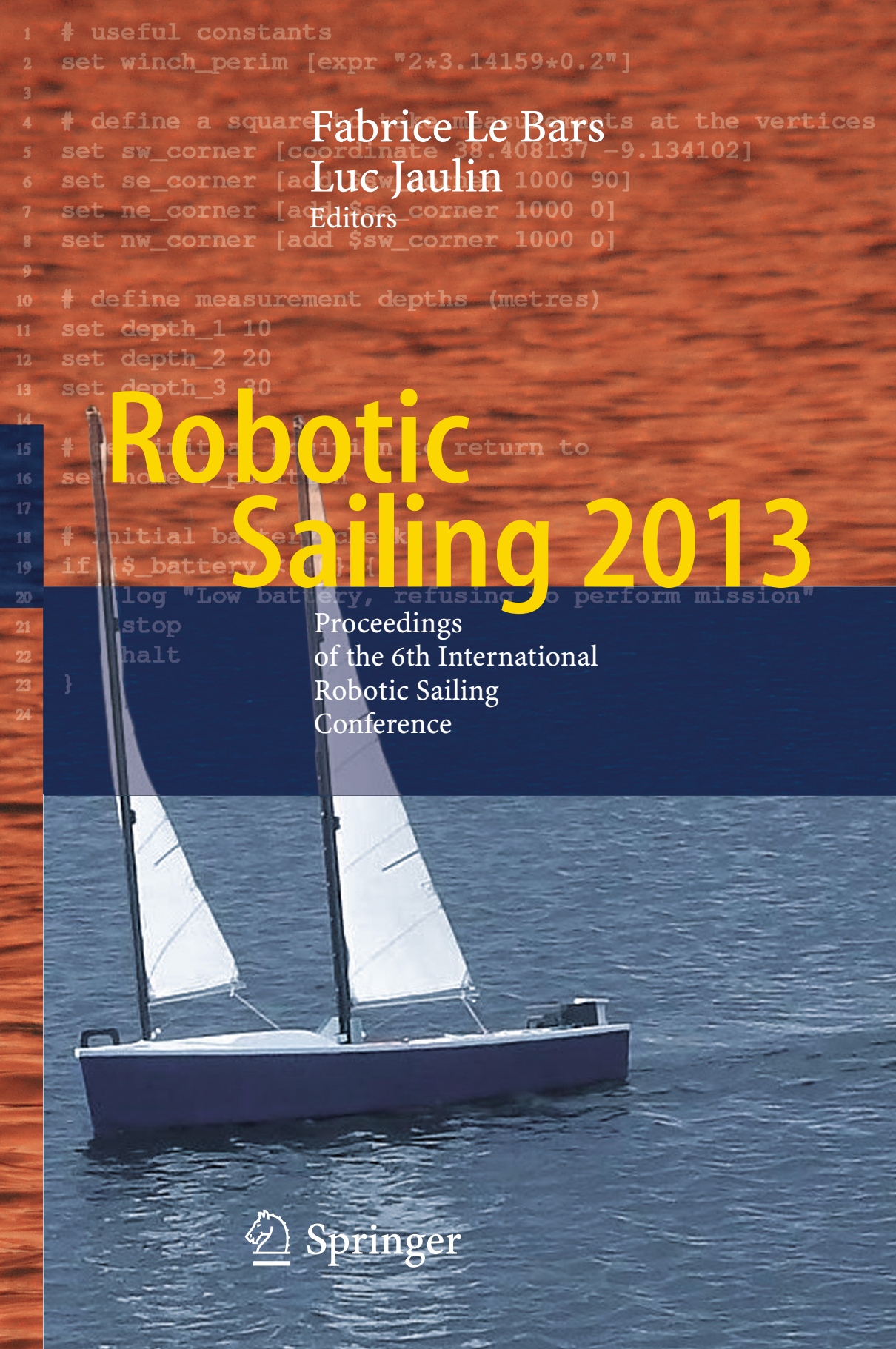

Fabrice Le Bars
Luc Jaulin
Editors
Robotic Sailing 2013
Proceedings of the 6th International Robotic Sailing Conference
Springer

Robotic Sailing 2013

Fabrice Le Bars · Luc Jaulin
Editors

Robotic Sailing 2013

Proceedings of the 6th International Robotic Sailing Conference

Springer

Editors
Fabrice Le Bars
ENSTA Bretagne
Brest Cedex 9
France

Luc Jaulin
ENSTA Bretagne
Brest Cedex 9
France

ISBN 978-3-319-02275-8 ISBN 978-3-319-02276-5 (eBook)
DOI 10.1007/978-3-319-02276-5
Springer Cham Heidelberg New York Dordrecht London

Library of Congress Control Number: 2013947071

Printed on acid-free paper

Springer is part of Springer Science+Business Media (www.springer.com)

Preface

Among all the different types of autonomous robots, the sailboat was probably the last to be fully automated despite the fact that humans use sailboats for thousands of years, contrary to Unmanned Aerial Vehicles (i.e. UAV, which is currently the most developed type of autonomous robot) w.r.t. planes, which exist only since the last century. However, an autonomous sailboat has several advantages:

- Almost unlimited energy: it uses the wind to move, sun and sea to charge its batteries while its power consumption is low compared to that of a motorboat for instance.
- Interesting payload capabilities with respect to its dimensions.
- Cheap.

This type of robot can be used for several applications:

- Oceanography and hydrography.
- Maritime environment monitoring: pollution detection and cleaning, fish studies...
- Meteorology.
- Continuous harbor monitoring: thanks to their important energetic autonomy and their low cost, several sailboat robots can be deployed to monitor local surface and submarine traffic and would notably reinforce systems currently used.
- Assistance and rescue in dangerous areas...

These proceedings contains the papers presented during the IRSC (International Robotic Sailing Conference) 2013 that has taken place in Brest, France, in conjunction with the WRSC (World Robotic Sailing Championship) from the 2nd to the 6th of September 2013. This is the 6th edition in a series with previous events held in Austria (2008), Portugal (2009), Canada (2010), Germany (2011) and the Wales/UK (2012). The World Robotic Sailing Championship (WRSC) is intended to promote the development of autonomous wind propelled sailing robots, through a series of short distance

races, navigation and autonomy challenges. The competition, originally designed for sailboats, was also opened this year to motorboats as a separate category, to try to bring together the scientific communities that work on different types of autonomous marine vehicles. The accompanying International Robotic Sailing Conference (IRSC) provides researchers the chance to exchange ideas with a scientific conference on a wide range of topics around autonomous marine robotics (especially sailing robots).

As a first part of these proceedings, different external and internal hardware design of autonomous sailboats will be described. The following part will present new ideas related to energy and power management, which are key challenges for robots such as those designed for the Microtransat, that need to stay several months at sea to cross the Atlantic ocean. The third part will be dedicated to modeling, simulation, control, and stability analysis of autonomous sailboats and the last part will focus on high level control architectures and algorithms.

The editors would like to thank all the authors, the program committee, all the sponsors, partners and other people that made possible the WRSC/IRSC 2013 in Brest. A special thanks to Annick Billon-Coat, secretary of the Pole STIC at ENSTA Bretagne and Maël Melguen, military student in internship between ENSTA Bretagne and Aberystwyth University, for all the time they spent on the preparation of this event.

July 2013 Fabrice Le Bars

Organization

General Chair

Fabrice Le Bars	ENSTA Bretagne, France
Luc Jaulin	ENSTA Bretagne, France

Proceedings

Fabrice Le Bars	ENSTA Bretagne, France
Luc Jaulin	ENSTA Bretagne, France

Co-Organizers

Annick Billon-Coat	ENSTA Bretagne, France
Maël Melguen	ENSTA Bretagne, France

Program Committee

Roland Stelzer	INNOC, Austria
Yves Brière	ISAE, France
Colin Sauzé	Aberystwyth University, United Kingdom
Mark James Neal	Aberystwyth University, United Kingdom
Alexander Schlaefer	University of Luebeck, Germany
Paul Miller	USNA, United States of America
José Carlos Alves	Universidade do Porto, Portugal
Nuno A. Cruz	Universidade do Porto, Portugal
Benedita Malheiro	ISEP - IPP, Portugal
Ole Blaurock	Luebeck University of Applied Sciences, Germany
Jan Sliwka	CMRE, Italy
Cedric Pradalier	GeorgiaTech Lorraine, France
Erik Maehle	University of Luebeck, Germany
Frédéric Plumet	Universite de Versailles, France
Vincent Creuze	LIRMM, France
Patrick F. Rynne	Florida Atlantic University, United States of America
Justin E. Manley	Teledyne Benthos, United States of America
Arnaud Le Breton	Protei project/DNV, Norway
Thierry Terre	Ifremer, France
Laurent Delauney	Ifremer, France
Olivier Reynet	ENSTA Bretagne, France
Benoît Clement	ENSTA Bretagne, France
Fabrice Le Bars	ENSTA Bretagne, France
Luc Jaulin	ENSTA Bretagne, France

Contents

Part I
Hardware Design

MARIUS: A Sailbot for Sea-Sailing

Cédric Anthierens, Elodie Pauly, and François Jeay

Abstract. This paper deals with the design process of a specific sailing robot designed for sea sailing. So we will present the aims and the context of the project and thus the specifications that guided the sailboat design. Next, a part will be devoted to the methods and tools chosen to define and create such a complex mechatronic system. The third part will describe the role and the influence of the embedded intelligence to safely carry out a given mission. Finally we will conclude this paper with up-coming works.

1 Goals and Specifications

Nowadays many investigations are led on AUV to explore underwater environment but the surface of the oceans as well [1]. All these systems have to adapt to an unknown and varying environment to usually carry out some long missions. The energy management is a key issue for mobile robots such as AUV, this is the reason why there is a great interest in exploiting green and free energy to increase the AUV autonomy. Gliders are based on very

Cedric Anthierens
SUPMECA TOULON LSIS UMR 7296, Maison des Technologies,
Place Georges Pompidou, 83000 Toulon, France
e-mail: cedric.anthierens@supmeca.fr

Elodie Pauly
ISEN, Maison des Technologies, Place Georges Pompidou, 83000 Toulon, France
e-mail: elodie.pauly@isen.fr

François Jeay
SUPMECA TOULON, Maison des Technologies, Place Georges Pompidou,
83000 Toulon, France
e-mail: francois.jeay@laposte.net

F. Le Bars and L. Jaulin (eds.), *Robotic Sailing 2013*,
DOI: 10.1007/978-3-319-02276-5_1,

known marine principles since they intelligently use electric energy from a battery and the ballast principle to carry out long range missions. Such a way to move in water involves dealing with water streams, this makes the system's states very dependent on the environmental parameters. This is the case also for sailing robots that rely only on the wind to move and reach a desired location. Numerous sailing robots work as autonomous surface vehicles to demonstrate the capabilities of an autnomous system to move within an unknown and harsh environment, to make do with and manage its own energy production and also to move within a varying environmment by relying only on a natural energy source, i.e. the wind. All these challenges are very ambitious but also very exciting.

The interest of such drones is to sail unlimitedly by remaining the most environment friendly as possible. So some natural phenomena related to marine wildlife can be monitored provided there is no disturbance generated by the instruments carrier. In this context, the authors decided to create a new sailboat that performs such kind of long missions. To do so and go further than the other sailboats did, the design of the robot was guided to make it robust and resistant against harsh environments in order to keep on properly working in any conditions. For safety reasons, a sailboat cannot be as big as classical sailboats to reduce the risk of dammage to others in case of troubles. Indeed we cannot claim that an embedded controller is always as safe as a skipper for sailing in any cases. Therefore it was decided to design a sailboat shorter than 2 meters and lighter than 100 kg as total weight. In addition, such a size is enough to embed several instruments for oceanographic inspection.

In order to control the sailboat, two modes are required, i.e. manual and automatic modes. This means the manual mode allows a user from a follower boat to remotely pilot the sailbot through to a GUI that runs on a laptop. This mode is necessary to edit the route, to have a look at the logged and current data of the sailbot and to possibly keep hand on the actuators to order the sailbot to stop for instance (it means to heave to). The automatic mode controls the sailboat in order to follow a desired route defined by GPS waypoints for example or to track a specific phenomenon or to stay on the spot to monitor what happens within to desired position. These both modes have implied some specifications for the embedded control unit of the sailbot and its software architecture.

Finally because of the location of our lab (south of France), we have designed a sailboat primarily to sail in Mediterranean Sea that represents a great interest in terms of marine observations. With no comparisons with Atlantic ocean, Mediterranean Sea is however often very windy this may generate big waves. Avalon badly experimented this environment because its rig got broken in last September off Saint Tropez [8]. Our sailbot is thus named MARIUS for Mediterranean Autonomous Robot ISEN Union SUPMECA with a nod to our both institutes ISEN and SUPMECA TOULON.

2 Process of Design

Such a complex mechatronic system cannot reasonably be tackled linear way. Indeed an iterative process is necessary to meet the issues like energy management, navigation skills and so on. An organic description is given below for needs of writing but we keep in mind that many interactions happened during the design process between the mechanics, electronics and control processing.

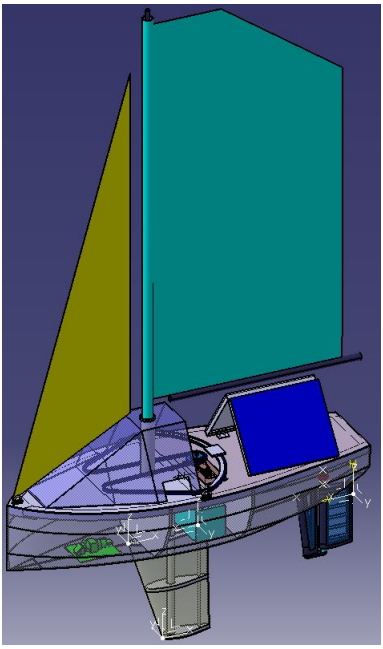
2.1 Mechanical Part

In opposition to other sailbots (ASAROME, VAIMOS, Erwan I), the whole design of Marius started from scratch instead of starting from an existing hull (MiniJi for instance) [1, 6, 7] . The hull was designed with Delftship software, which allows creating chines hulls that are easy to manufacture and cost efficient [4]. So the hull has a shape that favors a good stability and balance of the sailboat rather than a potential high speed (beam: 80 cm). A long keel was chosen instead of a bulb shaped keel to prevent the sailboat from dragging objects that might remain stuck on the keel (Table. 1). So the fin shape of the keel helps potential braced objects such as plastic bags, nets, fishlines... to glide away.

The skeg placed just forward of the rudder has also a fin shape to protect the rudder from the same hazards. The skeg contributes to guide the rudder in rotation as well. The whole hull works in water displacement mode and uses its whole length to raise its maximal speed. The water line passes just at the bottom level of the buttocks provided the total floatting weight nears 100 kg as planned. Obviously such an assumption takes into account the weight of all the components of Marius (including the battery, the instruments, wires and computer unit), the wind force depending on the dragging force (thus the hull shape and the waterline) that lead consequently to size the balance torque generated by the keel (thus the draft and the keel mass). We notice here that we are facing a complex design issue that implies an iterative process to design and to size all these interactive parts. In addition, we mention that a 30 kg payload can get aboard Marius without significantly changing its behaviour for sailing (provided this additional load is well placed within the hull).

A doghouse is fixed upon the deck to help the sailboat to turn back in case it rolled over. The entire hull was made from wood, fiberglass and epoxy whereas the doghouse is in foam and epoxy, the keel, the rudder and the skeg are made from steel, foam and epoxy. The entirely equipped hull weight is 70 kg (including 35 kg of the keel) and the draft is 80 cm long. These features provide the sailboat with a very good balancing torque, which highly contributes to the robustness of the mechanical part.

Table 1 Marius as CAD drawing and in reality on stand



The rig is composed of an carbon-epoxy mast that is 2.4 m long (upward from the deck), a 2.2 m^2 main sail, and a 0.7 m^2 jib sail. The mast is fitted in a 80 cm long stainless tube fixed to the bottom of the hull and the deck, whereas the mast top is braced to the deck by 3 dyneema ropes to give more rigidity to the rig (each rope withstands static loads higher than 1000 kg). The boom and thus the main sail are driven by a slider that moves along a circular rail from 60 ° at port side to 60 ° at starboard (Fig. 1). This dof is controlled by a DC gearmotor fixed on the inner side of the deck. The belt follows a W shaped path and is composed for a part of a chain and for another part of a dyneema rope. The chain is driven by a sprocket fixed on the motorized shaft whereas the rope is guided by the cheek blocks (pulleys) and the circular rail. A short rope is tied to the moving slider and to the sail boom to make them move together.The rudder shaft is also driven by the same type of actuator fixed under the deck, that transmits its torque through a chain and a cog system. Each dof is equipped with a rotative potientiometer, which provides an absolute angle measurement for the rudder and the main sail. The DC motor driver boards are placed into an electric box, which includes and gathers the supply sockets for all the instruments and also the control unit.

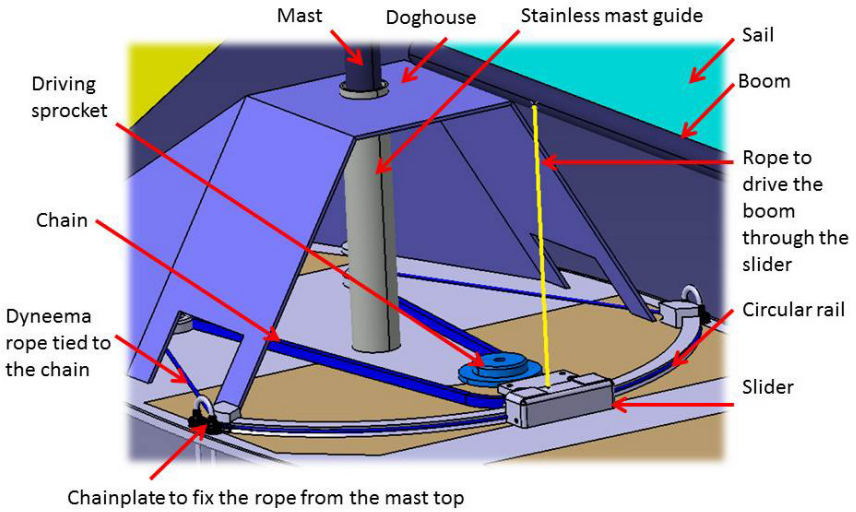


Fig. 1 Main sail actuation principle

2.2 Instruments and Energy

The skipper mode chosen to control MARIUS aims at reproducing the best as possible what skippers usually do. This means relying on several measured data, i.e. heading, GPS position, wind speed and direction, in order to properly control both actuators, i.e. the rudder and the main sail. Many other data related to the states of the sailbot could be monitored as well in order to improve to sailing skills but this might make the skippering algorithm much more complex without assuring noticeable improvements. Such improvements might be studied as future works after upcoming sailing tests are done.

So MARIUS is equipped with a magnetometer compass, a GPS and a wind sensor that are NMEA standard instruments. The wind sensor is an ultrasonic sensor that is placed on the mast top. The GPS is fixed on the deck close to the buttocks whereas the compass is fixed in the hull below the deck to the front of the boat far from any magnetic disturbances. These three instruments are plugged to the control unit through serial ports. Moreover a 3 axis accelerometer has been added not in order to directly control actuators but to monitor MARIUS' behaviors in relation with the wind and waves conditions. This sensor works as inclinometer and primarily helps the programmers to determine the best settings for the main sail angle (depending on the wind speed/direction and on MARIUS' velocity) and also to know whether MARIUS has rolled down.

A 90 Ah gel battery supplies energy for all electronic stuff. The free space inside the keel was initially supposed to be filled with NIMH batteries, but for economic reasons, such a solution was given up. Therefore a much bulkier (23 kg) and classical battery was chosen to be fixed on the bottom of the hull. Two 35 W photovoltaic pannels placed like a tent on the deck provide the battery with energy through a MPPT charger. This charger can simultaneously work with solar pannels or a wind generator. The design of this latter device is presently in progress. A dedicated vertical Savonius wind generator with helicoidal blades should equip Marius soon in order to contribute to the energy production (30 W targetted).

2.3 Control Unit

The control unit has been concurrently specified with the rest of the robot. Because of the short duration of the design phase of this project, a high level of programming language was required. The potential evolution of Marius missions and of its features have pushed the authors to select a modular control unit compatible with many types of signal (digital, analog...). For safety reasons, Marius must be remotely controlled if requested from 100 m around. This feature is necessary to request Marius to stay on the spot (sailing facing the wind), this allows the staff to catch it in order to drag it into the harbor for example. To do so the control unit must be compatible with a wireless communication type to create a private link to a host PC, which runs a GUI program aboard the follower boat. Finally the control unit must be able to be embedded aboard the sailboat and be compatible with the energy requirements (low consumption, low supply voltage if possible). For all this reasons, a Real Time target from National Instrument was chosen. The Compact-RIO 9076 provides four slots for C modules. It can be power supplied from 9 to 30 volts and its consumption cannot exceed 15 W. Presently, three connected C modules feature four RS232 serial ports where the GPS, the compass and the wind sensor are plugged, eight analog inputs for the 3 axis accelerometer and the both potentiometers (main sail and rudder), and four analog outputs to drive the both DC motors. About the half of the whole ressources remain avalaible therefore the actual energy consumption of the control unit is about 6 W. This control unit offers a USB port and a RS232 ports to connect peripherals and an internal storage memory of 512 MB. A bluetooth dongle (1,2 W) plugged on the control unit provides it with a long range wireless communication way (up to 300 m according to the provider and tested up to 100 m).

The control unit, a supply board, the two motor driver boards are placed in a watertight box, that protects the electronic stuff from water but allows evacuating the heat thanks to a large alloy wall. This electronic box is fixed on the bottom of the hull in the front. The inside frame of the hull includes drilled walls to help the convection thanks to a natural air flow. A thermal

sensor has been added to monitor the temperature inside the hull in summer time especially during very sunny days. The overheat ought to be avoided even during sunny days thanks to the shade of the solar pannels and the shade of the doghouse and also to the average temperature of the bottom of the hull (the water temperature rarely exceeds 30 degrees Celcius in Mediterranean Sea in summer).

3 Embedded Intelligence

The programs implemented on the control unit meet two main requirements, i.e. to remotely control Marius and to let Marius manage autonomously its behaviour in relation to its current mission. For this reason, the authors decided to split the embedded intelligence in two modes, i.e. the manual control mode and the automatic mode. All the programs are coded in LabVIEW language, which is a graphical language. It is especially interesting because the programs are hierarchically sorted and can easily interact with the hardware inputs/outputs. Many functions dedicated to signals like acquirement, measurement, treatement and analysis are already preprogrammed. The control unit Compact-RIO 9076 includes FPGA chips that give the opportunity to speed up the code execution and also to reduce to power consumption. In the first step, all the programs are implemented in scan mode, this means that inputs/outputs are scanned and the FPGA functions are not used. In this mode, the execution frequency reaches 500 kHz, this is very confortable for our application (the classical sample time for sensors and instrument does not exceed 20 Hz).

3.1 Manual Control Mode

In the manual mode, the pilot remotely controls Marius. It means to collect and to display all the data about Marius' states provided by the embedded sensors/instruments, to edit the path to follow as a GPS waypoint file and to directly drive the both actuators of the main sail and the rudder. The GUI that runs on the host laptop has been programmed in QT language. The main window gives the possibility to edit the path by clicking on the static or dynamic (if Internet is available) Google Map. All the waypoints are stored in a list that can be easily modified before uploading it to Marius. The map displays the desired and the actual paths. From the main tab, the pilot can check if all the instruments work properly. A big double-arrow button allows refreshing the data including the battery level. A permanent led shows the user if Marius is connected or not. There is a command window that is used to send requests to Marius. The sailbot does not transmit its data unless requested and connected. This helps to save energy by using the bluetooth

module only when necessary. A second tab is dedicated to the manual control of each actuator. The heading, the wind direction and the wind speed are continuously refreshed to help the pilot to easily control Marius (Fig. 7).

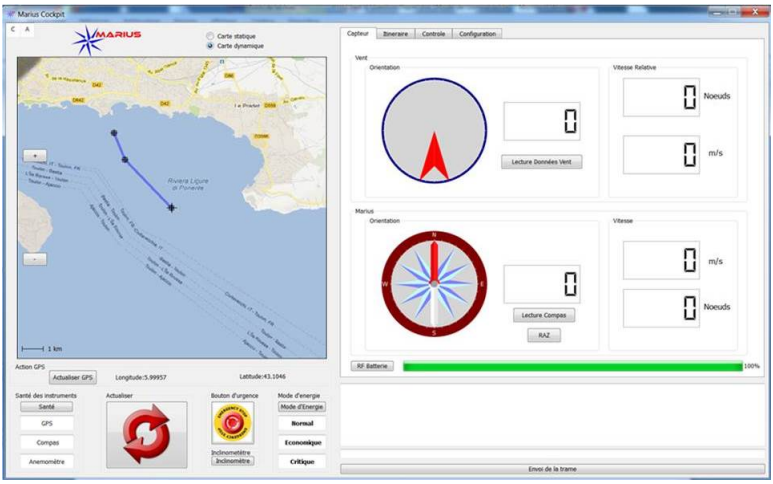


Fig. 2 Snapshot of Marius Cockpit - GUI for the remote control mode

The manual mode includes also an emergency button that launches an automatic function, which drives both actuators so that Marius turns until facing the wind. During this maneuver Marius goes upwind to heave to (i.e. keeping the rudder 10 ° on one side and the sail 10 ° on the other side). Of course the pilot can switch whenever he wants to the automatic mode. In this case, Marius will target the next waypoint it has not passed through.

3.2 Automatic Mode

The automatic mode was first designed to make Marius follow a path defined as a list of GPS waypoints. To do so, it was decided to get inspired from the algorithms that control Vaimos [3]. Despite the experience we gained from Avalon's challenge in last September in Toulon, we could not adopt the same control method because Marius' rig is not based on a balestron, thus it cannot jibe by turning its sail futher than the 120 ° range given by the circular rail and limited by the position of both lateral shrouds. The chosen algorithms take the wind conditions into account but do not consider potential obstacles. Indeed no device of obstacles detection was implemented aboard Marius yet. The implemented automatic control mode has to follow a path where each waypoint owns a value of dangerosity. This value illustrates the proximity of

dangerous areas or shore. Therefore Marius is allowed to pass more or less close to the considered waypoints.

Marius manages its energy through three levels of energy. We distinguish two main modes (normal and economy modes) where the sampling frequency switches from 5 Hz to 0.1 Hz for the instruments and the control of actuators. A third mode is a critical mode that prevents the battery from the deep discharge by turning Marius in idle mode (Marius gets to heave to). Afterwards it lets itself drift until the battery be charged again above 50%. The energy management has to be experimented during a long mission to validate its relevance, but this has not been done so far.

Fig. 3 Marius in test

4 Conclusion and Future Works

Marius project is a very young project (6 months) that led to an operational sailbot designed to be robust against harsh environments and climatic conditions. For a beginning, its control modes are basic but are supposed to be improved after several missions in forthcoming months (Fig. 3). Moreover, as it was mentionned before, some important features related to security and navigation skills are still missing on Marius. This is the reason why the obstacles detection and obstacles avoidance are included in future works to do. A long range transmitter for GPS position will also equip Marius in a short

future. Finally, some other evolutions will certainly stem from the requirements of various oceanographic missions.

References

1. Romero-Ramirez, M.A.: PhD thesis: Contribution à la commande de voiliers robotisés., Universite Pierre et Marie CURIE, Paris, France, p. 206 (January, 23, 2012)
2. Erckens, H., Busser, G.-A., Pradalier, C., Siegwart, R.Y.: Navigation Strategy and Trajectory Following Controller for an Autonomous Sailing Vessel 17(1), 45–54 (2010)
3. Jaulin, L., Le Bars, F.: A simple controller for line following of sailboats. In: IRSC 2012, Cardiff, UK (2012)
4. Naveau, M., Anthierens, C., Pauly, E., Courmontagne, P.: MARIUS: Design of a sailbot for oceanographic missions. In: Oceans 2013, MTS/IEEE, San Diego, September 23-26 (submitted, 2013)
5. Guo, Y., Romero Ramirez, M.A., Leng, S.H., Plumet, F., Benosman, R., et Gas, B.: Reactive Path Planning for Autonomous Sailboat using an Omni-Directional Camera for Obstacle Detection. In: IEEE International Conference on Mechatronics (ICM), Istanbul, Turkey, pp. 445–450 (2011)
6. L'Ifremer en videos, `http://wwz.ifremer.fr/webtv/Liste/Nouveautes/VAIMOS` (May 5th, 2013)
7. Erwan, I.: le MiniJI de l'Ecole Navale, `https://sites.google.com/site/minijiecolenavale/project-definition` (May 5th, 2013)
8. Grundmann, S.: Avalon Toulon 2 HD, `http://youtu.be/H7PBQG_crzY` (May 5th, 2013)

Development of a Low-Budget Robotic Sailboat

Christoph Schröder and Lars Hertel

Abstract. Building a boat is the first of many steps in robotic sailing. In this paper, we describe the development of a low-budget robotic sailboat based on the MaxiMOOP. This includes a conceptual design for entering the Microtransat Challenge, a approach for constructing the hull out of styrofoam, the development of a balanced swing rig using low-priced materials only and the use of a reduced set of sensors for autonomic control in order to decrease energy consumption. Moreover, a short field test showing the overall performance of our prototype boat and its seaworthiness, regarding boat speed, course stability and energy consumption is presented.

1 Introduction

Robotic sailing is a challenging task, in both building and controlling the boat. Therefore, it brings together many different disciplines, such as naval architecture, electrical engineering and computer science. In the past, our focus was put on the latter, i.e. the development of efficient and stable algorithms in order to control small robotic sailboats [7]. In this paper however, we will describe the design and construction of a low-budget prototype robotic sailboat. Its main advantages are the affordable price, the flexible area of operation due to its low weight and rather small size, as well as the ability to exchange hull and sail because of its modular design. Our boat therefore presents a low priced alternative to more advanced robotic sailboats such as the French VAIMOS [3], the Austrian Roboat [8] and the Portuguese FASt [2]. It costs around €350 (parts only, 2013 prices), is approximately 1,2 m long

Christoph Schröder · Lars Hertel
Institute for Robotics and Cognitive Systems, University of Luebeck,
Ratzeburger Allee 160, D-23562 Luebeck, Germany
e-mail: {cschroeder,hertel}@informatik.uni-luebeck.de

F. Le Bars and L. Jaulin (eds.), *Robotic Sailing 2013*,
DOI: 10.1007/978-3-319-02276-5_2,

and weighs not more than 13 kg. Its original goal was to compete in the Microtransat Challenge [2]. The development process, however, is still ongoing and we hope to meet that goal in the near future.

In the following paper, we will present the construction of the hull, sail and electrical components of the boat and point out the main ideas and design decisions. While building the boat, our emphasis was on designing a low-budget and at the same time robust prototype that would be able to perform well under difficult weather conditions. The latter needs future evaluations.

2 Methods and Material

In this section, we will first define a conceptual design for the boat including its components and then present the construction process of our prototype. A cost distribution of the used material is summarized in Table 1.

2.1 Concept

For the Microtransat Challenge we assume 80 to 140 days of autonomous operation. The exact time depends on the route [6] and the speed of the final boat. In Table 1 we give an overview of the planned power consumption of each electrical component together with an estimated time each component operates per hour. As a result our boat consumes approximately 20 W per day. Accordingly, a solar panel that produces about 25 W per day is needed.

Table 1 Cost distribution of the used material for our robotic sailboat

Name	Description	Price in Euro
Hull	MaxiMOOP [10] made of styrofoam, epoxy resin and glass fibre cloth	50
Computers	Raspberry Pi and Arduino Uno	60
Sail	Swing Rig made of pond liner and aluminum tubes	23
Sail Actuator	Grill Motor	19
Rudder	Aluminum sheets and aluminum tubes	8
Rudder Actuator	Grill Motor	19
Motor Driver	Pololu Dual VNH5019 Motor Driver	50
Motor Position (2)	AMS AS5040	30
Wind Sensor	AMS AS5040	15
Battery	Lead battery (12 V, 2.5 Ah)	15
GPS	Ublox LEA4-T GPS	80
Total		369

Table 2 Planned power consumption of individual components. The second half is based on the assumed seconds the component is powered per hour (ontime s/h).

Component	V	mAh	mW/h	ontime s/h	mW/h	mW/day
μProcessor	3.3	5	16.5	3600	16.5	396
Wind sensor	3.3	2	6.6	3600	6.6	158.4
Compass	3.3	2	6.6	3600	6.6	158.4
GPS	5	39	195	3600	195	4680
Sail	12	2000	24000	30	200	4800
Rudder	12	2000	24000	60	400	9600
Spot Connect	3.3	53	174.9	240	11.7	279.8
Total					836	20072

Due to the lack of solar energy during the night we use a buffer that is charged during the day as shown in Figure 1 and has a minimal capacity of 14.3 Wh for continuous operation.

During the mission, the efficiency of a solar cell decreases because of dirt and salt on its surface. Assuming a decrease of 20 % after the first 100 days on the water as shown in Figure 1, the buffer is not charged enough to power the boat alone. Therefore, we include lithium batteries that are not charged by the solar panel but deliver the missing power when the buffer is empty. With a relatively small backup of 68.4 Wh the time of operation easily exceeds 200 days (Figure 1) as the buffer is still charged partly during the day.

2.2 Hull and Rig

The hull of our boat is based on the MaxiMOOP, a modified version of the MOOP [10] (courtesy of Paul Miller, Mark Neal and Colin Sauzé). It possesses a narrower and deeper keel which improves hull speed and upwind performance of the boat. In order to build the hull, we first cut a negative version out of large styrofoam blocks. To do this as accurately as possible, we programmed a robotic arm to melt down the styrofoam surface using a heating wire out of tungsten. Afterwards, we applied three layers of glass fibre cloth for our boat to be waterproof and robust. To build a keel we first removed the styrofoam inside the boat, melted 5 kg of lead and filled it inside the bottom of the boat. Furthermore, to stabilize our boat, we installed five frames made of wood. At last, we covered our open hull with a fitting deck, made of wood as well, coated with glass fibre cloth. For debugging purposes, hull and deck are fixed with duct tape as a start. This allows to easily open the deck and quickly fix occurring errors.

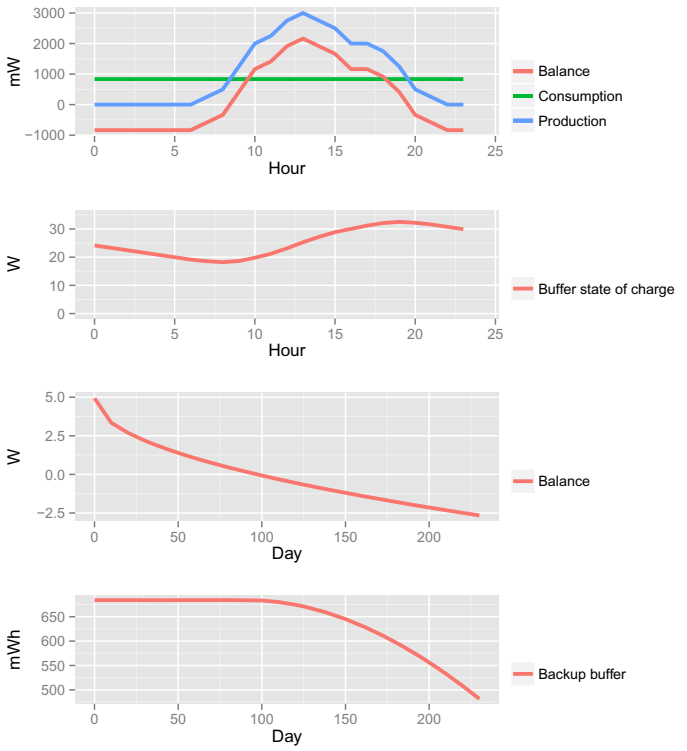
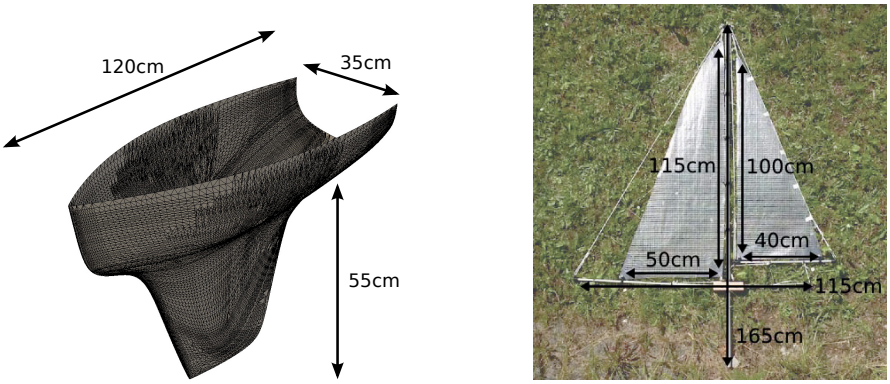


Fig. 1 The first figure describes the power balance during the day. It considers a total power production of $25000mW$ per day provided by the solar panels and a constant drain of $836.36mW$. The power consumption is based on Table 1. The second plot shows the buffer state over the day while the solar panel operates at 100% efficiency. In the third plot, the sum of production and consumption during the mission is shown, assuming 80% efficiency of the solar panels after 100 days on mission. The bottom plot shows the backup buffer charge which decreases after the solar cells can not charge the main buffer anymore.

Every opening in the final boat is a possible way for water to come in. Therefore, we reduced the number of holes needed by using a balanced swing rig. Thus, we sacrificed speed and agility for a robust and simple design which is also used by a number of other boats such as IBoat [4] and Avalon [5]. Again this simplifies the mechanical construction and algorithmic maneuvers since we gain agility through the rudder. Design and construction of sail and rudder are based on rough calculations. Their simple design and construction allows a fast iteration of size, form and trim.



(a) (b)

Fig. 2 The design of our prototype sailboat. Subfigure a) shows the hull of the MaxiMOOP (courtesy of Paul Miller, Mark Neal and Colin Sauzé) and its dimensions. In Subfigure b) the dimensions of our balanced swing rig are illustrated.

2.3 Sensors and Motors

Besides hull and rig, sensors and motors are important for robotic sailboats. They are crucial not only for optimal maneuvers but also need to be low-priced and robust to minimize money and time spent on maintenance. Consequently, we kept the number of sensors low and abstained from e.g. magnetometer, gyroscope and accelerometer. Our algorithms only depend on position and speed from GPS and the relative wind direction. In the current prototype we use ublox LEA4-T GPS chipset (ublox, Switzerland) that allows fast testing and accurate data for narrow maneuvers in small bays. The wind direction sensor is based on the AS5040 chip (Austria MicroSystems, Austria) [11] which is low-priced and reliable. It is placed on top of the rotating mast and the wind direction is calculated considering the current orientation of the rig.

On the mechanical side, rudder and sail are not rotated by industrial servo motors but grill motors. These DC motors come with a gear to achieve a high torque and low speed. The high torque saves energy by holding the position of the sail once it is in the right angle for a point of sail. Furthermore, we use the same rotation sensors as for the wind sensor to keep track of the gears rotation which reduces implementation complexity on the software side.

2.4 Electronics and Software

For quick development and easy live debugging we chose the Raspberry Pi (Pi) as our main control unit (see Figure 3 for a system overview). With a

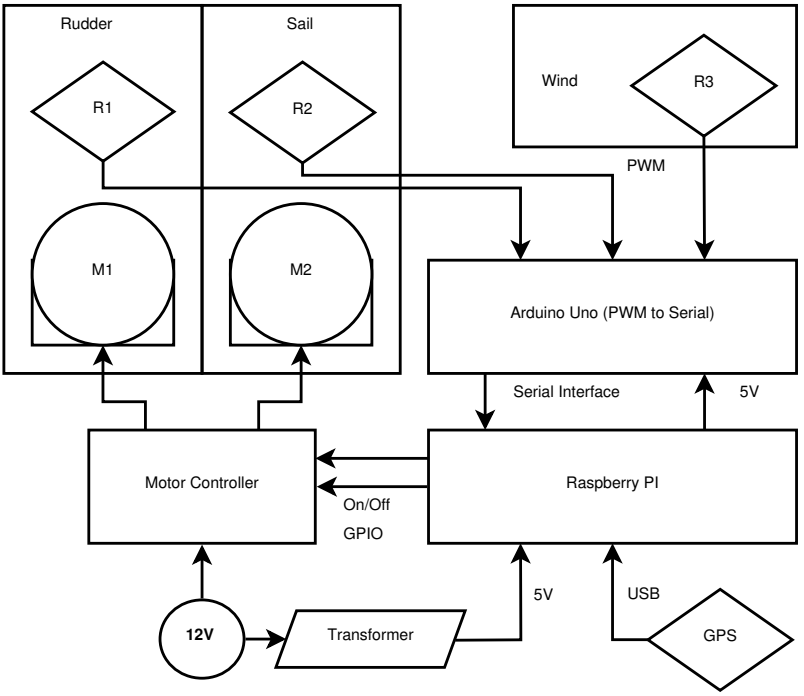


Fig. 3 Schematics of the boats control circuit. M1 and M2 denote Motors, R1-3 are AS5040 chips. The Raspberry Pi is the main control unit reading the sensor data converted by the Arduino and controlling the motors of sail and rudder by its GPIO pins. GPS is connected directly via USB.

700 Mhz ARM processor it runs a full Debian Linux environment. Communication for programming and debugging purpose is established over Wi-Fi with a USB dongle (Netgear N300). Although the Pi is good for fast iterations in software development there is no easy way to read the PWM signals from the wind and motor rotation sensors. To overcome this problem the Arduino reads the PWM signals and transmits the current sensor values via a serial connection to the Pi.

The GPS module is powered by and communicates to the Pi via USB, while the Pi itself is powered by a lead battery located in the keel of the boat. With a switching regulator the 12 V from the battery are converted to the 5 V input for the Pi. The motors however, are driven at their designed input voltage of 12 V. Consequently, we lose about 15 % during the conversion for all non motor electronics at about 240mAh. Rudder and sail are controlled by a separate motor driver. Although the driver allows linear regulation of the motor's speed, we currently only use the GPIO pins of the Pi to control both, the direction and the off states of each motor.

On the Raspberry Pi we use Java 1.8 for ARM which supports hardware floating point operations and JIT compilation. Further we can use existing libraries of our past projects and benefit from Java's ecosystem. Most notable are the Pi4J[1] project for easy access to the serial port and GIO pins of the Pi and GPSd4Java[2] for GPS module handling. Due to the computational power of the Pi's processor and sufficient amount of RAM, we are able to run the autonomous behavior not only on onshore and send the commands via Wi-Fi, but also we are able to do all necessary calculations on the boat itself.

3 Results

In the following, we give an impression of the general performance of our boat. Therefore, we divide the results of our field tests into three main categories, namely boat speed, course stability and energy consumption. Images of our prototype during the field tests are shown in Figure 4.

(a) (b) (c)

Fig. 4 The images show our prototype during the field tests, sailing on different points of sail

3.1 Boat Speed

In order to obtain data regarding the boat speed over ground, the boat was set up to sail a triangular course. The autonomous run lasted for 15 min

[1] `http://pi4j.com` The Pi4J project provides a bridge between the native libraries and Java for full access to the Raspberry Pi.

[2] `https://github.com/taimos/GPSd4Java` GPSd4Java is a library to use data from the GPSd daemon in Java.

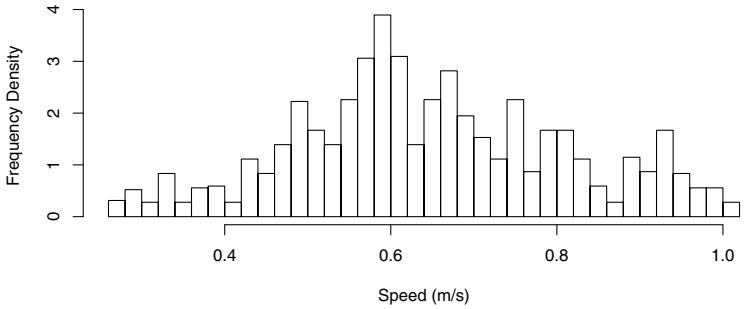
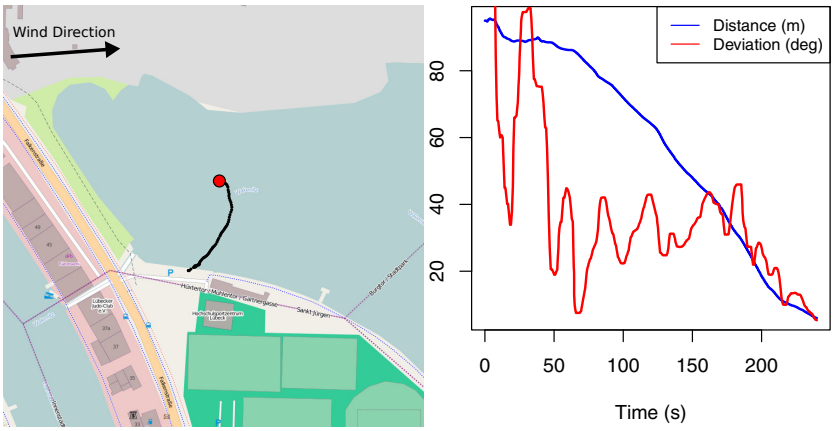


Fig. 5 A histogram showing the typical distribution of the boat speed over ground in m/s. The plot indicates that our boat reaches an average speed of 0.65 m/s and a maximum speed of 1 m/s in easterly winds of approximately 7 kn to 9 kn.

during moderate weather conditions in easterly winds ranging from 7 kn to 9 kn. The collected data was then filtered and outliers were removed. The histogram representing the speed distribution in meters per second is shown in Figure 5. It points out an average boat speed of 0.65 m/s. Furthermore, the histogram indicates that our prototype is capable of sailing at sustained speeds of more than 1 m/s, or 2 kn. Compared to the performance of the robotic racing Micro Magic (rrMM, [11]) under similar weather conditions, the results indicate a slight increase in boat speed [7].

3.2 Course Stability

Besides the boat speed, another important criterion in autonomous sailing is the stability or precision of the sailed course. In order to analyze the course stability, we measured the course deviation as well as the target distance while approaching a waypoint autonomously. The test was performed in easterly winds of 8 kn with modest gusts of wind, complicating the task additionally. Initially, the distance to the target was 100 m and the boat was facing away from the target. The test lasted for 4 min. The results are shown in Figure 6. The logged GPS data in Subfigure 6a illustrates that our boat was able to reach the waypoint autonomously, not perfectly though. More detailed information is given in Subfigure 6b. The distance to the target is constantly decreasing, indicating that the boat sailed towards the waypoint. As for the course deviation however, an unsteadiness is noticeable. Clearly, our boat did not reach the waypoint on an optimal path, but given that our boat is just a prototype and simple controllers for sail and rudder were used, the results illustrate the autonomy and maneuverability of our boat.



(a) (b)

Fig. 6 The figure illustrates the autonomy and maneuverability of our prototype. Subfigure (a) displays a map of the test area. The waypoint is plotted in red and the GPS data of the boat is plotted in black. Subfigure (b) points out the decreasing distance and course deviation to the waypoint. Map data OpenStreetMap contributors, CC-BY-SA.

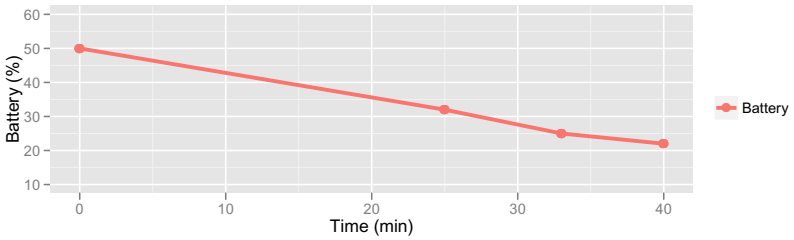


Fig. 7 Energy consumption of our boat. The battery was discharged by from 50% state of charge to 22%, indicating an average power consumption of 12.6 W/h. Note that we had to measure the voltage manually using a multimeter.

3.3 Energy

Especially important for long-term missions, such as the Microtransat, is a low energy consumption of a robotic sailboat. To test the consumption of our boat, we measured the change in voltage of the lead battery. Since no automatic measurement of the voltage is available for our boat, we had to manually measure the closed circuit voltage using a multimeter. During an autonomous run of 40 min, we measured the voltage of the battery four times while being under a discharge of 250 mAh. The connected data points are shown in Figure 7. Based on the capacity of the battery of 2.5 Ah we assume

a discharge rate of C/10 [9]. During the test, the charge condition decreased from 50 % to 22 %. This corresponds to a discharge of 8.4 W and gives an average power consumption of 12.6 W/h.

4 Discussion

Being only a prototype, our boat is far from entering the Microtransat Challenge. However, some design decisions have proven to be serviceable. Using a readily available hull instead of creating a new one saved a lot of time and the MaxiMOOP has already been proved and tested. Furthermore, cutting the hull out of styrofoam using a robotic arm saved a lot of money. However, thoroughly removing the styrofoam after having applied the coats of glass fibre cloth turned out to be unnecessary.

The energy consumption of the prototype is about ten times more (Section 3.3) than planned in the concept (Section 2.1). This has two main reasons. First, the software used during the test is configured to constantly adjust rudder and sail. During a long term mission these adjustments are planned once every minute for the rudder and once every two minutes for the sail. Second, the hardware of the prototype uses far more energy than the embedded hardware of the final boat. The Raspberry Pi alone drains about ten times more power than the planned microprocessor. Additionally, the Wi-Fi connection is not needed on the final boat. Despite consuming more power than planned for the final boat, using the Raspberry Pi and a Wi-Fi connection allow us to iterate fast. Building and deploying a new version of the control software is done within less than a minute without interrupting the boats operation for more than about 10 seconds. The range of the used Wi-Fi adapter of about 100 m limits the possibilities of further tests and will be replaced by a UMTS connection in the next iteration of the prototype.

As for the sail, a balanced swing rig performs well on water and uses only a single motor. Therefore, the energy consumption is reduced, being one of our main goals designing the boat. In addition, the pond liner turned out to be an excellent low-priced alternative to sailcloth. One weak point in the design are the predetermined breaking points in the link of mast and boom. So far a threaded rod is used for the connection, requiring a hole in the mast. This should be replaced by either adhesive or friction in the future.

Another important design decision is whether to position the wind sensor on top of the mast or on the deck. Despite having a rotating mast because of the swing rig, we placed the wind sensor on top of it. The advantage of measuring the wind direction more precisely was deciding. So far we have not noticed any disadvantages yet, knowing the exact mast rotation and being able to factor it into our calculations.

As for the rudder, our field tests revealed several lacks in its design. The rudder is overpowered and unbalanced and therefore slowing the boat down. Even though the rudder serves its purpose, it will be redesigned in the future.

Another interesting alternative is the use of a magnetic linkage between the rudder actuator and the rudder itself [10]. Moreover, this would avoid cutting a hole in the hull, being a possible error source of letting water inside the boat. In the latter case, a bilge pump for successor boats is planned.

Our results show the seaworthiness and maneuverability of our prototype. Compared to the performance of the rrMM, our results indicate an increase in both boat speed and course stability, which is not surprising given the larger size of our prototype. For a more detailed analysis of the overall performance of our boat, more data during extended field tests has to be collected.

5 Conclusion

Building a boat is the first step of many in robotic sailing. We have presented the development of a low-budget robotic sailboat. For costs around €350 we have built a fully functional prototype using either existing or low-priced materials only. The hull is based on the MaxiMOOP and was cut out of styrofoam using a robotic arm. The sail is a balanced swing rig and made of pond liner. As for the electronics, a Raspberry Pi and an Arduino Uno are used so far for the prototype, but will be replaced by energy-saving microprocessors in the future.

The results show the seaworthiness of our prototype. The boat is capable of sailing at sustained speeds of 2 kn in moderate weather conditions, probably faster with increasing wind speeds. Moreover, it is able to autonomously reach a target waypoint using GPS and wind direction sensors only. This reduced set of sensors decreases the energy consumption and is therefore beneficial for long-term missions, such as the Microtransat Challenge.

However, our prototype has to be improved in the future, using more energy-saving components, collecting and analyzing more data during extended field tests and adding a solar panel to the deck. Thus, we would be able to charge the battery during long-term missions and finally enter the Microtransat Challenge.

Acknowledgements. The authors would like to thank Alexander Schlaefer, Ralf Bruder, Felix Stahl, Sven Kind, Ulf Wohlers and Sebastian Klawiter for their contributions towards designing, constructing and testing the prototype.

References

1. http://www.microtransat.org
2. Alves, J., Cruz, N.: Fast - an autonomous sailing platform for oceanographic missions. In: OCEANS 2008 (2008)

3. Bars, F., Jaulin, L.: An experimental validation of a robust controller with the vaimos autonomous sailboat. In: Proceedings of the 5th International Robotic Sailing Conference (2012)
4. Briere, Y.: Iboat: An autonomous robot for long-term offshore operation. In: The 14th IEEE Mediterranean Electrotechnical Conference, MELECON 2008, pp. 323–329 (2008), doi:10.1109/MELCON.2008.4618455
5. Erckens, H., Buesser, G., Pradalier, C., Siegwart, R.: Avalon. IEEE Robotics and Automation Magazine 17(1), 45–54 (2010)
6. Gibbson-Neff, P., Miller, P.: Route planning for a micro-transat voyage. In: Proceedings of the 4th International Robotic Sailing Conference (2012)
7. Hertel, L., Schlaefer, A.: Data mining for optimal sail and rudder control of small robotic sailboats. In: Proceedings of the 5th International Robotic Sailing Conference (2012)
8. Klinck, K., Stelzer, R., Jafarmadar, K., Mellinger, D.: Aas endurance: An autonomous acoustic sailboat for marine mammal research. In: 2nd International Robotic Sailing Conference, Matosinhos, Portugal (2009)
9. Perez, R.: Lead-acid battery state of charge vs. voltage. Home Power, 66–70 (September 1993), http://www.scubaengineer.com/documents/lead_acid_battery_charging_graphs.pdf
10. Sauze, C., Neal, M.: Moop: A miniature sailing robot platform. In: Proceedings of the 4th International Robotic Sailing Conference (2011)
11. Schlaefer, A., Bruder, R., Heinig, M., Beckmann, D.: A new class for robotic sailing: The robotic racing micro magic. In: 4th International Robotic Sailing Conference (2011)

VAIMOS: Realization of an Autonomous Robotic Sailboat

Olivier Ménage, Aymeric Bethencourt,
Patrick Rousseaux, and Sébastien Prigent

Abstract. This paper demonstrates the relevance of using autonomous sailboats for the realization of long missions (several weeks) devoted to collecting measurements and observation of the marine environment at low cost. Ultimately, such a system should be used in place or in addition to current conventional systems such as drifting buoys or oceanografic ships. The paper introduces the electrical, mechanical and algorithmic realization of the sailboat and then demonstrates its robustness through several missions.

1 Introduction

The Ifremer *(Institut Français de Recherche pour l'Exploitation de la Mer)* Department of Technological Development has recently developed a small coastal vessel, electrically powered and remotely controlled. This boat, realized on the basis of a kayak of about four meters long, accepts several sensors mounted on demand, either fixed to the hull or on two small winches. This allows the realization of mini-profiles over a distance of thirty meters. The ASV (Autonomous Surface Vehicle) is not intended to go offshore. Its all-electric propulsion does not provide with more than a few hours of autonomy. Furthermore, it possesses no embarked intelligence: all the commands are deported on a computer where an operator controls his evolution.

Olivier Ménage · Patrick Rousseaux · Sébastien Prigent
Ifremer, Pointe du Diable, 29280 Plouzane, France
e-mail: {Olivier.Menage,Patrick.Rousseaux}@ifremer.fr,
Sebastien.Prigent@ifremer.fr

Aymeric Bethencourt
IHSEV, Lab-STICC, OSM, Pôle STIC, ENSTA Bretagne, 2 Rue F. Verny,
29806 Brest, France
e-mail: Aymeric.Bethencourt@ensta-bretagne.fr

F. Le Bars and L. Jaulin (eds.), *Robotic Sailing 2013*,
DOI: 10.1007/978-3-319-02276-5_3,

For the past three years, ENSTA Bretagne has developed a sailboat capable of crossing the Atlantic Ocean in a total autonomous way [2]. The first prototype is about one meter long and has already performed several convincing experiments. During last tests, it showed strong ability for navigation despite some problems related to mechanical design.

The objective is to build an autonomous prototype equipped with sensors of temperature and salinity and capable of measurements in two depths [2] [20]. This demonstrating sailboat will have to be able to sail autonomously, but also be piloted remotely. Objectives will include :

- Sail autonomously according to pre-given instructions (steering trajectory and / or goals and / or areas to avoid).
- Sail remotely controlled by an operator on land or on a ship via direct radio link.
- Endure difficult sea conditions.
- Provide information about its state (position, heading, speed, power, etc ...), regardless of its position.
- Receive orders for control and management of on-board equipment.
- Have sufficient energy independence for several days of operation, ultimately 3 to 4 weeks.
- Measure the temperature and salinity parameters on two depths.
- Store the acquired data: scientific data and overhead data.
- Transfer the scientific data at the lowest possible cost.
- Have a system of independent backup location. (The electronic for the GPS is completely separated from the rest of the ship, so even is everything else crash, the location of the sailboat would still be recorded.)

This paper is organized as follows : Section 1, 2 and 3 respectfully present the electrical, mechanical and I.T. realization of the *VAIMOS*. Section 4 presents the tests and missions realized and section 5 conclude the paper.

2 Electrical Realization

The purpose of this study is to realize a prototype robotic sailboat for two main reasons:

- Show viability of such a project, and provide results to support funding requests for the project.
- Serve as an experimental base for autonomous navigation algorithms developed by Ifremer or ENSTA Bretagne.

Given these objectives, it is clear that the "ultra low power" model, common to many embedded systems aspect, is not the top priority. However, the prototype must have some modularity to accommodate any sensor needed both to validate the algorithms and the scientific measurements. In addition, to validate the various tests, and debug the behavior of autonomous boat, it seems necessary to have access to all parameters of the boat from

shore or nearby ships. That is why we chose to build the digital architecture of the sailboat around an Ethernet board. The adoption of this technology will completely separate the embedded computer from the offshore computer. Both will seek information from the sensors or actuators to send their orders directly on the network. In addition, this architecture also provides the advantage of being able to implement any card on the boat easily, as long as it works with TCP / IP.

For this project, a new architecture has been implemented around existing radio technologies such as WiFi and Iridium satellite communications. This architecture can efficiently exchange information in real time (instrumentation, sensors, ...) and manage (control) mobile or stationary equipment without wired connections. All information, proprietary or not, can be multiplexed and encapsulated by a radio modem. This information, once submitted, are dispatched to different devices in real time. The choice of wireless radio technology in this architecture is justified partly by its reliability, robustness, and by its availability in many industrial applications, showing great maturity for this technology.

For this project, the wireless connection can convey information at high speed and over a distance of several kilometers. This information can be of several types: environmental data, measurements, navigation data, audio and video streams, proprietary data flow, etc...

3 Mechanical Realization

3.1 Building the Sailboat

Hull. For the basis of the hull, we chose to use a *mini-J*. This hull offers several advantages:

- It has a reasonable size (3.65 m) and a comfortable embeddable load (90 kg).
- It is self-righting and unsinkable.
- It is made locally and is available within two months.
- It can be purchased without the deck and original rigging.

For cons, there is no digital file of the hull, and changing the rigging requires full modeling of the hull to determine the new position of the sprit. It was therefore necessary to make an accurate manual hull model using specialized software in ship design.

Deck. The deck was made from the rib recorded directly on the hull, it consists of five layers of glass fabrics of 500 gr/m^2 associated with epoxy resin. To allow easy access to various electrical and mechanical components, it was drilled and reinforced to receive two panels of waterproof deck.

Rudder. For the sake of simplicity and speed, we decided to keep the rudder of origin. It has a surface area of 0.2 m^2 and a 8% offset. Doubling the

size of the rudder and the offset to 25% increased control of the ship without raising efforts. We chose to use a servo-motor to control the rudder. The winch comes with a roll of 32 mm diameter, which gives a tensile strength of 30 kg/cm divided by 3.2 cm which equals 9.375 kg, almost 10 kg, a distance of $10 * (1.6 * 2\pi) = 100$ cm which is a displacement of 1 meter. To be effective at tacking, the rudder must turn 45° port to starboard. We adjusted the length of the control arm to these values to give it the maximum torque. We obtained a lever arm of 38 cm. Moreover, to achieve the maximum angle values, the end of the arm moves only 22cm on each side, which allowed us to fit a hauling system on the cable channel. The pulley system allows us to carry a tension of about 20 kg on the control arm, providing a couple on the rudder of 20 kg x 38 cm = 720 kg.cm. We chose a *spectra* cable of a 1mm diameter with a 60 kg breaking point. This security coefficient of 3 gave us a margin that allowed us to limit the flow. We then added a support system on the cable tray to keep it always on.

Sail motorization. The sprit, with its offset design, can be operated without significant efforts. However winch trimming the sail should be able to absorb the constant efforts of the wind on the sail, and the jerks when tacking. No commercial winch meet the requirements, so we had to develop and to manufacture it ourselves. In order to monitor the movement of the sprit, we opted for a stepper motor which can be controlled via a dedicated card, in relative or absolute position.

3.2 Integration of the Scientific Measurement System

The system has to be able to perform measurements of temperature and salinity in two depths, one at the maximum depth possible on the shell (ideally two meters), and the other as close as possible to the surface (ideally within the first ten centimeters). The first solution was two small loggers (STPS type) and set them just under the hull outings. This solution had the advantage of being very quick and easy to make. However, these devices might be damaged when maneuvering or in case of grounding. In addition, the two protuberances from the hull would have hung algae and various objects. Finally, this solution prevented any real-time communication with the sensors, prevented real-time recovery of sensor data.

After some research, we decided to use the *NKE multiparameter probe MP*. The probe can be powered and interrogated remotely via an RS232 connection. It is equipped with temperature, salinity, chlorophyll, turbidity and oxygen sensors beyond our original specifications. The probe is placed onboard and its sensors in a measurement chamber. This chamber is supplied with sea water by two circulation pumps, mounted head to tail, and successively removing the water through two small holes in the shell open, close to the one surface and the other to the base of the keel. This solution,

Fig. 1 The hull with its sprit

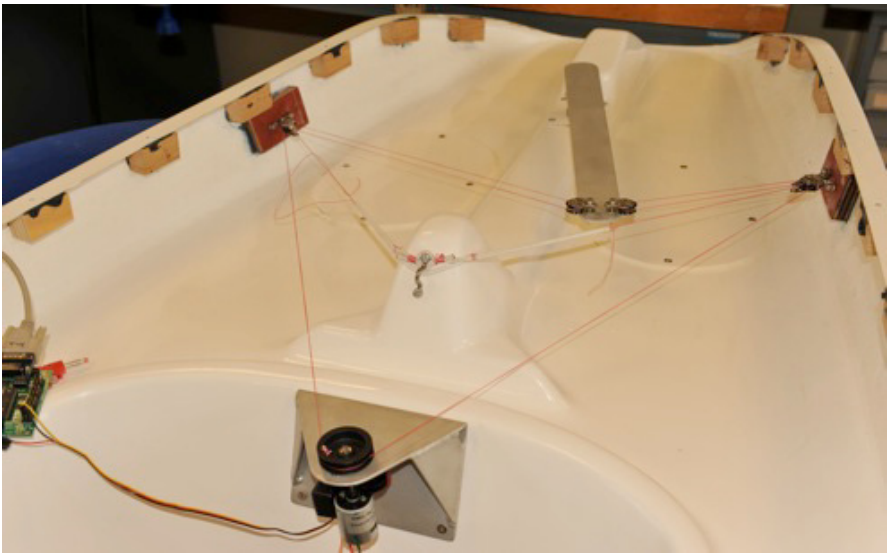


Fig. 2 Saffron control system composed of a halyard block, a tension spring, saffron arms and a winch

Fig. 3 Sensor apertures

although more complicated and time-consuming to develop, overcomes all the shortcomings of the previous.

4 I.T. Realization

4.1 Human-Machine Interface

Before starting to implement algorithms on the boat, it seemed rather necessary to try to control it remotely. Accesses to the various actuators and sensors of the boat is made through a network, and by opening a telnet port on the associated IP / serial converter. Ifremer, for *Mobesens* project had already developed a HMI using VB.net to control a kayak model with a remote, and retrieve all the information from the sensors graphically on a computer. To reduce development time, we sought and obtained the support of the team that had developed the previous HMI. We drafted the specifications of the *VAIMOS* HMI, and provided them all the frames of communication with the sensors and actuators of the boat.

Fig. 4 *VAIMOS* project and realization

4.2 Implementing Algorithms

To make the sailboat completely autonomous, it was necessary to provide it with robust navigation algorithms [6] [4] [14] which were implemented in the embedded Linux system. Two algorithms were needed to navigate the boat : rudder control and sail control [9] [18].

5 Testing

The first test of the boat was held at St Anne Portzic, June 24, 2011. Weather conditions were excellent, the wind was blowing west and stable at about 10 knots. During the first two hours of evolution, the boat showed a great aptitude to navigate which validated the choices made on the structural design. We were also able to validate the entire chain of command of the boat, from the remote control of the boat to the sensors and operating winches on board.

Thereafter, the sprit began acting strangely and it quickly became impossible to work on the sail trim. After unsuccessfully trying to restart the applications, we decided to stop the test and return to the Ifremer to

Fig. 5 First test of *VAIMOS* and OPTIMOUSSE

diagnose the problem. Back in the lab, it appeared that the cable used to control the sprit was stuck in the rear deck panel, which had the effect of dragging the stepper motor, which has lost its index position .

Then we performed various tests that helped to highlight two weak points:

• The tacking with a shocked sail, the cable flattens on the deck and can therefore blocks the rear deck panel.

• The index of the stepper motor is calculated by the control board. When the engine skid, this index is not related to the actual position of the sprit anymore.

The following solutions were implemented.

Changing the rear panel. The most effective and quick fix was to replace the rear panel adding a chamfer to the periphery so the cable would ride on it without any possibility to get hooked.

Establishing a procedure for setting the origin of the stepper motor. The most obvious way to solve this problem would be to add an encoder to the stepper motor that would have provided us with the actual position of the motor. However, given the price and availability of a good encoder, this solution was not possible. It would also have been possible to use a reed relay with a magnet set on the control cable of the sprit. That would have provided us with information on the return to zero but all the cables being outside, it would have exposed some electronic component to salt air.

Following these reflections, we opted for a software solution fairly simple and inexpensive. The solution was to lower the maximum torque on the

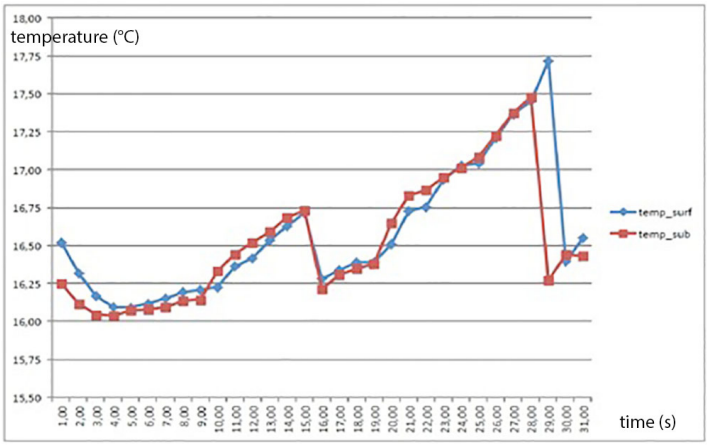


Fig. 6 Temperature measurements in two depths

stepper motor by giving it the order to border the sail for a time greater than that necessary to cover the total distance. After 60 seconds, the engine is in lined position and begin slipping. Therefore, we re-assign the origin at this point. This simple procedure will in future be carried out at regular intervals or in case of doubt about the consistence of the operations of the stepper motor.

During second test on July 1, 2011, we tried many times to put the boat in failing position, from which it recovered every time. Changes to the hatch made it impossible for the cables to jam. We also made several readjustments of the origin of the stepper motor with success. The boat was therefore fully functional.

First results of the probe. During the two tests, we launched the automatic acquisition of the NKE probe with the following cycles:

- 1 minute for ground pumping.
- Double measurement of all sensors.
- 1 minute for surface pumping.
- Double measurement of all sensors
- Wait 1 minute then start new cycle.

Although the test area was not suitable for measurements made by the boat, the system worked perfectly. Acquired data are displayed Fig.6.

First long term mission. Fig.7 gives a track in the Brest harbor by the sailboat. The desired trajectory is in red and is made by yellow waypoints and the effective trajectory is in green. Conditions were : South-West wind on the left, South-West wind of around 15 kn, 27km (17nm) in less than 5h for the right. For the parameters of the controller, we checked that the resulting controller guarantees the stability, provided that *VAIMOS* with its heading controller satisfy. The robot was always at a distance less than 30 meters to

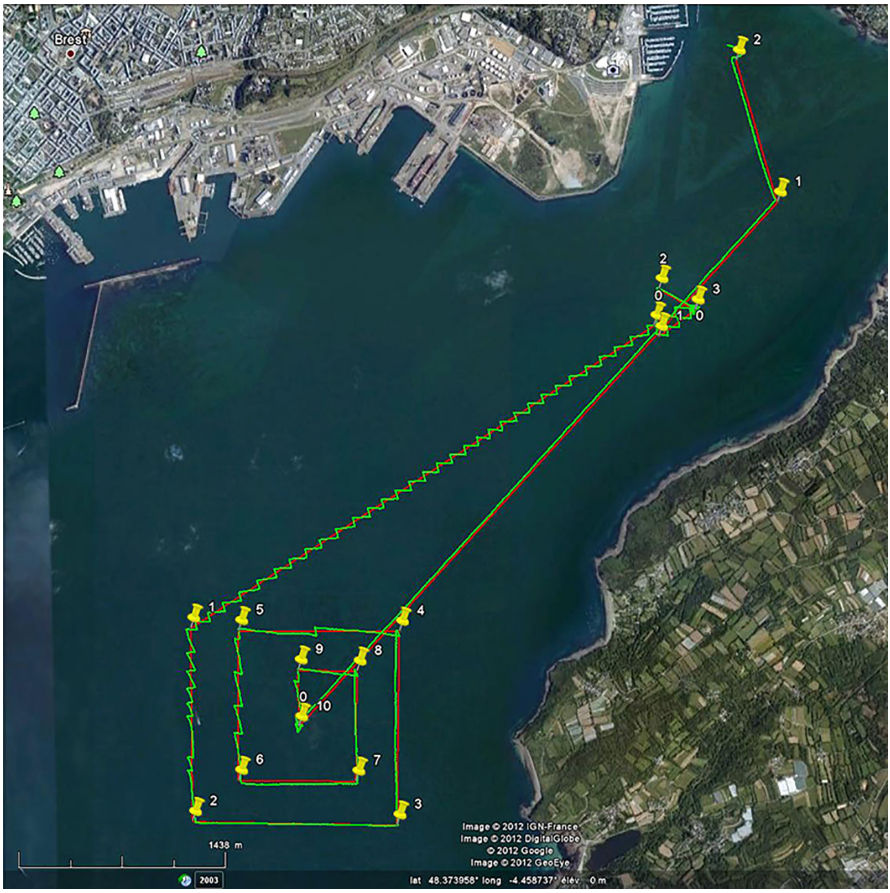


Fig. 7 GPS trajectory of the first long term mission

its line. Inside the square, the robot had to move upwind. It was in a close hauled mode and alternated starboard tacks with port tacks.

The second long term mission was 100km long between Brest and Douarnenez. The sailboat needed to be deviated twice: First because of a submarine coming back to Brest naval base, then because of a static boat in the sailboat trajectory. During these perturbations, the autonomous program was not changed nor stopped, the sailboat was taken by our chase motorboat. Therefore, the submarine and boat deviations illustrate the robustness of the controller, that was able to continue the mission as if nothing happened.

More details related to this mission and to the method (photos, C++, source code, videos) are available on http://www.ensta-bretagne.fr/jaulin/checking.html

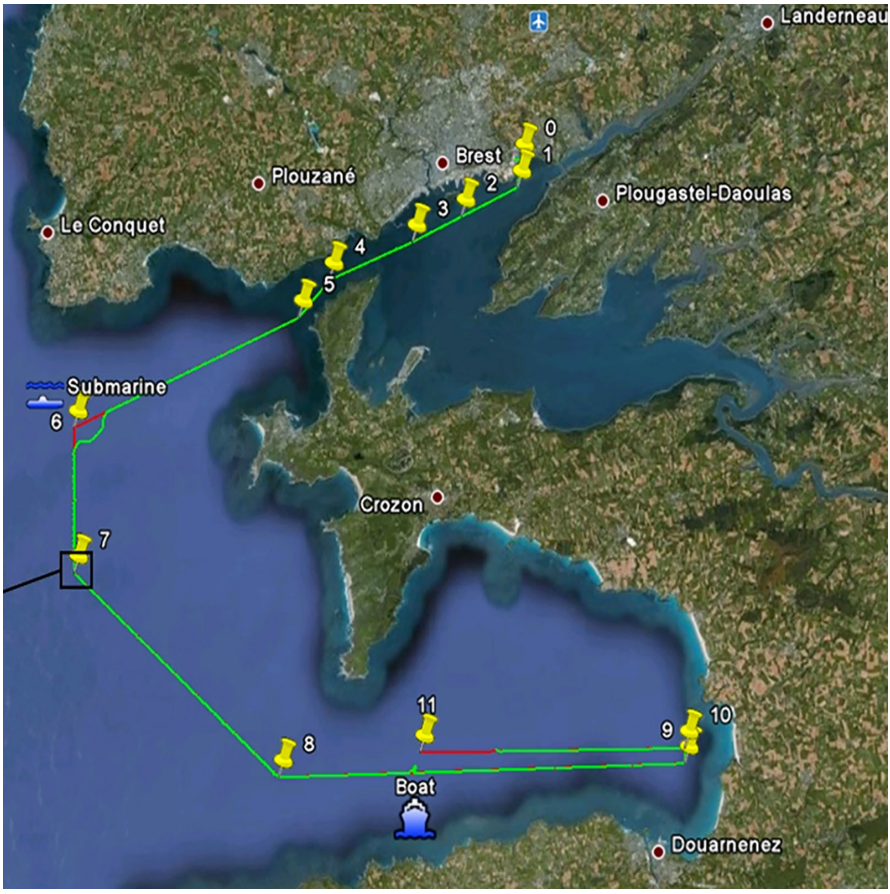


Fig. 8 Second long term mission

6 Conclusion

This paper has presented the different development phases of the autonomous sailboat *VAIMOS*. From a technological point of view, the project has opened a new path to an independent means of measuring environmental impact. From a scientific point of view, the project has brought a way to researchers to access data previously not easily measurable. Many tests and large scale missions of more than 100km have been performed. During those missions, the robot has never been at a distance more than 30 meters to its line. To our knowledge such accurate tracks for a sailboat robot in the ocean has never been done before.

Acknowledgements. The robot *VAIMOS* is the result of a collaboration between LPO (Laboratoire de Physique des Océans), RDT (Recherches et Développements Technologiques) of Ifremer (Institut Français de Recherche pour l'Exploitation de la Mer) and ENSTA Bretagne (Ecole Nationale Supérieure de Techniques Avancées). The authors render thanks to all people involved in the project: Y. Auffret, S. Barbot, L. Dussud, B. Forest, E. Menut, L. Quemeneur (RDT, Ifremer); F. Gaillard, T. Gorgues, J. Moranges, T. Terre (LPO) and B. Clément, Y. Gallou, L. Jaulin, F. Le Bars, O. Reynet, J. Sliwka and B. Zerr (ENSTA Bretagne).

References

1. Brière, Y.: The first microtransat challenge. In: ENSICA (2006), `http://web.ensica.fr/microtransat`
2. Gorgues, T., Ménage, O., Terre, T., Gaillard, F.: An innovative approach of the surface layer sampling. Journal des Sciences Halieutique et Aquatique 4, 105–109 (2011)
3. Guillou, G.: Architecture multi-agents pour le pilotage automatique des voiliers de compétition et extensions algébriques des réseaux de petri. PhD dissertation, Université de Bretagne, Brest, France (2011)
4. Herrero, P., Jaulin, L., Vehi, J., Sainz, M.A.: Combining set computation and feedback linearization for control. International Journal of Control Automation and Systems (2009) (to appear)
5. Jaulin, L., Le Bars, F., Clément, B., Gallou, Y., Ménage, O., Reynet, O., Sliwka, J., Zerr, B.: Suivi de route pour un robot voilier. In: CIFA 2012, Grenoble, France (2012)
6. Jaulin, L., Le Bars, F., Ménage, O.: An interval approach for stability analysis; Application to sailboat robotics. IEEE Transaction on Robotics 27(5) (2012)
7. Petres, C., Ramirez, M.R., Plumet, F.: Reactive path planning for autonomous sailboat. In: IEEE International Conference on Advanced Robotics, pp. 1–6 (2011)
8. Sauze, C., Neal, M.: An autonomous sailing robot for ocean observation. In: Proceedings of TAROS 2006, Guildford, UK, pp. 190–197 (2006)

An Arduino Compatible CAN Bus Architecture for Sailing Applications

Kévin Bruget, Benoît Clement, Olivier Reynet, and Bernt Weber

Abstract. This paper describes a Controller Area Network (CAN) Bus architecture based on Arduino compatible boards, to be used as an alternative communication system for robotic applications. This combines both, the robustness of CAN and the accessibility of Arduino software. The architecture is developed here to improve a Navigational Assistance System, which was initially created for disabled people. The system is composed of Arduino compatible boards, wired with various sensors and actuators, and communicating with an Human Machine Interface (HMI), directly accessible via a mobile phone or a tablet running on the open-source operating system Android. Information is transferred through the CAN bus architecture between multiple nodes (i.e. Arduino compatible boards) and the implementation of a CAN bootloader allows the reconfiguration of the nodes directly through the bus. The aim is to create a generic system able to work in various kinds of situations, adaptable to all kinds of users, including persons with all sorts of disabilities. This work will result in a demonstrator on a Miniji for the WRSC 2013 and an entirely joystick controlled boat for single handed sailing.

1 Introduction

Robotics in the sailing field is now a reality. Since 2008 with the first World Robotic Sailing Championship (WRSC) and International Robotic Sailing

Kévin Bruget · Benoît Clement · Olivier Reynet
IHSEV, Lab-STICC, OSM, Pôle STIC, ENSTA Bretagne, 2 Rue F. Verny, 29806 Brest, France
e-mail: {Kevin.Bruget,Benoit.Clement}@ensta-bretagne.fr, Olivier.Reynet@ensta-bretagne.fr

Bernt Weber
Splashelec, 18, rue de Pont à Louët, 29200 Brest, France
e-mail: bernt.weber@splashelec.com

F. Le Bars and L. Jaulin (eds.), *Robotic Sailing 2013*,
DOI: 10.1007/978-3-319-02276-5_4,

Conference (IRSC) [5, 9–11, 31], studies have shown that sailing robots can overcome the embryonic stage [18, 19, 23, 32]. Able to replace humans during competition or to realise autonomous measurements, its field of action becomes larger than before. Our approach is not to totally remove human action, but to assist it during information acquisition, decision process and execution (steering and trimming). Sailing globally has remained a field inaccessible for disabled people, because of the extreme mobility the sailor needs to acquire information and to manipulate commands of a boat. Sailing requires significant efforts that the disabled cannot afford. Splashelec aims to give disabled sailors access to those kinds of activities. The assistance system was initially composed of an electronic board and a joystick which allow a person to steer a boat manually as helmsman, forming part of a crew. When in need to free his hands, the helmsman can activate compass guided PID steering (autopilot).

Working with ENSTA Bretagne, an Android based Human Machine Interface (HMI) has been developed, in order to complete the system with visual navigational aid for disabled people [4]. To provide a more complete view of its environment for the skipper and to counterbalance his lack of mobility, some sensors, like wind sensor, compass and GPS were added to the system. Linked to an Arduino board, information is sent through Bluetooth to a tablet in charge of the information display.

At that point interfacing and cabling had already become complex and interfaces to connect extensions became a rare resource. Users then asked for joystick steering on smaller single handed boats, with need for supplementary actuators for the sails and more sensors to control the extra actuators. It was time to think about a new system architecture, with less connections for diminished cost and better reliability.

The present part of the project is the follow-up, which aims to develop a Controller Area Network (CAN [8]) bus interface board and the necessary software to allow communication using a more modular and flexible bus system, that is easily adaptable to all sorts of situations and disabilities. That is to say, developers can plug their own sensors to the system and show data directly through the tablet. On the software side, an Arduino bootloader compatible with CAN allows the programming of nodes directly through the bus. There is also a support for debugging messages relayed by the bus.

The main objective of this work is to provide to sailors, disabled or not, a navigational assistance system able to perform automated tasks (heeling limitation, autopilot, ...). It integrates the needed adaptability into the development process and aims to offer a solution for various kinds of disabilities. For example steering with a joystick compensates the lack of strength in the arm, the HMI centralises sensor information to compensate the lack of mobility and sensitivity (wind, boat speed,...). Furthermore, due to its open-source nature, the software system is entirely and easily modifiable and developers can add functionality or change the HMI according to users specific needs. Finally the system should be adaptable to every kind of boat.

This paper describes a complete system and its features, its architecture and the communication process between nodes. The demonstrator, an adapted Miniji boat will be presented at the WRSC 2013.

2 Related Work

2.1 Low-Level Architecture of Existing Robot Systems

Due to the complexity of tasks mobile robot are designed for, robots use generally embedded computers with substantial processing power and are often centered around one of them. Connecting to the robot's hardware is done using all sorts of interfaces available to the computer: USB, Ethernet, serial (RS232, RS435, RS485), and using hubs and port concentrators (e.g. NASA's [25] or Roboat [33]) where necessary. Employing microcontrollers in place of some port concentrators gives access to all the lower level electronic interfaces as I2C, SPI (NAO [29]).

In its simplest configuration, our system consists only of a joystick and a rudder actuator. In this configuration, there is no need for any embedded computer, the system implements only low-level reactive behaviours (compass controlled or joystick steering). A very simple microcontroller can execute these tasks, at low cost and with low current consumption.

As for the embedded computer centered architectures, everything must be wired to the one central microcontroller, but resources and available interfaces are more sparse on the lower abstraction level and the number of connected sensors and actuators is very limited. Centralized architectures become impractical when more components have to be connected to one simple microcontroller.

Bus systems offer a flexible and modular solution to interconnect microcontrollers, each having sensors and/or actuators attached. Historically used RS-485 multi-point serial cabling is more and more replaced with higher level CAN, with help of hardware implementations of the CAN protocol itself now integrated to microcontrollers. As for automotive and industrial field buses, CAN is today used in robots and results there in highly adaptable hardware architectures (Merten and Gross [24]).

Merten and Gross' robot architecture uses the CANopen [7] protocol. Open-source implementations of this protocol exist (CANFestival [6], CANopen SlaveLib [38] and CANopenNode [17]).

So we retained the CAN bus, but decided to go with an architecture able to work with very simple nodes and a network that is able to run without an embedded computer. Being able to run without embedded computer means being able to power-down the computer regularly for energy savings, or running entirely without it in simple configurations. And as we understand it, the three cited CANopen implementations do not allow to work without an

embedded computer or high performance microcontroller. The missing piece seems to be a suitable simpler protocol, we are not aware of a high-level (on-top of CAN) open-source protocol, simple enough for 8-bit microcontrollers, providing services necessary for sensors and actuators distributed over the bus, as for example synchronised sensor reading and distributed control loops.

2.2 Commercial Marine Electronics

2.2.1 Communication Networks

Displays and autopilots installed on today's sailing yachts use mostly multidrop serial buses with vendor specific proprietary protocols. Examples are Raymarine's SeaTalk [27] and NKE's Topline [13]. On some recent boats, CAN based, industry standard NMEA2000 [3] buses replace the proprietary protocols. For higher bandwidth requirements such as radar or echo sounder images, these buses are sometimes completed by extra Ethernet cabling (e.g. Furuno Navnet [14], Raymarine SeaTalkHS [26]).

Note that the Ethernet approach is power-hungry, costly and difficult to adapt to simple 8-bit microcontrollers, but industry choice CAN based NMEA2000 could be a good solution. The problem is that NMEA2000, and the J1939 protocol it is based on, are proprietary protocols [37]. Once again, as with CAN based robot architectures, CAN looks promising, but the missing piece is an adapted open source protocol, that would allow to design and integrate new hardware for different situations of handicap.

2.2.2 Autopilots

Commercial available autopilots are very close to joystick assisted steering for disabled people:

- Commercial autopilot actuators are used for steering by disabled. Splashelec uses for most boats actuators sold with commercial autopilots. A difference exists though in dimensioning for small vessels up to about 10 m length: a typical tiller autopilot uses an electrical motor with a power rating of about 10 to 20 W, which allows only for slow rudder movements compared to what a human helmsman can do. To give a disabled person using a joystick the same performance, the actuator has to be oversized compared to a commercial autopilot.
- Power electronics of commercial course computers as well Splashelec's model are designed around a H-bridge, allowing speed modulation for smooth rudder movements. There is no technical difference.
- Firmware of commercial autopilots is very specialized to course keeping following compass, wind or GPS data. All the autopilots known to us use

PID closed loop control, calling the PID coefficients "rudder", "counter-rudder" and "auto-trim" in the manuals.

Some commercial autopilots allow joysticks to control the rudder: pushing the joystick moves the rudder, when the joystick returns to the center, the rudder stays in position. A subset of these systems allows also for a proportional mode: the rudder follows the joystick (i.e. [28]). Such a system offers assistance to a disabled helmsman and some are used in this way [36].

Modifying firmware, which is not possible with commercially available autopilots, would allow for further adaptation to individual disabilities:

- joystick damping: many disabilities produce jerky hand movements. Simple position-averaging allows fine control of a sailing yacht.
- automatic adaptive correction of the rudder position corresponding to the joystick center: under sail, a boat is generally not completely balanced (i.e. weather helm), the rudder must keep an angle to steer straight. A simple proportional joystick must then stay continuously in an off-center position, against the spring, which is uncomfortable, and becomes impossible if the handicap implies reduced dexterity.

2.3 Existing Assistance Systems Designed for Disabled Sailors

The company "Hansa Sailing" [30] is a manufacturer selling a range of boats and assorted electric control systems destined for disabled sailors. Wiring applies star topology around a control box (Access Liberty Manual: [20]). Winch and rudder actuators have 2 wire connectors for DC-motors. The input devices need a 9 wire connection, typically a joystick with associated push buttons for mode selection. No sensors are associated to the DC-motors.

Steve Alvey's company [22] sells electric controls for the Martin 16 and other sailing boat types used by disabled people [2]. He also designs specialized systems on purpose. An example is the boat steered by Hilary Lister around Britain; She uses a three straw sip-and-puff interface [16], and has access to a Raymarine autopilot.

The manual of the Martin 16 system [1] shows a Raymarine 3 pin socket labelled "Sea Talk" for command interface connections (joystick, sip and puff, etc.), the bus system simplifying cabling. A second socket labelled "Helm Drive" mates the standard connector of the used Raymarine ST4000/SPX-5 electric actuator (2 used pins for direct DC-motor connection). No additional sensors are added to the actuator.

Experience using our rudder only steering system showed us that using a position sensor enables tactile feedback, by establishing a relation between rudder and joystick position. And, in this setup, a bigger rudder angle makes

water push harder on the rudder, as does the center return spring in the joystick. The helmsman feels the rudder, its position and the water force.

Our conclusion is that connection count of existing sailing assistance systems for disabled has reached a limit, where it becomes impractical and expensive, in the environment where every connector has to be waterproof and salt water resistant. Joystick box connections have of up to 9 wires, adding sensors to individual actuators would complicate the cabling even more. Bus systems have already been employed to user-interfaces, we think that a bus system should be extended to the entire system, including the actuators and the associated sensors. This would simplify installations and allow more advanced interactions, better user experience and make the systems more modular, flexible and adaptable.

3 System Description

The system is actually composed of two major parts, the Programmable Servo Controller (PSC) and the HMI, which displays sensor information and allows the skipper to activate the joystick or to use the autopilot. The communication between the bus and the HMI tablet uses Bluetooth, whereas internal communication is made entirely using the CAN protocol. Its behaviour can be assimilated to a closed-loop system (Fig. 1). Information which comes from the environment as any human input has influence on the actuators, that is to say the system will be able to perform automated tasks. An example is a standard course keeping autopilot, or joystick steering mode, with the autopilot taking over when necessary to limit heeling.

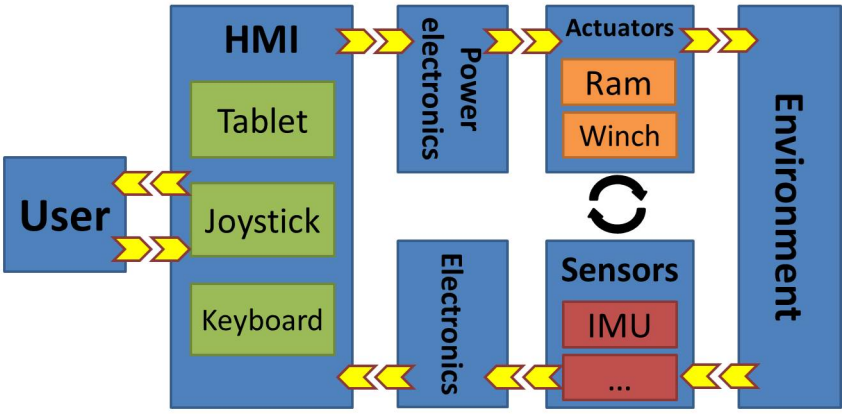


Fig. 1 The system is based on a closed-loop where the user can be removed to implement autonomous behaviour

The programmable servo controller PSC (Fig. 2) is a key component of the system and can do the work of an autopilot course computer, which can steer to wind or compass when associated to the corresponding sensors. The PSC board (6 x 7.5 inches) contains a microcontroller, power electronics for an electric motor (ram or winch), an electric clutch and the power supply including filter circuits (see Fig. 3). Various interfaces for rudder angle sensor, joystick and control keyboard already exist and, in future versions, a CAN bus interface will be integrated directly to the main circuit board. To add electric winches, we use one instance of the PSC for each.

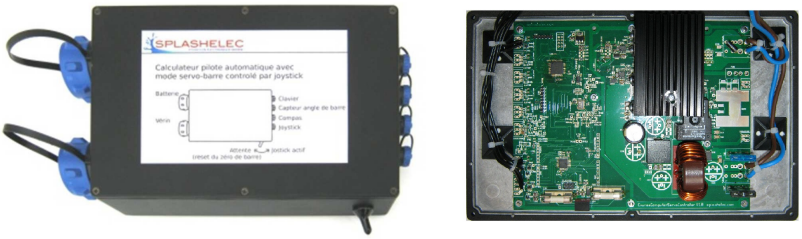


Fig. 2 PSC of the Splashelec system in a waterproof case. It contains several inputs for the keyboard, the compass, the rudder angle sensor, the battery.

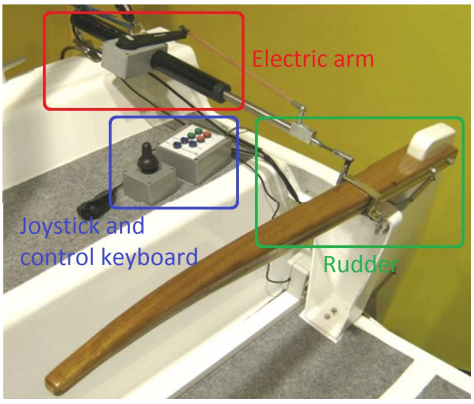


Fig. 3 The Splashelec system composed of an electric ram (in red), a joystick and a control keyboard (in blue) and the rudder (in green)

This PSC is based on open-source technology in order to allow the easy integration of new functionalities or to modify actual ones. The microcontroller is compatible to Arduino boards and can so be programmed with Arduino's Integrated Development Environment (IDE) software, which gives an access to the programming interface, existing libraries and various on-line examples. Results of this project (material and software) are publicly available on the

Internet [34]. Wired to this box, multiple sensors such as an Inertial Measurement Unit (IMU), a wind sensor or loch-speedo are linked into a CAN architecture. When using more than one actuator, each one uses its own dedicated PSC, with instances of the power electronics, local sensors, and a CAN bus interface. All the system, including mechanical parts, is transportable and can adapt to different boats.

The HMI, programmed for Android by using the Android Software Development Kit (SDK) tool for Eclipse [12], contains different areas (Fig. 4): in addition to textual information, some data, such as wind orientation and speed, compass and rudder angle, is also displayed in graphic form to allow a better understanding of the environment. A last area acts as a virtual keyboard to switch between autopilot and joystick mode and the user can modify the course to steer. The graphical layout of the HMI is realised in XML language and could be easily modified to fit needs of specific users. This results in a system (joystick, tablet display, extra keyboard,...), compatible with different disabilities, while still being usable by sailors without any disabilities.

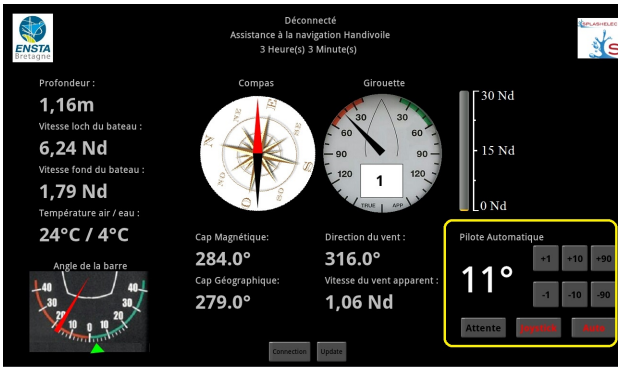


Fig. 4 The Human Machine Interface displays sensor information and contains a virtual keyboard (in yellow) to replace the physical one

4 CAN Bus Architecture

The need for a bus architecture came from the multi sensor and actuator problematic and the cabling and communication complexing with each added element. Used first in automotive applications, the CAN protocol is now widely available and starts migrating into many non-automotive applications. This open standardised high layer protocol provides a reliable message exchange system between various nodes.

The next section describes the protocol used in our system and its architecture. Finally the following section discusses the implementation of a Arduino CAN bootloader able to reconfigure nodes through the bus.

4.1 CAN Protocol

The choice of the CAN protocol for our architecture arose from the need of a broadcast communication mechanism, that had to be easy to use, able to work with multiple nodes and which provides a reliable communication protocol to exchange information from sensors and actuators. Indeed CAN is able to detect errors with no less than three mechanisms: Cyclic Redundancy Check (CRC) to verify message integrity, Frame check to verify that data is sent in the correct shape and acknowledgments to guarantee reception. Besides, adding nodes to an existing CAN network can easily be done, which meets our needs for a modular architecture.

Adding new nodes to the network is straight forward: adding a new sensor with interfaces as for example NMEA 183 (National Marine Electronics Association) or I2C (Inter Integrated Circuit), implies reading the data by the connecting nodes microcontroller and to put the data in a CAN frame. Arduino allows to do this in very few lines of code, using libraries for CAN and a plethoric choice of sensor interfaces.

4.1.1 Higher-Layer Protocol

At this point appears the need of a higher level CAN protocol to organize data in the CAN frames. Various higher-layer CAN protocols already exist such as SAE J1939 used in NMEA 2000, CAN Kingdom, or CANopen but it appears that no one fits our need of a simple CAN protocol. Too complex or not adapted for 8-bit microcontrollers, CAN protocols are not widespread and their code is not always open-source. It has been decided to develop our own protocol based on our needs, called SimpleCAN, into an Arduino library containing the essential functions to support our architecture. The protocol is designed in a way permitting to add new features afterwards.

4.1.2 System Architecture

Our architecture (Fig. 5) is based on Arduino compatibility and uses a CAN bus interface card we call the CANinterfacer. The CANinterfacer is compatible to an Arduino with a CAN shield on top. This on purpose designed board [35] uses an ATMEGA32U4 microcontroller as do the Arduino Leonardo and Micro. It is small in dimension (1.95 x 1.95 inch, slightly smaller than 5 x 5 cm) and can be programmed by USB using the Leonardo

bootloader and the Arduino IDE. It can be powered from the CAN bus by an on-board switching power supply accepting from 7 to 32 Volts, most of the I/O pins are available for local connections.

In the system, a group of elements (actuators, sensors, HMI elements,...), wired to a CANinterfacer, becomes a CAN node (Fig. 6). That is to say, each CANinterfacer typically uses Arduino libraries to convert input from various sources, for example analog inputs, NMEA 183 or I2C connected sensors, into a CAN messages. Putting a CANinterfacer between the new hardware and the bus to integrate it as standard node gives great flexibility.

In the same way, the upcoming version the PSC will contain an CANinterfacer and be a native node in our bus system. This type of node allows to integrate servo controlled actuators such as rams and winches with their associated sensors.

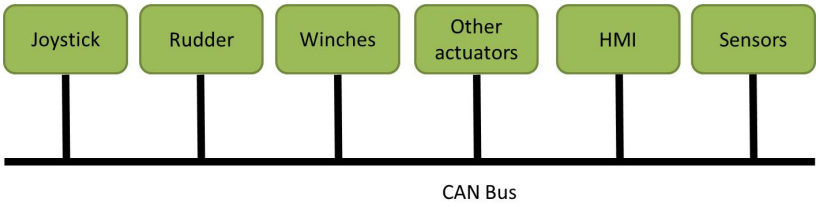


Fig. 5 CAN architecture of the system

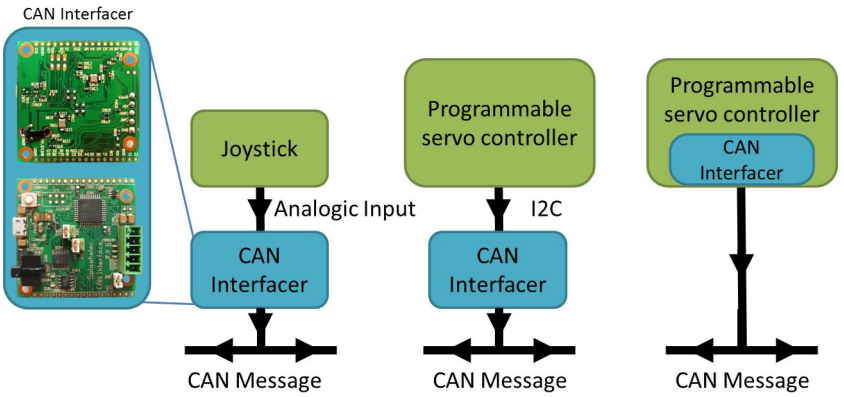


Fig. 6 Examples of a CAN node: The joystick and the Programmable Servo Controller (PSC) in its first version. The next version will have a CANinterfacer integrated into the board.

4.2 CAN Bootloader

The implementation of the CAN protocol opens the way to simplified programming of every node directly through the bus, one unique connection to the bus permits this. To obtain that, we need a new Arduino bootloader that accepts CAN programming commands for our CANinterfacer. Fabian Greif from the Robotics Club of Aachen, Germany, has already worked on the subject with very close hardware [15] and built its own CAN bootloader, that allows updating firmware and local code from CAN messages. Small modifications have been made, in order suit connections of our CANinterfacer. [35].

The programming operation is initiated by a Python script that initiates the communication process between the PC connected programming node and the node which needs to be reconfigured. More concretely, the programming node will act as an In-System Programming (ISP) interface: it receives the new program by USB or serial port and sends it encapsulated in CAN messages to reprogram the specified node. The whole process is detailed in [35].

4.3 Demonstrator

The final objective of the current project is to build a demonstrator to show robotic functionality during WRSC 2013 [21] that proves the flexibility of such an architecture. The choice of the boat fell on a Miniji (Fig. 7) made available by the association "Handivoile Brest", which is based on a small scale replica of a historic America's Cup hull. It is a single-handed sailing boat, ordinarily steered by foot pedals or with a steering wheel. In a comfortable position, it offers vivid sensations to the sailor housed in a bucket seat. Inexpensive and very technical, the boat type was adopted by many French disabled sailors. But it is also used as sailing robot for the VAIMOS project of Institut Français de Recherche pour l'Exploration de la Mer (IFREMER) and ENSTA Bretagne [18, 19, 23].

Furthermore, there is demand to increase the autonomy and safety, coming from instructors working around disabled sailing. The boat will be equipped with two electric winches and a rudder system, all interconnected by the CAN bus. By adding sensors to the bus (e.g GPS and IMU), this boat will be able to perform automated tasks, as does a sailing robot. Indeed, to increase security, we can limit the heeling or restrict the navigational area. Instructors will also be able to take control of the boat with a remote controller for safety reasons or even to activate the autopilot if the skipper becomes unable to manipulate commands.

Fig. 7 The system will be implemented on a Miniji sailboat to act as a demonstrator

5 Conclusion

In conclusion we have a system based on a CAN bus architecture which allows developers to connect new sensors or actuators, and to reprogram the system easily using Arduino technology. Such a system assists the sailor during navigation and can automate complex tasks. It helps disabled by easing the access to the sailing activity, or gives any other people navigational assistance. Based on the open-source approach, electronics and software can be modified according to specific requirements, numerous opportunities for development exist. During WRSC 2013, a prototype will demonstrate robotic functionality. The final aim is to obtain products with new features derived from robotic sailing, encouraging people to develop their own system modifications.

References

1. Alvey, S.: Martin 16 Power-Assist System - Mk IV, Operator Manual (2005), http://www.martin16.com/uploads/auto_mkiv.pdf
2. Alvey, S.: Martin 16 Power-Assist System - Mk IV, Brochure, http://www.martin16.com/uploads/autobrochure.pdf (cited June 23, 2013)
3. National Marine Electronics Association: NMEA2000(TM) standard. In: National Marine Electronics Association website (2012), http://www.nmea.org/content/nmea_standards/nmea_2000_ed3_00.asp
4. Brocheton, N., Bruget, K., Wibaux, A., Reynet, O., Clement, B., Weber, B.: Systeme d'assistance a la navigation handivoile. In: Proceedings of Handicap 2012: 7th Congress on Technical Assistances for Disabled People, Paris, France (2012)

5. Sauze, C., Finnis, J.: Proceedings of the 5th International Robotic Sailing Conference. Springer (2012)
6. CanFestival: CanFestival, a CANopen framework, http://www.canfestival.org/ (cited May 13 , 2013)
7. CiA: CANopen. In : CAN in Automation (2013), http://www.can-cia.org/index.php?id=systemdesign-canopen
8. CiA: CAN specifications. In : CAN in Automation, http://www.can-cia.org/index.php?id=can (cited May 13, 2013)
9. Collective: Proceedings of the 1st International Robotic Sailing Conference (2008)
10. Collective: Proceedings of the 2nd International Robotic Sailing Conference (2009)
11. Collective: Proceedings of the 3rd International Robotic Sailing Conference. Universidade do Porto (2010)
12. Eclipse: Eclipse (software) (2013), http://www.eclipse.org/
13. Nke Marine Electronics, http://www.nke-marine-electronics.com/home.html (cited June 23, 2013)
14. Furuno: Furuno Navnet (2012), http://www.navnet.com/
15. Greif, F.: CAN Bootloader. In : Universal CAN library. Roboterclub Aachen e.V. (2010), http://www.kreatives-chaos.com/artikel/can-bootloader
16. hilarylister.com: Sip & Puff (2012), http://www.hilarylister.com/d5483/hilary_s_boat/sip_amp_puff.aspx
17. Janez: CANopenNode, http://sourceforge.net/projects/canopennode/ (cited June 23, 2013)
18. Jaulin, L.: Modelisation et commande d'un bateau a voile. In: Proceedings of 3rd Conference Internationale Francophone d'Automatique. Douz,Tunisie (2004)
19. Jaulin, L., Clement, B., Gallou, Y., Le Bars, F., Menage, O., Reynet, O., Sliwka, J.: Suivi de route pour un robot voilier. In: Proceedings of 7th Conference Internationnale Francophone d'Automatique, Grenoble, France (2012)
20. Access Dinghy Sailing Systems Pty Ltd.: Access Dinghies OPERATIONS & SAFETY MANUAL - LIBERTY, http://www.sailingforall.com/imgs/74liberty_operation_&_safety_manual_pdf.pdf (cited June 25, 2013)
21. Melguen, M., Le Bars, F.: Official WRSC 2013 Website (2013), http://www.ensta-bretagne.eu/wrsc13/
22. Martin16: M16 sailboat for able and disabled sailors, http://www.martin16.com/ (cited June 23, 2013)
23. Menage, O., Gaillard, F., Gorgues, T., Terre, T., Rousseaux, P., Prigent, S., Auffret, Y., Dussud, L., Forest, B., Repecaud, M., Jaulin, L., Clement, B., Gallou, Y., Le Bars, F.: VAIMOS: Voilier autonome instrumente pour mesures oceanographiques de surface. In: Symposium on Vulnerability of Coastal Ecosystems to Global Change and Extreme Events, Biarritz, France (2011)
24. Merten, M., Gross, H.M.: Highly adaptable hardware architecture for scientific and industrial mobile robots. In: RAM, pp. 1130–1135. IEEE (2008)
25. Park, E., Kobayashi, L., Lee, S.Y.: Extensible hardware architecture for mobile robots. In: ICRA, pp. 3084–3089. IEEE (2005)

26. Raymarine: Ethernet. In: Raymarine website, http://www.raymarine.eu/view/?id=5537
27. Raymarine: SeaTalk/SeaTalk1, http://www.raymarine.eu/view/?id=5535&collectionid=26&col=5557 (cited June 23, 2013)
28. Raymarine: Autopilot Joystick. In: Raymarine website, http://www.raymarine.com/view/?id=2705 (cited June 24, 2013)
29. Aldebaran Robotics: NAO Software 1.14.3 documentation: low level architecture. In: website of Aldebaran Robotics, (cited June 23, 2013)
30. Hansa Sailing, http://hansasailing.com/ (cited June 23, 2013)
31. Schlaefer, A., Blaurock, O.: Proceedings of the 4th International Robotic Sailing Conference. Springer (2011)
32. Sliwka, J., Nicola, J., Coquelin, R., Megille, F.B.D., Clement, B., Jaulin, L.: Sailing without wind sensor and other hardware and software innovations. In: Proceedings of the 4th International Robotic Sailing Conference. Springer, Luëbeck (2011)
33. Stelzer, R., Jafarmadar, K.: The robotic sailing boat asv roboat as a maritime research platform. In: Proceedings of 22nd International HISWA Symposium (2012)
34. Weber, B.: Splashelec autopilot documentation. In: The Splashelec Wiki, http://wiki.splashelec.com/index.php/autopilot (cited May 13, 2013)
35. Weber, B.: The CAN Interfacer. In: The Splashelec Wiki, http://wiki.splashelec.com/index.php/caninterfacer (cited May 13, 2013)
36. wetwheels.co.uk: About the boat, http://www.wetwheels.co.uk/about-us/about-the-boat/ (cited June 24, 2013)
37. Wikipedia: NMEA 2000. In: English Wikipedia, http://en.wikipedia.org/wiki/nmea_2000 (cited June 23, 2013)
38. Zulliger, R., Tisserant, E.: CANopen SlaveLib, http://canopen.sourceforge.net/projects/slavelib/slavelib.html (cited June 23, 2013)

Part II
Energy and Power Management Strategies

Development of ARRTOO: A Long-Endurance, Hybrid-Powered, Oceanographic Research Vessel

Paul Miller, Colin Sauzé, and Mark Neal

Fig. 1 ARRTOO prototype during trials in Wales, March 2013

Abstract. This paper describes the design, development, construction and demonstration of a hybrid (sail and electric motor) powered, small, robotic research vessel. The two-metre vessel is capable of speeds of up to five knots under sail and six knots using an electric motor without the need for human interaction. It is launchable by two people from the beach or a vessel. Plans

Paul Miller
United States Naval Academy
e-mail: phmiller@usna.edu

Colin Sauzé · Mark Neal
Department of Computer Science, Aberystwyth University
e-mail: {cos,mjn}@aber.ac.uk

F. Le Bars and L. Jaulin (eds.), *Robotic Sailing 2013*,
DOI: 10.1007/978-3-319-02276-5_5,

are presented for scaling this design up to a 4.85 metre long version capable of reaching 10 knots when motoring.

1 Introduction

This paper details the design for a 4.85 metre long hybrid sail and electric motor autonomous robot surface vessel and the development of a two metre long proof of concept prototype. The two-month long Autonomous Robot for Rapid Transit and Ocean Observation (ARRTOO) project was undertaken by Aberystwyth University in response to a Small Business Research Initiative from the Natural Environment Research Council and Defence Science and Technology Laboratory for a long-endurance, unmanned vessel that was beach launchable and capable of supporting in-situ water quality sensors, meteorological sensors and towed array sonars. The full size vessel is required to provide long endurance under sail (at approximately two knots), rapid transit under power (approximately 10 knots) and to transition between the two modes autonomously. The prototype is a sailing robot with a retractable keel and reefable sails that permit high-speed motoring by reducing windage and drag. At the time of writing an autonomous control system for the boat has not yet been developed.

2 Prototype Design

Numerous vessel types conceivably meet the project requirements, including solar-powered catamarans [3], wave-powered surface vessels [4] and possibly Slocum gliders [1].Given the authors background in sail-powered, autonomous surface vessels a conventional monohull that used sails and an electric motor was selected.

In general, hull forms that are good sailers are rarely good under higher speed motoring. For this project the hull would be in the displacement mode while under sail and the semi-planing mode under power. As this would be an ocean-going vessel pounding had to be considered in both regimes. The final hull form selected was a V-bottom monohull developed from a traditional hull series that evolved in Bermuda, the Bahamas and along the eastern seaboard of the United States during the period 1870-1930 [2]. During that time frame the small craft commercial fleet was transitioning from sail to power. While the new internal combustion engines offered significant advantages their reliability was not high and the vessels often had to rely on sails for propulsion. Many jurisdictions also only allowed engine usage while transiting to and from the fishing grounds, but not while fishing. This meant the hull forms had to perform well both under sail and power and with a high degree of

seaworthiness, the same requirements shared by ARRTOO. The shape also accepts a very wide payload range with little detriment to performance or seaworthiness as befits its fishing boat heritage. Figure 2 shows the hull lines and 1 shows a photo of the completed boat for a two-metre prototype version of the vessel. The forward sections have significant deadrise to reduce pounding and deflect spray, while the aft run is flat with hard-chines to reduce drag under power. Results using the Delft3 and Savitsky Pre-Planing parametric hull resistance formulas indicate the vessel has a low drag profile up to three knots, indicating it will sail or motor easily and efficiently at speeds up to that point. The resistance curve above that is shallow and linear up to seven knots, indicating few vices and an easy to drive hull form. The predominantly flat sections are easy to build in a variety of materials. Table 1 shows the primary characteristics of the two-metre prototype.

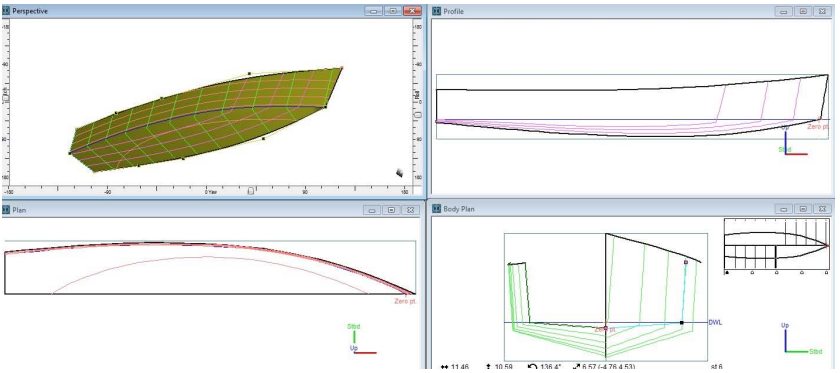


Fig. 2 Hull Lines for the Two-metre ARRTOO Prototype

The hull and deck were built over a station mold with 2mm marine ply covered on each side with 135 g/m2 glass cloth. The three station molds located at the masts and keel trunk became bulkheads. Both the untapered keel and rudder were 12% SD8020 airfoil sections. Construction of the 100 mm chord foils was in balsa covered with fiberglass. The keel additionally had two plies of 300 g/m2 unidirectional carbon fibre. To aid in launching, recovery and to reduce drag under power the epoxy/lead shot bulb was designed flatter than normal (2.5:1) and partially recesses into a moulded hollow in the hull. Similarly, the transom-hung rudder is retractable. On the full-scale vessel the rudder was planned to be in a rotating cassette as used on the International Canoe so that it would not interfere with the towed array launching and recovery equipment. Off-the-shelf fibreglass (OTS) tubes sized to fit the OTS 25 mm diameter carbon tubes were bonded to the bulkheads as mast steps. The two-masted, free-standing, sailing rig matches the hull form as well as providing mounting points for equipment. The untapered carbon-fibre masts

Table 1 Primary Characteristics of Two-Metre ARRTOO Prototype

Displacement	29.5 kg
Draft	0.83 m
Canoe body draft	0.089 m
WL Length	1.95 m
Beam max extents on WL	0.479 m
Wetted Area	0.822 m^2
Max sect. area	0.025 m^2
Waterpl. Area	0.745 m^2
Prismatic coeff. (Cp)	0.602
Block coeff. (Cb)	0.358
Max Sect. area coeff.(Cm)	0.624
Waterpl. area coeff.(Cwp)	0.798
LCB %	-52.4 from zero pt. (+ve fwd) % Lwl
LCF %	-56.9 from zero pt. (+ve fwd) % Lwl
KB	0.058 m
KG	-0.193 (keel dn) 0.09 (up) m
BMt	0.384 m
BML	6.027 m
GMt	0.635/0.352 m
GML	6.001 m
KMt	0.442 m
KML	6.085 m
Immersion (TPc)	0.007 tonne/cm
Length:Beam ratio	4.072
Beam:Draft ratio	5.403
Length:$\sqrt[3]{Vol}$ ratio	6.304

are the same length as the boat making land transportation and storage easier and, of major importance; the aft mast acts as a hoisting location for the drop keel arrangement. Figure 3 shows the preliminary sail plan for the full-scale vessel. Twelve millimetre Harken tracks and cars served to provide a low friction method of attaching both sails and the keel-lifting car. Most rigging featured equipment proven in previous Aberystwyth University [6] and USNA projects [5]. These included a SmartWinch 380 for the sail sheets and a Futaba S3306MG servo for the rudder. The sail halyard was raised and lowered using a toothed belt over a Graupner Sailwinch 4. The keel was raised and lowered using a in-house built steel lead screw driven by an electric motor. For propulsion a 700W electric motor powered by a Lithium Polymer battery drives a relatively traditional three-bladed "square pitched" propeller on a shaft.

A challenge for both the prototype and the full-scale vessel was determining the payload, which had only been vaguely defined. Similarly, the primary

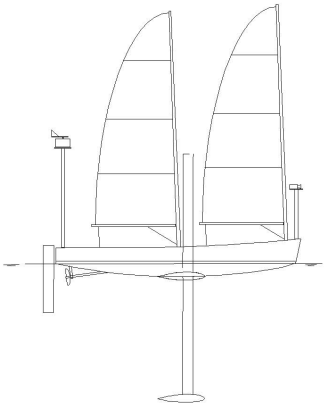


Fig. 3 Preliminary Sail Plan for Two-Metre Prototype

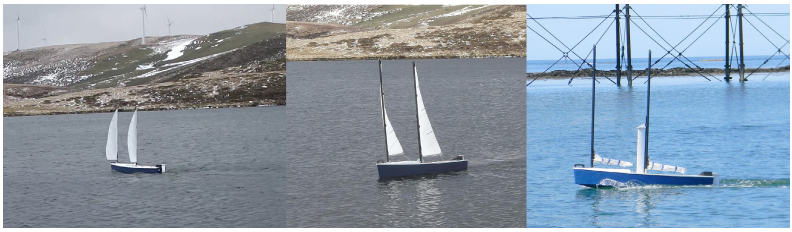


Fig. 4 The prototype sailing. Left: Sailing downwind, Centre: Sailing Upwind, Right: Motoring.

drive system was unknown at the time of design and weights and volumes were taken as educated guesses. Table 2 shows the prototype's as-built weights.

2.1 Prototype Performance

The 2 m prototype was tested on a number of occasions to assess its performance under sail and power in a range of conditions. The first test was made on a day with between 3 and 5 knots of wind and showed that even in light winds it was possible to average around 1.5 knots, tack, gybe and to sail effectively on all points. The robot was able to sail to within around 45 degrees of the wind. A further test was carried out in wind strengths between 15 and 18 knots and both sailing and motoring speed trials were undertaken. Under sail the robot was able to sustain speeds of around 2.5 knots with a peak of 4.5 knots. Motoring directly upwind in these conditions with the keel retracted but sails unreefed ARRTOO was able to maintain 4.9 knots.

Table 2 Weights of components of the two metre ARRTOO prototype

Area	Item	Weight (kg)
Hull	Deck w/misc hardware	2.3
	Bare hull structure	3.2
Keel	Keel	1.3
	Bulb	10.2
Rigging	Mast, Boom and Light	2.0
	Mast Tracks	0.9
	Hatch Covers	0.5
	Deck Hardware	0.7
Control Surfaces	Rudder	0.9
System Components	Sail Winches	1.4
	Wind Sensor, housing and wiring	0.5
	Batteries	1.5
	Oceanographic system	2.8
	Navigation System (w/case)	0.5
	Compass (w/case)	0.2
	Fuse/Switch Box	0.3
	Rudder Servo	0.4
	Total	29.5

These trials were undertaken fully ballasted and carrying an oceanographic payload (YSI-6600-V2 water quality monitoring sonde and pump system). A further motoring test in calm conditions with the keel retracted and the sails lowered yielded a peak speed of 6.4 knots. The boat performed well under sail with small amounts of lee helm in light winds (1-3 knots) and a little weather helm in windier conditions (15-25 knots). Figure 4 shows the prototype sailing downwind, upwind and motoring with the sails down. The trim angles under power matched the predicted two-three degrees pitch from the Savitsky analysis. The wake was flat and the bow spray was seen to deflect as hoped. Note in the motoring photo the bow knuckle is well clear of the water, which should result in acceptable manoeuvrability in the ocean.

To demonstrate its capabilities as an oceanographic observation platform, data from the sonde and GPS were recorded on an on-board computer and transmitted via a ZigBee radio link to a shore station. A sample plot showing turbidity measurements can be seen in Figure 5. This figure shows the turbidity is high immediately after launching (coloured pink) due to the

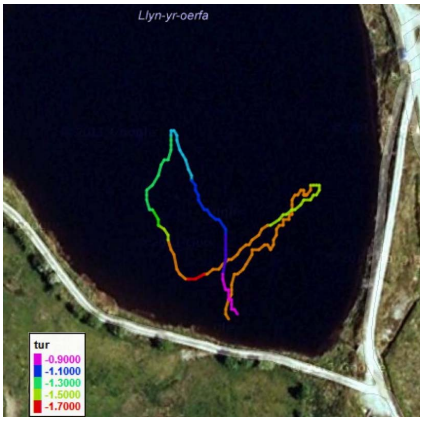


Fig. 5 A map showing the turbidity levels during the test run on Llyn-Yr-Orefa (52.4N,3.87W) near Aberystwyth, Wales

disturbance of sediment during launching and then reduces rapidly as the robot moves away from the launch area.

3 Full Sized Design

The full sized 4.85 m long ARRTOO was designed for self-sufficient deployment for two months, including a 100 nautical mile transit to and from the research area using the electric motor and on-station low-speed operation using sails.

3.1 Power Budget

The power budget for the full size design is shown in tables 3 and 4. It is based on figures from our experience on similar sized sailing robots in long term tests at sea. Various power generation options are available including up to 260 WPeak of photovoltaic solar panels, one or two Forgen 1000NT vertical axis 45 W wind turbines and the option to use the diesel engine to recharge the batteries. The solar panels alone should generate an average of 26 W, with a control system power requirement of 4 W this leaves 22 W for payload use. There will be a 90Ah 12V LiFePO4 battery to store payload power which will be recharged using these power sources, as well as from the engine alternator when motoring or when insufficient power is generated by the renewable sources.

Table 3 The power budget for devices consuming power

Component	Duty Cycle	Current (A)	Voltage (V)	Power (W)
Sensors:				
Microcontroller	1	0.02	3.3	0.066
Compass	1	0.005	3.3	0.0165
Wind sensor	0.05	0.06	12	0.036
GPS	0.1	0.025	3.3	0.00825
Actuators:				
Rudder actuator	0.1	2.5	12	3
Sheet actuators	0.01	6	12	0.72
Halyard actuators	0.001	6	12	0.072
Keel actuator	0.001	10	12	0.12
Communications:				
Microcontroller	0.2	0.02	3.3	0.0132
ZigBee module	0.001	0.02	3.3	0.000066
Iridium SBD modem	0.18	0.15	5	0.135
Satelite tracker (independent)	0.05	0.06	3.7	
			Total:	4.187016

Table 4 The power devices producing power. Note that the second wind turbine has a reduced efficiency because it is partially blocked by the first wind turbine and that the alternator is not used during long endurance missions due to fuel limitations and is not included in the total power budget.

Production	Efficiency	Peak Power (W)	Average Power
Solar P.V.	0.1	260	26
Forgen 1000NT (#1)	0.4	60	24
Forgen 1000NT (#2)	0.3	60	18
Engine Alternator	0.75	3700	2775
		Total:	68
		Available for payload:	63.813

3.2 Onboard Electronics

The control system will be based on an architecture developed for previous sailing robots built by Aberystwyth University. The system uses two microcontrollers: one for control of sailing, monitoring sensors etc and one

for managing (non-payload) data-logging and communication via the Iridium satellite network and ZigBee during launch and recovery. This split ensures that accurate control is maintained during extended data communication periods. A separate Iridium modem for the payload system reduces difficulties when integrating new payload systems and decreases the coupling between control and payload systems and helps to prevent faults in one system affecting the other. There is also provision for an additional communication device and AIS and RADAR systems to be installed. There is provision for up to eight payload devices, connected to the payload management microcontroller by either NMEA 0183, USB, RS232, I2C or SPI protocols as shown in figure 6.

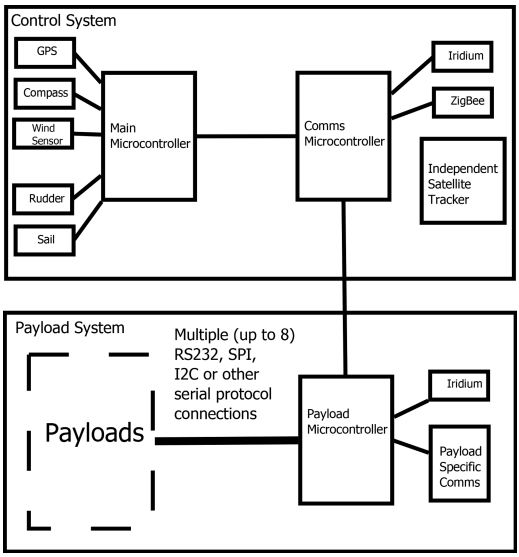


Fig. 6 A diagram showing the electrical systems

3.3 Design Improvements and the Full-Scale Concept

While the prototypes initial tests showed success, some improvements were noted for implementation in the full-scale vessel. These include:

- A retractable rudder instead of a transom hung rudder. This will keep the rudder from interfering with the towed array launch and recovery system.
- With the longer length the vessels stability will increase significantly (stability increases at a higher power than length), allowing for a higher

length/beam ratio (from 4 to 4.5). This will reduce resistance as well as weather and lee helm.
- The location of the maximum waterplane beam will move slightly further aft (about 5% of Load Waterline Length - LWL) to reduce spray and wave making drag. This also resulted in a narrower waterline forward which increased topside flare. This will improve spray deflection.
- Moving the keel slightly forward (about 50 mm on the 4.85 m vessel) will provide room for the keel lifting lead screw between the keel and mast.
- Increasing the keel chord about 25% will provide better low speed manoeuvrability at the expense of a little bit of friction drag.
- Making the leading edge of the keel flush with the bulb nose will improve weed and line shedding.
- Although the prototype consistently pointed up to 45 degrees from the wind, while under low-power autonomous control this will be increased to 50 degrees to avoid accidental tacking and minimize rudder movement.
- Using a toothed belt for halyards the sail will be hoisted using a long-throw bungee cord system and retracted using roller furling.
- Sheets will be trimmed using a long-throw servo arm rather than a winch to minimize the likelihood of tangling.

When the decision to build the prototype at two metres was made, the final size of the full-scale vessel was expected to fall within a range spanning three to six metres but the final payload details were unknown. The final length was determined from test feedback and the systems selected by the sponsors, resulting in an oceanographic payload capacity of 43 kg. The largest change was the addition of a towed-array style hydrophone. The hull form proposed for the full-scale autonomous research vessel was very similar to the hull form used for the demonstrator but scaled up to 4.85 metres and 380 kg. This weight budget includes 260 W of solar panels and two 45 W Forgen vertical axis wind turbines. The shallow draft with the keel retracted meant that launching from a trolley using a slipway or beach will still be viable with two personnel. A key decision was the choice to continue to use an electric propulsion motor as it allowed for multiple power generation sources. The primary power source proposed was an air-cooled 5 hp Cosworth diesel engine connected to a DC generator. Sustainable speeds of around 10 knots under power and 4 knots under sail are expected using the diesel-electric propulsion system and sailing rig proposed. Approximately 37 litres of fuel would be sufficient for the motoring expected. To cool the air-cooled engine the design calls for a heat-sink on the cylinder head that extends through the deck into a recessed well that is passively filled with seawater. The selection of the diesel generator provides an alternate power source at times when renewable power is insufficient.

3.4 Market Analysis

The potential for producing a fully functional system that meets the original requirements seems to be within reach using the approach taken here. The technologies required are now relatively well-proven and none of the components are particularly expensive. The cost of production will be relatively low and consists mainly of labour costs for integration and construction. A more difficult question concerns the size of the markets for such technologies: there are certainly two markets (the funders of this challenge) that see potential in this approach, but it is difficult to estimate the size of the markets before having a fully operational prototype to demonstrate. Some estimates of market size could be made based upon existing platform technologies and deployment systems, although the advantages offered by ARRTOO and differences in price and capability may significantly affect the size of the market share that ARRTOO could address. Existing alternatives include:

1. Ships of opportunity offer low-cost, long distance transects in an environment that offers unlimited power, good protection for instruments and data relay equipment (both from environmental threats and bio-fouling), and easy access for periodic maintenance when ships return to their home port. ARRTOO offers some of these advantages (relatively low-cost data for long transects) and the ability to automatically return to port periodically for maintenance, but suffers environmental and bio-fouling threats similar to those suffered by buoys.
2. ARGO floats offer very long-term deployment and very wide area coverage, but provide data from only single points and are extremely restricted in terms of the payloads that they can carry and the power available to run them.
3. Gliders have similar power and payload capacity constraints to ARGO floats, but offer the ability to gather data from long transects, although at speeds lower than many ocean currents. The low speeds attainable by gliders dramatically affect the locations and routes that can be tackled (in comparison to ARRTOO).
4. Wave gliders have been deployed in a range of scenarios and offer many of the same advantages as ARRTOO in long endurance mode (ability to out-pace many ocean currents, good power availability, flexible payload), but lack the ability to perform the rapid transits. Wave gliders seem to be a serious competitor for long endurance operation.
5. Drifting and anchored buoys have been the staple platforms for long endurance observation for higher power payloads for many years, and offer simplicity and robustness, but offer no capability for movement, and require regular expensive maintenance/recovery work using large dedicated ships. ARRTOO offers significant advantages in cost and data quality over both types of buoy.

The main technological challenges remaining in the development of the 4.85 metre ARRTOO are in the integration of payload and software elements rather than in the robot itself. In particular the "ground station", communications software and payload integration will be key to ARRTOO's viability as a product.

4 Conclusions

Small size, autonomous, hybrid sail and power robot vessels are feasible and offer an economical alternative to traditional long-endurance research vessels. There are three main innovations that have been brought together in this project:

- Hull design: it is the first time that this hull form been used in a robotic boat as a means to efficiently combine sail and power.
- Keel retraction: retractable keels and a range of actuation mechanisms have been relatively common in full size sailing craft, but retractable keels suitable for autonomous retraction and redeployment on small robotic craft is novel.
- Sailing to motoring transition: the ability for the vessel to move from sailing to motoring without manual intervention.

The hull design chosen scales down well to smaller vessels, so it is reasonable to expect that a 4.85 metre version will have similar performance (when scaled appropriately) and will meet the performance targets whilst retaining the flexibility of the original hull in terms of payload variability and sea-keeping performance. This is reflected in the performance seen in the trials performed and predicted using velocity prediction programs.

The use of the diesel engine with an alternator purely as a generator allows it to be run in a high-efficiency mode and also permits recharging the batteries if the renewable generation system has extended periods of low production. Work on the power budget for higher demand payloads and assessment of the deck area available for photovoltaic panels has indicated that the use of a wind turbine is required to increase power generation capacity. This in turn requires some deck area to be sacrificed for the addition of an arch upon which to mount it. The conclusion reached is that by scaling up the prototype and incorporating the design features identified in this project a 4.85 m robot will have sufficient payload capacity, sufficient power generation and performance capable of meeting the requirements of a long-endurance, autonomous oceanographic research vessel.

Acknowledgements. Funding for the project was provided by NERC and DSTL via the Technology Strategy Board under the SBRI LEMUSV_04. We would like to thank Dr. Mike Morabito of the U.S. Naval Academy who did a great job performing

the Savitsky motoring analysis within a very limited period of time. The fourth member of our team, Barry Thomas, invested numerous late hours and weekends helping to build ARRTOO and without his help the boat would not have been finished on time. We would also like to thank our industrial partners Cosworth, OSIL, BAE Systems and EADS.

References

1. Bachmayer, R., Leonard, N., Graver, J., Fiorelli, E., Bhatta, P., Paley, D.: Underwater gliders: Recent developments and future applications. In: Proceedings of the IEEE International Symposium on Underwater Technology (UT 2004), Tapei, Taiwan (2004)
2. Chapelle, H.: American Small Sailing Craft: Their Design, Development and Construction. W. W. Norton and Company, New York (1951)
3. Dunbabin, M., Grinham, A., Udy, J.: An autonomous surface vehicle for water quality monitoring. In: Proceedings of Australasian Conference on Robotics and Automation, ACRA (2009)
4. Manley, J., Willcox, S.: The wave glider: A persistent platform for ocean science. In: Proceedings of IEEE OCEANS (2010)
5. Miller, P., Hamlet, M., Rossman, J.: Continuous improvements to usna sailbots for inshore racing and offshore voyaging. In: Proceedings of the 5th International Robotic Sailing Conference, Cardiff, Wales, United Kingdom, pp. 49–60 (September 2012)
6. Sauze, C., Neal, M.: Moop: A miniature sailing robot platform. In: Proceedings of the International Robotic Sailing Conference 2011 (2011)

An Embedded Low-Power Control System for Autonomous Sailboats*

J. Cabrera-Gámez, A. Ramos de Miguel, A.C. Domínguez-Brito,
J.D. Hernández-Sosa, J. Isern-González, and E. Fernández-Perdomo

Abstract. This work presents a small and affordable autonomous sailboat platform designed to be transported and operated by one or two people without any special means. The sailboat is based on a RC One Meter class vessel equipped with a low power 8-bit microcontroller board and a set of navigation sensors (compass, GPS, wind vane, ...) and a 868 MHz RF module. It has been designed to serve as a low cost replicable testbed platform for research in autonomous sailing. The embedded control system makes the sailboat completely autonomous to sail a route determined as a sequence of waypoints, adapting its sailing point dynamically to wind conditions. The control system is completed with an off-board base station that permits to monitor and control the boat or defining a new route. The system is characterized by its long autonomy and robustness in case of communication failures.

1 The Sailboat

Autonomous sailboats have a large potential as high speed vehicles of virtually unlimited autonomy for environmental monitoring and sampling. Depending on their net displacement and dimensions, they can accept scientific payloads that maybe too large or power demanding to be integrated in other types of autonomous marine vehicles.

J. Cabrera-Gámez · A. Ramos de Miguel · A.C. Domínguez-Brito ·
J.D. Hernández-Sosa
Instituto Universitario de Sistemas Inteligentes y Aplicaciones Numéricas en Ingeniería (IUSIANI)
e-mail: jcabrera@iusiani.ulpgc.es

J. Cabrera-Gámez · A.C. Domínguez-Brito · J.D. Hernández-Sosa ·
J. Isern-González · E. Fernández-Perdomo
Dept. Informática y Sistemas, Universidad de Las Palmas de Gran Canaria, Spain

* This work has been partially funded by Canary Government and FEDER funds under ACIISI ProId2010/0062

F. Le Bars and L. Jaulin (eds.), *Robotic Sailing 2013*,
DOI: 10.1007/978-3-319-02276-5_6,

However, the development of autonomous sailboats is complex in terms of needed infrastructure and experimental costs. One way to overcome or reduce these limitations is to resort to a scaled down vessel, following a popular rule among small ship builders that states that the overall cost of a vessel goes proportional to the cube of its length.

In accordance with that vision, in this paper we present a small autonomous sailboat that has been based on a commercial RC boat and low cost or legacy off the shelf components. The motivation behind this approach has been to get an affordable open experimental platform which could serve as test bed for the development of navigation algorithms for sailboats.

1.1 The Vessel

The sailboat has been based on a carbon fiber One Meter class vessel with mainsail and foresail (LOA[2]: 100 cm; beam: 24.5 cm; draft: 14 cm; sail area: 0.61 m^2; displacement: 4.3 kg; mast height: 1.6m) and the first prototype has been named *ATIRMA, Autonomous TIRMA* after a memorable canary sailboat.

A ONE Meter Class sailboat was selected because it combines optimally good sailing capacities, cost, ample space under deck with easy access, extra payload capacity and dimensions that ease its operation and transport on a normal car.

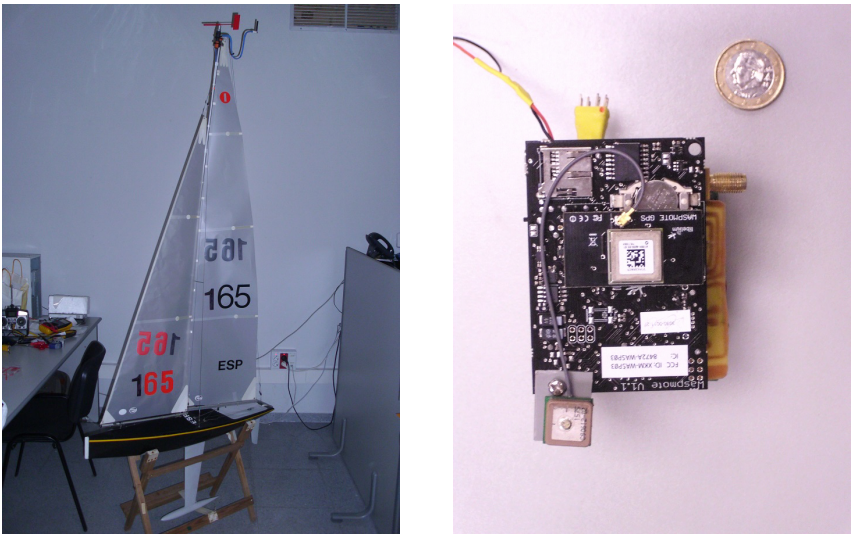


Fig. 1 The ATIRMA sailboat and the on board electronics

[2] Length Over All.

The sailboat employs two analog RC servos as actuators for the rudder and sail's sheet. They are powered from a six NiMH rechargeable AA cells (1.2V, 2700mAh) connected in series as a 7.4V battery pack with a capacity of approximately 20 Ah. It is equipped with a custom-made wind vane situated on top of the mast for sensing the apparent wind direction but not wind speed.

1.2 The Hardware

The vessel's electronics is made up of the following main components:

- An 8-bit microcontroller board
- An XBee PRO 868MHz RF module
- A GPS receiver
- Electronic compass with inclinometers
- Wind vane
- Current sensor

1.2.1 Microcontroller

The sailboat controller is a commercial credit-card size board based on a ATmega1281 microcontroller running at 8MHz. The microcontroller integrates 8KB of SRAM for data, 128 KB of FLASH memory for program and 4 KB EEPROM [3]. The board provides several UARTs, an I^2C bus, a micro SD card reader, a real-time clock, a three-axis accelerometer and several other sensors for measuring, for example, the board temperature or the battery level.

This board is ready to accept external hardware modules like a GSM/GPRS modem, a GPS receiver or different XBee RF communication modules. It has 5V and 3.3V on-board regulators. It is powered from a 3.7V 6000 mAh Li-Ion battery and consumes 9 mA under normal operating conditions. Suitable photovoltaic panels can be connected directly to the board to recharge the main battery.

1.2.2 RF Radio Module

For this prototype we have based all communications with the sailboat on XBee 868 Pro RF modules. These modules operate at the 868 MHz ISM band using only one channel. The bandwidth is 24 Kbps and the communications can be encrypted. The nominal range using a 4.5 dB dipolar antena in LOS conditions and free field is 40 km, but more realistic estimations are in the range of 10 km. It is possible to adapt the transmission power in five levels

till a maximum of 350 mW. It works at 3.3V and its current consumption is 500mA in transmission and 65mA in reception [1].

1.2.3 GPS Receiver

The board is prepared to accept an A1084 20 channel GPS receiver with an external antenna. This receiver is based on the SiRF III chipset and supports the NMEA0183 and SiRF binary serial protocols. We use the binary protocol to configure the receiver (elevation mask, signal strength mask, messages rates, ...) and rely on NMEA RMC and GGA messages for obtaining information about position, altitude, hdop, ground speed, course and time. It has an accuracy of less than 10 meters. It is powered by the on board 3.3V regulators and consumes 26mA [2].

Table 1 Power demands of system components

Component	Volt(V)	Current(mA)	Power(mW)
Microcontroller	3.3	9	29.7
GPS	3.3	26	85.8
XBee 868 PRO	3.3	65-500	330
TCM2-50	5	20	100
MA3 encoder	5	16	80
ACS712 board	5	7	35
		Electronics total	660.5
	Electronics battery	3.7V - 6000mAh	22200 mWh

Component	Volt(V)	Current(mA)	Power(mW)
Rudder servo	5	10-500	100
Sail winch	5	10-800	500
		Actuators total	600
	Actuators battery	7.4V - 2700 mAh	19980 mWh

1.2.4 Electronic Compass

The electronic compass is a legacy TCM2-50 board [4]. Basically, it provides tilt-compensated heading information and instantaneous pitch and roll angles over a RS232 interface. The board temperature and raw readings from three magnetometers can be also obtained. The compass readings are tilt and roll compensated till 50 degrees. The TCM2-50 can't operate for heeling or pitching angles over that limit.

This board is connected to one of the microcontroller TTL serial ports using a simple level converter circuit. The TCM2-50 has a maximum update rate of 20Hz. It is powered at 5V and consumes 20mA.

1.2.5 Wind Vane

The wind vane has been custom built from an Optimist wind vane attached to a US Digital's MA3 miniature absolute magnetic encoder [5]. The encoder is installed in an aluminum enclosure with a floating cap on top of the mast and connected to one of the analog inputs of the microcontroller. It allows to detect the direction of apparent wind but not its speed. It works at 5V and consumes 16mA.

1.2.6 Current Sensor

The current consumption at the actuator that controls the sails' sheet is measured by means of an ACS712 board [6]. The instantaneous current consumption is read as a voltage at a microcontroller's analog input. The ACS712 board integrates two potentiometers to adjust the intensity range being sensed and the acceptable levels of output voltage. This reading is used as an indirect measure of wind pressure in the sails. It is powered from 5V and consumes 7mA.

A summary of power demands of the main components of the system, along with the capacity of both batteries, is detailed in Table.1. The power consumption reflected in the table for the radio or the actuators are time averages based on laboratory and field measurements under normal operating conditions.

2 Control System

An external base station, a laptop equipped with a XBee USB adapter board, is used to communicate with the microcontroller on board the sailboat over the 868 MHz RF link. Both systems communicate regularly at a predefined but modifiable frequency. Using this radio link, the vessel can be monitored and controlled from the base station.

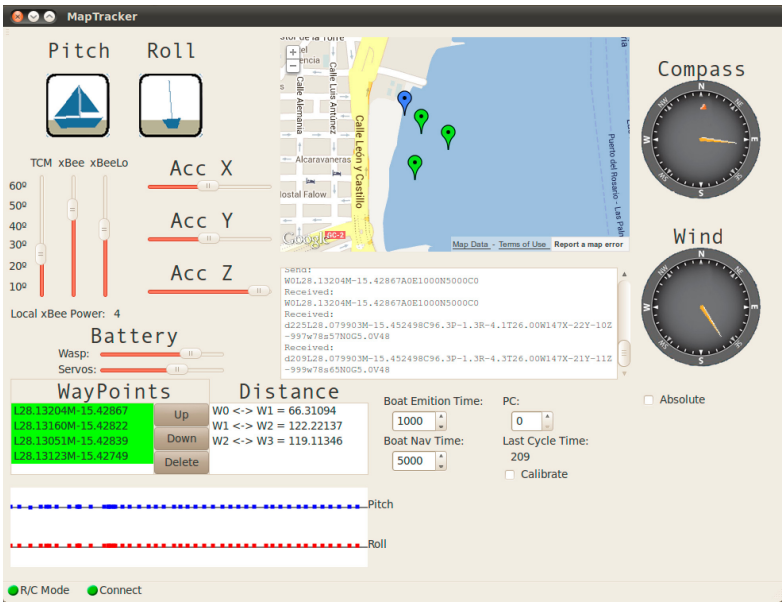


Fig. 2 Graphical user interface at base station

2.1 Software Architecture

The main software elements are the base station control application and the software that runs on the 8-bit microcontroller on board the sailboat. The base station is a Linux application with a Qt front end that relies on the libXBee [7] library to support the radio communications using the XBee radio modules. Using the graphical user interface (GUI) it is possible to add, edit or delete a sequence of waypoints, just by clicking on a Google map (see 2), to define a route for the sailboat.

The interface displays telemetry data received from the sailboat relative to sensor readings or position, bearing and speed of the sailboat. It is also possible to modify some thresholds and parameters like the frequency at which the telemetry packets are remitted or the minimum frequency at which the bearing selection function must be invoked.

2.1.1 Initialization

The initialization of the system is carried out normally with the vessel at shore, but could be done remotely as far as the radio link may reach. In this phase, the operational state of all onboard subsystems are verified and some sensors are calibrated, namely the wind vane and the inclinometers. The calibration steps can be omitted using the base station interface.

First radio communication, SD logging and battery levels are checked and afterwards on board regulators are switched on. Then the GPS receiver is configured to and the elevation and signal strength masks are configured in order to minimize noise in GPS readings. Once the GPS is configured, a first valid fix is awaited and then it will wait 20 seconds to stabilize the GPS measurements.

An optional final stage in initialization deals with the calibration of some sensors offsets. It requires to keep the sailboat in a horizontal position with 0 of pitch and roll and the wind vane pointing forward. This is done only once at the beginning of the experiment but can be avoided and previously calibrated offsets will be recovered from the EEPROM.

At the conclusion of this stage the sailboat will be in remote control mode and it will start sending telemetry data through the radio every 5 seconds by default.

2.1.2 Control Loop

Once the initialization has been accomplished, it will start the main control loop that has two possible modes of operation. In the autonomous mode the sailboat's control system selects the best bearing to arrive to the active waypoint. Alternatively, the remote or teleoperation mode permits to take full control of the sailboat from the base station. In both modes the telemetry is kept active.

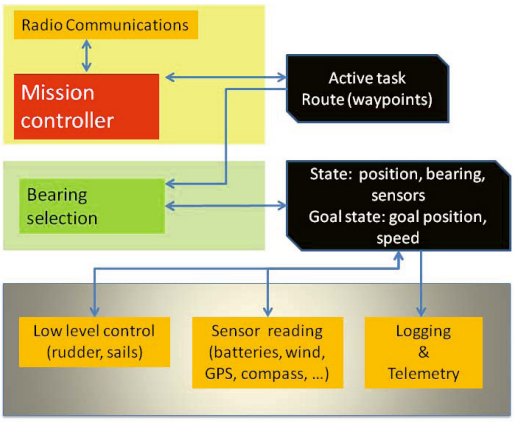


Fig. 3 On-board control architecture

2.1.3 Remote Control Mode

While the boat is in remote control mode it follows the sail and rudder position commands sent from the base station using a wireless game pad connected to the laptop running the base station application. In this mode, short radio packets are sent to the vessel at a frequency of 10 Hz. The transmission rate can't be too high because in this mode, the on board sensors are sampled, logged and telemetry packets are sent at the specified frequency to the base station.

2.1.4 Autonomous Mode

In the autonomous mode the navigation is fully under the control of the on board microcontroller. The control is organized around three levels of control. It is a layered architecture that shares with that presented in [12] a similar assignment of competences to some modules, although the one used here lacks a strategic long term routing module.

At the highest, the route controller simply manages the list of waypoints that define a route and selects the active waypoint. When the sailboat is inside the radius of precision of the waypoint, the route controller will change the active waypoint to the next one in the route. The list of waypoints is treated as a cyclic route by default, so when the last waypoint is reached, it will start again with the first one and the route is repeated.

At the next level, a bearing selection algorithm [10] is used to obtain the best (i.e. fastest) bearing to reach the active waypoint, given the current wind direction and boat position and heading. This control level is runs at an adjustable frequency but can be triggered also by a sudden wind roll. Note that as far we lack an estimation of wind speed on board, we run this algorithm using only the apparent wind.

At the lowest level, a fuzzy controller runs at the highest frequency of the control system and sets the sail and rudder positions to keep the sailboat under control on the bearing determined by the bearing selection algorithm. This controller is an adapted version of the controller described in [11], implemented using the EFLL library [8]. The main difference between the controller described in that paper and ours is that last one's outputs are absolute positions for sail and rudder while Stelzer et al. propose an incremental control system, i.e. outputs of the fuzzy control system are changes to the current settings.

The sailboat can transit into autonomous mode if a prolonged failure of radio communications is detected or because this control mode is explicitly commanded from the base station through a radio packet with the format $WxxLxxMxxAxxExx$. The preamble W identifies this packet as an autonomous mode command packet; the L and the M fields indicate the latitude and the longitude of the active waypoint; A indicates how far (in meters) can be the sailboat off the line that connects the current position of the boat

and the waypoint (it is equivalent to the PC parameter in [10]), finally, E indicates the emission period for telemetry messages.

When in this mode, the base station sends every 5 seconds short messages to verify the radio link. If these packets are not received at the vessel for 20 seconds, the active waypoint is deactivated and substituted by the coordinates that identify the "Home Point" and the sailboat will try to arrive to that point autonomously. This constitutes the Return To Home" or RTH behavior that has proved a valuable capability during field tests. This situation can be reverted from the base station as soon as the radio link is reestablished. In that moment, new waypoints and navigation parameters can be transmitted to the boat.

2.1.5 Robust Radio Connectivity

Loss of radio connectivity is something that may happen easily during sailing due to a variety of reasons and it is important to endow the sailboat with some recovery and continuity strategies to deal with these situations.In order to increase the robustness of radio communications on this uncertain scenario, the XBee radios are used in API mode and all exchanged messages have been limited in extension to make them fit within the payload of XBee API frames. Basically, this constraint reduces the complexity of recovering partially lost packets as all messages involve a single radio frame.

Accordingly, at the lowest level, the XBee radio modules have been programmed to resend automatically dropped or incorrect radio packets for a number of times. The loss of telemetry packets is not critical because they are still logged on the micro SD card available on board. More critical is the loss of command packets sent from the base station and these packets must be acknowledged explicitly from the sailboat. Otherwise, they are resent.

2.1.6 Sensor Sampling

Sensors are sampled at different rates depending on its potential rate of change and the temporal cost of a new reading in order to reduce the mean sampling time and hence, the duration of a control cycle.

The GPS sensor available on board has a maximum update rate of 1 Hz and it does not make sense to read it faster because it will deliver old estimates. Even, while reading at the nominal rate of 1 Hz, timestamps of new readings must be checked against the timestamp of the last delivered message to verify that it is indeed a new reading. If that is not the case, a new reading is attempted.

The compass board is programmed to produce a continuous flow of readings at a specific frequency (10 Hz approx.). This approach reduces the cost of reading from the TCM board. Each data packet contains the compass bearing, pitch and roll angles, the temperature of the board and, eventually, an

error code. Error codes appear normally when the magnetometers have become saturated or the pitch and roll angle limits have been exceeded. In those cases, these measurements are discarded. Compass packets may accumulate and overflow the microcontroller serial buffer if it is not read fast enough. This is not a problem because the serial buffer is circular and all messages but the last one are discarded. It is important to know that the GPS receiver and the compass are connected to two different serial ports that are in fact multiplexed on the same microcontroller's UART. This implies that it is not possible to receive continuously and simultaneously data from both devices.

The navigation is critically dependent on the adequate sampling rate of the set of on board sensors. With the limited computing power available and high temporal cost of sampling some sensors, a multi rate, smart sampling strategy is necessary. Sensors like the wind vane and the compass have a high update rate and the reading cost is very small. On the other side the GPS has a low update rate and interrogating the GPS receiver takes about 60 msecs. To deal with this situation, two strategies have been implemented. Firstly, fast sensors, and in particular the wind vane, are sampled several times within a single control cycle and filtered to produce better estimates of these magnitudes. Secondly, a Kalman filter is used to produce estimates of the position (lat, lon), orientation and speed. This filter helps to reduce the impact of some noisy GPS positions that may show up sporadically.

Sensors readings are monitored and they may trigger some alarms. For example, a sudden roll in wind direction over a predefined threshold will trigger the execution of the bearing selection algorithm during the next control loop. Also, battery readings are checked against low level thresholds and if low battery alarms are triggered they are notified to the base station.

All collected sensor data are packed and logged on board on a micro SD. A fraction of the logged packet is transmitted to the ground station as a telemetry packet at a predefined frequency. As commented previously, this frequency can be changed from the base station.

With the current hardware, the temporal cost of executing one control cycle is dominated by the temporal cost of the actions carried out during one control cycle. The subtasks that have the higher temporal cost are reading the GPS (60 msec), reading the compass (30 msec) and preparing and logging the telemetry packets (40 msec). Taking into account that some subtasks do not execute in every cycle, the shortest cycle time takes approximately 150 milliseconds and the largest 250 milliseconds.

3 Experiments

Several field trials have been carried out with the ATIRMA on the quiet waters of Las Palmas de Gran Canaria's port bay. An interesting achievement has been the potential power autonomy of the sailboat. We have carried out sailing tests over more that 8 hours in which the sailboat has been sailing

continuously and have registered the evolution of remaining capacity. We have never exhausted any of the batteries, even though the evolution of the capacity of each battery was dependent on the experimental conditions (wind intensity and frequency of communications) present during the tests. We plan to test the power autonomy in a close future extensively but currently our rough estimate is that 8 hours of operation under the parameters described in this paper consume approximately 20% of each battery capacity.

The foreseen autonomy exceeds of one day and it could be extended substantially adding supplementary batteries or with the installation of small and lightweight photovoltaic panels that the microcontroller is ready to accept. It must be noticed that with the current setup the servos are adjusted almost continuously and all navigation sensors are always on. If necessary, the implementation of some simple energy conservation strategies could reduce the power consumption even more and extent the autonomy significantly. For example,the winch controlling the sails is responsible for the largest part of the power consumed at the actuators side. A large amount of this power is wasted holding the position of the servo under the pressure of the wind and adjusting the sails to a new position. The substitution of this type of actuator by an actuator that could maintain the position without consuming power would have an appreciable impact in terms of energy conservation.

During these experiments the range of radio communications were tested using on board omni-directional dipole antennae of 4.5 and 0dBi gains. In all cases, the radio link was maintained over the full area of the bay (500 m approx.).

A video of one of the first trials at sea is available from URL [9]. During this video, both control modes, remote control and autonomous, have been exercised. While sailing in open water, away from swimmers or other vessels, the sailboat was on autonomous mode. Remote control was turned on occasionally when it was close to shore and/or bathers had to be avoided.

4 Discussion and Conclusions

This work has been motivated by the necessity of developing a small and affordable autonomous sailboat platform that could be transported and operated by one or two people without any special means. In agreement with those objectives, this paper has described the design of a low cost autonomous sailboat whose development has been based on a standard RC One Meter class vessel and off the shelf low power hardware components.

The main features of the described system are its flexibility as experimental platform, its large power autonomy and its robustness in case of communication failures. The amount of space and displacement available in a One Meter class sailboat severely restricts the volume and weight of the sensors and control electronics that can be installed on board. These restrictions have

an impact in terms computing power and number and type of the sensors that can be installed on board.

The main limitations of the control system described in this paper are its scarce computing power and reduced RAM memory. These restrictions have demanded a careful analysis and design of the control system to make it "fit" within the microcontroller memory and processing power, trying - at the same time - to reduce the span of a control cycle as much as possible.

The system described in this paper is very similar in scope to that described in [14], where it was presented a control system for autonomous sailboats based on a 50 MHz (64KB RAM) Cortex-M3 ARM7 processor board. The main difference between both systems, aside from the smaller computing power and memory of our system, is that our sailing control system is completely embedded on the on board processor, while in [14] the sailboat controller executes in a laptop outside of the boat.

Perhaps the most fragile element of the whole system is the wind vane, as noted by many others [15]. Wind vanes with movable mechanical parts are intrinsically prone to failure. Whilst commercial solutions exist for full scale sailboats, they are unpractical for a boat of small dimensions. Some solutions have been explored and described in the literature but a truly robust design is still to be achieved. An alternative design for a wind sensor (direction and intensity) has been described in [13].

5 Future Work

In the near term, future work will address the substitution of the GPS and compass board with more capable and up-to-date versions of these sensors and the incorporation of ultrasound sensors for aerial obstacle detection and avoidance. On the long term, we would foresee to replicate the ATIRMA and tackle the problem of route planning for cooperative surveying by a group of sailboats.

References

1. XBee 868 Pro specifications, http://www.digi.com/products/wireless-wired-embedded-solutions/zigbee-rf-modules/point-multipoint-rfmodules/xbee-pro-868#specs
2. A1084 GPS receiver hardware manual, http://ec-mobile.ru/user_files/File/Tyco/A1084_HM_V1.0.pdf
3. Libelium's Waspmote manual, http://www.libelium.com/v11-files/documentation/waspmote/waspmote-technical_guide_eng.pdf

4. PNI's legacy TCM2.5 electronic compass manual, http://www.pnicorp.com/download/347/99/TCM2.52.6Manualr09.pdf
5. US Digital absolute encoder MA3, http://www.usdigital.com/products/encoders/absolute/rotary/shaft/ma3
6. ACS712 product page, https://www.sparkfun.com/products/8883
7. LibXBee library, http://code.google.com/p/libxbee/
8. EFLL fuzzy logic library, https://github.com/zerokol/eFLL
9. ATIRMA video, http://www.youtube.com/watch?v=JoCVoFabJMg
10. Stelzer, R., Pröll, T.: Autonomous sailboat navigation for short course racing. Robotics and Autonomous Systems 56, 604–614 (2008)
11. Stelzer, R., Pröll, T., John, R.I.: Fuzzy Logic Control System for Autonomous Sailboats. In: FUZZ-IEEE 2007, pp. 97–102 (2007)
12. Stelzer, R., Jafarmadar, K.: A Layered System Architecture to Control an Autonomous Sailboat. In: Proceedings of TAROS 2007, Aberystwyth, UK (2007)
13. Alvira, M., Barton, T.: Small and Inexpensive Single-Board Computer for Autonomous Sailboat Control, Robotic Sailing 2012, pp. 105–116. Springer (2013)
14. Koch, M., Petersen, W.: Using ARM7 and uC/OS-II to Control an Autonomous Sailboat Robotic Sailing 2011, pp. 101–112. Springer (2012)
15. Neal, M., Sauze, C., Thomas, B., Alves, J.C.: Technologies for Autonomous Sailing: Wings and Wind Sensors. In: Proceedings of the 2nd IRSC, Matosinhos, Portugal, July 6-12, pp. 23–30 (2009)

Sailboat as a Windmill

Luc Jaulin and Fabrice Le Bars

Abstract. This paper proposes to transform a sailboat robot into a big wind turbine (or windmill) corresponding to the boat itself. The main idea is to make the sailboat rotating as fast as possible. When the wind open the sail, the mainsheet is able to pull a generator in order to produce electric energy. The resulting controller is simple to implement and its parameters are easy to tune. A simulated test-case shows that the proposed technique could generate an average power of approximatively 100W.

1 Introduction

Sailboat robots (see e.g. [17] [16] [6] [2] [3]) need energy for the actuators, for the sensors [20], for the embedded computer and for communication [21] [5]. Sonar panels cannot be considered as sufficient in many situations (during the night, or in cloudy areas) and we would like to consider other sources of energy that do not depend on the sun. A wind turbine or water turbine have sometimes been used, but the energy brought cannot be considered as significant [19]. In this paper, we propose to use the sailboat itself as a huge wind turbine, or equivalently to reconstruct a mobile windmill. The windmill behavior of the robot assumes the boat is in a station keeping mode. Such a mode can be chosen in case where the robot has to wait for a rendezvous, or when the robot has its batteries almost empty. We assume here that the robot has only two actuators: the rudder and a blocker for tuning the sail. The corresponding controller is illustrated by Figure 1, where u_1, u_2 correspond to the inputs (i.e., the rudder angle and the tuning of the sail) and $\mathbf{m}, \theta, \psi$ are the outputs (i.e., the GPS, the compass and the weather vane). If we consider that

Luc Jaulin · Fabrice Le Bars
IHSEV, Lab-STICC, OSM, Pôle STIC, ENSTA Bretagne, 2 Rue F. Verny, 29806 Brest, France
e-mail: {Luc.Jaulin,Fabrice.Le_Bars}@ensta-bretagne.fr

F. Le Bars and L. Jaulin (eds.), *Robotic Sailing 2013*,
DOI: 10.1007/978-3-319-02276-5_7,

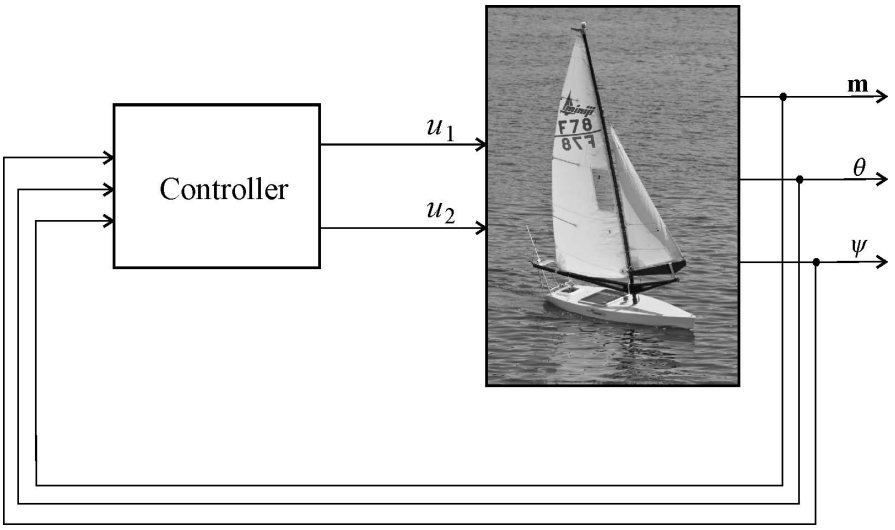


Fig. 1 The controller makes the robot rotating on itself as a windmill in order to produce energy

the blocker does not consume any energy, the only energy used for control is the rudder which consume less than 0.1W, if it is well balanced [21]. When the locker is open and the sail is opening pushed by the wind, the positive power delivered by the wind through the sail can be collected by a generator and stored inside batteries. A spring makes it possible to maintain the mainsheet tight, i.e., when the sail is in a flag mode the spring will rewind the mainsheet and close the sail. The power can be collected either at the mainsheet level via a winch or at the mastfoot. The purpose of this paper is to demonstrate the feasibility of the approach and to evaluate the amount of energy we could expect to collect with this technique.

The paper is decomposed as follows. Section 2 presents a model for the sailboat taking into account the energy and the blocker. Section 3 proposes a control strategy giving the robot windmill like behavior to produce energy. Some simulated experiments detailed on Section 4 show that it is possible to solve the station keeping problem [4] while collecting an average of 100W for the batteries.

2 State Space Model

Different types of models exists for sailboats [8], [9] [10]. To our knowledge, the most accurate one has been provided by Xiao and Jouffroy [23]. Here, to describe the dynamic of the sailboat robot, we propose a model that is

sufficiently accurate to illustrate the behavior of our controller and able to give an approximation of the energy that could be collected. Classically, a state space model for a robot has the form

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{u})$$

where $\mathbf{x}$ is the state vector and $\mathbf{u}$ is the input vector. Sometimes, it is more convenient to write this state equation under the form

$$\begin{cases} \dot{\mathbf{x}} = \mathbf{g}(\mathbf{x}, \mathbf{z}, \mathbf{u}) \\ \mathbf{z} = \mathbf{h}(\mathbf{x}, \mathbf{u}) \end{cases}$$

where

$$\mathbf{g}(\mathbf{x}, \mathbf{z}, \mathbf{u}) = \mathbf{g}(\mathbf{x}, \mathbf{h}(\mathbf{x}, \mathbf{u}), \mathbf{u}) = \mathbf{f}(\mathbf{x}, \mathbf{u}).$$

The vector $\mathbf{z}$ contains link variables which are intermediate variables that are used to shorten the equation. Link variables can correspond to forces, angles, ... and are often needed for the simulation to draw the robot and also to control that some feasibility state constraints are satisfied.

Model. The model is given by the following state space equations (see Figure 2).

$$\begin{cases} \text{(i)} & \dot{x} = & v\cos\theta + p_1 a\cos\psi \\ \text{(ii)} & \dot{y} = & v\sin\theta + p_1 a\sin\psi \\ \text{(iii)} & \dot{\theta} = & \omega \\ \text{(iv)} & \dot{v} = & \frac{f_s\sin\delta_s - f_r\sin u_1 - p_2 v^2}{p_9} \\ \text{(v)} & \dot{\omega} = & \frac{f_s(p_6 - p_7\cos\delta_s) - p_8 f_r\cos u_1 - p_3\omega v}{p_{10}} \\ \text{(vi)} & \dot{\ell} = & u_2 \text{ if } \gamma > 0 \\ \text{(vii)} & \dot{E} = & p_6 \ |f_s| \ u_2 \end{cases} \tag{1}$$

where the link variables are given by

$$\begin{cases} \text{(viii)} & \mathbf{w}_{\text{ap}} = & \begin{pmatrix} a\cos(\psi - \theta) - v \\ a\sin(\psi - \theta) \end{pmatrix} \\ \text{(ix)} & \psi_{\text{ap}} = & \text{atan2}(\mathbf{w}_{\text{ap}}) \\ \text{(x)} & a_{\text{ap}} = & \|\mathbf{w}_{\text{ap}}\| \\ \text{(xi)} & \gamma = & \cos\psi_{\text{ap}} + \cos\ell \\ \text{(xii)} & \ell = & |\delta_s| \text{ if } \gamma \leq 0 \\ \text{(xiii)} & \delta_s = & \begin{cases} -\tan^{-1}(\tan\psi_{\text{ap}}) & \text{if } \gamma \leq 0 \\ -\ell \ \text{sign}(\sin\psi_{\text{ap}}) & \text{otherwise} \end{cases} \\ \text{(xiv)} & f_s = & p_4 a_{\text{ap}}\sin(\delta_s - \psi_{\text{ap}}) \\ \text{(xv)} & f_r = & p_5 v\sin u_1 \end{cases}$$

This model is close to the models developed in [11], except that here, *(a)* we added the direction of the wind ψ and its amplitude a as parameters, *(b)* the control is not anymore the sail angle, but the length of the mainsheet, which is more realistic, *(c)* the speed of the robot is not considered as small

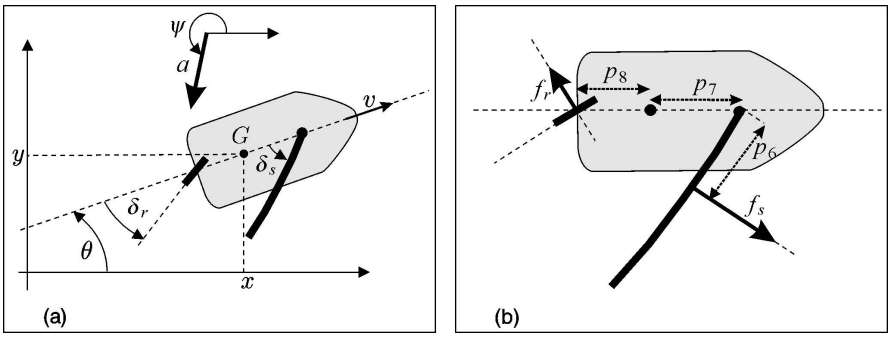


Fig. 2 Sailboat to be used as a windmill

compared to the true wind (the notion of apparent wind has thus to be introduced), *(d)* the angular friction now depends on the speed, which is more consistent with actual sailboats and *(e)* the length ℓ of the mainsheet and the energy of the batteries E are introduced as state variables. All quantities are expressed using the international unit system. For simplicity, the lenght ℓ of the mainsheet is expressed in radian (rad). ℓ corresponds to the absolute value of the maximal angle δ_s that could reach the sail when the mainsheet is tight. Let us now describe more deeply all variables involved in this model.

Inputs. The sailboat has two inputs. The first input $u_1 = \delta_r$ is the angle between the rudder and the sailboat. The second input u_2 corresponds to the blocker. When $u_2 = 1$, the locker is unblocked and the length of the mainsheet ℓ may increase (if the direction of the wind allows it). Otherwise, $u_2 = 0$ and the blocker is active.

State variables. The state variable occurring in our model (1) are $x, y, \theta, v, \omega, \ell, E$ where (x, y) are coordinates of the robot, θ is its heading, v is its speed along the main axis, ω is its rotational speed. The energy of the batteries E will increase with time. The length of the mainsheet ℓ corresponds of the maximal angle of the sail. In the particular case where the mainsheet is tight, it can be computed from the state variables θ, u_2, ψ, v and thus it cannot be considered as a state variable anymore. Therefore, the dimension of the state vector (either 6 or 7) changes with time. The sailboat thus corresponds to an hybrid system.

Parameters. In our model, p_1 is the drift coefficient, p_2 is the tangential friction, p_3 is the angular friction, p_4 is the sail lift, p_5 is the rudder lift, p_9 is the mass of the boat and p_{10} is the mass moment of inertia. The distances p_6, p_7, p_8 are represented in Figure 2. All parameters p_i are assumed to be known exactly. Two other quantities should also be considered as parameters: the speed a of the wind and its direction ψ.

Link variables. These variables are used to shorten the expression of the state equations. *(viii)* The vector $\mathbf{w}_{\text{ap}}$ corresponds to the apparent wind

expressed in the robot frame. The amplitude *(ix)* and the angle *(x)* of $\mathbf{w}_{\text{ap}}$ (in the robot frame) are denoted by a_{ap} and ψ_{ap}. *(xi)* The coefficient γ is positive if the mainsheet is tight. *(xii)* In this case, ℓ is a state variable and its evolution obeys to the differential equation $\dot{\ell} = u_2$. Otherwise, the mainsheet is tight, ℓ is a link variable and its value is equal to $|\delta_s|$. This change of status of ℓ is typical of what happen for hybrid systems. *(xiii)* When the mainsheet is not tight, the angle of the sail δ_s, is equal to $-\psi_{\text{ap}} \pm 2k\pi$ and it behaves as a flag. Since we want $\delta_s \in \left[-\frac{\pi}{2}, \frac{\pi}{2}\right]$, we have written $\delta_s = -\tan^{-1}(\tan(\psi_{\text{ap}}))$. When the mainsheet is not tight, δ_s is determined by ℓ and the direction of the apparent wind. *(xiv)* f_s represents the force of the wind on the sail and *(xv)* f_r is the force of the water on the rudder.

State equations. The two first equations *(i),(ii)* of (1) express that the boat follows its heading, but always looses with respect to the wind. Equations *(iv)* and *(v)* are obtained using the Newton laws. Equation *(vi)* tells us that the length ℓ of the mainsheet can only increases when the sail is inflated and when the blocker is off (i.e., $u_2 = 1$). Equation *(vii)* provides the power delivered to the batteries: when $u_2 = 1$, and $f_s \neq 0$, the sail opens with an angular velocity of 1rad.sec^{-1} and the power collected is $p_6 \ |f_s|$. Figure 3 represents the differential graph of the state equations. The state variables are represented by grey nodes and the inputs by square nodes. The integral relations are represented by bold arrows and the link relation by dotted arrows. The two bold dotted arrows illustrate that the differential dependency between u_2, γ and ℓ are valid for some conditions only.

Note that this model for the sailboat could be made more realistic by adapting the modeling tools described by Fossen in the context of marine vessel [7] to sailboats. But to our knowledge, realistic state equations for sailboats do not exist yet.

3 Controller

A classical approach to build controllers is to take a realistic model of the system to be controlled (such as [8] for the control of sailboats) and then to use classical control methods to get the controller. Here, we follow a pragmatic approach influenced by the potential field strategy proposed by [18] for sailboat robots. Our sailboat robot is assumed to have three sensors and two actuators. The controller will have some parameters which are easy to tune, two state variables $q \in \{1, 2, 3, 4\}$, and $t_0 \in \mathbb{R}^+$, two outputs $u_1 \in [-\frac{\pi}{4}, \frac{\pi}{4}]$, $u_2 \in \{0, 1\}$ and three inputs $\mathbf{m} \in \mathbb{R}^2, \theta \in \mathbb{R}, \psi \in \mathbb{R}$. Let us now describe all these variables.

Sensors (which also correspond to the input of the controller). The heading θ of the robot is measured by a compass. The angle of the wind ψ is returned by a weather vane (even if this sensor can sometimes be omitted

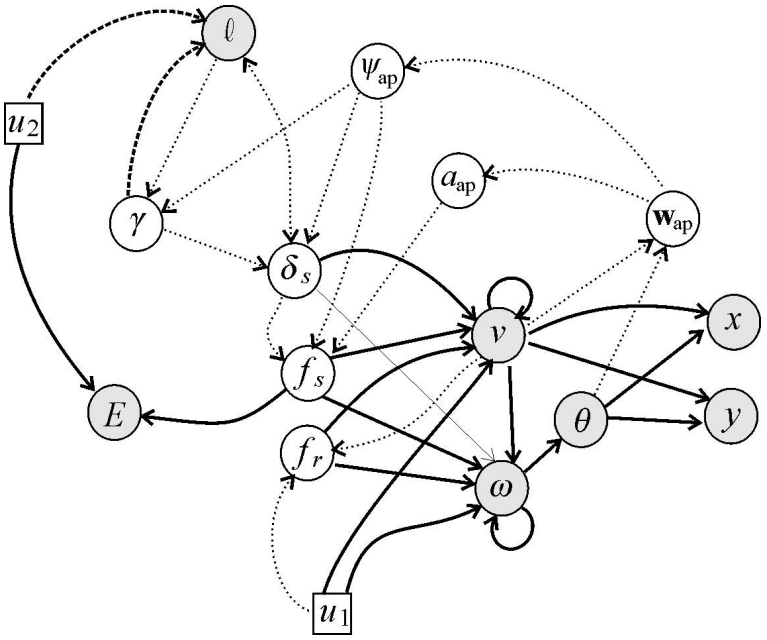


Fig. 3 Differential graph of our sailboat robot

as shown in [22]). The position $\mathbf{m}$ is given by a GPS. These sensors are the inputs of our controller.

Actuators (which also correspond to the output of the controller). The inputs of the robot are the angle $u_1 \in [-\frac{\pi}{4}, \frac{\pi}{4}]$ of the rudder and the blocker $u_2 \in \{0, 1\}$ which makes it possible to tune indirectly the length of the mainsheet.

Parameters. $\delta_r^{\max}$ is the maximal angle of the rudder is taken as $\delta_r^{\max} = \frac{\pi}{4}$. ζ is the close hauled angle. For the simulation, we will choose $\zeta = 1$rad.

State variable. The discrete variable $q \in \{1, 2, 3, 4\}$, gives three modes: the *traction* ($q = 1$), the *rewind* ($q = 2$) and the *positioning* ($q = 3, q = 4$). The start time t_0 corresponds to the time at which the timer is started when the controller is in the positioning mode.

The basic idea of the controller is to decompose the plane into three cones, the intersection of which corresponds to the origin, as illustrated by Figure 4. In the *mill cone* (painted gray), the robot rotates as a windmill to produce energy. In the mill cone, the robot slowly moves downwind. The points $\mathbf{m}$ that are inside the mill cone satisfy the inequality $\cos(\psi - \arg(\mathbf{m})) > -\cos\zeta$. In the hatched cone, the controller will carry favor to the close-hauled heading $\pi + \psi - \zeta$. In the white cone, it will prefer the heading $\pi + \psi + \zeta$.

We now give the details of the controller which is clearly influenced by the line following controller proposed in [14] [12] and already experimented [15] on the sailboat robot *VAIMOS*.

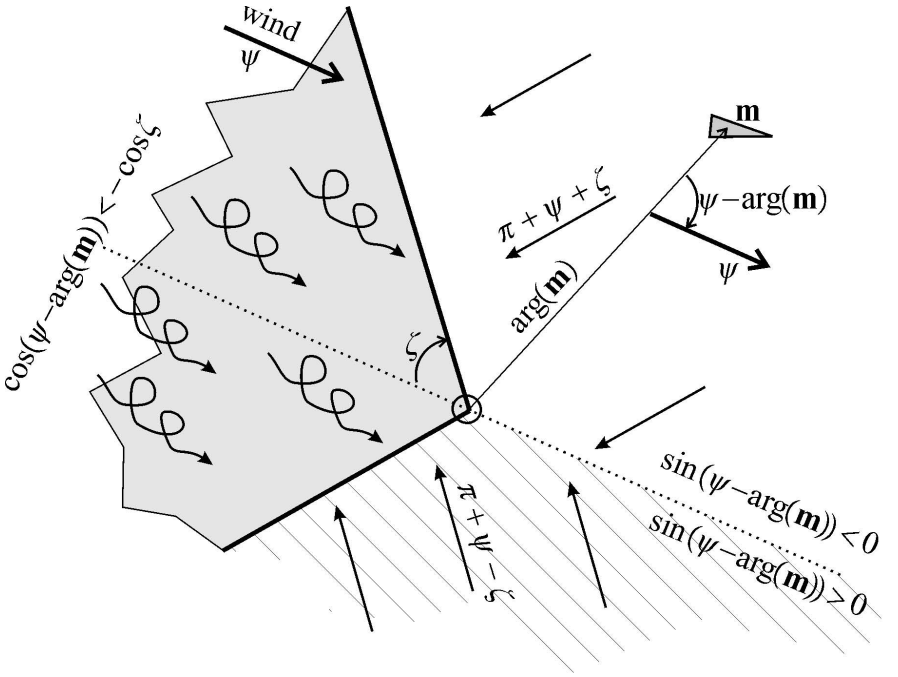


Fig. 4 Principle of the controller to maintain the robot around the origin still trying to spin as a mill in order to charge the batteries

Function in: $\mathbf{m}, \theta, \psi$; out: u_1, u_2; inout: q, t_0
1 if ($q = 1$ and $\psi \sim \theta$) then $q = 2$;
2 if ($q = 2$ and $\psi \sim \theta + \pi$) then
3 if $\cos(\psi - \arg(\mathbf{m})) > -\cos\zeta$
4 $t_0 = t$
5 if ($\sin(\psi - \arg(\mathbf{m})) > 0$) then $q = 3$; else $q = 4$
6 else $q = 1$
7 if ($q \in \{3, 4\}$ and $t - t_0 > 30$) then $q = 1$;
8 if $q = 1$ then $\bar{\theta} = \psi$
9 if $q = \{2, 4\}$ then $\bar{\theta} = \pi + \psi + \zeta$
10 if $q = 3$ then $\bar{\theta} = \pi + \psi - \zeta$
11 if ($\cos\left(\theta - \bar{\theta}\right) \leq 0$ or $q \leq 2$)
12 then $u_1 = \frac{\pi}{4}.\text{sign}\left(\sin\left(\theta - \bar{\theta}\right)\right)$
13 else $u_1 = \frac{\pi}{4}.\sin\left(\theta - \bar{\theta}\right)$
14 $\bar{\ell} = \frac{\pi}{2}.\left(\frac{\cos\left(\psi - \bar{\theta}\right)+1}{2}\right)$.
15 if $\bar{\ell} > \ell$ then $u_2 = 1$ else $u_2 = 0$.

Steps 1 to 7 correspond to the discrete event part of our hybrid controller. It is illustrated by the Petri net of Figure 5. The gray places represent actual states and white places represent fake states (the token leaves a fake place as soon as it enters in it). Bold arrows have a higher priority and are necessary to make the Petri net deterministic. Let us now describe the different discrete states for q.

- *Traction* ($q = 1$). The controller opens the sail and maneuvers to go downwind (see Step 8) as fast as possible (see Steps 11, 12). The controller escapes the state $q = 1$ at Step 1 as soon as $\psi \sim \theta$ (i.e., $\psi = \theta \pm 2k\pi$). When $q = 1$, the mainsheet pulls the generator and energy is produced.
- *Rewind* ($q = 2$). The controller makes the boat rotating toward the wind, in order to close the sail. When the robot is upwind ($\psi \sim \theta + \pi$), then the rewind mode terminates (see Step 2). If the robot is inside the mill cone, the controller goes to the state $q = 1$ at Step 6. Otherwise, depending on which cone the robot is, the controller chooses the states $q = 3$ or $q = 4$ at Step 5.
- *Positioning* ($q \in \{3, 4\}$). The controller chooses a close-hauled heading for 30 second, in order to bring closer to the mill cone.

Steps 8 to 10 provide the desired heading $\bar{\theta}$ to follow, depending of the value of q. When $q = 1$, the controller asks to go downwind (Step 8), Otherwise, it ask to go to a close hauled mode (Steps 9,10). Steps 11,12,13 tune the rudder (see [14] for more explanations). If the boat goes toward the wrong direction ($\cos\left(\theta - \bar{\theta}\right) \leq 0$) or if $q \in \{1, 2\}$, the rudder at its maximum, i.e., $\pm\frac{\pi}{4}$. otherwise, a proportional control is proposed (Step 13). For tuning of the sail, we propose to take a Cardioid model [13] at Step 14 to compute the right angle for the sail. At Step 15, the controller suggests to open the sail ($u_2 = 1$) in order to reach the desired length $\bar{\ell}$, by opening the blocker. This will mainly happen when $q = 1$.

4 Test-Case

In order to illustrate the principle of the controller, we now propose a simulation of the controlled sailboat robot. For the parameters, we have chosen

$$p_1 = 0.05,\ p_2 = 0.2\ \mathrm{kg \cdot s^{-1}},\ p_3 = 6000\ \mathrm{Kg \cdot m},$$
$$p_4 = 1000\ \mathrm{kg \cdot s^{-1}},\ p_5 = 2000\ \mathrm{kg \cdot s^{-1}},$$
$$p_6 = 1\mathrm{m},\ p_7 = 1\mathrm{m},\ p_8 = 2\mathrm{m},\ p_9 = 300\ \mathrm{kg},\ p_{10} = 10000\ \mathrm{Kg \cdot m^2}.$$

Except for p_6, these values correspond approximately to the coefficients of the sailboat robot *VAIMOS* [14]. The value for p_6 is almost zero for *VAIMOS*, due to the balestron rig (or balanced rig). For our application, it is important

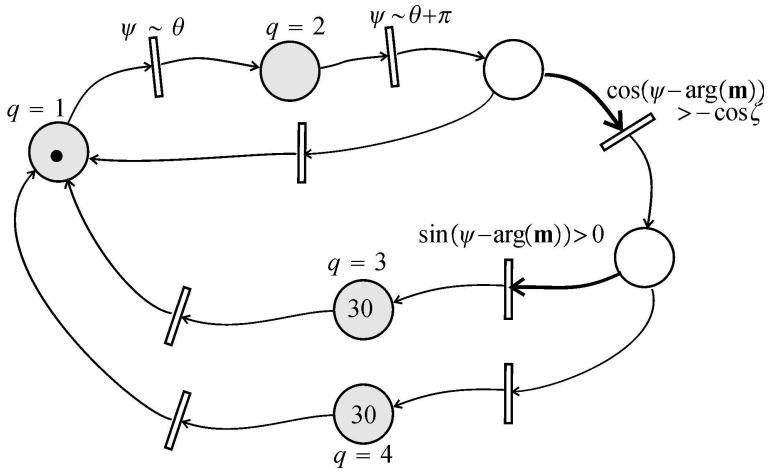


Fig. 5 Petri net associated with our windmill sailboat

to have an important p_6 for the energy production. For the speed a of the wind and its direction ψ, we took.

$$a = 4 \text{ m} \cdot \text{s}^{-1}, \ \ \psi = \pi.$$

For the parameter of the controller, we have chosen $\zeta = 1$ rad. The resulting trajectory is illustrated by Figure 6 where the robot is initialized at $\mathbf{m} = (-400, 200)$ (small black disk). The trajectory corresponds to 30 minutes and the average of the collected power is around 93W. On the picture, we clearly see that on the mill cone, the boat rotates and move downwind. As soon as it goes outside the cone, it comes back to the mill cone choosing the right close hauled heading. The executable program and the C++ source code of the simulator that has generated Figure 6 can be found at

`http://www.ensta-bretagne.fr/jaulin/mill.html`

Remark. Betz's law [1] claims that the maximum power that can be extracted from the wind, independent of the design of a wind turbine, is given by

$$P_{\text{Betz}} = \frac{8}{27} \rho S a^3,$$

where S is the surface of the turbine, ρ is the fluid density and a is the speed of the wind. From this formula, we can deduce that to get the same power than that collected by the batteries in our test-case, a wind turbine with a diameter of 2.4 meters would be needed. Of course, such a wind turbine would change significantly the dynamic performances of the sailboat.

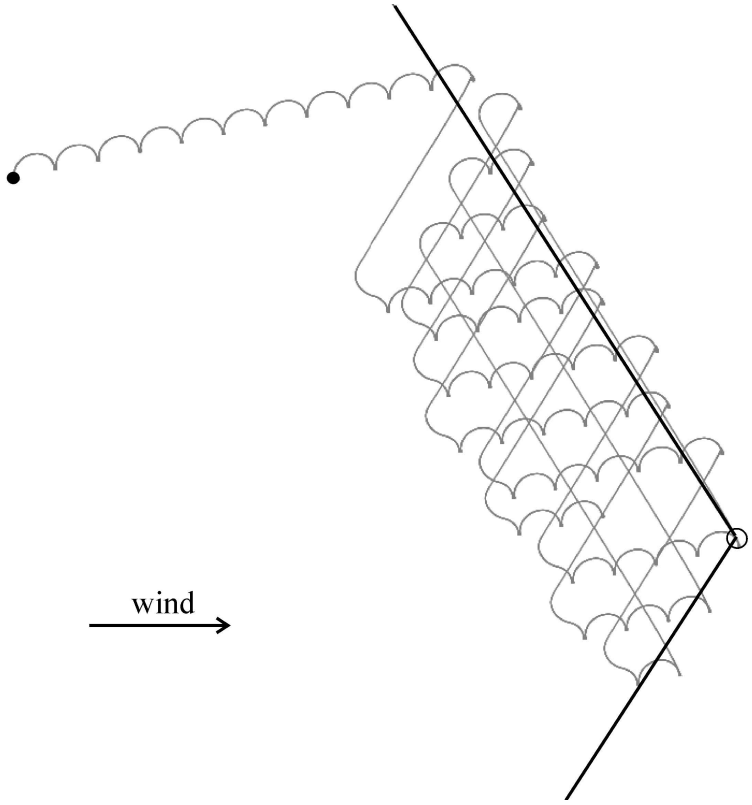


Fig. 6 Trajectory of the sailboat robot which tries to remain inside the circle and also to collect energy from the wind

5 Conclusion

In this paper we have presented a new concept that allows a sailboat to take advantage of a station keeping mode in order to charge its batteries. The basic idea is to transform the sailing boat into a windmill using a hybrid controller. When the wind inflates the sail, the mainsheet pull a generator that produces energy for the batteries. A test-case has shown that a mean power of 93W could be collected for a wind speed equal to 4 $\mathrm{m{\cdot}s^{-1}}$. All computations made by the controller can be performed using any cheap and low-powered microcontroller.

Acknowledgements. The robot *VAIMOS* is the result of a collaboration between LPO (Laboratoire de Physique des Océans), RDT (Recherches et Développements Technologiques) of Ifremer (Institut Français de Recherche pour l'Exploitation de la

Mer) and ENSTA Bretagne (Ecole Nationale Supérieure de Techniques Avancées). The authors render thanks to all people involved in the project: Y. Auffret, S. Barbot, L. Dussud, B. Forest, E. Menut, S. Prigent, L. Quemeneur, P. Rousseaux (RDT, Ifremer); F. Gaillard, T. Gorgues, O. Ménage, J. Moranges, T. Terre (LPO) and B. Clément, Y. Gallou, O. Reynet, J. Sliwka and B. Zerr (ENSTA Bretagne).

References

1. Betz, A.: Introduction to the Theory of Flow Machines. Pergamon Press, Oxford (1966)
2. Brière, Y.: The first microtransat challenge. In: ENSICA (2006), http://web.ensica.fr/microtransat
3. Cruz, N., Alves, J.: Ocean sampling and surveillance using autonomous sailboats. In: International Robotic Sailing Conference, Austria (2008)
4. Elkaim, G., Kelbley, R.: Station Keeping and Segmented Trajectory Control of a Wind-Propelled Autonomous Catamaran. In: Proceedings of the 45th IEEE Conference on Decision and Control, San Diego, USA (2006)
5. Elkaim, G., Lee Boyce Jr., C.: An Energy Scavenging Autonomous Surface Vehicle for Littoral Surveillance. In: ION Global Navigation Satellite Systems Conference (2008)
6. Erckens, H., Büsser, G., Pradalier, C., Siegwart, R.: Navigation Strategy and Trajectory Following Controller for an Autonomous Sailing Vessel. IEEE RAM 17, 47–54 (2010)
7. Fossen, T.: Guidance and Control of Ocean Vehicles. Wiley, New York (1995)
8. Gale, T., Walls, J.: Development of a sailing dinghy simulator. Simulation 74(3), 167–179 (2000)
9. Guillou, G.: Architecture multi-agents pour le pilotage automatique des voiliers de compétition et extensions algébriques des réseaux de petri. PhD dissertation, Université de Bretagne, Brest, France (2011)
10. Jaulin, L.: Modélisation et commande d'un bateau à voile. In: CIFA2004, (Conférence Internationale Francophone d'Automatique), CDROM. Douz (Tunisie) (2004)
11. Jaulin, L.: Représentation d'état pour la modélisation et la commande des systèmes (Coll. Automatique de base). Hermès, London (2005)
12. Jaulin, L., Le Bars, F.: A simple controller for line following of sailboats. In: 5th International Robotic Sailing Conference, pp. 107–119. Springer, Cardiff (2012)
13. Jaulin, L., Le Bars, F.: An interval approach for stability analysis; Application to sailboat robotics. IEEE Transaction on Robotics 27(5) (2012)
14. Jaulin, L., Le Bars, F., Clément, B., Gallou, Y., Ménage, O., Reynet, O., Sliwka, J., Zerr, B.: Suivi de route pour un robot voilier. In: CIFA 2012, Grenoble, France (2012)
15. Le Bars, F., Jaulin, L.: An experimental validation of a robust controller with the VAIMOS autonomous sailboat. In: 5th International Robotic Sailing Conference, pp. 74–84. Springer, Cardiff (2012)
16. Miller, P.H., Hamlet, M., Rossman, J.: Continuous improvements to USNA sailbots for inshore racing. In: 5th International Robotic Sailing Conference, pp. 49–60. Springer, Cardiff (2012)

17. Neumann, T., Schlaefer, A.: Feasibility of basic visual navigation for small sailboats. In: 5th International Robotic Sailing Conference, pp. 13–22. Springer, Cardiff (2012)
18. Petres, C., Ramirez, M.R., Plumet, F.: Reactive path planning for autonomous sailboat. In: IEEE International Conference on Advanced Robotics, pp. 1–6 (2011)
19. Rynne, P., von Ellenrieder, K.: Unmanned autonomous sailing: Current status and future role in sustained ocean observations. MTS Journal 43(1), 21–30 (2009)
20. Sauze, C., Neal, M.: An autonomous sailing robot for ocean observation. In: Proceedings of TAROS 2006, Guildford, UK, pp. 190–197 (2006)
21. Stelzer, R., Dalmau, D.E.: A study on potential energy savings by the use of a balanced rig on a robotic sailing boat. In: 5th International Robotic Sailing Conference, pp. 89–93. Springer, Cardiff (2012)
22. Xiao, K., Sliwka, J., Jaulin, L.: A wind-independent control strategy for autonomous sailboats based on voronoi diagram. In: CLAWAR 2011 (best paper award), Paris (2011)
23. Xiao, L., Jouffroy, J.: Modeling and nonlinear heading control of sailing yachts. IEEE Journal of Oceanic Engineering (2013)

Part III
Modeling, Simulation, Control, and Stability Analysis

Modeling and Control Design of a Robotic Sailboat

Hadi Saoud, Minh-Duc Hua, Frédéric Plumet, and Faïz Ben Amar

Abstract. This paper presents a method to obtain a full 6 degrees of freedom dynamic model of a robotic sailing boat starting from the description of forces and torques acting on it. A general 6-DOF model is first described and then simplified to obtain a 3-DOF control-oriented one. Relying on it, a rudder controller and a sail's trimming calculator are proposed. This controller has been validated using a numerical implementation of the proposed dynamic model.

1 Introduction

Thanks to their low energy consumption, autonomous sailing robots provide a promising solution for long-term missions and semi-persistent presence in the oceans and a lot of sailing robot projects have been launched recently all around the world [1, 3, 3, 4, 6, 8, 10, 12]. However, the nature of sailing boats implies restrictions on their navigation capabilities: thrust forces depend on uncontrollable and partially unpredictable wind. Furthermore, such vehicles exhibit complex behavior due to aero and hydrodynamic properties of sails and hull. Therefore, in order to improve the control of sailboats, a model reflecting the dynamic of the system and its relation with physical phenomenon (wind speed and marine current) is necessary.

One of the contributions of the present paper is the proposition of a simple but representative model for control design and also for the validation of the proposed controllers via simulation means.

The paper is organized as follow: first, the sailboat is divided into three subsystems (hull, mainsail and rudder) and the forces and torques acting

Hadi Saoud · Minh-Duc Hua · Frédéric Plumet · Faïz Ben Amar
Institut des Systèmes Intelligents et de Robotique (ISIR),
UPMC Université Paris 6, CNRS-UMR 7222, France
e-mail: {saoud,hua,plumet,amar}@isir.upmc.fr

F. Le Bars and L. Jaulin (eds.), *Robotic Sailing 2013*,
DOI: 10.1007/978-3-319-02276-5_8,

on each subsystem are described. Then, a general 6-DOF similar to [5] is proposed. Some assumptions are made to simplify the model to a 4-DOF one with roll and yaw as in [16]. For control design, a more simplified 3-DOF model without roll is proposed. Analysis of this model leads to the definition of a new heading controller using backstepping control method [7] with variable gains and a sail's trimming algorithm, both presented in the last section of this paper.

A numerical implementation of the 3-DOF model and controllers is done and some simulation results are presented.

2 Notation

- For any $x \in \mathbb{R}^{m\times n}$, $x^\top$ denotes the transpose of x and $\dot{x}$ its time-derivative.
- $\overrightarrow{x}$ denotes an affine vector associated with the vector space $x \in \mathbb{R}^3$.
- The scalar product of two vectors $\overrightarrow{x}$ and $\overrightarrow{y}$ is denoted as $\overrightarrow{x}.\overrightarrow{y}$, and their cross product as $\overrightarrow{x} \times \overrightarrow{y}$.
- $\{e_1, e_2, e_3\}$ denotes the canonical basis of $\mathbb{R}^3$. The Euclidean norm is denoted as $|\cdot|$.
- For any $x \in \mathbb{R}^3$, the notation $x_\times$ denotes the skew-symmetric matrix associated with x, i.e. $x_\times y = x \times y$, $\forall y \in \mathbb{R}^3$.
- For any affine vector $\overrightarrow{x}$ and any frame $\mathcal{X}$, $x^{\mathcal{X}}$ denotes the vector of coordinates of $\overrightarrow{x}$ in the basis of the frame $\mathcal{X}$.
- $R^{\mathcal{Y}}_{\mathcal{X}} \in SO(3)$ denotes the rotation matrix representing the orientation of the frame $\mathcal{X}$ with respect to (w.r.t.) the frame $\mathcal{Y}$. For any affine vector $\overrightarrow{x}$, one verifies $x^{\mathcal{Y}} = R^{\mathcal{Y}}_{\mathcal{X}} x^{\mathcal{X}}$.
- G: sailboat's center of mass (CoM).
- G_s, G_r, G_k: center of pressure of the sail, the rudder and the keel, all assumed to be fixed.
- $\mathcal{I} = \{0; \overrightarrow{\imath}_0, \overrightarrow{\jmath}_0, \overrightarrow{k}_0\}$: Inertial frame chosen as the North-West-Up frame.
- $\mathcal{B} = \{G; \overrightarrow{\imath}, \overrightarrow{\jmath}, \overrightarrow{k}\}$: Body frame fixed to the hull.
- $\mathcal{S} = \{G_s; \overrightarrow{\imath}_s, \overrightarrow{\jmath}_s, \overrightarrow{k}_s\}$: Sail-fixed frame.
- $\mathcal{R} = \{G_r; \overrightarrow{\imath}_r, \overrightarrow{\jmath}_r, \overrightarrow{k}_r\}$: Rudder-fixed frame, with $\overrightarrow{k} \equiv \overrightarrow{k}_s \equiv \overrightarrow{k}_r$.
- $m_0 \in \mathbb{R}$, $J_0 \in \mathbb{R}^{3\times 3}$: sailboat's mass and inertia matrix.
- $\overrightarrow{x}$, $\overrightarrow{x}_s$, $\overrightarrow{x}_r$: position of G, G_s and G_r w.r.t. the inertial frame.
- $\overrightarrow{\omega}$: angular velocity of the body-fixed frame w.r.t. the inertial frame.
- $\overrightarrow{v}$: linear velocity of G w.r.t. the inertial frame.
- $\overrightarrow{v}_s$, $\overrightarrow{v}_r$, $\overrightarrow{v}_k$: linear velocity of G_s, G_r and G_k w.r.t. the inertial frame.
- $\overrightarrow{v}_w$: wind velocity w.r.t. the inertial frame.
- $\overrightarrow{v}_c$: water current velocity w.r.t. the inertial frame.
- $\overrightarrow{v}_{as}$, $\overrightarrow{v}_{ar}$, $\overrightarrow{v}_{ak}$: apparent velocity of the sail, the rudder and the keel.
- x, R, ω, v: short notation of $x^{\mathcal{I}}$, $R^{\mathcal{I}}_{\mathcal{B}}$, $\omega^{\mathcal{B}}$, $v^{\mathcal{B}}$, respectively.
- $\nu := [v, \omega]^\top$.

- δ_s, δ_r: sail's angle and rudder's angle, respectively.
- α_s: sail's angle of attack.
- α_r: rudder's angle of attack.
- α_k: keel's angle of attack.
- S_s, S_r, S_k: surface of the sail, rudder and keel, respectively.
- ρ_{air}, ρ_{water}: air and water density, respectively.
- $C_s^L(\cdot)$, $C_s^D(\cdot)$: lift and drag coefficients of the sail, respectively.
- $C_r^L(\cdot)$, $C_r^D(\cdot)$: lift and drag coefficients of the rudder, respectively.
- $C_k^L(\cdot)$, $C_k^D(\cdot)$: lift and drag coefficients of the keel, respectively.

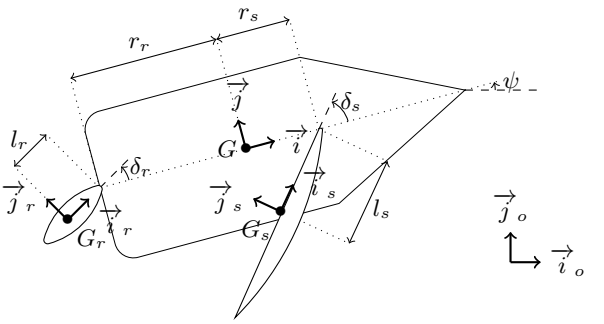


Fig. 1 Notation

3 System Modeling

3.1 Forces And Torques Acting on the System

The sailboat under consideration can be divided into three rigid parts: a sail, a rudder, and a main body composed of a keel and a hull. In what follows, the modeling of forces and torques acting on the sailboat is presented.

3.1.1 Aerodynamic Forces and Torques Acting on the Sail

Interactions between a rigid body and the surrounding fluid are governed by the Navier–Stocks equations. In the first approximation, the aerodynamic forces $\overrightarrow{F}_s$ acting on the sail can be expressed by a function dependent upon the constant air density ρ_{air}, the Reynolds number R_e and the angle of attack α_s. The latter variable is the angle between the apparent velocity $\overrightarrow{v}_{as}$ of the sail (defined as the difference between the velocity of G_s and the wind

velocity $\overrightarrow{v}_w$, i.e. $\overrightarrow{v}_{as} := \overrightarrow{v}_s - \overrightarrow{v}_w$) and the zero-lift line which coincides with the orthogonal projection of $\overrightarrow{v}_{as}$ onto the sail's plane.

First, let us derive the expression of the velocity $\overrightarrow{v}_s$ of G_s. Using the relations $\overrightarrow{GG_s} = r_s \overrightarrow{\imath} - l_s \overrightarrow{\imath}_s + h_s \overrightarrow{k}$ and $\overrightarrow{x}_s = \overrightarrow{x} + \overrightarrow{GG_s}$, one obtains the following relation expressed in the inertial frame:

$$x_s^{\mathcal{I}} = x + R(r_s e_1 + h_s e_3 - l_s R_{\mathcal{S}}^{\mathcal{B}} e_1)$$

Differentiating both sides of the above equation w.r.t. time, one gets:

$$\begin{aligned} & Rv_s^{\mathcal{B}} = Rv + R\omega_\times (r_s e_1 + h_s e_3 - l_s R_{\mathcal{S}}^{\mathcal{B}} e_1) - l_s \dot{\delta}_s R R_{\mathcal{S}}^{\mathcal{B}} e_3 \times e_1 \\ \Rightarrow \quad & v_s^{\mathcal{B}} = v + \omega_\times (r_s e_1 + h_s e_3 - l_s R_{\mathcal{S}}^{\mathcal{B}} e_1) - l_s \dot{\delta}_s R_{\mathcal{S}}^{\mathcal{B}} e_2 \end{aligned}$$

which also yields in the form of affine vectors:

$$\overrightarrow{v}_s = \overrightarrow{v} + \overrightarrow{\omega} \times \overrightarrow{GG_s} - l_s \dot{\delta}_s \overrightarrow{\jmath}_s$$

Consequently, the expression of the apparent velocity of the sail satisfies:

$$\overrightarrow{v}_{as} = \overrightarrow{v} - \overrightarrow{v}_w + \overrightarrow{\omega} \times \overrightarrow{GG_s} - l_s \dot{\delta}_s \overrightarrow{\jmath}_s$$

From here, the sail's angle of attack can be computed as:

$$\alpha_s := \operatorname{atan}\left(\frac{-\overrightarrow{v}_{as}.\overrightarrow{\jmath}_s}{\sqrt{(\overrightarrow{v}_{as}.\overrightarrow{\imath}_s)^2 + (\overrightarrow{v}_{as}.\overrightarrow{k}_s)^2}}\right) = \operatorname{atan}\left(\frac{-v_{as,2}^{\mathcal{S}}}{\sqrt{(v_{as,1}^{\mathcal{S}})^2 + (v_{as,3}^{\mathcal{S}})^2}}\right)$$

The aerodynamic force vector $\overrightarrow{F}_s$ can be decomposed in two components: the lift force $\overrightarrow{F}_s^L$, perpendicular to the apparent velocity, and the drag force, parallel to the apparent velocity. The lift force direction is characterized by the unit vector $\overrightarrow{e}_s^L := \sin\alpha_s \overrightarrow{\beta}_s + \cos\alpha_s \overrightarrow{\jmath}_s$, where $\overrightarrow{\beta}_s$ is the unit vector collinear with the vector $\overrightarrow{v}_{as,1,3} := (\overrightarrow{v}_{as}.\overrightarrow{\imath}_s)\overrightarrow{\imath}_s + (\overrightarrow{v}_{as}.\overrightarrow{k}_s)\overrightarrow{k}_s$. Besides, since the apparent velocity $\overrightarrow{v}_{as}$ can be expressed in the form $\overrightarrow{v}_{as} = |\overrightarrow{v}_{as}|(\cos\alpha_s \overrightarrow{\beta}_s - \sin\alpha_s \overrightarrow{\jmath}_s)$, one easily deduces that: From here, the aerodynamic lift and drag forces can be modeled as:

$$\begin{cases} \overrightarrow{F}_s^D = -\lambda_s C_s^D(\alpha_s)|\overrightarrow{v}_{as}|\overrightarrow{v}_{as} \\ \overrightarrow{F}_s^L = \lambda_s C_s^L(\alpha_s)|\overrightarrow{v}_{as}|^2 \overrightarrow{e}_s^L = \lambda_s C_s^L(\alpha_s)|\overrightarrow{v}_{as}|\left(\tan\alpha_s \overrightarrow{v}_{as} + \dfrac{|\overrightarrow{v}_{as}|}{\cos\alpha_s}\overrightarrow{\jmath}_s\right) \end{cases}$$

with $\lambda_s := \frac{1}{2}\rho_{air} S_s$, $C_s^L(\alpha_s)$ the lift coefficient and $C_s^D(\alpha_s) > 0$ the drag coefficient. Thus, the expression of the total resulting aerodynamic force $\overrightarrow{F}_s := \overrightarrow{F}_s^D + \overrightarrow{F}_s^L$ satisfies:

$$\overrightarrow{F}_s = -\lambda_s \left(C_s^D(\alpha_s) - C_s^L(\alpha_s)\tan\alpha_s\right) |\overrightarrow{v}_{as}|\overrightarrow{v}_{as} + \lambda_s \frac{C_s^L(\alpha_s)}{\cos\alpha_s}|\overrightarrow{v}_{as}|^2\overrightarrow{\jmath}_s \quad (1)$$

Finally, one deduces the resulting torque vector:

$$\overrightarrow{\tau}_s = \overrightarrow{GG_s} \times \overrightarrow{F}_s \quad (2)$$

The above deduction can be directly applied to obtain the expressions of the hydrodynamic forces and torques acting on the rudder and the keel.

3.1.2 Hydrodynamic Forces and Torques Acting on the Rudder

Similarly to the previous case, one deduces that the velocity of G_r satisfies $\overrightarrow{v}_r = \overrightarrow{v} + \overrightarrow{\omega} \times \overrightarrow{GG_r} - l_r\dot{\delta}_r\overrightarrow{\jmath}_r$, with $\overrightarrow{GG_r} = -r_r\overrightarrow{\imath} - l_r\overrightarrow{\imath}_r - h_r\overrightarrow{k}$. Subsequently, the apparent velocity of the rudder, defined as $\overrightarrow{v}_{ar} := \overrightarrow{v}_r - \overrightarrow{v}_c$, is given by:

$$\overrightarrow{v}_{ar} = \overrightarrow{v} - \overrightarrow{v}_c + \overrightarrow{\omega} \times \overrightarrow{GG_r} - l_r\dot{\delta}_r\overrightarrow{\jmath}_r \quad (3)$$

Then, similarly to the sail case, the total hydrodynamic force acting on the rudder can be modeled as:

$$\overrightarrow{F}_r = -\lambda_r \left(C_r^D(\alpha_r) - C_r^L(\alpha_r)\tan\alpha_r\right) |\overrightarrow{v}_{ar}|\overrightarrow{v}_{ar} + \lambda_r \frac{C_r^L(\alpha_r)}{\cos\alpha_r}|\overrightarrow{v}_{ar}|^2\overrightarrow{\jmath}_r \quad (4)$$

with $\lambda_r = \frac{1}{2}\rho_{water}S_r$, $C_r^L(\alpha_r)$ the lift coefficient, $C_r^D(\alpha_r) > 0$ the drag coefficient and α_r the rudder's angle of attack defined as:

$$\alpha_r := \text{atan}\left(-v_{ar,2}^{\mathcal{R}}/\sqrt{(v_{ar,1}^{\mathcal{R}})^2 + (v_{ar,3}^{\mathcal{R}})^2}\right)$$

Finally, the resulting torque vector is:

$$\overrightarrow{\tau}_r = \overrightarrow{GG_r} \times \overrightarrow{F}_r \quad (5)$$

3.1.3 Hydrodynamic Forces and Torques Acting on the Keel

Since the keel is rigidly attached to the sailboat's hull and its x-axis coincides with the one of the vehicle, one deduces that the apparent velocity of the keel satisfies:

$$\overrightarrow{v}_{ak} = \overrightarrow{v} - \overrightarrow{v}_c + \overrightarrow{\omega} \times \overrightarrow{GG_k}$$

with $\overrightarrow{GG_k} = -h_k\overrightarrow{k}$. The keel's angle of attack is defined as:

$$\alpha_k := \text{atan}\left(-v_{ak,2}^{\mathcal{B}}/\sqrt{(v_{ak,1}^{\mathcal{B}})^2 + (v_{ak,3}^{\mathcal{B}})^2}\right)$$

Similarly to the sail and rudder case, the total hydrodynamic force acting on the keel is given by:

$$\overrightarrow{F}_k = -\lambda_k \left(C_k^D(\alpha_k) - C_k^L(\alpha_k)\tan\alpha_k\right) |\overrightarrow{v}_{ak}|\overrightarrow{v}_{ak} + \lambda_k \frac{C_k^L(\alpha_k)}{\cos\alpha_k}|\overrightarrow{v}_{ak}|^2\overrightarrow{\jmath} \quad (6)$$

with $\lambda_k = \frac{1}{2}\rho_{water}S_k$. Finally, the resulting torque vector is:

$$\overrightarrow{\tau}_k = \overrightarrow{GG_r} \times \overrightarrow{F}_k \quad (7)$$

3.1.4 Hydrodynamic Resistance Forces and Torques

Hydrodynamic resistance is caused by the friction and the waves. In the first approximation, the resistance force on the hull can be modeled as the sum of linear and quadratic terms:

$$\begin{aligned}\overrightarrow{F_d} = &- c_1^\imath(\overrightarrow{v}_a \cdot \overrightarrow{\imath})^2 \cdot \overrightarrow{\imath} - c_2^\imath(\overrightarrow{v}_a \cdot \overrightarrow{\imath})\overrightarrow{\imath} - c_1^\jmath(\overrightarrow{v}_a \cdot \overrightarrow{\jmath})^2 \cdot \overrightarrow{\jmath} - c_2^\jmath(\overrightarrow{v}_a \cdot \overrightarrow{\jmath})\overrightarrow{\jmath} \\ &- c_1^k(\overrightarrow{v}_a \cdot \overrightarrow{k})^2 \cdot \overrightarrow{k} - c_2^k(\overrightarrow{v}_a \cdot \overrightarrow{k})\overrightarrow{k}\end{aligned}$$

with $\overrightarrow{v}_a := \overrightarrow{v} - \overrightarrow{v}_c$. A similar form for the resistance torque generated by rotational movement is:

$$\overrightarrow{\tau_d} = -c_3^\imath\omega_1^2\overrightarrow{\imath} - c_4^\imath\omega_1\overrightarrow{\imath} - c_3^\jmath\omega_2^2\overrightarrow{\jmath} - c_4^\jmath\omega_2\overrightarrow{\jmath} - c_3^k\omega_3^2\overrightarrow{k} - c_4^k\omega_3\overrightarrow{k}$$

Translation resistance coefficients along $\overrightarrow{\jmath}$ and $\overrightarrow{k}$ are higher than along $\overrightarrow{\imath}$, making damping more important on lateral movement. This is due to the shape of the hull (its length is more important than its height or width). For the same reason, rotational resistance coefficients around $\overrightarrow{\jmath}$ and $\overrightarrow{k}$ are less than around $\overrightarrow{\imath}$.

3.1.5 Restoring Force and Torque Acting on the Hull

Let ∇ be the displacement volume of water, B the center of buoyancy and M the ship metacenter, $\overrightarrow{F}_B$ the buoyancy force and $\overrightarrow{F}_G$ the gravity one. The restoring force and torque can be modeled as (see e.g. [5]):

$$\overrightarrow{F}_{res.} = \overrightarrow{F}_G + \overrightarrow{F}_B = -mg\overrightarrow{k}_0 + \nabla\rho_{water}\overrightarrow{k}_0$$

$$\overrightarrow{\tau}_{res.} = \overrightarrow{F_B} \times \overrightarrow{GB} = \overrightarrow{F_B} \times (\overrightarrow{GM} + \overrightarrow{GB}) = \overrightarrow{F_B} \times \overrightarrow{GM}$$

where the last equality is obtained under the approximation $\overrightarrow{k}_0 \times \overrightarrow{MB} \approx \overrightarrow{0}$.

3.2 Rigid-Body Equations of Motion

3.2.1 General 6-DOF Equations of Motion

The kinematic equations of motion of the sailboat are given by:

$$\begin{cases} \dot{x} = Rv \\ \dot{R} = R\omega_\times \end{cases} \tag{8}$$

For modeling the dynamics of the sailboat, the added mass effects are also taken into account. Some definitions are recalled (see, e.g., [5]). Since the center of mass G coincides with the origin of the body-fixed frame $\mathcal{B}$, the rigid-body inertia matrix M_{RB} and the Coriolis and centripetal rigid-body matrix $C_{RB}(\nu)$ have the following form [5]:

$$M_{RB} := \begin{bmatrix} mI_3 & 0 \\ 0 & J_0 \end{bmatrix}, \quad C_{RB}(\nu) := \begin{bmatrix} m\omega_\times & 0 \\ 0 & -(J_0\omega)_\times \end{bmatrix}$$

The added inertia and Coriolis/centripetal added matrices are denoted as [5]:

$$M_A := \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix}, \quad C_A(\nu) := \begin{bmatrix} 0 & -(A_{11}v + A_{12}\omega)_\times \\ -(A_{11}v + A_{12}\omega)_\times & -(A_{21}v + A_{22}\omega)_\times \end{bmatrix}$$

Let us also define $M_T := M_{RB} + M_A$ and $C_T(\nu) := C_{RB}(\nu) + C_A(\nu)$. From here, the general 6 DOF dynamic equations of motion are given by [5]:

$$M_T\dot{\nu} + C_T(\nu)\nu = \begin{bmatrix} F_d^{\mathcal{B}} + F_{res.}^{\mathcal{B}} + F_s^{\mathcal{B}} + F_r^{\mathcal{B}} + F_k^{\mathcal{B}} \\ \tau_d^{\mathcal{B}} + \tau_{res.}^{\mathcal{B}} + \tau_s^{\mathcal{B}} + \tau_r^{\mathcal{B}} + \tau_k^{\mathcal{B}} \end{bmatrix} \tag{9}$$

We now simplify the expressions of lift and drag forces and torques acting on the sailboat by assuming that the lift and drag coefficients of the sail, the rudder and the keel satisfy the relation:

$$C_i^D(\alpha_i) - C_i^L(\alpha_s)\tan(\alpha_i) = c_0^i, \quad i = s, r, k \tag{10}$$

with c_0^i some very small positive numbers, i.e. $c_0^i \ll 1$. An exemplified model for which relation (10) holds is (11), which is convenient for NACA 00XX profiles [11].

$$\begin{cases} C_i^D(\alpha_i) = c_0^i + 2c_1^i \sin^2(\alpha_i) \\ C_i^L(\alpha_i) = c_1^i \sin(2\alpha_i) \end{cases} \tag{11}$$

with c_0^i, c_1^i some positive parameters. This lift and drag coefficients model can be used for sails, rudder and keel and remains convenient for control and design purpose although it neglects the stall phenomenon. For modeling purpose, more accurate sails models can be used as [15]. From here, using Eqs. (1), (4), (6) and model (11), one deduces:

$$\begin{cases} F_s^{\mathcal{B}} \approx \lambda_s \dfrac{C_s^L(\alpha_s)}{\cos\alpha_s} |\overrightarrow{v}_{as}|^2 R_{\mathcal{S}}^{\mathcal{B}} e_2 = 2\lambda_s c_1^s |v_{as}^{\mathcal{S}}| v_{as,2}^{\mathcal{S}} (\sin\delta_s e_1 - \cos\delta_s e_2) \\ F_r^{\mathcal{B}} \approx \lambda_r \dfrac{C_r^L(\alpha_r)}{\cos\alpha_r} |\overrightarrow{v}_{ar}|^2 R_{\mathcal{R}}^{\mathcal{B}} e_2 = 2\lambda_r c_1^r |v_{ar}^{\mathcal{R}}| v_{ar,2}^{\mathcal{R}} (\sin\delta_r e_1 - \cos\delta_r e_2) \\ F_k^{\mathcal{B}} \approx \lambda_k \dfrac{C_k^L(\alpha_k)}{\cos\alpha_k} |\overrightarrow{v}_{ak}|^2 e_2 = -2\lambda_k c_1^k |v_{ak}^{\mathcal{B}}| v_{ak,2}^{\mathcal{B}} e_2 \end{cases} \quad (12)$$

The approximations made in (12) also indicate that the aero(hydro)-dynamic forces acting on the sail, rudder and keel are approximately orthogonal to their average planes. This approximation can also be found in [1,6].

Then, using Eqs. (2), (5), (7), and the approximations in (12), the aero(hydro)-dynamic torques are simplified as:

$$\begin{cases} \tau_s^{\mathcal{B}} = ((r_s - l_s\cos\delta_s)e_1 - l_s\sin\delta_s e_2 + h_s e_3) \times F_s^{\mathcal{B}} \\ \quad \approx 2\lambda_s c_1^s |v_{as}^{\mathcal{S}}| v_{as,2}^{\mathcal{S}} (h_s\cos\delta_s e_1 + h_s\sin\delta_s e_2 + (l_s - r_s\cos\delta_s)e_3) \\ \tau_r^{\mathcal{B}} = (-(r_r + l_r\cos\delta_r)e_1 - l_r\sin\delta_r e_2 - h_r e_3) \times F_r^{\mathcal{B}} \\ \quad \approx 2\lambda_r c_1^r |v_{ar}^{\mathcal{R}}| v_{ar,2}^{\mathcal{R}} (-h_r\cos\delta_r e_1 - h_r\sin\delta_r e_2 + (l_r + r_r\cos\delta_r)e_3) \\ \tau_k^{\mathcal{B}} = -h_k e_3 \times F_k^{\mathcal{B}} \approx -2\lambda_k c_1^k h_k |v_{ak}^{\mathcal{B}}| v_{ak,2}^{\mathcal{B}} e_1 \end{cases} \quad (13)$$

We will show next that the general 6-DOF equations of motion (8)–(9) can be greatly simplified to 4-DOF and 3-DOF models under some assumptions and approximations.

3.2.2 Simplified 4-DOF Equations of Motion

By approximating the hull to a volume with three mutually perpendicular axes of symmetry, the contributions of the off-diagonal elements in the added mass matrix can be neglected, i.e. $A_{12} \approx A_{21} \approx 0$ and A_{11} and A_{22} are diagonal.

In this case, Eqs. (9) can be rewritten as:

$$\begin{cases} M\dot{v} = -S(\omega)Mv + F_d^{\mathcal{B}} + F_{res.}^{\mathcal{B}} + F_s^{\mathcal{B}} + F_r^{\mathcal{B}} + F_k^{\mathcal{B}} \\ J\dot{\omega} = -S(\omega)J\omega + \tau_d^{\mathcal{B}} + \tau_{res.}^{\mathcal{B}} + \tau_s^{\mathcal{B}} + \tau_r^{\mathcal{B}} + \tau_k^{\mathcal{B}} \end{cases} \quad (14)$$

with $M := m_0 I_3 + A_{11} = \mathrm{diag}(m_{11}, m_{22}, m_{33})$ and $J := J_0 + A_{22}$.

In order to derive a 4-DOF simplified model, let us assume that the translational motion along $\overrightarrow{k}_0$ direction and the pitch rotational motion are negligible w.r.t. other motions. This leads to the approximations $x_3 = \dot{x}_3 = 0$ and $\theta = \dot{\theta} = 0$. The assumption on the pitch motion can be justified in practice due to the fact that the restoring level arm related to the pitch motion is sufficiently large w.r.t. the one related to the roll motion.

Under these approximations, the rotation matrix R can be simply expressed as a product of a yaw and a roll rotation matrix, i.e. $R = R_\psi R_\phi$, with:

$$R_\psi := \begin{bmatrix} \cos\psi & -\sin\psi & 0 \\ \sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad R_\phi := \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi & \cos\phi \end{bmatrix}$$

On the other hand, the fact that $\theta = \dot{\theta} = 0$ leads to the following relations between the angular velocities and the time-derivative of the roll and yaw Euler angles [13]:

$$\begin{cases} \dot{\phi} = \omega_1 \\ \dot{\psi} = \omega_3(\cos\phi)^{-1} \\ \omega_2 = \omega_3 \tan\phi \end{cases} \Rightarrow \begin{cases} \ddot{\phi} = \dot{\omega}_1 \\ \ddot{\psi} = \dot{\omega}_3(\cos\phi)^{-1} + \omega_1\omega_3 \tan\phi(\cos\phi)^{-1} \end{cases}$$

Besides, the relation $\dot{x}_3 = 0$ can be rewritten as:

$$e_3^\top R_\psi R_\phi v = v_2 \sin\phi + v_3 \cos\phi = 0 \;\Rightarrow\; v_3 = -v_2 \tan\phi \tag{15}$$

The assumption on the vertical motion also implies that $\overrightarrow{F}_B = -\overrightarrow{F}_G$ leading to $\overrightarrow{F}_{res.} = \overrightarrow{0}$. The restoring torque can also be approximated by $\tau^{\mathcal{B}}_{res.} = -mgl_\phi \sin\phi e_1$, with a constant restoring level arm $l_\phi > 0$.

The longitudinal and lateral velocities $V_{long.}$, $V_{lat.}$ of the sailboat are defined as:

$$V := \begin{bmatrix} V_{long.} \\ V_{lat.} \\ 0 \end{bmatrix} := R_\psi^\top \dot{x} = R_\phi v = \begin{bmatrix} v_1 \\ v_2\cos\phi - v_3\sin\phi \\ v_2\sin\phi + v_3\cos\phi \end{bmatrix} = \begin{bmatrix} v_1 \\ v_2(\cos\phi)^{-1} \\ 0 \end{bmatrix}$$

where the last equality is obtained using (15). The dynamics of V satisfy:

$$\begin{aligned} \dot{V} &= R_\phi \dot{\phi} S(e_1) v + R_\phi M^{-1}(-S(\omega)Mv + F_d^{\mathcal{B}} + F_s^{\mathcal{B}} + F_r^{\mathcal{B}} + F_k^{\mathcal{B}}) \\ &= \omega_1 S(e_1) V - R_\phi M^{-1} S(\omega) M R_\phi^\top V + R_\phi M^{-1}(F_d^{\mathcal{B}} + F_s^{\mathcal{B}} + F_r^{\mathcal{B}} + F_k^{\mathcal{B}}) \end{aligned}$$

which yields:

$$\begin{cases} \dot{V}_{long.} = -e_1^\top M^{-1} S(\omega) M R_\phi^\top V + e_1^\top M^{-1}(F_d^{\mathcal{B}} + F_s^{\mathcal{B}} + F_r^{\mathcal{B}} + F_k^{\mathcal{B}}) \\ \dot{V}_{lat.} = -e_2^\top R_\phi M^{-1} S(\omega) M R_\phi^\top V + e_2^\top R_\phi M^{-1}(F_d^{\mathcal{B}} + F_s^{\mathcal{B}} + F_r^{\mathcal{B}} + F_k^{\mathcal{B}}) \end{cases} \tag{16}$$

3.2.3 Simplified 3-DOF Equations of Motion

Now, we provide a more simplified 3-DOF model for control design purposes by further assuming that the roll rotational motion can also be neglected, i.e. $\phi = \dot{\phi} = 0$. This assumption is well satisfied for particular design of sailboats for which the restoring torque $\overrightarrow{\tau}_{res.}$ dominates all the other external torques

along the roll and pitch axes. Also, considering a null roll angle leads to maximize forces and torques since they are proportional to $\cos\phi$ [9,14]. Using this additional assumption and the approximation $m_{22} \approx m_{33}$, one deduces from Eq. (16) that the translational dynamics satisfy:

$$\begin{cases} m_{11}\dot{V}_{long.} = F^{\mathcal{B}}_{d,1} + F^{\mathcal{B}}_{s,1} + F^{\mathcal{B}}_{r,1} + F^{\mathcal{B}}_{k,1} + m_{22}\omega_3 V_{lat.} \\ m_{22}\dot{V}_{lat.} = F^{\mathcal{B}}_{d,2} + F^{\mathcal{B}}_{s,2} + F^{\mathcal{B}}_{r,2} + F^{\mathcal{B}}_{k,2} - m_{11}\omega_3 V_{long.} \end{cases} \tag{17}$$

The rotational dynamics is approximately given by:

$$\begin{cases} \dot{\psi} = \omega_3 \\ J_{33}\ddot{\psi} = \tau^{\mathcal{B}}_{d,3} + \tau^{\mathcal{B}}_{s,3} + \tau^{\mathcal{B}}_{r,3} + \tau^{\mathcal{B}}_{k,3} \end{cases} \tag{18}$$

If the approximations given in (12) and (13) are further used, one also ensures that $F^{\mathcal{B}}_{k,1} \approx \tau^{\mathcal{B}}_{k,3} \approx 0$, $\tau^{\mathcal{B}}_{s} \approx 2\lambda_s c^s_1 |v^{\mathcal{S}}_{as}| v^{\mathcal{S}}_{as,2}(l_s - r_s\cos\delta_s)$ and $\tau^{\mathcal{B}}_{r} \approx 2\lambda_r c^r_1 |v^{\mathcal{R}}_{ar}| v^{\mathcal{R}}_{ar,2}(l_r + r_r\cos\delta_r)$.

4 Heading Control and Sail's Angle Computation

The control design for sailboat's heading and longitudinal motion is considered based on the simplified 3-DOF model proposed previously. In the following, they are decoupled and studied separately.

4.1 Heading Control Design

Controlling the sailboat's heading cannot be assigned independently to the rudder's angle δ_r in all circumstances. In fact, this can be done only if the component around the $\overrightarrow{k}_0$ axis of the torque generated by the rudder is unbounded. However, this is obviously not true since this torque component is approximately proportional to the square of the norm of the rudder's apparent velocity $\overrightarrow{v}_{ar}$, and thus tends to zero if the apparent velocity $\overrightarrow{v}_{ar}$ vanishes. Therefore, the coordination between the sail's and rudder's angles would be necessary for the success of a given mission. This is the topic of our future work. In the current study we make the assumption that the torque produced by the rudder can compensate all other external torques around the $\overrightarrow{k}_0$ axis. Thus, for control design let us simply consider the following second-order system:

$$\ddot{\psi} = u + c(t) \tag{19}$$

with $u := \tau^{\mathcal{B}}_{r,3}/J_{33}$ the control input assumed to be unbounded for control design purpose, and $c(t) := (\tau^{\mathcal{B}}_{s,3} + \tau^{\mathcal{B}}_{k,3} + \tau^{\mathcal{B}}_{d,3})/J_{33}$ the perturbation term

assumed to be slowly time-varying (i.e. $\dot{c} \approx 0$) so that it can be compensated by an integral action.

Let ψ_r be the reference heading angle to be stabilized. Then, the control objective may be considered as the stabilization of ψ about ψ_r, or equivalently the stabilization of the error angle $\tilde{\psi} := \psi - \psi_r$ about zero. However, since the heading evolves on a circle, there is no distinction between $\tilde{\psi}$ and $\tilde{\psi} + k2\pi$, with $k \in \mathbb{N}$, and winding problem may occur. Therefore, it is geometrically more relevant to stabilize $\sin\tilde{\psi}$ about zero.

The control design is based on the backstepping procedure. Consider the first storage function $\mathcal{V}_1 = 1 - \cos(\tilde{\psi})$. By choosing a desired value for ω_3 as $\omega_{3,d} = -k_1 \sin(\tilde{\psi}) + \dot{\psi}_r$, with k_1 a positive gain, one deduces:

$$\begin{aligned}\dot{\mathcal{V}}_1 &= \sin(\tilde{\psi})(\omega_3 - \dot{\psi}_r) = \sin(\tilde{\psi})(\omega_{3,d} - \dot{\psi}_r) + \sin(\tilde{\psi})(\omega_3 - \omega_{3,d}) \\ &= -k_1 \sin^2(\tilde{\psi}) + \sin(\tilde{\psi})(\omega_3 - \omega_{3,d})\end{aligned}$$

Next, consider the second storage function $\mathcal{V}_2 = \mathcal{V}_1 + (1/2k_2)(\omega_3 - \omega_{3,d})^2$. One verifies that the time-derivative of $\mathcal{V}_2$ satisfies:

$$\dot{\mathcal{V}}_2 = -k_1 \sin^2(\tilde{\psi}) + \frac{1}{k_2}(\omega_3 - \omega_{3,d})(u + c - \dot{\omega}_{3,d} + k_2 \sin(\tilde{\psi})) \tag{20}$$

The disturbance term c is unknown but can be compensated by its estimate $\hat{c}$ whose dynamics are given by:

$$\dot{\hat{c}} = k_4(\omega_3 - \omega_{3,d}) \ , \ \hat{c}(0) = 0 \tag{21}$$

with k_4 a positive gain. Then, by choosing the control input:

$$u = -k_2 \sin(\tilde{\psi}) - k_3(\omega_3 - \omega_{3,d}) + \dot{\omega}_{3,d} - \hat{c} \tag{22}$$

with k_3 a positive gain and $\dot{\omega}_{3,d} = -k_1 \cos(\tilde{\psi})(\omega_3 - \dot{\psi}_r) + \ddot{\psi}_r$, one verifies that the time-derivative of the candidate Lyapunov function $\mathcal{V}$ defined as:

$$\mathcal{V} := \mathcal{V}_2 + \frac{k_2}{2k_4}(c - \hat{c})^2 = (1 - \cos(\tilde{\psi})) + \frac{1}{2k_2}(\omega_3 - \omega_{3,d})^2 + \frac{k_2}{2k_4}(c - \hat{c})^2$$

satisfies (using Eqs. (20), (21)):

$$\dot{\mathcal{V}} = -k_1 \sin^2(\tilde{\psi}) - (k_3/k_2)(\omega_3 - \omega_{3,d})^2 \leq 0$$

From here, by application of LaSalle's theorem one deduces that $\dot{\mathcal{V}}$ converges to zero, which in turn implies the convergence of $\sin\tilde{\psi}$ to zero and of ω_3 to $\dot{\psi}_r$. In fact, the convergence of $\sin\tilde{\psi}$ to zero implies that $\tilde{\psi}$ converges to either $k2\pi$ or $\pi + k2\pi$. In practice, this is not a problematic issue since the equilibrium $\tilde{\psi} = \pi + k2\pi$ is unstable and the good equilibrium $\tilde{\psi} = k2\pi$ is stable, i.e. the equilibrium $(\tilde{\psi}, \omega, \hat{c}) = (k2\pi, \dot{\psi}_r, c)$ is almost globally stable.

Now, it matters to determine the real control input for the rudder, i.e. the angle δ_r, as a function of the intermediary control variable u (see Eq. (22)). This is a nonlinear control allocation problem that necessitates further studies. Here, in order to deal with this problem we introduce some approximations. First, we assume that there is no water current, i.e. $\overrightarrow{v}_c = \overrightarrow{0}$. Secondly, the contribution of the rotation motion is negligible w.r.t. the translational motion in the computation of the apparent velocity $\overrightarrow{v}_{ar}$. Finally, the influence of the rudder dynamics on the sailboat is small so that the term dependent upon $\dot{\delta}_r$ in the definition (3) of $\overrightarrow{v}_{ar}$ can be neglected. Thus, in the first approximation one has $\overrightarrow{v}_{ar} \approx \overrightarrow{v}$. Besides, in practice the distance l_r is often very small w.r.t. r_r (i.e., $l_r \ll r_r$). We also assume that the sailboat's lateral velocity is small w.r.t. its longitudinal velocity. Therefore, with a good approximation one verifies that:

$$\tau^{\mathcal{B}}_{r,3} \approx -\lambda_r c^r_1 r_r |v|^2 \sin(2\delta_r)$$

and deduces:

$$\delta_r = -\frac{1}{2}\text{arsin}\left(\text{sat}_1\left(\frac{J_{33}u}{\lambda_r c^r_1 r_r |v|^2}\right)\right)$$

with the classical saturation function $\text{sat}_\Delta(x) := x\min(1, \Delta/|x|)$, $\forall x \in \mathbb{R}$.

4.2 Sail's Optimum Angle Computation

The stability limit of the previous heading controller depends on the bounds of the rudder torque which, in turn, is roughly proportional to the square of the boat's velocity. Thus, maximizing this velocity by choosing an optimum sail's angle is of primary concern. This will be achieved by assuming that the maximum longitudinal force generated by the sail will lead to a maximum longitudinal speed of the sailboat.

This can be done by maximizing $F^{\mathcal{B}}_{s,1}$. Let us calculate such an optimum value of δ_s. From Eq. (12) and using the assumption of (11) one obtains

$$F^{\mathcal{B}}_{s,1} = 2c^s_1 \lambda_s |v^{\mathcal{B}}_{as}| \left(-(\sin\delta_s)^2 v^{\mathcal{B}}_{as,1} + \sin\delta_s \cos\delta_s v^{\mathcal{B}}_{as,2}\right)$$

One obtains:

$$\frac{\partial F^{\mathcal{B}}_{s,1}}{\partial \delta_s} = -2c^s_1 \lambda_s |v^{\mathcal{B}}_{as}| \sin(2\delta_s - \beta)$$

with $\beta = \text{atan2}\left(v^{\mathcal{B}}_{as,2}, v^{\mathcal{B}}_{as,1}\right)$. The optimal condition leads to $\frac{\partial F^{\mathcal{B}}_{s,1}}{\partial \delta_s} = 0$ which yields $\delta^{opt}_s = \beta/2 + k\pi/2$ with $k \in \{0, 1, 2, 3\}$. These four values represent two positive and two negative peaks of $F^{\mathcal{B}}_{s,1}$. The peak of $F^{\mathcal{B}}_{s,1}$ can be calculated to determine δ^{max}_s the best new configuration (i.e. realistic value of δ_s that lead to a positive speed).

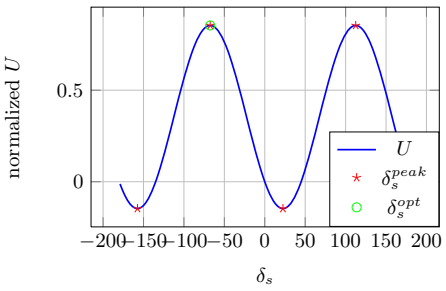


Fig. 2 Variation of the longitudinal force as a function of δ_s; 4 peaks are found: two negatives two positives. One of the positive peaks is unrealizable $|\delta_s| > 90\,\text{deg}$.

A major drawback of this algorithm is that it does not take into account $F^{\mathcal{B}}_{s,2}$, which may be important at δ_s^{opt}, thus generating an important leeway. Besides, the yaw torque generated by F_s at the chosen δ_s angle may become very large and cannot be compensated by the rudder.

5 Simulation Results

The model described in 3.2.3 and both trimming algorithm and heading controller have been implemented using Matlab and a variable-step solver with a maximum time step of $0.5s$.

To test this model, a first simulation set is done disabling trimming algorithm, using a fixed real wind speed but different real wind direction and setting different values of δ_s . Rudder's controller with $k_4 = 0$ is used to maintain the desired heading. Speed's polar diagram is then reconstructed as the one that embrace curves from the simulations (fig. 3). One can observe that when the boat attempts to navigate in irons, velocity decreases and reaches zero. In this reconstructed diagram, one can notice that the maximum velocity is reached when sailing downwind. This is due to simulator constants that may not be well tuned.

The heading and speed response to a sudden change of heading reference are illustrated fig 4. In this case, the wind speed is set to $5\,m/s$ with a coming angle of 0 and both sail's trimming algorithm and rudder's controller are used. Heading reference is first set to $\pi/2$ and is changed to $\pi/4$ at $t = 50\,s$. Simulation is done using the controller with parameters $k_1 = 4$, $k_2 = 1.4$, $k_3 = 2.5$ and $k_4 = 0.2$. As previously mentioned, the term $c(t)$ of Eq. (19) includes torque from both sail and hull and is assumed to be varying slowly. In the case where these torques can directly be measured or estimated, a feedforward term may be added to the controller in order to obtain a faster response of the system.

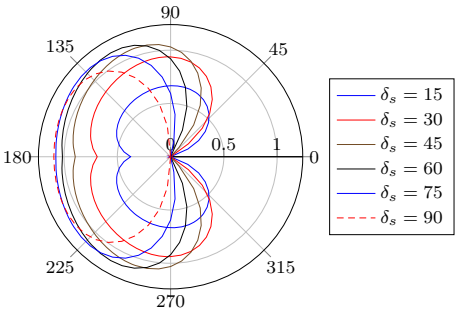
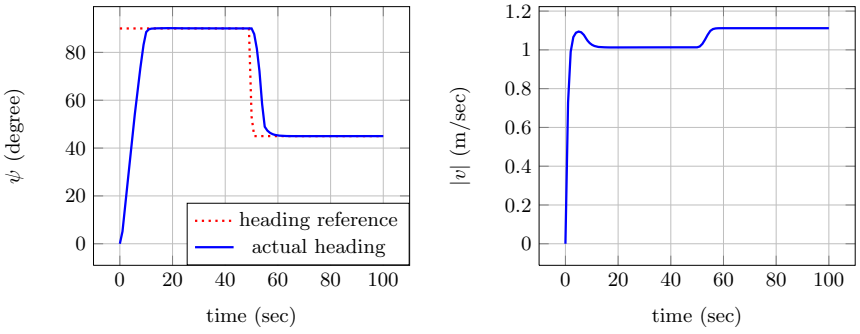
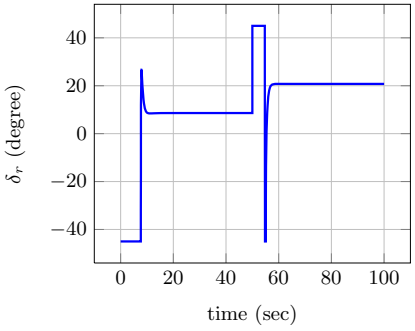


Fig. 3 Speed's polar diagram as a function of δ_s at wind speed of $5[m/s]$



(a) Evolution of boat's heading

(b) Evolution of boat's speed



(c) Evolution of input control

Fig. 4 Response of the system with $\psi_r = \frac{\pi}{2}$ for $t < 50\,s$, $\psi_r = \frac{\pi}{4}$ elsewhere

6 Conclusion

Based on a general 6-DOF dynamic model, a simplified 3-DOF ($x-y$ displacement and yaw rotation) sailboat's model has been derived and implemented

on a computer program. This 3-DOF model has been used to design a new heading control law. The controller acts on yaw angle to maintain a desired heading no matter the route direction is. Notice that this controller reaches its limit when boat's velocity is too low because the generated rudder's torque (approximately proportional to the square of the boat velocity) is not sufficient to counter the sail's torque and other disturbances. A sail trimming algorithm is also presented to compute an optimal sail's angle from lift and drag curves, maximizing longitudinal force in order to maximize longitudinal speed. This assertion hold true only for boats with few leeway. Besides, since no verification is done on the torque generated when using this optimal angle, the resulting torque may become too large to be compensated by the rudder, i.e. the yaw controllability is loss. Our future works will focus on designing new controllers overcoming these drawbacks and taking into account roll effect, implementing the 4-DOF and the 6-DOF models and finally comparing their simulation results with real sailboat tests.

References

1. Briere, Y.: IBOAT: an autonomous robot for long-term offshore operation. In: The 14th IEEE Mediterranean Electrotechnical Conference, MELECON 2008, pp. 323–329 (2008), doi:10.1109/MELCON.2008.4618455
2. Cruz, N., Alves, J.: Autonomous sailboats: An emerging technology for ocean sampling and surveillance. In: MTS/IEEE OCEANS (2008)
3. Elkaim, G.: The atlantis project: A gps-guided wing-sailed autonomous catamaran. Journal of the Institute of Navigation 53, 237–247 (2006)
4. Erckens, H., Busser, G.A., Pradalier, C., Siegwart, R.: Avalon: Navigation strategy and trajectory following controller for an autonomous sailing vessel. IEEE Robotics Automation Magazine 17(1), 45–54 (2010)
5. Fossen, T.I.: Handbook of Marine Craft Hydrodynamics and Motion Control. Wiley-Blackwell (2011)
6. Jaulin, L., Bars, F.L., Clement, B., Gallou, Y., Menage, O., Reynet, O., Sliwka, J., Zerr, B.: Suivi de route pour un robot voilier. In: CIFA 2012, pp. 695–702 (2012), http://hal-ensta-bretagne.archives-ouvertes.fr/hal-00728390
7. Khalil, H.K.: Nonlinear Systems, 3rd edn. Prentice-Hall (2001)
8. Neal, M.: A hardware proof of concept of a sailing robot for ocean observation. IEEE Journal of Oceanic Engineering 31(2), 462–469 (2006)
9. van Oossanen, P.: Theoretical estimation of the influence of some main design factors on the performance of international twelve meter class yachts. In: Chesapeake sailing yacht Symposium, Annapolis, Maryland (1979)
10. Petres, C., Romero-Ramirez, M.A., Plumet, F., Alessandrini, B.: Modeling and reactive navigation of an autonomous sailboat. In: 2011 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 3571–3576 (2011), doi:10.1109/IROS.2011.6094912

11. Pucci, D., Hamel, T., Morin, P., Samson, C.: Nonlinear control of PVTOL vehicles subjected to drag and lift. In: 2011 50th IEEE Conference on Decision and Control and European Control Conference (CDC-ECC), pp. 6177–6183 (2011), 10.1109/CDC.2011.6160712
12. Stelzer, R., Jafarmadar, K., Hassler, H., Charwot, R., et al.: A reactive approach to obstacle avoidance in autonomous sailing. In: International Robotic Sailing Conference, pp. 33–39 (2010)
13. Stuelpnagel, J.: On the parametrization of the three-dimensional rotation group. SIAM Review 6(4), 422–430 (1964), http://www.jstor.org/stable/2027966, doi:10.2307/2027966
14. Van Oossanen, P.: Predicting the speed of sailing yachts'(93-004). Van Oossanen & Associates, Wageningen, The Netherlands (1993)
15. Viola, I.M.: Downwind sail aerodynamics: A CFD investigation with high grid resolution. Ocean Engineering 36(12-13), 974–984 (2009), http://www.sciencedirect.com/science/article/pii/S0029801809001322, doi:10.1016/j.oceaneng.2009.05.011
16. Xiao, L., Jouffroy, J.: Modeling and nonlinear heading control for sailing yachts. In: OCEANS 2011, pp. 1–6 (2011)

Transverse Stability Problems of Small Autonomous Sailing Vessels

Jeffrey Holzgrafe

Abstract. Long term robotic sailboats will be required to withstand extreme weather conditions. However, typical small sail power autonomous sailing vessels (SP-ASVs) fall far outside the range of typical nautical stability experience. In this paper, I analyze three problems related to the stability of SP-ASVs. In the first I show that in general smaller boats will need proportionally greater stability, calling for deeper keels with heavier bulbs. In the second problem, I show that a vessel will be unstable while completely inverted if the inverted metacenter is below the center of gravity. In the last problem, I analyze the roll behavior of a SP-ASV in beam waves. I present a number of methods for estimating the roll parameters of SP-ASVs, and show that Olin College's Blackbody Radiation, a typical small sail power autonomous surface vessel, has a great deal of wave stability in roll motions.

1 Introduction

Small autonomous sailing vessels' (SP-ASV) low power consumption make them ideal for long term missions. However, this means they need to handle rough weather - that is, they have a high degree of static and dynamic stability. This paper introduces the terminology of stability in marine engineering while discussing three stability problems for small monohull sail power autonomous surface vessels in order of increasing nuance.

Most SP-ASVs are shorter than 4m - much smaller than a normal sailboat- because it lowers the material cost and handling difficulty for research groups. The SP-ASVs at the International Robotic Sailing Competition have keels that are considerably deeper than a standard scale sailboat. See for example

Jeffrey Holzgrafe
Olin College, 1 Olin Way Needham MA 02492
e-mail: jeffrey.holzgrafe@students.olin.edu

F. Le Bars and L. Jaulin (eds.), *Robotic Sailing 2013*,
DOI: 10.1007/978-3-319-02276-5_9,

[8]. In Section 2, I will explain why SP-ASVs have better performance with a deeper keel.

If an SP-ASV does capsize, the vessel should right itself quickly so that the rudder can afford some degree of control. In Section 3, I will develop a criterion which determines whether a vessel will be unstable in an inverted orientation.

Longitudinally symmetric vessels in rough conditions that have lost self-propulsion, for example if the sails are intentionally reefed, will be oriented by the waves so that they experience beam seas [17]. In these low control cases, it is important the vessel not capsize. In Section 4, I show how to estimate and measure dynamic stability parameters for a SP-ASV and apply the safe basin technique to analyzing her stability in beam waves.

2 Scaling and Static Stability

Conventional wisdom in SP-ASV design says that the keel should be proportionally deeper than that of a large sailboat to prevent excessive heel. Indeed some of the fastest SP-ASVs have keels that are as long as the hull, see for example the USNA's Gill the Boat [8]. In this section, I attempt to explain this rule of thumb using a simple model of boat stability.

Static stability of floating structures is caused by gravitational (low center of gravity) and form stability (wide hull shape) [3]. I will explain the non-proportional keels using a simple model of a sailboat shown in Figure 1 in which we ignore form stability by assuming the center of buoyancy does not move. This is a significant approximation, and is only accurate for vessels for which the center of gravity is below the center of buoyancy, such as with most SP-ASVs. For small angles of heel (conventionally less than 7 or 8 degrees, although the accuracy of the approximation depends significantly on the vessel), form stability behaves the same as gravitational stability, so the gravitational-only assumption will not change the conclusions. Form stability will be discussed, and a more complete, large angle, model of sailboat stability developed, in section 3 and 4.3.

I will develop a criterion of equilibrium in steady conditions (constant wind speed, calm water) and then determine how this changes when the boat is scaled. I will consider the center of buoyancy as the center of rotation and find the moments about it. Assuming the boat is in translational equilibrium, we can take $F_A = F_P$. F_A can be modeled using the usual drag equation, thus the heeling moment (M_w) due to the wind is

$$M_w = (R_A + R_P)\frac{1}{2}C\rho_A A_s v_w^2 \cos\theta \tag{1}$$

Where R_A and R_P are lengths shown in Figure 1, C is the drag coefficient of the sails, ρ_A is the density of air, A_s is the sail area, v_w is the wind

velocity, and θ is the heel angle from vertical. The moment is moderated by $\cos\theta$ because, while F_A is assumed to always be perpendicular to the sails, the wind-exposed sail area reduces with angle as $\cos\theta$.

The restoring moment (M_G) due to gravity is

$$M_G = R_G mg \sin\theta \tag{2}$$

Where R_G is the distance from the center of gravity to the center of rotation, m is the total mass of the boat, and g is the acceleration of gravity. The more complete description of restoring moment looks similar - a distance term times a weight term - for small angles.

The boat will be in equilibrium where these moments are equal, allowing us to approximate the angle of heel:

$$\tan\theta = \frac{(R_A + R_P)C\rho A_s v_w^2}{2R_G mg} \tag{3}$$

I will now consider two boats B (larger), and B' (smaller) that are geometrically similar, but differ by a linear scale factor a less than one. That is, if B is length L, then B' is shorter at length aL. m scales with the cube of the scale factor, A_s the square, and R_G, R_A and R_P are linear with the scale factor. Using this we can write the equilibrium condition of B' in terms of the quantities of B and relate their heel angles.

$$\tan\theta' = \frac{(R'_A + R'_P)C\rho A'_s v_w^2}{2R'_G m' g} = \frac{a(R_A + R_P)C\rho a^2 A_s v_w^2}{2aR_G a^3 mg} = \frac{\tan\theta}{a} \tag{4}$$

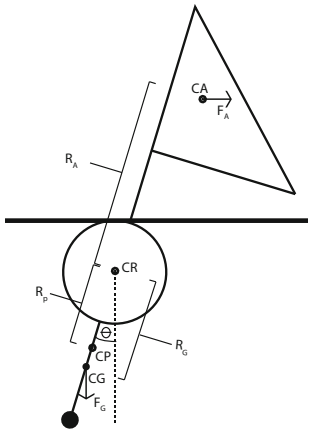


Fig. 1 The simplified boat system which ignores form stability. The aerodynamic and hydrodynamic forces, F_A and F_P from the wind acting on the sails at the center of aerodynamic effort (CA) and the water acting on the submerged surface at the center of hydrodynamic effort (CP), causes the boat to heel over. The weight of the boat, acting at the center of gravity (CG) causes the boat to return upright.

This means the smaller boat will heel more, because a is by definition less than one. This change is caused by the fact that the heeling moment depends on the cube of the scale factor and the restoring moment the fourth power. When the scale factor is less than one, the heeling force will be proportionally larger than the restoring force, creating a greater heel angle. Thus, because of basic scaling laws, smaller boats will tend to heel more. This is particularly noticeable for extremely small boats such as many SP-ASVs. To counteract this SP-ASVs require proportionally deeper keels and heavier bulbs.

It is important to note that this conclusion is largely qualitative. The simplified model of boat stability I used is unfit for quantitative comparison or design, but it shows the importance of scaling laws on SP-ASVs clearly.

3 Inverted Stability

If a SP-ASV capsizes in open water, it should revert to an upright position so that it can resume normal operation. Thus SP-ASVs should be designed so they are unstable in the inverted position.

Using the model from section 2, the boat would never be stable in an inverted position, so we need to include form stability to account for this possibility. Form stability can be expressed for small angles using the concept of a metacenter, a linearization about an equilibrium point and a fundamental concept of marine engineering. The metacenter is the point which is directly above the center of buoyancy for small heel angles. See Figure 2.

The metacenter is a function of the shape of the hull's water line plane:

$$\overline{BM} = \frac{\frac{2}{3}\int_0^L y^3\,dx}{\nabla} \tag{5}$$

Where x is oriented longitudinally, ∇ is the volume of water displaced, and the integral is taken over the water plane of the boat, see [5]. We can now consider an inverted boat to determine the conditions under which it will be unstable. Note that the inverted metacentric height is not in general equivalent to the upright metacentric height. If we look at the moment about

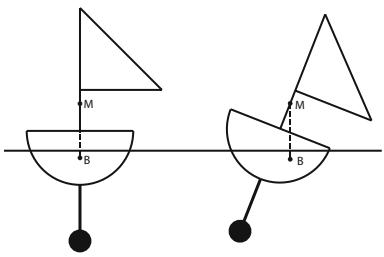


Fig. 2 The metacenter is a point fixed for small angles that is above the center of buoyancy

the metacenter, it will be positive if the CG is above the metacenter and negative otherwise. Thus, as shown in Figure 3, the boat will be unstable if the inverted metacenter is below the center of gravity in the inverted orientation.

Because SP-ASVs must have such low CGs (high when inverted) as shown in Section 2, they will have an unstable equilibrium in the inverted position, and thus will not be likely to get stuck upside down in rough weather.

4 Stability in Beam Seas

Ship roll in beam waves is strongly non-linear for large heel angles, such as those experienced during a capsize event. A number of techniques have been used to analyze the behavior of this system, including the harmonic balance method [14], the Melnikov method [2], Lyapunov direct method [9], multiple time scales method [12], Lyapunov exponents [7], and the safe basin technique [15]. I will apply this last technique to SP-ASVs.

In the safe basin technique, the portions of the initial condition phase space which do not lead to capsize are considered to be in the safe basin. Plots of this safe basin can be created for a variety of weather conditions to give insight into the beam sea stability of the vessel. Safe basin techniques has been applied to trawlers in beam seas [16], used to predict the probability of capsize in random beam seas [6], and to understand the frequency space response of boats in beam waves [20].

Full dynamical modeling, for example in a velocity prediction program, is considerably more powerful, and consequently more common in industrial sailboat design. However, such techniques are cumbersome, requiring detailed fluid simulations. The safe basin technique can address only limited problems, but it allows for insight into the beam stability of floating structures with a comparatively small amount of resources and information about the vessel.

The equation of motion of a ship in beam waves is made up of a number of components which need to be estimated for a given vessel. This section will also discuss methods of approximating these components for research groups without access to sophisticated marine engineering design software. To ground the analysis, I will be applying these techniques to Olin College's racing SP-ASV, Blackbody Radiation (Figure 4).

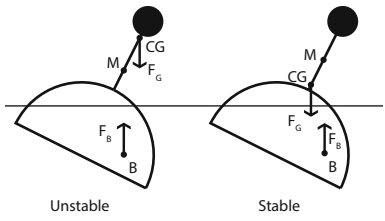


Fig. 3 A vessel will have an unstable inverted equilibrium if the CG is above the metacenter

4.1 The Roll Equation

A vessel rolling in beam waves will experience coupled roll, heave and sway motions. However, the heave and sway motions have relatively small effects on roll, although the long keels of SP-ASVs do increase the coupling [4]. Thus in most studies, only roll motion is considered by introducing a virtual roll center [1]. This simplifies the 3-DOF transverse motion problem into a 1-DOF system.

The equation of rolling motion for a ship in regular beam seas is:

$$I\ddot{\theta} + D(\dot{\theta}) + M_s(\theta) = M_e(t) \tag{6}$$

Where I is the total virtual moment of inertia about the virtual roll center, $D(\dot{\theta})$ is the damping moment function, $M_s(\theta)$ is the restoring moment function and $M_e(t)$ is the excitation moment caused by the waves. I ignore the effects of wind, assuming that the vessel has completely reefed its sails in rough weather conditions. I will now discuss ways to estimate each of these components of the equation of motion.

4.2 Virtual Moment of Inertia

Bodies rotating within a fluid must move not only themselves but also some of the fluid around them. This means that the effective, or virtual, moment of inertia of a body in a fluid will be higher than that of the body itself. See texts on fluid dynamics for more discussion, for example [19].

Calculating the added mass moment of inertia of a body in water is quite difficult for low symmetry objects like sailing vessels. However, it is relatively easy to determine experimentally. The un-damped natural roll frequency(ω_0) of a vessel in small angles is:

Fig. 4 Blackbody Radiation, Olin College's racing SP-ASV

$$\omega_0^2 = \frac{\Delta \overline{GM}}{I} \tag{7}$$

Where Δ is the vessel displacement, $\overline{GM}$ is the metacentric height, and I is the virtual moment of inertia [5]. Assuming that the effect of damping on the natural period is small, I measured the roll period of Blackbody in a free roll test by videotaping her as she is heeled over and released. From this I calculated the natural roll frequency ($\omega_0 = \frac{2\pi}{T}$ where T is the roll period), and knowing the displacement from direct measurement and $\overline{GM}$ from Equation 5, calculated the virtual moment of inertia.

4.3 Restoring Moment

A heeled vessel will experience a moment caused by the buoyant and gravitational forces, called the restoring moment. Current research emphasizes that restoring moment altering effects such as hull trim and sinkage can greatly complicate the roll dynamics [10]. However, I will assume the simplest system: an evenly trimmed boat with no sinkage. By assuming the gravitational force is always equal to the buoyant force, that is, by neglecting heave, we can determine this restoring moment from only knowledge of the center of gravity and center of buoyancy. Considering Figure 5 we see that the restoring moment is,

$$M_s = m\overrightarrow{g}X\overline{GR} = mg\overline{GR}\sin\theta = mg\overline{GZ} \tag{8}$$

Thus, to determine the righting moment we need only to find $\overline{GZ}(\theta)$. Most marine engineering design software can do this automatically given the vessels' displacement. However, if you do not have access to such software, it is possible to approximate it with more general CAD software. Do to this, I took our group's model of Blackbody, made it solid and set the material

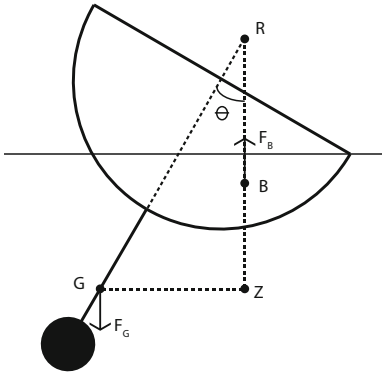


Fig. 5 The forces which create a restoring moment

to have the density of water. I then rolled the model and removed material from the model in a vertical plane moving downward. Once the weight of the model equaled the displacement of the vessel, the remaining parts of the model would be those below the waterline for the given heel angle. Then, I calculated the center of gravity of the remaining material, which would also be the center of buoyancy. I determined the center of gravity of the intact boat by weighing components of the vessel and estimating their center of gravity. Using the center of buoyancy and gravity for a number of heel angles it is relatively easy to determine a number of points in the $\overline{GZ}$ curve. See Figure 6 for the results.

The $\overline{GZ}$ curve must physically be an odd function of theta because the vessel is longitudinally symmetric. Thus it is makes sense to approximate the $\overline{GZ}$ curve with an odd polynomial. I used a third order odd polynomial, and fit the constants to the calculated points using a least squares fit over the range $0 < \theta < \frac{3\pi}{4}$:

$$\overline{GZ} \approx k_{r1}\theta + k_{r3}\theta^3 \tag{9}$$

You can see the fit in Figure 6.

4.4 Damping Moment

The damping moment is harder to approximate. I did so with a free roll test, using experimentally determined free roll dynamics to tune damping parameters. I took a video of the vessel as it was heeled over and then released in calm water. By analyzing that video, I determined the roll angle of the vessel at a number of times. I numerically solved the equation of motion (Equation 6) with the same initial conditions and using the previously discussed components and setting $M(t) = 0$. I then approximated the damping moment in another third order odd polynomial,

$$D(\dot{\theta}) \approx k_{d1}\dot{\theta} + k_{d3}\dot{\theta}^3 \tag{10}$$

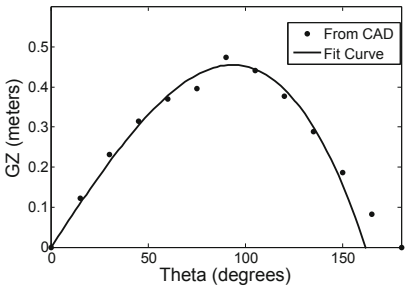


Fig. 6 The GZ curve for Blackbody Radiation, with approximation curve. The approximation is fit over 0 to 3/4pi radians.

and tuned the damping constants so that the time response fit the collected data. See Figure 7 for a comparison of the experimental and model results with tuned damping constants.

4.5 Excitation Moment

The excitation moment can be described by [13]:

$$a_0 \Delta \overline{GM} \pi \frac{H_w}{\lambda_w} \cos \omega t = a_0 \Delta \overline{GM} \pi s \cos \omega t \tag{11}$$

Where a_0 is the reduction coefficient for the effective wave slope, H_w is the wave height, λ_w is the wavelength, and ω is the wave angular wave frequency. The quantity $\frac{H_w}{\lambda_w}$ is known as the wave steepness, s.The wave steepness is physically limited to $\frac{1}{7}$ for deep water waves before they begin to cap [11].

The reduction coefficient is a function of wave frequency in general, but it changes little, so we will assume it is constant to simplify the system. This is a semi-empirical constant, and is difficult to estimate. However, Watanbe [18] developed a relationship from a fit to data from 60 vessels:

$$a_0 = 0.73 + 0.6 \frac{\overline{OG}}{d} \tag{12}$$

Where $\overline{OG}$ is the height of the center of gravity above the waterline and d is the draught. SP-ASVs falls well outside the design of normal ships, and thus this relationship is likely not accurate. More nuanced wave tank tests would be required to reliably estimate a_0. Nonetheless, I will use this empirically derived formula as a first approximation of the reduction coefficient. This is one of the major inaccuracies of the model.

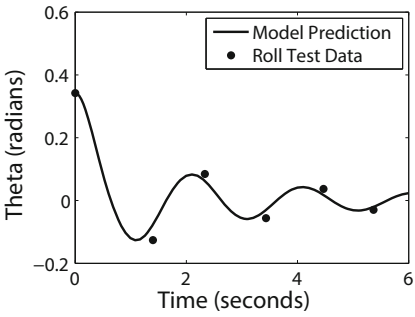


Fig. 7 The free roll response for Blackbody Radiation with tuned damping constants

See Table 1 for a summary of Blackbody's parameters:

Table 1 Table of the roll equation parameters for Olin College's Blackbody Radiation

Parameter	Value
I	8.2 kgm^2
k_{r1}	$0.42mrad^{-1}$
k_{r3}	-0.053$mrad^{-3}$
a_0	0.61
k_{d1}	$4.5Nmsecrad^{-1}$
k_{d3}	35 $Nmsec^3rad^{-3}$
Length	1.45 m
Beam	0.41 m
Draft	1.3 m
Displacement	20 kg

4.6 Safe Basin Technique

The safe basin is the area composed of the set of all initial conditions which do not lead to capsize. Visualizing the safe basin allows the stability of many initial conditions to be examined in one graphic. See Figure 7 for an example of a safe basin plot. To produce the safe basins, I swept over initial conditions $-2 < \theta < 2$ and $-2 < \dot{\theta} < -2$ in a 20 by 20 square. If the vessel heeled past $\frac{\pi}{2}$ or $-\frac{\pi}{2}$, I considered it to have capsized and thus not be a part of the safe basin.

Figure 8 shows the area of the safe basin normalized to the still water basin for a variety of wave steepnesses. The figure shows that the safe basin area remains relatively constant until a critical point where it rapidly begins to decrease. This sharp increase in the safe basin erosion is typical of

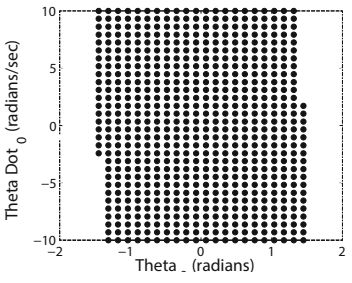


Fig. 8 An example of a safe basin plot. The black dot represent initial conditions which do not result in capsize.

rolling vessels. Ucer [16] takes the drop off point as the maximum allowable wave steepness for operation. Using this convention, the maximum allowable wave steepness for Blackbody Radiation is about 3. This is an extremely large steepness. As noted above, deep water waves are physically limited to a maximum physical wave steepness of about 0.14 before they must break.

This result indicates that Blackbody Radiation would be unlikely to capsize in non-breaking beam waves under the assumptions made in the model. The approximations made in identifying the parameters of the equation of motion add some uncertainty to this conclusion, but the extreme maximal wave height nonetheless suggests that Blackbody Radiation has considerable stability in roll motions.

5 Conclusions

SP-ASVs tend to have stability characteristics that are quite different from more traditional boats. Due to the scaling laws discussed in Section 2, SP-ASVs must have proportionally larger restoring moments than standard sail boats to prevent excessive heel. This means that either the boat must be heavier or have a lower center of gravity. Most groups designing racing SP-ASVs have opted to lower the center of gravity by increasing the depth of the keel and weight of the bulb. Such low center of gravity designs will usually have an unstable equilibrium in the inverted position, which is preferable in the case of capsize.

The higher restoring moment and higher drag coefficients derived from deeper keels and heavier vessels leads to a high degree of stability in beam waves, at least in roll motions. I presented a number of techniques to estimate dynamic parameters of a SP-ASV, and applied them to Olin College's Blackbody Radiation. Using the safe basin technique, I showed that Blackbody's maximum allowable wave steepness is high, indicating a great deal of stability in waves. Thus, the greater stability required for normal operation of SP-ASVs and the higher drag which comes with it serve the vessel well

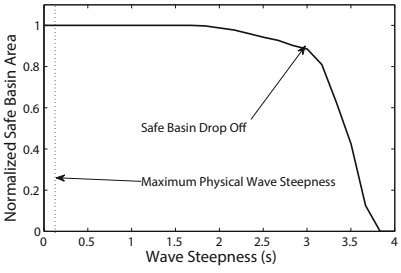


Fig. 9 The safe basin rapidly erodes after s = 3. This is considerably larger than the maximum possible wave height. Using Blackbody's parameters and $\omega = \omega_0$.

in high wave conditions with reefed sails. Using the techniques in Section 4, other research groups can estimate the wave stability of their own vessels.

Secondary environmental challenges such as waterproofing, wind gusting and strong currents, as well as other vessel motions such as parametric roll and un-reefed sail dynamics pose considerable difficulties. More work is required to understand the open-ocean stability of SP-ASVs.

References

1. Balcer, L.: Location of ship rolling axis. Polish Maritime Research, 3–7 (2004)
2. Flazarnao, J.: Predicting complicated dynamics leading to vessel capsize. University of Michigan, Ann Arbor (1990)
3. Fossati, F.: Aero-Hydrodynamics and the Performance of Sailing Yachts. International Marine (2009)
4. Francescutto, A.: Roll-Heave-Sway Coupling in Beam Waves. In: Chung, J., Mohamed, S., Mashashi, K., Toshiaki, S., Seok, W.H. (eds.) Offshore and Polar Engineering Conference, pp. 281–287. ISOPE (2002)
5. Lewis, E.: Principles of Naval Architecture Volume I. SNAME (1988)
6. Long, Z.J., Lee, S.K., Kim, J.Y.: Estimation of survival probability for a ship in beam seas using the safe basin. Ocean Engineering, 418–424 (2010)
7. McCue, L.: Chaotic vessel motions and capsize in beam seas. University of Michigan, Ann Arbor (2004)
8. Miller, P., Brooks, O., Hamlet, M.: Development of the USNA SailBots (ASV). In: IRSC, pp. 9-17. FEUP (2009)
9. Odabasi, A.Y.: Ultimate stability of ships. RINA Transactions 118, 237–262 (1976)
10. Odabasi, A.Y., Ucer, E.: Effect of Initial Bias on the Roll Response and Stability of Ships in Beam Seas. In: Neves, M., Kat, J., Umeda, N., Belenky, V., Spyrou, K. (eds.) Contemporary Ideas on Ship Stability and Capsizing in Waves, pp. 217–228. Springer, Heidelberg (2011)
11. Randall, R.: Elements of Ocean Engineering. SNAME (2010)
12. Sanchez, N.: Nonlinear motion of ships in longitudinal waves. International Shipbuild Program, 247–272 (1990)
13. Senjanovic, I., Cipric, G., Parunov, J.: Survival analysis of fishing vessels rolling in rough seas. Philosophical Transactions of the Royal Society A, 1943–1965 (2000)
14. Senjanovic, I., Fan, Y.: Some advances of the harmonic balance method. Journal of Sound and Vibration, 295–307 (1996)
15. Thompson, J.: Loss of engineering integrity due to the erosion of absolute and transient basin boundaries. In: Price, W.G., Temarel, P., Keane, A.J. (eds.) IUTAM Symposium on Dynamics of Marine Vehicles and Structures in Waves, pp. 313–320. Elsevier, Amsterdam (1991)
16. Ucer, E.: Examination of the stability of trawlers in beams seas by using safe basins. Ocean Engineering, 1908–1915 (2011)

17. Umeda, N., Ohkura, Y., Urano, S., Hori, M., Hashimoto, H.: Some remarks on theoretical modeling of intact stability. In: Neves, M., Kat, J., Umeda, N., Belenky, V., Spyrou, K. (eds.) Contemporary Ideas on Ship Stability and Capsizing in Waves, pp. 217–228. Springer, Heidelberg (2011)
18. Watanbe, Y.: Some contributions to the theory of rolling and its related problems. Transactions INA, 408–432 (1938)
19. White, F.: Fluid Mechanics, 3rd edn. McGraw-Hill (1994)
20. Wu, X., Tao, L., Li, Y.: The safe basin erosion of a ship in waves with a single degree of freedom. In: Australian Fluid Mechanics Conference, University of Sydney (2004)

Part IV
High Level Control Architectures and Algorithms

Multi-agents Decision Making Concept for Multi-missions Applications in Marine Environments

Oren Gal

Abstract. This paper presents unique multi-agents decision making and control concept in marine environments, formulating distributed and centralized approach. We map several major missions and introduce the conceptual implementations using multi-agents: patrolling in predefined area, reaching specific destination, following or monitoring specific object and getting back to home harbor. We present our algorithm scheme with logical concept of each module and simulate agents sensing capability. We demonstrate our concept in several scenarios simulations with advanced test-bed environment. Algorithm performances are also analyzed showing real-time running time.

1 Introduction and Related Work

Multi-agents decision making and control methods can be divided into two major disciplines, centralized and decentralized approaches. The basic idea of centralized approach [2] is to make all the decisions in one place. All tasks are concentrated by a single entity, named 'Central Task Planner and Scheduler' (CTPS). The CTPS translates the tasks into smaller tasks (sub-tasks), which will later be sent to the appropriate agents, according to their capabilities, their assignment and their workload.

Theoretically, the centralized approach appears to do the trick. It allows knowing in advance all the tasks to be done and the connections among them, allows choosing the most fitting decomposing of the problem to sub-tasks. Indeed, this is a significant advantage, as there is no disassembling which will be ideal for all missions. On the other hand, this approach does not fit a dynamic environment, in which unpredictable events may occur.

Oren Gal
Technion, Israel Institute of Technology, Israel, Haifa
e-mail: orengal@techion.ac.il

F. Le Bars and L. Jaulin (eds.), *Robotic Sailing 2013*,
DOI: 10.1007/978-3-319-02276-5_10,

Multi-agents in marine environment usually not in a constant contact with CTPS nor with each other, even though the CTPS requires a continuous stream of data about the forthcoming events in order to provide an effective response. Solutions to this problem (such as placing multiple sensors in the environment) are expensive and hard to apply.

On the other hand, at the decentralized approach, each agent is responsible for a group of tasks, and there is no need using entity such as CTPS. A predetermined disassembling is applied on the problem, and the agents can try to contact each other, in order to improve it. As mentioned above, this solution is problematic, as there is no disassembling which will be ideal for all problems. Despite this fact, the lack of the CTPS allows every agent to process the data it collects by itself, and, for example, plan its own trajectory using local sensors data and decide what the next action is. The benefit of this approach is, of course, the speed of reaction and the independence of the agents. Moreover, it allows real time reaction to dynamic changes in the environment. As said, this is a problematic matter in the centralized approach. While complete review of existing work on multi-agents motion planning is a very vast field, we focus on multi-agents in marine environment.

Y. Guo and L.E. Parker (2002) presented distributed control method of a group of robots. However, this method focused on velocity planning and optimal paths so that the robots will reach their destinations as quickly as possible, without colliding with each other. The authors do not discuss missions that are more complicated than trajectory planning (such as intelligence gathering), as well as the option to change an agent's destination during path autonomously.

Interesting work called The Cocktail Party Model [5], discusses the control of a group of robots without mentioning patrol or complex missions. It offers an entirely distributed approach, where there is no communication between the agents. Each agent plans and alters its destination dynamically whilst moving. The agents avoid static obstacles in the environment, as well as other agents, which are considered as "moving obstacles". This approach still not tackles decision making aspects except trajectory planning. L. Brumitt and Anthony Stentz (1996) introduced centralized approach that deals with complex missions that require more than one agent.

The concept allows alteration of missions after more knowledge of the environment has been accumulated, making it possible to choose a different path, with lower cost. Moreover, the presented method allocates multiple destinations per agent. However, the concept omits patrol or other complex missions beyond trajectory planning. Combination of centralized and distributed approach introduced by [2] and [4]. The method is based on synchronized the movement of the agents regardless specific mission. Recently, Agmon et al. (2011) introduced distribution of patrol points so that the maximum time each point is left unvisited by an agent is minimized. The authors presented centralized approach. However, the authors do not discuss other complex missions than patrol.

This paper presents unique multi-agents decision making and control in marine environment, dealing with combination of the distributed and centralized approach, whilst taking advantage of each of them: agents can plan and execute missions by themselves, whilst communicate with the central unit if need be. We present our algorithm scheme, performances, and demonstrate our concept in several simulations.

2 Decision Making Concept and Algorithm

In this section we present the different modules of our algorithm. We focus on patrol mission of multi-agents in marine environment. We present concept and method to allocate new missions for agents and agents behavior concept in case of discover new data by specific agent adding and changing current mission. Each agent can be modeled as Unmanned Surface Vehicle (USV), including CTPS agent which is one of the agent's group. For our analysis, friendly ship is not an agent nor group member, where communication between the agents can be modeled as Line of Sight (LOS) performances with propagation model in marine environments. In case of mission interference from CTPS, current agent's mission is interrupted till completing updated mission.

2.1 Patrol Point Distribution Module

The patrol point distribution is essential to fulfill patrol mission efficiency. Moreover, by point distribution agents can find and locate data and situation that would change their basic mission, while CTPS is not aware of this data beforehand. The CTPS distributes points between the agents before starting mission. Patrol points determined by operational user. These points are to ones to be visited by agents as frequently as possible, for operational reasons (for example, high chances of finding suspicious targets around them).

The objective is to minimize the time between visits at any of those points. As well known from coverage and patroling algorithms [6], this problem is NP-Hard. Therefore it will not be practical to find the optimal solution. Instead, we decided to give an equal number of points to each agent: every agent (besides the last agent) receives a number of points that is equal to the total number of important points divided by the number of agents, rounded down. The last agent receives the remaining points.

The CTPS initiated by giving the first point to the first agent. Then, it gives it the nearest point that wasn't given already. The CTPS routinely choose the next nearest point that wasn't given already. Once the agent got maximum points, the CTPS moves on to the next agent, until it reaches the final agent, which gets all of the points that were not chosen already. By that,

points are divided to a number of sectors (the same number as the number of agents), and each agent patrol the area that contains points which are close to one another. The patrol point distribution has to be done before the agents start their mission, and it could be divided again by CTPS after they start working, in case of a change to the set of points or to the number of agents.

2.2 Mission Distribution Module

Multi-agents missions can be variant from patrolling in predefined area, reaching specific destination, following or monitoring specific object, go back to home harbor, etc,. Distribution of missions between agents can be done by the agents themselves or by CTPS.

Each agent has independent priority queue of missions, in form of a maximum heap. Once a new mission is given to an agent, it starts at the bottom of the queue, and then works its way to the correct place in it. The agent updates its own active mission once given new mission. Updating mission

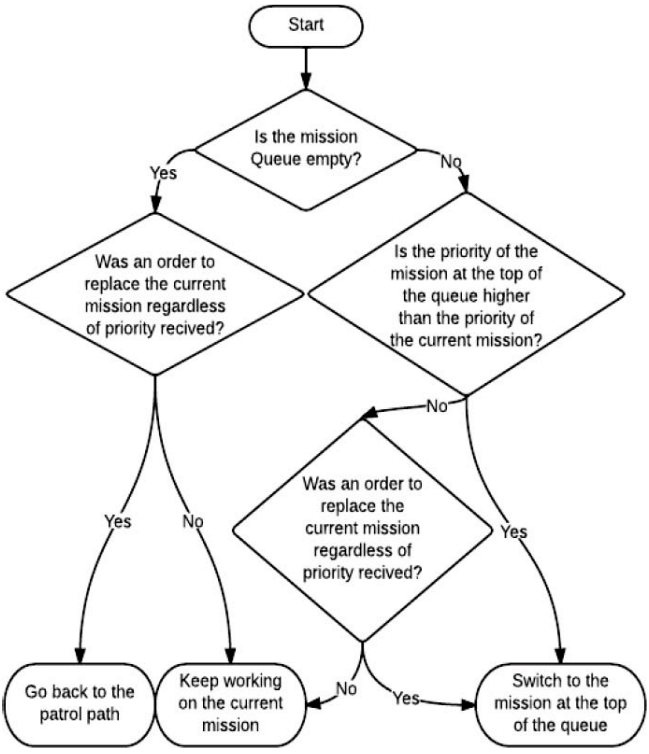


Fig. 1 Mission Distribution Module Flow Chart

can be given by the agent itself (for example, when it finishes working on previous mission from a existing heap or when the agent set mission to himself) or by the CTPS. Along with mission priority, the agent receives top priority concept state: Change mission using only mission priority, or replace current mission regardless to mission priority. The default and basic mission is trajectory tracking loaded beforehand to the agents. Mission distribution module flow chart is presented in Fig. 1.

2.3 Missions Managed by CTPS

Some missions, such as following mother ship or sending vehicles ahead mother ship, are given with missions priority to the CTPS by an outside source. The CTPS receives or decides by itself the number of agents required for each mission, decides which agents would be best for the mission and gives the mission to the appropriate agents. The choice of each agent for

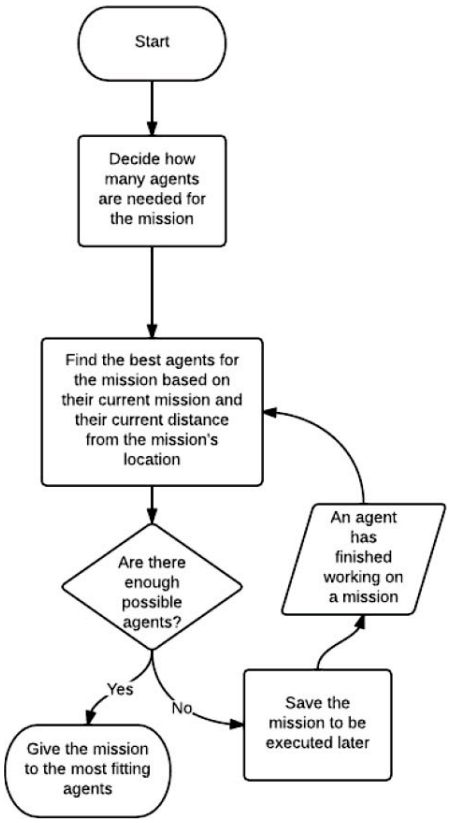


Fig. 2 CTPS Mission Module Flow Chart

every mission is based on the priority of the current mission for each agent (the CTPS would prefer to send agents that are doing less important jobs rather than agents doing important missions so that important missions will be done first, using prioprity level). Agent with minimal distance from agents current position to starting point mission would be preferred by CTPS.The CTPS can also save missions with relatively low priority to the current mission. When an agent complete mission it notifies the CTPS, so CTPS can allocate the agent to start working on a mission that has been saved.

Of course, some missions must be done by the agents without regard to their current task (such as an urgent need to return their home base). The CTPS will send those missions with extremely high priorities and an order to update the mission without regard to the current task. Flow chart of CTPS mission module describing the decision making process is presented in Fig 2.

2.4 Missions Discovered by the Agents

The agents can decided to take on tasks by themselves (for example, when an agents runs into a suspicious target whilst patrolling, it would start monitoring the target immediately). Of course, they notify the CTPS on their mission change, so that it could decide to allocate other agents to work on their original mission.

3 Algorithm Simulation Environment

Our simulation environment consists of grid map with a constant number of agents. Mission can be set as: patrolling in bounded area; following and monitor a suspicious target; escort friendly ship in a constant formation; return to agents home harbor. The default priorities of mission from high to low are: return to the harbor; monitoring of a suspicious target (as the target might leave the area in which the agents are permitted to be before the agents manage to monitor it). After that is the escorting of a friendly ship, and finally patrolling.

Monitoring a suspicious target goes on until the target is approved. Every suspicious target is initialized with time estimation for final targets identification, updated in time. The desired formation to escort friendly ship is defined by the friendly ship itself. The escorts agents receive the desired formation and apply the optimal formation using current number of agents that are available to this mission. Ending escort mission is defined by the ship which updates the CTPS.

The simulation environment takes into account collision avoidance capability between the agents and other obstacles. Patrolling points are by the user as an input to the simulation, and not being change during mission.

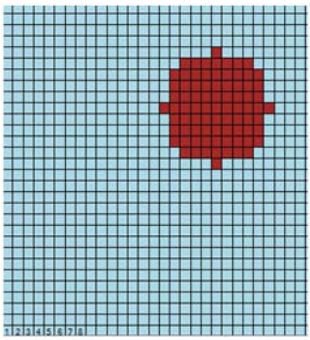


Fig. 3 Eight Agents in their Home Harbor

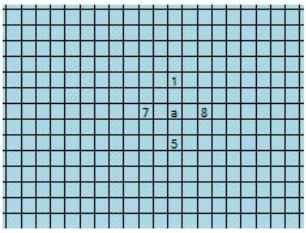


Fig. 4 Four Agents in Friendly Ship Escorting Mission

The algorithm is based on the well-known A* search algorithm in grid for path motion planning and one obstacle avoidance. Agents and suspicious targets are allowed move in cells which are adjacent to obstacle cells, whilst the friendly ships are only allowed move in cells which do not collide an obstacle cell. Each obstacle is represented by 2D disc. We simulate agents sensing capability, so each agent can send a request for information. Simulated agent capability generate nearby obstacles, as well as suspicions targets upon agents request.

3.1 Simulations

We demonstrate one input case that takes into account three major missions of friendly ship, suspicious target and home harbor implemented by eight agents and an obstacle. Simulation environment can be seen in Fig. 8, where the left side dedicatd to the map simulation environment, and the other parts used to set an obstacles and multi-agents mission's definitinos.

In Fig. 3, the eight agents (marked by the digits '1'-'8') in their home harbors (which is the starting point). Obstacle is marked as red circle.

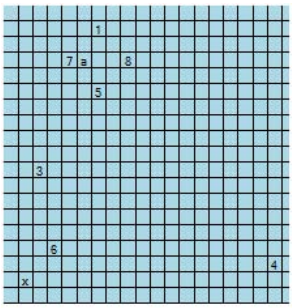


Fig. 5 Monitoring Suspicious Target Discovered by Agents

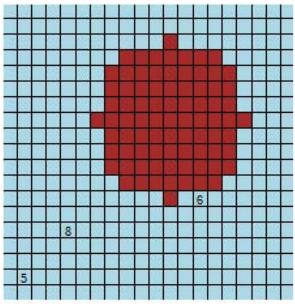


Fig. 6 Priority Mission sent by CTPS Home Harbor

During agents mission, friendly ship asking for escort. Figure 4 shows agents '1', '8', '5' and '7' escorting friendly ship marked by the letter 'a' with ships desired formation.

In Fig. 5, This picture shows the agents '1', '8', '5' and '7' still working on escorting mission around friendly ship 'a'. Meanwhile, agent '6' spotted a suspicious target marked as 'x', (sent upon agents request from agents sensing simulator) started to monitor suspicious. The agent updates CTPS, which allocate agents '3' and '4' to monitor the target.

In Fig. 6, agents completed their escort and monitor missions. CTPS set priority mission to the agents getting back to their home harbors. Agents '5', '6' and '8' are travelling around the obstacle in order to reach their respective harbors.

3.2 Performance Analysis

We have run multiple tests on the program in order to check its efficiency in terms of computational effort and running time. These tests included varying numbers of agents, as well as different patrol points and different missions. We used standard PC Intel Core i7 Q720, 1.60 GHz and 4 GB RAM. In

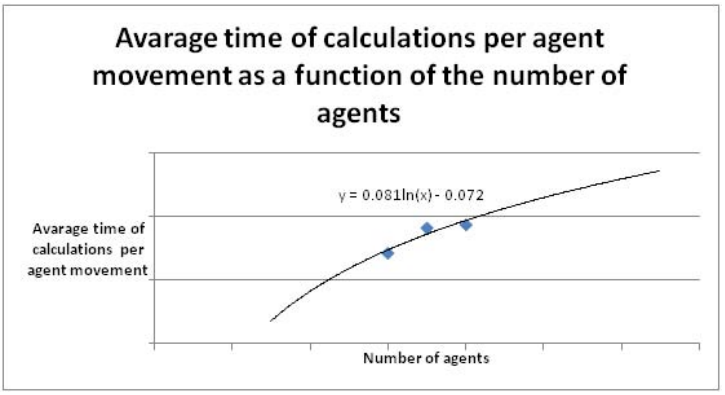


Fig. 7 Running Time vs. Numbers of Agents

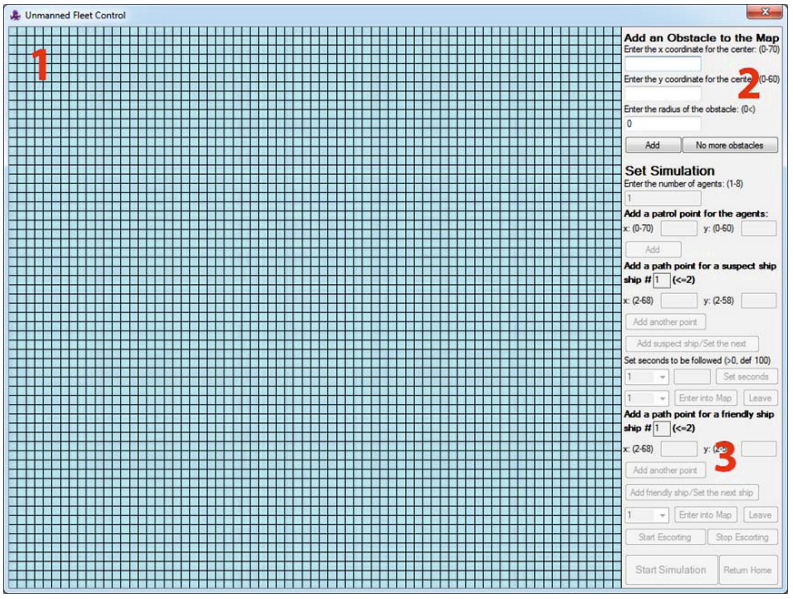


Fig. 8 Simulation Environment

Fig 7, we present running time vs. number of agents. Running time can be approximated as O(ln(n)), which is extremely efficient.

4 Remarks and Conclusions

We presented unique concept for multi-agents decision making for multi missions in maritime environment. Our algorithm consist of several modules,

using CTPS centralized control and distributed control for specific missions, with priority and well define flow chart for logical decision making. We simulate agent sensing capability and tested our basic algorithm decision making capabilities in several scenarios. In order to keep the program efficient, the map is defined as squared two kilometers, which allows displaying all the components and provides enough manipulation area for the agents.

Optimal distribution of patrol points among the agents is a NP-hard problem. Therefore, the given solution is heuristic approximation, yet good enough to achieve the goal. The concept and the approach introduced in this paper benefits from two most common approaches in practice today. It provides independence to the agents and can respond quickly to changes in a dynamic environment, and simultaneously allows communication between the agents in order to optimize the distribution of the missions.

The simulation includes developer GUI and debugging system for each agent as individual entity. We included extended user options to enter dynamic events during runtime, testing agents decision making and behavior. Future work will consider marine vehicle dynamic model, and more accurate model of current, winds and disturbances in maritime environments to approximate our algorithm and agents capabilities to real world environment.

Acknowledgements. The author would like to thank Hen Elimelech and Ron Zeitouny for their help during this research.

References

1. Brumitt, B.L., Stentz, A.: Dynamic Mission Planning for Multiple Mobile Robots. In: Proceedings of 1996 IEEE International Conference on Robotics and Automation, vol. 4, pp. 2396–2401 (1996)
2. Brumitt, B.L., Stentz, A.: GRAMMPS: A Generalized Mission Planner for Multiple Mobile Robots In Unstructured Environments. In: Proceedings of 1998 IEEE International Conference on Robotics and Automation, vol. 2, pp. 1564–1571 (1998)
3. Guo, Y., Lynee, P.E.: A Distributed and Optimal Motion Planning Approach for Multiple Mobile Robots. In: Proceedings of 2002 IEEE International Conference on Robotics and Automation Washington, DC, vol. 3, pp. 2612–2619 (2002)
4. Le Pape, C.: A Combination of Centralized and Distributed Methods for Multi-Agent Planning and Scheduling. In: Proceedings of 1990 IEEE International Conference on Robotics and Automation (1990)
5. Lumelsky, V.J., Harinarayan, K.R.: Decentralized Motion Planning for Multiple Mobile Robots: The Cocktail Party Model. Autonomous Robots 4(1), 121–135 (1997)
6. Agmon, N., Urieli, D., Stone, P.: Multiagent Patrol Generalized to Complex Environmental Conditions. In: Proceedings of the Twenty-Fifth Conference on Artificial Intelligence (2011)

MPL—A Mission Planning Language for Autonomous Surface Vehicles

Henrique M.P. Cabral, José C. Alves, Nuno A. Cruz, José F. Valente, and Diogo M. Lopes

Abstract. In this paper we present the specification of a scripting language for mission planning in Autonomous Surface Vehicles. Besides the specification of common missions based on a sequence of waypoints, the main goal was to extend the scope of application to specific need of autonomous sailing boats. These include, for example, the ability to react to environmental conditions in real time and reprogram accordingly. The proposed Mission Planning Language (MPL) hides the details of underlying control and navigation components, and focuses on the high-level procedures of a mission. We discuss the requirements and principles behind the language and show how oceanographic data collection missions can be significantly improved by such a facility. Finally, we illustrate an application example inspired on a real scenario of the FEUP Autonomous Sailboat (FASt).

1 Introduction

Autonomous surface vehicles (ASVs), in particular autonomous sailboats, provide a particularly interesting platform for oceanographic experiments and data collection. Their potential for large autonomy, together with the ability to fit many different types of sensors and establish constant communication with a remote base station, allow to carry missions of several weeks or months without expensive and time-consuming local monitoring [3, 4].

However, and despite the advances in control algorithms for these vessels in recent years, the specification of a mission remains, from the point of

Henrique M. P. Cabral · José F. Valente · Diogo M. Lopes
Master in Electrical and Computer Engineering, University of Porto,
Faculty of Engineering
e-mail: henrique.cabral@fe.up.pt

José C. Alves · Nuno A. Cruz
Department of Electrical and Computer Engineering, University of Porto,
Faculty of Engineering

F. Le Bars and L. Jaulin (eds.), *Robotic Sailing 2013*,
DOI: 10.1007/978-3-319-02276-5_11,

view of the end users, difficult and error-prone, sometimes requiring case-by-case adjustments to the control software. Presently several applications use friendly graphical user interfaces for specifying, simulating, monitoring and analysing the behaviour of autonomous vehicles [5, 8]. Although such environments hide the complexity of the low level systems and increase the tolerance to faults due to lack of proficiency of operators, they also constrain the variety of possible tasks that the systems are able to accomplish. For example, if a given mission planning system for an autonomous vessel only allows to define a sequence of waypoints, it may be impossible to specify reactions to unanticipated events such as dynamic obstacles or changes in environmental conditions.

Although there has been some effort to provide tools for high level mission planning specification, based on flexible and highly parameterizable commands, these have been mainly focused on motorized autonomous surface vehicles [6]. In the particular case of autonomous sailing boats, a typical mission needs to take into account the environmental conditions, in special the wind and sea state. Furthermore, as sailing boats have the potential for long term deployments, they may end up facing weather conditions that are impossible to predict accurately when the mission is planned. The ability to react to real time data and reprogram the remaining tasks according to these dynamic conditions is essential for successfully completing a mission and it is specially critical when safety is an issue.

In this paper we describe a mission planning language (MPL) capable of both high-level specifications and also low level control over the system operation. In this way, all control details and navigation restrictions (such as wind conditions in sailboats) are hidden from the user, and the same mission script could be executed transparently by both a sailboat and a powerboat. We present the language requirements and specification, as well as a discussion of the design principles behind MPL and some real use cases based on previous missions with the FEUP Autonomous Sailboat (FASt) [1].

2 Requirements

As stated in the introduction (Sect. 1), the main requirements for MPL are ease of use and, simultaneously, providing experienced users with the ability to specify lower-level control:

Control-oriented. Unlike the detailed algorithms underlying the operation of the vehicle, a mission plan typically involves very few calculations or complex data structures. Instead, the user should be able to specify simple[1]

[1] From their point of view, not the system's. Often, what seems simple to a user ("follow a circle") may require more or less complex maneuvers by the vehicle. These should be entirely transparent.

actions directly, as well as higher control structures (loops, conditionals) over these actions.

Modularity. Often during a mission—in particular during data collection—the same composite action must be repeated at different points in time and space. The language should provide mechanisms for the definition and reuse of such actions, such as functions or procedures.

Abstraction. Most missions should not have any need to access lower-level features of the system, such as direct actuator control. Instead, common primitives should conceal the user from such details, guaranteeing stability on the face of hardware and control system alterations.

Transparency. At the same time, more demanding applications should still be able to tweak the core functionality to satisfy their own requirements. This includes direct hardware access, as well as system state variables and control procedures.

Taking into account the usage context, both from the user's and the implementer's perspective, other requirements appear:

Instant feedback. Since operator failure or error in specifying the mission may easily lead to loss of data, time, or even the vehicle, it is vital that they receive immediate and accurate feedback in order to evaluate their actions.

Extensibility. If the language is to be used in a wide variety of ASVs, the designer cannot presume to foresee every possible action the user may want to perform. MPL should, therefore, provide mechanisms to define new commands and structures, particularly for hardware control.

Resource efficiency. Given the reduced resources under which many ASV systems operate, the language should be easy to parse and have a compact representation for internal use.

Easy integration. The language must be easy to couple with existing control systems, as well as accurate simulation environments for fast evaluation of the navigation pattern determined by a navigation script. This is particularly important for sailboats, where the true navigation path cannot be entirely planned due to the mostly unforeseeable wind and sea conditions at the time of the mission.

All considered, the requirements exposed above point to a command-based, interpreted language, with uniform parsing and a simple interface with the control system, as well as higher-level features such as loops and conditionals. As such, we have decided to model MPL on Tcl [7], which provides a well-tested boilerplate obeying several of the requirements.

3 The Language

MPL is, by design, a domain-specific language (DSL), in that it provides facilities especially designed for the control of ASVs. On the other hand, taking into account the variety of tasks one may want to perform in this category, it must also allow the creation of even more specialised languages. This thought informs not only the syntax but also the primitives supplied by the language. These will be presented and discussed in this section.

3.1 Execution Context

Within MPL, the *execution context* is defined as the set of all procedures (Sect. 3.3) and variables (Sect. 3.2) accessible to the user script at the current point in time. Separate user contexts are used between global execution and procedures, allowing the user to define local variables and nested procedures.

3.2 Variables

MPL variables can be either user variables or system variables. The distinction between string, numerical, or boolean types exists only for the values a user variable may take[2]. Strings may be of arbitrary length and are encoded internally as UTF-8; numerical values can be either real (stored as double precision IEEE 754 floating-point values) or integer (stored as 64-bit signed integers); and boolean values are stored as unsigned 8-bit values, with 0 interpreted as "false" and anything else as "true".

User variables. These correspond to regular, user-defined variables. They may take up any value of the types described below, and the type of value stored may change over the course of execution. User variables exist solely within the context of the script.

System variables. To access or modify any external system data (such as configurations, sensor values, or actuations), system variables must be used. These have direct correspondence with memory locations in the system and will always reflect the current value at those locations. Together with system-specific primitives, they make up the interface between user script and system code that is specific to MPL.

[2] The lack of an array or list value type is due only to the simplified syntax used (see also Sect. 3.4).

3.3 Procedures

MPL supports user-defined procedures through the `proc` primitive (Sect. 3.5.1). The initial execution context for a procedure contains, besides the procedures defined in the calling context, only system variables and the variables defined at the beginning of the procedure (i.e. those corresponding to its arguments). This means that no outside user variables are accessible by default. To bring external variables into the current execution context, the procedure may invoke the `extern` primitive. Additionally, procedure definitions may nest, since each is defined relative to the current execution context (see also Sect. 3.1).

Procedures may end and return a value through the `return` primitive. If no `return` statement is given, the value of the last executed statement is returned.

3.4 Syntax

A MPL script is a string, consisting of a series of commands separated by newlines. Each command, in its turn, is composed of a series of words, separated by whitespace (except newlines). The first word of a command is used to determine the procedure to be executed, which is passed the remaining words as arguments. In the particular case that the first letter of the first word of the command is a hash ("`#`"), the remainder of the line is ignored.

To enable the construction of complex commands, MPL provides two features. Both work by literal substitution into the initial command, forming a single word.

Command nesting. Anything between brackets ("`[]`") is considered to be a script in itself and is executed within the current context. Once everything up to the closing bracket is executed, the result of the last command is substituted for the original text.

Variable substitution. MPL variables can be referred using the dollar sign ("`$`"). Whenever the symbol occurs and is followed by a valid variable name (composed by alphanumeric characters and/or underscores), the value of that variable is substituted into the word.[3]

The third type of substitution is used to include special characters without attaching to them their usual meaning:

Backslash substitution. Whenever a character is preceded by a backslash ("`\`"), it is inserted in the word and the backslash is eliminated. This includes

[3] As mentioned before, there is no particular syntax for variables containing array values. A possibility would be to use, as in Tcl, `$var(<index>)`, but this has not yet been implemented.

characters that would otherwise hold special meaning, such as dollar signs or brackets.

All three substitutions are performed left to right, and no character is parsed more than once. This way, once a substitution is made, it will not trigger other substitutions. Furthermore, the result of a substitution is always included in the current word, never separated into two or more words.

The two remaining constructions, while somewhat redundant, provide a more convenient way of writing complex words without resorting to escaping (using backslashes) all whitespace or special characters:

Double quotes. Mainly intended for character strings. If a word starts with a double quotation mark (“`"`”), it is only terminated by the next double quote. Everything between the two marks is included in the word, but not the marks themselves.

Curly braces. If a word starts with an opening brace (“`{`”), everything until the next closing brace (“`}`”) is considered to be part of the word (except for the braces themselves). Unlike in quote-enclosed strings, no substitutions are performed.

This concludes the exposition of the language syntax. All of these constructions aim, in one way or another, to improve the flexibility of the basic command structure while maintaining syntactic “sugar” low and parsing simple. Additional syntax and meaning can always be added by specific commands, either native or user-defined.

3.5 Primitives

MPL primitives are designed to achieve two main goals: first, to enable the use of basic language features (namely, variables, control structures, and procedures); and second, to provide elementary navigation and control for the ASV. Accordingly, they will be presented in this section according to their purpose and functionality.

Note: in the following, `[<param>]` denotes an optional parameter, while `<par1|par2>` denotes a choice between two parameters.

3.5.1 Basic Language Features

`set` `<name> [<value>]`
Return the value of the variable `<name>`. If `<value>` is given, set it as the value of the variable. The variable is created if it doesn't exist.

if <cond> <body1> [<body2>]
The condition <cond> is evaluated (in the same way as an expr argument—see Sect. 3.5.4), and its result is interpreted as a boolean value (true or false). The statements in <body1> are executed if the result is true. If <body2> is given, it is executed if the result is false.

while <cond> <body>
The condition <cond> is evaluated, and its result is interpreted as a boolean value. If true, the statements in <body> are executed. The condition is reevaluated at the end of each execution, stopping only when it becomes false.

proc <name> <args> <body>
Create a new procedure <name>. Whenever the procedure is invoked as part of a command, the contents of <body> will be executed, with the supplied arguments being assigned to the local variables with names listed in <args>.

extern <var>
Look for the variable <name> in the global execution context, and bring it into the current context.

return [<val>]
Return from the current procedure. If <val> is specified, it's used as the return value for the procedure. Otherwise, the return value is empty.

halt
Stop execution of script.

3.5.2 Navigation

coordinate <lat> <lon>
Returns a coordinate value for the specified latitude/longitude pair (in decimal degrees), to be assigned to a variable such as a waypoint, or used in other navigation commands.

add <coord> <dist> <direction>
Produces a new coordinate from <coord> by adding the indicated distance, <dist> (in metres), in the specified direction (angle in degrees, clockwise from North).

Note: other convenience primitives could be included here, such as straightforward unit conversions (nautical miles to kilometres, decimal degrees to DMS, etc.).

3.5.3 Control

go <coord> [<body>] [-at]
Direct the vehicle to the indicated point <coord>. If <coord> contains more than one point, follow the points in sequence. Optionally, if a <body> is supplied, its commands are executed in parallel while the final point hasn't been reached. The -at option can be used with a <body> to execute it at every point.

follow <var> [<body>]
Instruct the vehicle to follow a moving point. The variable <var> is read once per control loop to find the point coordinates. If the optional <body> is specified, it is executed in parallel with the command.

station <coord> <dist> [-square]
Keep the vehicle within <dist> of the point <coord>. The strategy to keep the vehicle in place is implementation- and vehicle-dependent (such as following "eights" for a sailboat or motor control for powerboats). The metric can be changed to keep the vehicle within a square of side <dist> using -square.

stop
Make the vehicle come to a stop, possibly by maneuvering to a favourable position first. Note that currents and wind may still affect the position.

3.5.4 Miscellaneous

expr <str>
Evaluate the result of the arithmetic expression contained in the string <str>. Supports basic arithmetic operators (+, -, *, /) and relational operators (==, !=, <, <=, >, >=), as well as parenthesis grouping. Arithmetic operators take precedence over relational operators, and precedence among them is as usual. Associativity is left-to-right. Whitespace does not affect evaluation.

log <str>
Write text string <str> to log, together with timestamp. Depending on the implementation, this may be a local log or transmitted to a base station.

time [-iso|-asc]
Return the current system time as a 64-bit Unix time value. If the -iso option is used, the time is returned instead as an ISO 8601 string[4]. If the

[4] <YYYY>-<MM>-<DD>T<HH>:<MM>

`-asc` option is used, the time is returned as a string in the format of the standard C library function `asctime`[5].

`timer` `<interval> [<body>]`
Start a timer with a duration of `<interval>` milliseconds. The procedure returns only when the time is reached. If `<body>` is specified, it is executed when this happens.

`run` `<file>`
Load and execute the MPL script contained in `<file>`. The script is executed within the current context.

4 Use Case Analysis

In order to showcase the flexibility and applicability of MPL, we will now present a real situation based on a mission with FASt. In this mission, the boat was fitted with an electrical winch at the stern and a hydrophone was attached to the end of the spool line. The objective was to sail to a set of predetermined points, stopping at each point to lower the device to three different depths for a certain length of time.

Typically, the command interface for FASt would not allow such detailed procedures to be programmed, forcing us to adjust the existing communication protocols with mission-specific changes. This was neither desirable nor elegant. However, using MPL to script the mission, the task becomes straightforward. The used code may be found in Appendix. Some things worth remarking:

- The convention used for FASt MPL is that all system variables have names beginning with an underscore ("_"). In particular, `_position` contains the current GPS position of the boat, `_depth` contains the current hydrophone depth, and `_battery` contains the battery charge in percentage (0-100).
- Since actuator commands work through memory-mapped registers, some wait cycles are required to allow the system to reach the desired state (line 22).
- Nesting the procedure `lower` within `measure` makes sense for this application, since it isn't required anywhere else. If finer-grained control was necessary, it could be brought outside without changing any other part of the script.
- The usage of variables named `depth_1`, `depth_2`, and `depth_3` strongly points to the usefulness of a list or array construction. This was recognised

[5] `<Www> <Mmm> <dd> <hh>:<mm>:<ss> <yyyy>`

as an important feature early in development, but hasn't yet been included to keep the syntax and parsing as simple as possible (see footnotes 2 and 3). As mission scripts grow larger, it will quickly become necessary to include it.

5 Conclusions

In this paper we presented a mission planning language (MPL) that provides a simple to use and flexible framework for planning high-level missions for ASVs. Although this is in an early stage of development, a significant set of features have already been seamlessly integrated into the FASt autonomous sailing boat. This proved to be an easy way to program complex missions that react to dynamic conditions perceived from the on-board sensors in real time. MPL is also well-suited to integration with high-level mission simulators, so that users can plan a complete mission offline with a high degree of confidence in the correctness of the real behaviour of the vehicle.

We believe MPL has the potential to be used in all ASV command systems, but also to be expanded to include other autonomous vehicles, such as AUVs and UAVs. For this, the existence of a standard implementation of the basic language is paramount, and that will be the objective of future developments.

Appendix

```
# useful constants
set winch_perim [expr "2*3.14159*0.2"]

# define a square to take measurements at the vertices
set sw_corner [coordinate 38.408137 -9.134102]
set se_corner [add $sw_corner 1000 90]
set ne_corner [add $se_corner 1000 0]
set nw_corner [add $sw_corner 1000 0]

# define measurement depths (metres)
set depth_1 10
set depth_2 20
set depth_3 30

# get initial position to return to
set home $_position

# initial battery check
if {$_battery < 30} {
    log "Low battery, refusing to perform mission"
    stop
    halt
}

# go to a point and perform measurements while stopped
proc measure {point} {
   # lower hydrophone to given depth (in metres) and wait some time (minutes)
   proc lower {d t} {
        extern winch_perim
        set _winch [expr "($d-$_depth) / $winch_perim"]
        while {$_depth < $d} { }
        timer [expr "$t*60000"]
    }

    # go to given point and lower hydrophone to specified depths
    log "Going to point ($point)"
    go $point {
        log "Point reached, lowering hydrophone to $depth_1"
        lower $depth_1  1
        log "Lowering to $depth_2"
        lower $depth_2  1.5
        log "Lowering to $depth_3"
        lower $depth_3  2
        log "Raising hydrophone"
        lower 0         0
    } -at
}

# main mission
measure $sw_corner
measure $se_corner
measure $ne_corner
measure $nw_corner

# return home
go $home {stop} -at
```

References

1. Alves, J.C., Cruz, N.A.: FAST—an autonomous sailing platform for oceanographic missions. In: Proceedings of the MTS-IEEE Conference—Oceans'(2008)
2. Benjamin, M.R., Schmidt, H., Newman, P.M., Leonard, J.J.: Nested autonomy for unmanned marine vehicles with moos-ivp. Journal of Field Robotics 27(6), 834–875 (2010)
3. Caccia, M.: Autonomous surface craft: prototypes and basic research issues. In: 14th Mediterranean Conference on Control and Automation, MED 2006, pp. 1–6. IEEE (2006)
4. Cruz, N.A., Alves, J.C.: Autonomous sailboats: an emerging technology for ocean sampling and surveillance. In: OCEANS 2008, pp. 1–6. IEEE (2008)
5. Meier, L., Tanskanen, P., Heng, L., Lee, G.H., Fraundorfer, F., Pollefeys, M.: Pixhawk: A micro aerial vehicle design for autonomous flight using onboard computer vision. Autonomous Robots 33(1-2), 21–39 (2012)
6. Newman, P.M.: Moos-mission orientated operating suite. Massachusetts Institute of Technology. Tech. Rep 2299(08) (2008)
7. Ousterhout, J.: Tcl-a universal scripting language. lecture at MIT (1995)
8. Zurich, E.: Qgroundcontrol: Ground control station for small air land water autonomous unmanned systems (2013), `http://qgroundcontrol.org/` (accessed July 2013)

Author Index