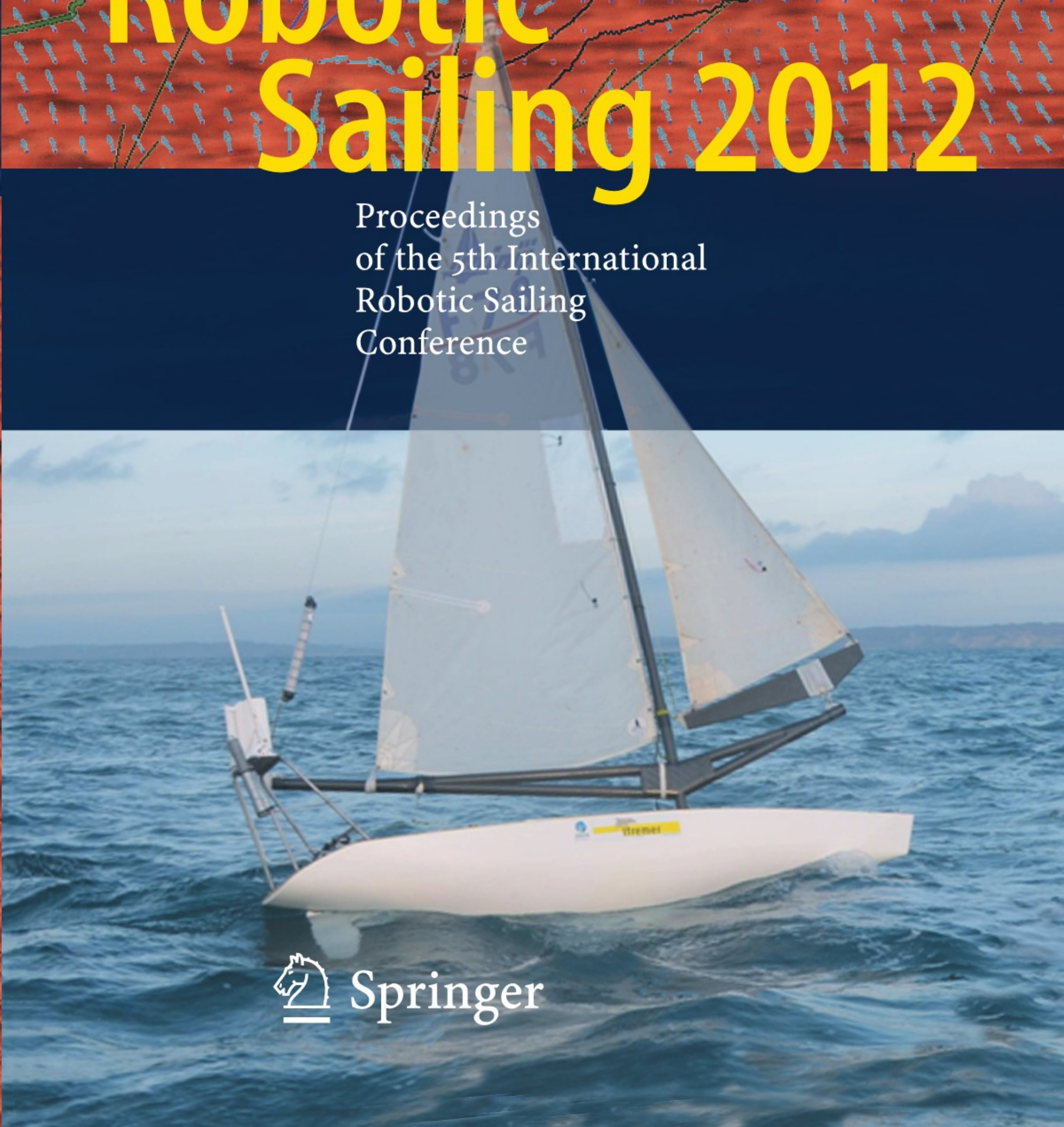
Colin Sauzé
James Finnis
Editors

Robotic Sailing 2012

Proceedings
of the 5th International
Robotic Sailing
Conference



Springer

Robotic Sailing 2012

Colin Sauzé and James Finnis (Eds.)

Robotic Sailing 2012

Proceedings of the 5th International Robotic Sailing Conference

Springer

Editors

Dr. Colin Sauzé
Department of Computer Science
Aberystwyth University
Penglais
Aberystwyth
SY23 3DB
United Kingdom

Mr. James Finnis
Department of Computer Science
Aberystwyth University
Penglais
Aberystwyth
SY23 3DB
United Kingdom

ISBN 978-3-642-33083-4 e-ISBN 978-3-642-33084-1
DOI 10.1007/978-3-642-33084-1
Springer Heidelberg New York Dordrecht London

Library of Congress Control Number: 2012945749

Printed on acid-free paper

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

Preface

The IRSC 2012 is the 5$^{\text{th}}$ conference to have taken place on the subject of robotic sailing since 2008. These were preceded by two meetings of robotic sailors at the warm-up events for the Microtransat Challenge in 2006 and 2007. In this time the number of teams interested in robotic sailing has increased from just three in 2006 to over 20 today. Three attempts have been made to cross the Atlantic autonomously in the Microtransat Challenge, unfortunately none of these have been successful. Most recently a team from ENSTA-Bretagne in France attempted the crossing, but a failure in their tracking system caused them to become disqualified from the Microtransat. Although, there is still a hope the boat might reach its destination.

These failures only serve to highlight the difficulty involved in robotic sailing and the need to bring together many disciplines including naval architecture, systems and electrical engineering and computer science. To build a successful robot which can remain at sea for months at a time, a boat will need to be physically robust, able to sail in all conditions and feature the most robust and fault tolerant electronics and software.

These proceedings summarise the current state of the art in robotic sailing. They are split into three parts: part one presents work on collision avoidance and route planning, part two presents work showing extended field trials of real sailing robots and part three covers the design of sensors, controllers and rigs.

The IRSC was only possible with the help and co-operation of all the authors, the program committee, all our sponsors and the staff at Cardiff Bay Yacht Club, who are hosting the conference and competition. A special thanks must go to Jim Finnis, who only started work for Aberystwyth University three weeks before the proceedings were submitted. Upon starting work he was given the daunting task of proof reading every paper word by word and building a (sometimes very long) list of corrections to send back to the authors.

July 2012 Colin Sauzé

Organization

General Chair

Colin Sauzé	Aberystwyth University

Proceedings

Colin Sauzé	Aberystwyth University
Jim Finnis	Aberystwyth University

Program Committee

Colin Sauzé	Aberystwyth University, UK
Paul Miller	US Naval Academy, USA
Alexander Schlaefer	Universität zu Lübeck, Germany
Luc Jaulin	ENSTA Bretagne, France
Jan Sliwka	ENSTA Bretagne, France
Roland Stelzer	INNOC, Austria
Karim Jafarmadar	INNOC, Austria
Hannes Hassler	INNOC, Austria
Nuno A. Cruz	Faculdade de Engenharia da Universidade do Porto, Portugal
José Carlos Alves	Faculdade de Engenharia da Universidade do Porto, Portugal
Benedita Malheiro	Instituto Superior de engenharia do Porto, Portugal
Ralf Bruder	Universität zu Lübeck, Germany
Ole Blaurock	University of Applied Sciences, Lübeck, Germany

Contents

Part I
Collision Avoidance and Route Planning

Tracking Objects Using PHD Filter for USV Autonomous Capabilities

Oren Gal and Eran Zeitouni

Abstract. Most of the work on automatic detection tracking and classification of unmanned applications over the past twenty years has been focused on ground and aerial vehicles. Recently, the research has also focused on unmanned surface and underwater vehicles for autonomous capabilities.The ability to recognize and identify obstacles becomes more essential with USVs autonomous capabilities, such as obstacle avoidance, decision modules, and other Artificial Intelligence (AI) abilities using low cost sensors. This paper presents multi-target automatic algorithm stages to acquire, identify, and track targets from an Unmanned Surface Vehicle (USV) located in marine environments with LIDAR sensor challenging clutter. We present several clutter models and formulations to handle clutter phenomena. We propose the Probability Hypothesis Density (PHD) Bayes filter, challenging clutter for multi-target tracking.

1 Introduction

One of the most difficult challenges for USV navigation is the recognition of obstacles around the vehicle without human intervention. This task is known as Automatic Target Detection (ATD). An efficient ATD system should achieve a high detection percentage for targets while maintaining a minimal false-alarm rate. This means that it must preserve an optimal balance between a high detection rate and a low error probability. However, ATD algorithms are very sensitive and unstable regarding clutter elements, i.e. elements that are not targets but still part of the scenes with similar characteristics to the targets. Dealing with clutter in ATD algorithms and multi-target environments have been extensively studied [5, 6, 7, 8, 9]. Common, well-known methods try to separate between the targets and noises with Blind Source Separation (BSS) [9, 10, 11].

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

Eran Zeitouni
Ort Braude College, Israel

Several methods with advanced filter characteristics proposed over the last few years, such as the Joint Probabilistic Data Association Filter (JPDAF) [5], Multiple Hypotheses Tracking (MHT) [6], and the multi-target particle filter [7, 8] use observations weighted by their association probabilities. A new and efficient formulation without an explicit base between measurements and targets is Random Finite Sets (RFS) [9], demonstrated in [10]. The main idea behind RFS is based on two different kinds of collections. The first consists of the individual targets and is called the 'set-valued state', while the second consists of the individual observations and is called the 'set-valued observation'. This kind of modeling allows allows the estimation of the targets in the presence of clutter in a Bayesian filtering framework [9, 10, 11]. Advanced RFS-based filters such as the multi-target Bayes filter, the Probability Hypothesis Density (PHD) filter [9, 10, 12] and their implementations [12], have also generated substantial interest.

In our paper we demonstrate tracking and decluttering targets in marine environments by using these kinds of filters. One of the key tasks which intelligent autonomous marine craft have to perform is safe and efficient navigation, which depends directly on the reliable perception of the environment. A high quality sensor is required for close to mid ranges, as a substantial part of the detection and alert belt of the USV lies within this range. As with many other sensors, the primary limitations of LIDAR sensors in marine environments are related to clutter.The Velodyne HDL-64E 3D-LIDAR provides 3D range scans [1]. Typical data in marine environments with sea clutter can be seen in Fig. 1. Therefore, it ought to be either filtered, or the obstacles distinguished by an efficient algorithm using a PHD filter to detect and track multiple targets.

2 Advanced Clutter Models

Exact knowledge of sea clutter is highly important in target detection and classification, and of course permits an efficient tracking. Many algorithms use static clutter models for target detection; an extensive study can be found in [14]. We introduce the main concepts of clutter models which aim to predict sea clutter. First, the 1D Stochastic model, which is an extension of the classical approach. It relies on the phenomenological model of the dynamics of the sea, as can be seen in Fig. 2. Basically, two kinds of waves are encountered at the surface of the sea, generated by two different mechanisms, capillary waves and gravity waves. Capillary waves are generated by the influence of the wind and express the surface tension of the water, while gravity waves are mainly generated by the accumulation of gravitational forces and are the main energy carrying factor. The combined effects of capillary and gravity waves over the scattered electromagnetic waves translate into a composite echo, which is the sum of two components- one having a Gamma PDF (Probability Density Function), corresponding to a large scale, slow varying physical structure, and the other having a Rayleigh PDF, corresponding to a small scale, rapid varying physical structure as can be seen in Fig. 3. The advantage of this model is the compatibility with the existing radar processor and detection algorithms [4]. However,

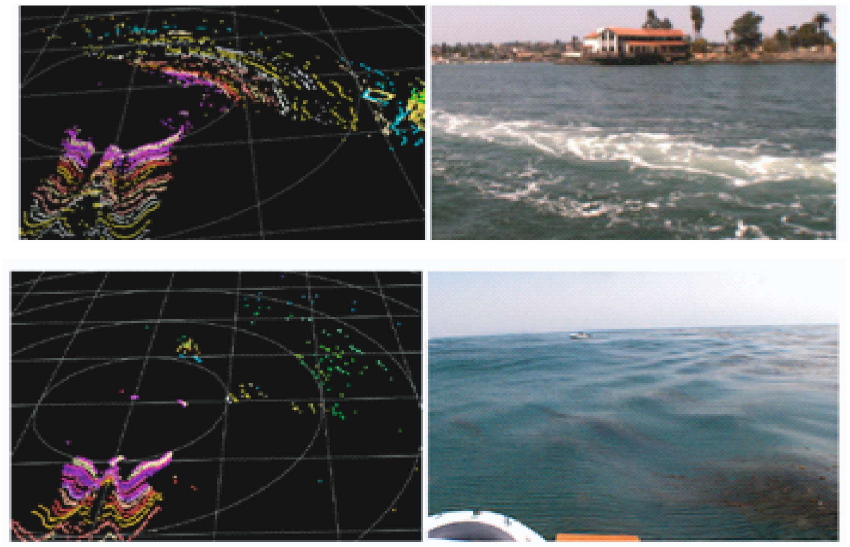


Fig. 1 Velodyne HDL-64E 3D laser scanner LIDAR in Marine Environment [Velodyne Records]

this model has been stretched to the limit- it's quite difficult to obtain a valid set of parameters for an assumed PDF from recorded sea clutter samples, leading to poor results. Moreover, it is not logical to describe the sea using one variable. However, adding more dimensions removes simplicity.

Second, is Texture Realization, which ignores the classic detection algorithms. A set of real data representing measured sea clutter samples is recorded. Then, this information is used to extract a "mask" filter of the clutter, which would permit the reproduction of its stochastic and correlation properties, using a completely new technique. This approach seems quite promising, even though it depends on the training performance of neural networks. More work on real data is required before credibly validating it.

The third is the chaotic model, which assumes that the processes involved in sea clutter generation are non-random, but purely deterministic phenomena. This approach also requires a new radar processor paradigm. Results obtained from one of the prototypes of this model seems are very promising [15], achieving perfect detection. More real data validation and noise robustness are required for credible validation. Moreover, this model involves fractals (non-integer dimensions). The chaotic model is not a trivial one for the standard detection models, due to the statistical character of the model. The classic chaotic model introduced by [15] also known as the Exponential Sensitivity to Initial Condition (ESIC) claims that two systems governed by similar terms will have divergent evolutions, even in similar initial conditions. The model can be expressed as an exponential relation. Let $c(0)$

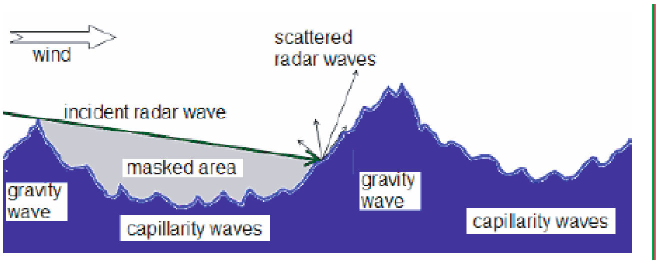


Fig. 2 The phenomenological model of sea surface and interaction with electromagnetic waves [14].

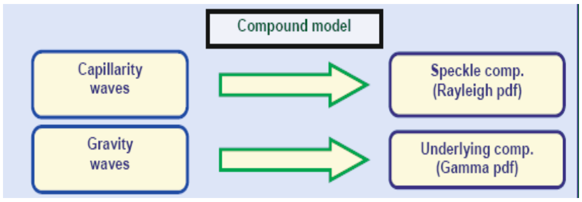


Fig. 3 The compound model of the sea clutter [14].

be the small separation between the initial condition of the two systems (at time $t = 0$). Then, the separation between their states at the time t can be written as:

$$c(t) \approx c(0)e^{\lambda t} \tag{1}$$

where λ is a positive quantity known as the Lyapunov exponent.

3 PHD Object Tracker

As a result of the inefficiency and inaccuracy of all the models above, there is a need for a different approach to track multi-objects. The objective of the multi-object tracking problem is to estimate the state of an unknown number of objects, based on the measurements of the objects corrupted by noise, in the presence of clutter. The classical approach for solving this problem is to apply a stochastic filter such as the Kalman filter [2, 3] or its variants to each object, and use a data association technique such as the Nearest Neighbor to assign the appropriate measurement to each object and track each object separately. An alternative and a more elegant approach is to consider the multi-object set as a single meta-object and the measurements received by the sensor as a single set of measurements, and model them as Random Finite

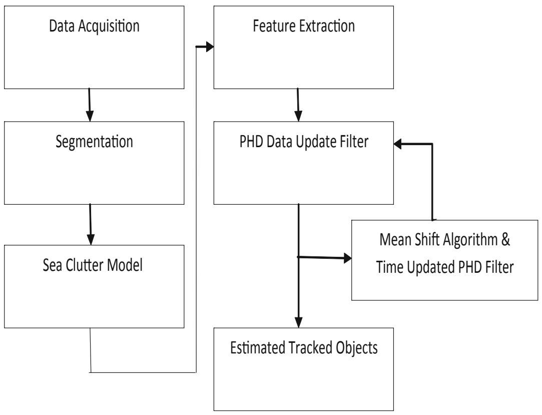


Fig. 4 Block diagram of PHD Tracker Filter

Sets (RFS). This allows multiple objects to be estimated in the presence of clutter, and any data association uncertainty to be cast in a Bayesian filtering framework. Optimal Bayesian multi-object tracking is not yet practical due to its computational complexity. However, a practical alternative to the optimal filter is the Probability Hypothesis Density (PHD) filter, which propagates the first order statistical moment of the full multi-object posterior distribution. The original algorithm is intractable, thus a recursive algorithm which propagates the posterior intensity is employed, which involves Gaussian mixtures. The different stages of this method are described in Fig. 4. We tested our algorithm in simulations with recorded data from Velodyne LIDAR using 1.8GHz Intel Core CPU.

3.1 *Stabilized 3D LIDAR*

The Velodyne HDL-64E provides 3D range scans by rotating an array of 64 beams around its vertical axis producing around 1.2 million points per second. Usually, the sensor is mounted and stabilized on top of the mobile platform providing range scans with a full FOV in horizontal direction. In case of unstabilized LIDAR, USVs roll pitch and heave are compensated for using IMU measurements and the GPS location is part of the Lidar inputs . In the horizontal direction, the array provides 360 degrees field of view (FOV) with an angular resolution of approximately 0.09 degrees. Vertically, the pitch angles range from -24.8 to +2 degrees. The Velodyne HDL-64E LIDAR can detect a target of one meter in length from a distance of 100 meters. Its range measurement accuracy typically is within 10 cm.

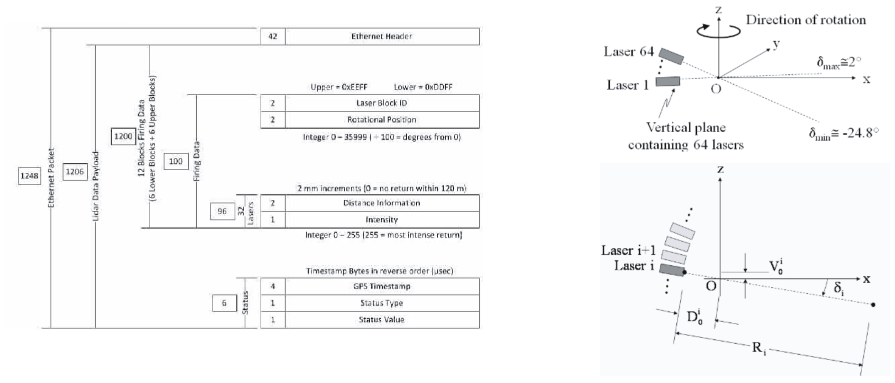


Fig. 5 The UDP Packets Structure and Axes Parameters [1]

3.2 Data Acquisition

The 3D point cloud data from each scan is projected onto a cylinder whose axis is the rotational axis of the LIDAR. This projection yields a range image, whose pixel intensity values correspond to the distance measurements. This is a standard way to represent LIDAR data in different terrain, commonly used in applications such as aerial vehicles for urban terrain modeling [1]. The LIDAR use UDP structure data to the main computer, UDP packets structure and axes parameters on the USVs can be seen in Fig. 5.

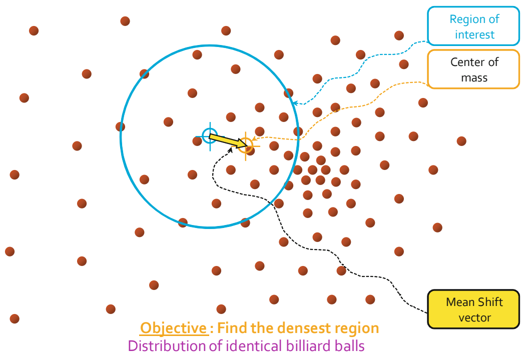

Region of interest
Center of mass
Mean Shift vector
Objective : Find the densest region
Distribution of identical billiard balls

Fig. 6 Mean Shift Algorithm Illustration - Step 1

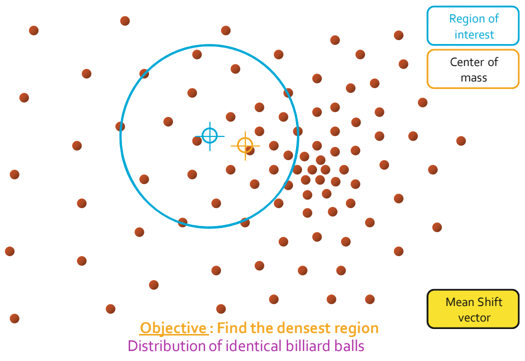

Region of interest
Center of mass
Mean Shift vector
Objective : Find the densest region
Distribution of identical billiard balls

Fig. 7 Mean Shift Algorithm Illustration - Step 2, Location of center of mass and center of region

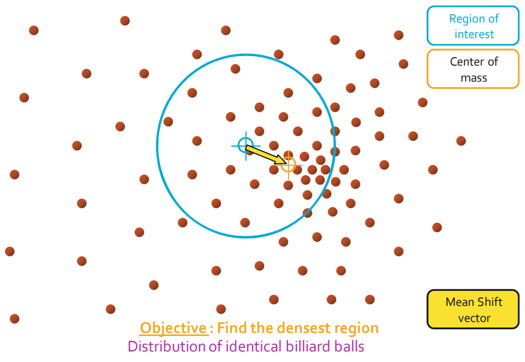

Region of interest
Center of mass
Mean Shift vector
Objective : Find the densest region
Distribution of identical billiard balls

Fig. 8 Mean Shift Algorithm Illustration - Step 3, Mean shift vector from center of mass and center of region

3.3 Segmentation

The range image is segmented using a mean shift segmentation technique. It consist of two steps: mean shift filtering of the original range image data, followed by clustering of the filtered data points. The centroids of the segmented cluster are used as measurement z axis values to update the PHD filter prediction. We illustrate our mean shift algorithm on a distribution of identical billiard balls, which is identical to LIDAR 3D range scans. We can see in Fig. 6 the region of interest as same as the range of the LIDAR in our case, and the center of mass (such as the 3D range scans). Mean Shift is proportional to the normalized density gradient estimate

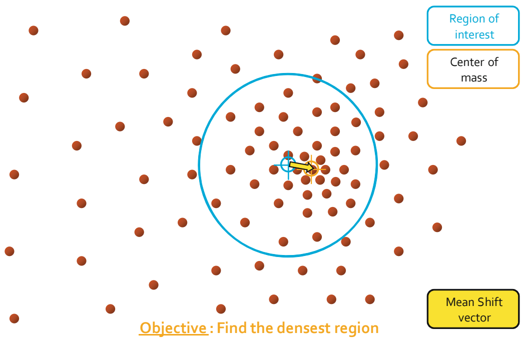


Fig. 9 Mean Shift Algorithm Illustration - Step 4, Mean shift vector from center of mass and center of region

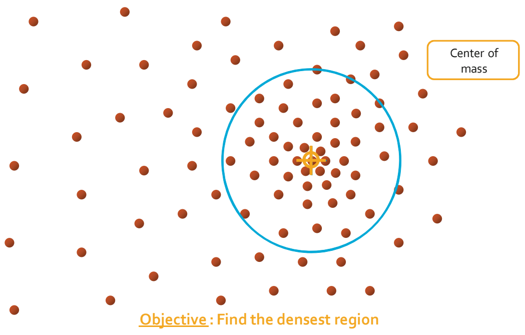


Fig. 10 Mean Shift Algorithm Illustration - Step 5, Convergence of center of region to center of mass

obtained with kernel. The mean shift vector is change as can be seen in Fig. 7 - Fig.9, until convergence can be seen in Fig. 10.

We introduce our initial results of mean shift segmentation implemented on two test cases. In Fig. 11 and Fig. 12 the noises from the background are cleaned, and the points related to the object are connected successfully. Our next is to test our implementation on real records from LIDAR with our mean-shift segmentation.

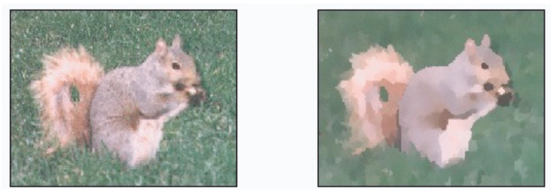


Fig. 11 First Test Case Mean Shift Algorithm. The original picture can be seen at the left, and the one with mean shift algorithm on the right.

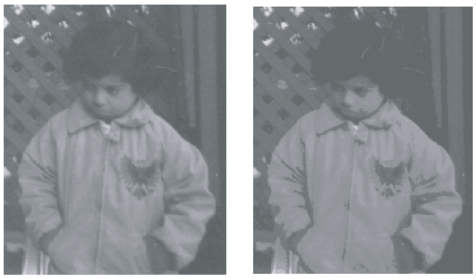


Fig. 12 Second Test Case Mean Shift Algorithm. The original picture can be seen at the left, and the one with mean shift algorithm on the right.

4 Conclusion

This paper presents an initial research direction for tracking multi-targets in marine environments for USV autonomous capabilities. LIDAR sensors are very precise, however, they are challenged by clutter and moving platforms for target detection and tracking in real time. We present several models for clutter formulation as an options for decluttering LIDAR measurements. Additionally, we propose the PHD filter, which is an advanced RFS-based filter. We describe the main stages for multi-object detection and tracking using this filter. We demonstrated our segmentation and mean-shift implementation on two test cases. Future work will focus on testing our implementation on sea records and sea experiments with the LIDAR sensor and PHD filter for testing and validation of our concept testing algorithm efficiency and power consumption integrated into small USV.

References

1. http://www.velodynelidar.com/lidar/hdlproducts/hdl64e.aspx
2. Jazwinski, A.H.: Stochastic Processes and Filtering Theory. Academic Press, New York (1970)
3. Anderson, B.D., Moore, J.B.: Optimal Filtering. Prentice-Hall, New Jersey (1979)

4. Askne, J.: Remote Sensing Using Microwaves, Department of Radio and Space Science. Chalmers University of Technology. Göthenburg, Sweden (2002)
5. Bar-Shalom, Y., Fortmann, T.E.: Tracking and Data Association. Academic Press, San Diego (1988)
6. Blackman, S.: Multiple Target Tracking with Radar Applications. Artech House, Norwood (1986)
7. Hue, C., Lecadre, J.P., Perez, P.: Sequential Monte Carlo methods for multiple target tracking and data fusion. IEEE Trans. Trans. SP 50, 309–325 (2002)
8. Hue, C., Lecadre, J.P., Perez, P.: Tracking multiple objects with particle filtering. IEEE Trans. AES 38, 791–812 (2002)
9. Mahler, R.: Multi-target Bayes filtering via first-order multi-target moments. IEEE Trans. AES 39, 1152–1178 (2003)
10. Goodman, I., Mahler, R., Nguyen, H.: Mathematics of Data Fusion. Kluwer Academic Publishers (1997)
11. Mahler, R.: An introduction to multisource-multitarget statistics and applications. Lockheed Martin Technical Monograph (2000)
12. Mahler, R.: A theoretical foundation for the Stein-Winter Probability Hypothesis Density (PHD) multi-target tracking approach. In: Proc. 2002 MSS Nat'l Symp. on Sensor and Data Fusion (Unclassified) Sandia National Laboratories. San Antonio, TX, vol. 1 (2000)
13. Ma, W.-K., Vo, B., Singh, S., Baddeley, A.: Tracking an unknown time-varying number of speakers using TDOA measurements, A random finite set approach. IEEE Trans. Signal Processing (54), 3291–3304 (2006)
14. Totit, F., Emanuel, R., Lucian, A., Cornel, I.A.: Advanced Sea Clutter Models and Their Usefuleness for Target Detection. MTA Review (2008)
15. Haykin, S., Puthusserypady, S.: Chaotic Dynamics of Sea Clutter. Chaos 7, 777–802 (1997)

Feasibility of Basic Visual Navigation for Small Robotic Sailboats

Tobias Neumann and Alexander Schlaefer

Abstract. Image based navigation is a key research focus for many robotic applications. One complication for small sailing robots is their limited buoyancy and rather rapid motion. We studied whether it would still be feasible to use video data for basic navigation in an inshore race course scenario. Particularly, we considered methods for detecting the horizon and buoys, as well as estimating rotations via optical flow. All methods have been tested on a set of manually annotated scenes representing different sailing and lighting conditions. The results show that detection rates of more than 80% for the horizon and more than 94% for buoys can be achieved. Moreover, a comparison of the average optical flow with compass data indicates that rotations of the boat can be estimated. Hence, the methods should be considered in addition to other sensors.

1 Introduction

Keeping a constant lookout at all times is part of the traditional navigation approach, e.g., to avoid collisions. While communication based systems like the Automatic Identification System (AIS) are potentially simpler and more robust, vision based methods may be more versatile and a number of approaches have been proposed [3, 7, 4]. Building a general purpose system to understand images in maritime environments is a challenging task, and we consider a more limited scenario with small sailing robots [8]. Our boats have limited buoyancy and allow for only very small and lightweight cameras to be installed. They are also moving substantially, even in very small waves. While we typically use an onshore server to communicate

Alexander Schlaefer · Tobias Neumann
Institute for Robotics and Cognitive Systems,
University of Luebeck,
Ratzeburger Allee 160, D-23562 Luebeck
e-mail: schlaefer@rob.uni-luebeck.de,
neumann@informatik.uni-luebeck.de

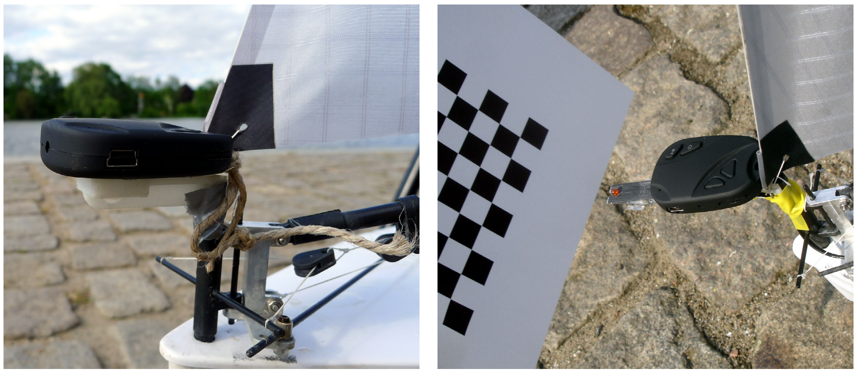


Fig. 1 A small 15 g GUNCAM mounted to the bow of an rrMM boat (a). Before sailing, the camera is calibrated (b).

position data [1], one advantage of a vision based approach could be a better ability to detect near field collisions and to augment sensor data with information obtained from images. We present methods for detection of horizon and buoys and we propose to augment sensor data, e.g., from compass, accelerometer, or gyroscope, with motion information obtained from the images.

2 Material and Methods

We obtained test data from a small camera (GUNCAM, www.guncam.de) with a weight of only 15 g mounted to the bow of a robotic racing Micro Magic (rrMM) sailing robot (Fig. 1). The camera has an image resolution of 720 x 480 pixels and a frame rate of 30 fps. All video data is directly recorded to a microSD card, which in our setup can store approximately 1 h of data.

So far, three different objectives have been considered: the detection of the horizon, the detection of buoys, and the estimation of the optical flow.

2.1 Horizon Detection

Finding the horizon is a typical task in maritime scene analysis. We consider an inshore setting where the horizon is typically defined by the shoreline. While this avoids sea and sky being indistinguishable, new issues arise, e.g., due to reflections. Initially, we used the following steps to find candidate lines in the images. First, the images are filtered with a 9x9 Gaussian kernel and a Canny edge detector [2] was applied. Second, a Hough transform was used to detect lines in the preprocessed images. The resulting set of lines C is ordered by descending length, i.e., the first line returned by the algorithm contains the largest number of edge pixels. We call C the candidate set. Obviously, if the horizon goes through the whole images, it is likely to be among the long lines. However, given possible reflections and small waves, other lines can actually be longer. To reduce the likelihood of misclassification, we considered the following additional assumptions:

1. The horizon remains relatively stable over time
2. The line representing the horizon is similar in two consecutive frames
3. The horizon runs typically close to the image center for our camera setup

Each line in C is represented by a point P and an angle m between the line and the x-axes. Consider two lines l_1 and l_2 represented by P_1, m_1 and P_2, m_2. Then we define the dissimilarity h between the two lines as

$$h(l_1, l_2) = \alpha \, \|P_1 - P_2\| + |m_1 - m_2| \tag{1}$$

where α is typically 0.5 and denotes the relative importance of a difference in the points and in the angle. Note that the points returned by the method are unique for the same line.

We now consider k consecutive video frames such that C_k is the set of candidate lines for the current frame. Moreover, we restrict $|C_i|$ to be at most n, i.e., only the n longest candidates in C_i are retained. For $c \in C_k$ we sort $\bigcup C_i | i = 1, 2, \ldots k-1$ according to h, i.e., such that the most similar lines come first. We then compute the mean dissimilarity for c and the first m items of the resulting set. The candidate c from the current frame with minimal mean dissimilarity is considered the horizon, as similar long lines have frequently occurred in the previous frames. To account for the observation that the horizon is typically a line close to the image center, we additionally consider the distance to center when evaluating the candidates.

In case no line is detected in the current video frame or if the dissimilarity of all candidates exceeds a threshold, the horizon line from the last frame is added to the candidate set and returned. Figure 2 illustrates the assumptions underlying the approach, i.e., that the horizon is typically close to the image center and more stable than, e.g., lines induced by waves.

2.2 Buoy Detection

Typically, buoys have a distinctive shape and color to be easily detectable. Our main purpose is the detection of marks when following a race course. Hence, we can assume that the colors are exceptionally bright, e.g., yellow, orange, or red. To use color for object detection, we first convert the images to the hue-saturation-value (HSV) color space. We then applied a threshold and a morphological closing operation for cleaning contours and used OpenCV to label each object in the resulting image. Subsequently a minimally enclosing ellipse was determined for each potential object, which in case of a buoy allows to estimate the center point and the radius.

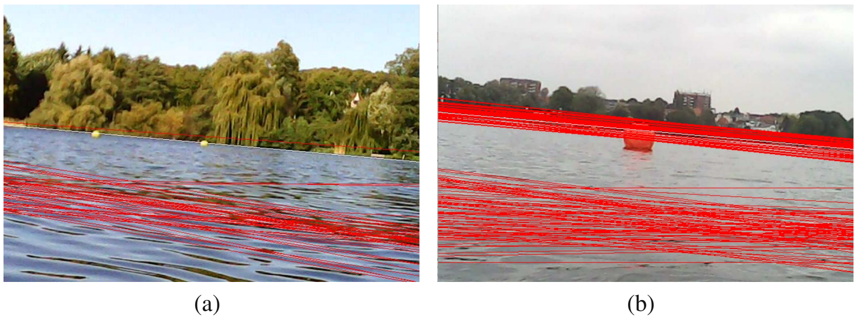


Fig. 2 Typically the horizon is close to the image center (a). The image in (b) shows a set of candidates from 30 consecutive frames, illustrating that the actual horizon is fairly stable.

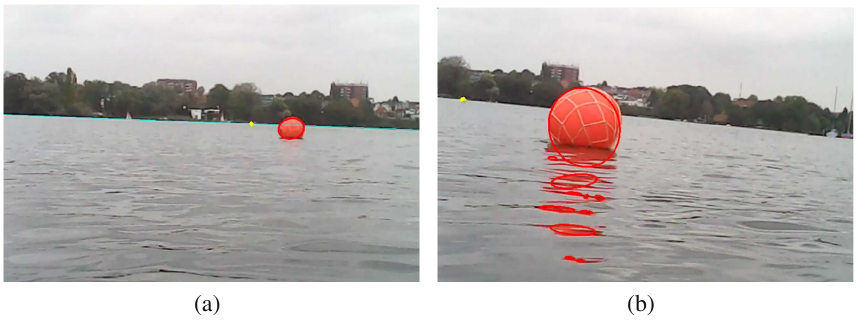


Fig. 3 An example for detecting a red and yellow buoy (a). Reflections can substantially complicate the detection (b). Note that bright sunshine and calm would lead to even more reflections.

Figure 3 illustrates the method and also highlights that reflections can cause substantial problems. Again, we can argue that reflections due to waves change over time. Using an approach similar to that used for the horizon, we consider a number of consecutive frames to identify stable objects. All objects are represented by the center point and radius of the respective ellipse. Clearly, as the boat can be very close to the buoy, the change in its position can be rapid. Hence we use a dissimilarity function that weights the position with 0.1 and the radius with 0.9.

2.3 Rotation Detection

To detect rotations of the boat we computed the optical flow using the Lucas-Kanade method [6]. First, we used the horizon to partition the image. We only considered features above the horizon in order to avoid artifacts from waves. For every two consecutive frames the mean flow over all features detected in both images was computed and stored.

2.4 Incorporating Sensor Data

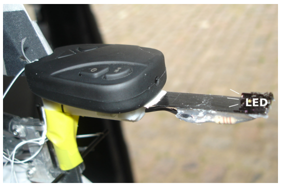
Obviously, it is interesting to study how information from image data and the sensor data available onboard can be considered jointly, e.g., to compute the expected position of horizon and buoys in subsequent image frames or to improve navigation by estimating the boat's motion from the image data. Our current design is limited in that the onboard computing power is not sufficient for video processing or storage. While we are working to integrate more powerful hardware, our current experimental setup was based on independent recordings with the GUNCAM. To synchronize camera and sensor data, a small LED was mounted in front of the camera. It was switched on via the onboard control unit following a Fibonacci sequence and its status was recorded with the log-data. During offline processing, the state of the LED was automatically detected in the video data, and the frames were labeled accordingly. Video and sensor data were then combined by correlating the state of the LED in the two data sets. Figure 4 illustrates the setup, how a certain subset of pixels was used to detect the state of the LED, and how sensor and image data were correlated.

2.5 Data

We collected 6 h of video data on 11 different days and in different weather and light conditions. From this data set we extracted five scenarios of 11 s to 16 s length representing different conditions (Fig. 5). The actual weather conditions and the lengths of the video sequences are summarized in Table 1. Each of the video frames was manually annotated, i.e., from the horizon or buoy candidates the correct one was identified and stored. This data was then used to run tests quantifying the detection rate for horizon, buoys and rotation.

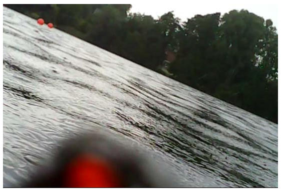
Table 1 Summary of length and wind and weather conditions for the five test scenes.

scene	weather	wind (kn)	length (s)	total number of frames
1	sunny	2-3	16	480
2	cloudy	6-12	14	440
3	cloudy	4-8	13	394
4	rainy	8-12	11	341
5	cloudy	6-10	13	409

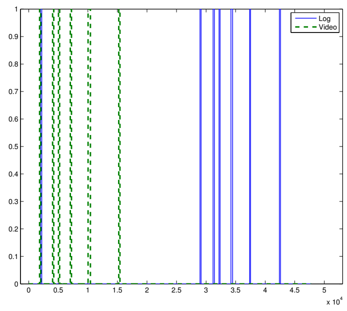


(a)

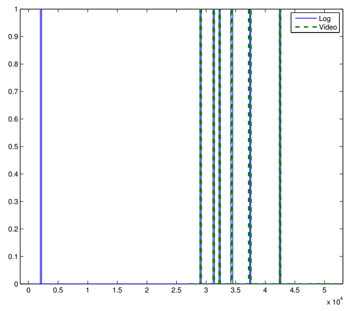
(b)



(c)



(d)

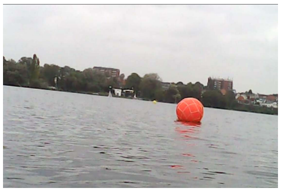


(e)

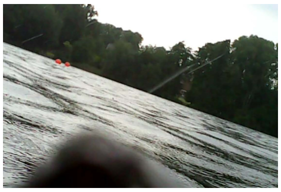
Fig. 4 The camera with LED (a), examples for images with LED switched off (b) and on (c), and the log and video data before and after correlation, respectively (d,e). For the latter two, blue indicates log data while the dotted green lines indicate video data.

(a) Scene 1

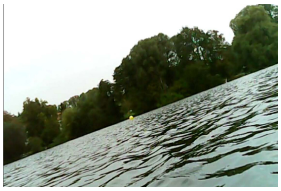
(b) Scene 2



(c) Scene 3



(d) Scene 4



(e) Scene 5

Fig. 5 Randomly chosen images from the five scenes we analyzed. Note another rrMM boat in scene 2. More details on the weather conditions are summarized in Table 1.

3 Results

The actual images were processed offline using OpenCV on a computer with an Intel Q6600 2.40 GHz processor and 4 GB memory running a 64 bit Fedora-Linux.

3.1 *Horizon Detection*

Table 2 summarizes the performance of horizon detection using $k = 16$, $n = 3$, and $m = 5$. All scenes resulted in candidate lines for each frame, except for scene 1, where only 61 % of all frames yielded at least one candidate line. Taking the frames with at least one candidate line as the basis, the detection rate for the horizon ranged from 81 % for scene 4 to 90 % for scene 1. The mean runtime ranged from 24 ms to 46 ms.

Table 2 Results of the horizon detection tests including the total number of frames, the number of frames for which at least one line was detected, the number of correctly identified horizon lines, and the runtime.

scene	total frames	with lines	with correct horizon	runtime (ms)
1	480	293	265	23.95
2	440	440	376	43.75
3	394	394	337	20.60
4	341	341	277	33.05
5	409	409	343	39.49

3.2 *Buoy Detection*

Only scenes 3, 4, and 5 contained buoys, with scenes 3 and 4 each containing two. Hence, Table 3 summarizes results for five tests with red and yellow buoys in distances ranging from approximately 5 m to more than 100 m. Candidate objects were visible in slightly more than 50 % for scene 4, and in more than 80 % of the frames for scenes 3 and 5. The yellow buoy for scene 3 was detected in 78 % of the frames actually containing it, while all the other buoys were detected in more than 94 % of the frames. Between 36 % and 68 % of the frames resulted in a stable detection of the correct buoy, i.e., the buoy was correctly labeled in subsequent frames. However, for the red buoy in scene 3, an irrelevant object was incorrectly identified as a stable buoy in 6 % of the frames.

Table 3 Results of the buoy detection tests including the scene, the object to detect, the approximate distance, the total number of frames, the number of frames with the object visible, the number of frames where the object was detected as a candidate, the number of frames where the object was correctly / wrongly classified as stable, and the runtime.

scene	buoy	distance (m)	total frames	visible	candidate	stable	wrong	runtime (ms)
3	red	5-10	394	326	326	212	20	45.69
3	yellow	100	394	333	259	126	0	43.25
4	red (1)	50	341	178	175	64	0	44.87
4	red (2)	50	341	199	195	82	0	43.46
5	yellow	100	409	355	333	242	0	43.96

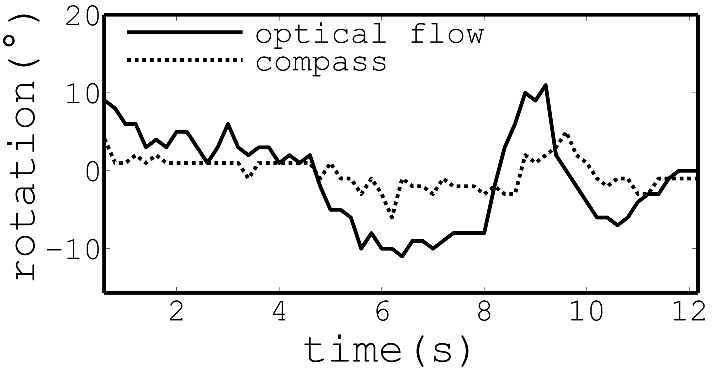


Fig. 6 The plots show the mean optical flow vs. the logged compass data for scene 1.

3.3 Rotation Detection

Figure 6 illustrates the results of the optical flow experiments for scene 1. Clearly, flow and compass values are not the same. However, the trend, i.e., turning port or starboard, is similar and a further calibration may lead to better agreement of the magnitude of change.

4 Discussion

We considered basic visual information for navigating small sailing robots in in-shore race scenarios. More than 80 % accuracy in detecting the correct horizon is far from perfect but still illustrates that a fairly simple approach yields good results. Moreover, adapting the preprocessing of the images – particularly the Gaussian filter – to the sailing conditions could further improve the performance. While one could argue that technically we identify the shoreline, we would hold that for a small boat with the camera mounted a few centimeters above the water, this close to the actual horizon. A more obvious limitation is the dependency on the illumination, which will require more tests in sunny conditions. However, at least in our home area in northern Germany good sailing conditions often coincide with cloudy weather.

Clearly, illumination also effects the buoy detection. Another problem is the correct labeling of the detected objects over multiple frames, which only worked for two out of five scenes. Note that part of this issue is related to the buoy leaving and entering the image, and a better integration with the boats' sensors may be used to compute the expected position of the buoy. For example, in scene 4 the buoys were frequently out of sight, which relates to the poor performance in stable detection. However, an important goal of buoy detection in our scenario is the rounding of marks, and it is promising that the test scene with a close proximity to the mark resulted in a larger number of consecutive frames where the buoy was correctly detected. Further work will consider the actual size of the buoy for this particular scenario.

The proposed methods allow estimating basic information about changes in the robot's pose. It is straightforward to get the approximate heeling angle from the detected horizon. Likewise, the detected rotation can indicate course changes. While the current results are preliminary and require further calibration, a visual compass has been considered before and presents a promising approach [5]. Particularly as the motions of a small sailing robot are characterized by substantial pitching and rolling even in small waves, which can compromise the compass readings. A further possible use of visual information is estimating the distance to buoy, if their size is known. In fact, we consider the methods most useful in the proximity of buoys, where their estimates may be more accurate than, e.g., GPS data. For example, a close rounding of a mark could be based on image data rather than GPS.

Considering the runtime, horizon and buoy detection could be performed at a rate of 20 Hz to 45 Hz. Although the computations were done offline and separately on a rather powerful computer, we believe that further optimization of the code and more powerful microprocessors will allow for sufficiently fast onboard online processing.

5 Conclusion

Rapid motion and limited buoyancy complicate image guided navigation for small robotic boats. However, our results indicate that basic information can be derived, which is sufficiently robust to augment other sensors. The heeling of the boat and rotations can be estimated and the horizon and buoys can be detected. Further work should address the robustness and runtime of the proposed methods.

References

1. Ammann, N., Hartmann, F., Jauer, P., Krüger, J., Meyer, T., Bruder, R., Schlaefer, A.: Global Data Storage for Collision Avoidance in Robotic Sailboat Racing – the World Server Approach. In: Schlaefer, A., Blaurock, O. (eds.) Robotic Sailing, vol. 79, pp. 157–166. Springer, Heidelberg (2011)

2. Canny, J.: A computational approach to edge detection. IEEE Trans. Pattern Anal. Mach. Intell. 8(6), 679–698 (1986)
3. Gal, O.: Automatic Obstacle Detection for USV's Navigation Using Vision Sensors. In: Schlaefer, A., Blaurock, O. (eds.) Robotic Sailing, vol. 79, pp. 127–140. Springer, Heidelberg (2011)
4. Huntsberger, T., Aghazarian, H., Howard, A., Trotz, D.C.: Stereo vision based navigation for autonomous surface vessels. Journal of Field Robotics 28(1), 3–18 (2011)
5. Labrosse, F.: The visual compass: Performance and limitations of an appearance-based method. Journal of Field Robotics 23(10), 913–941 (2006)
6. Lucas, B.D., Kanade, T.: An iterative image registration technique with an application to stereo vision. In: Proceedings of the 7th International Joint Conference on Artificial Intelligence - Volume 2, IJCAI 1981, pp. 674–679. Morgan Kaufmann Publishers Inc., San Francisco (1981)
7. Ramirez, R., Guo, Y., Ieng, S.H., Plumet, F., Benosman, R., Gas, B.: Omni-directional camera and fuzzy logic path planner for autonomous sailboat navigation. In: 2011 Iberoamerican Conference on Electronics Engineering and Computer Science (CIIECC 2011), pp. 335–346 (2011)
8. Schlaefer, A., Beckmann, D., Heinig, M., Bruder, R.: A New Class for Robotic Sailing: The Robotic Racing Micro Magic. In: Schlaefer, A., Blaurock, O. (eds.) Robotic Sailing, vol. 79, pp. 71–84. Springer, Heidelberg (2011)

Optimization-Based Weather Routing for Sailboats

Jorge Cabrera-Gámez, José Isern-González, Daniel Hernández-Sosa, Antonio Carlos Domínguez-Brito, and Enrique Fernández-Perdomo

Abstract. In this paper we propose a deterministic route planner for a sailboat suitable for areas where high quality wind and currents forecasts are available. An optimization based approach is used with the objective of minimizing the time required to arrive at a destination. Several simulations have been performed using high resolution regional forecasts from HIRLAM and MyOcean models in order to test the validity of this method.

1 Introduction

Long term or weather routing deals with the problem of finding a sequence of waypoints connecting given pairs of starting and ending coordinates taking into account weather forecasts and other possible constraints. The solution to this problem has a number of interesting applications in marine navigation for ships, e.g. minimization of fuel consumption or improvement of passenger comfort; or, in sailboats, safe routing and the planning of long distance regattas.

Route planners can be classified into deterministic route planners [1], these are not suitable if the uncertainty of the weather forecast is very high, and non-deterministic route planners, where an ensemble of weather forecasts is used to perform the planning [5].

In this paper, we will focus on a deterministic route planner for areas where high quality wind and currents forecasts are available. We will discuss its application to the problem of optimizing the route of a sailboat with the objective of minimizing the time required to arrive to a destination.

Jorge Cabrera-Gámez · José. Isern-González · Daniel Hernández-Sosa ·
Antonio Carlos Domínguez-Brito · Enrique Fernández-Perdomo
Instituto Universitario de Sistemas Inteligentes
y Aplicaciones Numéricas en Ingeniería, Siani
Universidad de Las Palmas de Gran Canaria,
Canary Islands, Spain
e-mail: jcabrera@iusiani.ulpgc.es

The article is organized into four sections devoted, respectively, to describing the route planner and the wind and current forecast data sources, presenting some simulation results and finally summarizing the main conclusions and future work.

2 The Route Planner

The architecture of the route planner consists of two levels. At the inner level it uses the approach described by Stelzer and Pröll in [2] to select the best bearing, given the wind direction, sailboat position and bearing, and destination coordinates. At a second level, the router tries to find an optimal route that minimizes time to destination. At this level the influence of currents may also be taken into consideration.

The selection of the best route between two points, given winds and currents forecasts, is performed by an unconstrained nonlinear optimization of the time needed to reach the destination. The procedure is based on the Nelder-Mead simplex algorithm [6] and is known as the *fmisearch* function in Matlab. In this paper the constant goal has been to minimize the duration of the navigation, but it can be easily adapted, for example, to reach a predetermined rendezvous point with a minimal time delay.

Routes are defined by a set of a few intermediate waypoints. The optimization explores the possible routes found by displacing these waypoints from their initial positions. The number of waypoints used is a parameter of the algorithm and it depends basically on the duration of the regatta. In this paper we have used two basic approaches for defining the initial localization of waypoints. If the algorithm is capable of finding a route without introducing any intermediate waypoint, then the set *n* of waypoints is obtained from this initial route by taking *n* points along the trajectory separated by a regular time lapse. Note that this is equivalent to using the original algorithm of Stelzer and Pröll [2] to define the full route. This approach is not viable in cases where the algorithm fails due to the presence of obstacles. In those cases the initial set of intermediate waypoints is spread uniformly over the line defined by the departure and destination points.

The route is only approximately defined by the final position of the waypoints. There is a second parameter that controls how far from a given waypoint the route is allowed to vary. As the waypoints are only used to explore the space of possible routes, this parameter, called radius of precision, is normally set to a large distance, typically several kilometers.

Algorithm

1. Define how many intermediate waypoints are to be used and the precision radius (maximum allowed distance for passing a waypoint).
2. Set up the size of the searching area around the initial route. This parameter defines how far the route waypoints can be displaced from their initial localization.

3. Choose the time step for simulating the sailboat motion. In this paper we have chosen 60 seconds.
4. Obtain an initial approximation for the route and set up the initial set of waypoints. An initial approximation for the route can be obtained in different ways: maximum circle, direct rhumb, etc.
5. Run an unconstrained nonlinear search over possible localizations for intermediate waypoints with the objective of minimizing the route's time.

The planner may or may not include the effect of currents in the definition of the route, as will be shown later. Surface currents are obtained from high resolution ROMs models and the values corresponding to the instantaneous position and time of the sailboat are defined by linear interpolation of the grid values.

3 Winds and Currents

The route planner uses numerical, high resolution (0.05° resolution in latitude/longitude) weather forecasts produced by the Spanish Meteorological Agency (AEMET). These forecasts [4] are produced using a HIRLAM model that provides one analysis and 12 forecasts (+3h) in GRIB1 format files, covering a period of 36 hours. A new update is produced every 6 hours. The planner uses the wind field computed at a 10 meters interval over ground.

The planner can also be fed with ocean currents provided in NetCDF format. In this paper, current maps are obtained from MyOcean (IBI domain) or ESEOCAN ROMs provided by Puertos del Estado (Spanish Harbor Authority).

The planner uses a simple kinematic model of a sailboat where the physical modeling of the vessel is summarized in its polar diagram. As a byproduct of the ROM model, the planner is capable of dealing with routes along coastlines.

4 Results and Discussion

Some of the results obtained in simulation are summarized in the following figures. To avoid repetition in each simulation, we first summarize the elements of the presentation that are common to all figures.

Simulations are sketched as a series of snapshots running left to right, and then downwards. Note that every snapshot is time stamped (hours and minutes of simulated time) along the top edge. To make reading the figures easier, normally only the wind field is displayed. When the wind speed is over 6 m/s, wind arrows are colored in green. The same style of presentation is used with current fields. In this case, current arrows are colored deep blue when currents are equal or over 0.3 m/s. When displaying both current and wind fields, the wind field is shown at a lower resolution to aid visualization. All simulations have been run using the highest resolution data available.

Two routes are depicted in most figures. Routes in magenta are the routes obtained by the application of the algorithm presented in [2]. In this case, the algorithm tries to set bearings that maximize the Velocity Made Good (VMG) to destination at each simulation step. We term this approach as "direct to goal" or DtG. Green curves denote trajectories obtained by the route planner through optimization. In all cases, trajectories are simulated with a temporal resolution of 60 seconds and the same polar diagram used in [2]. The precision radius used at intermediate waypoints was 2 km and 300 m for destination.

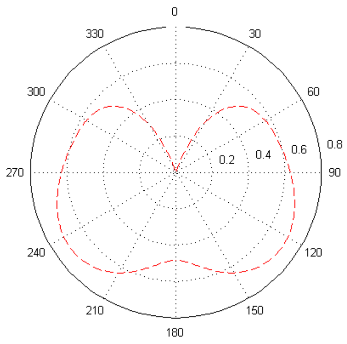


Fig. 1 Polar diagram used in the simulations at wind speed of 1 m/s [2].

The route planner is programmed in Matlab. Even though it is unoptimized Matlab code (using only one processor core), it takes 348 seconds on an Intel Core i7 -2630QM 2GHz/ 8GB of RAM laptop to solve the route depicted in Figure 3.

We have not considered leeway effect nor do we model any velocity decrement when tacking. These two aspects are seen as future enhancements to the simulation.

Figures 2 and 3 show simulation results in a scenario located on the western part of the Canary Islands archipelago. Both series of figures depict the routes obtained, respectively, with a DtG approach or with the route planner. Figure 2 does not include the effect of the currents, which, as Figure 3 demonstrates, are an important factor in this specific case. Wind field is depicted at a lower resolution in Figure 3, but it is the same shown in Figure 2. In these simulations, the number of intermediate waypoints was set to four.

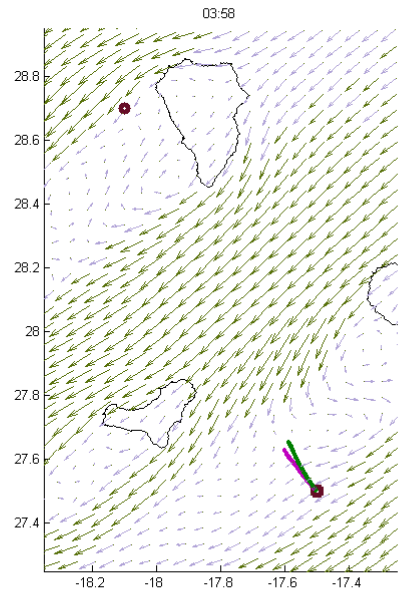


Fig. 2-1

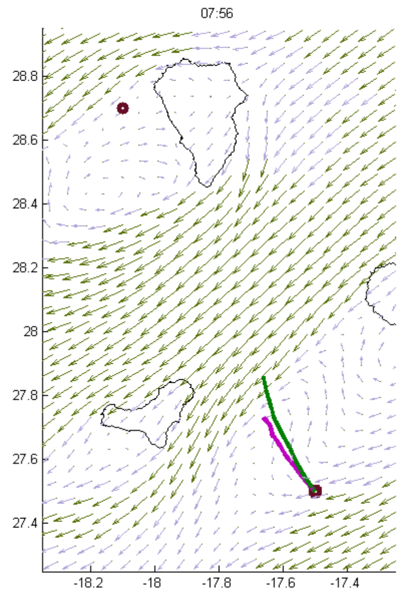


Fig. 2-2

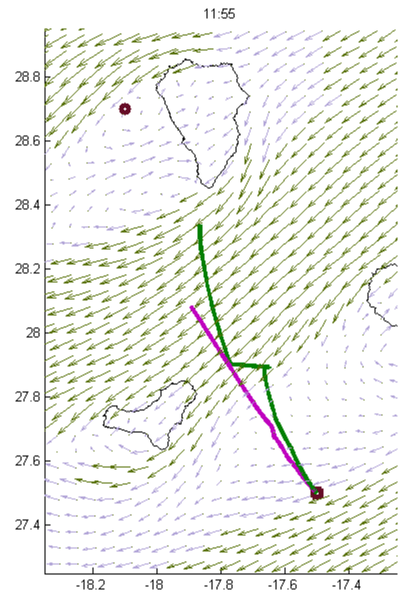


Fig. 2-3

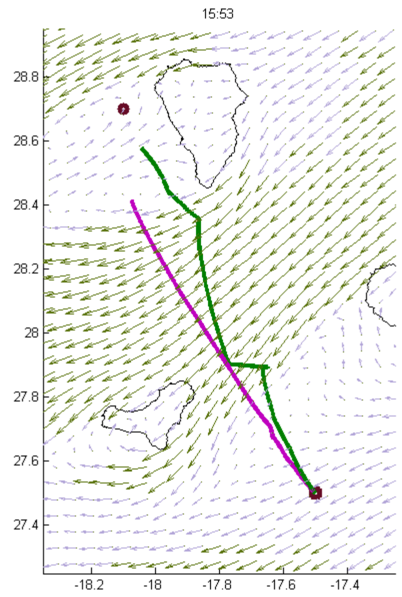


Fig. 2-4

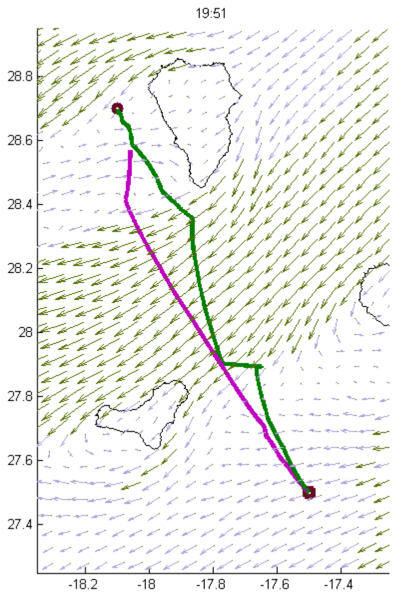


Fig. 2-5

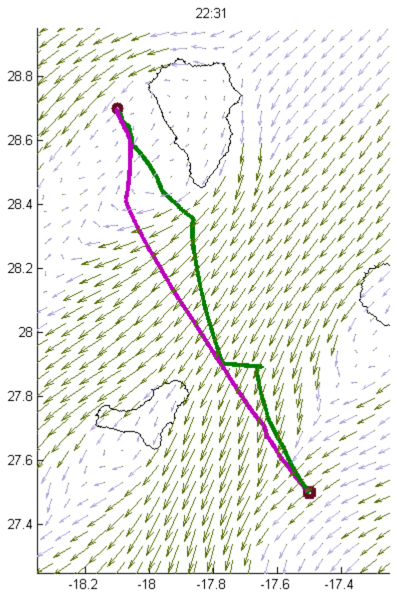


Fig. 2-6

Fig. 2 Simulation snapshots including only the effect of winds in a scenario located on the western islands of the Canary archipelago. In these figures only the wind field is shown. In this case, the optimized route arrives to destination nearly 4 hours before than the DtG route.

Trade winds, blowing in the Canaries from northeast in summer and autumn, are clearly appreciable in these figures. This wind regime, combined with the islands' high relief (La Palma, max. height is 2426 m), is responsible for the appearance of strong eddies at the southwest of the islands that alter the current patterns in those zones. Wind vortices are also clearly appreciable leeward of the islands, especially to the west of La Palma (the most northern island).

The results arising from the simulation included in Figure 2 show that the optimized route takes advantage of knowledge of the spatial distribution of the wind in advance. The resulting route is about 3 hours and 30 minutes shorter than the DtG route.

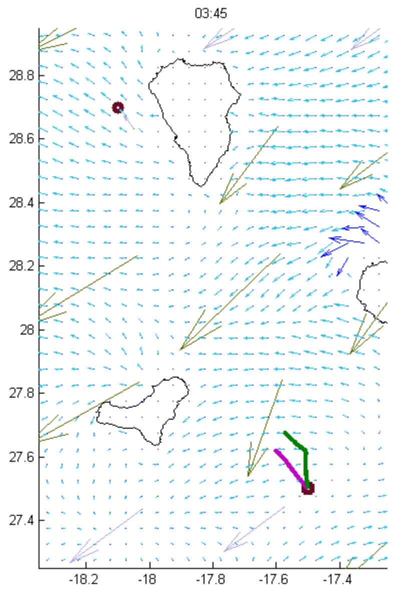


Fig. 3-1

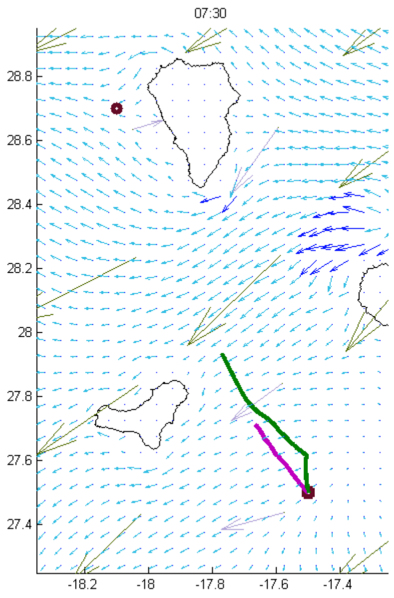


Fig. 3-2

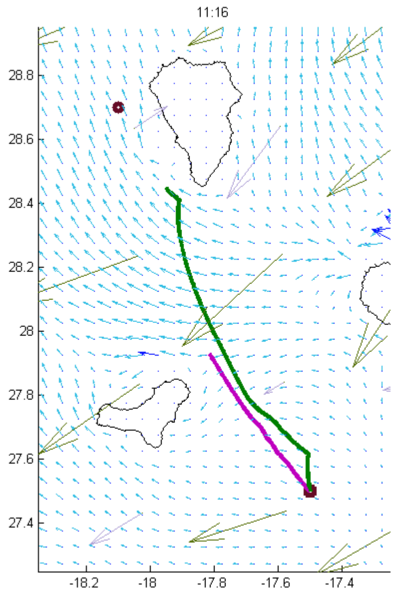


Fig. 3-3

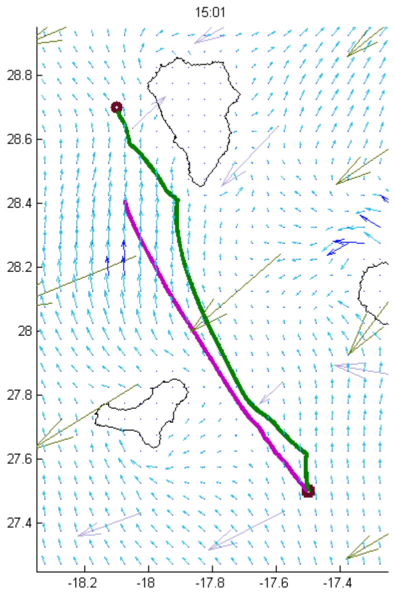


Fig. 3-4

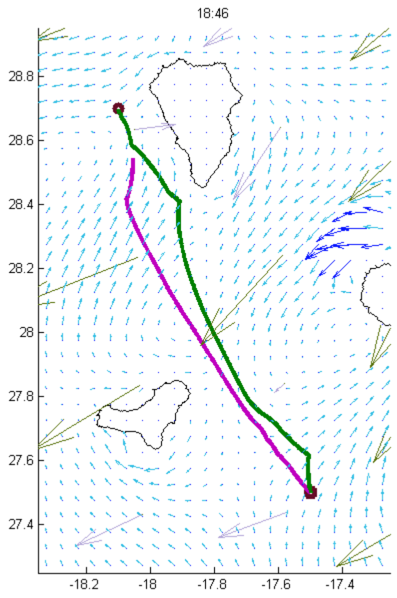


Fig. 3-5

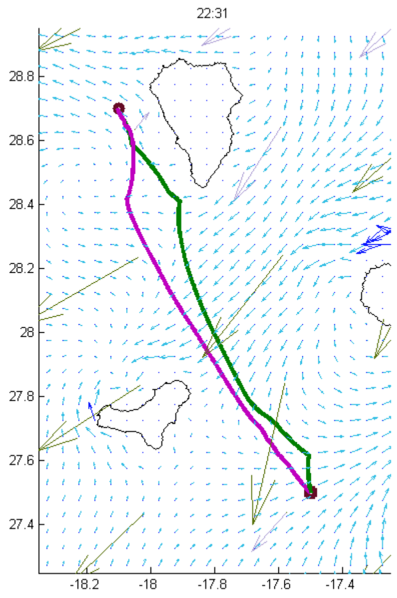


Fig. 3-6

Fig. 3 Simulation snapshots including only the effect of winds and currents in the same scenario (and data) used in Figure 2. These figures show the current and the wind (at lower resolution) fields.

Figure 3 shows the routes resulting from taking into account the effect of currents over the same scenario. In this specific case the effect of the currents is very important and in fact the optimized trip time is reduced by 6h 30m. Note that the routes produced by the DtG in figures 2 and 3 are identical.

Finally, the series of snapshots included in Figure 4 depicts the planning of a route from Las Palmas de Gran Canaria (Gran Canaria) to Arrecife (Lanzarote), a classic regatta in the Canaries. The simulation shown takes into consideration the effect of adverse currents, which occur over the entire route. In fact, in this case if the effect of currents is ignored the simulation gives a total time that is shorter by 54 minutes.

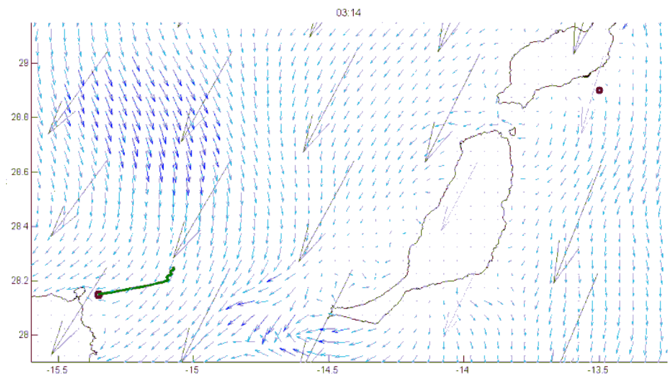


Fig. 4-1

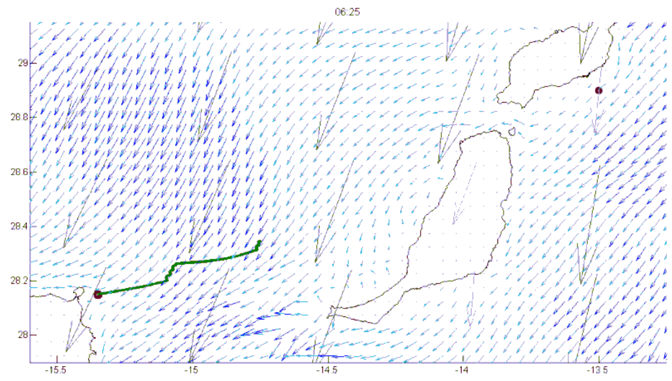


Fig. 4-2

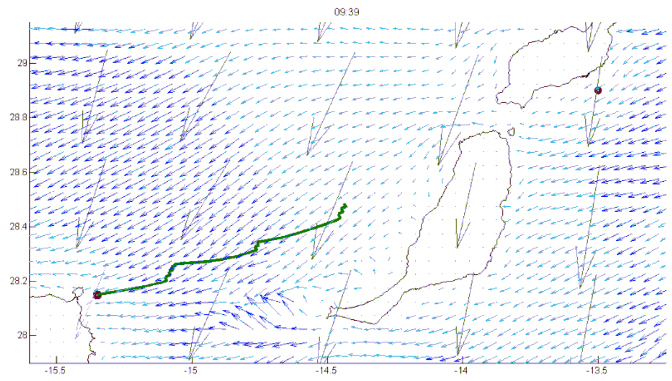


Fig. 4-3

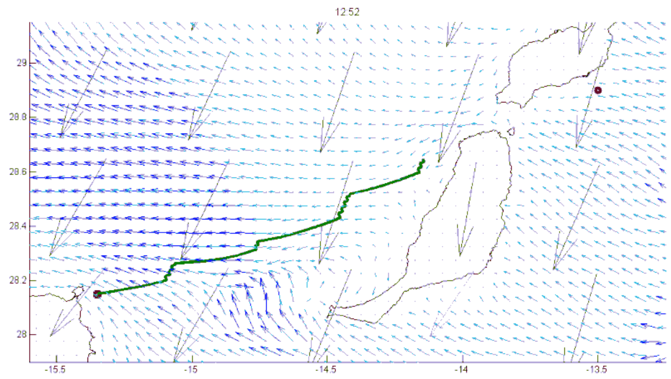


Fig. 4-4

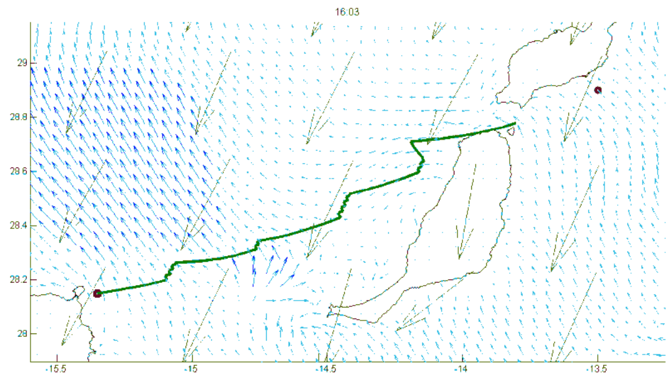


Fig. 4-5

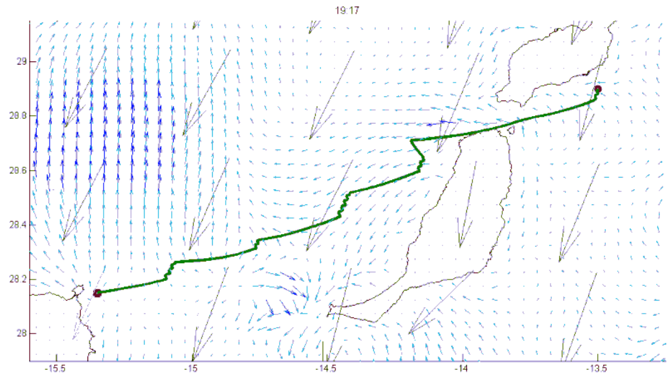


Fig. 4-6

Fig. 4 A route for a classical regatta in the Canary Islands.

The results achieved in this paper should be considered preliminary. In simulation and with a number of simplifications, it has been demonstrated that high resolution wind and current forecasts can play a significant role in optimizing sailboat routes. However, simulations are quite different from using an actual boat, as a number of factors ranging from unmodeled aspects (e.g. leeway or wave influences) to forecast credibility have not been studied in this preliminary work.

Future work will address the effect of leeway and waves over the planned route. The extension of the route planner to operate with forecast ensembles or - in general - take into consideration the credibility of wind and current forecasts is also foreseen.

Acknowledgments. This work has been partially supported by the project ProID20100062 funded by the Autonomous Government of Canary Islands (Agencia Canaria de Investigación, Innovación y Sociedad de la Información), FEDER funds and the Departamento de Informática y Sistemas at ULPGC. The authors thank Puertos del Estado and AEMET (Agencia Estatal de Meteorología) for granting access to their numerical products and data repositories.

References

1. Langbein, J., Stelzer, R., Frühwirth, T.: A Rule-Based Approach to Long-Term Routing for Autonomous Sailboats. In: Schlaefer, A., Blaurock, O. (eds.) Robotic Sailing, vol. 79, pp. 195–204. Springer, Heidelberg (2011)
2. Stelzer, R., Pröll, T.: Autonomous sailboat navigation for short course racing. Robotics and Autonomous Systems 56, 604–614 (2008)
3. AEMET, AEMET's High Resolution Limited Area Model (2012), `ftp://ftpdatos.aemet.es/modelos_numericos/` (accessed May 2012)
4. Puertos del Estado, Forecast & Measurement networks (2012), `http://www.puertos.es/en/oceanografia_y_meteorologia/redes_de_medida/index.html` (accessed May 2012)
5. Kristensen, N.M.: Sensitivity to ensemble wind and current input. Master thesis in Geosciences Meteorology and Oceanography, University of Oslo (2010)
6. Lagarias, J.C., Reeds, J.A., Wright, M.H., Wright, P.E.: Convergence Properties of the Nelder-Mead Simplex Method in Low Dimensions. SIAM Journal of Optimization 9(1), 112–147 (1998)

Part II
Extended Field Tests

Data Mining for Optimal Sail and Rudder Control of Small Robotic Sailboats

Lars Hertel and Alexander Schlaefer

Abstract. Finding the optimal parameter settings to control a sailing robot is an intricate task, as sailing presents a fairly complex problem with a highly non-linear interaction of boat, wind, and water. As no complete mathematical model for sailing is available, we studied how a large set of sensor data gathered in different conditions can be used to obtain parameters. In total, we analyzed approximately 2 million records collected during more than 110 hours of autonomous sailing on 55 different days. The data was preprocessed and episodes of stable sailing were extracted before studying boat, sail and rudder trim with respect to speed, course stability, and energy consumption. Our results highlight the multi-criteria nature of optimizing robotic sailboat control and indicate that a reduced set of preferable parameter settings may be used for effective control.

1 Introduction

Trimming a sailboat for optimal performance is an intricate task. Typically, various lines allow adjusting the shape and position of the sails depending on the overall situation, e.g., wind force, sea state, available crew, etc. Conventionally, ambitious sailors have created sail trim charts or tables to summarize the optimal settings. For robotic sailing, the energy needed to maintain boat trim becomes another criterion to consider [3]. Moreover, forgiving and stable sailing characteristics can reduce the wear on the mechanical components. Hence, the overall boat performance is characterized by different, partially conflicting objectives and presents a multi-criteria optimization problem. Interestingly, for a robotic boat the data required to compute and control the position of sails and rudder is available and can be easily stored and

Alexander Schlaefer · Lars Hertel
Institute for Robotics and Cognitive Systems,
University of Luebeck, Ratzeburger Allee 160, D-23562 Luebeck
e-mail: schlaefer@rob.uni-luebeck.de,
hertel@informatik.uni-luebeck.de

analyzed. Yet, the means to control the boat trim are typically limited, e.g., most current robots cannot adjust the sail area. Often, weight, energy demand and potential lack of mechanical stability are reasons to keep the parameters that can be controlled at a minimum. We studied how the sailing performance of a small robotic sailboat is affected by boat trim and the use of sail and rudder actuators during sailing. To assess to what extend the parameters can be optimized we used up to four similar robotic racing Micro Magic (rrMM) boats [5] simultaneously. Employing data mining techniques we analyzed more than 110 hours of sailing spanning a total of 55 different days in fall 2011. Our results indicate that a careful analysis can be used to identify a good balance between performance, robustness, and energy usage.

2 Material and Methods

We used up to four similar rrMM boats to gather data for various conditions. The boats are rather small with a length over all of approximately 0.54 m [5]. Our design employs three servos to independently set jib, main sail and rudder positions (Figure 1a). Moreover, the weight and longitudinal position of the keel can be varied. The boats communicated basic sensor data at 5 Hz to an onshore computer, which controlled all boats simultaneously. Sensors included apparent wind direction, apparent wind speed, 3D compass, 3D accelerometer, 3D gyroscope, GPS data, and servo angles. The wind speed was precluded from the analysis, as calibration and comparability among the different boats were found to be unreliable. In total 2 million data records spanning approximately 110 hours of sailing on a small lake were collected. The data represents a variety of conditions, e.g., wind speeds ranging from 4 kn to 14 kn, different wind directions, and waves of up to approximately 0.2 m. Note that the latter seems small but is substantial given a hull length of 0.54 m.

Initially, we had to calibrate the boats to collect comparable data. Wind direction, position of sails and rudder and proper operation of the sensors were checked. Immediately before sailing, compass and accelerometer were calibrated again. Moreover, some tests followed a specific protocol to ensure comparable conditions. The boats were set up to simultaneously sail the same course, i.e., close hauled, beam reach, broad reach or downwind, and to maintain the same apparent wind angle using the rudder. Hence, wind and wave conditions were virtually identical and only one parameter was varied at a time. Typically, the tests were done in westerly winds of approximately 7 kn to 9 kn and with minor waves. Figure 1b illustrates this scenario showing three boats with different sail trim on a broad reach course.

In order to run the specific tests, the boats had to be coordinated. First, they performed a station holding maneuver at a point p. Second, the same start time and apparent wind angle were set for all boats. Third, when the time was reached, the boats started to sail the same course simultaneously while keeping a fixed sail position and maintaining the same apparent wind angle by controlling the rudder accordingly. Fourth, after leaving the test area, i.e., a circle with center p and radius r, the boats sailed back to p, and the procedure was repeated. Figure 2 illustrates how the test was performed. The left plot shows the test area with center p and

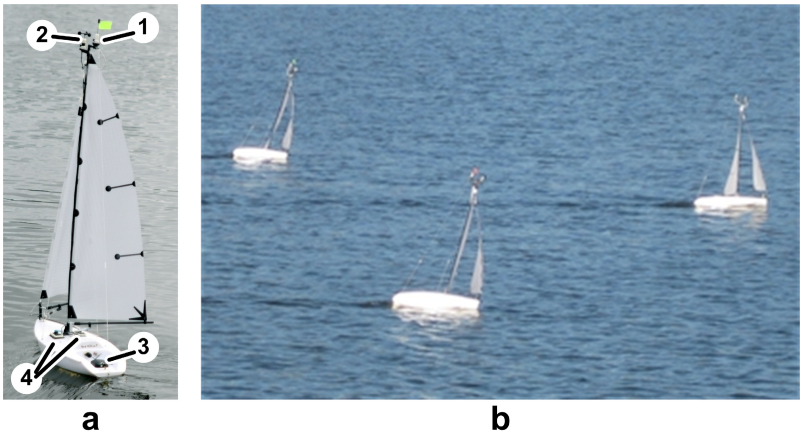


Fig. 1 (a) An illustration of an rrMM boat showing the wind sensors (1,2), the servos linked to the booms (3), and the GPS antenna (4). (b) The image shows three rrMM boats with different sail trim gathering data on a broad reach course.

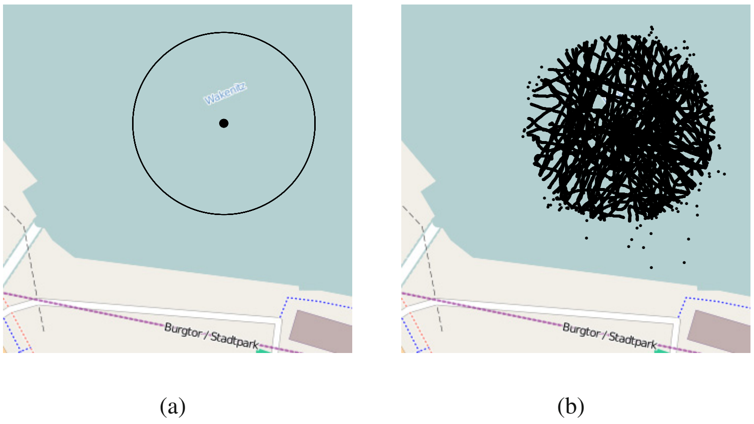


Fig. 2 To ensure comparable conditions, data was collected following a specific protocol. Subfigure (a) shows a schematic of the circle with center p and radius r in which the tests were performed. Subfigure (b) summarizes collected data during 6 h of testing.

radius r and the right plot summarizes data acquired during 6 h of testing. For our tests, the radius r was approximately 35 m.

Based on the collected data, we considered a number of questions with respect to optimal sail and rudder control. First, we analyzed the complete data set to get an understanding of the overall sailing performance of the boats. Second, we studied the trim of the sails, the keel position, and the frequency of sail and rudder position updates considering the specific test runs. Particularly, three boats were set up to keep the jib angle 7° smaller, similar, or 7° larger than the main sail angle; to have the keel mounted in a forward, central or aft position; and to have a keel bulb weighing 370 g, 470 g, and 580 g, respectively, mounted in the central position. Note that the jib offset of 7° was determined manually. Criteria to evaluate the sailing performance included boat speed, course stability, and energy efficiency.

Boat speed and course stability were based on GPS data and apparent wind, respectively, and evaluated in intervals, i.e., the mean speed and the standard deviation of the apparent wind were computed over a set of consecutive sensor readings. Clearly the GPS data is preprocessed by the sensor, but as we did not change the GPS settings during the test, the data still allows comparing the relative sailing performance. Moreover, the GPS latency was accounted for by shifting the data by approximately 2 s. In principle it would be possible to estimate the energy consumption from the recorded battery voltage. However, as the actual load on the servos substantially impacts the voltage readings, it is hard to get a reliable comparison during sailing. Instead, we considered the number of servo movements and analyzed the energy usage in a separate onshore test. In this test, three boats were set up identically in a laboratory environment and different patterns of sail and rudder motion were simulated. The scenarios included only rudder motion, only sail motion, and concurrent rudder and sail motion. To study the effect of limited sail motion, we performed two test runs. First, a boat was set up to sail according to the compass course and adjust the sails frequently. Second, the boat was set up to sail according to the apparent wind and correct the course less frequently. Hence, the test was done in similar conditions and the sails were set for the actual apparent wind, while the second test required less sail motion. In two similar experiments the frequency of rudder motion was restricted and the gain for the P-controller was varied to assess the resulting change in boat performance.

The actual data mining consisted of different stages. In a preprocessing step inconsistent data was excluded, e.g., the data had to be within the specifications of the respective sensor. Subsequently a median filter was applied. Furthermore, we required the boat to move with at least 0.5 kn and the position of the sails had to be consistent with the apparent wind reading. The latter is of interest as we considered fixed sail angles, and a fast wind shift can result in a rather fast and stable backward motion of the boat (note that the servos are linked to the booms, compare [5]). In a second step, we augmented the data by new attributes derived from existing ones. For example, the standard deviation in a sliding window was computed to study stability over time, e.g., of the apparent wind or the course.

In a third step, stable episodes were selected, i.e., sets of consecutive data records indicating stable sailing. Consider our data set S sorted in chronological order and let $t(s_i)$ denote the time stamp of record s_i. Then an episode E of length l is a chronologically ordered subset of S that contains all records $s_i | (t(s_i) \geq t(e_1)) \wedge (t(s_i) \leq t(e_l))$. A stable episode is an episode for which additional constraints hold. Typically, we required a GPS speed of at least 0.5 kn and a standard deviation of the apparent wind below 20°, as this indicates that no abrupt changes in wind direction or course occurred. The typical length was $l = 50$ records, which is equivalent to

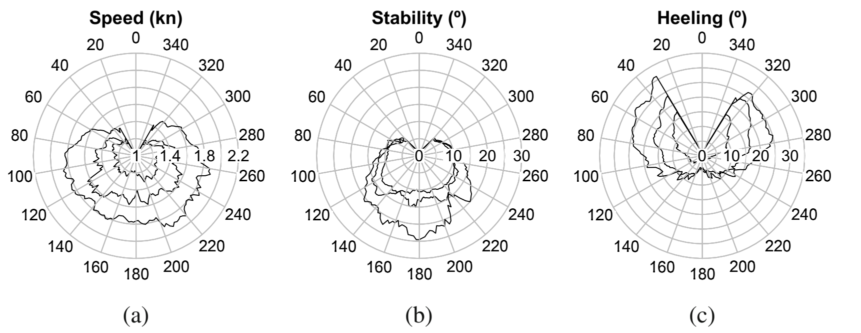


Fig. 3 The figure illustrates how the performance of the boat depends on wind speed (2 Bft to 4 Bft) and point of sail. Polar plot (a) shows that the boat speed increases with the wind speed. A beam or broad reach is typically fastest. Subfigure b) indicates that particularly the downwind course becomes less stable with increasing wind speed. c) Higher wind speed causes more heeling on close hauled or beam reach, but not on broad reach or downwind.

approximately 10 s of sailing. We used a greedy approach that successively removed all stable episodes from S, and for each stable episode basic statistics were computed and stored for further analysis.

3 Results

We divide the results into three main categories, namely general performance, boat trim, and energy consumption.

3.1 *General Boat Performance*

The typical boat performance is summarized in Figure 3, which is based on stable episodes. However, to include more data the parameters were relaxed to require a standard deviation of 30° and a an episode length of 25 records. The plots show the mean values for the respective conditions. Clearly, the boat speed depends on wind speed and point of sail, and Figure 3a illustrates that the boats reached up to 1.9 kn for 11 kn wind speed. A beam or broad reach is typically fastest, although the plot indicates that the difference for beam reach and downwind is not substantial. The stability of the sailing itself also depends on the point of sail as shown in Figure 3b. For a close hauled course the stability does not change with increasing wind speed while it goes from 10° to 25° for a downwind course and 5 kn to 11 kn, respectively. The heeling is summarized in Figure 3c, which shows an increase in heeling for increased wind speed and close hauled and beam reach courses, while there is very little heeling on the downwind course.

Table 1 Impact of the keel position on the boat performance. The table summarizes speed, stability, and heeling for different keel positions and different courses. The values represent mean and standard deviation for the respective combination.

Criteria	Keel Position	Close Hauled	Beam Reach	Broad Reach	Downwind
Speed (kn)	aft	1.05 ± 0.29	1.33 ± 0.28	1.43 ± 0.32	1.31 ± 0.41
	central	1.17 ± 0.29	1.44 ± 0.34	1.53 ± 0.42	1.40 ± 0.46
	forward	1.17 ± 0.28	1.35 ± 0.27	1.46 ± 0.42	1.40 ± 0.51
Stability (°)	aft	8.5 ± 10.7	13.2 ± 6.7	13.6 ± 13.4	15.2 ± 8.3
	central	14.0 ± 7.9	12.9 ± 7.4	12.6 ± 11.3	13.0 ± 14.7
	forward	18.1 ± 6.0	20.2 ± 12.0	13.9 ± 11.5	12.3 ± 7.1
Heeling (°)	aft	20.5 ± 10.7	17.6 ± 11.0	1.5 ± 9.6	1.6 ± 2.8
	central	18.0 ± 8.6	12.2 ± 10.1	2.1 ± 8.5	2.1 ± 5.0
	forward	24.9 ± 10.6	19.3 ± 13.9	4.8 ± 8.6	7.3 ± 2.5

Table 2 Impact of the keel weight on the boat performance. The table summarizes speed, stability, and heeling for different keel weights and different courses. The values represent mean and standard deviation for the respective combination.

Criteria	Keel Weight	Close Hauled	Beam Reach	Broad Reach	Downwind
Speed (kn)	light	1.21 ± 0.1	1.38 ± 0.25	1.48 ± 0.27	1.12 ± 0.32
	middle	1.33 ± 0.15	1.52 ± 0.17	1.53 ± 0.23	1.35 ± 0.29
	heavy	1.22 ± 0.18	1.38 ± 0.17	1.42 ± 0.27	1.47 ± 0.3
Stability (°)	light	8.5 ± 5.5	17.4 ± 8.1	21.7 ± 11.6	12.1 ± 4.0
	middle	9.8 ± 6.1	20.7 ± 5.9	22.3 ± 8.6	10.8 ± 3.9
	heavy	11.7 ± 5.6	19.1 ± 3.5	25.9 ± 14.5	16.4 ± 4.0
Heeling (°)	light	24.9 ± 6.6	17.2 ± 5.3	11.6 ± 7.4	5.9 ± 5.4
	middle	17.4 ± 7.3	15.1 ± 6.2	9.0 ± 6.5	6.2 ± 0.9
	heavy	17.4 ± 6.5	13.1 ± 3.7	7.3 ± 6.4	1.5 ± 1.5

3.2 Boat Trim

The Micro Magic allows changing the position and weight of the keel to adapt the boat trim to different wind and wave conditions. Table 1 summarizes the results for different positions of the 370 g standard keel in westerly winds of approximately 7 kn - 9 kn . A central keel position generally led to the highest speed and good stability, reaching a mean speed of 1.53 kn on the broad reach. Notable exceptions were improved stability when the keel was mounted aft on a close hauled course and forward on a downwind course.

To study the effect of different keel weights the light (370 g), middle (470 g), and heavy (580 g) keels were mounted to the central position and tested in westerly

Table 3 Impact of the sail positions on the boat performance. The table summarizes speed, stability, and heeling for a close, normal, and open jib angle. The values represent mean and standard deviation for the respective combination.

Criteria	Sail Position	Close Hauled	Beam Reach	Broad Reach
Speed (kn)	close	1.23 ± 0.22	1.48 ± 0.19	1.70 ± 0.18
	normal	1.23 ± 0.17	1.50 ± 0.15	1.67 ± 0.30
	open	1.22 ± 0.21	1.46 ± 0.25	1.65 ± 0.27
Stability (°)	close	10.4 ± 5.2	13.2 ± 3.1	20.0 ± 9.5
	normal	10.2 ± 5.5	18.2 ± 6.0	21.1 ± 9.8
	open	11.6 ± 5.6	27.4 ± 8.3	28.6 ± 13.7
Heeling (°)	close	20.7 ± 8.6	18.1 ± 6.5	10.7 ± 7.5
	normal	22.2 ± 7.0	19.6 ± 6.0	10.6 ± 6.8
	open	22.5 ± 8.0	20.6 ± 8.2	11.3 ± 8.6

winds of approximately 8 kn. The results shown in Table 2 indicate that the middle keel yields the best speed in these conditions. Stability is generally best with the light keel, which is also related to more heeling.

While the keel position affects the lateral area and cannot be modified during sailing, adjusting the sails to control the effective sail area is possible at any time. To address the trade-off between speed and stability, three different settings were defined: *close* (jib angle 7° smaller than main angle), *normal* (jib angle equal to main angle), and *open* (jib angle 7° larger than main angle). Note that a wing-on-wing configuration is typically much more stable and faster downwind, and hence this course was excluded from the test.

The results of our test in westerly winds of approximately 9 kn are shown in Table 3. There are no substantial differences with respect to the speed. However, the stability tends to be better for the *close* setting, which causes slightly less heeling, except for the broad reach.

Given that speed, stability and heeling are related, we analyzed all stable episodes from the complete data set with respect to speed and heeling. Figure 4a shows a typical plot indicating that an optimal heeling angle can be determined. To do so, we looked for the positive and negative, i.e., starboard and port, heeling and computed the angles that maximize the speed separately. The average of the two absolute values was considered the optimal heeling angle. This process was repeated for different data sets for wind speeds from 6 kn to 11 kn, leading to the plot shown in Figure 4b. Although we cannot readily infer what the corresponding parameters (keel position, keel weight, sail position) related to the values shown in the plot were, the optimal heeling for the different courses is similar in all wind conditions.

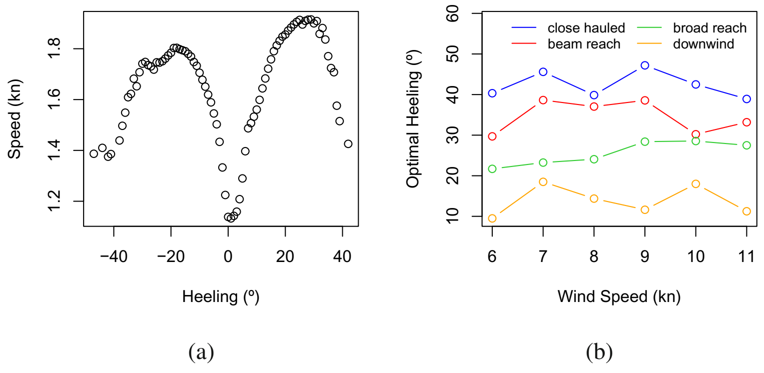


Fig. 4 The figure illustrates how heeling and speed are related and an optimal heeling angle regarding point of sail can be determined. Plot (a) shows the relation between heeling and speed for a broad reach course and an optimal heeling angle of approximately 20°. Subfigure (b) illustrates the optimal heeling angle depends on the point of sail, but not on the wind speed.

Table 4 The table corresponds to Figure 6 and summarizes boat speed, frequency of sail motion, and frequency of rudder motion for the two different control strategies.

Navigation	Median (kn)	Max (kn)	Sails (1/min)	Rudder (1/min)
Wind	1.497	1.886	8	211
Compass	1.536	1.886	376	198

3.3 Energy

In order to analyze the impact of sail and rudder activity on the energy consumption, we initially calibrated and monitored the battery voltage throughout systematic tests onshore. Figure 5a shows the course of the voltage readings for three boats without actuator motion, i.e., sail and rudder were kept in the same position and only mainboard, sensors, and communication consumed energy. The plots in Figure 5b show the change in voltage when only the rudder (blue), only the sails (green), and rudder plus sails (red) were moved in a regular pattern, in addition to the other onboard electronics. Note, that the values shown have been computed from the calibration data by considering a baseline of 8.4 V and subtracting only the difference due to sail and rudder motion, i.e., the plots represent the ideal course if only actuators were present.

Results for the experiment testing the performance with respect to a different frequency of sail changes are summarized in Figure 6 and Table 4. The figure shows that the boat effectively sailed close hauled, beam reach, and broad reach courses

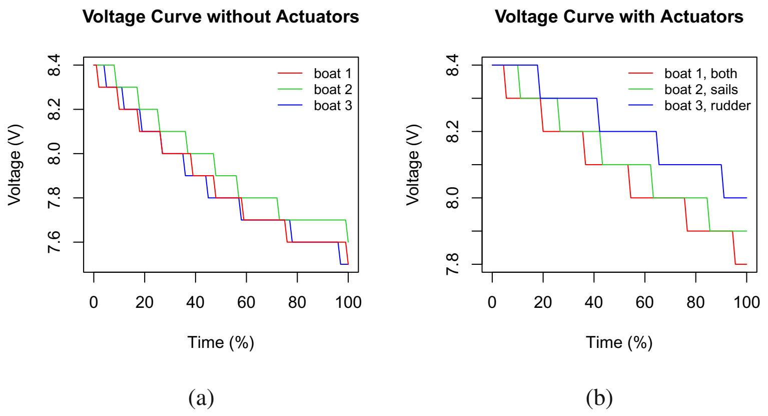


Fig. 5 Summary of onshore tests to analyze the impact of sail and rudder activity on energy consumption for three different boats. Subfigure (a) shows the voltage curve for calibration purposes without actuator motion. Subfigure (b) shows only the influence of the actuators on the battery voltage, i.e., the results already consider the calibration data. Note that the latter was done by subtracting the calibration data from the difference in consecutive voltage readings, and the actual readings would of course be lower than in subfigure (a).

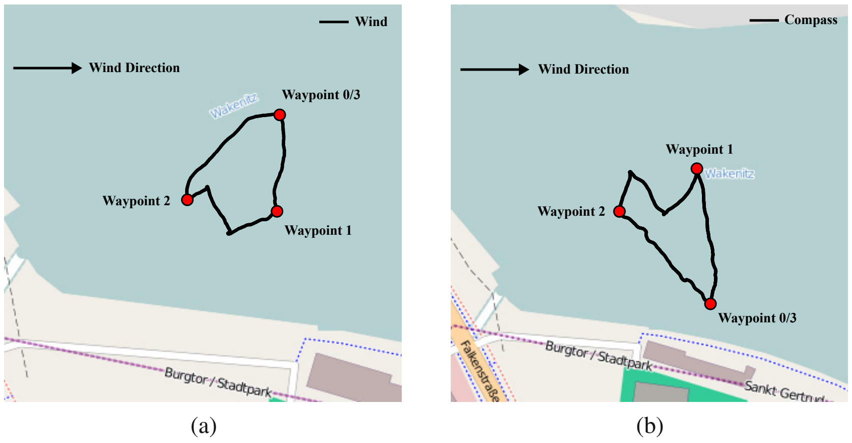


Fig. 6 Two different triangular courses set up for sailing tests in comparable conditions. (a) Control is based on the apparent wind, i.e., the rudder is moved to maintain the same apparent wind direction. (b) Control is based on the compass, i.e., a constant compass reading is maintained on the straight legs. Note that (a) allows fixed sail position for the different courses because of the constant apparent wind angle.

while completing a full triangle in westerly winds of approximately 11 kn. The table summarizes median and maximum boat speed after outlier removal, and the number of sail and rudder motion per minute required to complete the course.

Table 5 Impact of the frequency of rudder updates on boat speed, stability, and energy consumption for different courses. Note that the surrogate for energy consumption is the average change in the rudder position per control cycle. Generally, updating the rudder less often can reduce energy consumption.

Criteria	Frequency	Close Hauled	Beam Reach	Broad Reach	Downwind
Speed (kn)	rare	1.30 ± 0.17	1.27 ± 0.18	1.33 ± 0.45	1.54 ± 0.51
	often	1.22 ± 0.14	1.56 ± 0.10	1.59 ± 0.22	1.57 ± 0.19
	always	1.24 ± 0.18	1.47 ± 0.20	1.48 ± 0.29	1.33 ± 0.28
Stability (°)	rare	21.4 ± 8.1	25.7 ± 6.1	46.1 ± 21.8	22.8 ± 5.0
	often	6.2 ± 4.8	18.8 ± 6.2	23.2 ± 9.4	15.3 ± 3.8
	always	8.1 ± 6.6	20.8 ± 5.8	20.9 ± 12.8	12.7 ± 5.6
Energy ($^\circ/cycle$)	rare	0.07 ± 0.06	0.08 ± 0.07	0.12 ± 0.19	0.16 ± 0.04
	often	0.11 ± 0.06	0.19 ± 0.14	0.23 ± 0.09	0.23 ± 0.09
	always	0.58 ± 0.26	0.61 ± 0.34	0.98 ± 0.48	0.81 ± 0.41

Table 6 Impact of different gain for the P-controller for rudder on the boat performance for different courses. Note that the surrogate for energy consumption is the average change in the rudder position per control cycle.

Criteria	Gain (K_P)	Close Hauled	Beam Reach	Broad Reach	Downwind
Speed (kn)	low	1.10 ± 0.19	1.28 ± 0.21	1.30 ± 0.29	1.24 ± 0.26
	medium	1.18 ± 0.22	1.46 ± 0.21	1.56 ± 0.22	1.43 ± 0.16
	high	1.19 ± 0.11	1.35 ± 0.16	1.46 ± 0.18	1.52 ± 0.18
Stability (°)	low	8.4 ± 5.7	17.2 ± 10.8	22.4 ± 13.0	19.0 ± 9.6
	medium	5.9 ± 3.4	10.9 ± 6.1	17.9 ± 6.1	14.6 ± 4.3
	high	16.5 ± 2.0	22.7 ± 4.0	29.7 ± 6.8	14.6 ± 2.9
Energy ($^\circ/cycle$)	low	0.69 ± 0.34	1.05 ± 0.73	1.27 ± 0.49	1.20 ± 0.74
	medium	0.63 ± 0.3	0.98 ± 0.5	1.13 ± 0.32	0.92 ± 0.48
	high	0.26 ± 0.22	0.44 ± 0.28	0.35 ± 0.39	0.25 ± 0.31

The effect of reduced frequency of rudder updates is shown in Table 5, where *rare*, *often*, and *always* correspond to an update rate of 0.2 Hz, 0.5 Hz, and 5 Hz, respectively. Note that the communication to the boat operates at 5 Hz. Updating *often* results in the best speed, except for a close hauled, where *rare* updates are preferable. For stability, *often* is best for close hauled and beam reach, while *always* leads to better results for broad reach and downwind courses. We considered the average change in the rudder position weighted by the update rate as a surrogate for the energy consumption.

The update frequency and the gain of the underlying P-controller are related, and another experiment was run to study the effect of setting a gain of 4 (low), 7 (medium), and 11 (high). Table 6 shows the results indicating that a *medium* setting leads to high boat speed and good stability. The average rudder motion is decreasing with higher gain.

4 Discussion

One of the challenges in sailing is the large number of parameters that can be optimized. Some objectives like pointing high and going fast are conflicting and the best trade-off typically depends on the overall conditions, e.g., wind speed, sea state, or currents. For robotic sailing, the need to budget the energy required for sail and rudder control further complicates the problem. We approached the problem in a data driven fashion and presented some results illustrating how to obtain good parameter settings for rrMM-class sailing robots.

The analysis of the total data set shows that the boats have a fairly wide range of operation given their size. As expected, the boats are less stable on broad reach and downwind courses and stronger winds cause excessive heeling on a close hauled course. Adjusting the position and weight of the keel can substantially improve speed and stability. The central position is best for the moderate test conditions, although the results regarding the keel weight indicate that the heavier 470 g keel is preferable and further test runs for all position and weight combinations should be considered. Clearly, some results like the heeling for a centrally mounted keel may still be affected by outliers and confounding factors.

More interestingly, the boat speed is relatively insensitive to small changes in the actual sail position, even if the jib is sheeted closer than the main sail. Yet, this setup improves the course stability, particularly on a beam reach. This may indicate that the boat was slightly overpowered but should also be considered as a standard setup, as course stability can affect energy consumption. Another result from analyzing the complete data set is the preferred heeling for each course. Heeling has previously been proposed to control the sail position [6], and our data suggests that this can be a reasonable approach that should also include the respective point of sail.

As expected, adjusting the sails required considerable energy. However, the results also indicate that less frequent adjustments and control based on the apparent wind can lead to similar sailing performance. This should be taken into account when designing control software, e.g., good sailing performance may be possible even if only a limited a set of fixed sail positions can be achieved [4]. Similarly, a careful analysis of the parameters for rudder control can further reduce the energy consumption. In fact, medium update frequency and medium gain resulted in the best speed and stability for our P-controller. One explanation could be that too frequent rudder motion works against the boat's trim, i.e., in a situation where the boat would return to a stable sailing condition by itself, additional rudder motion could actually lead to larger deviations.

Given the complex non-linear interactions of boat, wind, and water it is currently not feasible to establish a comprehensive mathematical model of sailing, and a limited set of sensor data will not be sufficient to explain the boat performance completely [1, 2]. However, our results illustrate that such data can be helpful to get insight in the potential trade-offs with respect to different optimization objectives. Moreover, even though not all parameters can be measured and are known, the data can still serve as a benchmark indicating how much the settings can be improved. Our initial idea was to data mine a large set of data collected during arbitrary sailing in order to allow for unexpected results. Yet, it turned out that specific test runs can substantially speed up the analysis by producing stable episodes, and hence we consider establishing a set of test cases that can be used to quickly adjust and check the boat trim before sailing. We also intend to run further multiple boat tests, which proved to be a viable approach to get comparable results during actual sailing.

5 Conclusion

Finding optimal parameter settings for autonomous sailing is an intricate task involving multiple conflicting objectives. Currently, no complete mathematical model of sailing is available. Our results indicate that a comprehensive analysis of actual sensor data can lead to a better understanding of the boat performance in different sailing conditions. While some results confirmed our expectations, the analysis provided additional insight into the trade-offs involved in trimming a small robotic sailboat. Particularly, a "lazy" approach to controlling the sail position seems preferable with respect to course stability and energy management. However, the boat trim, including the position and weight of the keel, can impact not only on the boat speed, but also on the amount of control required. The results highlight the multi-criteria nature of optimizing robotic sailboat control and indicate that a reduced set of preferable parameter settings may be used for effective control.

References

1. Adriaans, P.W.: From Knowledge-Based to Skill-Based Systems: Sailing as a Machine Learning Challenge. In: Lavrač, N., Gamberger, D., Todorovski, L., Blockeel, H. (eds.) PKDD 2003. LNCS (LNAI), vol. 2838, pp. 1–8. Springer, Heidelberg (2003)
2. Anderson, B.: The physics of sailing. Physics Today, 38–43 (2008)
3. Sauzé, C.: A neuro-endocrine inspired approach to power management in sailing robots. Ph.D. dissertation, University of Wales, Aberystwyth (2010)
4. Sauzé, C., Neal, M.: MOOP: A Miniature Sailing Robot Platform. In: Schlaefer, A., Blaurock, O. (eds.) Robotic Sailing, vol. 79, pp. 39–53. Springer, Heidelberg (2011)
5. Schlaefer, A., Beckmann, D., Heinig, M., Bruder, R.: A New Class for Robotic Sailing: The Robotic Racing Micro Magic. In: Schlaefer, A., Blaurock, O. (eds.) Robotic Sailing, vol. 79, pp. 71–84. Springer, Heidelberg (2011)
6. Stelzer, R., Pröll, T., John, R.: Fuzzy logic control system for autonomous sailboats. In: Fuzzy Systems Conference, pp. 97–102 (2007)

Continuous Improvements to USNA SailBots for Inshore Racing and Offshore Voyaging

Paul H. Miller, Matt Hamlet, and Jeff Rossman

Abstract. On-the-water testing and design iterations identified numerous ways to improve the USNA SailBots, resulting in increased speed, reduced power consumption, greater maneuverability, increased power generation of the existing boats and the development of a new boat. Projects included a second-generation voyaging hull and balanced rig, improved bulb, keel and rudder/skeg designs, analysis of multihulls, incorporation of a wind turbine and solar panels for theoretically unlimited voyage duration and a "jibe-only" maneuverability code. On-the-water testing included a 21-nautical mile voyage and oceanographic bottom profiling. The result is an improvement of the probability of success of a transatlantic voyage from last year's 58% to nearly 90% and the reduction in expected time from 19.4 days to 16.9 days.

1 Introduction

Autonomous surface vessels offer significant opportunities for research and surveillance at reduced costs compared to manned vessels. One way to reduce costs is to reduce the vessel size. A six-year program at the United States Naval Academy has led to the construction of five, two-meter long "SailBots" that demonstrate the potential for these craft. The program's primary goal is educational, allowing the students' the opportunity to apply their naval architecture and systems engineering skills to a hands-on project. Secondary goals include the experience gained from participating in international technical competitions and developing the technology and usefulness of small autonomous vessels. Earlier design iterations of the USNA boats are described in previous IRSC papers (1-3). The first three boats were designed for short course racing in the SailBot Class. The last two boats were designed for longer voyages.

Paul H. Miller · Matt Hamlet · Jeffrey Rossman
United States Naval Academy
e-mail: phmiller@usna.edu

The USNA program follows an annual cycle beginning on the last day of competition. A list of suggested improvements is made by the graduating team. That list becomes the basis for projects and the new design by the students starting their final year in the fall. The students learn construction techniques by making the improvements and then go on to design and build their new boat. After competing with it they write up their suggestions and the cycle repeats. This paper describes the research and testing during the 2011-2012 academic year that resulted in a number of improvements, including:

- Bulk, keel, rudder and skeg designs
- Hull design for offshore use
- Analysis of multihulls for small autonomous sailboats
- Balanced Rig design
- Power management
- Power generation
- Sensor design and location
- Gybing algorithm

2 Program Overview

Originated in 2006, the USNA SailBot team is organized around two, two-course sequences held during the undergraduate students' senior year. Midshipmen pursue a four-year Bachelor's of Science degree in one of 22 majors. Roughly 65% of the students pursue a technical degree. After graduating the midshipmen are commissioned as officers in the United States Navy or Marine Corps and serve a minimum of five years to repay their free education.

While students from all majors are eligible to participate in the SailBot Program, almost all have come from either the Naval Architecture or Systems Engineering majors. Naval Architecture majors may elect to take one or both three-credit (six hours per week) courses during their final year. Titled, "Autonomous Surface Vessels", the fall elective focuses on design, while the spring course covers fabrication and evaluation. Systems Engineering majors participate in the program through their capstone engineering course. The fall course introduces project management and planning and culminates in a detailed proposal. The spring course implements the plan. Early in the program a problem in continuity from year to year was noted; having a team of just seniors meant that each year a good amount of time was spent bringing the new team up to speed. This was partially remedied by introducing fall and spring one-credit (two hours per week) courses for the juniors. The 2011-2012 academic year team included three naval architecture seniors and five juniors. As no systems engineering students signed up for the team, Matt Hamlet, a 2009 graduate served as the systems team.

3 Naval Architecture Improvements

As described in Reference 2, the keel and bulb of the racing SailBots are high aspect ratio designs. Previous iterations used standard airfoils and the bulb was designed using techniques more applicable to large yachts. At the conclusion of last year the team proposed that a new keel and bulb should be designed for GTB with the three goals of: decreasing drag, decreasing weight (a 2 kg goal) and improving righting moment. Working with Paul Bogataj and using CFD (computational fluid dynamics) and FEA (finite element analysis), a new keel and bulb were designed and built. Tank testing of the boat with the new components showed a drag reduction of up to 5.3% (fig. 2). The experiment was carried out in the USNA's 115 meter towing tank. The results may actually underpredict the resistance reduction however as alignment problems during the runs with the new keel indicated that the vessel was slightly yawed.

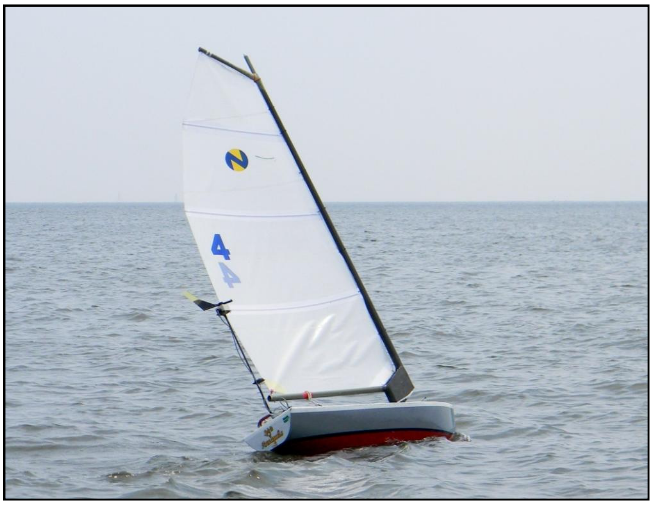


Fig. 1 *Spirit of Annapolis* (Boat #4) on the Chesapeake Bay, July 2012.

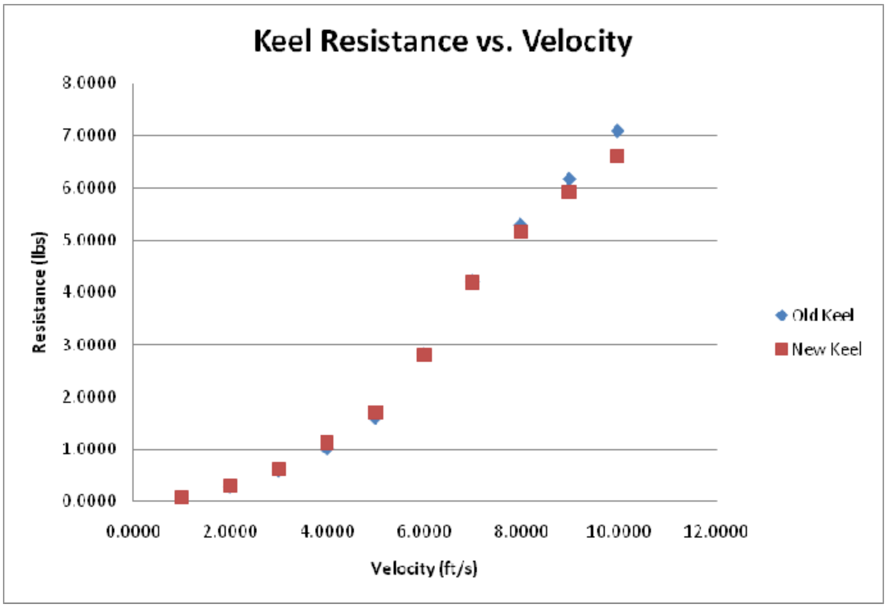


Fig. 2 Full-scale tank testing results comparing old and new keel/bulb designs on *Gill the Boat*.

The two major changes were materials based. First, the decision was made and the funding received, to outsource the bulb fabrication. This allowed us to use nearly pure lead rather than lead shot encapsulated in epoxy, increasing the density and decreasing the volume by 29%. The wetted surface area decreased by 28%, implying that both induced and friction drag would be reduced. The second was replacing the 17-4 ph stainless steel keel with Grade 5 titanium. While this lowered the keel density, the lower elastic modulus meant that to reach the target bending deflection of 15 cm at 90 degree heel, the keel structural shape would need to be increased. The old stainless keel used airfoils with thickness ratios ranging from 12-16%. The new keel increased the thickness to 14-16%. To offset the increased induced drag the planform area was reduced 7% by greater tapering toward the tip. This also had the benefit of raising the center of effort, reducing the heeling arm.

Titanium has a well-earned reputation for high cost, but in the end the total cost was comparable to the stainless keel. Although the raw material cost was 60% higher, the machining cost was lower and no heat treating was required. The final cost was only US$50 more; $1200 versus $1150. The developed airfoil shape was similar to a J5012, although with a bit sharper leading edge and a slightly extended thick midsection to increase the structural moment of inertia. Figure 3 shows the two keels and bulbs. The final weight of the new keel and bulb together was 2 kg less while the righting moment increased 6% as the keel was nearly 3 kg less and 1 kg was transferred to the bulb. To reduce interference drag the intersection of the keel and bulb was moved aft on the bulb, changing the keel sweep angle from six to nine degrees.

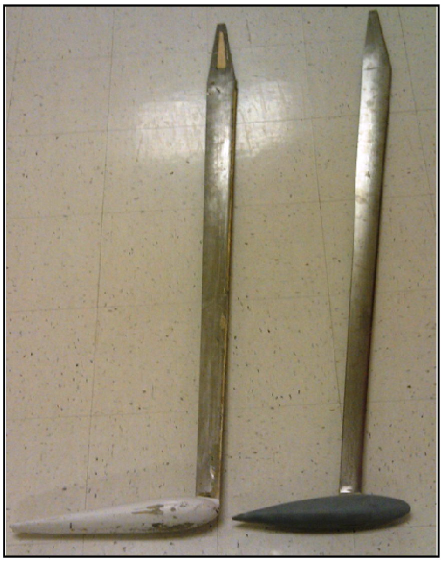


Fig. 3 Old (left) and new keel and bulb for GTB

Three other modifications were made to GTB over the year. One was to replace the cable-tie system of attaching the main sail to the mast with a luff-rope system. While this improved the aerodynamics and sail control it added weight. The second was to change the jib to a balanced design like those used on most model yachts. This increased the sail area and should have reduced the power consumption, although none was noted. The final change was to move the wind sensor from the top of the mast to a mast at the stern. This is described more in the section describing the changes to Boat #4, *Spirit of Annapolis*.

Lessons learned from sailing *Spirit of Annapolis* (SOA) (see Ref 3 for details on her design) showed that while her full keel design did provide exceptional directional stability and shed weeds easily (the prime design criteria), it also made the boat more directionally stable than desired. She was so directionally stable that she was unable to turn quickly enough to complete a tack. To improve her maneuverability greater yaw moment was required, combined with a lower yaw moment of inertia. Keeping the weed shedding criteria it was possible to modify the keel and remove approximately half the wetted surface area while increasing the yaw moment by moving the rudder further aft. The modified design was successfully tested and demonstrated that the boat was faster, based on comparisons with GTB and more maneuverable (she could now tack in light air and her turning radius was smaller) while maintaining excellent directional stability. The modification was not entirely successful however as she was still unable to tack in waves of greater than 0.3 m wave height, and by removing submerged volume she lost about 3 cm of freeboard. A third generation offshore design with smaller improvements was incorporated in boat #5 and a “gybe only” algorithm was developed to compensate for the inability to tack in all conditions. Figure 4 shows the old and modified keel and rudder designs.

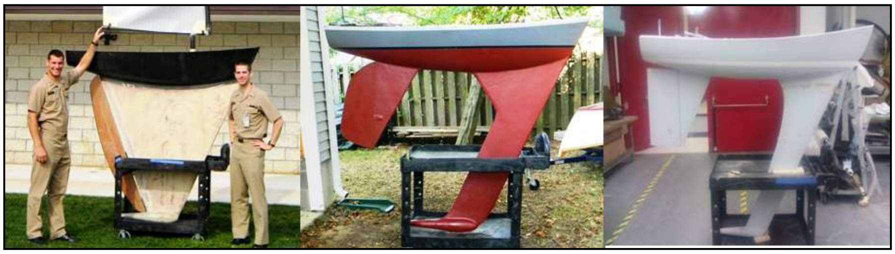


Fig. 4 SOA’s original(left) and modified (center) keel shapes with W2H

The main focus of this year’s team was designing and building an improved SOA. Boat #5, named, *Waiting to Happen* (W2H) including improvements in hull shape and appendages. The hull improvements included: reduced core density from 288 to 160 kg/m^3 to save weight, increased freeboard to provide more reserve buoyancy, greater waterplane area to improve immersion, more rounded sections to reduce wetted surface area and greater beam. Construction of the hull and deck used fiberglass rather than carbon to improve reception of the internal antennas, while the overall weight was reduced. The appendages used foam rather than plywood as the core material and were machined to 5-7% NACA 00 sections rather than the flat plate sections of SOA. The bulb on the new boat was 4 kg lighter, but with the greater beam the initial stability was the same. The Principal Characteristics of SOA and W2H are shown in Table 1.

Each student developed their own designs and submitted them to a “virtual race” run using the Velocity Prediction Program (VPP) presented in Reference 4. A course simulating a nautical mile long race that featured an equal amount of beating, reaching and running was used and run in wind speeds of 6, 15 and 24 knots. The boat that sailed the course most quickly was chosen as the design to build. Using the procedure described in Reference 5, the new design was predicted to complete the Northern Route across the Atlantic in 16.9 rather than 19.4 days.

Table 1 Principal Characteristics of USNA Boats 4 and 5

		SOA	W2H
LOA	m	2	2
LWL	m	1.86	1.85
Beam	m	0.33	0.48
Draft	m	1.5	1.5
Depth	m	0.43	0.41
Sail Area	m^2	1.9	1.8
Disp	kg	52.2	44
Cp		0.54	0.56
LCB % aft of FP		50%	53%
LCF % aft of FP		54%	56%
"SA/Disp"		14	15.4
"Disp/L"		226	192

The team also looked at multihulls. Catamarans and trimarans have demonstrated significantly higher performance than monohulls in many applications that do not include large changes in displacement. The potential existed for SailBots to have multihulls as evidenced by the somewhat successful Queen's, RMC and Waterloo teams in previous years. Ultimate stability and drag at the small sizes were identified obstacles and a design study combined with limited tank testing showed that the potential is there, but may not be as easy to attain as it is in larger craft. Both catamaran and trimaran designs were compared in VPP studies (Figure 5) and a catamaran design was tested in the tank to validate the VPP. Each design was limited to meeting the SailBot Class Rules and the weights and center of gravity were estimated for standard construction methods and likely locations of the systems. For instance, the CG of the catamaran was slightly higher than that of the trimaran due to the ability of the trimaran to store more equipment in its center hull. The estimated displacements of the monohull, catamaran and trimaran were 27, 24, and 23 kg respectively, while the sail area was kept constant. The results indicated that the monohull configuration of GTB had lower resistance throughout much of the normal speed range. As the sail area is almost unrestricted in the SailBot Class, resistance is a good measure of performance.

While the multihulls showed promise, two issues raised concerns. The first was weight. A large part of the weight of small autonomous vessels is the systems payload, including batteries. Multihulls are typically most successful when they have relatively low displacement to length ratios, but at these short lengths it may not be possible to reduce the payload enough to reach acceptable levels. Additionally, stability scales to the 4th power with vessel length, making small vessels significantly less stable than their bigger sisters. At the same time, the waves are just as large, creating a common situation where waves may lead to capsize. As multihulls are typically not self-righting, a multihull SailBot runs a high risk of ending its voyage early due to capsize.

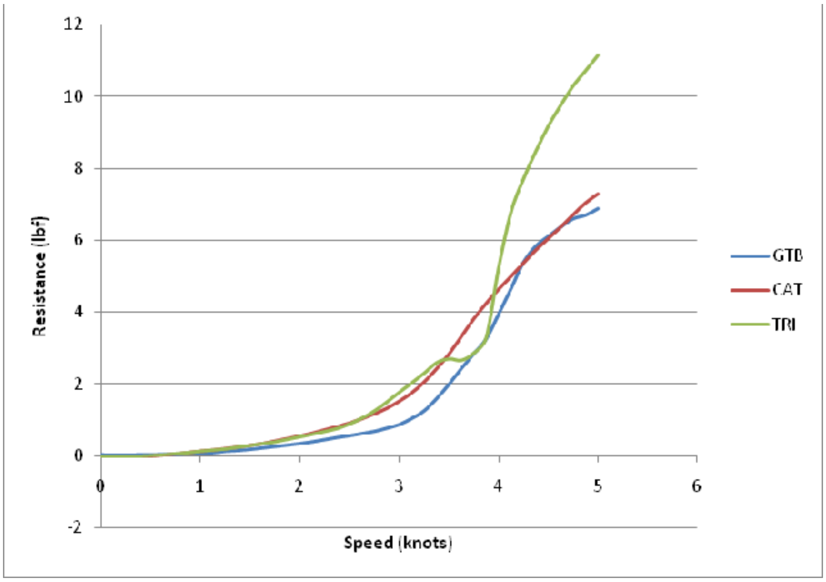


Fig. 5 Resistance Predictions for Monohull (GTB), Catamaran and Trimaran SailBots.

A highlight of the year was sailing GTB across the Chesapeake Bay and back. The 21 nautical mile voyage took six hours and was sailed in winds ranging from 0-21 knots and waves up to two feet. Figure 6 shows her in the middle of the Bay on the return leg. While GTB is not designed to cross an ocean, and the conditions were less than those expected in the North Atlantic, it was gratifying to see that she could complete a longer voyage.

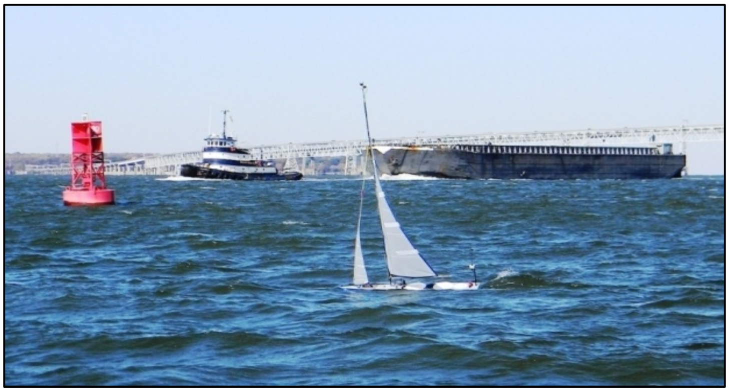


Fig. 6 Gill the Boat crossing the Chesapeake Bay autonomously

4 Power Management and Generation Improvements

Boat 4 was built with an experimental single sail balanced rig designed to have the fewest moving parts, resulting in theoretically higher reliability. In addition, the mast was offset aft of the leading edge to reduce the force required to trim the sail (see Fig. 1). The rig was designed with 15% of the area forward of the rotation axis based on the location of the center of pressure of an airfoil with the angle of incidence (Ref. 6). While the maximum forward location should allow almost 25% of the area forward of the rotation axis, concern existed over the ability to control the rig's rotation in light air, particularly the need to generate enough moment to overcome the friction in the mast tube. Experience with the rig showed that the percentage of forward area could be higher, and the new boat has 16% as a small improvement. The initial rig, designed for winds from 10-30 knots, did not allow the boat to perform well in light air, so a rig with 40% more area was created with a larger main and a small jib that rigged on an extension of the boom (Figure 7), resulting in 18% of the area forward of the rotation point. The basis for the sloop design was the widely-used Balestron or AeroRig. The "sloop" rig worked well, giving the boat improved performance in light air. With the keel and skeg modifications however the boat began performing acceptably in light air and the more complex rig was removed.

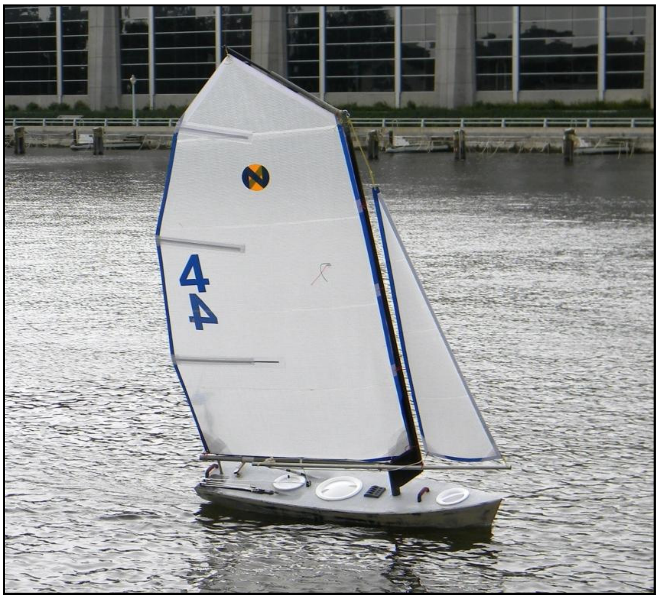


Fig. 7 SOA sailing with a balanced Balestron sloop rig in light air. The larger rig consists of boom and gaff extensions and the addition of a jib that rotates with the mast.

The major issue identified as restricting our program's ability to take on a multi-week voyage was power consumption, generation and storage. Lithium polymer, nickel metal hydrid and nickel cadmium batteries were tried, but ultimately a lithium-iron type was adopted. Lithium-iron batteries do not have the highest energy potential-to-weight ratio of off-the-shelf batteries, but are significantly safer than those that do. The chemical composition of Lithium-iron batteries make these extremely less likely to catch on fire and explode when being discharged or recharged. As a result, risk analysis models showed that minimal weight gained was an acceptable design specification loss that resulted in a significant reduction in potential of fire at sea. SOA was outfitted with three 12V, 36 Amp-hr, 2.2 kg Shorai batteries.

Before the charging system could be designed the power consumption was needed. Using a small Watts Up DC Inline Meter we recorded the power consumption on SOA over a six-month period of tests, resulting in an average consumption of 0.6 amps. Using the predictions from the routing analysis (Ref. 5) for the longest likely passage of 27 days, SOA would need 400 amp-hrs to cross the North Atlantic. As the battery storage capacity is only 108 amp-hr, improvements were needed. Options included: much greater power storage, on-board power generation, lower power consumption and shorter passage time. All four approaches were factored in to W2H's design.

W2H's larger hull volume was designed to hold four 36 Amp-hr Shorai batteries, raising the storage capacity to 144 amp-hr. A "150 mA" solar panel (40 x 13 x 2 cm) was mounted to SOA and was sailed in various conditions of cloud cover, time of day and windspeed (to factor in heel angle) to determine the actual output in sailing conditions. The average output over 14 sailing hours was only 20 mA, just 13% of the rated value. Six of the panels could be mounted on SOA or W2H giving a total charge of approximately 2 amp-hr per day versus 14.4 amp-hrs used. The deficit meant that the batteries would no longer power the ship systems in about 10 days. Even with the faster passage time for W2H, it was not enough. Another popular solution for power generation is the wind turbine. A 50-watt micro-turbine will be tested over the next year. To track the boat an independently powered GPS satellite tracker, the SX-1 by LiveViewGPS was mounted on deck and tests showed it works quite well. Battery life at 1 transmission per hour is six months.

From the conception of design, requirements and specifications were developed to reduce power consumption. Seventy five percent of power requirements are in support of moving the system actuators to control the rudder and sails. Actuators that incorporated a worm screw mechanical design were chosen, so that there would be little to no power consumption when the actuators were not moving, even though they were under load. High gear ratios were chosen to limit servo throw. Second, reduced power consumption was achieved through the semi-balanced rigs in SOA and W2H. Lastly, sensor input filtering as well as proportional controls were implemented to reduce the frequency of operation.

5 Vessel Improvements—Systems

Reliability improvements, which are directly tied to the success of USNA SailBot program, are proportionate to our ability to accurately test not only structural but also system components in a live environment through on-the-water trials. Over the past year these trials, primarily testing SOA and W2H showed unique system control challenges that we had not seen in previous hull designs.

SOA and W2H both presented a unique challenge to the autonomous control systems. Unlike the predecessor racing boats, these long distance voyaging hulls could not reliably tack through the eye of the wind. A simple solution of having the boat gybe when it would have normally tacked up wind was implemented, and that naturally led to numerous problems in other code routines. Through repeated simulated and on-the-water tests, data received from the onboard sensors showed that during a gybe, noise and error from many of the sensors could not be overcome. As a result, the control code is continually making determinations when it can trust sensor data and when it should be rejected. The solution was that during a gybe, heading and navigational data are temporarily turned off and reactivated by a timing device when the maneuver is complete. In general SOA will complete a gybe in 15-30 seconds with the loss of less than 10 meters of windward progress.

Setting fixed time limits in the control code is extremely inadvisable in systems control, as lot of error could arise. A cost and risk comparison was conducted, and the results showed that since this control code was designed for long passages, the error that could arise was acceptable, vice the added costs and sophistication of better sensing devices.

Although rudder and keel modifications made SOA and W2H more maneuverable, they are still significantly slower to react to changes in heading than previous hulls. Through on the water testing it was found that SOA and W2H were susceptible to getting stuck head-to-wind if the wind quickly shifted or wave action forced the bow up. In these situations it was observed that once the boat lost its forward momentum, the drag of its sails, rig, and hull would force the boat to begin to sail backwards, although it would remain in irons. The solution was to use the backward motion to the control systems advantage. Using the wind direct sensor, when a head to wind condition state is measured through a data filtering algorithm that takes in account for average and standard deviation tolerances, the control code will position the rudder in such a way that the boat will back down onto a closed haul course.

Regarding wind sensor data, while the top of the mast is the best location to get the most accurate wind direction reading it has the disadvantage of raising the vessel's center of gravity, leading to reduced performance. Additionally, the rotating masts on boats 4 and 5 required either indexing the mast's rotation or moving the sensor to an alternate location. A special purpose mast near the stern was built to hold the wind sensor and GPS antenna (see Figure 1). The mast height was chosen to locate the wind sensor at roughly the sail's center of pressure. The success of this location with SOA led the team to move the sensor for GTB to an aft mast.

The results of all the design improvements were incorporated in to the team's Velocity Prediction Program to generate new performance polars. The predictions indicated speed improvements in all conditions and up to one minute per nautical mile in optimum conditions, with SOA taking an average 1548 seconds and W2H 1488 seconds to sail one mile, an average of 2.4 knots. The increased speed and power system also improved the theoretical probability of success of a transatlantic voyage. Ignoring for a moment the fact that the boat would run out of electrical power before finishing the voyage, using the method described at last year's IRSC (Ref. 5), the probability of success for the voyage increased from 58% to nearly 90%. The biggest increases were due to the improved reliability seen in tests and the nearly 15% reduction in expected time to cross.

In addition to improving the SailBot and TransAtlantic boats, the USNA team also delved in to oceanographic research using one of the older SailBots. A wide-aperture sonar transducer from Airmar (DST800) was mounted on Boat #1. The boat then made passes collecting bottom profile data that was later plotted and compared to existing charts. This program will hopefully continue as it offers a lowcost way to survey and identify the frequent shoaling of the Chesapeake Bay.

Continuous improvement is a proven technique to achieve success and each year our students improve our boats while they apply their learned engineering knowledge. Our hope is to shortly put SOA and W2H out to sea.

Acknowledgments. Funding for the USNA SailBot team was provided by the Naval Sea Systems Command and Project Engineering Office – Ships, US Navy. Significant help on the project was provided by Professor Brad Bishop, Mr. Joe Bradsaw, the Technical Services Department, the USNA Model Shop and the Sail Loft of the United States Naval Academy. In addition to the authors listed, the team included Jordan Roach, Tyler Turbeville, Clayton Callander, Patrick Dillow, Adam Floyd, Joe Urbashich and Joseph Stark, all of whom contributed to this work.

References

1. Miller, P., Brooks, O., Hamlet, M.: Development of the USNA SailBots (ASV). In: 2nd International Robotic Sailing Conference, Porto, Portugal, July 9 (2009)
2. Miller, P., Beal, B., Capron, C., Gawboy, R., Mallory, P., Ness, C., Petrosik, R., Pryne, C., Murphy, T., Spears, H., Hamlet, M.: Increasing Performance and Added Capabilities of USNA Sail-Powered Automonous Surface Vessels (ASV). In: 3rd International Robotic Sailing Conference, Kingston, Canada, June 7 (2010)
3. Miller, P., et al.: Development of a Sail-Powered Autonomous Surface Vessel (ASV) for Trans-Atlantic Voyaging. In: 4th International Robotics Sailing Conference, Lubeck, Germany, August 16-17 (2011)
4. Martin, D., Beck, R.: PCSail, A Velocity Prediction Program for a Home Computer. In: The Proceedings of the 15th Chesapeake Sailing Yacht Symposium, Annapolis, MD, USA (January 2001)
5. Gibbons-Neff, P., Miller, P.: Route Planning for a Micro-transat Voyage. In: Schlaefer, A., Blaurock, O. (eds.) Robotic Sailing, vol. 79, pp. 183–194. Springer, Heidelberg (2011)
6. Gutelle, P.: The Design of Sailing Yachts, p. 46. International Marine Publishing Company, Camden (1979)

Human-Computer Interface for Doğuş Unmanned Sea Vehicle

Samet Batı, Hamdi Atacan Oğul, Cengiz Karaçizmeli, and Dilek (Bilgin) Tükel

Abstract. Unmanned vehicle systems are becoming increasingly prevalent on the land, in the sea, and in the air. Human-Computer interface design for these systems has a very important role in mission planning. The objective of this work is to design a unmanned sea vehicle and necessary software that can perform off-line path planning, vision management, communication, sensor control, and data management and monitoring of the unmanned sea vehicles.

1 Introduction

Sea power is a very important factor in military, commercial and transportation applications. There is a great interest in unmanned vehicles in the maritime domain from military and research institutes. The history of unmanned vehicles has its roots as far back as 425 BC [11]. The first self-flying robot bird is propelled by compressed air. Modern concepts were begun to be developed during the First and Second World Wars. The first navy USVs were radio-controlled drone boats which was used for collecting radioactive water samples (1946) and performing mine clearance operations (1960s). In 1985, the first modern USV "The Owl" was designed around the base of a jet-ski by International Robotic Systems Inc. In 1995, Navtec Inc was established and developed a fully autonomous navigation system using global positioning system and compasses along with a radar-based obstacle avoidance system. The MK II [5] was the first USV to be deployed for a

Samet Batı · Hamdi Atacan Oğul · Cengiz Karaçizmeli · Dilek (Bilgin) Tükel
Engineering Faculty,
Doğuş University,
Istanbul, Turkey
e-mail: `{2k35038,2k831007,2k836004,dtukel}@dogus.edu.tr`

real world mission in 1995 in the Middle East. There are also many academic research projects involved in the development of USVs. Several Catamaran type USVs have been developed such as SESAMO, an Italian catamaran USV [11]. In 2004, the Marine and Industrial Dynamic Analysis (MIDAS) Research Group at Plymouth University designed a twin-hull catamaran USV named Springer [7]. Springer research programme aimed to design and build a new advanced intelligent integrated navigation and autopilot (IINA) system.

Energy efficiency and use of renewable energy are very important for long missions. Wave powered USV "Wave Glider" has this ability and it crossed the Pacific Ocean, the longest distance ever attempted by a USV in 2012 [11].

The Dogus Unmanned Sea Vehicle (Dogus-USV) project is funded by Dogus University with the goal of reconnaissance and surveillance of the Turkish coasts. Being unmanned makes it possible for the vehicle to stay in the open sea for a long time without returning to base. It uses solar energy and maneuvers into different positions and paths using onboard cameras and global positioning system (GPS). As shown in Fig. 1, it is equipped with solar panels and batteries. The specifications of the vehicle are given in table 1.

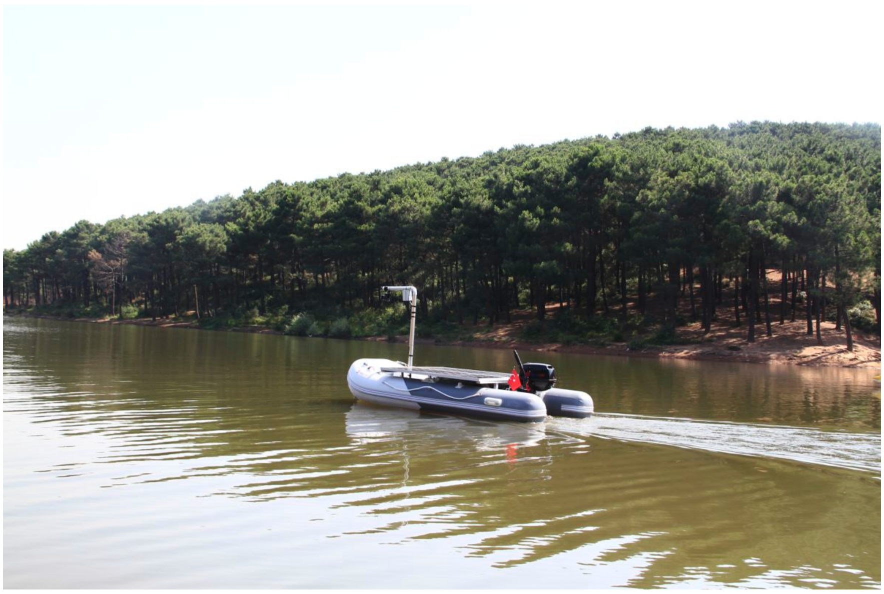


Fig. 1 Doğuş-USV trials on Aydos lake, 05, July, 2012

Table 1 The Specifications of Dogus-USV

Weight	256 kg
Length/Width/Height	330/151/110 (cm)
Power	5 HP
Speed	16 knot
Motor	Parsun F5ERL
Motor Cooling	Water
Batteries	Gel
Battery Capacity	100 Ah per battery
Number of Batteries	4
Solar Panels	Lorentz LA-Series
Solar Panel efficiency	For 10 year 90%, for 20 year 80%
Rudder Control	DC motor
Controller	U1 Ultra PC-Intel Atom Z520 single core, 1.33 GHz
Communication	WiFi, GPS, RF, 3G
Vehicle Controller	Arduino Mega 2560

2 System Components

The Dogus-USV was constructed as a sea vehicle that could be either remotely controlled or autonomous. The team designed and built the boat starting with the body of a inflatable boat and modified it to fit an electric motor, propellers, U1 computer, microcontroller, sensors, cameras, batteries, and solar panels. The components are selected and constructed to perform long missions in harsh environments without stopping or recharging. Dogus-USV is designed to be able to operate in harsh sea conditions; it is equipped with sensors that can instantly report errors to the control center. The movement of the rudder is achieved via mechanical steering with a geared motor and a powered chain.

2.1 Motor Specifications

The vehicle is powered by a high efficiency brushless motor. Max power is 4.8 KW (over 6hp) but this motor is rated at 5 hp continuous. It has a high current protection system [2]. The water-cooled version is preferred since the boat is designed to work over long ranges. It needs 100 amps of continuous current and a 10 second "power boost" of 140 amps for full speed. It needs a 48 V DC battery system. The inverter box is also water cooled. For the forward and reverse modes, relay based on-off control is applied. A servo motor is used for changing speed.

There is also another relay that cuts the power using a different channel in case of an emergency. Standard props are used. We have installed and tested our system in a lake with great results.

2.2 Solar Panels

The boat can supply its own energy and completely refill its empty batteries in 7 hours from its solar panels (Fig. 1). The power supply consists of two solar panels of 130 W, four marine gel batteries of 100 Ah for 12 V each. The panels are highly efficient and durable against sea water. The panel surface is coated with a hydrophobic layer. The dimensions of each solar panel is 669x1556x37mm and its weight is 16 kg [8]. Fig. 2 shows the electrical performance of solar panels.

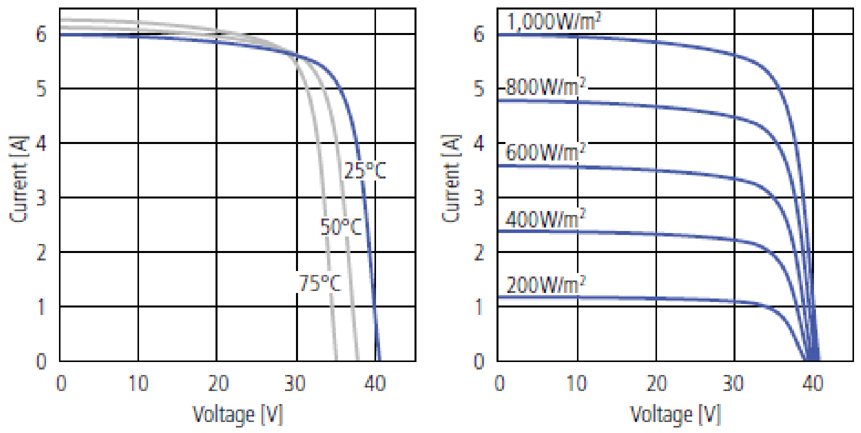


Fig. 2 Solar Panel Electrical Performance[5]

2.3 Batteries

The marine gel type batteries selected can power the electric motor for up to 6 hours at top speed. Lead acid gel batteries are common choice for sea vehicles to prevent foaming. Four accumulators have been placed in Dogus-USV with care being taken to ensure they balance correctly. The characteristics of the batteries are shown in Fig. 3.

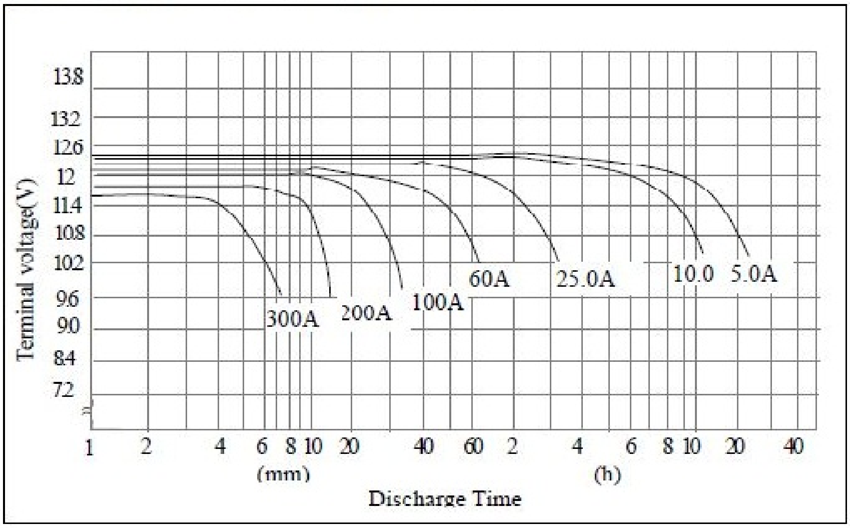


Fig. 3 Discharge curve of marine type gel batteries [1]

2.4 Rudder

The rudder is driven using a 214:1 gear head motor[4]. An external encoder[3] shown in Fig. 4 is mounted to measure the position of the rudder

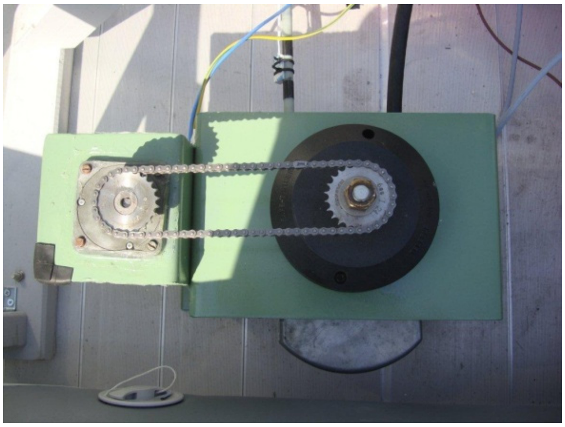


Fig. 4 Rudder mechanism an external encoder- ϕ 50mm-500 pulses/turn[3].

2.5 Inverter

An inverter is a critical component, performing voltage and DC/AC conversion to recharge the batteries which it can do in under 5 hours.

2.6 Solar Panel Charge Controller Circuit

This is a waterproof circuit controlling the current flowing from the solar panels to the batteries.

3 Controller Hardware

A general system overview of unmanned surface vehicles is shown in Fig. 5. Three different controllers have been used in our system: a master computer, an onboard computer and a microprocessor based controller. The master computer in the base station runs the Windows 7 operating system; the onboard computer is a U1 computer with the Debian 6.0 Xfce Linux with kernel 2.6.38 operating system which communicates with the microcontroller and master computer. The microcontroller board gathers sensors information and sending it to the onboard computer and receives commands back via a 9600 bps software serial port. The master computer displays and monitors the current status of the vehicle. The connections between the computers are controlled by a trigger system which checks for discontinuities in the links.

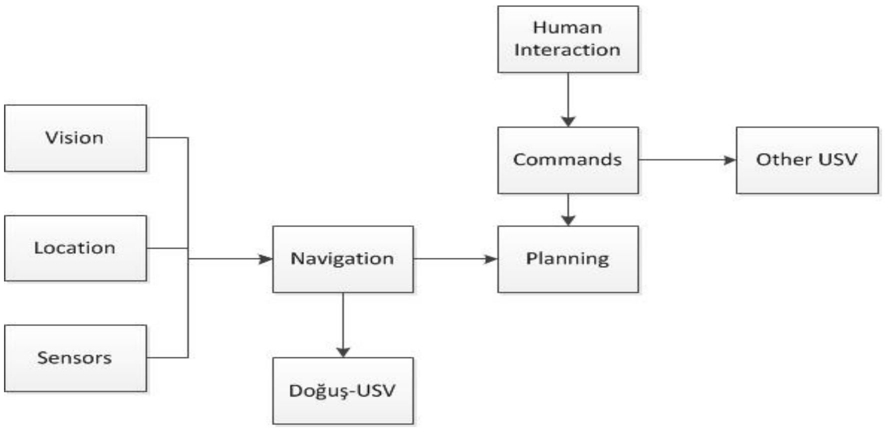


Fig. 5 General System Overview

The connection between master and onboard computers was implemented by sending and receiving commands over TCP/IP and UDP sockets. UDP packet contains GPS location and status information. The TCP socket can issue speed and direction commands.

4 Human-Computer Interface

The Human-Computer interface (HCI) design for Doğuş-USV (Fig. 6) aims to provide a user friendly environment for mission planning and monitoring. The developed software tools should have the capability to provide operators of a multi-vehicle, common control station with decision support for real-time re-tasking and re-planning of multiple assets and the ability to visualize data and information. The interface software for this system is implemented in the C# programming language, with communications developed in Java. It includes four main sections:

- Vision management
- Off-line planning and localization
- Vehicle control
- Monitoring and data acquisition

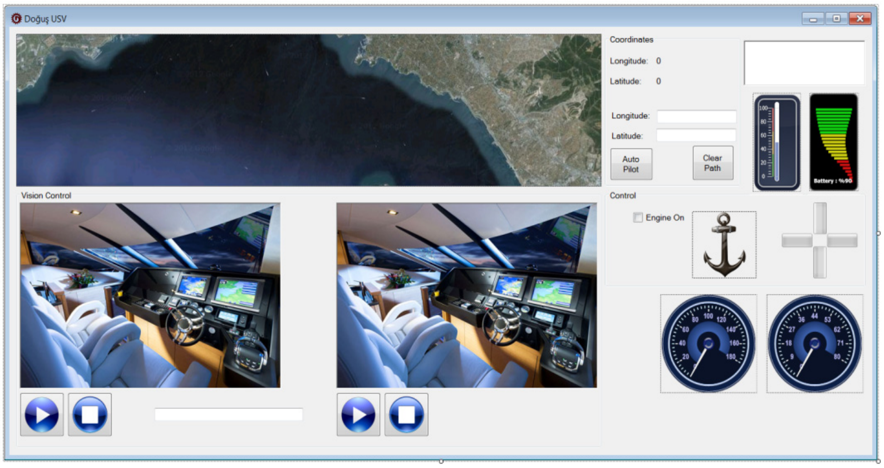


Fig. 6 Computer-Human interface main screen

HCI provides the communication and visualization mechanism between the operator and the navigation system on board. The operator uses this interface to receive video streams, as well as other data acquired onboard, and to define the route, given as a set of locations which can be located with a geographical information system using a simple point-and-click interface. As part of the HCI, littoral maps of the Fenerbahçe and Marmara Sea were used in conjunction with an onboard global positioning system (GPS). A human operator can monitor the coordinates as longitude and latitude which aids the determination of a point for manual usage. The USV is commanded using a joystick. Some other tasks, such as motor on/off, are assigned to additional joystick buttons. For turning while maneuvering forward with a constant speed, the joystick's slider property was used while the speed was fixed. The polling sets only one direction, backward or forward. The monitoring module was developed using DevExpress; adding a set of gauges and different kinds of indicators to the user interface. The speed, position and battery status of USV and motor temperature can be monitored. The design is flexible and it is very easy to add new properties. In addition to DevExpress gauge controls, progress bars are also used to monitor joystick polling. The interface provides video streams to the operator for monitoring the vehicle path. Following the route on a map makes it easier to visualize spatial coordinates.

4.1 Vision Management

Two different Internet Protocol Cameras (IP Cam) are installed in the USV. Each camera is built upon a 640 x 480 pixels resolution image sensor [2] with digital output. The camera (Fig. 7) allows about 330 degrees of horizontal pan and about 70 degrees of tilt and has night vision capability supported by ten infrared illuminators built in. To stream these images, the video sub-library of AForge.NET[9] was used. AForge.NET is an open source C# framework designed for developers and researchers in the fields of Computer Vision and Artificial Intelligence, image processing, neural networks, genetic algorithms, fuzzy logic, machine learning and robotics .

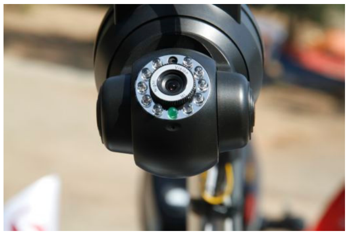


Fig. 7 Night vision Ip Camera

4.2 Offline Planning and Localization

A map was used to see the real time position of Doğuş-USV by obtaining the coordinates from an onboard GPS. While moving the mouse, the user can see the longitude and latitude coordinates on the map, which helps with him for manual task planning mode. In addition to these properties the user can plan a path , drawing it in a similar manner to a painting program, sending the coordinates to the on-board computer. Drawing a path using a mouse clicks is easy and user friendly. The designed route can be simulated as shown in Fig. 8. In the figure, the small dots represent the user entry points. When the operator presses the “simulate button”, the simulation starts and the D shaped circles appear which represent the USV positions.

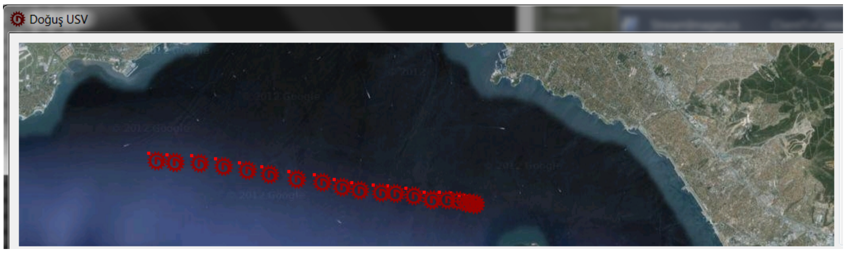


Fig. 8 Doğuş-USV Path Simulation

Navigation without obstacle avoidance, however, provides only limited capabilities in a real-world mission[7]. In order to reduce the reliance on operator oversight, we will add an obstacle avoidance capability to our system in the next step of our project.

4.3 Vehicle Control

A low level layer was run on the Arduino microcontroller and interfaced to the motors, sensors and GPS. It received commands via a serial port from the onboard computer which was running a Linux based operating system. Through this interface the onboard computer could send target positions and receive data from the sensors and status information.

4.3.1 Connection through Master Computer to Arduino

The main mission of the onboard computer is managing the connection between the master computer and an Arduino microprocessor shown in Fig. 9 as a link bridge to receive and transmit for both sides. The application receives control data from the joystick via the master computer and sends it to the Arduino, and receives data from the Arduino including GPS information, temperature and speed. This data is then sent to the master computer which than displays the status of Doğuş-USV to the user.

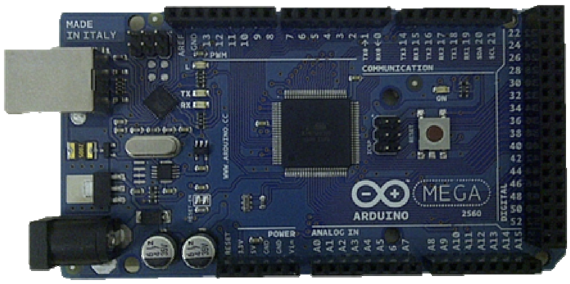


Fig. 9 Arduino Microcontroller Board

The master computer application was written in C#, the onboard computer application was written in Java, and the Arduino code was in a form of C++. This caused problems with cohesion. The data flow itself was designed as a byte stream The solution was efficient, however transmission of the datasuffered from noise problems. We tried to solve the problems using by removing the noise and re-transmitting the data.

An Arduino Mega was used to control motors and gather sensor and GPS data. It sent the sensor data to the onboard computer, and it received the control data from the onboard computer. The Mega's 54 digital I/O pins are enough to receive sensor data and transmit the control data.

4.4 Monitoring and Data Acquisition

The onboard control and navigation subsystems are responsible for sensor data acquisition and actuator control based on tasks uploaded to it from the base station. The communications were implemented using Wi-Fi links which transmit data, commands and video stream between the Doğuş-USV and the base station. A single 1024 byte UDP or 1024 byte TCP packet was sent at a rate of 5 Hz.

The compression method of the images captured by cameras mounted on the USV is M-JPEG [2]. Motion JPEG (M-JPEG) is a video codec where each video frame is separately compressed into a JPEG image. At low bandwidth availability, priority is given to image resolution. The available camera can capture and compress, 15 images (640 x 480 pixels resolution) per second, and then make them available as a continuous flow of images over the network to the base station.

5 Conclusions

We have shown our approach to building an unmanned sea vehicle with the goal of reconnaissance and surveillance. In July 2012, we tested our hardware and basic software components. In the next stage, we will improve the path-planning and sensor integration parts of our project.

Acknowledgements. The authors would like to thank Doğuş University for funding this project and Prof. Dr. Elif Çepni and Prof. Dr. Mitat Uysal for their supports.

References

1. Battery, `http://www.mutlu.com.tr/get_content.cfm?id=6` (accessed June 2012)
2. Camera, `http://www.chargerbuy.com/blog/files/2011/01/Foscam-FI8918W-Quick-Installation-User-Manual2.pdf` (accessed June 2012)
3. Electric Motor, `http://www.parsunmarine.com/product.php?about=1&pzid=1&id=1` (accessed June 2012)
4. Encoder, `http://www.opkon.com.tr/Urun_Detay.aspx?ID=62` (accessed June 2012)
5. Geared Motor, `http://www.buehlermotor.de/C12572C600247071/vwContentByKey/W273RD58524WEBREN` (accessed June 2012)
6. Owl MK II, `https://www.wamiltons.com/project/index.php` (accessed June 2012)
7. Springer: The Unmanned Surface Vehicle, Plymouth University, UK, `http://www.springer-usv.com` (accessed June 2012)
8. Solarpanel, `http://www.lorentz.de/en/support/download?docsearch=la170-24s&pgroup=la-series` (accessed June 2012)
9. Vision software, `http://www.aforgenet.com` (accessed February 2012)
10. Larson, J., Bruch, M., Ebken, J.: Autonomous navigation and obstacle avoidance for unmanned surface vehicles. In: Proceedings of the SPIE, vol. 6230, pp. 623007 (2006)
11. Motwani, A.: A Survey of Uninhabited Surface Vehicles. MIDAS Technical Report: MIDAS.SMSE.2012.TR.001 (2012)

An Experimental Validation of a Robust Controller with the VAIMOS Autonomous Sailboat

Fabrice Le Bars and Luc Jaulin

Abstract. A sailboat is a strongly non-linear system that has, however, been proven to be easily controllable. Indeed, its mechanical design has been evolved over thousands of years with two main concerns: having a fast, reliable and efficient vehicle which can be easily controlled by humans. This article describes the functionality, the validation process and the performance of a simple controller, inspired by what navigators do, through tests made on the sailboat robot *VAIMOS* built by IFREMER for oceanography. This controller requires tweaking a few parameters with real physical meaning while ensuring accurate trajectory following, needed to make oceanographic measurements in a specific area.

1 Introduction

In order to make oceanographic measurements, IFREMER (Institut Français de Recherche pour l'Exploitation de la Mer) has designed a sailboat robot (see e.g. [4] [1] [16] [15] [6] [17] [2] [14] [18] for more information on autonomous sailboat robots) with a length of 3.65m based on a Mini-J hull: *VAIMOS* (Voilier Autonome Instrumenté pour Mesures Océanographiques de Surface, see figure 1 and [7] [11] [12]). This robot has:

- An oceanographic probe and pumps that make it possible to measure various parameters near the water surface and at a depth of about one meter (temperature, salinity, chlorophyll, turbidity, etc.).
- A Linux-based embedded computer.
- A weather station that measures the wind speed and direction as well as GPS position.
- An AHRS (Attitude and Heading Reference System).

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

- A Wifi and Iridium communication system.
- Actuators for sail and rudder control (step-by-step motor that controls the maximum sail angle and servomotor to control the rudder angle).

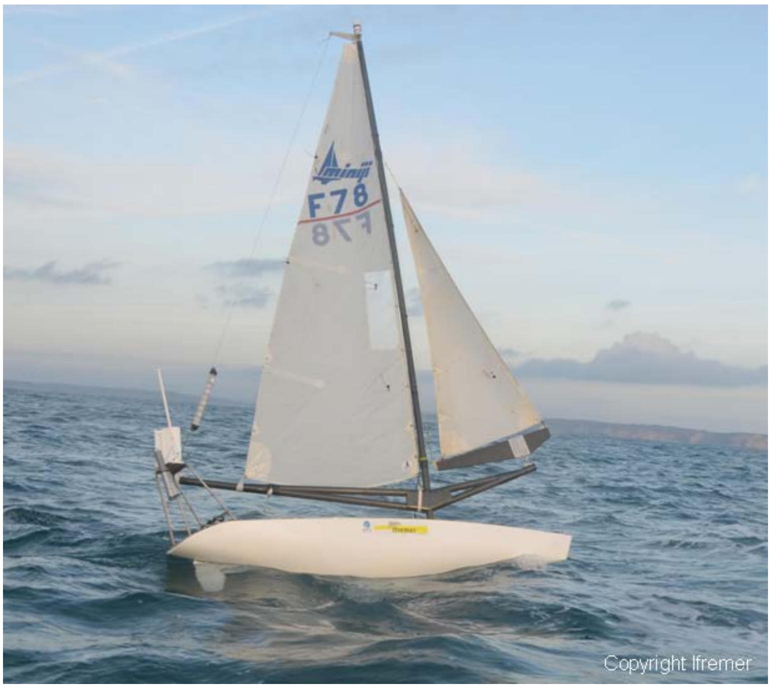


Fig. 1 The autonomous sailboat *VAIMOS* in the sea.

Its aim is to assist and / or replace currently used oceanographic boats and fixed or floating buoys, which have several drawbacks: oceanographic boats need a crew and their missions are expensive, it is sometimes difficult to set up fixed buoys in deep seas, floating buoys move randomly according to wind and currents and do not always stay in desired areas, etc. An autonomous sailboat has several advantages:

- Almost unlimited energy: it uses wind to move, sun and sea to charge its batteries while its power consumption is low compared to that of a motor for instance.
- Interesting payload capabilities with respect to its dimensions.
- Accuracy (vs floating buoys) and ease of setup (vs fixed buoys). The operators need only program a predefined trajectory and launch the sailboat from a harbor: it should then go on the area of interest and cover it while storing measurements and communicating by satellite, sending subsets of data and status information before coming back to its harbor.
- Cheap (about 20000€, probe excluded).

In addition to oceanographic missions, this type of robot could be used for other applications [3] [5]:

- Continuous harbor main entrance monitoring. Thanks to their important energetic autonomy and their low cost, several robots like *VAIMOS* could be deployed to monitor local surface and submarine traffic and would notably reinforce systems currently used.
- Heterogeneous swarms of robots. Submarine swarms of autonomous robots can quickly and covertly monitor a given area, however the use of small submarines alone can have several limitations:
  - It is more difficult to retrieve energy underwater. Therefore, small autonomous submarine robots can rarely work for long distances or time.
  - Localization and communication are difficult in passive mode (it is not always possible to use active sensors if covertness is a requirement).

Adding autonomous sailboat robots to submarine swarms could solve some problems: long distance transport, energy backup, communications with the base or surrounding boats, localization thanks to GPS, etc.

VAIMOS has been automated to be able to cover autonomously an area as accurately as possible, while saving energy. For this purpose, a line following algorithm [10] [11] has been developed to guarantee that the robot always stays in a predefined strip (of 25m width for example), despite maneuvers inherent to course changes, tacks, etc. In this way, the sailboat becomes as accurate as a motorboat. Because some courses are difficult to follow depending on wind orientation (a problem inherent to any sailboat), its regulator has 2 types of strategies: nominal route or tack. A basic controller stage provides heading control. In tack mode, heading should be around 45° from the wind orientation (this is the close-hauled angle). Therefore, the boat oscillates around the wind direction, the amplitude of the oscillations being the strip width, and the sail angle is at its minimum. In nominal route mode, the heading to follow is around the line made by the 2 current waypoints (the previous one and the next one), with an attractiveness angle to the line depending on the distance to the line (maximum of 45° for example). The sail is opened depending on the wind direction and the desired heading using a simple formula.

The main idea of this article is to show that in order to have a reliable autonomous robot, theoretical validation of its algorithms, using interval methods for example (see [9] for more information on interval analysis) is needed but we must also validate the assumptions made (state equations, bounds on errors, coefficients, etc.) using other methods to complete the validation process. For these reasons, we first made a theoretical validation using interval analysis and Lyapunov methods [12]. Then, a HIL (Hardware In the Loop) simulator was developed. Finally, real experiments in Brest harbor and between Brest and Douarnenez (Brittany, France) were made.

The robust sailboat controller developed will be explained in section 2. Section 3 will be about the theoretical validation of the controller. The HIL simulator used as an additional validation method and tool to plan real experiments will be described in section 4. Finally, section 5 will show the results of real tests with *VAIMOS*.

2 Controller

Due to the functioning of a sailboat, some headings are difficult to follow depending on wind orientation. Therefore, most of the controllers have 2 different modes: nominal, when the heading to follow is feasible, or tacking i.e. oscillation around 45° from the wind orientation (close-hauled angle), when it is not directly feasible. Most existing regulators use waypoint following instead of line following for reachable headings [17]:

- The robot takes a heading in the direction of the waypoint.
- The waypoint is reached when the boat is in a predefined radius.
- Unfortunately, nothing prevents it from drifting between waypoints (because of water currents, wind, etc.).

Some use also potential fields to define no-go zones for the sailboat [14], cost functions, fuzzy logic and the polar speed diagram of the sailboat (VMG: Velocity Made Good) [18]. One of the first sailboats using a line following approach was *Atlantis* (and *HWT X-1*, its successor) [5] [4].

The inputs of a sailboat such as *VAIMOS* are δ_r the rudder angle and $\delta_s^{\max}$ the sail maximum angle (δ_s, the angle of the sail should depend on $\delta_s^{\max}$ and the wind orientation w.r.t. the sailboat orientation). The outputs are the position $\mathbf{x}$ obtained from the GPS and expressed in a local coordinate system, the wind speed V and orientation ψ from the weather station and the heading θ of the sailboat from the AHRS, used as a compass (see figure 2). Note that it is possible to avoid using a weather station and keep $\delta_s^{\max}$ as a constant using the methods described in [19]. The line following controller of *VAIMOS* (described in detail in [10], [11] and [12]) is composed of several parts (see figure 3):

- A primitive controller stage for heading control. The angle of the rudder is set by proportional regulation w.r.t desired heading $\bar{\theta}$ if we are close to this desired heading, bang-bang regulation (two point control) if far from desired heading:

$$\delta_r = \begin{cases} \delta_r^{\max}.\sin\left(\theta-\bar{\theta}\right) & \text{if } \cos\left(\theta-\bar{\theta}\right) \geq 0 \\ \delta_r^{\max}.\text{sign}\left(\sin\left(\theta-\bar{\theta}\right)\right) & \text{otherwise,} \end{cases} \tag{1}$$

  with $\delta_r^{\max}$ the maximum rudder angle. The sail is opened depending on the wind direction and the desired heading using a simple formula:

$$\delta_s^{\max} = \frac{\pi}{2}.\left(\frac{\cos\left(\psi-\bar{\theta}\right)+1}{2}\right). \tag{2}$$

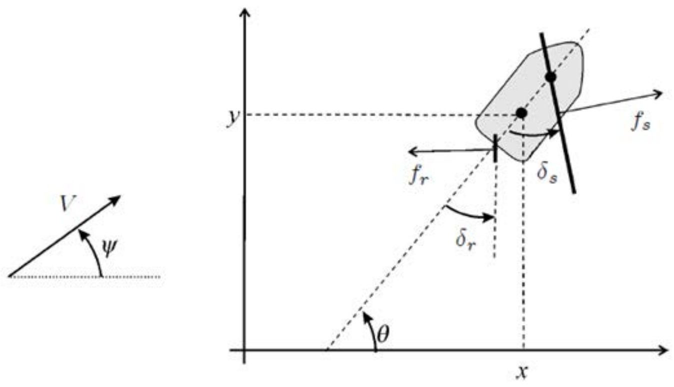


Fig. 2 Notations: ψ and V define the wind orientation and speed, f_s is the force of the wind on the sail and δ_s the angle of the sail, f_r is the force of the water on the rudder and δ_r the rudder angle, $\mathbf{x} = (x,y)$ and θ are the boat position and orientation.

- A supervisor decides between 2 modes: nominal route or tack. It should always send feasible headings to the primitive controller. In nominal route mode, the heading to follow is given by the line made by the 2 current waypoints **a** and **b**, with an attractiveness angle to the line depending on the distance to the line. In tack mode, the heading is $\pm 45°$ from the wind direction where $45°$ is the close-hauled angle: the sailboat oscillates around the wind angle, the amplitude of the oscillation being the width of the strip around the line.
- A navigation manager sends lines formed by 2 waypoints $\mathbf{a}_j$ and $\mathbf{b}_j$ to the supervisor and validates lines. A line is validated when the sailboat reaches the perpendicular of the line at $\mathbf{b}_j$, i.e. the validation condition is:

$$\left\langle \mathbf{b}_j - \mathbf{a}_j, \mathbf{x} - \mathbf{b}_j \right\rangle \geq 0 \qquad (3)$$

3 Theoretical Validation of the Controller

In order to validate the line following controller developed, several tools have been used:

- Validation using interval analysis and Lyapunov methods.
- HIL (Hardware In the Loop) simulator.
- Real experiments in Brest harbor and between Brest and Douarnenez (Brittany, France).

A new interval method for nonlinear stability analysis has been developed (it is described in detail in [12]). The main idea is to represent uncertain systems by

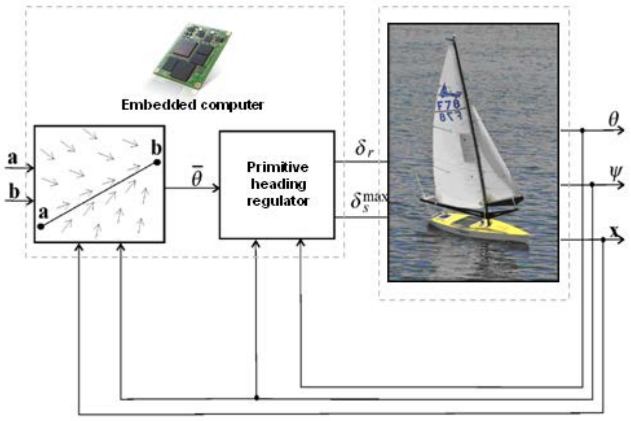


Fig. 3 Principle of the line following controller of *VAIMOS* (with previous notations).

differential inclusions and then apply Lyapunov analysis methods to transform the stability problem in a set inversion problem (see [13] and [9] for more information on interval analysis and set inversion problems). In this way, it is possible to demonstrate that for all possible perturbations:

- There exists a subset of the state space which the system cannot escape when it enters it.
- If the system is outside this subset, it will not stay outside forever.

However, even if these methods can validate theoretically the robustness of the controller (i.e. the robot will stay in a strip around its target line), additional methods must be used to adjust the hypothesis (state equations, bounds on sensors errors, etc.). To prepare as much as possible for real experimentation, an HIL (Hardware In the Loop) simulator (inspired by [8]) has been developed to simulate the robot's trajectory and sensor data on a computer while using the controller directly on the sailboat as if it were at sea.

4 HIL Simulator

Most existing simulators use the polar speed diagram of the sailboat to determine its movement, or alternatively use several predefined scenarios. Therefore, they might miss some singular situations which should be detected and acted upon to fully validate a controller. State equations inspired from [8] were used for our controller validation purposes:

$$\left\{\begin{array}{ll}
\sigma & = \cos(\theta - \psi) + \cos(\delta_s^{\max}) \\
\delta_s & = \left\{\begin{array}{ll} \pi - \theta + \psi & \text{if } \sigma < 0 \\ \delta_{s\max} \operatorname{sign}(\sin(\theta - \psi)) & \text{otherwise} \end{array}\right. \\
f_r & = \alpha_r v \sin(\delta_r) \\
f_s & = \alpha_s V \sin(\theta + \delta_s - \psi) \\
\dot{x} & = v \cos(\theta) + \beta V \cos(\psi) + V_c \cos(\psi_c) \\
\dot{y} & = v \sin(\theta) + \beta V \sin(\psi) + V_c \sin(\psi_c) \\
\dot{\theta} & = \omega \\
\dot{\omega} & = \frac{(l - r_s \cos(\delta_s)) f_s - r_r \cos(\delta_r) f_r - \alpha_\theta \omega + \alpha_w h_w}{J_z} \\
\dot{v} & = \frac{\sin(\delta_s) f_s - \sin(\delta_r) f_r - \alpha_f v^2}{m} \\
\ddot{\varphi} & = \frac{-\alpha_\varphi \dot{\varphi} + f_s h_s \cos(\delta_v) \cos(\varphi) - m_{eq} l_{eq} g \sin(\varphi)}{J_x} \\
\dot{\varphi} & = \dot{\varphi}
\end{array}\right. \tag{4}$$

with v the sailboat speed, ω the rotation speed, φ the roll, assumed to be pendular, with coefficients α_φ (fluid friction), h_s (height of the sail force application point), m_{eq} (mass of the equivalent pendulum), l_{eq} (length of the equivalent pendulum), J_ω (inertial moment), V_c and ψ_c the sea current speed and orientation, h_w the height of waves, β the coefficient of drift due to wind, α_r, α_s, α_f, α_θ, α_w various fluid friction coefficients and r_r, r_s, the distance from the sailboat mass center to the rudder and mast respectively (see also figure 2). Then, the behaviour of the state equations was tested on a 3D simulator. Using these results, an HIL (Hardware In the Loop) simulator was finally developed to simulate the robot trajectory and sensors data on a computer depending on the input lines, expected wind and sea conditions and a user-defined initial position while using the developed controller on the embedded computer to control the robot actuators as if the sailboat were in the sea to study mechanical wear as well as the robustness of most of the embedded electronics. HIL simulation means that the real hardware (here the embedded computer and actuators) is used in simulations (see figure 4):

- First, the simulator using the state equations previously defined is started on a normal computer with a user-defined initial state. It generates simulated sensor data (θ, ψ, $\mathbf{x}$) from rudder (δ_r) and sail ($\delta_s^{\max}$) inputs that will be decided by the controller and user-defined sea (h_w, V_c, ψ_c) and wind (V, ψ) conditions.
- Then, the controller is started on the embedded computer of the sailboat. It takes a list of lines to follow (formed by waypoints $\mathbf{a}_j$, $\mathbf{b}_j$) as for a real experiment and controls its actuators as usual, but also sends a copy of its outputs for the actuators (δ_r and $\delta_s^{\max}$) to the simulator, and uses simulated sensor data (θ, ψ, $\mathbf{x}$) rather than the data from its real sensors.
- Finally, log files generated by the controller are retrieved and displayed in real time using GOOGLE EARTH and a custom-built dashboard.

The communications are made possible by the fact that all the embedded devices are accessible by Wifi.

Several simulations were made in different configurations to prepare for real experiments, for example navigating a course of more than 100km between Brest and

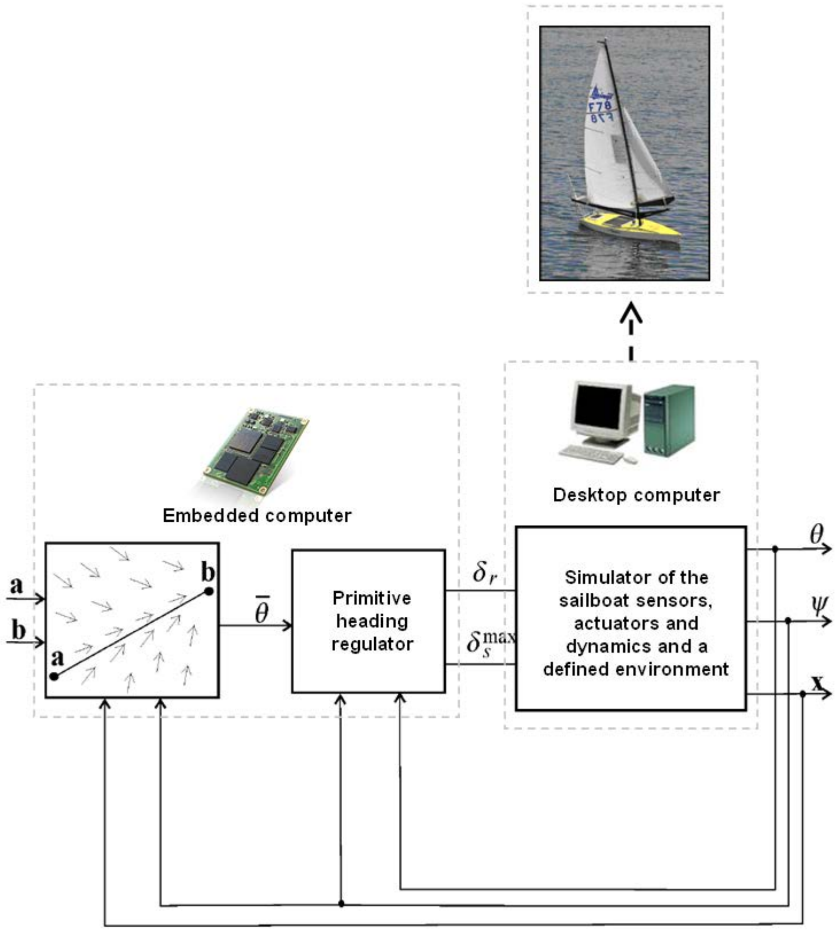

Embedded computer
Desktop computer
a
b
$\bar{\theta}$
Primitive heading regulator
δ_r
$\delta_s^{\max}$
Simulator of the sailboat sensors, actuators and dynamics and a defined environment
θ
ψ
$\mathbf{x}$

Fig. 4 Principle of the HIL simulator of *VAIMOS*.

Douarnenez. A first simulation made with a North wind of 14 knots is shown in figure 5. An initial position just in front of Brest harbor was indicated to the simulator at start. Other simulations were made as the expected weather conditions for the date fixed for the real test was changing, and to test different ways of covering the bay to minimize the tacks and shorten the total time (here around 40 hours).

5 Real Experiments

Real experiments of particular trajectory patterns have been made in Brest harbor to test *VAIMOS* in all wind configurations while taking oceanographic measurements for IFREMER (see figure 6). Small real tests are important. For example, some

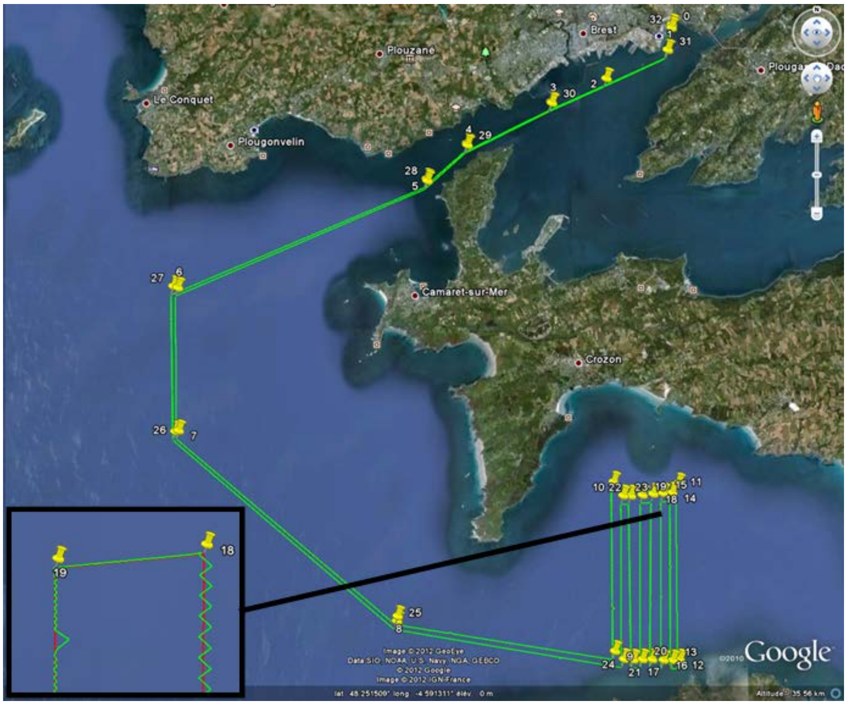


Fig. 5 Simulation of *VAIMOS* going from Brest to Douarnenez. The desired trajectory (red lines made by yellow waypoints) and the effective (simulated) trajectory (green) seem to overlap. However, if we zoom, we see there were tacks. We can also see sail motor calibration steps (every 2 hours) which make the sailboat drift for one minute.

magnetic perturbation problems making it necessary to move the AHRS far from the rest of the electronics were detected during these tests.

Finally, a long autonomous mission between Brest and Douarnenez on the 17-18th January 2012 was attempted with *VAIMOS* (see figure 7). It made more than 500 oceanographic measurements over 105km in 19h. The wind was around 12 knots and from South.

During the mission and after, a dashboard was used to analyse all the log files produced by the embedded program. For example, near the end of the experiment, we see that the sail angle measured by the weather station (which is on top of the sail and has an integrated compass), in purple is incoherent with the one deduced from the input, in pink. This was probably due to the mechanical problem in the sail control system that we discovered at the end. Because it was during the night, it was difficult to see the sail angle without the dashboard (see figure 8).

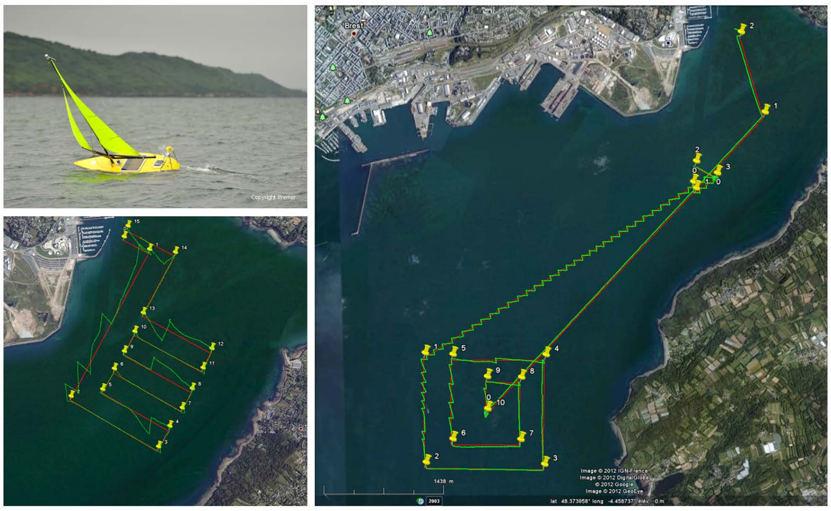


Fig. 6 Tests in Brest harbor. Desired trajectory (red lines made by yellow waypoints) and effective trajectory (green). South-West wind on the left, South-West wind of around 15 knots on the right. 27 km (17 nm) was travelled in less than 5 hours in the journey shown on the right.

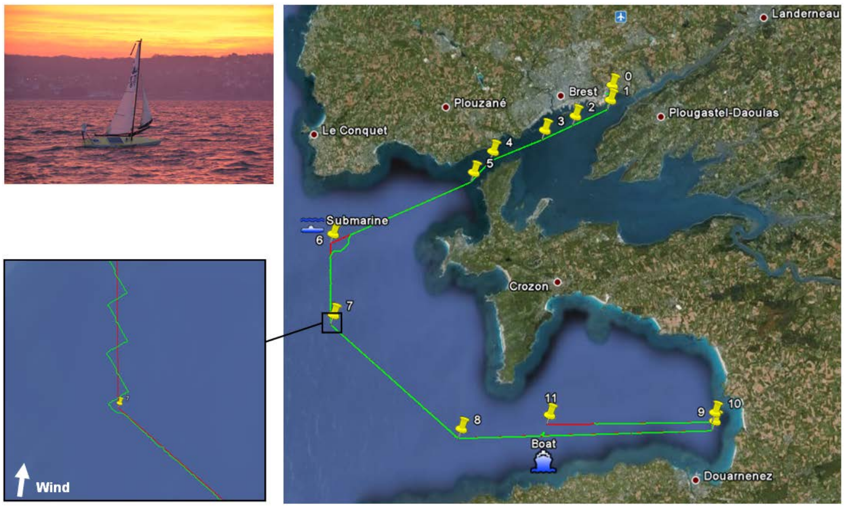


Fig. 7 Brest-Douarnenez. The sailboat needed to be deviated twice: first because of a submarine coming back to Brest naval base, then because of a static boat in the sailboat trajectory. During these perturbations, the autonomous program was not changed nor stopped, the sailboat was taken by our chase motorboat. Therefore, the submarine and boat deviations illustrate the robustness of the controller, which was able to continue the mission as if nothing happened.

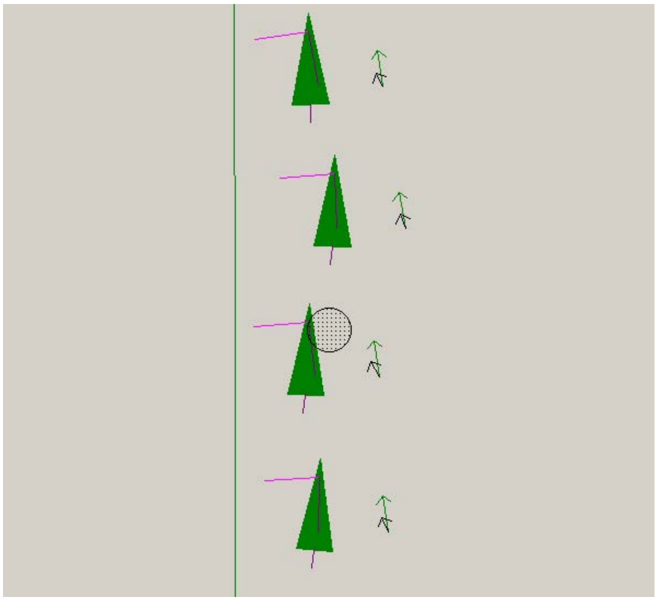


Fig. 8 Analysis of log files using a dashboard. The green vertical line is the line to follow, the small black arrow is the wind direction (measured by the weather station), the small green arrow is the desired heading, the big green arrow represents the sailboat (from GPS and compass data), with its rudder in purple and 2 estimations of its sail angle in pink (estimated from the inputs) and purple (measured by the integrated compass of the weather station, placed on top of the mast). As we see, these two estimations are in contradiction. These information are drawn at regular time intervals.

6 Conclusion

In this article, we showed that theoretical methods such as interval analysis can be used to theoretically validate robot control algorithms. However, in robotics we must use other validation methods such as HIL simulation and real tests to check and correct any hypotheses made. Different experiments were carried out with the *VAIMOS* robot for that purpose, while demonstrating the operational interests of an autonomous sailboat for oceanography.

Acknowledgements. VAIMOS has been built from 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 Bretagne). The authors would like to thank LPO and RDT to have enabled the use of VAIMOS as a test platform for algorithms from ENSTA Bretagne. They thank also everyone that has contributed to the project : Yves Auffret, Patrick Rousseaux, Sébastien Prigent, Loïc Dussud, Stéphane Barbot, Loïc Quemeneur, Bertrand Forest (RDT, IFREMER) and Olivier Ménage, Fabienne Gaillard, Thomas Gorgues, Thierry Terre (LPO, IFREMER).

References

1. Brière, Y.: The first microtransat challenge. ENSICA (2006), http://web.ensica.fr/microtransat
2. Bruder, R., Stender, B., Schlaefer, A.: Model Sailboats as a Testbed for Artificial Intelligence Methods. In: IRSC 2009 (2009)
3. Cruz, N.A., Alves, J.C.: Ocean sampling and surveillance using autonomous sailboats. In: IRSC 2008 (2008)
4. Elkaim, G.H., 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.H., Lee Boyce Jr., C.O.: An Energy Scavenging Autonomous Surface Vehicle for Littoral Surveillance. In: ION Global Navigation Satellite Systems Conference (2008)
6. Erckens, H., Bússer, G.A., Pradalier, C., Siegwart, R.Y.: Navigation Strategy and Trajectory Following Controller for an Autonomous Sailing Vessel. IEEE RAM 17, 47–54 (2010)
7. 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)
8. Jaulin, L.: Modélisation et commande d'un bateau à voile. In: CIFA2004 (Conférence Internationale Francophone d'Automatique), Douz (Tunisie). CDROM (2004)
9. Jaulin, L., Kieffer, M., Didrit, O., Walter, E.: Applied Interval Analysis, with Examples in Parameter and State Estimation. In: Robust Control and Robotics. Springer, London (2001)
10. Jaulin, L., Le Bars, F.: A simple controller for line following of sailboats. In: IRSC 2012, Cardiff, UK (2012)
11. 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)
12. Jaulin, L., Le Bars, F., Ménage, O.: An interval approach for stability analysis; Application to sailboat robotics. Submitted to IEEE Transaction on Robotics (2012)
13. Jaulin, L., Walter, E.: Set inversion via interval analysis for nonlinear bounded-error estimation. Automatica 29(4), 1053–1064 (1993)
14. Romero-Ramirez, M.A.: Contribution à la commande de voiliers robotisés. PhD dissertation, Université Pierre et Marie Curie, France (2012)
15. Rynne, P.F., von Ellenrieder, K.D.: Unmanned autonomous sailing: Current status and future role in sustained ocean observations. MTS Journal 43(1), 21–30 (2009)
16. Sauze, C., Neal, M.: An autonomous sailing robot for ocean observation. In: Proceedings of TAROS 2006, Guildford, UK, pp. 190–197 (2006)
17. Sliwka, J., Reilhac, P., Leloup, R., Crepier, P., de Malet, H., Sittaramane, P., Le Bars, F., Roncin, K., Aizier, B., Jaulin, L.: Autonomous robotic boat of ENSIETA. In: 2nd International Robotic Sailing Conference, Matosinhos, Portugal (2009)
18. Stelzer, R., Pröll, T.: Autonomous sailboat navigation for short course racing. Robot. Auton. Syst. 56(7), 604–614 (2008)
19. Xiao, K., Sliwka, J., Jaulin, L.: A wind-independent control strategy for autonomous sailboats based on voronoi diagram. In: CLAWAR 2011, Paris (2011) (best paper award)

Part III
Controller, Sensor and Rig Design

A Study on Potential Energy Savings by the Use of a Balanced Rig on a Robotic Sailing Boat

Roland Stelzer and Daniel Estarriola Dalmau

Abstract. The drive for the sail trim has been identified as one of the largest consumers on the autonomous sailing boat ASV Roboat. The presented work analyses the potential energy savings of a balanced rig compared to the conventional sloop rig, which is currently in use on this boat. Results of a computer simulation show that a balanced rig can save about two-thirds of the power needed for the sail trim.

1 Introduction

Recent events, such as the devastating tsunamis in Asia, the Deepwater Horizon oil spill in Gulf of Mexico, accidents involving refugee boats off the coast of Lampedusa in Italy, and pirate activities in the Gulf of Aden have clearly and impressively emphasized the importance of a fully integrated ocean observation system [11]. Robotic sailing boats represent a rapidly emerging technology for various tasks on lakes and oceans. They offer the possibility of sampling an area of interest with high temporal and spatial resolution at low cost [7].

The effect of a balanced rig on the power consumption of a robotic sailing boat has been investigated using the example of ASV Roboat [13], the current world champion in robotic sailing. The basis for the ASV Roboat is the commercially available boat type Laerling [8]. It has a length of 3.72 m and consists of a 60 kg keel-ballast, which will bring the boat upright even from the most severe heeling. Including batteries the overall weight of the boat is about 300 kg. It currently features a conventional sloop rig. The sail area of mainsail and foresail together is 5.4 m^2. The average power consumption of the ASV Roboat is approximately 35 W. This measurement was taken during a test sailing on the Baltic Sea at an average wind speed of 6.5 m/s.

Roland Stelzer · Daniel Estarriola Dalmau
INNOC - Austrian Society for Innovative Computer Sciences, Vienna, Austria
e-mail: roland.stelzer@innoc.at

The main power source of the ASV Roboat is a set of solar panels providing up to 285 W_p of power during conditions of full sun. 285 W_p corresponds to approximately 30 W of average output over a whole year under central European weather conditions [5, 12]. To cover the night period, electricity is stored in two deep-cycle batteries of 1.92 kWh together (80 Ah at 12 V each).

In order to be able to operate the boat at least for a limited period in conditions of reduced sunlight (night, bad weather) or if the solar power system breaks down, the boat is also equipped with a direct methanol fuel cell (EFOY Pro 1600). It delivers 65 W and features as a backup energy supply. The fuel tank contains 28 l of methanol. With a methanol consumption of 1.11 l/kWh as stated in the data sheet, the boat can operate about a month with the fuel cell as its only source of electricity.

The boat is currently showing a slightly negative energy balance. The drive for the sail trim has been identified to be one of the largest consumers in the entire system. Therefore, a balanced rig design as a replacement for the current sloop rig on ASV Roboat has been analysed with regard to potential energy savings. Table 1 gives an overview of the power consumption of the current version of ASV Roboat. It can be seen that the sail gear is the biggest consumer beside the main computer on board.

Table 1 Power consumption of ASV Roboat (September 2011)

Component	Model	Power	Pct.
Embedded PC	VIA EPIA-MIII	15.20 W	44.0 %
Power supply	ATX	0.31 W	0.9 %
GPS receiver	Maretron GPS100	1.80 W	5.2 %
Compass	Maretron SSC200	1.62 W	4.7 %
Depth, speed, water temperature sensor	Maretron DST100	0.58 W	1.7 %
NMEA2000 protocol converter	Maretron USB100	1.82 W	5.3 %
Weather station	Airmar PB200	2.64 W	7.6 %
WiFi access point	Buffalo WHR-HP-G54	3.4 W	9.8 %
3G modem	HUAWEI E220	2.5 W	7.2 %
Satellite modem	Iridium SBD 9601 (switched on 2 min/h)	0.06 W	0.2 %
Sail gear	self-construction (non-balanced sloop rig)	4.58 W	13.3 %
Rudder gear	self-construction (balanced rudder)	0.02 W	0.1 %
Overall sum	**34.53 W**		

2 Research Hypothesis

The general aim of the presented work is to evaluate the effect of a balanced rig on power consumption and sailing performance. The aim of the simulations described below is to validate the following hypotheses:

1. A balanced rig provides great potential to save power. Based on our own experience with the balanced rudder system of the ASV Roboat, we anticipate a potential saving of at least 50 % from a balanced rig design.
2. The sailing performance is not negatively affected by the use of a balanced rig instead of a sloop rig.

3 Methodology

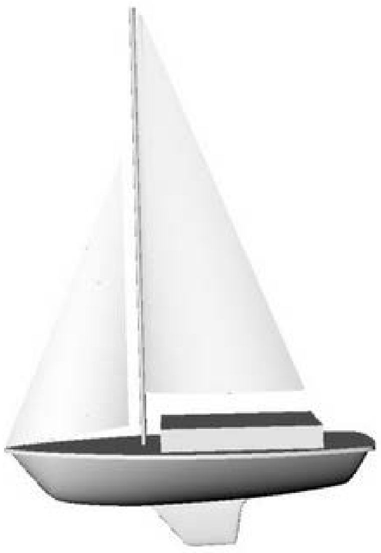
On a conventional sloop rig, which is the most common rig type on sailing vessels, relatively high power is needed to tighten the sails against wind force. As being self-sufficient in terms of energy is one of the major goals in robotic sailing, the rig design has become the focus of attention. A balanced rig design (also known as Balestron rig, Aerorig^{TM}, swing rig, and EasyRig^{TM}) offers great potential for saving power [1, 9]. A balanced rig consists of an unstayed mast carrying a main and jib (see Fig. 1(b)). The main boom extends forward of the mast (the mast passes through the boom) to the tack of the jib. The main and jib are sized in such a way that the force from the mainsail is slightly stronger than that from the jib. That is, the combined centre of effort is just behind the mast.

Balanced rigs have been used on the autonomous sailing boats *Avalon* [6], *VAIMOS* [4] and *IBoat* [2]. Furthermore, most of the rigid wing sails used on robotic sailing boats can be considered to be balanced rigs [3, 10].

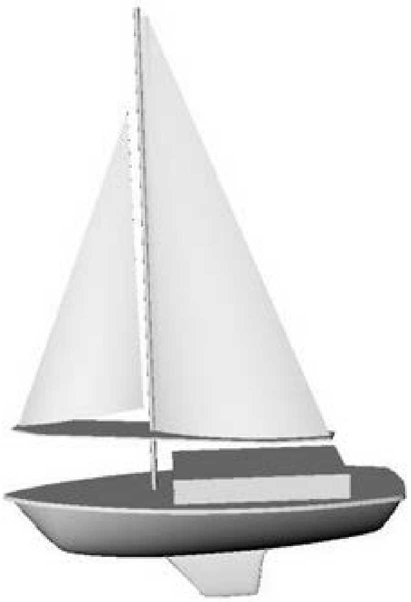
We have simulated several points of sail with the corresponding sail positions for a wind speed of 6 m/s. Figs. 1(a) and 1(b) show the two models which have been compared. Mast height, sail area and sail shape have not been modified.

For this study we used the software Star CCM+ for the simulation process, Icem Ansys for the preprocessor and Rhinoceros for the design. Firstly, the existing geometry (which includes many extraneous details) has been modified in Rhinoceros to leave only the elements which affect the simulation - mainly the upper side of the sailboat.

Both sails have been divided in ten vertical stripes parallel to the axis of rotation. The forces on each of the stripes are used to calculate the overall torque for each of the two sails. Table 2 shows the parameters used in the simulation process.



(a)



(b)

Fig. 1 Analysed rig designs: (a) original sloop rig and (b) modified balanced rig

Table 2 Simulation parameters

Space	three dimensions
Movement	stationary
Motion	stationary
Material	gas
Flow	segregated flow
Equation of state	ideal gas
Time	steady
Viscous regime	turbulent
Reynolds average turbulence	k-epsilon turbulenc

4 Results and Conclusions

Simulations have been carried out for several true wind angles from 30 deg to 180 deg. The torques caused by the wind forces acting on the sails have been analysed and have shown a mean energy saving of 68 %. Fig. 2 shows the energy savings for the simulated points of sail.

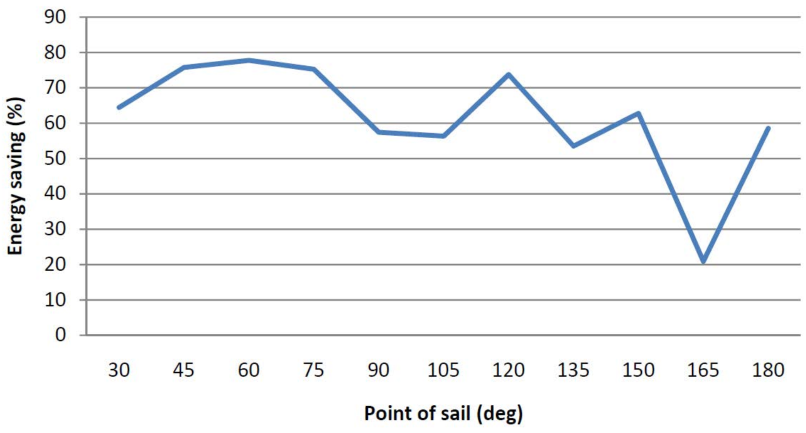
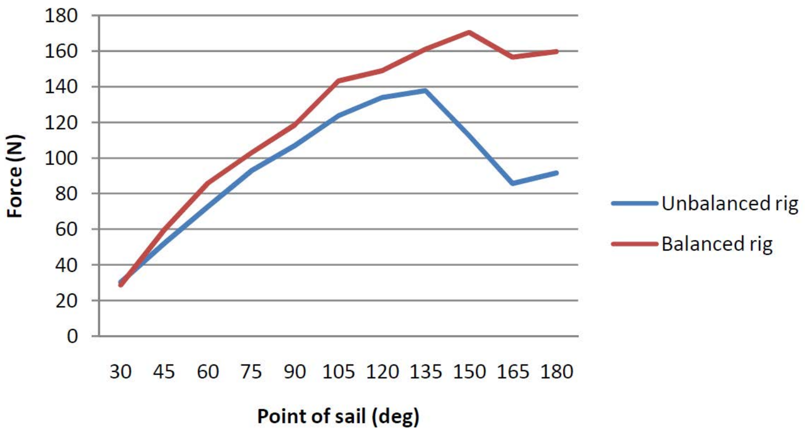
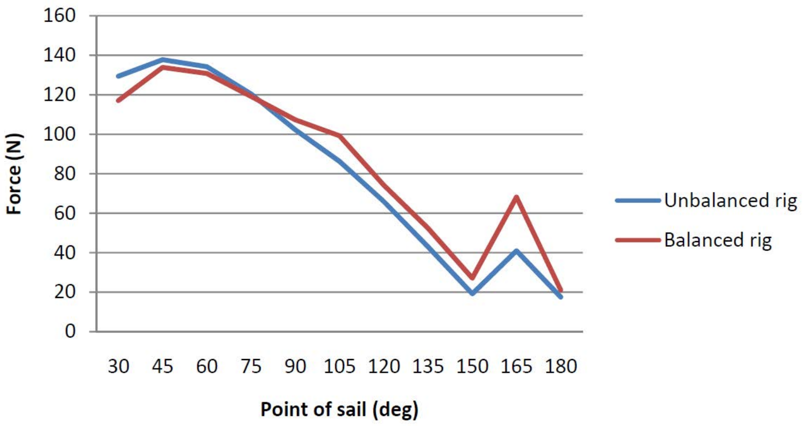


Fig. 2 Energy savings (reduction of the sail torque) for the simulated points of sail: 0 in irons, 180 running downwind

Furthermore, the forward and lateral forces on the boat have been simulated (see Fig. 3). The results show that the forward force is similar on close hauled courses, but is significantly higher for the balanced rig when running downwind. The lateral force, which leads to lateral drift, is not significantly affected by the change of the rig.



(a)



(b)

Fig. 3 Results of performance analysis: (a) forward force and (b) lateral force

According to the simulation a balanced rig provides the theoretical potential of saving approximately 68 % of power on the sail drive without negative implications on the sailing performance. This value reflects only the expected reduction of the sail torque and therefore assumes a degree of efficiency in the sail drive of 100 %. Therefore, the percentage of power saving in practice might be slightly smaller as there is a certain fixed amount of power consumption for the sail gear, regardless of the torque on the sails. However, the simulation results are promising and useful as a first approximation. It is planned to validate and further investigate the influence of a balanced rig design on power consumption in real world experiments on the ASV Roboat.

References

1. BalancedRig: BalancedRig website (2009), http://balancedrig.com/description.html (accessed April 16, 2009)
2. Briere, Y.: Iboat: An autonomous robot for long-term offshore operation. In: The 14th IEEE Mediterranean Electrotechnical Conference, MELECON 2008, pp. 323–329. IEEE (2008)
3. Elkaim, G.: System identification for precision control of a wingsailed GPS-guided catamaran. Ph.D. thesis, Standford University (2002)
4. Bretagne, E.: Autonomous sailing boat VAIMOS (2011), http://www.ensta-bretagne.fr/en/index.php/actualite/autonomous-sailing-boat-vaimos/ (accessed: July 11, 2012)
5. Finanztest: Geld verdienen auf dem Dach. Finanztest 7, 35 (2006), http://www.test.de/themen/bauen-finanzieren/meldung/-Solarstrom/1386918/1386918/1391398/
6. Giger, L., Wismer, S., Boehl, S., Büsser, G., Erckens, H., Weber, J., Moser, P., Schwizer, P., Pradalier, C., Siegwart, R.: Design and construction of the autonomous sailing vessel Avalon. In: Proc. 2nd Int. Robotic Sailing Conf., pp. 17–22 (2009)
7. Klinck, H., Stelzer, K., Jafarmadar, K., Mellinger, D.: Aas endurance: An autonomous acoustic sailboat for marine mammal research. In: Proceedings of International Robotic Sailing Conference, Matosinhos, Portugal, pp. 43–48 (2009)
8. Laerling: Laerling website (2012), http://www.laerling.nl (accessed: July 15, 2012)
9. Multirig Website (2009), http://www.multirig.com/the_balestron_rig.htm (accessed: April 16, 2009)
10. Neal, M., Sauze, C., Thomas, B., Alves, J.: Technologies for Autonomous Sailing: Wings and Wind Sensors. In: Proceedings of the 2nd IRSC, Matosinhos, Portugal, July 6-12, pp. 23–30 (2009)
11. Rynne, P., von Ellenrieder, K.: Unmanned autonomous sailing: Current status and future role in sustained ocean observations. Marine Technology Society Journal 43(1), 21–30 (2009)
12. Schrag, H.: Jahreserträge aus Photovoltaik-Anlagen unserer Kunden (2011), http://www.schrag-sonnenstrom.de/9.html (accessed: September 28, 2011)
13. Stelzer, R.: Autonomous sailboat navigation. Ph.D. thesis, De Montfort University, Leicester, UK (2012)

A Discrete-Component 2D-Array Wind Sensor without Moving Parts for a Robotic Sailboat

Taylor Barton and Mariano Alvira

Abstract. Most common wind sensor designs are based on moving wind vanes and are potentially susceptible to mechanical failure. This paper explores an alternative approach without moving parts for robotic sailboat applications. The new wind sensor operates on a thermal principle and is constructed of a 2D-array of discrete, surface-mount components. This paper describes the experimental work on a prototype version of this sensor. Although not yet integrated into a robotic sailboat system, the sensor is a potentially interesting alternative to a mechanical transducer.

1 Introduction

Electronic wind sensors are an essential element in the majority of robotic sailboat designs, although sensorless designs have been proposed [4]. A measurement of apparent wind direction is generally used as a control input to determine the optimal sail trim, and for path planning for upwind sailing. For small (under 2 m) robotic sailboats designed to operate as e.g. autonomous sensing vehicles, the wind sensor design must be both robust for long-term voyages and small/lightweight enough to be placed at the top of the mast. Furthermore, an inexpensive solution is preferred.

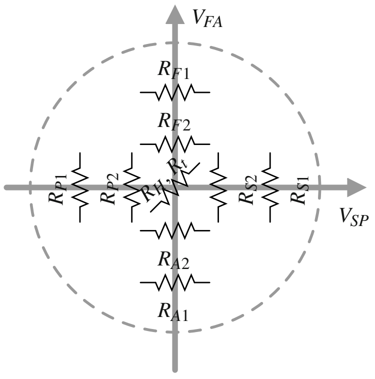
Typical wind sensors for robotic sailboats are based on a moving vane whose angle is converted to an electric signal through a transducer. Potentiometer-based wind sensors are simple to construct but the contact between the wiper and fixed conductor of the potentiometer may wear over time. A contactless solution, for example based on a magnetic encoder such as the popular Austria Microsystems AS5030, is mechanically preferable to a potentiometer, but requires bearings and a more substantial structure [2]. Furthermore, it is good design practice to limit the number

Taylor Barton
Seascope, Everett, MA USA
e-mail: tbarton@seascopetech.com

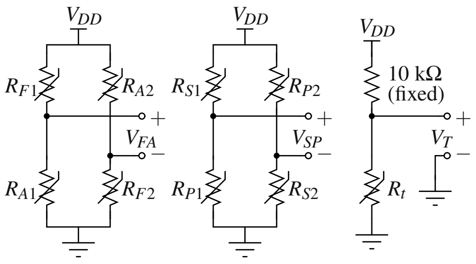
Mariano Alvira
Seasacope, Everett, MA USA
e-mail: mar@seascopetech.com

of moving parts on a sailboat, particularly one intended to operate for long ocean voyages. One option is an ultrasonic wind sensor, although these are sensitive to the temperature and humidity of the air.

This work explores an alternative approach based on thermal conduction as a possible wind sensor for a robotic sailboat. The approach is based on a two-dimensional (2D) array of thermistors operating on a thermal principle. A variety of integrated versions have been demonstrated in CMOS [5]. The sensor presented in this work takes advantage of the availability of small surface-mount components to use the concepts and structure of the integrated devices in [3], reproduced in Fig. 1, in a discrete-component version. It has no moving parts and a sensing element measuring 9 mm $\times$ 9 mm. The prototype sensor consumes between 100-500 mW depending on operating mode, and is robust to changes in the sailboat heeling angle. It has a relatively slow response compared to other sensor types due to the thermal time constant, which results in a low-pass or averaging behavior. However, the many advantages of the approach, including its mechanical robustness, small size and weight, make it a good candidate for robotic sailboat control.



(a) Physical arrangement



(b) Circuit schematic

Fig. 1 Wind sensor schematic and sketch showing approximate layout. The effect of the wind is decomposed into fore-aft and starboard-port vectors. An additional thermistor R_t is included to measure the temperature of the heating resistor R_H, and is directly mounted to R_H.

2 Theory of Operation

Thermal 2D-array wind sensors are based on the relation between the flow of the fluid over the sensor and the heat transfer from a central heating element to sensing elements arranged as in Fig. 1. The sensor consists of two full-bridge thermistor connections arranged in quadrature. When wind flows over the sensor, the upstream and downstream elements are heated differentially, resulting in a voltage across the bridge that corresponds to both the angle of incidence of the wind as well as its strength [3]. For the robotic sailboat application, the two bridges are set up to correspond to the fore-aft and starboard-port axes. An additional thermistor R_t is included to monitor the temperature of the heating resistor, and is mounted directly to R_H.

3 Implementation

The sensor is constructed on a four-layer printed circuit board (PCB) using 10 kΩ surface mount thermistors in an 0402 package (Murata NCP15XH103F03RC) and a 39 Ω resistor R_H in a 1210 package. The total component cost per sensor, excluding PCB cost, is \$2.35 (Qty. 1).

Four versions were built (see Fig. 2), with combinations of heating resistor and thermistor components on the same or opposite sides, and placing the heater resistor either in alignment with or at a 45 degree angle to the thermistor bridges. Although the thermistors are placed farther from the center heater in the angled-resistor design, this version was included because it reduces the asymmetry caused by the rectangular resistor shape. In measurements no significant difference between the rotated and aligned configurations was found. Prototypes were also tested having the heating resistor on the opposite side from the components, but this configuration put the components directly in the wind, creating turbulent flow. The version in Fig. 2(a) is used for all measurements in this work. For prototyping purposes, all five voltages (two from each bridge plus the temperature measurement) are buffered and sampled by an analog to digital converter (ADC) on a microcontroller.

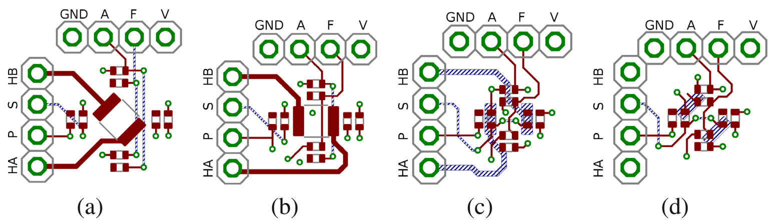


Fig. 2 The four prototype PCB layouts, top (*blue, hashed*) and bottom (*red, solid*) copper layers only. Not shown are the inner two copper layers used to route power and ground. The version with all same-side components and angled resistor in (a) is used for all measurements in Sect. 4.

3.1 Optional Additional Circuitry

Figure 3 shows additional analog circuitry that can be included in the sensor system. The heating resistor R_H can be enabled by the transistor operating as a switch to turn off the wind sensor when not needed; alternatively, the transistor could be used to control the amount of current applied to R_H. Shown also is a scheme where bridge voltages V_{FA} and V_{SP} are amplified using an instrumentation amplifier as in [3], then A/D converted by the microcontroller. This instrumentation amplifier circuit was implemented for testing. However, it was found that the raw sensor signal was at least 50 ADC codes in amplitude even for the lowest-power setting (100 mW) and the lowest wind speed (1.7 m/s). The measurement noise amplitude under these low-power, low-wind conditions was on the order of 30 degrees and therefore marginally within the tolerance required for basic robotic sailboat operation. All of the measurements in this paper are from the sensor directly without amplification and with the heating resistor connected directly across the supplies.

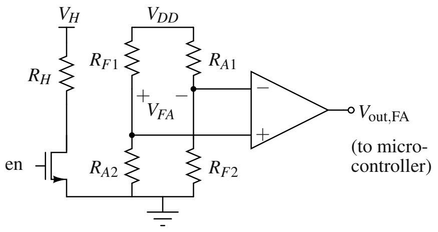


Fig. 3 Optional analog circuitry, including an enable switch, and an instrumentation amplifier that can be included to buffer and amplify each bridge voltage [3].

4 Measured Performance

The prototype wind sensor was implemented on 1.6 mm thick FR4 material PCB with pin headers used for signal and power as shown in the photograph in Fig. 4. This section describes the results of our characterization in a simple wind tunnel constructed from a three-speed fan and using boxes of straight drinking straws to create uniform flow. A "smoke stick" and Kestral wind sensor were used to verify the flow and calibrate the wind speed measurement.

4.1 Static Measurements

For these measurements, the sensor is mounted on the shaft of a two-phase potentiometer. The sensor is rotated to each position, the measurement allowed to settle, and then the thermistor bridge voltages and the pot voltages are recorded. From these measurements, the actual and measured angle of incidence for the wind can be derived. The wind sensor angle measurements were characterized under two operating modes: 500 mW heating power, corresponding to the maximum rating of

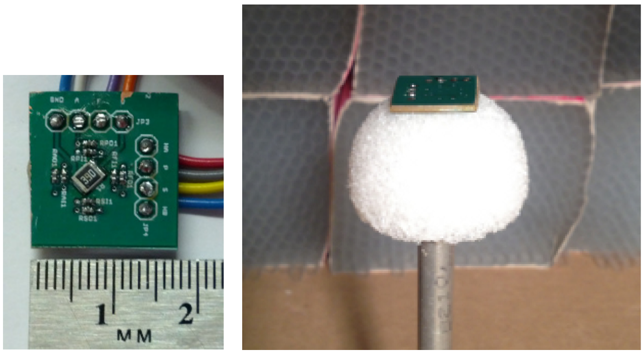

(a) PCB component side. (b) Mounted and in the wind tunnel.

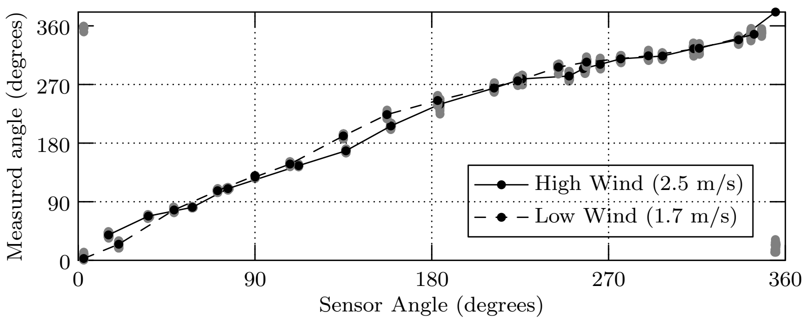
Fig. 4 Photograph of wind sensor (a) PCB component side (thermistor mounted to central heating resistor is omitted), and (b) mounted to spherical base with component side down. This prototype board has the heating resistor and sensing components only.

the heating resistor, and 100 mW heating power, corresponding to the power required by the AS5030, a commonly used component for wind sensing. The resulting transfer characteristics are shown in Fig. 5 for two wind speeds. The grey points in Fig. 5 represent the raw measurement data; the black points are averaged data. The higher-power measurement has lower noise, as expected, with the worst-case output variation in sensor output for a given wind angle approximately 30 degrees including variation with wind speed. If the wind speed is known (see Sect. 4.2), the worst-case output variation is only 15 degrees, and typical variation is on the order of 5 degrees, sufficient for the robotic sailboat control algorithm used by the authors. The nonlinearity in the static transfer function can be corrected for using a digital look-up table.

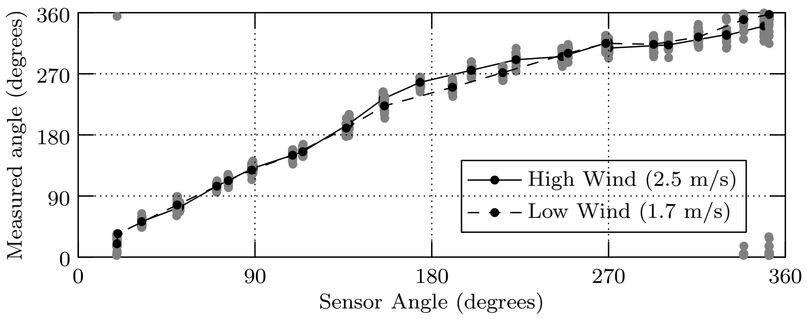
Figure 6 shows the incremental changes in the thermistor bridge voltages over the 360° measurement sweep from Fig. 5. Some asymmetry and nonlinearity can be seen these nominally-sinusoidal curves. The total bridge voltages V_{FA} and V_{SP} include a DC operating point that is not shown in this plot. The operating points of the two bridges are not equal, and depend on the temperature of the PCB. This offset is due to mismatch in the thermistor bridge elements. In order to generate the plots in Fig. 5, the offsets are calculated by averaging over the entire measurement and removed.

4.2 Offset Variation with Temperature

Although in theory the differential voltages at the bridge outputs will be zero in zero wind, mismatch in the bridge elements creates DC thermistor bridge offset voltages that must be removed before computing the wind angle. These voltages are temperature-dependant and require characterization over a range of temperatures



(a) High-power (500 mW) operation



(b) Low-power (100 mW) operation

Fig. 5 Wind sensor angle measurements for high and low power, and two wind speeds: raw data (*grey*), and averaged data (*black*). These measurements were made without any amplification of the bridge voltages.

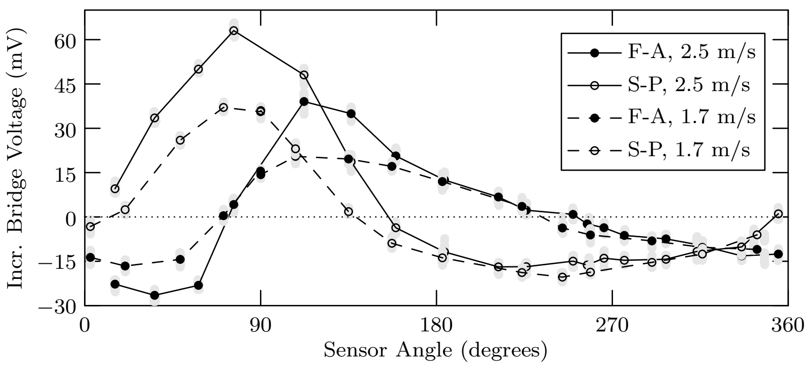


Fig. 6 Fore-aft (F-A) and starboard-port (S-P) incremental bridge voltages for high and low wind speeds. The resulting measured wind angles for the two wind speeds are nearly identical (see Fig. 5).

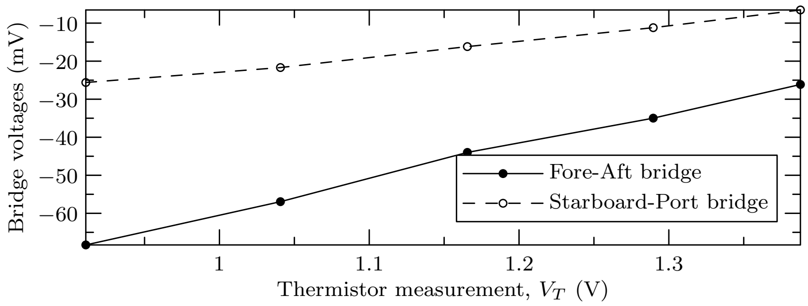


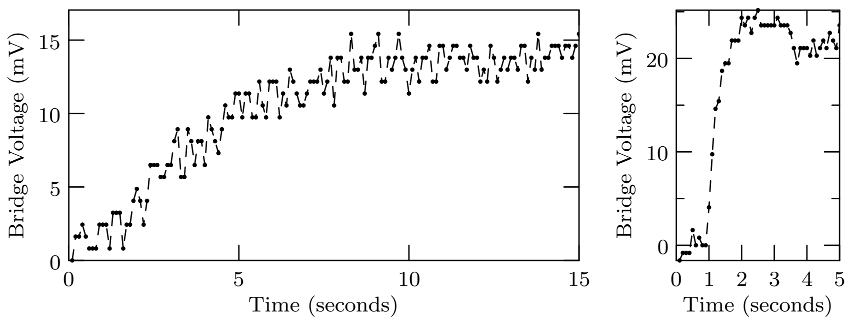
Fig. 7 Variation in bridge offset voltages over different temperatures. These measurements were made without wind.

for correction. The variation in bridge voltages over different temperatures, shown in Fig. 7, is characterized using the thermistor R_t mounted to the heating resistor. Although not verified with real-world sailboat operation, it should be possible to use this information to correct for the offset variation by continuously monitoring the sensor temperature and using the appropriate offset in the wind angle calculation. A similar look-up table based temperature offset cancellation method is used in [3].

The thermistor monitoring the heating resistor temperature can also be used as the sensor in a feedback loop driving resistor R_H to a constant temperature. Either this constant-temperature approach or the constant-power method used in this work can provide a measure of wind speed in addition to direction. In the constant-power method, the resistor temperature can be compared to that of an identical heater/thermistor pair not exposed to the wind. For the constant-temperature approach, the amount of power needed to maintain the heater resistor temperature is related to wind speed [3].

4.3 Sensor Bandwidth

The sensor bandwidth was measured by observing the transient response to an abrupt step in angle. As shown in Fig. 8(a), the 10-90% bridge voltage settling time in response to a step in angle is 6 s, corresponding to a low-pass filter with a pole at $f_p = 0.06$ Hz. Although the desired operating mechanism for this sensor is convection, i.e. the asymmetric transfer of heat from the central resistor to the thermistor bridges by the wind, the slow response is dominated by the time constant associated with heat *conduction* through the PCB. If the conduction path is removed by turning over the PCB so that the components are on top, the rise time is improved by an order of magnitude. The low-pass pole in this components-on-top configuration is located at 0.5 Hz. Despite its faster response time, the components-on-top configuration was ruled out because the turbulence caused by the components in the air stream dramatically increases the measurement noise.



(a) Components mounted on underside of PCB. (b) Components on top.

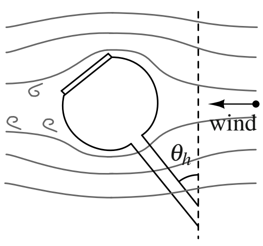
Fig. 8 Transient response of a thermistor bridge to a step in sensor angle. In (a), the components are mounted on the underside of the board, and the time constant is dominated by heat conduction through the PCB. The significantly faster response in (b) is a result of inverting the sensor PCB so that the components are on top.

If the component side of the PCB were filled in using a potting compound or other filler, it could be given a smooth surface with only a thin coating over the components. Compared to the insulating nature of the PCB fiberglass material, it would not be difficult to reduce the thermal conduction time constant significantly compared to that of the existing sensor. Although this method has not yet been verified, it should be possible to make a version of the sensor that has the combined performance of the angle accuracy of Fig. 5 and the speed of response in Fig. 8(b). It is important to note that water on the sensor surface will also affect the thermal properties of the sensor, and that the sensor should be shielded from rain.

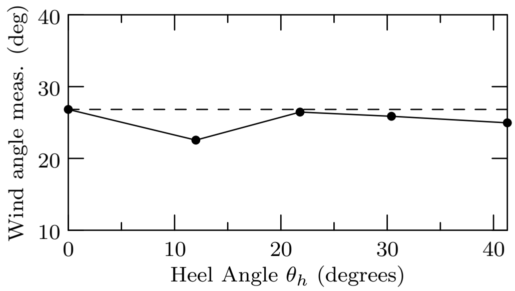
4.4 Heeling Angle

One challenge specific to wind sensor design for sailboat applications is operation of the wind sensor under various heeling angle conditions. The sensor must continue to provide accurate wind measurements even when rotated about the fore-aft axis. In the case of mechanical wind sensors, this means that the sensor rotor must be well-balanced so that the changing gravitational vector does not affect the sensor measurement. Here, it requires some consideration of the wind-flow over the face of the sensor. As can be seen in Figs. 4, the wind sensor was mounted on a hemi-spherical styrofoam piece. This shape was chosen so that the incident wind will flow over the face of the sensor even when the boat is heeling. The sketch in Fig. 9(a) shows conceptually how streamlines might flow over the sensor, including friction effects. The maximum heeling angle θ_h for which the sensor provides a useful measurement of wind direction will be limited by the friction effects [1].

The sailboat will experience the largest heeling force when sailing on a close-hauled point of sail, and furthermore will only heel to leeward (i.e. away from the wind). To approximate these conditions, then, the sensor was measured under



(a) Conceptual sketch



(b) Wind angle measurement

Fig. 9 The sensor is mounted to a spherical base so that wind will flow over the sensor face when the boat is heeled (a). In (b), measured wind angle is plotted when sensor angle is held constant and heel angle θ_h is varied.

various heeling angles to leeward for a heading pointing 35 degrees from the wind direction. The sensor was tilted to varying angles and the measured wind angle was recorded after the sensor output had settled. As shown in Fig. 9(b), the variation in the measured wind angle over a range of heeling angles between 0° and 43° is less than 4°.

In practice, it may be beneficial to characterize the effect of heel angle under a greater variety of sailing conditions. If the variation in measurement due to heel angle is found to be too great for a particular sailing control scheme, it could be characterized and compensated for using a measurement of the heel angle. Because an accelerometer is commonly included on robotic sailboats as part of a tilt-compensated compass, a measurement of heel angle is available if required.

5 Conclusions and Future Work

The thermal wind sensor was developed as an alternative to a vane-based mechanical design, with the objective of removing the potential for mechanical failure. The resulting wind sensor is lightweight, small, and inexpensive. At the same time it requires significant signal processing, particularly to correct for temperature-dependant offsets, in order to be integrated into a robotic sailboat system. The wind sensor can be operated at power levels comparable to those used in AS5030-based designs, but requires somewhat higher power levels (500 mW) for better noise performance. This design therefore represents one possible choice among tradeoffs for wind sensors for robotic sailboats, emphasizing mechanical robustness and size over power consumption and computational complexity. Some further development is required, particularly experimentation with coating the sensor components to get an improved response time, and most importantly implementation and testing as part of a robotic sailboat.

References

1. Marchaj, C.: Aero-hydrodynamics of sailing. International Marine Publishing, Camden (1988)
2. Neal, M., Sauzé, C., Thomas, B., Alves, J.: Technologies for Autonomous Sailing: Wings and Wind Sensors. In: Proceedings of the 2nd International Robotic Sailing Conference, Matosinhos, Portugal, pp. 23–30 (2009)
3. Shen, G., Qin, M., Done, Z., Huang, Q.: An intelligent wind sensor system with auto-zero function. In: Proceedings of the International Solid-State Sensors, Actuators and Microsystems Conference, pp. 256–259 (June 2009)
4. Sliwka, J., Nicola, J., Coquelin, R., de Megille, F.B., Clement, B., Jaulin, L.: Sailing without Wind Sensor and Other Hardware and Software Innovations. In: Schlaefer, A., Blaurock, O. (eds.) Robotic Sailing, vol. 79, pp. 27–38. Springer, Heidelberg (2011)
5. van Oudheusden, B.: Silicon flow sensors. In: IEE Proceedings D - Control Theory and Applications, pp. 373–380 (September 1988)

Small and Inexpensive Single-Board Computer for Autonomous Sailboat Control

Mariano Alvira and Taylor Barton

Abstract. This work presents a computer board designed for autonomous robotic sailboat control. Taking advantage of the current availability of feature-rich processors such as the LPC3130 from NXP and the MC13224v from Freescale Semiconductor used in this work, our design emphasizes low cost and power consumption, as well as small size. At the same time, the system is not excessively specialized; it runs 32-bit Linux and has network capability via Ethernet, WiFi, cellular or Bluetooth USB sticks. The computing system presented in this work is applicable to a variety of robotic sailboat applications, including making a 0.5 m Graupner Micro Magic fully autonomous without relying on a shore-side base station for computation.

1 Introduction

For true autonomy, a robotic sailboat must do all course planning and control onboard the vessel, without relying on an on-shore base station for computation. For very small robotic sailboats, it has previously been difficult to incorporate sufficient computing power on-board given the size of the craft [13]. Class rules for the 0.5 m robotic racing Micro Magic (rrMM) Class at the World Robotic Sailing Championship (WRSC) 2011 and WRSC 2012 allow on-shore computation as a result of this constraint. Boats in the Sailbot and Microtransat classes at WRSC generally contain a more complete computing system to meet the requirement of full autonomy. These boats are typically larger and custom-designed vessels and thus have more relaxed form factor, cost, and power constraints compared to the rrMM.

For robotic sailing competitions like WRSC and Microtransat [8], teams commonly use commercial off-the-shelf (COTS) available electronics for both control hardware and sensors, for example to ease transitions between student groups

Mariano Alvira · Taylor Barton
Seascope, Everett, MA USA
e-mail: {mar,tbarton}@seascopetech.com

[9, 14]. Alternatively, evaluation kits or otherwise pre-existing hardware for a selected microcontroller are used [6, 13]. All of these systems contain to some degree a combination of COTS hardware mixed with custom elements, but the choice of that boundary is varied. While COTS hardware offers many advantages to the development process it has two main disadvantages: it often has either excessive or inadequate features, and typically has a form factor that is not exactly suitable.

For small boats, the disadvantages of COTS circuit boards become very pronounced. By designing a custom PCB for our computing system we were able to choose a main CPU that did not have an existing suitable COTS system but otherwise is very well suited for the application due to its low power, cost, and required board area. Designing a custom control board also allowed us to mitigate form factor concerns with the rrMM by using connectors and layout well suited for the vessel.

This paper presents a control board with enough computing resources for full autonomy suitable for Sailbot and even Microtransat class boats while meeting the tight power, size, and cost budgets required for the rrMM class. So that others can treat the board we have designed here as a COTS component, we have released the hardware design files and software used under suitable Open and/or Free Licences.

2 Hardware Overview

While the primary design goals of the system are cost, power, volume, and computing performance, a secondary consideration is the run-time operating system and choice of computing core. Autonomous sailing competitions often require a significant number of high-level functions such as route-planning, mission/game logic, and data logging. Although standard C on a microcontroller is appropriate for processing sensor samples and digital control, it quickly becomes cumbersome for these more general-purpose computational tasks. An operating system like Linux makes many tasks much easier, such as interfacing with mass storage SD cards, dealing with hardware drivers (e.g. for USB peripherals), and full-featured networking. These features typically require a "Distribution based-OS" (or "full OS") [1] such as any of the numerous OS distributions that use the Linux kernel (e.g. Debian, Arch, Ubuntu, Android). On the other hand, real-time operations such as sensor and control loops, are best done with a dedicated microcontroller running some kind of "real-time operating system" (RTOS). With this in mind, the hardware system is designed as a hybrid "full OS" + "RTOS" system. The system is made up of a baseboard with a main CPU (LPC3130) running Linux, as well as a general-purpose M12 MC13224v module from Redwire [10] running Contiki and serving as a real-time coprocessor. The system block diagram is shown in Fig. 1.

[1] The terms RTOS and OS have varied meaning and scope. For the purposes of this paper, we are calling a "full OS" anything that runs on CPUs with managed memory and provides a rich set of high level programs: i.e. a full Linux distribution. We are calling an "RTOS" any code that runs on a microcontroller or CPU incapable of running a "full OS" and whose purpose includes strict timing requirements.

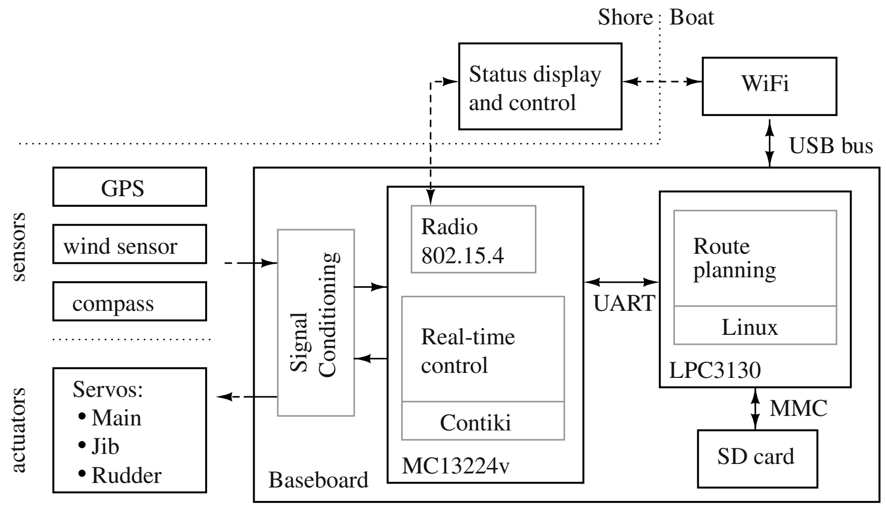


Fig. 1 Generalized system block diagram.

The baseboard is custom-designed for the robotic sailboat application. It contains all system-specific electronics as well as a full Linux system (running Arch Linux, although Debian is also possible) with the LPC3130 200MHz ARM9 with 32MB of RAM. Networking capability is achieved via a USB stick, an option which provides a great deal of flexibility and simplicity, and is also low cost (USB ethernet sticks can be obtained for approximately the same cost as the chips they contain). Wifi, Bluetooth, and cellular modems USB sticks can also be used. As they are swappable, ethernet can be used on shore for development and a wireless option can be used at runtime. Mass storage is accomplished with the SD card interface on the LPC3130. 2GB micro SD cards are used as they are inexpensive and offer plenty of disk storage for the system. The system boots from the SD card and all system files are contained there.

The power subsystem, not shown in the block diagram for clarity, also resides on the baseboard. Terminal pin headers receive wiring from the sailboat system, such as from sensors, actuators, and the power system which is located on the baseboard.

For the processor running the "RTOS" functions, we are using an MC13224v in the form of the M12 module and the Contiki Operating System [3]. The M12 is a general-purpose module that has been designed to be the smallest and easiest way to integrate an MC13224v into a design. The MC13224v is well supported in Contiki, which is a mature open-source "RTOS" written in standard C with a strong emphasis on networking. All real-time control and data processing of the sailboat is done using Contiki's "Protothreads" [4]. The LPC3130 and MC13224v communicate over serial line internet protocol (SLIP) with the UART on the LPC3130 connected to the first UART on the MC13224v. Then the MC13224v simply looks like another computer on the LPC3130s IP network (and vice versa). This greatly simplifies the development system as all outward APIs can use standard IP based methods to interact.

Unlike many microcontrollers, the MC13224v also includes a wireless 802.15.4 radio (the same physical layer as Xbee or any Zigbee system). This gives an additional communication option that can be used as the primary communication or in tandem with a wireless option on the LPC3130. A system very similar to this with the M12 module alone and computation done on an on-shore computer running Linux was used by the authors at WRSC 2011 to control the "Friend" Micro Magic class sailboat.

With the system hardware in place, autonomous boat operation is achieved by running Python scripts in Linux. This allows for very quick prototyping and development of the boat's behaviors without impeding crucial real-time control operations. The Wifi connection provides a normal networking interface so the boat can be accessed through tools like SSH or the web page it serves. Logging is easily done by writing to a file on the SD card. See Section 4 for more details regarding the software system.

A breakdown of the power consumed by the subsystems of the board is shown in Table 1 below.

Table 1 Power consumption broken out by subsystem.

Subsystem	Power
LPC3130 + RAM:	165 mW
LEDs (3x):	51 mW
Power supply:	88 mW
Wifi:	844 mW
MC13224v:	17 mW
802.15.4 TX:	
@ 4 dBm:	100 mW
@ 20 dBm:	500 mW
Total without comm.	321 mW
+ 802.15.4 @ 4 dBm	421 mW
+ wifi	1165 mW
+ all comm.	1986 mW

3 Implementation Details

The Bill of Materials and approximate cost of the board is included below in Table 2. Photographs of the 79 mm $\times$ 108 mm board in Figs. 2 and 3 show the relative placement of the components.

Three LED indicators are present on the top of the board. A red LED indicates when the system is powered. Each processor has one green LED that it can control.

Also on the top side of the board and shown in detail in Fig. 4 is a debug connector that can receive a Tag-Connect cable [15]. This connector gives access to the UART

Table 2 Bill of Materials for the complete single-board computer.

Part No.	Description	Cost (USD)
LPC3130	ARM9 200MHz CPU	3.78
AS4C16M16S	32MB SDRAM	1.75
AMP-1010058159	micro SD card connector	1.05
	2GB micro SD card	2.79
TPS54140 (×2)	switching power supply	4.74
UE27AC	USB connector	0.25
	Wifi stick	10.00
M12	MC13224v module	12.00
OPA4330 (×2)	Operational amplifiers	3.96
	Ancillary components	6.00
	PCB	3.20
	Assembly	8.98
	Total cost	58.50

from the LPC3130 as well as reset lines for each CPU. This port can be used to work with the serial debug console from the OS running on the LPC3130. A multiplexer typically connects this UART to the MC13224v when the Tag-Connect cable is not present.

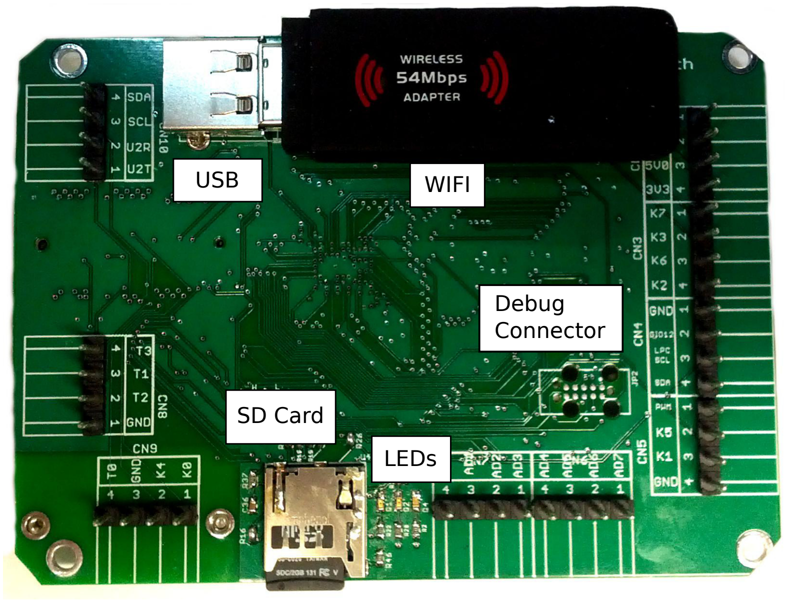


Fig. 2 Top of computer board. The board measures 79 mm × 108 mm.

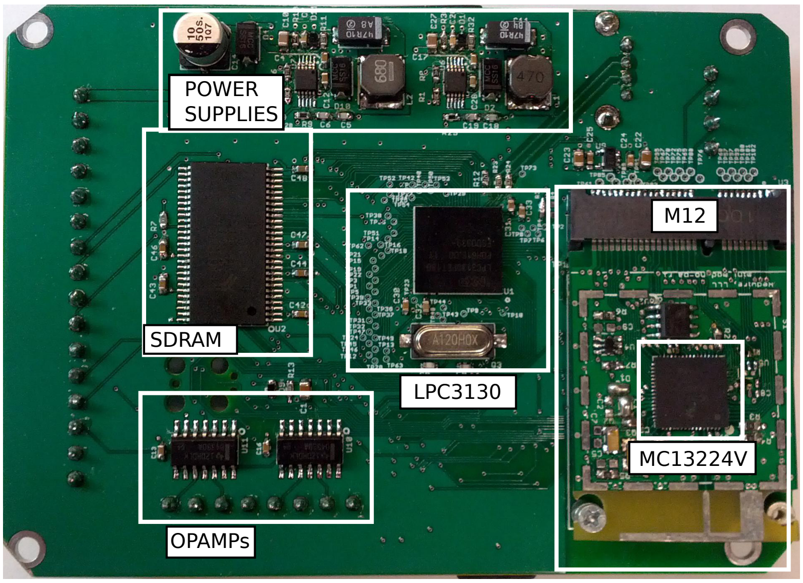


Fig. 3 Bottom of computer board.

Eight chopper stabilized and low-offset operational amplifiers are used to buffer analog input signals going to the MC13224v.

Two switch mode power converters are present on the board to generate 5.0V and 3.3V necessary for the on board components. The output of these regulators is also available on the external pin connections to be used by the system components. The switching converters were chosen for their very robust input ranges. The input voltage range is 7-48V. A polarity diode is included to protect the board from incorrect wiring.

The components that should be easily accessible have been placed on the top side of the PCB and include: the wire harness connectors, indicator LEDs, USB stick, SD card slot, and debug connector. We used pin-headers that can receive wire-to-board screw terminal blocks (see Fig. 5) so that the system wiring can be easily installed or changed.

The form factor of the computer board is sized to fit through the hatch opening in the deck of the rrMM, as shown in Fig. 6. With slight modifications to either the board or to the rrMM, it could be made to fit inside the hatch itself.

4 Software Overview

The control software performs two major functions: real-time control of the sailboats various actuators based on information from its sensors, and high-level

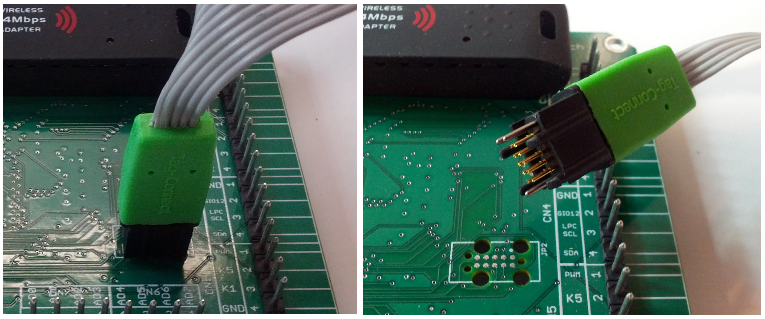


Fig. 4 Tag-Connect debug port.

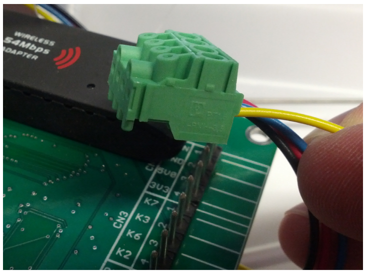


Fig. 5 The controller board has 36 connections for 3.5 mm spacing removable wire-to-board screw terminals.

functions including path planning. As discussed in Section 2, our design uses different CPUs and operating systems for these functions.

Low-level and real-time operations are performed by the MC13224v and the Contiki Operating system. Various hardware peripherals on the MC13224v are configured to interface with the sensors and systems on the robotic sailboats. Pulse width modulator (PWM) pins are configured to create the pulse train necessary to drive RC servos used for rudder and sail controls. The second serial UART interfaces with a GPS. The I2C peripheral communicates with a digital compass and accelerometer. Finally, integrated analog-to-digital converters are used to measure the output from wind sensors.

Once the peripherals are configured, the various inputs and outputs are processed using Contiki's "protothreads" mechanism. A "protothread" is a thread-like structure that is implemented using co-routines [4]. This mechanism results in isolated "processes" that run concurrently. Contiki's protothreads are implemented in standard C and do not need any special complier directives or hardware support. Separate protothreads are used to parse and convert GPS messages, operate real-time control loops, and to update information and control resources used by the next upper layer (i.e. code running on the main CPU).

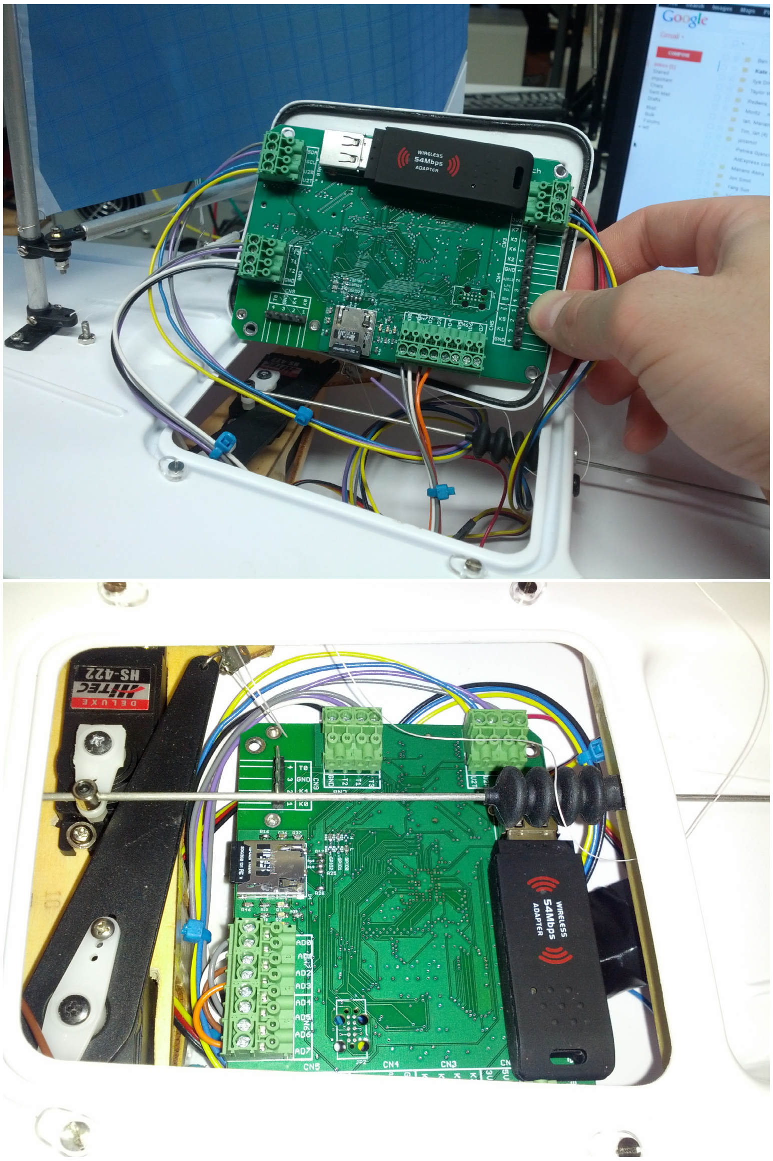


Fig. 6 Control board mounted in the hatch of a racing Micro Magic (top photograph) and inside the hull (bottom photograph).

Contiki has a rich set of networking features. We take advantage of these to expose a REST-ful interface to the sailboat [11]. The MC13224v receives an IP address from the networking environment. Then, other connected devices can perform GET and POST methods on these resources to query or set them appropriately.

Higher-level game functions are scripted in Python running in Linux on the LPC3130. The sailboat resource API is implemented as a "boat" class with state variables corresponding to the various parameters such as rudder angle and sensed wind speed. Logging and debug functions are implemented by writing files to the filesystem.

5 Design Motivation

This section describes in greater detail the motivation for and design decisions of the controller hardware. In particular, once a "full OS" + "RTOS" combination has been selected, there are a great number of available 32-bit ARM processors capable of running a "full OS." Furthermore, we examine whether there is a significant penalty in cost and/or power consumption in choosing to include "full OS" capability by comparing our approach to others used for robotic sailboat designs.

5.1 Choice of ARM Processor

In choosing a processor for the "full OS" processor, we opted to limit our search to 32-bit ARM processors capable of this task. Any of the available options satisfies the power and area constraints imposed by the rrMM; ARMs are widely used in mobile phones and consumer electronics for precisely this reason. Because every major semi-conductor company produces "full OS" capable ARMs, this design constraint does not significantly narrow down the possible options. The remaining design dimensions to explore are cost versus system performance.

When considering system cost and performance, the memory type and size compatible with a particular CPU must be considered in addition to the other CPU performance metrics. Because memory cost can vary by an order of magnitude depending on type and byte density, it has a significant impact on system cost. To characterize the tradeoff of cost versus performance, we have therefore grouped available ARM CPUs by the type of memory controller they have: SDRAM, DDR2, and DDR3. Table 3 shows the byte density and cost of the various memory types. The clock speed, core type, and cost are listed in Table 4 for a variety of CPUs with ARM cores ARM9, ARM11, and A8, in increasing order of performance. For cost numbers in these tables, we are using the qty 1000 prices from various common distributors (Digikey, Mouser, Future, etc...). Other criteria such as inclusion of a floating-point processor (FPU) or graphic processor, or specific peripherals are not considered. (Note: some designers may want an FPU on their computing platform. For our system we have decided that is not a requirement as software floating point emulation is fast enough on the selected CPU.)

We selected the lowest cost and performance corner to use in this design. This choice results in a LPC3130-based system which is a 200MHz ARM9 with 32MB of SDRAM, and an approximate bill-of-material (BOM) cost for the computing system of $5 ($3.5 CPU + $1.5 RAM).

Table 3 Comparison of byte density and cost of various types of RAM.

Memory type	SDRAM	DDR2	DDR3
Max bytes per chip	32 MB	128 MB	256 MB
Cost (USD per qty 1000)	$1.5	$4.8	$10.1

Table 4 Cost of various ARM CPUs grouped by performance class. For this work we selected the lowest cost and performance corner of the design space.

CPU	LPC31, IMX23	IMX25, 28, 35, 50, 51	IMX 53, TI OMAP
Memory Type	SDRAM	DDR2	DDR3
Memory Size (limited by type)	32MB	128 - 256MB	256-1024MB
Speed	200MHz	400MHz	600 - 1000 MHz
Core	ARM9	ARM9 - 11	A8
CPU price	$3.5 - $5	$7 - $12	$20 - $34
Total price w/ memory	$5 - $6.5	$11 - $20	$26 - $43

5.2 Comparison to Other Approaches

Table 5 contains a summary of selected computing systems used in other works, grouped according to a single "full OS" system, single "RTOS"/microcontroller system, and hybrid "full OS" + "RTOS" systems. There is a great amount of variability in the type of computing systems used for robotic sailing. Almost universally, however, a microcontroller (RTOS) is used somehow in the system. Often a "full OS" is not used. This work demonstrates that adding "full OS" capability is not detrimental to the power budget of a system, however, a single "RTOS"-only system is the lowest power option. It is important to note that the power consumption in this work is low enough to dominated by other system components such as communications power.

Table 5 also shows that communication systems used are also varied. As this work uses a USB interface for communication, a variety of different systems are easily supported. 802.15.4 is natively supported by the microcontroller we use for our RTOS functions.

Table 5 Computer system parameters from selected work. Power derived from datasheets when not explicitly cited by author. MOOP w/ gumstix shown [12].

System	Full OS	RTOS	Approx. power	Comm	Fits in rrMM?
[13]		X	70 mW	Bluetooth	yes
[6]		X	850 mW	Bluetooth	probably
[14]		X	200 mW	800-900MHz	probably
[5]	X		8000 mW	VHF	no
[12]	X	X	1250 mW	Wifi	probably
[9]		X	6 mW	802.15.4	no
This work	X	X	165 mW	802.15.4 + various	yes

6 Open Hardware and Software Release

All hardware and software files for the production of this board have been released under open hardware and open software licenses and are available at: `http://www.seascopetech.com/SBC`.

7 Conclusion

We have designed and constructed a small low-cost single board computer that is well suited for robotic sailboats including small boats such as the Graupner Micro Magic. The single-board computer uses both an ARM9 processor to run a complete operating system distribution such as Arch Linux or Debian, as well as an ARM7 microcontroller to perform real-time control operations and hardware interfacing. It supports a variety of communications systems such as Wifi, Bluetooth, or cellular modems via on-board USB; 802.15.4 is supported naively by the ARM7 co-processor. The board can be constructed for approximately $60 (USD). The full system consumes about 250mW (excluding the power consumed for communication) which is comparable to other low-power controllers for robotic sailboats. Finally, we have released the hardware and software under open source licenses to benefit the robotic sailing community.

References

1. Alves, J., Ramos, T., Cruz, N.: A reconfigurable computing system for an autonomous sailboat. In: Proceedings of the International Robotic Sailing Conference, Breitenbrunn, Austria, pp. 13–20 (May 2008)
2. Barton, T., Alvira, M.: A Discrete-Component 2D-Array Wind Sensor without Moving Parts for a Robotic Sailboat. In: Sauze, C., Finnis, J. (eds.) Robotic Sailing 2012, vol. 121, pp. 95–104. Springer, Heidelberg (2013)
3. The Contiki OS: The Operating System for the Internet of Things, `http://www.contiki-os.org/` (accessed on July 8, 2012)

4. Dunkels, A., Schmidt, O., Voigt, T., Ali, M.: Protothreads: simplifying event-driven programming of memory-constrained embedded systems. In: Proceedings of the 4th International Conference on Embedded Networked Sensor Systems, Boulder, Colorado, USA, pp. 29–42 (2006)
5. Giger, L., Wismer, S., Boehl, S., Büsser, G.-A., Erckens, H., Weber, J., Moser, P., Schwizer, P., Pradalier, C., Siegwart, R.: Design and Construction of the Autonomous Sailing Vessel AVALON. In: Proceedings of the 2nd International Robotic Sailing Conference, Matosinhos, Portugal, pp. 17–22 (July 2009)
6. Koch, M., Petersen, W.: Using ARM7 and μC/OS-II to Control an Autonomous Sailboat. In: Schlaefer, A., Blaurock, O. (eds.) Robotic Sailing, vol. 79, pp. 101–112. Springer, Heidelberg (2011)
7. Kushalnagar, N., Montenegro, G., Schumacher, C.: IPv6 over Low-Power Wireless Personal Area Networks (6LoWPANs): Overview, Assumptions, Problem Statement, and Goals. Internet Engineering Task Force Request for Comments 4919 (2007), http://www.ietf.org/rfc/rfc4919.txt
8. Microtransat, http://www.microtransat.org/ (accessed on July 15, 2012)
9. Miller, P., Brooks, O., Hamlet, M.: Development of the USNA SailBots (ASV). In: Proceedings of the 2nd International Robotic Sailing Conference, Matosinhos, Portugal, pp. 9–16 (July 2009)
10. Redwire Store (Redwire is owned and operated by Mariano Alvira), http://redwirellc.com/store/ (accessed on July 8, 2012)
11. REpresentational State Transfer (REST) entry on Wikipedia, http://en.wikipedia.org/wiki/Representational_state_transfer (accessed on July 15, 2012)
12. Sauzé, C., Neal, M.: MOOP: A Miniature Sailing Robot Platform. In: Schlaefer, A., Blaurock, O. (eds.) Robotic Sailing, vol. 79, pp. 39–53. Springer, Heidelberg (2011)
13. Schlaefer, A., Beckmann, D., Heinig, M., Bruder, R.: A New Class for Robotic Sailing: The Robotic Racing Micro Magic. In: Schlaefer, A., Blaurock, O. (eds.) Robotic Sailing, vol. 79, pp. 71–84. Springer, Heidelberg (2011)
14. Sliwka, J., Reilhac, P.-H., Leloup, R., Crepier, P., De Malet, H., Sittaramane, P., Le Bars, F., Roncin, K., Aizier, B., Jaulin, L., et al.: Autonomous Robotic Boat of ENSIETA. In: Proceedings of the 2nd International Robotic Sailing Conference, Matosinhos, Portugal, pp. 1–7 (July 2009)
15. Tag-Connect Official Website, http://www.tag-connect.com/ (accessed on July 15, 2012)

A Simple Controller for Line Following of Sailboats

Luc Jaulin and Fabrice Le Bars

Abstract. This paper proposes a simple controller for sailboat robots. The resulting controller is simple to implement and its parameters are easy to tune. Its complexity is low enough to be applicable for sailing robots with very limited computation power. The presentation contains all the necessary details to allow a fast and reliable implementation of a sailboat robot controller which follows a line. The paper also presents a simple collision avoidance strategy based on interval analysis.

1 Introduction

This paper deals with the problem of controlling a sailboat robot. It describes in a pedagogical way a controller that has been made generic enough to be used for a large class of sailboat robots. Note that it is the controller that has been implemented on the sailboat robot Vaimos [1] and has been proved to be very efficient and robust in several convincing experiments. We did our best to make the controller understandable by students that are not specialists in sailboat robotics. Our problem is motivated by the microtransat challenge [2][20] where autonomous sailboat robots have to cross the Atlantic ocean from East to West. Figure 1 illustrates the control loop to be considered in this paper. Sailboats are nonlinear hybrid systems involving strong perturbations such as waves. Moreover, to our knowledge, no realistic state equations are available for sailboats. For these reasons, existing methods from control theory [14] may not be appropriate for building reliable controllers for sailboat robots. Now, sailboats have been designed for thousand of years to be easily controlled by humans. A pragmatic approach that mimics the control strategy of sailors is thus chosen here to build reliable controllers.

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

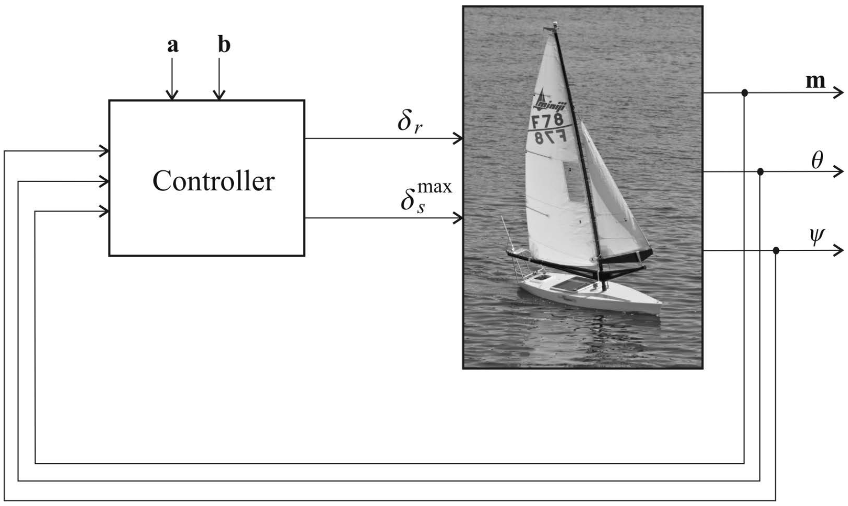


Fig. 1 Controller of a sailboat robot

The paper is organized as follows. Section 2 describes the pragmatic controller. An extension taking into account that the Earth is not flat is then considered in Section 3. In Section 4, an elementary collision avoidance strategy is proposed. A conclusion is given in Section 5.

2 Controller

A classical approach to build controllers is to take a realistic model of the system to be controlled (such as [6] for the sailboat) 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 (see also [4]). The sailboat is assumed to have three sensors and two actuators. The controller will have some parameters which are easy to tune, some reference variables and one binary state variable. Let us now describe all of them.

Sensors. 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 as shown in [22]). The position **m** is given by a GPS.

Actuators. The inputs of the robot are the angle of the rudder δ_r and the maximum angle for the sail $\delta_s^{\max}$ (which is directly related to the length of the mainsheet).

Parameters. $\delta_r^{\max}$ is the maximal angle of the rudder (we shall set $\delta_r^{\max} = \frac{\pi}{4}$), r is the cutoff distance (i.e., we want that the distance of the boat to the line be less than r; we shall choose $r = 50$m), γ_∞ is the incidence angle (we take $\gamma_\infty = \frac{\pi}{4}$) and ζ is the close hauled angle (we choose $\zeta = \frac{\pi}{3}$).

References. Two points $\mathbf{a},\mathbf{b}$ which define the line to be followed.

State Variable. This will be a discrete variable $q \in \{-1,1\}$ corresponding to the favored tack.

We propose the following algorithm to describe the controller [12]. This algorithm (presented in [11] in a theoretical form) is given in its low level form to allow a fast and reliable implementation by anyone who wants to build a controller for sailboat robots.

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

The controller has one state variable $q \in \{-1,1\}$. This is why it is both an input and an output variable of the algorithm.

Step 1. We compute the algebraic distance of the boat to the line. If $e > 0$ the robot is on the left of the line and if $e < 0$, it is on the right. In practice, since the Earth is not flat, it is important to have a reasonable distance between $\mathbf{a}$ and $\mathbf{b}$ (less than 100km). In the formula, the determinant between two vectors is defined by

$$\det(\mathbf{u},\mathbf{v}) = u_1v_2 - v_1u_2.$$

Step 2. If $|e| > \frac{r}{2} = 25m$, we are quite far from the line and the tack variable q is allowed to change its value. If for instance $e > 25m$, then q will be set to 1 and will keep this value until $e < -25m$.

Step 3. We compute the angle φ of the line to be followed (see Figure 2). In the statement, $\mathrm{atan2}(\mathbf{u}) = \mathrm{atan2}(u_1,u_2)$ represents the angle of the two-dimensional vector $\mathbf{u}$ with respect to East.

Step 4. We compute the nominal angle θ^* (see Figure 2) given by

$$\theta^* = \varphi - \frac{2.\gamma_\infty}{\pi}.atan\left(\frac{e}{r}\right).$$

This expression for θ^* makes the line attractive. When $e = \pm\infty$, we have $\theta^* = \varphi - \frac{2.\gamma_\infty}{\pi}.\left(\pm\frac{\pi}{2}\right) = \varphi \pm \gamma_\infty$, i.e., the robot has a heading which corresponds to the

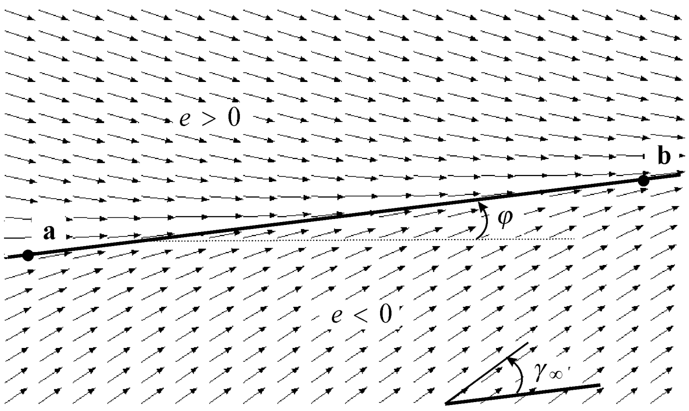


Fig. 2 Nominal vector field θ^* that the robot has follow when possible. The variable γ_∞ corresponds to the incidence angle when the distance to the boat is large ($|e| > 500m$).

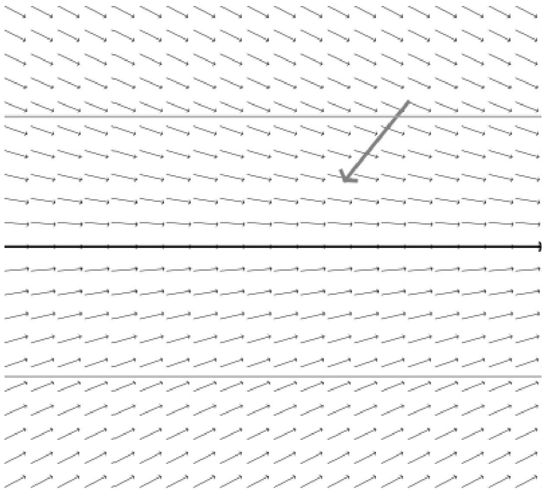


Fig. 3 The normal field may be inconsistent with the wind (bold arrow); The line to be followed corresponds to the x-axis.

angle γ_∞. For the cutoff distance $e = \pm r$, we have $\theta^* = \varphi \pm \frac{2.\gamma_\infty}{\pi}.\frac{\pi}{4} = \varphi \pm \frac{\gamma_\infty}{2}$ and for $e = 0$, $\theta^* = \varphi$, which corresponds to the direction of the line. As illustrated by Figure 3, some directions θ^* may be inconsistent with the current wind.

Step 5. If $\cos(\psi - \theta^*) + \cos\zeta < 0$, the course θ^* corresponds to a direction which is too close to the wind which the boat is unable to follow (see Figure 4). The course θ^* is thus inconsistent with the wind. If this happens, we choose a close hauled mode, i.e., the new feasible direction becomes $\bar{\theta} = \pi + \psi \pm \zeta$ (at Step 7).

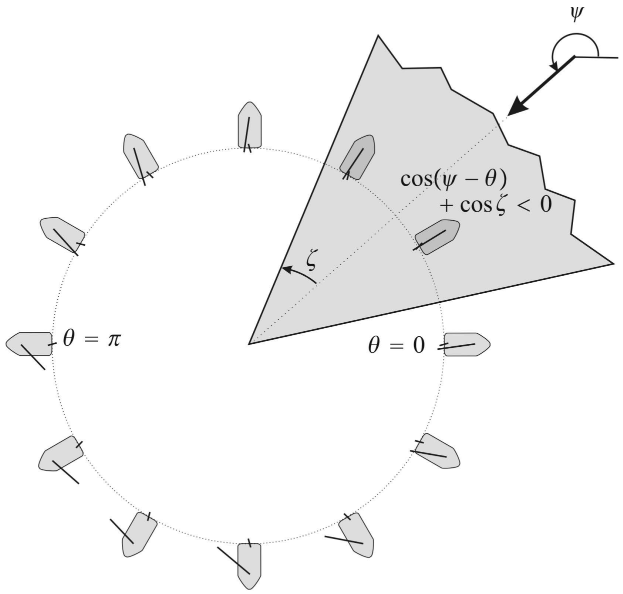


Fig. 4 Some directions for the sailboat are not feasible. These unfeasible courses forms the no-go zone painted grey.

Figure 5 represents the corresponding vector field. Thin arrows correspond to the nominal field and thick arrows correspond to the corrected field (when the latter is different from the nominal field) In this representation, we have removed the hysteresis effect induced by the tack variable q (it is equivalent to saying that we always have $q = \text{sign}(e)$).

Step 6. This step implements what we call the *keep close hauled strategy*. If $|e| < r$ and $\cos(\psi - \varphi) + \cos\zeta < 0$, we force the close hauled mode even if the route $\bar{\theta}$ is feasible. For efficiency reasons, when the line is against the wind, we do not want to loose against the wind. This is illustrated by Figure 6 where the conventions are those of Figure 5. Note that in this figure we took $\zeta = \frac{\pi}{3}$ (which corresponds to a boat that has difficulties in going upwind in a close hauled mode) which makes the line against the wind. A video with more explanations can be found at [9].

Step 7. We are in the close-hauled mode and we choose. $\bar{\theta} = \pi + \psi - q.\zeta$ (the wind direction plus or minus the close hauled angle ζ). The hysteresis variable q makes it possible to keep the current tack for 25 more meters to the line even if the line to be followed has been crossed.

Step 8. If the nominal route is satisfactory, we keep it.

Step 10. At this level, the feasible course $\bar{\theta}$ has been chosen and we want to tune the rudder. If the robot has a consistent direction, we perform a proportional control with respect to the error $\sin\left(\theta - \bar{\theta}\right)$. This is illustrated by Figure 7, Quarters 1 and 2.

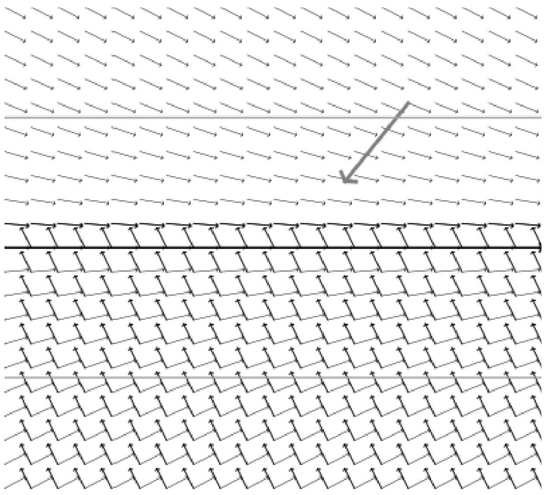


Fig. 5 Vector field provided by the algorithm if we remove Step 6. Thin arrows correspond to nominal routes. Thick arrows correspond to corrected routes when the nominal route is not feasible.

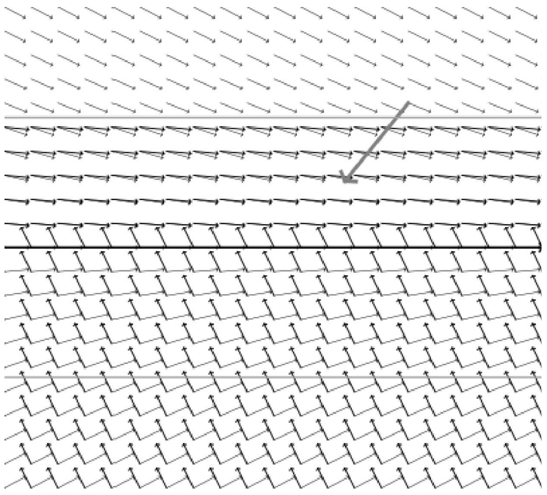


Fig. 6 Vector field provided by the algorithm including Step 6. Thin arrows correspond to nominal routes. Thick arrows correspond to corrected routes based on the keep close hauled strategy.

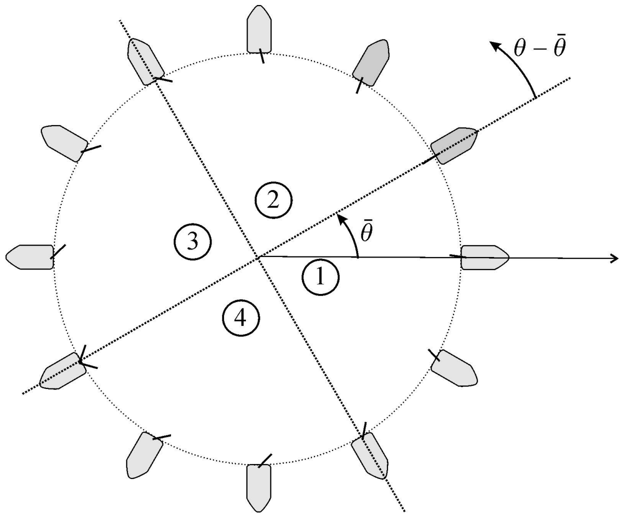


Fig. 7 Four quarter technique for the tuning of the rudder to go in the route $\bar{\theta}$. Quarters 1 and 2, we have $\cos\left(\theta - \bar{\theta}\right) \geq 0$ and a proportional control is applied; Quarters 3 and 4, we have $\cos\left(\theta - \bar{\theta}\right) \leq 0$ and a bang-bang control is chosen; Quarters 2 and 3, we have $\sin\left(\theta - \bar{\theta}\right) \geq 0$ then we turn left ($\delta_r > 0$); Quarters 1 and 4, we have $\sin\left(\theta - \bar{\theta}\right) \leq 0$ then we turn right ($\delta_r < 0$).

Step 11. If the robot does not have a consistent direction, i.e. $\cos\left(\theta - \bar{\theta}\right) < 0$, the rudder is tuned at the maximum (see Figure 7, Quarters 3 and 4).

Step 12. The length of the mainsheet is tuned with respect to the cardioid relation $\delta_s^{\max} = \frac{\pi}{2}.\left(\frac{\cos\left(\psi - \bar{\theta}\right)+1}{2}\right)$ [12]. Note that if $\psi - \bar{\theta} = \pm\pi$ (the boat is wind ahead), $\delta_s^{\max} = 0$ whereas $\delta_s^{\max} = \frac{\pi}{2}$ if the wind comes from abeam.

Remark. In practice, in a direct mode (i.e. when φ corresponds to a feasible course), a bias of 10 meters could occur, i.e. the distance to the line does not converge to 0. An integrator term could avoid this bias. To implement the integrator, it suffices to replace Step 4 by the two following statements:

$$\begin{cases} z = z + \alpha * dt * e \\ \theta^* = \varphi - \frac{2.\gamma_\infty}{\pi}.\text{atan}\left(\frac{e+z}{r}\right) \end{cases}$$

where dt is the sampling time. The variable z corresponds to the value of the integrator and converges to the bias we had without the integrator. The coefficient α should be small enough to avoid any change in the behavior of the controlled sailboat. For instance, if for $e = 10m$ for 100 sec. we want a correction of 1m, we shall take $\alpha = 0.001$. As soon as the distance to the line is higher than 50 meters (during the initialization, for instance), if the robot switches to another line (as it is the

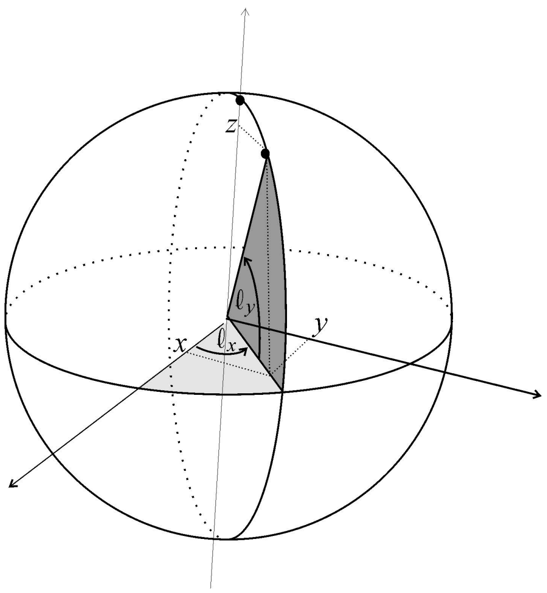


Fig. 8 Geographic reference frame

case when a line is validated), or if the robot switches to a close hauled mode, the integrator z should be forced to 0.

3 Earth Is Not Flat

We now want to take into account the fact that the Earth is not flat. We shall now adapt the controller of the previous section to our new situation. Denote by ℓ_x and ℓ_y the longitude and the latitude of a point which is located at the surface of the Earth. The transformation into the geographic coordinate system is given by

$$\mathscr{T}:\begin{pmatrix}\ell_x\\ \ell_y\end{pmatrix}\to\begin{pmatrix}x\\ y\\ z\end{pmatrix}=\begin{pmatrix}\rho\cos\ell_y\cos\ell_x\\ \rho\cos\ell_y\sin\ell_x\\ \rho\sin\ell_y\end{pmatrix}\qquad(1)$$

where $\rho = 6371000m$ is the radius of the Earth (see Figure 8).

Consider three points $\mathbf{a},\mathbf{b},\mathbf{m}$ at the surface of the Earth (see Figure 9). The vector

$$\mathbf{n}=\frac{\mathbf{a}\wedge\mathbf{b}}{\|\mathbf{a}\|\,\|\mathbf{b}\|}$$

is normal to the plane $(\mathbf{oab})$ and has a norm equal to 1. The algebraic distance from $\mathbf{m}$ to the plane $(\mathbf{oab})$ is given by

$$e=\mathbf{m}^{\mathrm{T}}.\mathbf{n}.$$

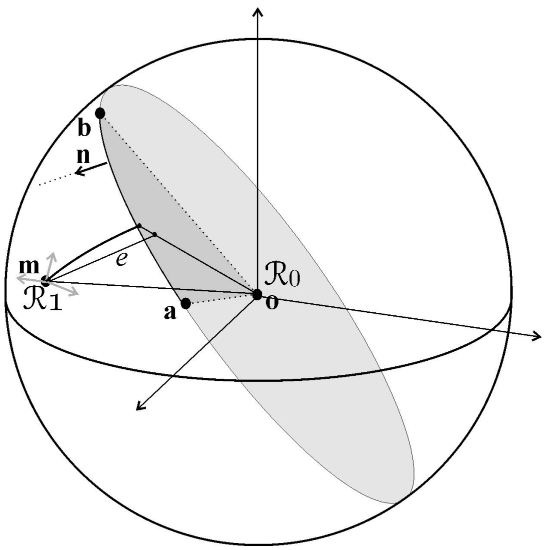


Fig. 9 Line (**ab**) that has to be followed by the robot.

Let us differentiate the relation (1). We get

$$\begin{pmatrix} dx \\ dy \\ dz \end{pmatrix} = \mathbf{J}. \begin{pmatrix} d\ell_x \\ d\ell_y \\ d\rho \end{pmatrix}$$

where

$$\mathbf{J} = \begin{pmatrix} -\rho\cos\ell_y\sin\ell_x & -\rho\sin\ell_y\cos\ell_x & \cos\ell_y\cos\ell_x \\ \rho\cos\ell_y\cos\ell_x & -\rho\sin\ell_y\sin\ell_x & \cos\ell_y\sin\ell_x \\ 0 & \rho\cos\ell_y & \sin\ell_y \end{pmatrix}.$$

This formula can be used to find the geographic coordinates of the cardinal directions. For instance, the vector corresponding to the East is given by the first column. Equivalently, we are able to build a East-North-Elevation frame $\mathscr{R}_1$ around the robot (in grey on Figure 9). The corresponding rotation matrix is obtained by normalizing each column of the Jacobian matrix $\mathbf{J}$:

$$\mathbf{R} = \begin{pmatrix} -\sin\ell_x & -\sin\ell_y\cos\ell_x & \cos\ell_y\cos\ell_x \\ \cos\ell_x & -\sin\ell_y\sin\ell_x & \cos\ell_y\sin\ell_x \\ 0 & \cos\ell_y & \sin\ell_y \end{pmatrix}.$$

The transformation relation to move from the geographic frame $\mathscr{R}_0$ to the robot frame $\mathscr{R}_1$ is

$$\mathbf{v}_{|\mathscr{R}_1} = \mathbf{R}^{\mathrm{T}}.\mathbf{v}_{|\mathscr{R}_0}. \quad (2)$$

To get φ, take the vector $\mathbf{b}-\mathbf{a}$, express it in the $\mathscr{R}_1$ (using (2)) frame, project it into the $(\mathbf{i},\mathbf{j})$ frame (by selecting the two first rows) and take its argument (using the *atan2* function). This gives

$$\varphi = \text{atan2}\left(\mathbf{M}.\left(\mathbf{b}-\mathbf{a}\right)_{|\mathscr{R}_0}\right),$$

where

$$\mathbf{M} = \begin{pmatrix} -\sin \ell_x^{\mathbf{m}} & \cos \ell_x^{\mathbf{m}} & 0 \\ -\cos \ell_x^{\mathbf{m}} \sin \ell_y^{\mathbf{m}} & -\sin \ell_x^{\mathbf{m}} \sin \ell_y^{\mathbf{m}} & \cos \ell_y^{\mathbf{m}} \end{pmatrix}.$$

The resulting controller is given by the following table

```
Function in : $\theta, \psi, \ell_x^{\mathbf{a}}, \ell_y^{\mathbf{a}}, \ell_x^{\mathbf{b}}, \ell_y^{\mathbf{b}}, \ell_x^{\mathbf{m}}, \ell_y^{\mathbf{m}}$;
         out: $\delta_r, \delta_s^{\max}$; inout: $q$
  $\mathbf{a} = \mathscr{T}\left(\ell_x^{\mathbf{a}}, \ell_y^{\mathbf{a}}\right)$; $\mathbf{b} = \mathscr{T}\left(\ell_x^{\mathbf{b}}, \ell_y^{\mathbf{b}}\right)$; $\mathbf{m} = \mathscr{T}\left(\ell_x^{\mathbf{m}}, \ell_y^{\mathbf{m}}\right)$;
  $e = \mathbf{m}^{\mathrm{T}}.\frac{\mathbf{a}\wedge\mathbf{b}}{\|\mathbf{a}\|\|\mathbf{b}\|}$;
  if $|e| > \frac{r}{2}$ then $q = \text{sign}(e)$;
  $\mathbf{M} = \begin{pmatrix} -\sin \ell_x^{\mathbf{m}} & \cos \ell_x^{\mathbf{m}} & 0 \\ -\cos \ell_x^{\mathbf{m}} \sin \ell_y^{\mathbf{m}} & -\sin \ell_x^{\mathbf{m}} \sin \ell_y^{\mathbf{m}} & \cos \ell_y^{\mathbf{m}} \end{pmatrix}$;
  $\varphi = \text{atan2}\left(\mathbf{M}.\left(\mathbf{b}-\mathbf{a}\right)\right)$;
  $\theta^* = \varphi - \frac{2.\gamma_\infty}{\pi}.\text{atan}\left(\frac{e}{r}\right)$;
  if $\cos\left(\psi - \theta^*\right) + \cos\zeta < 0$
    or $(|e| < r$ and $(\cos(\psi - \varphi) + \cos\zeta < 0))$
        then $\bar{\theta} = \pi + \psi - q.\zeta$;
        else $\bar{\theta} = \theta^*$;
  end;
  if $\cos\left(\theta - \bar{\theta}\right) \geq 0$ then $\delta_r = \delta_r^{\max}.\sin\left(\theta - \bar{\theta}\right)$;
  else $\delta_r = \delta_r^{\max}.\text{sign}\left(\sin\left(\theta - \bar{\theta}\right)\right)$;
  $\delta_s^{\max} = \frac{\pi}{2}.\left(\frac{\cos\left(\psi - \bar{\theta}\right)+1}{2}\right)$.
```

4 Avoiding Collisions

In this section, we assume again (as in Section 2) that the Earth can be approximated by a plane on which a Cartesian frame *Oxy* is available. Consider the situation where m other boats are detected at time $t = 0$ by our robot, for instance using an AIS (Automatic Identification System). Note that the time $t = 0$ is chosen as the reference time and does not correspond to the beginning of the mission. We assume that we measure the speed and the position of these boats with a known accuracy. The speed is considered as constant. More precisely, the position of all these boats is assumed to satisfy

$$\mathbf{m}^i(t) = \mathbf{a}^i.t + \mathbf{b}^i, \quad i \in \{1, \dots m\}$$

where $\mathbf{a}^i$ and $\mathbf{b}^i$ are vectors of $\mathbb{R}^2$ which correspond to the speed vector and the initial position of each boat. Since we measure these two quantities with a known accuracy, we have two boxes $\left[\mathbf{a}^i\right]$ and $\left[\mathbf{b}^i\right]$ which enclose $\mathbf{a}^i$ and $\mathbf{b}^i$, respectively. Moreover, we assume that the trajectory of our robot is described by

$$\mathbf{m}^0(t) = \mathbf{a}^0.t + \mathbf{b}^0.$$

The two vectors $\mathbf{a}^0$ and $\mathbf{b}^0$ can be obtained by taking into account the characteristics of the line to be followed (see Section 2) and the speed of our robot (which can be estimated from the previous GPS measurements). Again, we assume that we have two boxes $\left[\mathbf{a}^0\right]$ and $\left[\mathbf{b}^0\right]$ which enclose $\mathbf{a}^0$ and $\mathbf{b}^0$. We shall propose a method based on interval analysis [17] [13] [21] to prove that our robot's trajectory is safe. Interval analysis is a numerical tool able to deal with nonlinear problems involving uncertainties (see, e.g. [5], [15], [10], [16] and [19] in the context of robotics). It has also been used in the context of sailboat robotics [8]. First, recall some basic interval operations that will be used later.

$$\begin{aligned}
[x^-,x^+] + [y^-,y^+] &= [x^- + y^-, x^+ + y^+]\\
[x^-,x^+] - [y^-,y^+] &= [x^- - y^+, x^+ - y^-]\\
[x^-,x^+] * [y^-,y^+] &= [\min(x^-y^-,x^+y^-,x^-y^+,x^+y^+), \max(x^-y^-,x^+y^-,x^-y^+,x^+y^+)].
\end{aligned}$$

For instance

$$[2,3]*[-1,2]+[3,4] = [-3,6]+[3,4] = [0,10].$$

Proposition 1. If for all $i \in \{1,\ldots m\}$,

$$\left\{\begin{array}{cl}
0 \notin \left(\left[a_x^0\right] - \left[a_x^i\right]\right) * [0,t_{\max}] + \left[b_x^i\right] - \left[b_x^0\right] & \text{or}\\
0 \notin \left(\left[a_y^0\right] - \left[a_y^i\right]\right) * [0,t_{\max}] + \left[b_y^i\right] - \left[b_y^0\right] & \text{or}\\
0 \notin \left(\left[a_y^0\right] - \left[a_y^i\right]\right) * \left(\left[b_x^i\right] - \left[b_x^0\right]\right) - \left(\left[a_x^0\right] - \left[a_x^i\right]\right) * \left(\left[b_y^i\right] - \left[b_y^0\right]\right) &
\end{array}\right.$$

then the trajectory of the robot is collision free inside the time interval $[0,t_{\max}]$.

Proof. Our trajectory is *collision-free* inside an interval $[0,t_{\max}]$ if

$$\forall i \in \{1,\ldots m\}, \forall t \in [0,t_{\max}], \mathbf{m}^i(t) \neq \mathbf{m}^0(t).$$

By taking the contrapositive of this proposition, we get that if the system

$$\left\{\begin{array}{c}
\left(\mathbf{a}^0 - \mathbf{a}^i\right).t + \mathbf{b}^0 - \mathbf{b}^i = 0,\\
t \in [0,t_{\max}]\\
\mathbf{a}^0 \in \left[\mathbf{a}^0\right], \mathbf{b}^0 \in \left[\mathbf{b}^0\right], \mathbf{a}^i \in \left[\mathbf{a}^i\right], \mathbf{b}^i \in \left[\mathbf{b}^i\right],
\end{array}\right.$$

or equivalently

$$\left\{\begin{array}{c}
\left(a_x^0 - a_x^i\right)t + b_x^i - b_x^0 = 0\\
\left(a_y^0 - a_y^i\right)t + b_y^i - b_y^0 = 0\\
t \in [0,t_{\max}]\\
a_x^0 \in \left[a_x^0\right], b_x^0 \in \left[b_x^0\right], a_y^0 \in \left[a_y^0\right], b_y^0 \in \left[b_y^0\right]\\
a_x^i \in \left[a_x^i\right], b_x^i \in \left[b_x^i\right], a_y^i \in \left[a_y^i\right], b_y^i \in \left[b_y^i\right]
\end{array}\right. \tag{3}$$

has no solution for all $i \in \{1,\ldots m\}$, then our trajectory is collision free. Now, from the two first lines of (3), we get

$$\left(a_y^0 - a_y^i\right)\left(b_x^i - b_x^0\right) - \left(a_x^0 - a_x^i\right)\left(b_y^i - b_y^0\right) = 0.$$

The fundamental theorem of interval analysis [17] applied on the three equations terminates the proof. ■

Method for Avoiding Collisions. We propose to take $t_{\max} = 10\,\text{min}$ and apply the following procedure every minute:

Step 1. *Normal mode*. Using proposition 1, check if the current course (with angle φ) that is followed by the robot is collision free. If it is not the case, go to Step 2.

Step 2. *Anchor mode*. Anchor (virtually) the robot for 10 minutes. Go to Step 1.

5 Conclusion

In this paper we have presented a simple controller to allow a sailboat robot to follow a line. The controller is easy to understand, implement, test and debug, compared to other more sophisticated controllers such as the one developed by Guillou [7] or by Bruder et. al. [3]. All computations can be performed using any cheap and low-powered microcontrollers, which is a key point in the context of sailboat robotics where the energy is highly limited. The controller can easily be adapted to build controllers which are less generic, but more efficient since they can be tuned on a particular sailboat. A simple collision avoidance strategy based on interval analysis has also been presented in the last part of the paper.

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 Bretagne). The authors render thanks to LPO and RDT for having made available *Vaimos* to test the controllers presented in this paper. They also thank 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. Le Bars, F., Jaulin, L.: An Experimental Validation of a Robust Controller with the VAIMOS Autonomous Sailboat. In: Sauze, C., Finnis, J. (eds.) Robotic Sailing 2012, vol. 121, pp. 73–84. Springer, Heidelberg (2013)
2. Brière, Y.: The first microtransat challenge. ENSICA (2006), http://web.ensica.fr/microtransat

3. Bruder, R., Stender, B., Schlaefer, A.: Model sailboats as a testbed for artificial intelligence methods. In: International Robotic Sailing Conference, Matosinhos, Portugal (2009)
4. Cruz, N.A., Alves, J.C.: Ocean sampling and surveillance using autonomous sailboats. In: International Robotic Sailing Conference, Austria (2008)
5. Drevelle, V., Bonnifait, P.: High integrity gnss location zone characterization using interval analysis. In: ION GNSS (2009)
6. Gale, T.J., Walls, J.T.: Development of a sailing dinghy simulator. Simulation 74(3), 167–179 (2000)
7. 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)
8. Herrero, P., Jaulin, L., Vehi, J., Sainz, M.A.: Guaranteed set-point computation with application to the control of a sailboat. International Journal of Control Automation and Systems 8(1), 1–7 (2010)
9. http://youtu.be/pHteidmZpnY
10. Jaulin, L.: A Nonlinear Set-membership Approach for the Localization and Map Building of an Underwater Robot using Interval Constraint Propagation. IEEE Transaction on Robotics 25(1), 88–98 (2009)
11. Jaulin, L., Le Bars, F.: An interval approach for stability analysis; Application to sailboat robotics. Submitted to IEEE Transaction on Robotics (2012)
12. 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)
13. Kearfott, R.B., Kreinovich, V. (eds.): Applications of Interval Computations. Kluwer, Dordrecht (1996)
14. Khalil, H.K.: Nonlinear Systems, 3rd edn. Prentice-Hall (2002)
15. LeBars, F., Sliwka, J., Reynet, O., Jaulin, L.: State estimation with fleeting data. Automatica 48(2), 381–387 (2012)
16. Lucidarme, P., Hardouin, L., Paillat, J.L.: Variable geometry tracked vehicle (vgtv) prototype: conception, capability and problems. In: Humans Operating Unmanned Systems (HUMOUS) Conference, France - Brest, pp. 115–126 (2008)
17. Moore, R.E.: Interval Analysis. Prentice-Hall, Englewood Cliffs (1966)
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. Ramdani, N., Poignet, P.: Robust dynamic experimental identification of robots with set membership uncertainty. IEEE/ASME Transactions on Mechatronics 10(2), 253–256 (2005)
20. Sauze, C., Neal, M.: An autonomous sailing robot for ocean observation. In: Proceedings of TAROS 2006, Guildford, UK, pp. 190–197 (2006)
21. Tucker, W.: The Lorenz attractor exists. Comptes Rendus de l'académie des Sciences 328(12), 1197–1202 (1999)
22. Xiao, K., Sliwka, J., Jaulin, L.: A wind-independent control strategy for autonomous sailboats based on voronoi diagram. In: CLAWAR 2011, Paris (2011) (best paper award)

Author Index