Navigation Strategy and Trajectory Following Controller for an Autonomous Sailing Vessel

Hendrik Erckens (corresponding author), Gion-Andri Büsser,
Dr. Cédric Pradalier, Prof Dr. Roland Y. Siegwart
Autonomous Systems Lab, Swiss Federal Institute of Technology (ETH) Zurich
Leonhardstrasse 25, 8092 Zurich, Switzerland
E-mail: herckens@student.ethz.ch

***Abstract*—This paper describes the design and implementation of the autonomous sailing vessel AVALON. This boat, designed for participating in the MICROTRANSAT, is designed to autonomously cross the Atlantic Ocean. We present here a specially robust mechanical design and the navigation software that will plan optimal navigation course and efficiently control the boat on this course. The path planner uses an A* algorithm to generate the fastest path to a given destination. It is able to avoid both static and dynamic obstacles. By using a given polar diagram and measured wind data, it takes into account manoeuvrability constraints of a sailboat. The control system takes care of rudder and sail in order to follow a given heading while optimising speed. Additionally, the system is capable of automatically conducting manoeuvres such as tack and jibe. At the time of this writing, AVALON and its software systems have been successfully tested more than two weeks in short deployments lasting several hours with winds ranging from 0 up to 30 knots.**

I. INTRODUCTION

This paper introduces the autonomous sailboat AVALON as seen in figure 1. The general objective for building an autonomous sailing vessel is to further research and development in the area of unmanned, autonomous robotic vehicles that are exposed to heavy environmental conditions like the Atlantic Ocean. It has been established that there is a demand for autonomous sailboats to be used for ocean sampling and surveillance [1] but also for the implementation of this type of software in manned sailing vessels. Either by passively proposing optimal sailing routes or by actively supporting the sailor in dangerous situations by piloting the sail or rudders, an integrated autonomous sailing system could back up and facilitate a lot of situations. For example, if a single hand sailor falls over board, the system could detect this incident and automatically execute a Man Over Board manoeuvre.

A. Our Goals and Objectives for this Work

Our boat AVALON was designed to compete in the MICROTRANSAT Challenge and thus has to withstand the harsh conditions on the Atlantic Ocean.

The goal of this paper is to present a software that is capable of calculating the optimal path to reach a given destination as quickly as possible, and the design of a controller able to safely and efficiently guide the sailboat along the calculated path, taking into account the current environmental conditions, such as wind speed.

B. Review of Existing Autonomous Sailing Boats

An autonomous sailboat has to be capable of sailing without human intervention. That means, that all tasks that are normally carried out by the sailor on a conventional sailboat have to be done autonomously. That involves navigation as well as working the rudder and sail in order to steer the defined course. Other autonomous sailboats have been developed mostly by MICROTRANSAT participants [2], [3], [4], [5], but also commercially usable products [6] have been built. In contrast to most of the existing work, we use a path planner to avoid obstacles. Additionally, our controller switches between many different modes of sailing, in order to make better use of wind shifts.

II. A BOAT DESIGNED FOR SURVIVAL

To proved an insight into the platform that was used to implement the proposed software, this section summarises some details about the main components of AVALON. The mechanical system was more comprehensively described in [7]. Table I shows the major technical details of AVALON.

A. Rig

The rigging system is one of the most important parts of the whole assembly. A defect in its structure will inevitably cause the whole project to fail. The following aspects were considered for the design:

- High loads and forces on the mechanical structures due to strong winds and heavy weather conditions.
- Highest demands on reliability. There is no chance to repair anything throughout the journey.
- Preferably efficient force transmission from the actuator to the sail in order to save as much energy as possible

During the design phase it turned out that a balanced rig prevails over a conventional rig. AVALON's rig does not use any ropes that could generate knots and jams. It is pivoted on a central bearing without shrouds and stays. The result is a simple and reliable construction.

B. Hull

The hull was designed using the CAE software FRIENDSHIP-FRAMEWORK and was laminated inside a female mold using glass fibre sandwich. Epoxide resin was

sucked into dry glass fibre material by vacuum in a so called *Infusion Process*. Compared to conventional laminating, this method is much cleaner and more convenient.

TABLE I
TECHNICAL DATA OF AVALON

Variable	*Description*	*Value*
L_{OA}	Length over all	$3.95\,m$
B_{max}	Maximal beam	$0.7\,m$
T	Draft without keel	$0.25\,m$
D_{max}	Draft over all	$2\,m$
H_{rigg}	Height of the rigg	$5.7\,m$
A_{sail}	Total sail area	$8.4\,m^2$
∇	Submerged volume	$0.44\,m^3$

C. Keel and Rudder

1) Keel: In order to achieve a sufficient righting moment and stability during heavy weather situations, AVALON's keel with a draft of two meters consists of a slim fin with a 160 kg ballast bulb.

The fin and parts of the bulb were made of high-modulus carbon fibre in precisely milled polyurethane female molds. After hardening and tempering, the bulb was filled with lead and the two halves were glued together.

2) Rudder: A twin-rudder system was selected for AVALON to make sure to have sufficient steering effect in every sailing situation. Angular mounted twin-rudders provide better control at high heeling angles compared to single rudders.

Assembled inside the hull, the rudder actuators are well sealed and protected against water and humidity.

D. Power Supply

The main power supply is realised through two square meters of solar panels providing a total of 360 Wp. The collected energy is stored in four lithium-manganese batteries. Each battery consists of 70 single cells and has a capacity of 600 Wh at a nominal voltage of 25.2 volts. Lithium-manganese batteries were chosen mainly because of their weight but also because they are fairly safe to use.

For back-up power, the boat has a direct-methanol fuel cell on board. This fuel cell is automatically activated when the voltage drops under a certain value, the *switch on* voltage. It then charges the batteries until the *switch off* voltage is reached. In theory, the solar cells provide enough power for the boat's systems. The fuel cell only serves in case of enduring bad weather or other unforeseeable circumstances.

Fig. 1. Avalon ready to sail on the lake of Zürich

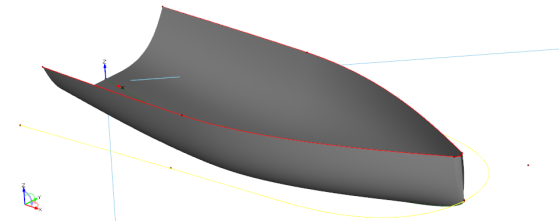


Fig. 2. Hull

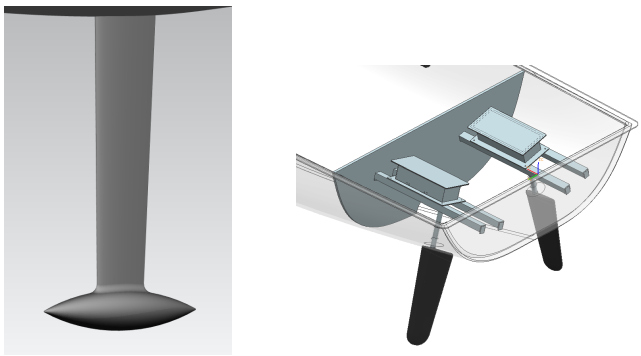


Fig. 3. Keel with fin and ballast bulb (left), Twin rudder system (right)

III. NAVIGATION STATEGY

This chapter will give a general overview of the software developed for AVALON and discuss the interaction between control system and path planner. Sections IV and V will then go further into details of the actual controller and path planner respectively.

The core hardware part and the brain of the system on the sailing vessel is a main computer MPC21 from DIGITALLOGIC is a 500 MHz device with 1024 MB RAM and a compact flash hard-drive. The device has an average power consumption of about 8 watts, the protection of a metal housing and a total weight of less than 1 kg.

The base of the entire software structure is the middleware DDX [9] that runs on a Linux Operating System. This software manages a shared memory called the Store and thus provides a means of communication between individual programs that all run in parallel (see figure 5).

Fig. 4. Main PC and brain of AVALON [8]

Sensor drivers will collect data from the specific sensors (see section III-A) and store it in the DDX Store, from where it can be accessed by the control and path planning programs.

A distinct advantage of DDX is the modular software structure. If ever a program crashes, it can be restarted without interrupting the other processes. The different subprograms used on AVALON are depicted in figure 5.

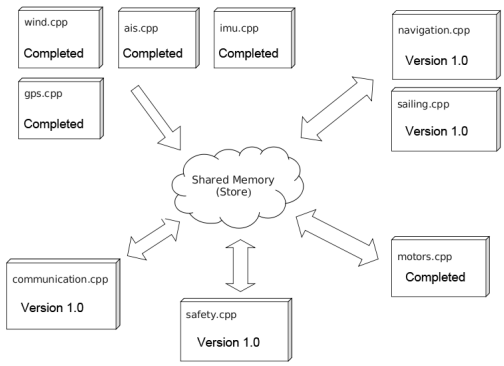


Fig. 5. Software organisation

A. Sensors

All desired information such as position, heading or speed are measured by several sensors located all over the boat. To get a fully controllable system, data is collected from the following sensors:

1) GPS & IMU: The localization of the boat is performed by an *Inertial Measurement Unit* (by X-SENS) that is combined with a GPS receiver. Using the accelerations in all 6 degrees of freedom and a high update rate of 120 Hz, occasional deviations of the GPS-data will automatically be smoothed out. The system therefore produces highly accurate positioning.

2) Wind: Mounted on top of the mast, the wind-sensor provides wind speed and direction. AVALON has an ultrasonic wind-sensor that promises less mechanical failure than a conventional sensor with a turning wheel. The sensor by German company DEIF is IP-66 approved and connected to the main computer via a RS-485 interface.

3) AIS: This Sensor receives data such as position and velocity from other boats via VHF. The AIS system is an additional means of perception that ensures that collisions with large commercial ships are avoided.

B. Navigation Management

The overall navigation management is being performed by a program further referred to as trajectory tracker. It's two main tasks are to trigger the local planner of the path planning algorithm (see section V) and to extract from a planned path a sailable boat heading that can be followed by the controller (see section IV).

The program keeps track of the boats location as well as it's environment and, in case of predefined emergency cases, is able to set new destination coordinates.

Having a state machine architecture, the trajectory tracker can switch between the following states:

1) Standard Navigation: If not approaching a target or buoy, this state generates headings for the sailor, going from way-point to way-point. If the wind changes more than 20° or if the ship distances itself from the current trajectory for more than a predefined distance, a new calculation will be initiated.

2) Goal Approach: If approaching a target position, the trajectory tracker will always send a heading that is directing exactly towards that point.

3) Buoy Approach: This is a state designed especially for short course racing. Due to regulations that always require a port-side surrounding of the regatta buoys, this state will - when approaching a buoy - always write a heading that will lead the boat around the buoy.

4) New Calculation: In this state, the trajectory tracker manages and supervises the new calculation. After having checked that the new way-points have been stored, it switches back to *Standard Navigation*

IV. CONTROL SYSTEM

A. Objectives

This section describes the design of a controller that is capable of robustly steering a sailboat along a predefined course while always setting the sail to the optimal angle to the wind in order to generate as much driving force as possible. For that, it also has to be able to conduct manoeuvres such as tack and jibe. At the same time, the controller should take into account the changing environmental conditions, such as wind speed and direction and react accordingly. Furthermore, since the path planner (section V) calculates a global trajectory and relies on the control system to follow this course, the autonomous sailor has to compensate for drift.

B. Existing Works

There are various systems available that assist a sailor in steering a sailboat. Most of these are commercially available autopilots, that operate either with a built in magnetic compass or with a wind vane [10]. These systems are finished products, that have been used for many years and are very robust and reliable. We did not use commercial products, mainly because of interfacing issues. Autopilots are designed for a human sailor on board who does the navigation and has to tune the sails. Although some of the systems are even able to perform tacks and jibes, the manoeuvre has to be confirmed by the human sailor pressing a button on the autopilot's

control panel. Since the software integrated in commercial autopilots is typically closed source, this problem cannot be solved by simply reprogramming the device. Furthermore, most autopilots only control the rudder, so we would still have to figure out a way to tune the sail.

Apart from the commercial products, there are also several scientific works regarding control systems for autonomous sailboats, most of them also by participants in the MICROTRANSAT. Developments reach from a fuzzy logic controller for both rudder and sail [11], over an LQG controller based on a non-linear 3 degrees of freedom (DOF) model [12], up to a complex self learning AI system that has been successfully tested on an Open60 racing yacht. Our work is mainly influenced by Briere [2], who describes a simple controller based on a state machine.

C. System Modelling

In order to design a robust controller and identify parameters in a controlled environment, a simulator is very helpful. Although there are several sailing simulators available, most of them are made for gaming. Rather than having an interface that allows them to be steered by a self-developed control system, they are meant to be operated by hand. For this reason we implemented a simulator in Matlab/Simulink.

To gain a general system that could later be extended with boat/wave interaction, the boat was modelled as a rigid body with full 6 DOF. The system has 12 state variables: 3 for position, 3 for attitude, 3 for velocity and 3 for turning rates. Inputs to the plant are rudder angle γ_{rudder}, sail angle γ_{sail}, true wind speed $v_{\text{wind\_true}}$ and direction Ψ_{wind}, while outputs are the boat's position and attitude and their corresponding derivatives.

Forces were introduced at various points of the boat to model the interactions between water and hull as well as wind and sail.

1) Sail Force: In order to calculate the force generated by the sail, the apparent wind angle is needed which is computed in a vector operation from the true wind angle and the boat's velocity. The 2 components of the sail force F_{sail} are then modelled using the standard approximation for air foils in a moving fluid

$$\begin{aligned} F_{\text{sail\_lift}} &= \frac{1}{2} \cdot \rho_{\text{air}} \cdot v^2_{\text{wind\_app}} \cdot A_{\text{sail}} \cdot c_l(\alpha_{\text{sail}}) \\ F_{\text{sail\_drag}} &= \frac{1}{2} \cdot \rho_{\text{air}} \cdot v^2_{\text{wind\_app}} \cdot A_{\text{sail}} \cdot c_d(\alpha_{\text{sail}}) \end{aligned} \tag{1}$$

where ρ_{air} is the density of air, A_{sail} is the apparent area of the sail and c_l is the lift coefficient which depends on the sail's shape and the angle of attack α_{sail}.

2) Rudder Force: The rudder force is also modelled using equation 1, just with different parameters. Since the rudders are at the stern of the boat, the force induces a moment around the vertical axis and thus affects the boat's heading.

3) Resistances: There are several damping forces that depend on the velocity and rate of turn of the boat. Those are mainly the resistance of hull and keel in all 3 axes of translational freedom and the damping of rotations. These are modelled using the equation

$$F = d \cdot v^2 \tag{2}$$

where v is the velocity or rate of turn and d is a parameter that has to be identified in experiments.

After the first tests with the real boat, we found the real boat behaviour to be quite different from the simulation. To improve this, more tests and parameter identification needs to be done. Due to time constraints this has not been done at the time of this writing.

D. Controller Principle

In our control system layout, rudder and sail are controlled in two separate Single Input Single Output (SISO) systems that are assumed to be independent of each other. During testing, this assumption has proved to be very reasonable.

1) Sail Control: For the sail, the controlled value is the angle of attack. This directly depends on the sail's angle. Since AVALON's sail and rudder motors already come with a built in positioning controller, this subsystem is sufficiently controlled by just writing a reference value to the motor controller.

This reference value is derived from a predefined optimal Angle Of Attack (AOA) that was found in experiments. However, for safety reasons, the wanted AOA also changes dynamically depending on the wind speed. Basically, as the wind speed increases, the wanted AOA approaches zero, which reduces the sail force generated by the wind. With this behaviour, the boat remains steerable even in strong winds.

2) Rudder Control: The second subsystem controls the boat's heading using the rudder as input. Since the heading changes dynamically and is not directly dependent on the rudder angle, this system takes a little more consideration. For this purpose, a heading error minimising PID controller was designed in an iterative process using the Matlab simulation. It was then tested and optimised in the real boat (see section IV-E).

In order to compensate for leeway drift, the boat has to sail somewhat closer to the wind than the desired heading DH_{nav} calculated by the path planner. Since drift changes considerably with the wind and wave conditions, it is important to know the current drift in order to compensate it. AVALON is equipped with an Inertial Measurement Unit (IMU) with a built in GPS receiver. With this sensor, the drift speed v_{drift} can be accurately measured at any time. From the drift, a new desired heading $\text{DH}_{\text{sailor}}$ is calculated using the equation

$$\text{DH}_{\text{sailor}} = \text{DH}_{\text{nav}} + \arctan\left(\frac{v_{\text{drift}}}{v_{\text{boat}}}\right) \tag{3}$$

3) State Machine: A sailboat can not sail at all angles to the wind. Figure 6 shows the ranges that we cannot sail in gray. In order to sail as fast as possible, it has to be differentiated between different types of sailing. To do that, the controller is designed as a state machine that switches states depending on the angle to the wind. The 5 states are depicted in figure 6.

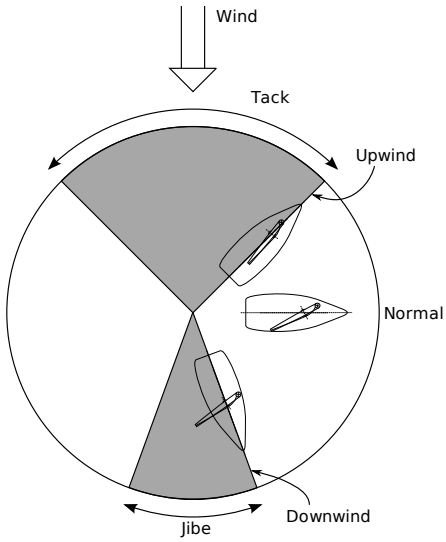


Fig. 6. Possible courses of a sailing yacht with respect to the wind and corresponding control system states. The range with wind directly from behind could be sailed in theory, but is not very efficient and rather unstable.

4) Normal Sailing: If the desired heading given by the path planner (section V) is in the sailable (white) range, the system is in state `Normal Sailing`. In this state the rudder controller follows the given desired heading, while the sail controller permanently adjusts the sail to achieve the optimal angle of attack on the sail. If the desired heading changes within the sailable range, the rudder controller simply follows the new course while the sail controller maintains a constant angle of attack.

5) Upwind Sailing: If the desired heading for whatever reason is set in the not sailable (gray) range, it cannot be directly followed. The objective in this case is to make as much velocity towards the wind as possible. In `Upwind Sailing` mode the sail is set to the tightest position that is reasonably possible. While the sail angle is kept at this constant position, the rudder controller now keeps a constant angle to the true wind and thus takes advantage of all wind shifts.

E. Test Results and Analysis

Several tests were undertaken to verify and optimise the control system. Before any testing on the water was done, the state machine's transition conditions had to be verified. This was done by putting the boat on a trailer on a sufficiently windy day and turning it around its vertical axis while constantly checking the current state. We encountered many problems that were caused by the sign change between -180^o and $+180^o$. Some time could have been saved, if the whole control program, including the state machine, had been tested more thoroughly in the simulation environment.

The controller was then tested for its ability to keep a desired heading and to react to changes in desired heading. After some tuning of the PID parameters we found that the parameters in table II yield fast reactions without too much overshoot. A step response in winds of about 20 knots is illustrated in figure 7. Note that 20 knots is already a strong

TABLE II
PID PARAMETERS

P	0.3
I	0.005
D	30

wind and that in lower wind speeds we experienced much less oscillation around the reference value. Even this $\pm 10^o$ oscillation that is mostly caused by waves can barely be seen on the boat. We then also applied disturbances to the boat by deviating it from its course by hand. The reaction was the same as with changes in desired heading.

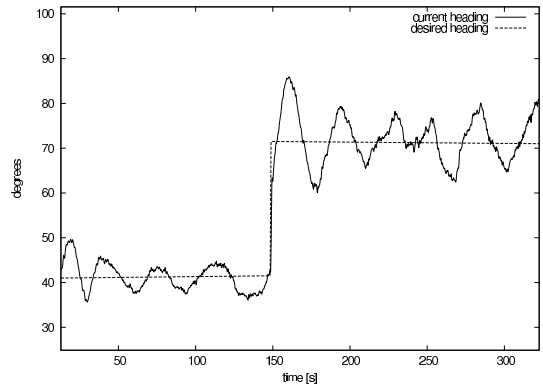


Fig. 7. Step response in state `Normal Sailing`. Conditions were 20 kn wind from 140°

Figure 8 shows current heading, desired heading and wind direction during three tacks. Between the tacks, the controller state machine is in state `Upwind Sailing`. It can be very well seen, that the boat keeps a constant angle to the wind and takes advantage of wind shifts as expected. A lot of

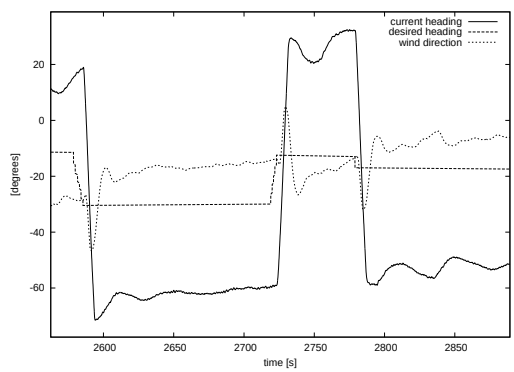


Fig. 8. 3 tacks, with `Upwind Sailing` in between.

time was invested into correct timing during tack and jibe. In the beginning we had several problems that resulted in inappropriate sail tuning during or right after manoeuvres. This sometimes even resulted in the boat sailing backwards.

Figure 9 shows how the boat follows a trajectory that was calculated by the path planner. The path planner's calculation was started at the bottom left where the dashed line begins. After the calculation was completed, the boat jibes and sets the new course. After a few seconds the drift compensation

starts to work and the boat's real course approaches the desired trajectory.

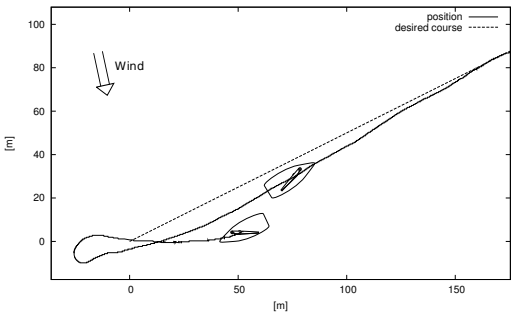


Fig. 9. GPS plot of the controller following a desired trajectory calculated by the path planner

V. Navigation Strategy

A. Principle of motion planning for a sailing boat

Due to the nature of every sailing vessel, AVALON's manoeuvrability is limited in relation to wind direction. Additionally, due to the non-holonomic properties of a sailing vessel, it is not possible to change the heading on the spot immediately. The trajectory tracker (see section III-B) has to keep that in mind and operate in a way that allows a certain space to make heading changes. For the implementation of a navigation algorithm, these constraints imply that using accurate wind information for every calculation is essential.

B. Literature Research

Path planning in the naval industry is not a new topic. For many years, commercial ships have used weather forecasts to generate an optimal path with respect to speed or bypassing bad weather situations. For sailing ships, commercial solutions by companies like B & G or NORTHSTAR exist, but do not offer any possibility to be integrated into our embedded system. Looking at different navigation and routing strategies, two fundamentally different ideas emerge: Roland Stelzer [3], who has published his approach also for implementation in an autonomous sailing boat, projects the speed vector on the target direction and tries to find a sailable heading that maximises that projected speed vector. Although in small scale it might find a good and fast path, this approach is not sufficient when it comes to larger search areas with obstacles and additional inputs like weather forecasts.

The other approach is to use grid based path planning algorithms that allow us to generate optimal solutions considering the path from start to goal and enable the integration of specific constraints (see section V-A).

Breadth first method and *Depth first* are both based on a very simple principle and serve as archetypes for many important algorithms [13], but may result in very long running time. *Dijkstra* on the other hand is a directed search [14] and guarantees optimal results. A* is a modified version of *Dijkstra* that also considers the heuristic distance estimate of the path from current position to the destination [15], which makes it more directed than *Dijkstra*. One of the problems of grid-based planning algorithms is that the angles are artificially constrained. The Theta*-Algorithm [16] deals with that handicap and produces more realistic looking paths than A* or even smoothed A*. A further extension of A* is D* [17], which incorporates dynamic changes (i. e. moving obstacles) along the path and does not require to replan the whole path from current position to destination anymore.

The proposed approach uses a simple A* algorithm which is modified to comply with all the sailing constraints. It must also be noted that there is few computation time constraints for a sailing boat crossing the Atlantic ocean. Even if path planning requires a handful of minutes, this is still an acceptable behaviour.

C. Planning Strategy

Considering the distance of about 4'200 nautical miles AVALON has to sail on its journey across the Atlantic Ocean and its slow speed of about 4 to 5 knots[1], it is essential to distinguish between a global and a local planning. Global planning will be done by fixing way-points across the Atlantic based on expert knowledge, weather and current analysis. The local planner will then plan from current position to the next global way-point using wind data from the on board sensor. The planning algorithm will be triggered and controlled by a trajectory tracker (section III-B), which knows exactly towards which way-point AVALON is sailing at the moment.

D. Interfaces with the autonomous sailor

An important part of the software structure is the interface between the navigation program and the sailor (see section IV). The main reason why we decided to pass *headings* instead of way-points from the trajectory tracker to the sailor, was that the navigator calculates an optimal path with respect to path duration for specific angles to the wind. Drift that pushes the vessel off the trajectory will be handled by the sailor, a new calculation will only take place after the boat has a certain deviation from the set trajectory (see section III-B).

a) Wind data: The wind the local path planner uses is an average of the measurements over the last 90 seconds. The sailor will then follow the received desired heading using a much more accurate wind direction that was averaged only over a period of 5 seconds (see section IV). Figure 10 shows a comparison of the two different wind sets.

E. Planning algorithm

As explained earlier, *Dijkstra*'s graph search is the basic algorithm used to perform the path search. Starting at an initial grid-point, the algorithm expands over the whole field until the final grid-point has been reached. Every evaluated grid-point will be assigned a certain cost, a time estimate of how long it will take the boat to sail from the initial grid-point to the point that is currently being evaluated. The final optimal path

[1] 1 knot is 1 nautical mile per hour

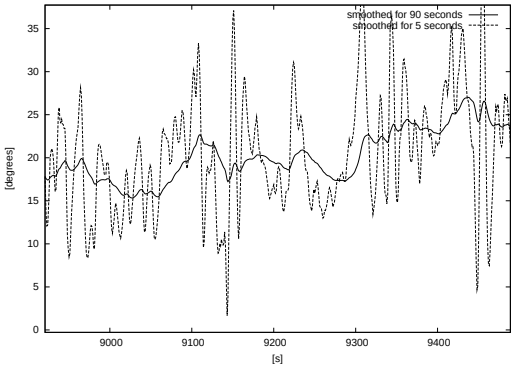


Fig. 10. Differently filtered wind information

is then determined by always following the neighbor with the lowest costs from the final to the initial grid-point.

To be able to allocate costs for ship movements, i. e. moving from one grid-point to a neighbor, we work in 3 dimensions, with x and y for the geographic position and θ for the heading of the boat at that position. To get sailable and reasonable results, the following costs have been implemented additionally:

- Sailing-Time from start node
- Estimated sailing time to the end node (heuristic)
- Cost for heading changes
- Cost for manoeuvres (tack and jibe)
- Cost for offset from the straight connection between start and end node (tunnel cost).

1) Grid compatible coordinate system: To represent GPS points on a map, a coordinate transformation is necessary. Since we plan only for short distances ahead of us (local planning), we can use a very simplified transformation into meter coordinates

$$x = R_E \cdot \cos(\text{lat}) \cdot \frac{\pi}{180\,deg} \cdot \text{lon}$$
$$y = R_E \cdot \frac{\pi}{180\,deg} \cdot \text{lat} \tag{4}$$

where R_E is the earth radius, lon and lat the GPS coordinates in degrees. This transformation assumes a flat water surface [3]. In order to initialise the local map as small as possible, the algorithm places the origin as close as possible to either start or target coordinate.

2) Polar diagram: In order to calculate the boat speed for a specific angle to the wind, a polar diagram[2] of AVALON is needed. Since creating a detailed diagram for our boat would require constant wind and wave conditions over a long time, we created a polar diagram based on existing diagrams for bigger vessels. Testing has shown that we get reasonable results by restricting the sailable *angle to the wind* to the area between 45^o and 160^o (see figure 6).

3) Heading: In order to limit the size of the 3D grid used in this paper, we use a discretisation of the possible boat heading into 16 possible headings. In usual 2D grid-based path planning, transition from one cell to its 8 neighbours is considered. This results in only 8 possible changes of orientation. By considering a neighbourhood of 24 cells (8 cells around the starting cell, plus 16 cells one cell further), we can reach 16 different heading changes in one transition. This leads to smoother paths and smaller number of way-points on the final path.

We further reduce the computational complexity of the planner by considering that the maximum change of heading in one transition is 90^o. This restriction is consistent with the experience of human skippers.

[2]In boating term, the polar diagram gives a boat velocity as a function of its heading to the wind and the wind velocity

F. Cost Factors

1) General Costs: The general aim of this algorithm is to find the shortest path for a sailing vessel to go from point A to point B. The general cost we give for every movement is therefore *sailing time to the next node*, calculated by equation 5. The speed solely depends on the angle to the wind and the wind speed.

$$\text{sailing time} = \frac{\text{distance to next node}}{\text{speed}} \tag{5}$$

2) Heuristic Estimate: The heuristic estimate is used to evaluate the remaining time to the goal. According to [14], the only constraint on the heuristic is that it has to underestimate the real cost of the remaining path. For simplicity sake, the heuristic returns the time it would take to sail to the goal at an exaggerated speed of 8 knots. More complicated options have been considered, but none of them provided well defined behaviour in all situations.

3) Turning Cost: To generate paths as smooth as possible, the algorithm places costs C_{new} for heading changes, using a simple linear factor f in equation

$$C_{\text{new}} = f \cdot |\theta_{\text{curr}} - \theta_{\text{new}}| \tag{6}$$

where θ_{curr} and θ_{new} are the respective headings.

4) Cost for Tack or Jibe: By giving a certain cost for tack or jibe, we can indirectly control the number of manoeuvres AVALON sails. The principle is very simple, we check if the sign of $(\theta_{\text{curr}} - \text{winddirection})$ and $(\theta_{\text{next}} - \text{winddirection})$ is the same. If it is different, an additional cost will be added to the total cost.

5) Tunnel cost: In order to favour paths that stay closer to the straight line path, we allocate costs increasing with the distance from the direct connection of start and target node. It has proved best to increase the cost only little for the areas in the middle of the tunnel but increase it abruptly when closing in on the borders.

6) Obstacle avoidance: To avoid islands and coastal regions, the algorithm gives extremely high costs for passing prohibited terrain. To locate these areas, it parses information from a map.

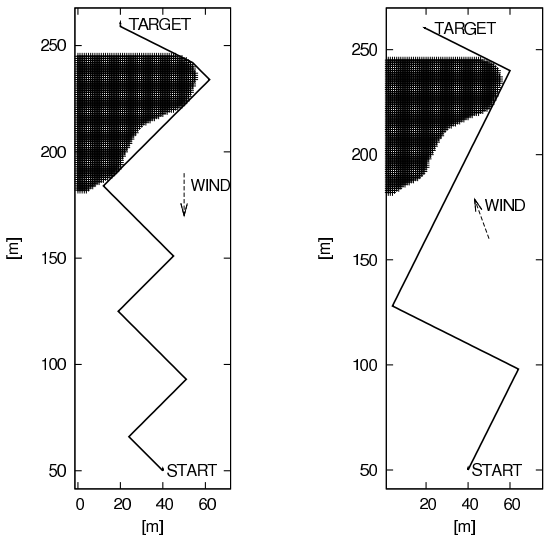
To avoid commercial ships on the Atlantic, AVALON can receive AIS signals from most ships travelling over the Atlantic. Knowing ship speed, direction and current position, our algorithm will place no-go areas around potential collision locations to make sure to bypass them.

G. Tests and Discussion

1) Testing environment: Most tests and parameter optimization was done by visualising the generated path on screen and discussing its quality. Figure 11 shows two calculated path examples, both having to pass an obstacle and dealing with difficult wind direction like upwind and downwind. Analysis has shown that the addition of a tunnel cost is redundant since the heuristic estimate also favours paths that are close to the direct connection of *start* and *target* node. Although adding an artificial tunnel cost might not always result in optimal path solutions, we used the tunnel for safety reasons to make sure that the vessel never sails beyond predefined boundaries.

Another important and not yet solved problem is the final heading at the target node that has to be known in order for *Dijkstra* to start the search. At the moment the algorithm chooses a final heading that is as close as possible to the direct heading of a straight line from start to destination but always differs at minimum 45^o from the wind direction. This usually produces good results but it does not take obstacles into account which means that sometimes an additional, unnecessary turn is produced.

In practice, this issue will be solved by starting planning a new path before completing the execution of the current one.



(a) Upwind (b) Downwind

Fig. 11. Path examples for upwind and downwind sailing with obstacles

VI. Conclusion and Outlook

The proposed navigation and control software has been successfully implemented in our autonomous sailboat AVALON. Several short-run tests both on Swiss lakes and on the Atlantic Ocean have been carried out and have shown that the described control and navigation system work. AVALON was exposed to wind conditions ranging from almost 0 to 30 knots. The experimentally found control parameters have proven to be well defined, keeping the vessel on course even while sailing in rough sea states.

In very low wind speeds it was seen that the boat is able to sail much better by itself than steered by us via remote control, because it is always aware of the exact wind direction, even when human sailors hardly notice any wind at all.

Fig. 12. Avalon during testing in high winds

However, especially in reference to a possible crossing of the Atlantic Ocean, several tests over a bigger period of time as well as longer distances have to be performed in a next stage.

By processing received AIS-signals, AVALON is able to identify other ships in close proximity and therefore prevent collisions. However, small boats that don't send position information cannot be recognized at this stage and further research has to be done in identifying obstacles visually.

Implementing a conventional and proved path planner like Dijkstra's for our navigation system has proved well. Visually and analytically evaluated results show that we achieve better solutions by taking the whole path into account than by always optimising the heading at every current position. However, by adding additional costs (like tunnel cost), optimal path solutions are not always guaranteed.

Our goal for future work is to further optimise the software, especially regarding energy consumption. For example a hysteresis could be added to the sail tuner so that the sail does not constantly adapt to minor changes in wind direction and speed. A further area of improvement could be in making use of weather forecasts in the global navigation. For our boat this does not add much value, because it is too slow to react to forecast weather changes.

References

[1] N. Cruz and J. Alves, "Autonomous sailboats: an emerging technology for ocean sampling and surveillance."

[2] Y. Briere, "IBOAT: An autonomous robot for long-term offshore operation," in *Electrotechnical Conference, 2008. MELECON 2008. The 14th IEEE Mediterranean*, 2008, pp. 323–329.

[3] R. Stelzer and T. Pröll, "Autonomous sailboat navigation for short course racing," *Robotics and Autonomous Systems*, vol. 56, no. 7, pp. 604–614, 2008.

[4] J. Alves and N. Cruz, "FASt-An autonomous sailing platform for oceanographic missions."

[5] C. Sauze and M. Neal, "Design Considerations for Sailing Robots Performing Long Term Autonomous Oceanography," in *Proceedings of The International Robotic Sailing Conference, 23rd-24th May*, 2008, pp. 21–29.

[6] May 2009, http://www.harborwingtech.com.
[7] L. Giger, S. Wismer, S. Böhl, G.-A. Büsser, H. Erckens, J. Weber, P. Moser, P. Schwizer, C. Pradalier, and R. Siegwart, "Design and Construction of the Autonomous Sailing Vessel Avalon," in *Proceedings of The 2nd International Robotic Sailing Conference, 10th July*, 2009, pp. 17–22.
[8] 2009, http://www.digitallogic.com.
[9] P. Corke, P. Sikka, J. Roberts, and E. Duff, "Ddx: A distributed software architecture for robotic systems," in *Proc. Australian Conf. Robotics and Automation*, Canberra, December 2004.
[10] D. Hochseeportverband *et al.*, *Seemannschaft*. Delius, Klasing, 2008.
[11] R. Stelzer, T. Proll, and R. John, "Fuzzy logic control system for autonomous sailboats," July 2007, pp. 1–6.
[12] G. Elkaim and R. Kelbley, "Station Keeping and Segmented Trajectory Control of a Wind-Propelled Autonomous Catamaran," in *Proceedings of the Conference on Decision and Control*.
[13] C. Leiserson, R. Rivest, T. Cormen, and C. Stein, *Introduction to algorithms*. MIT press, 2001.
[14] S. LaValle, *Planning algorithms*. Cambridge University Press, 2006.
[15] P. Hart, N. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *Systems Science and Cybernetics, IEEE Transactions on*, vol. 4, Issue 2, pp. 100–107, July 1968.
[16] A. Nash, K. Daniel, S. Koenig, and A. Felner, "Theta^*: Any-Angle Path Planning on Grids," in *Proceedings of the National Conference on Artificial Intelligence*, vol. 22, no. 2. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999, 2007, p. 1177.
[17] A. Stentz, "The Focussed D^* Algorithm for Real-Time Replanning," in *International Joint Conference on Artificial Intelligence*, vol. 14. LAWRENCE ERLBAUM ASSOCIATES LTD, 1995, pp. 1652–1659.