.
5.2.2.2. Electronic systems – Rationale

Table 3 provides the choice rationale behind some of key electronics systems used on the AUV.

Table 3: Electronic components rationale.

<table>
<thead>
<tr>
<th>Subsystem</th>
<th>Rationale</th>
</tr>
</thead>
<tbody>
<tr>
<td>Single Board Computer</td>
<td>Small form factor, sufficient processing power and prior experience of working with it</td>
</tr>
<tr>
<td>Sparkfun 9 DOF IMU</td>
<td>Low cost, open source AHRS software availability</td>
</tr>
<tr>
<td>Arduino</td>
<td>Ease of use and availability</td>
</tr>
<tr>
<td>HD Webcam</td>
<td>Ease of use over USB and HD picture quality</td>
</tr>
<tr>
<td>Bullet 2HP wireless router</td>
<td>Compact size with very high output power (1W)</td>
</tr>
<tr>
<td>Honeywell Pressure sensor</td>
<td>Sufficient accuracy (~ 3cm) of depth measurement</td>
</tr>
<tr>
<td>Johnson’s 1000 GPH bilge pumps</td>
<td>Static tests in a bucket showed that 750 GPH motors provided sufficient thrust. The thrust produced by them was 1 lb. This would have been a too tight window for depth control and thus 1000 GPH motors were chosen.</td>
</tr>
</tbody>
</table>

5.2.3. Software System

The software system can be broken down into two categories; the AUV software and the Arduino software.

5.2.3.1. AUV Software

The Beagleboard xM Single Board Computer (SBC) was selected to serve as the primary on-board processor, due to its low price and high CPU speed (1 GHz). It runs a multi-threaded application written in C++ on a Linux-based operating system, to perform the
functions shown in Figure 13. OpenCV [11] was used for image acquisition and processing tasks.

There are 5 threads executing in parallel on the Beagleboard, with the following tasks:

- Image Acquisition from the camera – IMAQ thread.
- Detect if submarine is in field of view – DETECT thread.
- Get data from the various sensors – DAQ thread.
- Send motion commands to the Arduino – CONTROL thread.
- Communicate with the GCS – COMMS thread.

Figure 13: The software architecture of the AUV software, with 5 threads executing continuously.

5.2.3.2. Arduino Software

The Arduino software is responsible for measuring the battery level, temperature, depth and leakage status of the AUV. It also receives motion commands from the AUV software and accordingly controls the motor Pulse-width Modulation (PWM). We made a custom communications Arduino library called CommClient to interface with the Arduino software which has proven to be very robust in practice.
5.2.4. AUV Localization

Our localization subsystem consisted of an overhead camera at the GCS, with the camera fixed, pointing towards the pool surface. We also had a balloon floating on the surface which was attached to the AUV with a wire. Using the assumption that the balloon is exactly over the AUV, and perspective transformation algorithms, we can estimate the 2D position of the AUV/balloon in the pool’s frame by getting the balloon’s pixel position in the image, as shown in Figure 14. In practice, there were quite a few errors since the camera calibration needs to be precise; our average error was around 2 ft. Unfortunately, we were unable to integrate AUV localization with the rest of the system due to lack of time.

Figure 14: AUV localization.

5.2.5. Performance

Table 4 shows the performance parameters achieved by the AUV.

Table 4: AUV performance.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Max forward motion speed</td>
<td>0.5 m/s</td>
</tr>
<tr>
<td>Max backward motion speed</td>
<td>0.35 m/s</td>
</tr>
<tr>
<td>Max diving speed</td>
<td>0.2 m/s</td>
</tr>
<tr>
<td>Depth control accuracy</td>
<td>± 2.5 cm</td>
</tr>
<tr>
<td>Weight (with ballast)</td>
<td>12.82 Kg (18.58 Kg)</td>
</tr>
<tr>
<td>Battery life</td>
<td>70 minutes</td>
</tr>
<tr>
<td>Detection range</td>
<td>7 m</td>
</tr>
</tbody>
</table>
5.3. Stationary Detection System

5.3.1. Mechanical Design

For the detection system we decided to use a stationary underwater camera. To house the camera we made an underwater case, in a similar way as we did the AUV’s hull. We used a 17 cm diameter PVC pipe, this time only 25 cm long. Similarly to the AUV we attached a transparent acrylic dome to the front part and made a polyethylene end cap for the back part.

We could have done a different kind of case, since this was just for the camera, but as we already had the experience, and the materials, needed to build this kind of case, we chose it over other alternatives.

To mount the camera inside this case, we made an acrylic board where the camera was attached. A USB cable was used to connect the camera to the ground control station. Using a cable grip we made a watertight grommet.

To keep the camera stationary underwater, we made a steel frame that can be mounted on the edge of the pool, and additional ballast weight was added to it to avoid movement due to the waves generated in the pool.

Figure 15 shows the detection system mounted in the pool.

![Figure 15: Stationary Underwater Detection System.](image)

5.3.2. Submarine Detection

The detection algorithm for the UDS is almost identical to the algorithm running on the AUV. The key difference is that in this case, we do not use HSV colour segmentation to detect the submarine. Instead, a much more robust method called codebook background subtraction was used. This led to really good results and we were able to meet our performance requirement of successful detection in 90% of the cases.

For the input image shown in Figure 16, the algorithm was successfully able to extract the foreground object (shown in Figure 17), in this case the human submarine. The next step is to get the contour of the object (shown in Figure 18), and based on a few additional
characteristics such as length and area of the contour the algorithm decides if a submarine is in fact in front of the camera or not.

Figure 16: Colour detection.

Figure 17: Background subtraction.
5.3.3. Range Finding

Finding the distance of an object from a camera using only a single camera is a challenge. This problem was handled in [8]. The algorithm described needs 4 non-coplanar equidistant points on the object and uses those points to calculate the distance. The OpenCV vision software includes an implementation of this algorithm and that was used to calculate the distance of the target submarine from the camera. Figure 19 shows the tripod used as the reference object being held underwater.

Figure 19: Range finding algorithm running in UDS.
5.3.4. Performance

Table 5 shows the performance parameters of the detection subsystem compared with the initial system and subsystem requirements.

Table 5: UDS performance.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Weight (with ballast)</td>
<td>2.2 Kg (7.4 Kg)</td>
</tr>
<tr>
<td>Detection range</td>
<td>7 m</td>
</tr>
<tr>
<td>Detection time</td>
<td>Less than 1 second</td>
</tr>
</tbody>
</table>

5.4. Ground Control Station

The GCS was located at the edge of the pool, and consisted of a laptop with a human operator along with the communications equipment, as shown in Figure 20.

![Figure 20: The GCS setup – laptop, joystick, Wi-Fi router and antenna.](image)

The GCS software was written using the open source graphics library QT and C++. It consists of 2 subsystems, called the AUV GUI and the Detection GUI, which execute on the GCS laptop. The AUV GUI is the interface for communicating with the AUV; it displays the information sent from the AUV to the operator and also relays commands from the GCS to the AUV, as shown in Figure 21. The Detection GUI receives images from the Detection system and processes them to detect the presence of a submarine, as shown in Figure 20. Finally, the Robot Operating System (ROS) publish-subscribe framework was used to communicate between the AUV GUI and the Detection GUI.
5.5. Simulation of robot coordination behaviour

As part of the initial project requirements, the robots built should co-ordinate their actions to track an enemy submarine. To simulate this coordinated tracking behaviour, Q-learning techniques were used to make the robots learn and efficient coordinated behaviour mechanism.

As part of the initial project requirements, the robots built should co-ordinate their actions to track an enemy submarine. To simulate this coordinated tracking behaviour, Q-learning
techniques were used to make the robots learn and efficient coordinated behaviour mechanism.

Q-Learning is a popular reinforcement learning technique that is used to make agents learn behaviour in a simulated environment.

The behaviour of the agent can be divided into three components:

1. Learning algorithm, by which the agent updates its knowledge on the basis of experience in the world.
2. Action selection algorithm, which dictates how the agent selects its actions on the basis of its knowledge.
3. Exploration strategy, which alters the action selected by the mechanism above with the purpose of gathering new information from the world

Figures 23, 24 and 25 show the simulation of the collaboration algorithm done in Matlab, the first figure shows two AUVs (green and blue) searching for the submarine (yellow). The second one, shows a graph with the results of a run of twenty simulations (the x axis corresponds to the amount if simulations and the y axis to the amount of steps taken by the AUV) where it can be seen that the AUVs have learnt the path followed by the submarine and after the 11\textsuperscript{th} run the AUVs detect the submarine almost immediately.

![Figure 23: Screenshot from one simulation.](image_url)
Figure 24: Results from twenty consecutive simulations.

Figure 25: Path followed by the AUV that detects the submarine.
VI. Components

Table 6 shows the list of elements that were developed by team STINGRAIS and those that were acquired.

Table 6: Components developed and acquired for the system.

<table>
<thead>
<tr>
<th>Developed</th>
<th>Acquired</th>
</tr>
</thead>
<tbody>
<tr>
<td>AUV aluminium frame</td>
<td>Hull (PVC pipe)</td>
</tr>
<tr>
<td>Propeller shrouds</td>
<td>Transparent Domes</td>
</tr>
<tr>
<td>Internal electronic components board</td>
<td>Motors (bilge pumps)</td>
</tr>
<tr>
<td>Batteries tray</td>
<td>Propellers</td>
</tr>
<tr>
<td>Motor shafts and couplings</td>
<td>Cable grips for waterproofing</td>
</tr>
<tr>
<td>AUV and detection system back cap</td>
<td>Arduino 256 Microcontroller Board</td>
</tr>
<tr>
<td>Detection system frame</td>
<td>H Bridges</td>
</tr>
<tr>
<td>Leakage sensor</td>
<td>Single Board Computer</td>
</tr>
<tr>
<td>Power reset system</td>
<td>Power Over Ethernet</td>
</tr>
<tr>
<td>Power distribution PCB</td>
<td>Wireless router</td>
</tr>
<tr>
<td>Arduino Shield</td>
<td>IMU/AHRS</td>
</tr>
<tr>
<td>Battery voltage monitoring system</td>
<td>DC-DC Converters</td>
</tr>
<tr>
<td>AUV localization algorithm</td>
<td>Temperature sensor</td>
</tr>
<tr>
<td>Submarine detection algorithm</td>
<td>Depth sensor</td>
</tr>
<tr>
<td>Software architecture for AUVs</td>
<td>Batteries</td>
</tr>
<tr>
<td>Motor Control Module</td>
<td>Cameras</td>
</tr>
<tr>
<td>Submarine localization algorithm</td>
<td>Solid state relays</td>
</tr>
<tr>
<td>Motion planning algorithm</td>
<td></td>
</tr>
<tr>
<td>Multi-Agent collaboration simulation</td>
<td></td>
</tr>
<tr>
<td>Graphical User Interfaces</td>
<td></td>
</tr>
<tr>
<td>Internal state measurement system</td>
<td></td>
</tr>
<tr>
<td>Communications module</td>
<td></td>
</tr>
<tr>
<td>Pressure sensor liner</td>
<td></td>
</tr>
</tbody>
</table>
VII. Results and conclusions

7.1. Challenges faced

One of the major challenges we faced, which took a long time to get solved was waterproofing. We tested a lot of different ways to waterproof the AUV, but testing it was very difficult and time consuming as we had to go to the swimming pool for each test. After various tests, in which we weren’t able to completely seal the hull of the AUV, we came up with the idea of pumping pressurized air inside the AUV and then apply soapy water to the hull’s surface. In this way, bubbles appeared wherever there was a leak. This method didn’t require to get the AUV to the pool and didn’t need an enormous amount of water (we used less than a cup) so these factors allowed us to hold this test safely in the laboratory. If we had done this test at the beginning we would have saved a lot of time, especially pool time, which was very limited for the project.

Another major challenge was trying to do image processing under the varying light conditions both underwater and at the surface. Our submarine tracking algorithm would frequently fail due to false positives, and this problem was particularly acute when the AUV was at the pool’s surface.

The testing time at the pool was limited and thus it was essential for the system to always work on reaching the pool. This was a serious system integration challenge and it eventually led to the development of a very reliable and robust AUV solution.

7.2. System performance evaluation

Table 7 shows the comparison between the project requirements and the performance achieved by the system.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Requirement</th>
<th>System performance</th>
</tr>
</thead>
<tbody>
<tr>
<td>Area of operation</td>
<td>300 m² (15 m x 20 m)</td>
<td>300 m² (15 m x 20 m)</td>
</tr>
<tr>
<td>Patrol Border length</td>
<td>15 m</td>
<td>15 m</td>
</tr>
<tr>
<td>Operating Depth</td>
<td>1 m to 3 m</td>
<td>1 m to 3 m</td>
</tr>
<tr>
<td>Maximum Depth</td>
<td>4 m</td>
<td>4 m</td>
</tr>
<tr>
<td>Operating Temperature</td>
<td>26.7°C to 30.5°C</td>
<td>26.7°C to 30.5°C</td>
</tr>
<tr>
<td>Number of submarines to be detected</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Speed of the submarine</td>
<td>0.6 m/s</td>
<td>1.25 m/s</td>
</tr>
<tr>
<td>Maximum possible single mission duration</td>
<td>2 min</td>
<td>35 seconds average duration per mission</td>
</tr>
<tr>
<td>Turn Cycle Time (assuming no failures)</td>
<td>Under 30 minutes</td>
<td>Under 30 minutes</td>
</tr>
<tr>
<td>GCS equipment weight</td>
<td>Less than 100 lbs.</td>
<td>10 lbs.</td>
</tr>
<tr>
<td>-----------------------</td>
<td>--------------------</td>
<td>---------</td>
</tr>
<tr>
<td>Detection success rate (in an event of submarine entering system’s detection range)</td>
<td>90%</td>
<td>90%</td>
</tr>
<tr>
<td>Individual subsystem’s detection range</td>
<td>5 m.</td>
<td>7 m.</td>
</tr>
<tr>
<td>Position of the submarine to be reported</td>
<td>Cartesian with respect to pool’s top corner</td>
<td>Cartesian with respect to pool’s top corner</td>
</tr>
<tr>
<td>Accuracy of AUV and submarine position</td>
<td>Inside a 2 m radius circle of the measured position</td>
<td>Error ±0.3 m.</td>
</tr>
<tr>
<td>Maximum time available for detection</td>
<td>7 seconds</td>
<td>Less than 1 second for system to detect the submarine.</td>
</tr>
<tr>
<td>Minimum continuous tracking time</td>
<td>30 seconds</td>
<td>30 seconds</td>
</tr>
<tr>
<td>Maximum time allowed to lose contact</td>
<td>3 seconds</td>
<td>2 seconds</td>
</tr>
</tbody>
</table>

Besides the accomplishment of these parameters, we also met the following specifications for the subsystems:

- Report the detection of the submarine to the GCS: The stationary detection system was linked with the GCS, and once a submarine was detected, this information was passed to the GUI, and the operator could see a white box surrounding the submarine, indicating the detection.
- Send emergency commands to the AUV: The GUI has the option for teleoperating the AUV, which overrides the autonomous mode. So whenever the user has the need to control the AUV, he can do so, with the controls built in the GUI or by using an external joystick.
- Display internal status of the AUV: The GUI shows continuously the internal temperature and battery level of the AUV. And also has an indicator that shows if water has gotten inside it due to a leak.
- System needs to be portable by men without the need of any special lifting machines: The whole system is portable by average men; the only needed equipment for the project was a cart to transport the system in a safe way from the laboratory to the swimming pool. Figure 26, shows the system’s components inside the cart used for transport.
- Display the position of the AUV: We were able to develop an accurate enough localization system that could determine the position of the AUV in the pool. Unfortunately we weren’t able to integrate this with the whole system due to lack of time. And this problem also affected us in not being able to display the position of the submarine.
7.2.1. Strengths

Motion control: We achieved a very precise motion control on the AUV. With the AUV being able to move in a straight line (forward or backwards) and, when diving or surfacing, it does it keeping its base parallel to the water surface. So whenever the AUV moved, it was able to maintain its orientation.

The system’s components are in a mature state that allows them to be set up by anyone. Any of the team members can get the system started and run a test with it. In this way, our system does not need the presence of all the team members to operate, just a minimum of two people for supervision, and of course one human submarine.

The system was made to be reliable enough to withstand the transport from the lab to the pool, and be able to work once we arrived there. We never had a failure, due to handling or transport.

7.2.2. Weaknesses

Our detection algorithm relies on the target’s colour for detection. Due to light refraction properties of water, external light conditions directly affect its performance. Since all pool tests and the demo were held during the morning, we did not face a big issue with this problem, because the sunlight during that time of the day wasn’t too variable.

As the localization system couldn’t be implemented, our system does not have a reliable way for the AUV to localize itself, or for the GCS to know where the AUV is, therefore, the position of the submarine cannot be determined.

7.3. Conclusions and lessons learnt

The physical properties of underwater vehicles are in direct relation with their dynamic behaviour; this means that the shape estimation for the AUV must be very accurate in order to be able to build a good dynamic model. A lot of references were studied by the team, but
these only worked as a guide to the final design, because each different body underwater has its own hydrodynamics.

For verifying the theoretical model, the building of a prototype was very useful for initial estimations and measurements.

Waterproofing is a very important aspect when working with underwater vehicles. No water should get inside the AUV. For this we did a lot of testing with different kinds of glue, cement or sealants (refer to testing and results from [10]). But another factor that has to be considered is the pressure that the structures face while working underwater, which also affects their watertight properties. A useful way to test for water tightness is the use of compressed air and soapy water, this way leaks can be found without needing to go to the pool, as described in point 7.1.

A reliable electrical system is extremely necessary for this project. As such the team had limited testing time available at the swimming pool (avg. 80 min per test); therefore failure of any key electrical component during field testing would severely limit the available testing time.

The main lesson learnt in the software design is to test the software sufficient times and use test cases that mirror the real situation as closely as possible. The collaboration code needs to be built taking into account the capabilities/limitations of the hardware and software units of the system.

The process of system integration was one key thing learnt by the team members. We realized that subsystems could not be developed independently from the other, because in the end this would make them hard to integrate instead, when making some component or writing some code, that was in direct relation with another component of the system, this relation has to be considered during all the development process, thus making the final integration process much smoother.

### 7.4. Future work

One of the elements that can be considered for future work is making a second AUV, so that the whole tracking area, or at least the major part of it, is covered by the system. Then with these two AUVs, the collaboration algorithm developed by our team can be implemented and tested in the real system.

The detection system can be improved by adding additional stationary cameras, in order to be able to cover the whole length of the patrol border line.

The localization system worked very well, but couldn’t be implemented due to lack of time. So as part of the future work of the project is the implementation of the localization system on the AUV.
References

Websites consulted

[10]  https://docs.google.com/?tab=mo&authuser=0#folders/0B23l6eWD4ArYNjViMWE3ZWQtYjNmNS00YTItMTQtYzQtNzYtYzUwMTI5YWQ5ZGJh

Appendix “A”

Identification of resources and requirements of the functional analysis

Based on the functional architecture, each of the second level functions (X.Y), have been evaluated, and their inputs, desired expected outputs, external constraints and required resources have been determined.

The following figures show the analysis made for each of these functions.

![Figure A.1: AUV State measurement requirements analysis.](image)

![Figure A.2: Submarine detection requirements analysis.](image)
Figure A.3: Submarine tracking requirements analysis.

Figure A.4: Motion control requirements analysis.
Figure A.5: AUV Communications requirements analysis.

Figure A.6: GUI communications requirements analysis.
Figure A.7: GUI Displaying requirements analysis.
Appendix “B”

Hull design

The design of the hull was decided using the information from [2], [4], [6] and [9]. This helped us in coming up with the design shown in the main body of the report.

Once we knew how the AUV would look like, we focused our attention on obtaining the main parameters needed to model its hydrodynamic behaviour. For this we used the information in [2], [3], [6] and [9] and realized that the two main forces that affect the AUV are buoyancy and drag.

Our goal was to make the AUV neutrally buoyant, which means making the weight force of the AUV equal to the buoyant force.

\[ B = W \]

This allows the AUV to remain at a specific depth without the use of actuators, saving battery time. To do this we needed to know the exact volume of water displaced by the AUV, in order to make the calculations necessary to get the required weight.

To measure the drag force, we use the formula:

\[ D = \frac{C_d \cdot \rho \cdot V^2 \cdot A}{2} \]

Where,

- \( \rho \): density of water.
- \( V \): speed of the AUV.
- \( A \): Reference area.
- \( C_d \): is the Drag Coefficient, which is defined by

\[ C_d = \frac{24}{R} + \frac{6}{1 + \sqrt{R}} + 0.4 \]

Where,

- \( R \): Reynolds number.

The Drag Coefficient for some specific shapes can also be obtained from tables like the ones found in [6] and [9], which were empirically obtained.

Another force that affects the motion of the AUV is lift, which is defined by:

\[ L = \frac{C_l \cdot \rho \cdot V^2 \cdot A}{2} \]

Where,

- \( C_l \): Lift Coefficient. This one was obtained from [9].
Using these parameters, we developed a Matlab program that can estimate these parameters for the AUV, whose GUI is shown in Figure 8 of the main text.

Having the AUV’s inner volume and knowing its weight, we ran the program and obtained a total drag of 2.4 Newtons, a buoyancy reserve of 15 Kg, a lift force of 0.04 Newtons and a required thrust of 0.38 Watts.

Figures B.1 to B.5 show the theoretical results obtained.

Figure B.1: Drag Coefficient at different L/d ratio.

Figure B.2: Drag Force at different speeds.
The first results obtained were very favourable for the Hull we had in mind, so we decided to maintain the torpedo shape and started designing with these parameters in mind. Figure B.5 shows the evolution of the AUV’s design, from version A to the final design in version D. The version A design was the initial idea of how we wanted the AUV to look like. With two propulsion motors at the back and two depth control motors, one at each side. Version B was actually our first design for the AUV, which was then upgraded to version C, which considered the transparent dome, and the internal distribution for the electronics.

The change from version C to D was due to various factors:

- The hull by itself was not very easy to handle or carry, and if we added the fins, it wouldn’t have made it easier for us; instead it might have probably gotten worse.
- The weight of the motors implied the use of strong fins, possibly metallic. This would have been an issue when trying to mount the fins to the hull.
- We wanted to minimize the holes in the hull (minimize possible leaks) so the fins would have been glued to the hull, which wasn’t the best option, because the probability of losing a fin during operation increased.
If using fins in a similar way as the ones on version C, we would have needed a structure that could hold the AUV when outside of the water.

A frame provided a strong support for the motors.

The frame served as a handling device for the AUV, making its handling easier.

The weight of the frame helped in lowering the CG, making the AUV more stable.

The frame provides a structure that can hold the AUV while not in water.

The frame could be used to add the ballast weights, in this way this weight takes the CG even lower, increasing the AUV’s stability.

For the motor positioning, we made various tests in the pool with different positions for both sets (propulsion and depth control). And we got the best answer with the position shown in version D. Where the depth control motors were placed in the same longitudinal plane of the CG, and the propulsion motors in the rear of the AUV in the same horizontal plane of the CG.

For buoyancy control we added a total of 5.626 Kg using a steel bar which was mounted in the centre line at the bottom of the frame, and some extra weight in the front part of the bar to take the CG towards the bow, because the AUV was trimmed by the stern. Figure B.6 shows the bar used for ballast.

With this weight added to the AUV, we achieved zero-trim. Then the AUV was able to move forward and backwards in a straight line motion.

Reaching neutral buoyancy was a very difficult task for us, because it has to be 100% accurate, which was hard to accomplish with the available materials. But we finally got very close to this point having the buoyancy of the AUV slightly positive, which helped the AUV to remain on the surface, and in case of an emergency in would help it to reach the surface.
The depth control motors had no problem managing this positive buoyancy and were able to take the AUV to the bottom of the pool or keep it at a constant depth without problems. Besides, the diving and surfacing motion was done in a very stable way, keeping the horizontal plane of the AUV parallel to the surface at all times, without the need of continuous compensation with the motors.

The internal electrical components were mounted on an acrylic board, which is sled in and out the AUV in order to give easy access to all the components, the same was done for the batteries, which were mounted on a steel tray, using Velcro. This gave an easy way to remove the batteries from the tray for charging. Figure B.7 shows the board with the components mounted on it; B.8 shows the rails for the electronic components and battery tray inside the hull.
Figure B.8: AUV interior.
Appendix “C”
Details of all Electrical Subsystems

C.1. High Level Processor – Beagle Board SBC

The SBC is the main brain of the AUV. Its main functions are to:

1. Detect the Submarine using computer vision.
2. Localize and navigate the AUV in the pool.
3. Track the submarine (in case of tracking AUV)
4. Communicate and Collaborate with the other AUV.
5. Communicate with the Ground station to allow the user to monitor the system.

Beagle Board XM is the SBC which is used for the above tasks is shown in Figure C.1.

![Beagle Board SBC](image)

Figure C.1: Beagle Board SBC.

C.2. Arduino Mega 256 Microcontroller Development Board

The Arduino Mega 256 is the secondary processor on-board the AUV. It is mainly responsible for:

1. Controlling the thrusters using PWM based on the commands given by the SBC.
2. Acquiring data from sensors such as:
   a. Depth sensor
   b. Temperature sensor
   c. Hull leakage sensor
   d. Monitoring the voltage of batteries on board the AUV
3. Passing this sensor data to the SBC
4. Implementing a depth control system
5. Executing fail-safes in case of
   a. Overheating inside the hull
   b. Leakage inside the hull
   c. Critically Low battery level
   d. AUV sinking below rated depth
In all the above cases, the Arduino would send a signal to the SBC and thereafter simply fire the depth control thrusters to their maximum potential and immediately surface the AUV. Figure C.2 shows the Arduino Board.

![Arduino Mega Development Board](image)

**Figure C.2: Arduino Mega Development Board.**

### C.3. **Cameras – Logitech Pro 9000 Webcam**

The camera is the primary sensor on the AUV responsible for submarine detection and tracking.

Figure C.3 shows the camera.

![Logitech Webcam](image)

**Figure C.3: Logitech Webcam.**

### C.4. **IMU/AHRS – Sparkfun’s 9 Degrees of Freedom (DOF) Razor IMU**

The Sparkfun 9 DOF razor IMU shown in Figure C.4 provides the data from a 3 axis accelerometer, 3 axis gyroscopes and a 3 axis magnetometer over a 3.3V Transistor-Transistor Logic (TTL) serial interface. The IMU further uses an ATMEGA 256, which is provided with an Arduino boot loader. This would allow the team to implement an Extended Kalman Filter (EKF) on the raw IMU data being read by the ATMEGA 256 on the IMU. The 3.3V Future Technology Devices International (FTDI) chip is used to convert 3.3V TTL serial data between the IMU and the SBC. Using this chip, the IMU can be accessed over a USB port from the SBC.
C.5. Pressure Sensor – 19C005PG1K by Honeywell

The 19C005PG1K pressure sensor by Honeywell shown in Figure C.5 is used to measure the depth of the AUV inside the pool. The pressure sensor senses the water pressure acting on it and outputs a voltage that varies linearly with the water pressure.

C.6. Wireless Router – Ubiquiti Bullet 2HP

The Ubiquiti Bullet 2HP wireless router will be used for the inter AUV communications as well as for communications between the AUVs and the Ground Station. The router has 30dB output power and is Powered over Ethernet (PoE) using a PoE injector.


The LM 35 temperature sensor is used to sense the internal temperature of the hull.

C.8. Battery Voltage Monitoring circuit

Two voltage dividers are used to sense the voltage levels of the electronics and motor battery. The ADC circuit on the Arduino then translates them to digital values and it’s reported to the user interface.

C.9. Hull Leakage Sensor
An IC 555 based water level indicator is modified to work as a Hull Leakage Sensor. The high and ground terminals of the sensor are mounted very closely and as soon as any water enters the hull, and covers both the terminals, the sensor is triggered and it gives a logic high which is read by the Arduino. This is later reported to the ground station and the AUV executes the safety manoeuvre.

**C.10. Custom Arduino Shield**

The battery voltage monitoring system, the hull leakage sensor and temperature sensor have been implemented on a custom Arduino shield. Besides these, the shield also has ports to connect additional systems like servos (if required for manoeuvring), motor drivers and pressure sensor to the Arduino.

**C.11. H-Bridges- VNH3SP30 Integrated Motor Driver Circuit based**

From a preliminary survey of commonly used motors on DIY ROVs, it was found that most ROVs use a modified bilge pump coupled with a 50 mm propeller. These motors can draw up to 6A at maximum load. Keeping some factor of safety, the H-Bridges that would drive the motors should be able to provide 10A current to each motor. The Pololu High Current motor driver board, shown in Figure C.6 is a compact solution for driving the motors. It uses the VNH3SP30 motor driver integrated circuit.

![Figure C.6: Motor Driver.](image-url)