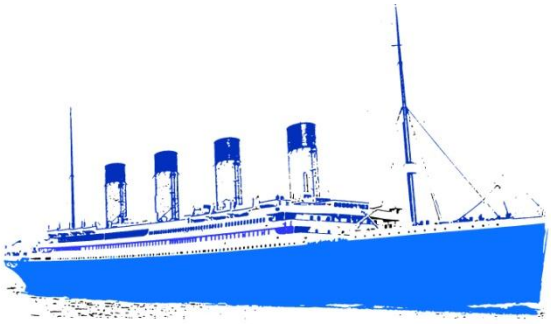
ENSC 440 Post Mortem
Dynamic Positioning Control System

Group 5

Bardia Bogharti – bboghrat@sfu.ca
Bengt Haunerland – bkh4@sfu.ca
Lucie Hiltner – lhiltner@sfu.ca
Yalda Majdi – ymajdi@sfu.ca
Carl Wahlstrom – cwahlstr@sfu.ca



TITANIC

POSITIONING

Table of Contents

I. List of Figures

II. List of Tables

1. Introduction

Marine vessels are sometimes required to maintain a heading or a position in situations where anchoring is difficult or impractical. An automated method of maintaining a heading or position is called Dynamic Positioning. Dynamic Positioning is prevalent in the oil industry, where large transport ships dock at offshore oil platforms. The vessels are required to steadily maintain precise coordinates for hours while cargo is transferred. The position of the vessels must be held constant despite exterior forces, such as wind, waves, and tidal pull. This must be reliably done to ensure the safety of the workers on the rig and the ship, and that no product is spilled in the process. [1]

A Dynamic Positioning System (DPS) is a control network that uses the ship's thrusters to adjust and maintain the ship's position according to feedback input from multiple sensors. The ship's location is adjusted and maintained by the DPS controller, which makes a calculation based on the current location of the ship using a GPS, and sensor information gathered from wind and water current data.

The purpose of the project is to implement a DPS on a 10-meter-long boat. The boat will maintain position and heading against external forces, such as the current in the Fraser River. The system will be expandable, and with the addition of further software, will be able to implement full dynamic positioning. The proposed requirements for the DPS are defined in this Design Specification.

2. Main functions

2.1. Sensors and Inputs

This section lists the sensors that are used in the project. The following sensors are required to implement a DPS-0: GPS, MRU, joystick, and emergency-stop-button. The GPS, MRU, and anemometer are used for collecting data. The joystick is used for controlling the desired location and heading. The emergency stop is used for safety to manually stop the DPS. Table 2.1, below shows the sensors and inputs that will be used in the proof-of-concept design.

Sensors and Inputs	Product Number	Specifications	Justification
GPS	MD 762-1009-07A SR 0404-14391-0001	Serial RS-232, Average GSP resolution is 1 m	Re-used, marine grade
MRU [5]	TSR-100	Anodized aluminium, 7-36 Vdc power, 1cm heave resolution, 5% heave accuracy, +/- 0.5 degree orientation accuracy, serial RS-232	Produced by Think Sensor research, has a compass and measures heavy, pitch, roll, and yaw.
Anemometer [6]	Garmin GWS 10	Wind speed and wind angle measurements. Wind speed/angle filtering included	has built in filter so that measurements can be take mean or average wind speed
Joystick	X03-57540	Microsoft Sidewinder. USB input, linux compatible	Re-used, has z-axis rotation
Emergency Stop button [7]	Game Switch - SPDT Square Red Lens, On-(On), Illuminated	5A/125VAC, Plastic pushbutton assembly,	Light gives feedback and is red colour is obvious for stop

Table 2.1: Products used for proof-of-concept

Figure 2.1 below shows the data streams in the final proof-of-concept system. All the inputs are sent to the embedded computer along serial lines (RS-232) or USB ports. The controller is contained on the embedded computer and all calculations will be performed on the embedded computer using C/C++ functions. The controller

interfaces with the monitor and the motor controller unit, which is used to control the direction and throttle of all the motors on the boat.

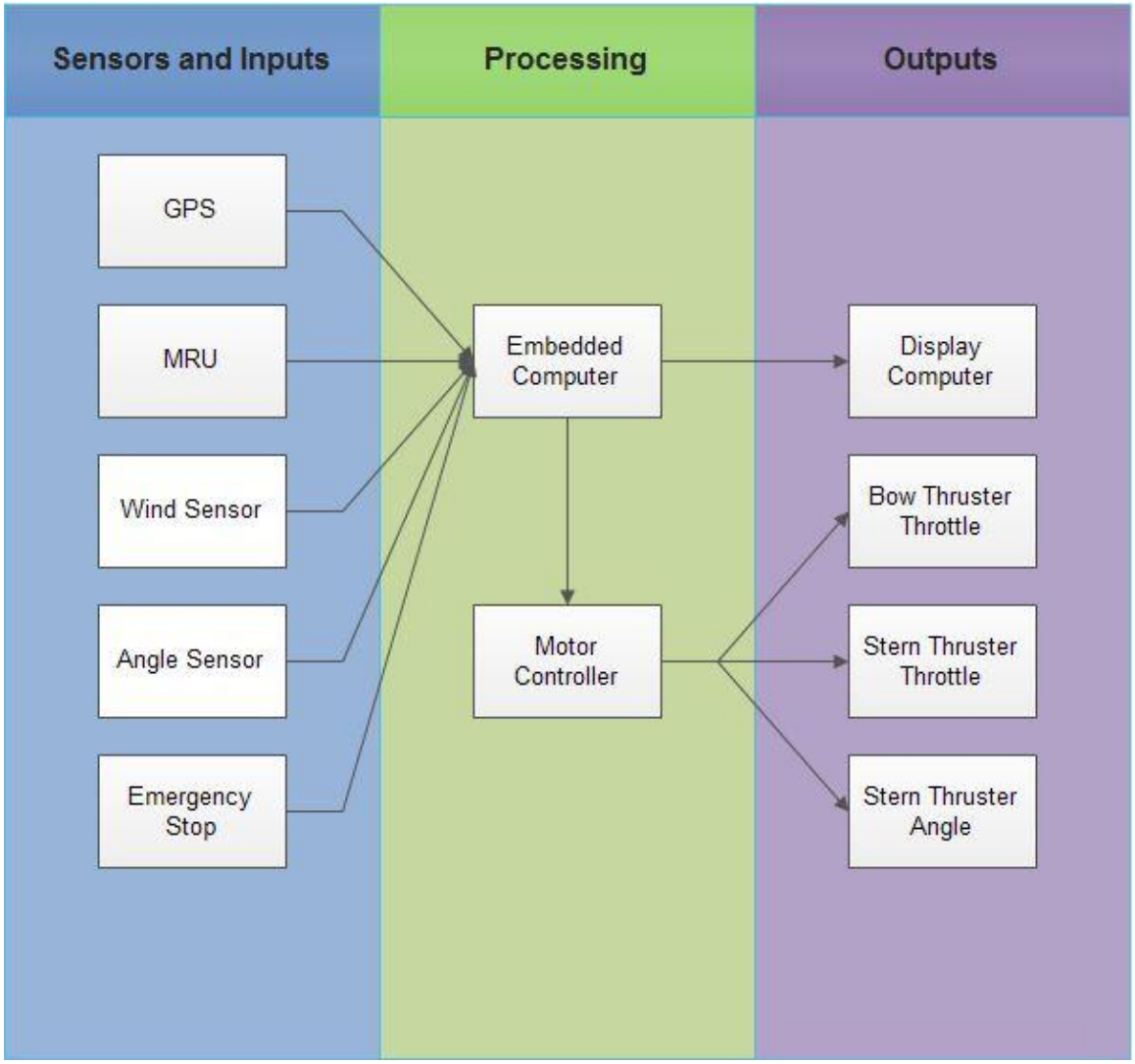


Figure 2.1: Flowchart showing data streams for DPS-0

2.2. Controller

The goal of the PID controller is to process the signals sent in from the sensors, and output a fast and stable response. The proportional (P) component reduces rise time of the response, so the DPS reaches its desired position faster. The integrator (I) component reduces steady state error caused by slowly varying ocean currents and wave drifts. The differentiator (D) component reduces overshoot so that the system thrusters do not overcompensate.

Figure 2.2 shows the control system that will be used on the proof-of-concept. The controller is made up of several different components including a PID component, a wind feedforward system, wave filtering, state estimators, and thrust allocation. The wind feedforward system takes in wind speed and direction from an anemometer and then compensates for the mean wind. The wave filtering filters out high frequency wave components from both wind and ocean waves, since the boat cannot respond fast enough to compensate for high frequency ocean waves and wind gusts. The state estimator is used to estimate unmeasured states such as linear and angular velocities. The state estimator is implemented using a Kalman filter. The last part of the control algorithm is the thrust allocation which will tell the two stern thrusters which way to turn and what speed to use. It will also tell the bow thruster when to turn on and whether to spin clockwise or counter clockwise. The output will move the thrusters and provide the new position and heading, which is measured by the GPS and MRU. The GPS and MRU provide feedback into the system and are compared to the desired input position from the joystick. The entire controller system will be contained on the embedded computer. [3]

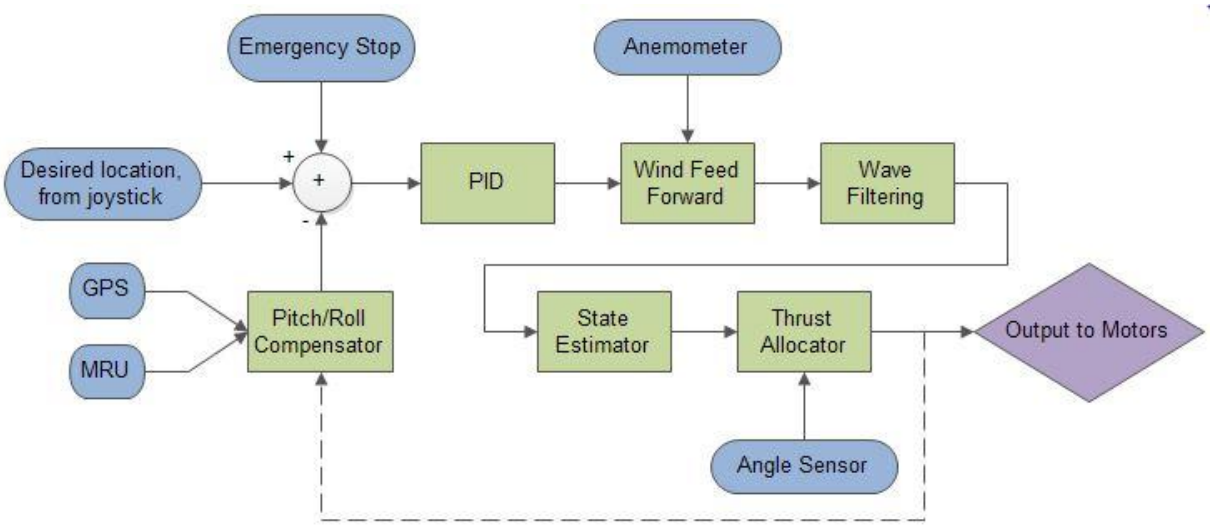


Figure 2.2: Flowchart of the DPS controller. The colour-coding matches Figure 2.1

The proof-of-concept design uses a development board with a separate USB to serial adapter. The USB to serial adapter will be enclosed in a metal container, with the wires temporarily secured. The model doesn't use a joystick or a state estimator as originally planned. The joystick was replaced with keyboard control since the production model will have an integrated display computer and joystick, therefore making it impractical to deal with separate driver installation for the joystick on the current model. The keyboard can model the joystick very well. The state estimator very difficult to model and is not a requirement for the proof-of-concept model. The

code for the state estimator could take years to finish and therefore will be added during summer work terms.

3. Cost Analysis

Component	**Cost Estimate ($)**	**Final Cost ($)**
TSR-100 MRU	loaned from TSR	
GPS	loaned from TSR	
USB to 4-Port Serial	60	60
Embedded Marine Computer	1500	1500
Clipper Wind Sensor	1129	230
Cables, Connectors etc.	500	50
Use of boat (Fuel and Captain's time)	1000	
Motor Controller for steering Hydraulic motor	220	220
Linear actuators to attach to throttles	520	260
Total	**4929**	**2320**

Table 3.1 below shows the financial breakdown of the project. Some of the components have been loaned from Think Sensor Research (TSR). TSR financed all

other components. The costs for cables, connectors, and use of the boat are approximate.

Component	**Cost Estimate ($)**	**Final Cost ($)**
TSR-100 MRU	loaned from TSR	
GPS	loaned from TSR	
USB to 4-Port Serial	60	60
Embedded Marine Computer	1500	1500
Clipper Wind Sensor	1129	230
Cables, Connectors etc.	500	50
Use of boat (Fuel and Captain's time)	1000	
Motor Controller for steering Hydraulic motor	220	220
Linear actuators to attach to throttles	520	260
Total	**4929**	**2320**

Table 3.1: Financial breakdown of project

The total cost for all the equipment as outlined in the table above is $2320 (excluding the cost of loaned components). Our initial cost estimate was higher because we were expecting to use an ultrasonic wind, which is more accurate and

more expensive the wind sensor that we ended up using. We also did not have to pay for fuel and other boat expenses since we didn't get on the boat.

4. Schedule comparison

Milestones	Planned Due Date	Milestone Achieved date
Prepare Parts list	01/30/2014	01/31/2014
Inspect boat	01/30/2014	01/24/2014
Order Parts	02/15/2014	03/06/2014
Controller Implementation	03/1/2014	04/1/2014
Input implementation	03/1/2014	04/7/2014
Integrate and Debug	04/1/2014	04/15/2014
Implement on boat	04/1/2014	TBA

Table4.1 shows, the milestones outlined in the proposal and current progress. Unfortunately we were unable to get onto the boat before the demo, however installing the system on the SFU test boat is still a plan for the summer.

Milestones	Planned Due Date	Milestone Achieved date
Prepare Parts list	01/30/2014	01/31/2014
Inspect boat	01/30/2014	01/24/2014
Order Parts	02/15/2014	03/06/2014
Controller Implementation	03/1/2014	04/1/2014
Input implementation	03/1/2014	04/7/2014
Integrate and Debug	04/1/2014	04/15/2014
Implement on boat	04/1/2014	TBA

Table4.1: Comparison of initial milestones and current progress

* The MRU and GPS were available by the planned date. The embedded computer and Anemometer were ordered later.

5. Challenges

The embedded computer took longer then expected to arrive. We were able to use a loner development board, this allowed us to could continue to work on the project instead of waiting for the computer to arrive.

The display provided some technical challenges, since we were initially planning to do the display on the embedded computer using the Linux operating system. However installing graphic interface programs on the Linux computer became very difficult and time consuming. Linux also doesn't support the C# graphic interface program, which we were more familiar with. The solution to this problem became to use a windows laptop for the display and collected data from the embedded computer through an Ethernet cable. We chose to do this since the production version would have a windows computer for the user interface and therefore it was

not practical to do the display in a Linux environment. This allowed us to use C# and to circumvent the difficult graphic interface installations.

6. Group Dynamic

The group was organized in a democratic fashion, where everyone opinion was listened to and decisions were made as a group. The administrator would identify how the tasks would be spilt up, but the group decided together who was best suited for each role.

There was one instant where 2 members were working on the same section of code and didn't communicate to one another that they were working on the same thing. This resulted in 1 person being upset that their hard work was not going to be used. However once the issue was identified the group tried to find away to make sure that miscommunication does not happen again. The team established a plan that if people start working on something that was not assigned to them during a group meeting, then they would email the group to see if anyone had started that part before starting to work on it.

7. Reflection

Overall the workload distribution was evenly distributed. Table 7.1 shows the work distribution for the tasks required for this project.

Responsibility	**Bardia**	**Bengt**	**Carl**	**Lucie**	**Yalda**
Input interfacing	**XX**				
Data Parsing	**X**				**XX**
PID				**XX**	
Wind Feedforward		**XX**			
Filtering		**X**	**XX**	**X**	**XX**
Thruster Allocation		**XX**			
Display			**XX**		
Output	**XX**				

Interfacing					
Documentation	**X**	**XX**	**XX**	**X**	**X**
Administrative Tasks		**XX**		**X**	

Table 7.1: Shows technical responsibilities. Where XX denotes primary role and X denotes secondary role.

7.1. Bengt

For this project I took on the project management responsibilities. The main thing I learned is that I do not need to know all the details of how things work before I assign certain tasks. For example when reading the Fossen textbook I spent time trying to fully understand each part of the controller algorithm rather than just looking at the big picture. When doing project management work in the future, I would read enough to assign specific tasks and let each person find the technical details for their specific task. This would speed up the process and allow people to start working on there specific tasks earlier. Then it would leave room for going back and learning the technical details later and being available to help team members understand the technical information. Learning the high level design first and not getting stuck on learning the low level details is important because it allows team members to contribute more easily.

From a technical perspective I learned how to program a control system in the C++ language. I picked up many important skills such as finding and linking libraries that perform certain functions and simplify the code, for example using the Eigen matrix library allowed for easier matrix math operations. Using established libraries makes certain operations more efficient and reliable, since it has been widely established. It is also important to make sure that all group members use the same library for the same operations because this makes integrating the code more efficient.

In this project I learned how to implement and design a control system using matrix math operations. More specifically I learned to implement the wind feedforward and thruster allocation part of the control system. For the thruster allocation part I had to learn how to linearize a non-linear matrix and how to invert the linearized matrix. I also developed the skills necessary to implement the mathematical algorithm in C++.

I also learned how to multithread input sensor data. This is important because the input sensors all send data in at different rates. The MRU is the fastest and the wind sensor is the slowest. If the code is single threaded a latency is built up on the MRU data, since it has to wait for the other sensors to finish updating. To solve this

problem I learned to create a thread for each individual sensor to allow multiple operations to be done simultaneously. Then when the thread read the data it locks off the processing function so that there are not conflicting processed data sets.

7.2. Bardia

In this project my roll was mostly the serial IO interfacing through Linux kernel functions. I learned a lot about asynchronous serial communication and RS232 handshaking protocols. I found this experience very valuable because if one technical skill that I can say I learned from 440 was interfacing hardware and software. I learnt a lot about how to parse serial inputs and how to deal with data synchronization.

Second skill I can take away from capstone is testing and debugging. I learnt about significance of these skills and how they play a major role in the design process. Also I had to learn quite a bit about compilers to learn various issues that came up in setting up the integrated development environment.

Having the chance to work on the actuators in our system, I encountered a whole side of engineering I have never dealt with before; that is the electromechanical devices and their interfacing. I learnt about PWM signaling and how different encoders provide digital feedback.

Aside from the technical skills obtained, I also learnt how important it is to stay communicated with your teammates to ensure that task will be integrated easily when design processes is done in parallel among the group mates.

All in all, 440 was a great experience for me during which I was able to gain valuable knowledge about myself, engineering, and have the chance to make friends and work with some very good people along the way.

7.3. Carl

One issue with Titanic Positioning, initially, was team communication. Sometimes people were difficult to get a hold of. Sometimes team meetings took place without the full team, and those absent were not always updated. By the end of the project, our communication had improved drastically, and everyone was working together.

I learned that the best attitude is a lighthearted one; to laugh at setbacks and unforeseen barriers rather than become anxious and stressed.

I came to properly appreciate that focusing on individual achievement is not a healthy way to approach a project. I learned that it is better to just continue working on an unfinished goal, rather than become angry at someone who has not contributed as much as originally desired.

I came to truly appreciate the need to fail faster. We had some issues waiting for hardware to arrive. We should have filled that time more with writing internal components of the controller that were not affected by the sensors or actuators. Had we done this, we would have found internal bugs with our algorithms and math for the controller sooner, and there would not have been as much of a crunch to complete the project once we were integrating the hardware.

I learned that a journal can be a useful device. It can help me keep track of the timeline of a project when it all started to blur together. It was also a useful source of scrap paper and spit-balling ideas. I realized that the journal does not have to be a pristine document, clinically documenting my progress, and sterile.

The first hurdle we faced were the various documents. During these I learned what kind of audience to write for and what style to write in (or sometimes the opposite) when constructing a technical document.

The first major technical problem we faced was setting up the IDE environments before coding. Finding appropriate C++ libraries, and then configuring the environment and code so we could utilize them, took a lot of time. It was something none of us had much experience with, and something that engineers are never taught in any class. I am now better at properly storing and linking to libraries.

In order to eliminate the noise of our output, we had to low-pass filter the data. We attempted to use filter libraries and pre-made filters to perform this function for us. We realized that all the filters were using the frequency domain, while we needed the time domain. We began to look into Fourier transforms. Eventually we realized that it would be faster to write our own simple, discrete-time filter, which I puzzled out and implemented with no external libraries.

When we had to create a GUI, we originally just used the console window and text files. I learned how to manipulate what was displayed, and how to use different types of data streams in C++. Later, we decided to implement a GUI on a Windows platform in C#. I was the only member with any C# experience, and so I took on the task. To create this GUI, I had to learn how to use serial port communication protocols, as well as creating a dynamic display, which I had never done before.

Overall, working with Titanic Positioning was an excellent experience. Especially at the end of the project, the team came together and we completed many goals. There were many moments of celebrations, and more than a few groans as we discovered new problems. We kept the mood light and remained optimistic though the process, and produced an interesting demo. I'm looking forward to working with these fine engineers in the future.

7.4. Lucie

Since we were programming the DPS for Think Sensor Research, I learned about the design process and aspects of the design that were required to bring a product to market. Designing a practical system that can be implemented in industry, and having the support to do so is the reason why this project was so appealing to me in the first place. One of the difficulties we faced was the time it took to order and receive the products. I learned that relying on others for critical parts of the project is necessary, and will often cause the project to progress at a slower rate than I would like. I learned that it's important to plan ahead for these "wait times", and in the future to plan out what can be worked on when parts (this could extend to parts of the code – not just hardware) are not available. Another thing I learned that I had not done before was designing with standards in mind, in this case from the American Bureau of Shipping. This project gave me a realistic look at what it will look like when I am working in industry and will help me with project planning and time management in the future.

The first thing I had to do was research navigation principles. This took a long time and lots of reading, but I learned about a whole area of engineering that I had not considered before. Designing the PID controller was challenging. I spent a long time learning about control systems, control theory, and designing and tuning PID controllers. Despite the long time I spent learning about control loops, the code for the PID controller was relatively simple and not very long. Next time I will try to use my time more efficiently. I will do this by looking at how people have managed to accomplish a task in the past, and try to do something similar to what has already been done, as it applies to my project. Then I can go back and learn theory that is not so broad and applies only to what I am working on.

Programming the state estimation part of the controller was the most challenging thing for me technically, both because the theory was hard to understand and because it was hard to code. The coding involved reading in the coordinates from the GPS and the MRU and using rotation matrices to make the coordinates appropriate for calculation. The data from the MRU had to be processed in order to

compensate for centripetal acceleration when the boat turns, and compensate for pitch, roll and heave of the boat. The GPS had to be processed in order to compensate for the offset from the GPS to the center of the boat (picture the GPS swaying in the air from side to side above the boat in the waves). After all the raw data from the GPS and MRU was processed, it had to be put into a matrix, linearized by differentiation, and fed into a Kalman filter for state estimation. I didn't complete this part of the code, partially due to problems with integration and transforming the filter into c code from matlab, and also because most of the data that is required for the estimation requires data (and tuning) from the boat, which we did not end up having access to in time. I plan on continuing working on the state estimator in the summer, and taking more time to understand the theory well.

7.5. Yalda

In this project I was mostly involved in data parsing and programming the controller. It took me a while to understand how to interpret data from the sensors, and how to communicate with them through serial ports. It was really exciting and challenging at the same time, because I never actually got a chance to do a similar work during school. Therefore I had to spend a good portion of my time doing research about the design procedure.

Even though I have not taken the embedded system course yet, working on this project gave me a good opportunity to understand a lot of concepts about embedded systems in advance. For instance, I was able to understand what an embedded system does, and how I can communicate with several components through an embedded system and interpret the data I obtain from it.

I also worked on making a digital filter for the control system. This also gave me a good opportunity to refer to what I had learnt from the related course. In that course however, we made a filter on a bred board, so I had never experienced coding a digital filter in C++. Therefore I had to do a lot of research on how the digital filters work, where to start. and how to interpret the output from a digital filter.

This project also gave me a good opportunity to develop my coding skills further. I gained experience on how to find useful libraries and functions online and link them to my code to simplify the coding procedure. It also gave me an opportunity to refresh my memory on C++ coding techniques and syntaxes.

In terms of personal skills, I learnt how to manage my time properly. How to deal with stressful situations where we were very close to deadlines and still had not

finished the required task. I learnt that the easiest way to avoid stressful situations was to plan well in advance. Also I learnt that it is always good to have a plan B in case your first plan does not work as expected.

I was also able to improve my teamwork skills. I received a lot of help from the other team members when I was stuck on a task, and I also tried to help others as much as I could. Working on this project made me believe that you do not really have to know every details of a project in order to improve it. There were times that I had to spend so much time doing research and working on a piece of code, in the end however, I noticed that I was not on the right track and that had to start it from the beginning. I learnt that even in these situations I could gain a lot of knowledge and experience that could be useful later.

8. Conclusion

Goals Met

The groundwork for a DPS-0 was firmly established. The sensors were properly implemented, and the system was able to receive data from them. The entire control algorithm, from sensor input to thruster allocation, was implemented. The sensor data was properly passed to the control algorithm, and used effectively.

Control code for some of the actuators was written.

A rudimentary graphic user interface (GUI) was created that displayed all the sensor data and some of the control data. The GUI was implemented on a Windows laptop provided by Titanic Positioning.

Goals to Complete

The project will be continued during the summer by a couple members of the team under the employment of Think Sensor Research. The option will be available for other members to join during the fall semester as either a coop or full time employment.

The actuators will be connected to the control algorithm, and will be controlled by the output data.

The Graphic User Interface will be improved to show data more clearly, and to allow inputs from the user. A joystick will be added for user input.

The project will be upgraded from a proof-of-concept to a production version. The control code will be optimized and streamlined. The display will be presented on a

marine display package. The current embedded computer will be replaced with a marine rated embedded computer, with built-in serial ports. The GPS will be upgraded to a marine grade GPS. A north-seeking gyroscope will be used to acquire more accurate heading data than the MRU.

9. Appendix A: Meeting Minutes

Titanic Positioning

MINUTES

January 10, 2014

10:30-11:00

McKenzie Cafe

Present: Bengt, Blake, Carl, Lucie, Yalda

Absent: none

Purpose of Meeting: To establish a plan for the project proposal and to develop any questions that we have for Pavel.

Minutes:

Bengt called the meeting to order at 10:30.

A. Project Proposal: Review Rubric

How should we split up the work for the proposal

Discussion: Establish what can be done before meeting with Pavel and what needs to be discussed with him on Tuesday (12:30).

Action:

- Proposal Introduction/background: Yalda
- Scope/risk/benefits: Bengt
- Market/competition/research rationale: Lucie
- Company details: Blake
- Establish that project planning and cost will be discussed with pavel

C. questions for Tuesdays meeting

Discussion:

- Project planning
  - Develop a plan for the project and exact details on how much of the project will be done in 440
  - Establish milestones
- Cost considerations
  - Develop list of equipment needed (to prepare for research on cost)
- Software
  - What OS will be using linux/windows
  - What language c/c++
  - Will pavel provide licenses for home use or will basic lab computers have the software or lab space available

Action: Issues will be discussed on Tuesday.

D. establish contact info and programs used for file sharing

Discussion:

- Provide reading material to Blake so that he can get caught up with the overall project
- Ensure everyone has access to:
  - Google docs
  - Skype/facetime/google hangouts
  - Dropbox

Action: Blake will email group the email he uses for dropbox and google docs.

D. Next Meeting Date

The next meeting was arranged for January 14, 2014 at 12:30-2:30 in the underwater research lab.

Meeting was adjourned early at 11:00.

Titanic Positioning

MINUTES

January 14, 2014

12:30-1:30

Underwater Research Lab

Present: Bengt, Carl, Lucie, Yalda, Pavel (Think Sensor Reseach)

Absent: Bardia

Purpose of Meeting: To discuss the purpose of the project and establish logistics on cost.

Minutes:

Pavel called the meeting at 12:30

A. Discuss scope of project

To get a manual control DP system running on a boat in Barnett Marina.

Discussion: Consider options for how to implement the system

Action:

- Option 1: implement using PLC (needs floating point operation)
- Option 2: implement using a embedded computer
- Enter options into proposal and decide later which is better

B. discuss equipment

Discussion: discuss list of parts needed

- Marine GPS
- Gyroscope
- Motion reference unit
- Joystick for controller (serial port)

Action: Once equipment parts have been found email Pavel the list and he will order the parts

C. establish next steps

Discussion: the following actions to be completed after proposal is done

- Get on the boat to see what we have to work with

- Possibly rewire the boat if needed
- Get list of equipment put together

D. Next Meeting Date

The next meeting was arranged for January 20, 2014 to fine tune proposal.

Meeting was adjourned early at 1:30.

Titanic Positioning

MINUTES

February 3rd, 2014

12:30-1:30

Underwater Research Lab

Present: Bengt, Carl, Lucie, Yalda, Pavel (Think Sensor Reseach), Bardia

Absent: none

Location: Underwater Research Lab

Purpose of Meeting: To get information on the Motion Reference Unit and discuss next steps

Minutes:

Pavel called the meeting at 12:30

A. Discuss MRU, GPS and other hardware

MRU is made by Think Sensor Research and is supplied by Pavel.

Discussion: Discuss hardware

Action:

- MRU was brought by Pavel and is stored in the Underwater Research Lab
- MRU user manual provide with specs of MRU
- Discussed joystick needs to be 3 axis and have USB connection
- Consider using CBC connectors (IP67 rated for water spray)
- Should use RS-232 serial connectors for the proof of concept
- Wind speed needs to be serial since USB does not have long enough cables

B. discuss equipment needed for interfacing with the boat

Discussion:

- Hydraulic steering auto control will be supplied by Pavel (since the hydraulic steering control is out of scope)
- Throttle control still needs to be addressed

Action: Once equipment parts have been found email Pavel the list and he will order the parts

C. establish next steps

Discussion: research operating systems that we want to use for the embedded computer

- Options: linux, embedded linux, windows, embedded windows, QNX
- Needs to be compatible with processor of embedded computer
- Needs to have device drivers readily available (not part of scope to write drivers)
- Look for embedded libraries for GPS etc.

D. Next Meeting Date

The next meeting was arranged for February 7, 2014 to discuss research for OS and settle on a embedded controller.

Meeting was adjourned at 1:30.

February 11, 2014, Minutes

Sorry folks, I did not write down quite everything in the journal.

Pavel provided us a computer with Ubuntu installed on it, and a USB-serial splitter. We can begin testing sensors and writing code with this machine. We need to provide a monitor, keyboard, and mouse. The computer is being stored in Bardia's locker (#12) in Lab 1. Ask Bardia for combo.

Pavel said he would order an embedded computer soon. We mentioned that our preferred model does not explicitly support Linux, but Pavel was sure that it would.

Pavel recommended that we put Ubuntu onto a flash drive so we can begin working on the code from any computer. 16-Gigabyte drive will apparently hold all the OS, IDE, and drivers needed.

Pavel likes the idea of using Linux for the project. Perhaps use Windows (C#) for the display computer. The display computer will not be required until working on larger ships with longer cables, so can be safely ignored for now.
I would more comfortable with C because I've never used C#. I'm not sure how similar they are. - lucie

An internal deadline for wiring the test boat has been set around March 14, 2014.

Pavel showed us the batteries (Marine Deep Cycle 12V) we can use for mobile testing at SFU. Located in URL.

Bardia, Bengt, and Carl worked on the Functional Spec document in Lab 1 a little bit. Internal deadline for FS is Friday, Feb 14. Pavel would like a copy of FS so he can send it to his contacts. Suggest including Pavel during weekend review/edit of FS.

Titanic Positioning

MINUTES

February 18th, 2014

4:00-4:30

Underwater Research Lab

Present: Bengt, Carl, Lucie,

Absent: Yalda, Pavel (Think Sensor Reseach), Bardia

Location: Underwater Research Lab

Purpose of Meeting: To discuss next steps of software developing

Minutes:

Bengt called the meeting at 4

A. discuss next steps for software development

Discussion:

Action:

1. Set up linux environment
2. Find and install drivers for MRU, GPS, wind sensor, angle sensor, and autopilot interface
3. Plug in sensors and confirm communications
4. Revisit and create a plan for group coding

B. discuss missing hardware

Discussion:

- Wind sensor
- Angle sensor either potentiometer or digital encoder
- Motor/throttle control

Action: Once equipment parts have been found email Pavel the list and he will order the parts

C. establish next steps

Discussion: research operating systems that we want to use for the embedded computer

- Options: linux, embedded linux, windows, embedded windows, QNX
- Needs to be compatible with processor of embedded computer
- Needs to have device drivers readily available (not part of scope to write drivers)
- Look for embedded libraries for GPS etc.

D. Next Meeting Date

Friday February 21st

Meeting was adjourned at 4:30.

Titanic Positioning

MINUTES

February 18th, 2014

4:00-4:30

Underwater Research Lab

Present: Bengt, Bardia, Lucie,

Absent: Yalda, Pavel (Think Sensor Reseach), Carl

Location: Underwater Research Lab

Purpose of Meeting: To discuss progress and integration of software development

Minutes:

Bengt called the meeting at 12:30

A. discuss progress of Software

Discussion:

- Wind Feedforward Function completed
- Parsing from MRU and GPS completed (still need to get data for yaw)
- PID Loop (still need to add matrix support and error function)

Action:

- Bengt: thruster allocation
- Bardia: i/o of motor controller
- Lucie: filtering and possibly state estimation, also add matrix capacity to PID
- Carl/Yalda: display

B. Next chance to work on boat and integrate code

Discussion:

- Start wiring boat on Thursday or Friday
- Meet on the weekend to integrate code

Action: confirm that we can get acesss to the boat

D. Next Meeting Date

Meeting was adjourned at 4:00.