RUAUV - ÆGIR

Arnar Ingi Halldórsson, Ágeir Jónasson, Bjarki Jóhannson, Guðjón Einar Magnússon, Ingvi Ólafsson, Jóhann Ingi Gunnarsson, Sævar Steinn Hilmarsson, Sveinn Elmar Magnússon, Úlfar Karl Arnórsson, Þór Tómasarson

T-420-HONX Hönnun X 2014

Instructor: Jón Guðnason

Stefán Freyr Stefánsson

Torfi Þórhallsson

June 26, 2014

1 Prefix

The Reykjavík University AUV team presents Ægir the sea giant, god and king of the sea creatures from Norse mythology. The mechanical part was mostly finished in 2013 in a mechanical design course and for the last seven months the AUV team has been developing the electronics, control, acoustics and software side. The RU-AUV has never been this high-tech mostly because more computer vision implementation and the acoustics. Other improvements are in software architecture and a new motion controller designed parallel to the mechanics of the AUV.

2 Background

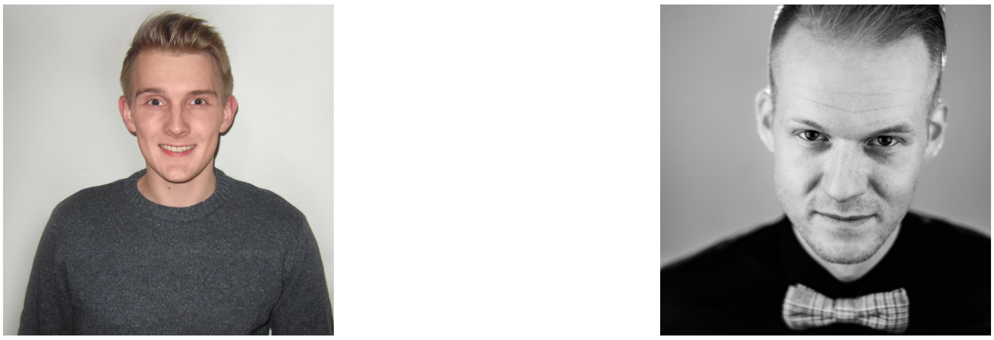

Figure 1: Arnar Ingi Halldórsson and Ásgeir Jónsson

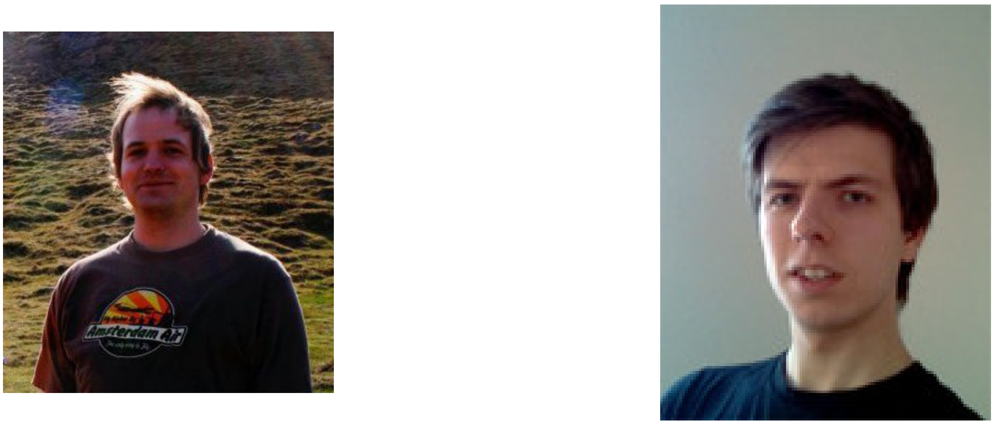

Figure 2: Bjarki Jóhannsson and Guðjón Einar Magnússon


Figure 3: Ingvi Steinn Ólafsson and Jóhann Ingi Gunnarsson

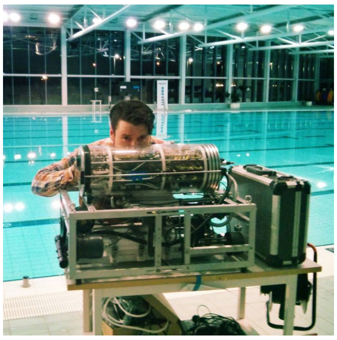


Figure 4: Sævar Steinn Hilmarsson and Sveinn Elmar Magnússon


Figure 5: Þór Tómasarson and Úlfar Karl Arnórsson

Name	Email address	Status
Arnar Ingi	arnarh12@ru.is	Computer Vision
Ásgeir	asgeirj09@ru.is	Mission Control
Bjarki	bjarkij10@ru.is	Acoustics
Guðjón Einar	gudjonm11@ru.is	Mission Control
Ingvi Steinn	ingviso@gmail.com	Motion Control
Jóhann Ingi	johanng12@ru.is	Mission Control
Sævar Steinn	saevar12@ru.is	Acoustics
Sveinn Elmar	sveinnel@gmail.com	Mission Control
Þór	thortom12@ru.is	Acoustics
Úlfar Karl	ulfar11@ru.is	Motion Control

3 Mission Control

Mission control ties together the capabilities of all the other subsystems in order to autonomously finish a mission. It keeps track of the mission goal and breaks it down to low level commands to control the AUV. A mission is composed of many tasks, mission control manages the lifetime of each task and makes decisions based on its success or failure.

3.1 Mission Configuration

Once the AUV is in the water it can operate fully autonomously but first a mission needs to be defined. A mission is a list of commands and goals for the AUV to follow.
To define the mission a XML file is used. XML was chosen because it is relatively easy to read, write and parse. A mission definition file defines any number of task lists and the order in witch they should be run. A task list is a list of tasks that will be run one after another. Each task list can contain any number of tasks. A task can be as simple command such as go to a certain depth or it can be a complex intent such as find and bump a buoy. In general tasks are kept as general as possible and parameters are used to define specific details. Every task has a defined timeout, that is the number of seconds the task is allowed to run before it is canceled. The trick is to give the task enough time to be able to finish under normal conditions but not so much that time is wasted waiting for a lost cause. Reactions can be defined for events such as timeout or errors, this can a recovery task list or a command to abort or restart the mission.

```
<quest>
  <tasklist id="start" next="foobar">
    <task name="Idle" timeout="10">
      <param name="time" value="5.0"/>
    </task>
  </tasklist>
  <tasklist id="foobar" next="@none">
    <task name="Movement" timeout="12" onTimeout="MovementRecovery">
      <param name="surge" value="1.0" />
      <param name="sway" value="0" />
    </task>
    <task name="Positioning" timeout="10" onTimeout="@restart">
      <param name="depth" value="200" />
      <param name="yaw" value="10" />
    </task>
  </tasklist>
  <tasklist id="MovementRecovery" next="@none">
    <task name=’’Positioning’’ timeout=’’10’’ onTimeout=’’@abort’’>
      <param name="depth" value="0.0" />
    <task/>
  </tasklist>
</quest>
```

Figure 6: Mission XML example

3.2 Quest Manager

The quest manager belongs to mission control and is responsible for managing the execution of task lists and tasks. When a quest starts the quest manager looks for a task list named start and executes the tasks one after another. Once a task list is done executing the quest manager checks if it has another task list defined as next, if so it runs that task list. This process is repeated recursively until there is no next task list. When running a task list, the tasks are executed in the same order as they are defined. Tasks use an interface provided by the quest manager to read sensor data and send commands to move the AUV. As long as no task calls for abort the list will run to the end. Tasks are guaranteed to take no more time than the defined timeout even if the task crashes or gets stuck in a loop. This is done by running each task on a separate thread and forcing the thread to join at timeout.

3.3 Sensor Manager

The sensor manager is where calibration and filtering is applied to sensor data and two or more sensors can be fused together for a more accurate reading. The main sensors used to steer the AUV can be put into three groups, yaw sensors, depth sensors and point sensors. Point sensors track the AUV’s offset from a target point using computer vision.. The sensor manager makes it easy to swap between different methods for each group.

The sensor manager sits between sensor output and the PID controllers in the motion controller. The sensor manager allows the running task to configure what sensor are used to control the AUV. This is useful to make the motion controller react to the environment faster than the task sends commands. An example of this is when the AUV needs to keep a fixed heading towards a buoy or any vision target. Instead of the task monitoring the location of the buoy and constantly updating the target heading it can tell the sensor manager to use the camera as the main heading sensor. Now the motion controller reacts immediately to any error and the task is free to think about the goal at a higher level.

4 Motion Control

A robost motion controller is essential for underwater vehicles. To describe all translations and orientations of a rigid body in a 3-dimensional euclidian space 6-degrees of freedom is needed, due to the mechanics of Ægir it is limited to four degrees of freedom which simplifies our control system.

4.1 Modeling

A dynamic model was made which consists of differential equiation that hold information about the postition, velocity and acceleration of Ægir with respect to input.

4.1.1 Controller

A PID controller is used for the control systems of the AUV which calculates the error or difference between the target and current value to minimize its difference by controlling the input to the actuators. A separate PID control loop with a ramp input is used for heave and yaw allowing for individual tuning. Numerous tests where done on Ægir to get optimal PID value allowing for robust control. Each parameter of the PID control can also be tuned from mission control GUI. The AUV has no sensors to obtain translation so surge and sway is controlled by an open loop system, except for tasks using computer vision. Than it is possible to use them as feedback. The computer vision delivers the error from the center of the camera to the object and that error is fed to the controller.

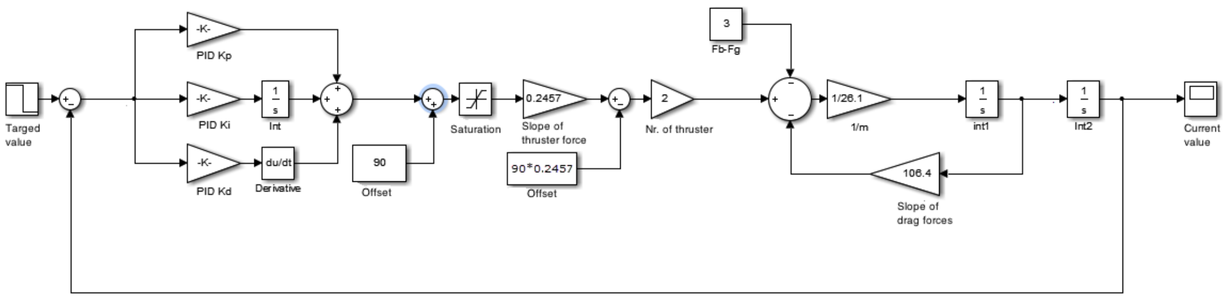

Figure 7: Simulink simulation for heave

4.1.2 Actuators

Six SeaBotix BD150 thrusters are onboard Ægir with two for each possible degree of freedom. Each pair is connected to a MD22 - Dual 24Volt 5Amp H Bridge motor-controller that is supplied with 18 V voltage from the batteries through DCDC converters. Each is controlled with a input signal from 0-180° with 0°

at full reverse, 180° full forward and 90° giving no power. Although each thruster becomes saturated below 45° and above 135° giving peak power of 10 N.

4.1.3 Hydrodynamic Effects

Drag forces have great affect on all objects moving in liquid. In order to estimate the forces acting on Ægir a CFD computation was done using simple CAD model. The data was plotted and linearized, the slope of the force is used in the dynamic model of Ægir.

4.2 Feedback and sensors

For a feedback control system the measured feedback needs to be consistent and reliable but the control will never be better than the weakest link and the sensors are essential part of the control system. Our system consists mainly of two sensors for active feedback of geometric position one for depth and another for heading. Other virtual sensors are fused together from various signals, for exempel a bump sensor that compares acceleration data for some period of time and checks if it goes over a certain limit.

4.2.1 Depth

The depth sensor used for feedback in the depth controller has a pressure range between 0 and 2.5 bar and the output ranges from 0 to 5V. This sensor is connected to the Arduino Mega 2560 micro controller which is used as an I/O for the computer. A analog signal of 10 bit comes from the sensor and is calibrated as pressure and then to depth.

4.2.2 Heading

For heading feedback a inertial measurement unit or IMU from SparkFun (9 degrees of freedom - Razor IMU SEN-10736) is used. An IMU is actually three sensors, a gyroscope, accelerometer and a magnetometer. All three sensors are triple axis and are fused together into Euler angles or orientations about every axis. The magnetometer measures changes to magnetic field strength that allows us to calculate angles relative to the magnetic fields and by using the accelerometer to reference the way down by measuring in what direction the gravity vector has we can use the magnetometer in all planes. The gyroscope which measures angular velocity can be integrated over time to estimate relative rotation around an axis. All this data is then fused together to give the best and dynamic response of rotation. Due to magnetic distortion the raw data from the gyro scope is integrated to be able to surge straight. The computer vision is also used for heading feedback.

5 Acoustics

The goal of the RUAUV acoustic team is to develop a passive sonar, able to locate a sound source(pinger) in three dimensional space. To accomplish this task, a hydrophone array, an A/D converter and angular calculation scripts are required.
The signal chain, shown in Figure 8, consists of hydrophones that are amplified and run through the A/D converters on a Discovery STM32F4 development board. The Discovery constantly runs one of the

channels through real time digital filters to detect when the signal from the pinger has arrived. Upon ping detection, the 3 signals are sent to the main computer via USB, unfiltered.

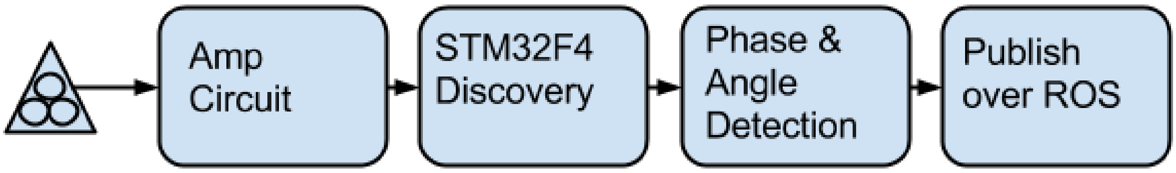

Figure 8: The sonar's signal chain.

5.1 Hydrophone Array

The hydrophone array consists of three Teledyne Reson TC4013 hydrophones put together in a triangle shape. These hydrophones were chosen due to there small size, light weight and good sensitivity.

5.2 Amplifying Circuits

There are three amplifying circuits that consist of a pre amplifier with high input impedance and little gain, two high pass filters and and output amplifier with high gain and very low output resistance. The input impedance and gain of the circuits are tailored for the TC4013 hydrophones.

5.3 STM32F4 Discovery

The STM32F4 Discovery board handles the data sampling and detection of the pinger signal. The board is equipped with an ARM Cortex M4 processor, along with three 12 bit AD converters. The board has a 160 MHz CPU with floating point unit(FPU), 192 KB RAM and requires 5 V supply voltage.
After detecting a ping the Discovery board sends data collected from the hydrophone array over to the onboard computer via USB for phase detection and angle calculations.

5.4 Publish over ROS

The AUV's main computer finds phase difference in signals from the three hydrophones. These phase differences are translated into time differences and from there into two angles giving relative position of the sound source in both a horizontal and plane and elevation. These angles are then published over the ROS system for higher level algorithms to work with.

6 Computer Vision

Computer vision plays a vital part in the Robosub competition in San Diego. There are several important tasks that rely on image processing, such as being able to find a path on the floor of a pool and the direction it is heading. These tasks require forward and downward vision. The sub has two Point Grey Chameleon cameras where one faces forward and the other downwards. The images from the cameras are captured and published on a ros topic using the FlyCapture drivers from Point Grey and camera1394 ros package. The resolution of the cameras is 1280 by 960 pixels but the images are resized to 640 by

480 pixels to increase processing speeds.

Previous versions of the sub have used image processing to control the sub directly. The current version is more segmented and uses image processing only as a sensor that publishes raw data. This allows for a more flexible and simplified design.

All image processing is written in Python using the OpenCV Library. Previous versions of the sub have used Matlab with an image processing toolbox. Python was chosen due to its simplicity and its rapid prototyping ability.

Each task has a seperate image processing module. These modules use a combination of color thresholding, edge detection, contour finder and optical flow to find the object of interest of the respective task.