Internship Report

François CÉBRON
CI2018 - ROB

Sensors and control of an autonomous sail-boat

Internship performed at the University of Plymouth:
University of Plymouth, Drake Circus
Plymouth, Devon, PL4 8AA - United Kingdom

(Internship : from 26/06/2017 to 15/09/2017
viva : ../10/2017)

Supervisor:
Doctor Jian WAN, Lecturer, University of Plymouth - AMS
Tutor:
Professor Luc JAULIN, Professor in robotics, ENSTA Bretagne - LabSTICC

Privacy policy

There is no specific advice concerning the content of this document, because the code associated is open-source and available on the net. You will find it at the following URL: https://framagit.org/fcebron/AutonomousSailboatPlymouth.

Acknowledgement

This internship would not have been possible without Doctor Jian WAN and his guidance and reactivity.

I am grateful with Professor Luc JAULIN, thanks to whom I may not have found this internship and my work would have been much more complicated on autonomous sail-boats.

Lastly, I would like to thank Bob WILLIAMS for all its help and the technical support he provided to me by printing in 3 dimensions all parts I had designed. I also thank Mike to have spent some time to help me adjusting one part I designed in 3D.

François CÉBRON

Abstract

This report summarizes my work in turning a model sail-boat into an autonomous one, at the University of Plymouth.

This begins with the system engineering to present the specifications of this system, including the presentation of all components.

Then you will find the roadmap I chose to follow, with the detailed steps.

In the end, there is a presentation of the miscellaneous work I did.

Key-Words : sail-boat, autonomous, robotics, line-following, potential field

Résumé

Ce rapport résume le travail que j'ai effectué en stage en rendant autonome un voilier de modélisme (radio commandé), à l'université de Plymouth.

Pour commencer, j'évoque la partie d'ingénierie systèmes pour présenter le cahier des charges ainsi que tous les composants du système.

Puis vient la partie parlant de la roadmap que j'ai suivit, avec le détail de chaque étape.

Pour finir, je présente le travail en plus que j'ai fourni.

Mots-clés : voilier, autonome, robotique, suiveur de lignes, champs de potentiels

Table of content

Introduction

University of Plymouth

I performed my internship at the University of Plymouth (see Figure 1, Page 5). This University was founded in 1862 and today it hosts more than 23,000 students each year.

Figure 1: University of Plymouth, View of the Campus (Source [1])

My supervisor was Doctor Jian Wan, a lecturer in control systems engineering and member of Autonomous Marine Systems (AMS) Research Group.

During this internship, I worked in autonomy at the University, doing some code, electronics, mechanics and robotics.

My project

My project was a proof of concept to evaluate the feasibility of setting up an autonomous sail-boat. In fact I developed a prototype of autonomous sail-boat, starting with a RC-Laser model sail-boat (of 1 meter length) and some cheap components including an Arduino Mega. This project will be declined to a bigger boat, because this is the first

time that such a robot is built at this university, so the aim was to discover all issues and concepts bound to robotics sail-boats.

This will be improved with students of the University and possibly used to compete at the World Robotics Sailing Championship (WRSC[2]). This is a robotic competition which consists in developing an autonomous sail-boat and improving the way to sail with it, in order to develop the knowledges around robust algorithms to control autonomous sail-boats. Moreover, this comes with ecological concerns, and the idea to have an autonomous an energy-efficient system, powered by the wind and or the sun.

This project may have some prospects with a possible partnership with the University of Plymouth and the society Msubs [3]. The company is is currently working on the Mayflower autonomous ship [4](pioneer trimaran autonomous to cross the Atlantic Ocean, and was named in reference to the 400th anniversary of the Pilgrim father's boat) and that I had the opportunity to visit with Professor Luc JAULIN and Doctor Jian WAN.

Possible applications of this technology

Robotics is a field in which we develop automation solution to perform some tasks. So this project is interesting itself, but some applications could be more tangible like, for example :

- To use the automated system for marine engineers and perform samples in the sea (conductivity, deep, magnetic field detection,...).
- For harbour protection, if equipped with high range sensors.
- To allow people of having an insight on boats in danger, or locations on the sea which are perilous (like for example the Bermuda Triangle).
- It can also be used as a relay to send data from an Autonomous Underwater Vehicle (AUV), because means to communicate underwater are very limited in range and technology, so we need to use a buoy or a boat to relay data to a workstation on the lands.

Chapter 1

System engineering

To introduce clearly the aim of my project, I will use tools from System Engineering.

1.1 Specifications

I chose to use the APTE Method to describe the Specifications. This method was created by a society name APTE and is composed of some tools which most famous ones are the Horned Beast and the Octopus one. You can see the main idea exposed with the following Horned Beast (see Figure 1.1 Page7).

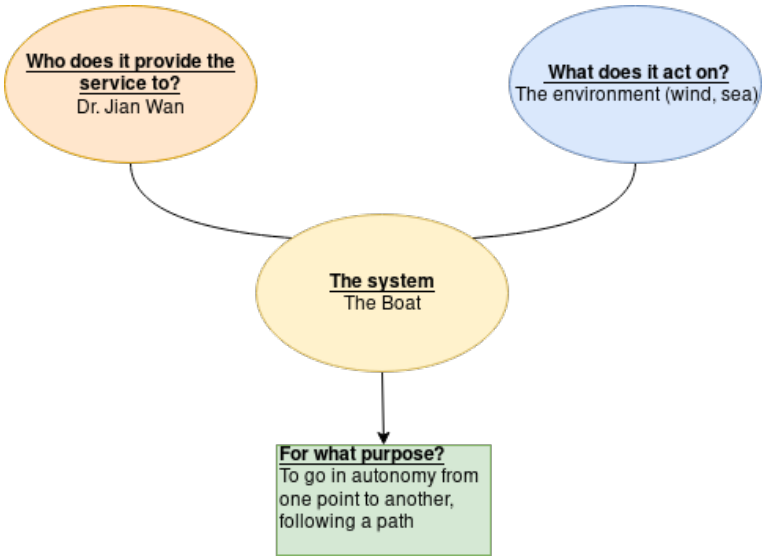

Figure 1.1: Horned Beast of the system

The system is supposed to follow the following requirements:

- The boat is supposed to go in autonomy from one location to an other, following a path, planned before the beginning of the mission.
- The system has to generate some logs to be able to plot the position of the boat, its orientation, the orientation of the wind, of the mainsail and the rudder.

- The bat has to be capable of competing at the World Robotics Sailing Championship (WRSC [2]).
- It has to be powered by an Arduino.

1.2 Functional Architecture

To have a good overview of the system and the function of each component, I chose to use a FAST diagram (see Figure 1.2 Page 8). This highlights the main functional solutions to perform the objective.

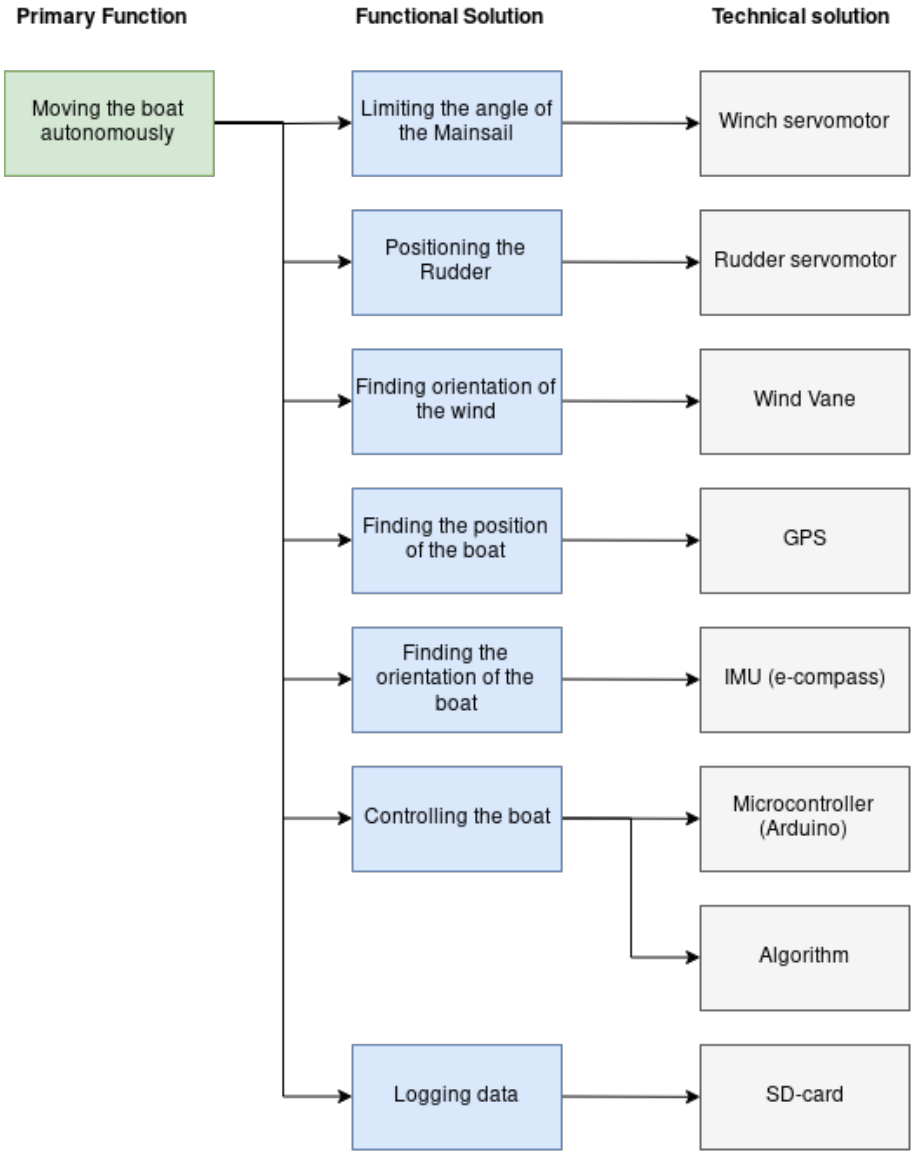

Figure 1.2: FAST diagram

1.3 Technical Architecture

The system an its algorithm requires to have several data like :

- The position of the Boat and its aim.
- The orientation of the Wind.
- The heading of the Boat.

Considering these data, we are capable of controlling the Boat using a line-following method, if we can control :

- The angle of the Main Sail.
- The angle of the Rudder.

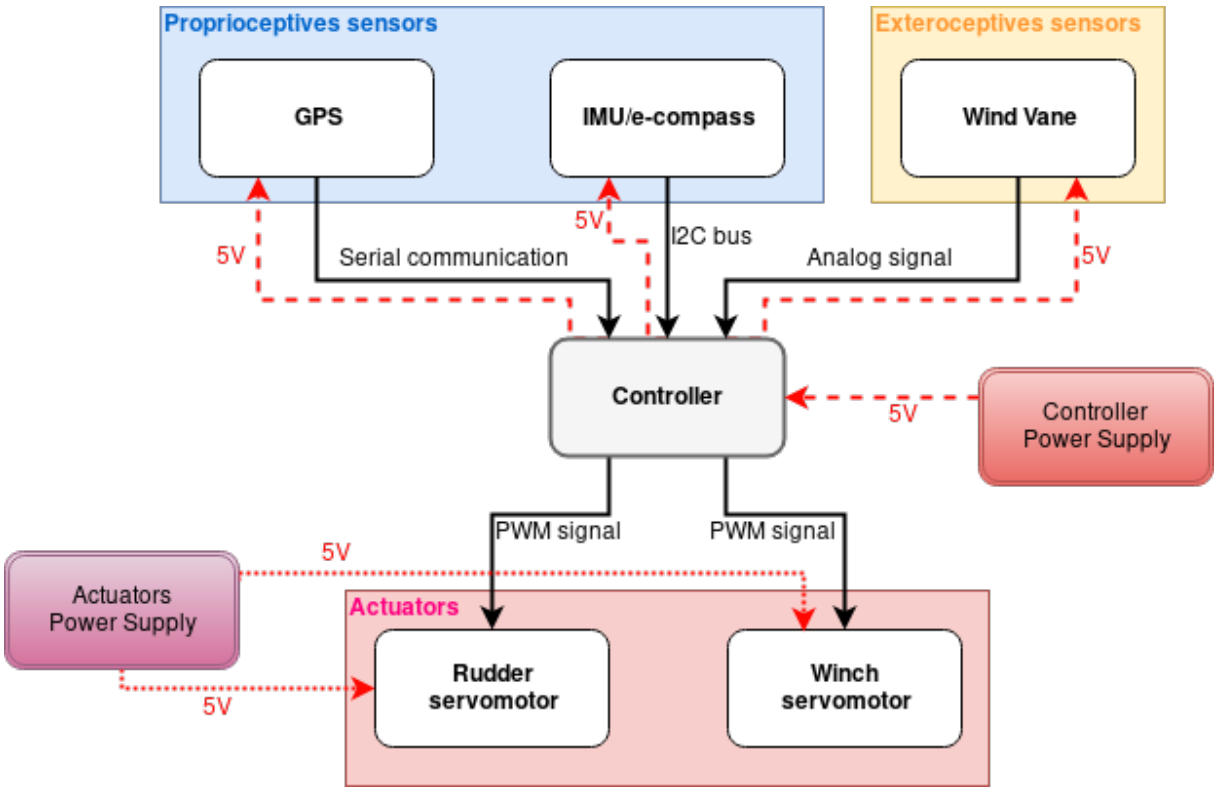

Figure 1.3: Electronic architecture of the system

1.3.1 The Boat

The boat I had to work with was a RC-Laser of 1 meter length (see Figure 1.4, Page 10).

The most important characteristics of this boat are:

- The rudder and the winch are set in motion with included servomotors.

Figure 1.4: RC-Laser (Source [5])

- The transmission system to set the rudder and the mainsail in motion are included.
- The mast is bent and free-rotating with regards to the boat.

1.3.2 The Controller

The controller is an Arduino Mega 2560. This is an open-source and very common, easy-to-use microcontroller.

The Arduino Mega 2560 is a 8-bits microcontroller which possesses 54 digital I/O pins, 16 analog inputs pins, 4 hardware serial ports (UART), an I2C bus and an SPI bus. Among the digital pins there are 6 digital pins usable for interrupts and 15 pins providing PWM output. This controller requires a power supply with at least 5V and with a maximum of 12V. The on-board power managing system can provide a

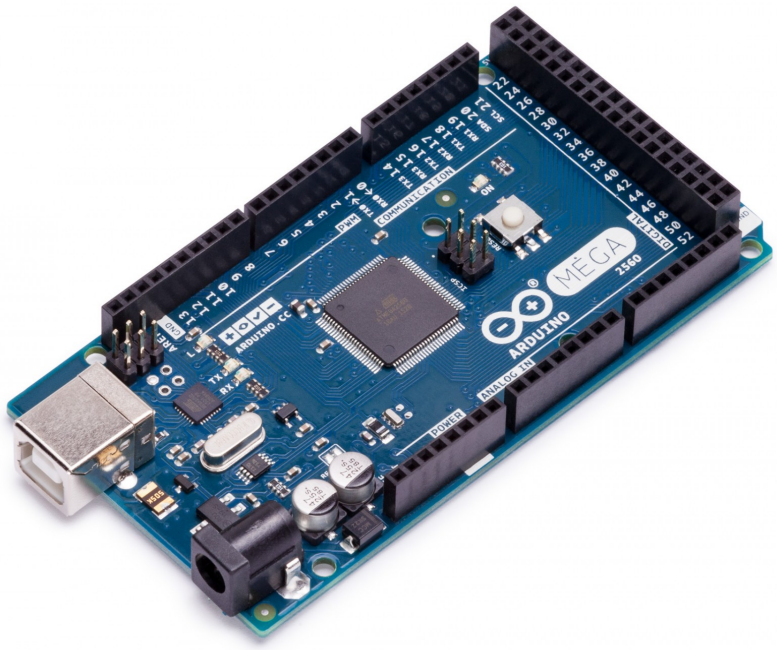

Figure 1.5: Arduino Mega 2560 (Source [6])

5V power supply and also a 3.3V one.

1.3.3 Sensors

Wind vane

The wind vane used is an "e-vane from Inspeed"(see Figure 1.6, Page 12). This is a hall-effect sensor which is analogic and sends the position on the head through a signal in a range going from 5% to 95% of the input voltage (Vcc - I chose 5V to have a better accuracy). The resolution of this sensor is 0.025 degree and its accuracy goes from +/-0.3 to 0.5% of the signal range.

This sensor is used to find the orientation of the wind with regards to the boat. To find the absolute orientation of the wind (with regards to the North) we need to add the orientation of the boat - which is given by the Inertial Measurement Unit (IMU).

The location of this sensor was a bit problematic, because usually this kind of sensors is supposed to be located on top of the mast, to have the less disturbed measure possible. In our system, the mast is bent, so it is not really easy to stick the sensor on top of it. Moreover the mast is free-rotating with reference to the boat (usually, only the boom is rotating), so the wire of the wind-sensor would have generated disturbances to the movement of the mainsail and we would have needed an encoder to know its position with regards to the boat.

Knowing all these things, I have decided to put the wind vane on the back of the boat, because in most of the cases the wind will came from the back to have the boat move. This was difficult because to put it too much in the back would have change the

Figure 1.6: e-vane from Inspeed (Source [7])

balance of the boat, so I have decided to put it the closest possible to the mainsail.

Interesting thing to know:

Sometimes the head of this sensor could leave its body (in the case that you try to remove its support for example), if this thing happens you will need to open the sensor, put its magnet back to the axis and put everything back together (it would not be fix by only putting back the head in the body).

GPS

The GPS I used is a GY-NEO6MV2 (See Figure 1.7 Page 13). This kind of sensor uses Serial port to communicate with the Arduino, moreover it is a 3.3V sensor so you need to use a level-shifter (See Figure 1.17 Page 20) between it and the Arduino board.

To parse data from it, it is required to use the Tinygps++ library [9]. After this I used a a flattening formula which is useful to simplify the upcoming computations. This simplification is valid if the robot does not moves more than 100km (For more details, see source [10]):

$$X = EarthRadius \times (latitude - GpsLatInit) \times cos(GpsLongInit) \tag{1.1a}$$

$$Y = EarthRadius \times (longitude - GpsLongInit) \tag{1.1b}$$

IMU

The IMU I started with was an "ArduIMU" which is a clone of the "9 Degrees of Freedom - Razor IMU" (See figure 1.8 Page 14).

Figure 1.7: GPS GY-NEO6MV2 (Source [8])

This is an accurate IMU, but after all the calibration protocol, and having modified the code I reached a dead end. In fact this sensor communicates through SPI and UART (Serial Port). I couldn't use the SPI protocol because the sensors required to be master on this protocol and I was already using it with the SD-card reader (on which the Arduino had to be master). So I tried the UART and it worked well (1 datum lost every 10 sent), but once I tried to use it on my main code I had more than 19 datum lost among 20 sent. In fact the serial protocol uses a buffer to receive data and in my code I think that as I used another serial port to communicate with the computer and some other land-bus using interrupts (it may be hidden inside of the driver), It corrupted data I was supposed to receive from the IMU. So this sensor is easy to use but only with computers. There is another version of this one, implementing an I2C land-bus and it seems to be feasible to use it with an Arduino, but not the one I had.

So we decided to buy another IMU communicating through I2C land-bus, an "MPU9250 9-axis 9DOF Acc, Gyro, Compass Module" (See Figure 1.9 Page 15).

This sensor is a bit different that the Razor. There is a Digital Motion Processor (DMP) inside, which is supposed to contain parameters of the factory calibration but is also proprietary and we can't easily access to its code.

This sensor has to be powered with 5V but its wires sending data needs to work on 3.3V so I also used the level-shifter (See Figure 1.17 Page 20) with this sensor.

With this sensor, I only used the magnetometer (e-compass), because I tried to implement a tilt compensated compass but I did not succeed and I would have needed more time to complete it.

The calibration protocol contains the following steps:

- The soft-iron
- the hard-iron

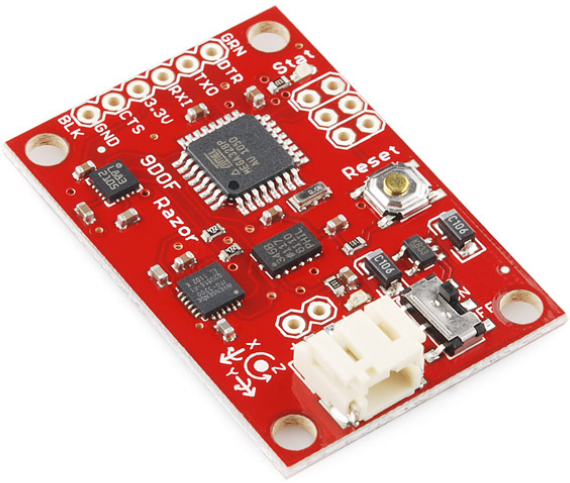

Figure 1.8: 9 Degrees of Freedom - Razor IMU (Source [11])

- Geographic compensation

To understand this, it is required to see the magnetometer as an arrow oriented to the magnetic North of the Earth. The arrow is normalised, so if we move around the sensor we will draw a sphere.

Due to the soldering and other parameters, the sensors's output does not draws a sphere every time. So the first part, called soft iron, consists in finding the coefficients to have it look like a sphere. (if you follow a plane, it is more an ellipse than a circle. So we use a vector containing the corrected coefficients to compensate soft iron values.

The second part called hard-iron consists in finding the centre of the sphere and translating it to the position (0,0). This is traduced by an offset.

There is another part which consists in correcting the values of the compass to make it point to the geographical North. This depends on the place we are on the Earth.

1.3.4 Actuators

The two following servomotors are considered as analog servo, not because we need some analog pins to communicate with them, but because we command them with PWM (so digital pins).

Winch Servomotor

The Winch Servomotor used is a "Hitec HS-785HB Winch Servo" (see Figure 1.10, Page 16).

The Winch servomotor has a range going from 0 to 2826 degrees. This is used to limit the angle of the mainsail. To find the exact equation to control the mainsail I first

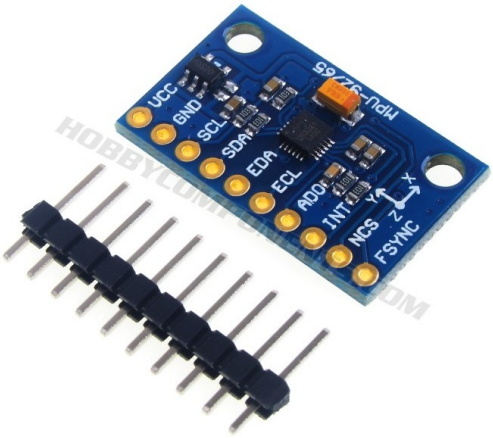

Figure 1.9: MPU9250 9-axis 9DOF Acc, Gyro, Compass Module (Source [12])

began with the vectorial formula of the geometrical system, of the sheet (See Figure 1.11 Page 17).

So the law controlling the sheet is the following one (DistanceWinchBoom is the height of the boom, with regards to the winch):

$$t1 = DistanceMastMainsailSheet^2 + DistanceMastRing^2 \tag{1.2a}$$

$$t2 = DistanceWinchBoom^2 \tag{1.2b}$$

$$t3 = 2 \times cos(sailAngle) \times DistanceMastMainsailSheet \times DistanceMastRing \tag{1.2c}$$

$$SheetCommand = \sqrt{t1 + t2 - t3} \tag{1.2d}$$

After this, I used Pythagore's formula to find the length of the rope (See Figure 1.11 Page 17):

$$ropeCommand = \sqrt{|(RopeRingMax - sheetCommand)^2 - DistanceRingRope^2|} \tag{1.3a}$$

$$ropeCommand = RopeMax - ropeCommand \tag{1.3b}$$

In the end, I only had to transform the length of rope into an angle of the servomotor

$$winchCommand = \frac{2 \times ropeCommand}{WinchDiameter} \tag{1.4a}$$

$$winchAngle = winchCommand - WinchOffset \tag{1.4b}$$

Figure 1.10: Hitec HS-785HB Winch Servo (Source [13])

Rudder Servomotor

The Rudder Servomotor used is a "Hitec HS 322 HD" (see Figure 1.12, Page 18).

For the rudder, the main concern is to discover the range on which it has to work, because the mechanical system puts some thresholds and if we go beyond, we can destroy the whole system.

RC transmitter & receiver

The RC transmitter & receiver is a "Joysway J4C05 2.4GHz 4CH Transmitter Receiver" (see Figure 1.13, Page 19).

The component has 4 emission channels and 5 reception channels (the 5th is used to know if the remote controller is “on” or “off”).

The main issue is that it is a cheap module, so we can’t read values in direct flux from the receiver, we need to wait about 10 milliseconds between each readings. Moreover, the emission is not continuous and sometimes you have latencies, so that you can wait 1 second before receiving your signal.

On this component, I have used only 3 channels :

- Channel 1: for the rudder (the sideways stick - right stick)
- Channel 3: for the mainsail (the throttle stick - left stick)
- Channel 5: for the status (the power button makes the signal go from 1000 to 2000 on the receiver)

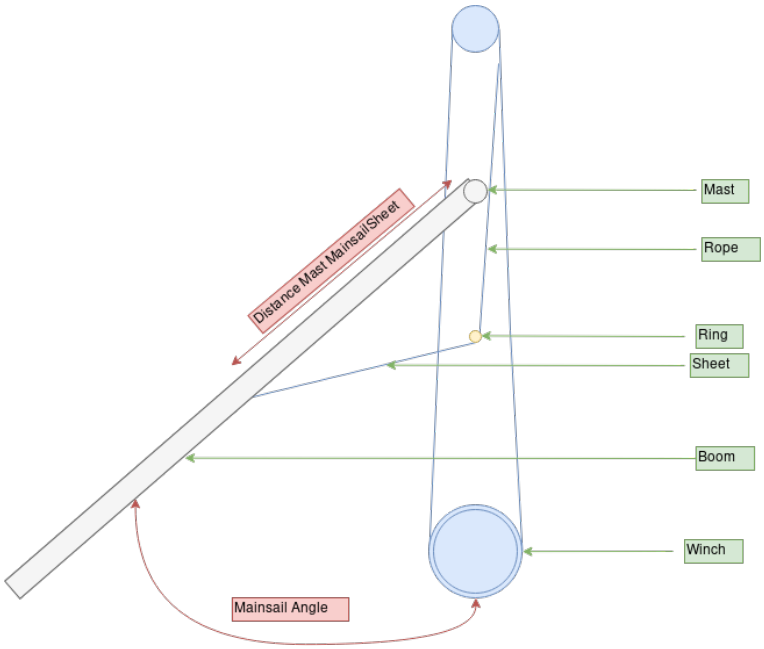

Figure 1.11: Top-down view of the ropes of the boat

1.3.5 Data logging components

SD card reader

To have a log of the system, I used a "Micro SD card reader module for Arduino" (See figure 1.14 Page 19).

This module uses the SPI protocol to communicate with the Arduino. The most important thing to know according to this module is that it only accepts to write data with 8.3 file-name format. This means that to write data in a file, you need to name it with a word containing at least 8 characters, a dot and an extension name containing 3 characters.

LCD Screen

The LCD screen (See Figure 1.15 Page 20). This module is more for debugging than to use it on the system once finished.

The black component which is on the back of the screen (see Figure 1.15 Page 20) is an I2C driver, so this works through this protocol.

Real Time Clock

The Real Time Clock (RTC) I chose to use is a DS3231 (See Figure 1.16 Page 20).

This component allows the Arduino to have the exact time and date, because each time the Arduino is turned off, its internal clock is set back to zero. I could have used the GPS for the time and date (it is included inside of the NMEA frames), but the GPS is not synchronised every-time, so I would had data without times. I also chose to use

Figure 1.12: Hitec HS 322 HD (Source [14])

it to generate a file-name (to easily find the log I named them as following, "MMDDH-HMM.TXT" with MM for month, DD for day, HH for hour and MM for minutes).

The main issue with this module is that it contains 2 I2C ports and the main one (which can't be modified) has the same address than the IMU, so I had to chose to use one or the other and I removed the RTC module. Later on, it would have to be changed or the IMU's address.

1.3.6 Miscellaneous

Level shifter

In order to allow the IMU to communicate with the Arduino, I needed to convert a 5V signal into a 3.3V one. For this purpose, I used a bi-directional level shifter (see Figure 1.17, Page 20) which purpose is to adapt the maximum range of a sinusoidal signal from one level to another. In my case I put it between the sensor an the Arduino board, the lower-level reference I/O is linked to the 3.3V signal while the higher-level reference is to the 5V signal.

Figure 1.13: Joysway J4C05 2.4GHz 4CH Transmitter Receiver (Source [15])

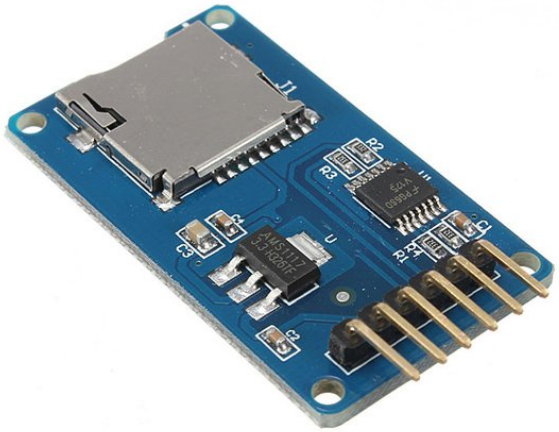

Figure 1.14: Micro SD card reader module for Arduino (Source [16])

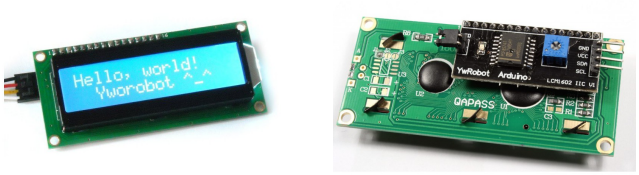

Figure 1.15: The LCD screen (Source [17]

Figure 1.16: RTC DS3231 (Source [18])

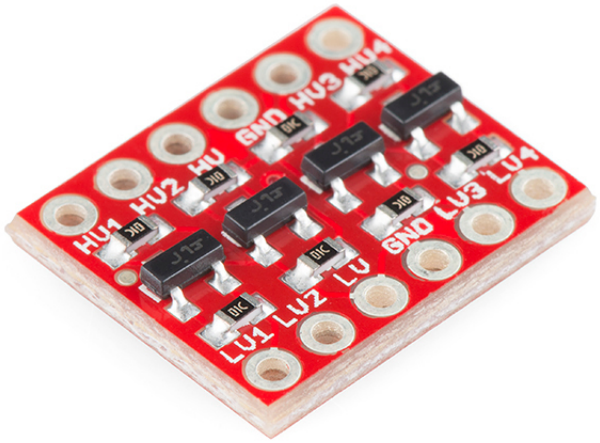

Figure 1.17: Bi-Directional Logic Level Converter (Source [19])

1.3.7 The Algorithm

The line-following algorithm

I chose to apply a potential field method on the line-following. For this, I took the algorithm from Luc JAULIN and Fabrice LEBARS (Source [20]). I selected this one, because it is optimised for a microcontroller.

The algorithm is the following one:

Function in: $\mathbf{m}, \theta, \psi, \mathbf{a}, \mathbf{b}$; out: $\delta_r, \delta_s^{\max}$; inout: q

1 $e = \det\left(\frac{\mathbf{b}-\mathbf{a}}{\|\mathbf{b}-\mathbf{a}\|}, \mathbf{m}-\mathbf{a}\right)$
2 if $|e| > \frac{r}{2}$ then $q = \text{sign}(e)$
3 $\varphi = \text{atan2}(\mathbf{b}-\mathbf{a})$
4 $\theta^* = \varphi - \frac{2.\gamma_\infty}{\pi}.\text{atan}\left(\frac{e}{r}\right)$
5 if $\cos(\psi-\theta^*)+\cos\zeta < 0$
6 or ($|e| < r$ and $(\cos(\psi-\varphi)+\cos\zeta < 0)$)
7 then $\bar{\theta} = \pi + \psi - q.\zeta$.
8 else $\bar{\theta} = \theta^*$
9 end
10 if $\cos\left(\theta-\bar{\theta}\right) \geq 0$ then $\delta_r = \delta_r^{\max}.\sin\left(\theta-\bar{\theta}\right)$
11 else $\delta_r = \delta_r^{\max}.\text{sign}\left(\sin\left(\theta-\bar{\theta}\right)\right)$
12 $\delta_s^{\max} = \frac{\pi}{2}.\left(\frac{\cos\left(\psi-\bar{\theta}\right)+1}{2}\right)$.

Figure 1.18: Line-following algorithm (Source [20])

To understand easily the algorithm, the following schematics is helpful:

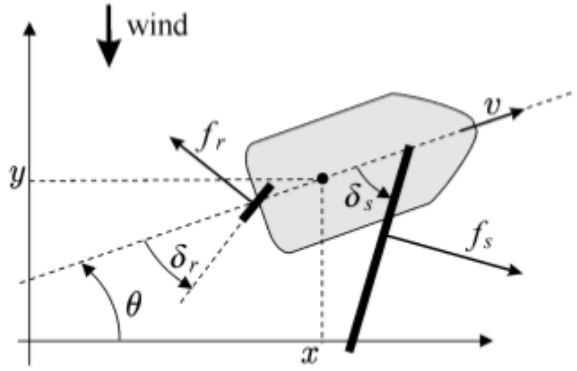

Figure 1.19: Complementary schematics for the algorithm (Source [20])

Details of the algorithm:

- 1 - The sign of 'e' indicates the side of the line, on which we are: e < 0 i.e. right side and e > 0 i.e. left side.
- 2 - If this step is validated, the boat has to tack. 'q' indicates the side of the tack.
- 3 - ϕ is the angle of the aim (line between 'a' and 'b), in the geographical reference, so the angle is with respect to the East.

- 4 - Correction of the trajectory, with the attraction of the line.
- 5-7 - If this is activated, the command has to be corrected.
- 8 - No correction.
- 10 - Controlling the rudder: soft command of it.
- 11 - Bang bang command.
- 12 - Finding the exact command of the sail.

The extension to way-point following

To extend the line-following code into a path-following one, I chose to add some transition area (the yellow circles in the schematics - See Figure 1.20 Page 22). The principle is very easy, when the boat reach a distance lower than the width of the corridor, the algorithm increment its aim, so the next way-point is the new aim.

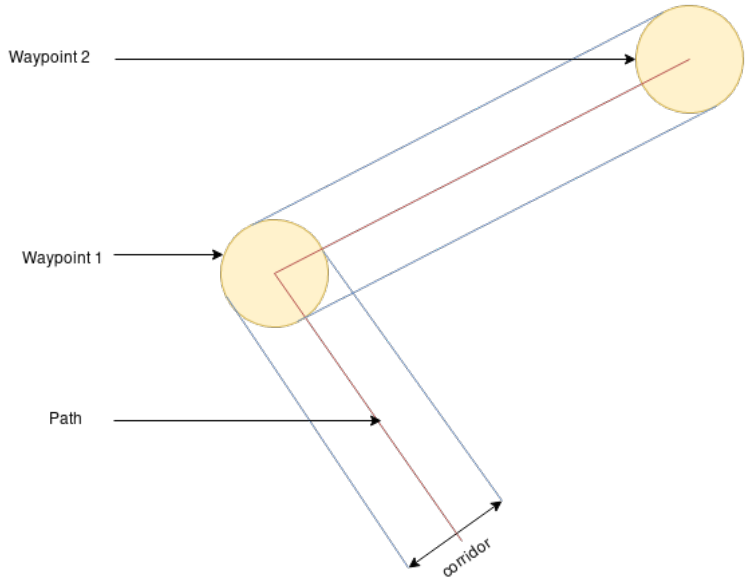

Figure 1.20: Schematics of the path-following

Chapter 2

Project management

To tackle this project I tried to follow a roadmap which I had updated after with the exact dates (see Table 2.1, Page 23). The roadmap may not be exact because in my way of doing, when I worked during a long time on one thing, I switch to something else, to vary and keep me dynamic. So I started to design the wind-sensor support a lot before the date I have put.

ID	Task	Beginning	End
T01	System Engineering	26/06/2017	07/07/2017
T02	Implementation of modules for the manual mode (RC, servos)	07/07/2017	20/07/2017
T03	Manual mode	21/07/2017	24/07/2017
T04	State of the art of the different ways to control the boat	24/07/2017	27/07/2017
T05	Logging mode (LCD, SD card, serial)	27/07/2017	03/08/2017
T06	Implementation of each module	04/08/2017	28/08/2017
T07	Creation of the support for the wind sensor	11/08/2017	16/08/2017
T08	Creation of the cover for the boat	22/08/2017	24/08/2017
T09	Creation of the case for the IMU	24/08/2017	28/08/2017
T10	Implementation of the algorithm for the automation of the boat	28/08/2017	15/09/2017
T11	Wiring lasting	13/09/2017	15/09/2017

Table 2.1: Roadmap of the project

T01: The System Engineering task concerns everything I did to fully understand the way to perform my work (reading publications, finding datasheets, using methods like FAST,...).

T02: The Implementation of modules for the manual mode (RC, servos) task concerns the implementation of all components required to control the boat with the remote (so every components included with the RC-laser), with Arduino. So I had to

cope with the fact that the RC-receiver was not generating (or receiving) a continuous signal, and also to find the geometrical law to control the angle of the mainsail with the winch servomotor.

T03: The Manual mode task was only to integrate everything, fix integration bugs and clean up the code.

T04: The State of the art of the different ways to control the boat, was the time I used to read publication, and try to understand which was the most feasible solution according to the components I had (The Arduino has few computation capacity, so for example matrix computation is not advised on this platform).

T05: The Logging mode (LCD, SD card, serial) task was the moment I realised that as the project contained more than one file, I couldn't use the serial port easily on each file (due to some priority issues of its initialization). So I decided to develop my own library, containing the serial port but also the SD-card reader, and the LCD screen. Then I reviewed it and added a system of priority among log messages in order to have every information saved and to easily sort them (parsing code) to separate debugging information from logging data.

T06: Implementation of each module task was the same thing that the T02, but in order to have an autonomous boat, so I had to add the GPS chip, the Wind vane, the IMU and the RTC. During this task I discovered that the IMU I chose to use was adapted to be used with a computer but not to be used with a microcontroller (my code with only the IMU had an acceptable success rate, but once integrated with the full code I had less that 5% success rate of reading data), so we changed the IMU.

T07: The creation of the support for the wind sensor task was the time I used CATIA to design the top part of the boat, then the wind vane (to have a good reference) and at least I designed the support of the wind vane around it. Then I had it be 3D printed and adjusted.

T08: The creation of the cover for the boat task was the same as for the wind-vane, because the wind-vane involved to have a wire coming from the outside and going inside of the electronic case, so I preferred to design a cover with a hole to allow the wire to go in. I also designed a mould for an O-ring to be sure that it is waterproof, but the 3D printed case is not well suited to be used with O-rings.

T09: The creation of the case for the IMU task was the same as for the cover. I tried to use the same design. The IMU has to be away from magnetic disturbances, so I tried to kept it on top of the boat (away from the electronics components); but the case was so tiny that the O-ring did not stayed so I chose to put a silicon seal instead .

T10: The implementation of the algorithm for the automation of the boat task was first to code in C-language the algorithm extracted from the publication[20]. Then to integrate this code in the main one and adapt all other codes to work with it. And in the end to simulate some systems to be able to have the boat work without them (for example, in the end I had to make a demo in front of people financing the project, so I simulated a GPS with the e-compass).

T11: The Wiring lasting task was to re-think the way everything was plugged to

had the less wires possibles to avoid issues due to wires which are disconnected.

Concerning the application of the roadmap, I chose to train myself at the same time of coding in the idea that a team could be working on the code at the same time that me. So I used Git, I tried to segment my code to make it easy to understand. I also used Doxygen to generate a documentation.

To sum it, with this project I tried to improve the way I work and to learn things which are really important for a coder today.

Chapter 3

Miscellaneous work done

3.1 Code

I developed a debugging library for Arduino and my project, implementing the Serial Port, a LCD screen, a LED, an SD card reader and a Real Time Clock (RTC). This library contains several debugging levels in order to have an output that can be parsed to show only interesting informations according to your aim (log, debug or both at the same time).

To save all data and debugging logs in the same file, I chose to add a number on the first column of each line in this file, indicating the debugging level:

- 3: The highest level (mostly informations for the user, through the screen).
- 2: Data.
- 1: Name of the function which is working.
- 0: Lower level: all important informations like the values of temporary variables.

So, for example if I want to print all GPS locations of the boat, I will parse the file and keep only data which indices are "2", and so on.

I also managed to use Doxygen, so that I have generated an HTML documentation of my code to make it easier for anyone to understand the way my code is done.

I used Git with framagit (a GitHub-like website, created with Gitlab), so that my code is open-source and all improvements (commits) are noticeable.

3.2 3D-design

I have design the Wind-Vane's support to let it stand on the boat (See Figure 3.1, Page 27).

The Wind-Vane support had to validate the following requirements :

- To have the Wind-Vane stand vertically.

- To avoid the Boom from hitting the head of the Wind-Vane.
- Not to disturb functioning state of the sensor (allowing wires to go in the sensor, allowing the user to tune screws...).

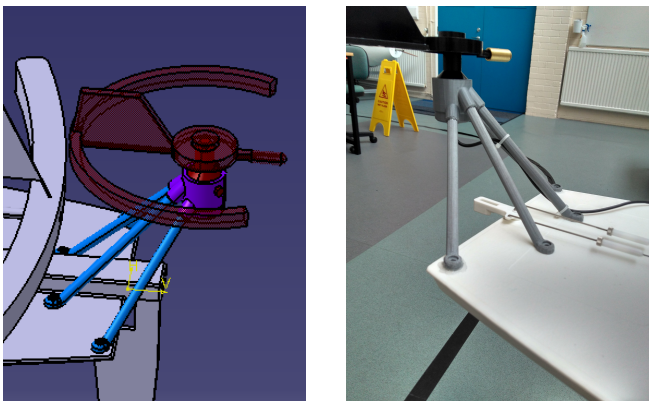

Figure 3.1: (left) 3D-Design of the Support. (right) Wind-Vane support printed

I designed the cover of the boat to be sure that all electronics components were not in contact with water (See Figure 3.2, Page 27).

Figure 3.2: (left) 3D-Design of the cover of the boat. (right) Cover printed

The Cover had to validate the following requirements :

- Not destroying the top of the boat.
- To allow wires from the IMU and the Wind Vane (I also added an output for the GPS), to go inside of the boat.
- To be easy to seal/open.

I also designed the IMU case to let it be away from all magnetic disturbances and also to be waterproof (or at least splash-proof, because 3D printed materials are not water-proof) (See Figure 3.3, Page 28).

The Case had to validate the following requirements :

- To allow wires from the IMU to go outside of the case.

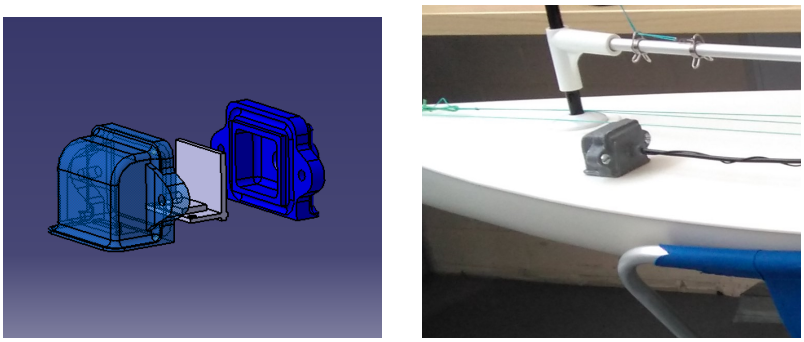

Figure 3.3: (left) 3D-Design of the case of the IMU. (right) Case printed

- To be easy to seal/open.
- Not to let the IMU move inside.

To cope with all measurement requirements, I had designed the top part of the boat to design my parts with a reference. I added no-go zones (like the mechanical transmission of the rudder) or the potential position of some parts (like the extrema of the boom or the wind-vane, with semi-circles that you can notice in the Figure 3.4, Page 29).

Moreover, as every parts I had to design would be printed in 3D, I took into account the following constraints of the 3D printer (Makerbot Replicator 2):

- The size of my parts couldn't exceed the size of the printing area ($160mm \times 160mm \times 160mm$). Sometimes this induced to re-design the part to keep it tough enough (See the Wind-Vane support which is composed of 5 parts, Figure 3.1 Page 27).
- For every holes, I had to add an correction coefficient on the diameter (because the software respects externals dimensions, but not internals ones like holes). Furthermore, this was not an exact science, because when you have a printer, you are supposed to perform a lot a tests to know the maximum angle of the printing (when you have a bent profile) and the correction on the diameter of holes. But I asked people who used the printer and they gave me 2 coefficients, so I decided to use one for tiny holes ($+0.2mm$ on the diameter) and the other one for big holes ($+0.4mm$ on the diameter).

Following these requirements, it worked well because I had only one part which was not exactly fitting, but after a bit of adjustment it was perfect.

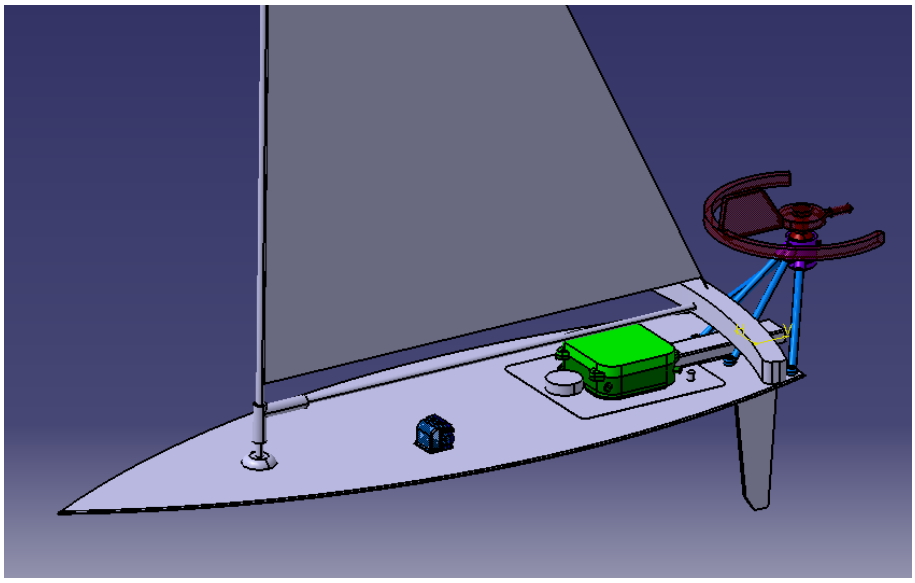

Figure 3.4: The 3D modelling

Conclusion

Improvements of the system

According to the main features, actually the boat is not capable of doing some position keeping

To add some important features, I think that implementing some water-detection sensors could be interesting in order to protect all electronics components by cutting the power-supply in case of water-intrusion.

Moreover, one major feature is to be able to see logs in real time, in order to see problems before they happen and to be capable of developing some supervision tools to manage the boat's path. To do this, I think that an ESP8266 could be really useful (it is an Arduino-like board, which costs less than 2£ and provide Wifi connectivity).

What I have learnt during this internship

Technical Side

Concerning the technical side, I worked a lot to learn some specific skills, like using Git to manage my code. Doxygen to comment-it and document-it, and I have tried my best to optimise my code with Arduino, using pre-processor macro to have a lighter code on the microcontroller. Moreover I tried to apply an Object-Oriented way to program, without an Object-Oriented language (I used C language). So that each sub-system can be deactivated and the full code will work (with a simulated sub-system in most of the cases) in order to perform some unit test and integration tests.

Organizational Side

Concerning the organization, I have discovered that I have to work on my way of doing things to better think before starting doing them. This is the main explanation of why I did not completed the project on time, because I was to busy trying to build definitive things when I needed to go straight to the point. I think that my way of approaching a problem needs to follow more a System-Engineering method, to be able to step back and look at the overall picture of the system, more than looking for the perfect way to do things.

To conclude

This internship was very interesting, because it was my first time being abroad and I had to work and interact in English. Moreover, I succeeded in making my point being understood, even with people which don't have the same way of working than me. My fluency in English has improved.

I also discovered in what the research field consists in, so I know a bit more what will be my professional project. Thanks to this I had the opportunity to fulfil my knowledges in C language and Arduino.

I found this international experience very rich, and I would likely work abroad again.

ENSTA Bretagne

RAPPORT D'EVALUATION
ASSESSMENT REPORT

Merci de retourner ce rapport en fin du stage à :
Please return this report at the end of the internship to :

ENSTA Bretagne – Bureau des stages - 2 rue François Verny - 29806 BREST cedex 9 – FRANCE
☎ 00.33 (0) 2.98.34.87.70 - Fax 00.33 (0) 2.98.38.87.90 - stages@ensta-bretagne.fr

I - ORGANISME / *HOST ORGANISATION*

NOM / *Name* University of Plymouth

Adresse / *Address* Plymouth University, Drake Circus, Plymouth, Devon, PL4 8AA, UK

Tél / *Phone (including country and area code)* +44 1752 586157

Fax / *Fax (including country and area code)* _____

Nom du superviseur / *Name of placement supervisor* Jian Wan

Fonction / *Function* Lecturer in Control Systems Engineering

Adresse e-mail / *E-mail address* jian.wan@plymouth.ac.uk

Nom du stagiaire accueilli / *Name of trainee* François Cébron

II - EVALUATION / *ASSESSMENT*

Veuillez attribuer une note, en encerclant la lettre appropriée, pour chacune des caractéristiques suivantes. Cette note devra se situer entre **A (très bien)** et **F (très faible)**
*Please attribute a mark from **A (very good)** to **F (very weak).***

MISSION / *TASK*

- La mission de départ a-t-elle été remplie ? *Was the initial contract carried out to your satisfaction?* — A B C D E F

- Manquait-il au stagiaire des connaissances ? *Was the trainee lacking skills?* — ☐ oui/*yes* ☑ non/*no*

  Si oui, lesquelles ? / *If so, which skills?* _____

ESPRIT D'EQUIPE / *TEAM SPIRIT*

- Le stagiaire s'est-il bien intégré dans l'organisme d'accueil (disponible, sérieux, s'est adapté au travail en groupe) / *Did the trainee easily integrate the host organisation? (flexible, conscientious, adapted to team work)*

  A B C D E F

Souhaitez-vous nous faire part d'observations ou suggestions ? / *If you wish to comment or make a suggestion, please do so here* _____

COMPORTEMENT AU TRAVAIL / *BEHAVIOUR TOWARDS WORK*

Le comportement du stagiaire était-il conforme à vos attentes (Ponctuel, ordonné, respectueux, soucieux de participer et d'acquérir de nouvelles connaissances) ?
Did the trainee live up to expectations? (Punctual, methodical, responsive to management instructions, attentive to quality, concerned with acquiring new skills)?

A B C D E F

Souhaitez-vous nous faire part d'observations ou suggestions ? / *If you wish to comment or make a suggestion, please do so here* Can be more pragmatic in project management

INITIATIVE – AUTONOMIE / *INITIATIVE – AUTONOMY*

Le stagiaire s'est –il rapidement adapté à de nouvelles situations ? A B C D E F
(Proposition de solutions aux problèmes rencontrés, autonomie dans le travail, etc.)

Did the trainee adapt well to new situations? A B C D E F
(eg. suggested solutions to problems encountered, demonstrated autonomy in his/her job, etc.)

Souhaitez-vous nous faire part d'observations ou suggestions ? / *If you wish to comment or make a suggestion, please do so here* _____

CULTUREL – COMMUNICATION / *CULTURAL – COMMUNICATION*

Le stagiaire était-il ouvert, d'une manière générale, à la communication ? A B C D E F
Was the trainee open to listening and expressing himself /herself?

Souhaitez-vous nous faire part d'observations ou suggestions ? / *If you wish to comment or make a suggestion, please do so here* _____

OPINION GLOBALE / *OVERALL ASSESSMENT*

❖ La valeur technique du stagiaire était : A B C D E F
Evaluate the technical skills of the trainee:

III - PARTENARIAT FUTUR / *FUTURE PARTNERSHIP*

❖ Etes-vous prêt à accueillir un autre stagiaire l'an prochain ?

Would you be willing to host another trainee next year? ☑ oui/*yes* ☐ non/*no*

Fait à _____, le _____
In Plymouth, UK, on 14/09/2017

Signature Entreprise _____ Signature stagiaire
Company stamp _____ *Trainee's signature*

School of Engineering
University of Plymouth
Drake Circus
Plymouth
Devon
PL4 8AA

Merci pour votre coopération
We thank you very much for your cooperation

Glossary

AUV Autonomous Underwater Vehicle.

DMP Digital Motion Processor.

FAST Functional Analysis System Technique.

I2C Inter-Integrated Circuit.

IMU Inertial Measurement Unit.

PWM Pulse With Modulation.

RTC Real Time Clock.

SPI Serial Peripheral Interface.

List of Figures

Bibliography

[1] Education Technology. Superfast wi-fi for plymouth students. http://edtechnology.co.uk/Article/superfast_wifi_for_plymouth_students_.

[2] WRSC. World robotic sailing championship. https://www.wrsc2017.com/.

[3] MSubs. Msubs underwater & equipment. http://msubs.com/.

[4] Mayflower Autonomous Ship Ltd. Mayflower autonomous ship. http://www.mayflowerautoship.com/.

[5] Intensity Sails. Rc laser. http://www.intensitysails.com/rclacoretosa.html.

[6] Arduino. Arduino mega 2560 rev3. https://store.arduino.cc/arduino-mega-2560-rev3.

[7] Inspeed. e-vane wind direction sensor. http://inspeed.com/wind_speed_direction/Vane.asp.

[8] DX. Gy-neo6mv2 flight controller gps module - blue. http://www.dx.com/p/gy-neo6mv2-flight-controller-gps-module-blue-232595.

[9] Mikal Hart. Tinygps++. http://arduiniana.org/libraries/tinygpsplus/.

[10] Luc JAULIN. Mobile robotics. pages 69–74 & 153–157, 2016.

[11] Sparkfun. 9 degrees of freedom - razor imu. https://www.sparkfun.com/products/retired/10736.

[12] Hobby Components. Mpu9250 9-axis 9dof acc, gyro, compass module. http://hobbycomponents.com/sensors/720-mpu9250-9-axis-9dof-acc-gyro-compass-module?search_query=9dof&results=1.

[13] Phidgets. Hitec hs-785hb winch servo. https://www.phidgets.com/?tier=3&catid=22&pcid=19&prodid=241.

[14] Wetronic. Hitec hs-322hd servo. https://wetronic.nl/webshop/Hitec_hs-322hd_servo.

[15] RadioSailing. Joysway j4c05 standard 2 4 ghz transmitter receiver. http://www.radiosailing.co.uk/joysway-24ghz-j4c05-radio-system-1049-p.asp.

[16] Tindie. Micro sd card reader module for arduino. https://www.tindie.com/products/mmm999/micro-sd-card-reader-module-for-arduino/.

[17] Banana-Pi. Afficheur lcd 2x16 i2c bleu. https://e.banana-pi.fr/afficheurs-ecrans/33-afficheur-carte-lcd-2x16-i2c-bleu.html.

[18] ebay. Rtc real time clock memory module for arduino ds3231 at24c32 iic precision. http://www.ebay.com/itm/RTC-Real-Time-Clock-Memory-Module-For-Arduino-DS3231-AT24C32-IIC-Precision-/141135165489.

[19] Sparkfun. Bi-directional logic level converter hookup guide. https://learn.sparkfun.com/tutorials/bi-directional-logic-level-converter-hookup-guide.

[20] Luc Jaulin & Fabrice Le Bars. A simple controller for line following of sailboats. https://www.ensta-bretagne.fr/jaulin/paper_jaulin_irsc12.pdf.