The debut of AEOLUS, the Autonomous model Sailboat of ETH Zurich

Marco Tranzatto[1], Alex Liniger[2], Sergio Grammatico[2], Alberto Landi[1]

***Abstract*—Autonomous sailboats are good candidates for long term oceanic surveys, since they use wind power as their main mean of propulsion. Controlling a sailboat, however, is typically not an easy task due to the high variability in the wind and the side drift of the boat. In this paper, we describe how we design and set up the control architecture that allows *Aeolus*, the autonomous model sailboat of ETH Zurich, to sail upwind and execute fast and smooth *tack* maneuvers. Different controllers to actuate the rudder have been implemented and validated for both the upwind sailing and the tack maneuver. We present experimental results obtained during several autonomous sailing tests at the lake Zurich.**

***Keywords*—autonomous marine vehicle, autonomous model sailboat, upwind sailing, tack maneuver, embedded control.**

I. INTRODUCTION

Since a sailboat cannot move along every direction at every time, the wind direction must be taken into account to plan an achievable navigation course. In fact, there is a certain direction relative to the wind where a sailboat cannot navigate at all, which is indeed called "no-sail zone" [1].

In this paper, we consider two main tasks for autonomous model sailboats: sailing with a fixed heading and executing a fast and smooth *tack* maneuver. The fixed heading can be either a constant compass course or a fixed heading with respect to the measured wind. The tack maneuver is a typical sailing action which allows the boat to turn its bow into the wind through the "no-sail zone", so that the direction from which the wind blows changes from one side to the other. To make the model sailboat sailing, we can hence combine three possible actions: sail on *starboard* haul, sail on *port* haul and *tack*. Sailing on starboard (port) haul means that as a result of the course of the sail boat, the wind blows from the starboard (port) part of the vessel.

Many research groups have been developing model sailboats in recent years, for instance the UBC team from the University of British Columbia [2], the TRST team from the Tufts University [3], the OLIN robotic sailing team [4], and the FAST sailing boat team from the University of Porto [5]–[7]. Most studies on sailing boats focus on decoupling the control system of the rudder from the one of the sails. Among others, [8] explains how to identify a linear second order model for the steering dynamics of a model sailboat, and to design a PI (proportional-integral) feedback law to control the rudder.

The authors are with the: [1]Department of Information Engineering, University of Pisa, Italy; [2]Automatic Control Laboratory, ETH Zurich, Switzerland. E-mail addresses: marco.tranzatto@gmail.com, {liniger, grammatico}@control.ee.ethz.ch, alberto.landi@dsea.unipi.it.

Fig. 1. Aeolus sailing at the lake Zurich.

Therein, the PI controller is used only to track a desired course angle while sailing, while another controller is in charge to regulate the sails. Since a mathematical model of the dynamics of a model sailboat is typically not easy to derive, some works use fuzzy logic (namely, empirical rule-based logic) to control both the rudder and the sails. The main idea is to exploit the knowledge of the "helmsman", which can be expressed in terms of a rule-based control system [9]–[11]. In this paper, we describe the autonomous model sailboat under development in the Automatic Control Laboratory at ETH Zurich, called *Aeolus*, depicted in Figure 1. The paper is divided in four main sections: hardware and software setup, modeling, course navigation and tacking. The first section briefly explains the hardware and embedded software setup of Aeolus. The second presents a simple yet useful method to identify a linear model of the yaw dynamics, that will be used to tune the controllers for the rudder. The third section shows two different controllers used to track a desired heading while sailing upwind. The last section analyzes three different approaches to execute the tack maneuver; experimental results from several tests at the lake Zurich are also presented.

II. AEOLUS SETUP

We started with an international one-meter RC model sailboat, and installed specialized electronics.

Our hardware is mainly composed by an autopilot control unit and a weather station. Since sensing the wind is clearly fundamental for a sailboat, we use a dedicated device for this objective. The weather station we employ is the AIRMAR WS-200WX [12], which we have mounted above the bow. The main information it provides, allowing the boat to sail, is the apparent wind (direction and speed), estimated true wind (direction and speed) and GPS position. All these values

978-1-4799-8736-8/15/$31.00 ©2015 IEEE

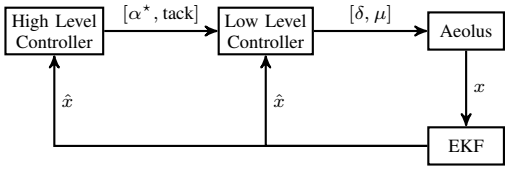

Fig. 2. Software architecture.

are sampled every 200 ms. Our autopilot is the PIXHAWK board [13], an independent, open-source, open-hardware microcontroller. It integrates a flight management unit and an input/output module in only one component, and is able to send and receive data over a radio transmission link. The PIXHAWK board provides a POSIX-compatible real time operating system, where many applications can run in parallel. Its input/output model is equipped with many sensors, such as IMUs (Inertial Measurement Units): accelerometer, gyroscope, magnetometer. The main software architecture is shown in Figure 2. To estimate the state of Aeolus (position, velocity, attitude, etc) we rely on an indirect extended Kalman filter (EKF) readily available in the PIXHAWK firmware. It has a loosely coupled compensation to integrate inertial measurements from IMUs and GPS positions, as explained in [14]. This filter uses a general kinematic model, so no specific tuning based parameters of Aeolus are required. We have structured the onboard software in a hierarchical way: a high level application sends commands and reference actions to a low level controller, which, reading the information from the sensors, computes the input actions (sail and rudder angles) to follow the desired reference. In this way, the software structure emulates the typical task division between navigator/tactician and helmsman on real sailboats. Additionally, we decouple the control of the ruder and the sail, similar as done in [8], [11].

III. MODELING AND IDENTIFICATION

In order to tune and analyze the closed-loop behavior induced by the controllers as explained in the next sections, a dynamical model of the sailboat is required. In [15] it is shown how to derive a nonlinear four-degree-of-freedom (4-DOF) dynamic model for a sailing yacht, including the roll dynamics, using the notation introduced by [16]. This procedure can in principle be applied to our model sailboat, but then the modeling phase must be followed by a parameter identification phase. Since identifying all the parameters of the nonlinear model is typically challenging, we decided to identify only the yaw and yaw rate dynamics of our sailboat using a linear model, as suggested in [8], [17]. In [8] a continuous-time transfer function, from the rudder angle to the yaw rate output, is identified, which mathematically describes the input/output behavior of the system, in this is the behavior from the rudder input to the yaw rate. Once this model has been identified, a pole is added at the origin, so that a transfer function from the rudder to the yaw angle is also directly obtained. In [17] a discrete-time trasfer function, again for the yaw rate dynamics relative to the rudder angle input, is identified instead. Starting from these two works, we next describe a possible identification phase that can be executed by a "helmsman", using the remote controller to command the model sailboat. The boat sails upwind with a constant velocity, with the rudder being in the middle position; when the vessel has enough longitudinal speed, a step command on the rudder is given, that is, a strong steering input. This step command produces a fast variation in the yaw rate and in the yaw angle of the boat, that are recorded and transmitted via the radio link to a PC located on the shore.

Using the data from the above identification phase, we identify a state space description of the yaw and yaw rate dynamics. Specifically, we define the state vector at time k as

$$x_k = [\omega_k, \psi_k]^\top \tag{1}$$

where ω is the yaw rate and ψ is the yaw, or heading, angle. We assume that the system dynamics can be described by the discrete-time linear system

$$x_{k+1} = Ax_k + B\delta_k \tag{2}$$

Where x_k is the state of the boat and δ is the rudder command. A similar state space description has been shown in [18], where the unknown parameters are adaptively estimated. We use a slightly different approach, and estimate each parameter of the model without an adaptive procedure. The two-state-space model (2), whose matrices are

$$A = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix},\ B = \begin{bmatrix} b_1 \\ b_2 \end{bmatrix} \tag{3}$$

We identify two slightly different types of models: the *grey* and the *black* type models. In the *grey* type we use the knowledge about the physical meaning of the two variables in the state vector. We assume that the yaw rate ω (at time $k+1$) depends only on the previous yaw rate (at time k) and on the rudder command, thus in (3) we impose $a_{12} = 0$. Then, we assume that the yaw angle ψ (at time $k+1$) is the integral of the yaw rate and it is not directly affected by the rudder command. Therefore, in (3) we impose $a_{21} = \Delta_t$ (time interval between the time instants k and $k+1$), $a_{22} = 1$ and $b_2 = 0$. In this way, we are implicitly using the forward Euler method to integrate the yaw rate signal. It then follows that in the *grey* model we have to identify only the parameters a_{11} and b_1. In the *black* type model we do not make any prior assumption, thus we identify the full matrices A and B in (3).

By collecting the yaw rate, the yaw and the rudder signals experimentally, we follow a least square error procedure to compute the matrices A and B. Numerically, the following *black* type model has been derived:

$$A = \begin{bmatrix} 0.7078 & -0.0124 \\ 0.0744 & 0.9986 \end{bmatrix},\ B = \begin{bmatrix} -0.3089 \\ -0.0228 \end{bmatrix}, \tag{4}$$

where the yaw rate and the yaw angle variables are meant in radians. We have validated the model in (4) using data collected in different navigation tests at the lake Zurich, and one example of validation in Figure 3. It is important to point out that typically there are different wind and sea (wave) conditions between the day when a model is identified and

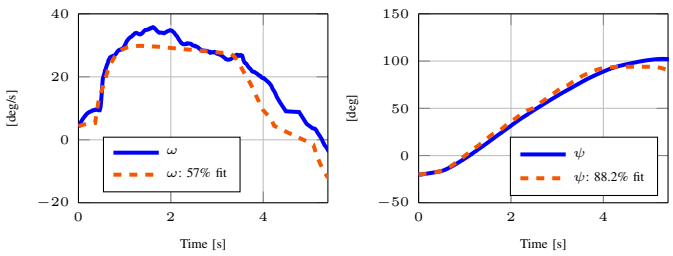

Fig. 3. Output cross validation: The solid line is the real measurement; the dashed line is the output predicted by the model.

the days when it is validated. We notice that despite the fit percentage for the yaw rate is not very high (57%), we achieve a quite good fit for the yaw response (88.2%).

IV. Collecting and Filtering Data

The main information used to sail are provided by the extended Kalman filter and by the weather station. The extended Kalman filter supplies the heading angle, that is, the compass direction where the bow is pointing to. The weather station supplies the apparent and estimated true wind (direction and speed), as well as the GPS position (latitude, longitude and altitude) and the GPS course over ground. The course over ground is the direction over the ground the vehicle is currently moving in; it can be different from the heading, if there is drift (caused by either the wind or the waves), see Figure 4. We refer to the heading angle as ψ, to the estimated true-wind-direction angle as σ and to the course-over-ground angle as χ, see Figures 4, 5.

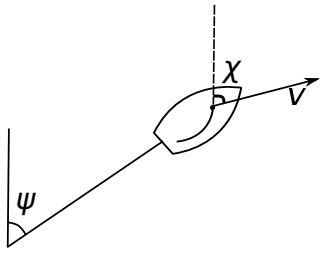

Fig. 4. Representation of the heading angle ψ (where the bow is pointing to) and the course-over-ground angle χ (where the boat is going to). v is the velocity vector of the vessel.

A. Filtering the row data

In order not to follow too much high-frequency wind shifts, we design two moving average filters for the raw measurements of the direction and speed of the estimated true wind. The wind direction takes values between $-180°$ and $180°$, where $0°$ is the geographic North, $+90°$ is the East, etc. Note that the wind direction discontinuity at the beginning/end of the scale requires special processing to compute a valid mean value. We employ the single-pass procedure developed by Mitsuta in [19] to compute a mean wind direction. Figure 6 shows the effect of these filters on the raw wind direction measurement.

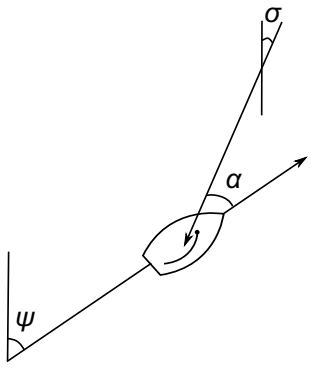

Fig. 5. Illustration of the relative heading angle α: ψ is the heading of the boat and σ defines the direction of the true wind.

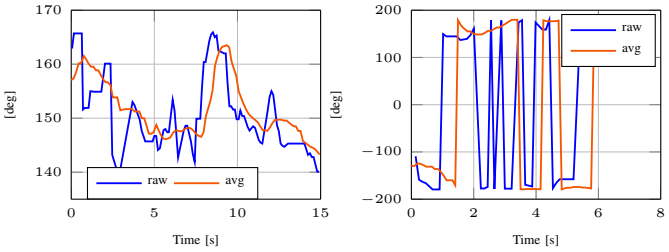

Fig. 6. True wind direction from the weather station (blue) and averaged value from the filter (orange). On the left an example of the delay introduced by the moving average filter. On the right the mean value of the wind direction when the raw measurement switches between $-180°$ and $180°$, obtained by using the Mitsuta mean.

B. Filtering the relative heading

We now consider the heading angle α with respect to the wind direction, which reads as

$$\alpha = \psi - \sigma, \tag{5}$$

as illustrated in Figure 5. Setting a reference value $\alpha^\star$ for this angle, the high-level controller tells Aeolus the desired orientation relative to the wind. For example, if $\alpha^\star = 45°$ Aeolus should sail upwind (that is, between "close hauled" and "beam reach"), if $\alpha^\star = 90°$ Aeolus should sail at beam reach, etc. The sign of $\alpha^\star$ determines the haul: a positive value corresponds to starboard haul (the wind is blowing from the right side of the vessel), meanwhile a negative value corresponds to port haul (the wind is blowing from the left side of the vessel).

In order to overcome drift due to currents and waves, it is possible to use the course over ground value χ, provided by the GPS signal, instead of the heading ψ. Unfortunately, the GPS signal sometimes drops, even for long periods of time, up to 10 seconds. Therefore, here we define two angles to take into account both the drift compensation and the updated measurements:

$$\alpha_\psi = \psi - \sigma \tag{6}$$

$$\alpha_\chi = \chi - \sigma. \tag{7}$$

Our estimated α angle is computed as a convex combination between these two angles as follows:

$$\alpha = (1-\lambda)\alpha_\chi + \lambda\alpha_\psi \tag{8}$$

where λ is a function of the last time when the course-over-ground signal has been updated, that takes values in $[0, 1]$. Specifically, the more time has elapsed without a course-over-ground update, the more λ tends to 1; when a new course over ground is received, λ is then reset to 0. Since this design can cause discontinuities in the α estimate when a new course over ground is received, we employ a moving average filter on the α value to smooth out the actual estimate.

V. Tracking a constant heading

A low level regulator is in charge of controlling both the sails and the rudder. The reference heading angle for the low regulator is $\alpha^\star$, and is set by the high-level controller. Using the equation (5), we can specify either a constant compass course or a constant heading to the wind as reference. The first case is simply obtained by setting $\sigma = 0$; the second case uses a more general formulation where σ is provided by the weather station.

Here we design two rudder controllers: a standard proportional (P) controller and a more sophisticated nonlinear one (NL). Let use normalize the rudder command δ to be in the range $[-1, 1]$. Given the reference angle $\alpha^\star$ and the current estimate α, the heading error reads as $e := \alpha^\star - \alpha$, and the P rudder controller sets the rudder command δ as

$$\delta = \delta_{\mathrm{P}}(e) = k_{\mathrm{p}}\, e,$$

for some $k_p > 0$.

Instead, the NL controller defines a nonlinear gain $k(e)$ as

$$k(e) = \frac{k_{\mathrm{p}}}{1 + c_{\mathrm{p}}|e|} \tag{9}$$

for some $k_{\mathrm{p}}, c_{\mathrm{p}} > 0$, and hence sets the rudder command to

$$\delta = \delta_{\mathrm{NL}}(e) = k(e)\, e.$$

The main idea of this nonlinear controller comes from [20]. The controller (9) acts as a proportional gain when the error e is small, but its behavior changes when the error is large. The c_{p} constant can be in fact used to tune the control action when the error is large; here we tune (9) so that the larger c_{p} is, the smaller the gain $k(e)$ and so the rudder action. This behavior is used also in the next section to execute a special *tack* maneuver.

These two controllers are able to track a reference angle. Using the identified numeric model in (4), we can study the stability of the closed-loop system, both in the linear and the nonlinear case. After several experimental tests, we can tune k_{p} for both stability and tracking purposes to the value $k_{\mathrm{p}} = 0.35$. Based on the results from the P controller, we can tune the NL one in (9) with $k_{\mathrm{p}} = 0.35$, $c_{\mathrm{p}} = 0.35$.

As for the sail control, we use a simple rule-based law: the more Aeolus is sailing opposite to the wind direction, that is, the smaller α is, the more we close the sails.

VI. Tacking maneuvers

When sailing upwind, a tack maneuver allows the boat to change haul without getting stuck against the wind. Since during the maneuver the boat crosses the no-sail zone, it should be executed in the fastest and smoothest possible way, in order not to get stuck against the wind. Three possible ways of carrying out a tack are here developed and tested: the *implicit*, the *dedicated*, and the *optimal* one. The last two maneuvers require a switch from the rudder controller used during upwind sailing to the controller developed for the tack maneuver.

A. Implicit tack maneuver

The implicit tack is done if the high-level controller simply changes the reference $\alpha^\star$ of the low level heading controller, without sending a "tack-now" command. We have noticed that the implicit tack done using the P controller produces larger overshoot, compared to the NL implicit one. Two main reasons cause undesired overshoots: a strong initial rudder command and the delay introduced by the average filters. When the reference is changed, for example from $-45°$ to $45°$, the P controller sets a more aggressive rudder command, compared to the NL controller, in which the c_{p} coefficient is actually tuned to be less aggressive when the error is large. Moreover, the moving average filters used to filter the raw measurements introduce a delay of about 2 s, thus if the tack maneuver is executed too fast, for example by using an aggressive rudder command, the delay introduced by the average filters induces an overshoot in the heading angle response. To avoid this overshoot, a less aggressive implicit tack maneuver should be executed. Summarizing, our NL controller can both control the rudder to sail upwind at constant reference and execute a smooth tack maneuver without significant overshoot.

B. Dedicated tack maneuvers

On the other hand, the dedicated and the optimal tack maneuvers are executed by the low-level controller when the high-level layer sends the tack-now command and updates $\alpha^\star$. Therefore, the low-level application is now aware that a tack must be carried out, and can hence perform special actions for the maneuver. From our field experience, we mention three critical actions to be taken into account when tacking. First, the window size of two average filters (on the wind direction σ and on the α angle) is set to 1 sample only; second, the α angle is computed using only α_ψ in (6), not by using α_χ in (8); third, a specialized tack regulator (either the dedicated or the optimal) takes control of the rudder during the maneuver. The first action overcomes the undesired delay typical of the implicit tack caused by the filters. The second action is needed because during a fast tack maneuver, it is likely to loose the updated course-over-ground measurement. In our implementation, a specialized tack regulator (either the dedicated or the optimal) controls the rudder until the tack is considered completed, that is until the error signal is within specified bounds for about 1 s. Namely, we then switch back to the course-navigation controller when the actual heading is close enough to the new reference.

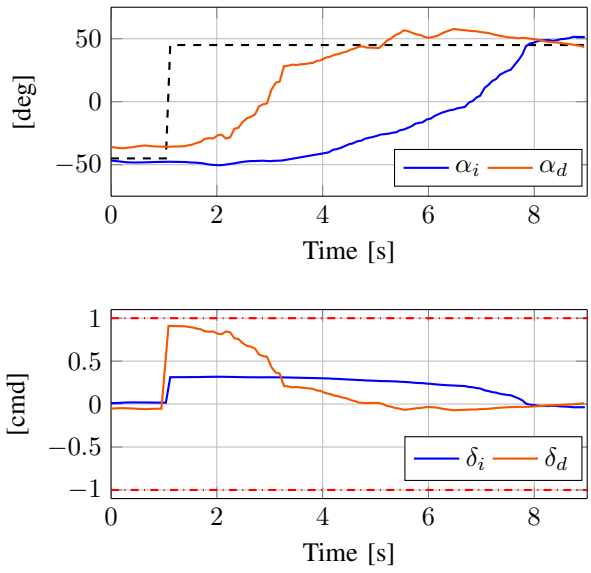

Fig. 7. Lake test: Comparison between implicit and dedicated tacks from port to starboard haul. The upper plot shows the reference $\alpha^\star$ (heading with respect to the wind direction) in dashed black, the implicit (solid blue) and the dedicated (solid orange) tack responses. The lower plot shows the rudder limits (dotted red), the implicit (solid blue) and the dedicated (solid orange) rudder commands. The implicit tack takes $\sim$7 s, while the dedicated one takes $\sim$3.7 s only.

Let us now discuss in detail our dedicated and optimal tack controllers. The dedicated controller is just the nonlinear one in (9), with $k_\mathrm{p} = 0.7366$ and $c_\mathrm{p} = 0.1$ tuned to obtain a more aggressive behavior and execute a faster, as well as smooth, tack. With our numerical choice, when the error is $|e| = 90°$ we have $|\delta| = 1$, thus the rudder is working at its saturation boundary. Figure 7 shows a comparison between the implicit and the dedicated tack controllers, during a tack from port to starboard haul. We point out that this more aggressive regulator does not produce overshoot thanks to the special actions explained above.

Linear quadratic optimal tack controller

By optimal tack controller we mean a Linear Quadratic Regulator (LQR), where "optimality" is with respect to an infinite-horizon cost function. The LQR is a state-feedback controller, meaning that the control action is a static feedback law of the state x as follows: $\delta = \delta(x) = K_\mathrm{LQR} x$.

To exploit the model derived in the modeling section, we make the assumption that the true wind direction does not change during the tack maneuver, which is a practically reasonable assumption if the maneuver is fast enough. In this way, a tack maneuver can be just seen as a change in the heading angle ψ. For example, a tack from port ($\alpha^\star = -45°$) to starboard haul ($\alpha^\star = 45°$), can been seen in two equivalent ways: (a) we require a change in the α angle of $\Delta\alpha = 90°$; (b) we require a change in the heading angle ψ of $\Delta\psi = 90°$, i.e., $\Delta\psi = \Delta\alpha$, based on (5) assuming a constant wind-direction angle σ. Thus, tacking results in steering the state of the system in (1) from the initial value $x_\mathrm{i} = [\omega_\mathrm{i}, \Delta\psi]^\top$ to the final value $x_\mathrm{f} = 0$, where the latter consists in achieving the desired $\alpha^\star$ with zero yaw rate.

Let us rewrite the model in (1) in state-space form with

$$\hat{\delta}_k := \delta_k - \delta_{k+1} \tag{10}$$

$$\hat{x}_k := [\omega_k, \psi_k, \delta_{k-1}]^\top, \tag{11}$$

where $\hat{\delta}$ is the new control input and $\hat{x} \in \mathbb{R}^3$ is the extended state. Namely, the extended state at time k, $\hat{x}_k$, contains the yaw rate and yaw angle at time k and the rudder command δ_{k-1} injected into the system at the previous step $k-1$. The input $\hat{\delta}$ is the difference between the actual rudder command at time k, and the previous one; in other words, the real rudder command δ provided at the time k is then $\delta_k = \delta_{k-1} + \hat{\delta}_k$.

The state space matrices $\hat{A}$, $\hat{B}$ corresponding to the extended state dynamics become

$$\hat{A} := \begin{bmatrix} A & B \\ 0 & I \end{bmatrix}, \; \hat{B} := \begin{bmatrix} B \\ I \end{bmatrix}. \tag{12}$$

This state-vector extension allows us to define the following cost function for the LQR strategy:

$$\sum_{k=0}^{\infty} \|\hat{x}_k\|_Q^2 + r\hat{\delta}_k^2, \tag{13}$$

where the matrix $Q \succ 0$ and $r > 0$ are design choices. The LQR gain K_LQR is computed such that the state-feedback law $\hat{\delta}(\hat{x}) = K_\mathrm{LQR}\hat{x}$ minimizes the cost function (13), subject to the unconstrained discrete-time dynamics $\hat{x}_{k+1} = \hat{A}\hat{x}_k + \hat{B}\hat{\delta}_k$. The main reason to extend the state vector as in (10) is to assign a cost penalty to the control action δ as well as to the control variation $\hat{\delta}$.

Once we obtaine the matrices of the extended model in (12) using the values in (4), we can tune Q and r, both via numerical simulations and via field tests. Let us hence choose the numerical values

$$Q = \mathrm{diag}(1, 3, 1), \; r = 35.$$

A comparison between the dedicated and the optimal tack is depicted in Figure 8. The optimal controller is able to complete the tack maneuver about 0.5 s faster than the dedicated one. Although in the first phase (1 to 2 s) the rudder command from the dedicated controller was more aggressive, the rudder command from the LQR stayed longer close to the saturation than the dedicated controller in the second phase (2 to 5 s). This difference allows the optimal regulator to execute a faster maneuver, without overshoot. The path of Aeolus during an optimal tack maneuver is depicted in Figure 9.

VII. Conclusion

We have presented the hardware and software setups of Aeolus, the autonomous model sailboat of ETH Zurich. We have proposed a simple technique to identify a linear state space model of the yaw dynamics relative to rudder commands. We have designed two controllers for tracking a fixed heading angle. Furthermore, we have designed three controllers to

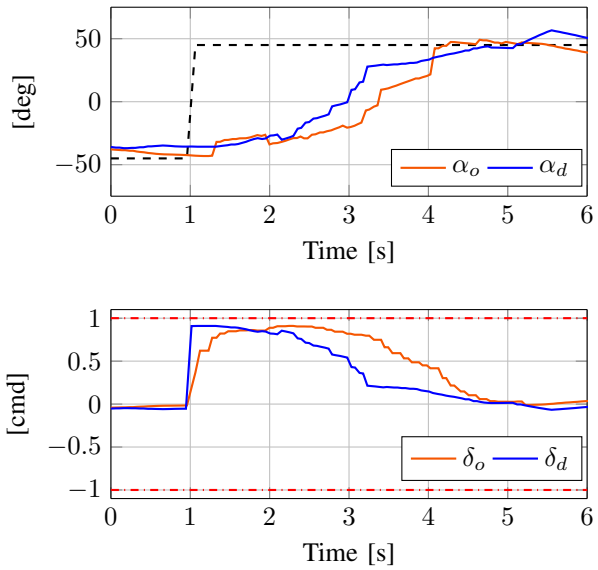

Fig. 8. Lake test: comparison between optimal and dedicated tacks from port to starboard haul. The upper plot shows the reference $\alpha^\star$, the heading with respect to the wind direction (dashed black), the optimal (solid orange) and the dedicated (solid blue) tack responses. The lower plot shows the rudder limits (dotted red), the optimal (solid orange) and the dedicated (solid blue) rudder commands. The optimal tack takes $\sim$3.2 s, while the dedicated one takes $\sim$3.7 s.

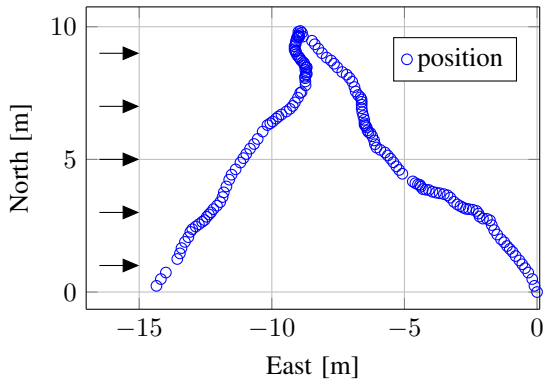

Fig. 9. Path of Aeolus (the GPS receiver is located above the bow) while executing an optimal tack. Starting from the origin position, Aeolus sails upwind at port haul. A tack is executed at (−9,10), and then Aeolus sails upwind at starboard haul. The black arrows show the mean wind direction measured by the weather station.

execute a tack maneuver. All control laws have been implemented and validated on Aeolus in several autonomous sailing tests at the lake Zurich. We have achieved experimentally fast and smooth tack maneuvers, mainly by employing special actions in filtering state measurements and computing the rudder commands. We believe that our setups, design choices and experimental evidence provide useful insights for further research in the field of autonomous sailing.

Acknowledgment

The authors would like to thank Jonas Wirz, Marcello Colombino and Dr. Henrik Hesse for their support on related project work, Prof. Roy Smith and Prof. John Lygeros for their project supervision. The authors acknowledge the KIM research grant 2013/‘14 from ETH Zurich.

References

[1] F. Plumet, C. Petres, M.-A. Romero-Ramirez, B. Gas, and S.-H. Ieng, “Toward an autonomous sailing boat,” *IEEE Journal of Oceanic Engineering* (in press), 2014.

[2] UBC SailBot. (2014) Web: `http://ubcsailbot.org`. University of British Columbia.

[3] TRST. (2014) Web: `http://sites.tufts.edu/roboticboat`. Tufts University.

[4] Olin Robotic Sailing Team. (2014) Web: `http://olinroboticsailing.com`. Franklin W. Olin College of Engineering.

[5] FAST. (2014) Web: `www.fe.up.pt/fast`. Faculty of Engineering, University of Porto.

[6] J. C. Alves and N. A. Cruz, “FASt - an autonomous sailing platform for oceanographic missions,” in *Proc. of the IEEE Oceans*, Quebec City, Canada, 2008, pp. 1–7.

[7] N. A. Cruz and J. C. Alves, “Navigation performance of an autonomous sailing robot,” in *Proc. of the IEEE Oceans*, St. John’s, Canada, 2014, pp. 1–7.

[8] ——, “Auto-heading controller for an autonomous sailboat,” in *Proc. of the IEEE Oceans*, Sydney, Australia, 2010, pp. 1–6.

[9] E. C. Yeh and J.-C. Bin, “Fuzzy control for self-steering of a sailboat,” in *Proc. of the Int. Conf. on Intelligent Control and Instrumentation*, vol. 2, Singapore, 1992, pp. 1339–1344.

[10] J. Abril, J. Salom, and O. Calvo, “Fuzzy control of a sailboat,” *International Journal of Approximate Reasoning*, vol. 16, no. 3, pp. 359–375, 1997.

[11] R. Stelzer, T. Proll, and R. I. John, “Fuzzy logic control system for autonomous sailboats,” in *Proc. of the IEEE Int. Fuzzy Systems Conference*, 2007, pp. 1–6.

[12] Airmar. (2014) Web: `www.airmartechnology.com`.

[13] Pixhawk. (2014) Web: `http://pixhawk.org/start`.

[14] D. H. Titterton and J. Weston, *Strapdown Inertial Navigation Technology*, 2004, ch. 13.7.

[15] L. Xiao and J. Jouffroy, “Modeling and nonlinear heading control for sailing yachts,” *IEEE Journal of Oceanic Engineering*, vol. 39, no. 2, pp. 256–268, 2014.

[16] T. I. Fossen, *Guidance and Control of Ocean Vehicles*, 1994.

[17] T. Emami and R. J. Hartnett, “Discrete time robust stability design of PID controllers autonomous sailing vessel application,” in *Proc. of the IEEE American Control Conference*, 2014, pp. 1993–1998.

[18] L. Xiao, T. I. Fossen, and J. Jouffroy, “Nonlinear robust heading control for sailing yachts,” in *IFAC Conf. in Maneuvering and Control of Marine Craft*, Arenzano, Italy, 2012, pp. 404–409.

[19] Y. Mori, “Evaluation of several single-pass estimators of the mean and the standard deviation of wind direction,” *Journal of climate and applied meteorology*, vol. 25, no. 10, pp. 1387–1397, 1986.

[20] A. Balestrino, A. Landi, and L. Sani, “CUK converter global control via fuzzy logic and scaling factors,” *IEEE Trans. on Industry Applications*, vol. 38, no. 2, pp. 406–413, 2002.