AI on the Ocean: the RoboSail Project

Martijn L. van Aartrijk[1] and **Claudio P. Tagliola**[2] and **Pieter W. Adriaans**[3]

Abstract. Sailing represents a new area of research within the field of applied AI. This paper gives a global overview of an ongoing broad-scoped research programme, called the RoboSail project (www.robosail.com). Part of this project is the realization of a semi-autonomous intelligent pilot and a decision support system, assisting sailors in optimizing sailboat performance. The project encompasses a wide range of various real-world AI problems. In solving these problems, we argue that our approach to *symbol grounding* by means of using *jargon* amounts to a significant reduction in the problem's complexity. The first implementation of this system has been successfully tested by some of the leading sport teams in the world of short-handed sailing.

1 INTRODUCTION

The *RoboSail Project* [16] is an new initiative to take the latest developments in the fields of Artificial Intelligence to the high seas.

The task of sailing consists of a whole range of components: first of all the physical aspects of wind, water and waves. On a more global scale, weather systems tend to continuously change, from a complete calm up to a gale attaining to hurricane force. Next to the reactive aspect of handling the rudder is the tactical notion of navigation. Usually, these tasks are handled by a complete crew; in single handed sailing, however, everything on board has to be done by only one skipper. Thus endurance, fatigue and loss of concentration become very important aspects of sailing. When there are so many different tasks to be carried out, the amount of actual steering can be as small as 15% to 20% [1]. Therefore, the benefits of having a better (i.e. more intelligent) autopilot are obvious. Research to improve autopilot systems is of utmost importance for sailing's professional teams.

The RoboSail Project's ultimate goal is to create a semi-autonomous, intelligent, computer system, which can learn to steer a sailboat optimally, in close cooperation with the human sailor onboard. We consider there to be no point, and no challenge, in trying to create a fully autonomous sailing robot: international law and the ORC[4] rules prohibit the use of completely autonomous vessels. The first would deem a robotized sailing vessel illegal, the latter would make it practically useless in official races[5].

The sport of sailing is not a trivial matter: [1, 5, 8] testify to this sufficiently. It is an activity that consists of a large number of interesting problems: it contains senso-motoric elements as well as elements of high-level reasoning. The actual steering of a sailboat can be viewed as a purely senso-motoric issue, while navigational aspects require a higher level, more cognitive approach. In between these two extremes are activities like *trimming* the sails[6]. The reason why sailing is widely regarded as 'difficult' is due to the nature of the world in which it takes place: a sailing vessel resides in a complex, dynamical environment, governed by aërodynamics and hydrodynamics, two mathematically intractable problems [5]. The best approximations are so-called Velocity Prediction Programs [11], based on various forms of finite-element analysis. Therefore, at best, only *partial* mathematical models can be used. The best way to capture the full dynamics of the world is thus to *learn* from experiences. For these and other reasons [19], it has been argued that sailing is an interesting sport from the perspective of applied AI: the combination of different skills is a major challenge to the AI community. Apart from the mechanical aspects, which represent an important issue onboard any ocean racing vessel, the scope of all the different aspects involved in sail racing are totally unlike any paradigm currently used in applied AI. Moreover, the focus of the RoboSail project is not 'just' on creating a completely autonomous system, but a semi-autonomous one that actively cooperates with its human 'colleague', to ultimately combine the best of both: the physical qualities of the human, com-

Figure 1. The *Syllogic* sailing lab

[1] RoboSail systems BV, Waterlandlaan 120, 1441 RW Purmerend, The Netherlands, email: martijn@robosail.com
[2] RoboSail systems BV, email: claudio@robosail.com
[3] RoboSail systems BV & Institute for Language, Logic and Computation (ILLC), University of Amsterdam, email: pietera@science.uva.nl

[4] Ocean Racing Committee, the international authority on (race-) sailing regulations.

[5] Actually, some people even dispute the legality of single-handed sailing. This discussion, however, is still open.

[6] Sailtrim: the choice of sails and the way they are configured

bined with the decision-making and accurate control capabilities of an intelligent computer system. To achieve this, not only decision-making should be implemented, but also a carefully designed system of two-way communication: the computer as sparring partner, or co-skipper.

The system described in this paper has been implemented and tested on our sailing lab, the *Syllogic* (Figure 1).

2 ARCHITECTURE

The starting point of the design of the X-Pilot is the *Mission Statement*:

> "The X-Pilot should be able to navigate a sailing vessel through a real-world sailing environment; it should do so autonomously though in collaboration with a human sailor, in an optimal, reliable and robust manner."

From this statement, we encounter a series of problems, some of which are classic AI problems. The first comes from the senso-motoric part of the problem: how can we map 'human feeling' unto an algorithm? Secondly, in a complex, dynamical environment like that of sailing, how can we obtain and use expert knowledge? If the world of sailing is so complex and highly unpredictable, can we even hope to find ways in which efficient learning is possible at all?

The first approach considered was one inspired by that taken in [13] and similar research: we tried to learn a complete model by means of Reinforcement Learning. First experiments however, resulted in a very slow rate of convergence, and a quick calculation showed us a sailing vessel would need several trips around the world to gather enough information to learn to sail efficiently.

As soon as we added common knowledge to the system, the rate of convergence would increase. For example, the concepts of 'left' and 'right' increased the rate of convergence for the rudder control unit by a factor of 100. Therefore, by increasing the knowledge, both common and specific for sailing, it is argued that the hopes of efficiently learning the task of sailing can indeed be upheld: we propose to use the domain language, or *jargon*, as a base for identification of the important regions within the problem space. From this, the following conjecture is stated [19]:

> "Autonomous systems operating in complex, dynamical environments should be based on expert-rules, whose atomic elements are grounded using both symbolic and behavioristic approaches. These two fundamentally different concepts should be brought together using hybrid software architectures."

It was hypothesized that jargon can be used to this end, because this subset of human language has evolved to express certain dynamics and events typical to the domain. If this world is a complex and dynamic one, then this evolved set of words will indicate certain phase transitions, ie. events that mark boundaries between certain stable regions, in which efficient learning techniques are most likely to be feasible.

In conjugation with this observation regarding jargon, we have the notion of *granularity*. Choosing the right granularity is of paramount importance, since certain events can only be measured at certain levels of granularity. Looking at the right place in the wrong way means missing some or maybe all potentially useful information. Jargon reduces the problem's complexity by providing rough boundary's on the problem space, so search and/or learning algorithms can explore smaller search spaces.

The challenge thus lies in bringing together high and low levels of reasoning, each at their appropriate level of granularity. This hybridization should be achieved by using behavioristic techniques for the low-level areas, and rule-based reasoning for the highest level. In between these two extremes, we encounter the traditional problem of *symbol grounding*. The approach taken in this paper is to ground elements of sailing jargon, following a methodology described by Harnad [9]. We thus employ an approach combining techniques with rule-based reasoning, set in an architecture based on *agent technology*.

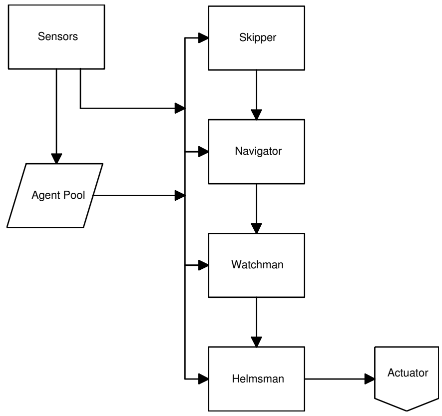

Figure 2. Command Hierarchy and Agent Pool Architecture

An architecture based on these ideas has thus been designed. It is shown in figure 2 and is based on a merge of the subsumption architecture by Brooks [3] and the Xavier-architecture by Simmons [15], combined with agent technology, in the form of an *agent pool*. Within this pool, each agent instantiates one or more concept from sailing jargon.

Let us view an example. One of the expert rules extracted from multiple sailing experts is the following:

> "If you are sailing close-hauled and there is a gust of wind then steer the boat a bit windward."

This rule can then be interpreted and enriched to the following script, which is stated in more explicit terms:

> "If the apparent wind angle ϕ is between ϕ_a and ϕ_b and the apparent wind speed average $\overline{\alpha}$ is around α_q and $\overline{\alpha}$ increases by a factor of f for more then t seconds then steer the ship ξ degrees windward."

Within the RoboSail architecture, each of the separate ideas contained in the script, *agents* have been created. This idea has been inspired by the research of Minsky [12], who argued for the interpretation of the human brain as a collection of cooperative agents.

For example, there is an agent which identifies if the ship is indeed sailing *close hauled*[7]. There is also an agent which judges if there is a gust or not. These beliefs are then used in the rule-based reasoning component.

[7] Close hauled means the ship is sailing close to the wind, i.e. sailing against the wind at an angle of about 45 degrees true wind.

3 IMPLEMENTATION

3.1 Hardware

A practical implementation of the X-Pilot system starts with the design and implementation of hardware. The hardware involved is built on the Controller Area Network, or CAN bus [7], the de-facto standard in automotive applications. The CAN bus provides an extremely reliable and robust medium for data transport. As depicted in figure 3, the four main components of the RoboSail net are:

- Intelligent Rudder Control Unit (iRCU)
- Motion Sensor Unit (iMSU)
- General Processing Unit (GPU)
- Display & Control Units (DCU's)

The iRCU is RoboSail's advanced 75 Ampère pilot drive unit, making it the most powerful rudder drive unit in existence. It has been developed in close cooperation with NIKHEF[8]. The iMSU contains

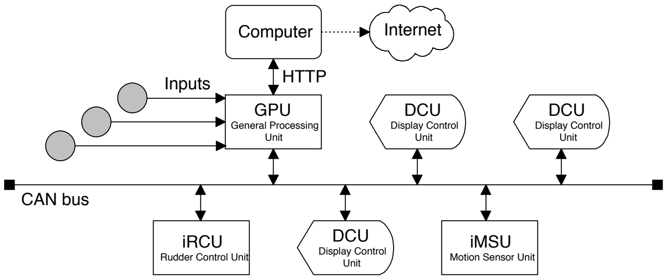

Figure 3. RoboSail's CAN-based network architecture

sensors to measure the motions of the ship in all 6 Degrees Of Freedom (DOF), as well as a high-speed and extremely accurate self-calibrating compass. The interface to the rest of the world is the GPU, which interfaces to an external computer and/or the Internet through HTTP, and also receives and interprets legacy data from external sources, like GPS[9] information and the like. Finally, the DCU's have been developed to allow the human sailor to easily interface with the system, even in the harshest conditions. All hardware has been designed to be extremely robust: operating temperatures range from -40^oC to $+85^oC$, and all equipment has been stored in fully watertight cases.

3.2 Software

Figure 4 sketches the structure of the implementational architecture of the RoboSail X-Pilot. The first layer in the architecture is the *Sailing Development Kit*, or SDK: an extensive API that includes various bi-directional interfacing software, error correction and data filtering algorithms. It also contains an extensive library of Machine Learning and data enhancement algorithms. The SDK also provides the event-based data infrastructure for the entire system. On top of the SDK, two layers of *sensors* and *virtual sensors* have been defined. Sensors are the software representation of the hardware sensors mounted on and in the ship, corrected for transmission and consistency errors.

[8] Dutch National Institute for Nuclear and High-Energy Physics
[9] Global Positioning System

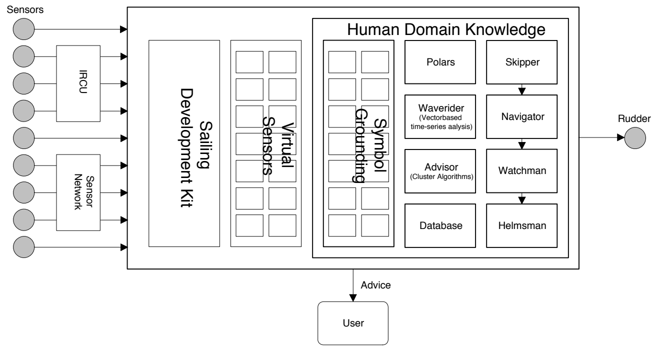

Figure 4. Implementational Architecture

Virtual sensors compute new data from information from multiple sensors. A virtual sensor can be implemented by any sensor fusion approach, although we use, among others, linear Kalman Filters, n-dimensional Extended Kalman Filters [10] (for $n \geq 2$) and an instance of an Covariance Intersection Filter [18].

The next layer is the symbol grounding layer. It consists of a number of agents, each of which is responsible for grounding one or more related concepts from sailing jargon. For example, there are separate agents which hold beliefs about wind force and wind direction, about the state of the sail trim, etc. Together, they form a *context* of higher-level concepts, that can be used by other agents to aid in reasoning about the state of the world, the ship and the actions undertaken by the X-Pilot. This information can then be used for more reasoning, or serve as feedback for a performance measure.

Separate from this *agent pool* is the *command hierarchy*, the complex of four agents that are ultimately responsible for the system's actions. The *Helmsman*, ultimately responsible for the control of the rudder, is based on a PID controller, equipped with an adaptive variant of Ziegler-Nichols open-loop tuning [20].

The *Watchman* has the ability to advise the Helmsman to temporarily adjust its course. The Watchman contains a rule base, with rules stated in terms of agent beliefs. Therefore, the Watchman can reason in the same terms as a human can, provided enough elements of the language have been grounded. For example, the script given in section 2, translates to Watchman pseudo-code as:

```
IF
    WindAgent.WindMode = CLOSEHAULED
        AND
    GustAgent.Gust = TRUE
THEN
    Helmsman.Luff(GustValue)
END IF
```

The last variable 'GustValue' is either based on expert knowledge, or can be iteratively learned by the agent responsible; in this case the Navigator agent. When the state of the world changes (for example, the boat accelerates) some agents detect this change (e.g. the boat starts *planing*) and advise the Helmsman to alter it's way of steering. Also, the human skipper can be advised to watch or retrim the sails.

Other important components of the system are:

- Advisor

- WaveRider
- Polars
- Database and Data Explorer
- ModelBuilder

The *Advisor* is an important part of the decision support system: it employs an expert knowledge-enriched k-Nearest Neighbor (kNN) algorithm to advise the sailor on differences between the current sailtrim and those that have been experienced by the X-Pilot in the past. This information is used to inform the human skipper of the current state of and changes in performance.

The *WaveRider* is an implementation of a Linear Predictive Coding (LPC) algorithm from the area of time-series analysis [6]. Like in several speech recognition systems, it predicts waves on the basis of past measurements and estimates their relative height and direction. This information is used to take advantage of the potential speed-increasing surfing effect that can arise from steering off of waves.

The *Polars* learn a dynamical model of the boats characteristics in different wind speeds, wind angles and wave patterns. These can be used to advise sailor, sail maker and boat designer on possible flaws in the boats performance.

All information in the system is continuously being stored in a *database*, which can be analyzed both on- and offline in the *Data Explorer*. Here, a collection of clustering algorithms, support vector machines and association rule learners can distil the data into new sailing knowledge. For testing and development purposes, we enhanced the SDK with the *Model Builder*, a real-time, dynamic visual programming system in which all parameters and data flows of the program can be displayed and changed. This addition might seem trivial, but when testing in severe conditions, any alteration which has to be done in code can effectively terminate a testing session.

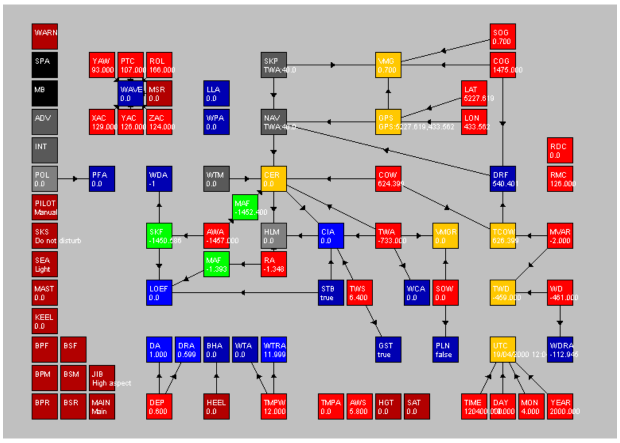

Figure 5. ModelBuilder visual programming environment

4 TESTS & RESULTS

The first installment of the complete X-Pilot on the *Syllogic Sailing Lab* Open40 was done in parallel to existing autopilot systems, which enabled us to compare results.

After calibration of the X-Pilot, the first tests showed us that it could handle basic steering behavior. However, after enabling the higher level reasoning it showed some clear advantages over the other autopilot systems. Due to the encoded sailing knowledge in the rule base, it could actively and intelligently cope with situations in which the other pilots would fail: typically, those pilots persistently follow a straight line, relative to magnetic compass or wind angle, whereas the X-Pilot can choose to deviate from this.

Data analysis after the firsts tests showed that the lower level physical layer which handled rudder control, had a more efficient power consumption and a higher boat performance compared to the other pilot systems. Besides the numerical comparisons, from expert human sailors we also received numerous statements that the X-Pilot sails better, smoother and more intelligently.

The decision support system was tested in an endurance test which would take the Syllogic Sailing Lab over the Atlantic Ocean in an actual sail race. The continuous feedback on boat performance and sail trim advice between the X-Pilot and the skipper alerted the skipper whenever the boat performance dropped, thus enabling him to keep the boat fine tuned and always sailing at an averaged performance of 85%. During long sailing trips, a human skipper normally identifies possible sail trim improvements when performance has dropped below 75%, with the decision support system, the human skipper is alerted as soon as performance drops below 85%.

The decision support system is currently deployed on the *djuice dragons*[10], a fully crewed V60 sailboat which competes in the *Volvo Ocean Race 2001/2002*[11]. The decision support system helped the team to select and adjust the sail wardrobe before the race. During the race, it continuously advises the (human) navigator and helmsman on trim improvements and sail wardrobe changes.

The complete X-Pilot system has also been applied on the *Kingfisher Open60*[12]. A successful Atlantic Ocean crossing showed good results in endurance and applicability of the X-Pilot on larger and more powerful vessels. In the competition season of 2002, the Kingfisher Open60 will compete in a number of Grand Prix races.

Figure 6. The Kingfisher Open60

5 CONCLUSIONS

We have presented an overview of the current state of work of the RoboSail Project, an initiative aimed at developing and building a semi-autonomous autopilot for sailing vessels and improving the level of understanding of applying Artificial Intelligent in real environments. This area of research focuses upon the seamless integration of state-of-the-art intelligent autonomous systems and their human colleagues.

10 See http://www.djuice.com/dragons.
11 See http://www.volvooceanrace.org
12 See http://www.kingfisherchallenges.com.

The RoboSail X-Pilot is an adaptive intelligent system, rooted in solidly designed hardware, based on a Controller Area Network (CAN). We have sketched a novel architecture, based on those by Brooks and Simmons, but enriched with an *agent pool*, performing the task of grounding raw sensor data to symbolic domain language.

For the integration between the physical, senso-motoric reality and the rule based reasoning aspects to work, a special subset of natural language, in the form of jargon, has been used as a bridge. Through this jargon, the complete problem domain is divided into smaller domains, designated by different terms from the jargon. This results in a significant decrease in problem complexity.

To support such a jargon based system, an agent pool was created in which a number of agents reside, each of which is concerned with one specific element of operation, one term of the jargon. Since each agent is a little autonomous system in itself, it can for itself determine the optimal grain-size of operation, be it in its design or through learning algorithms. Since the landscape of the problem is high-dimensional and chaotic, this potential diversity is necessary.

Furthermore, the argument holds the other way around: jargon also implies a measure for optimal granularity. Through this argument, the creation of an agent pool determines the correct granularity of each sub-domain, thus described terms function as glue between high-level reasoning and the physical aspects of the real world.

The system described here has been successfully tested on our own racing yacht, as well as by some of the leading teams in short-handed sailing. Results have been positive and encouraging: the X-Pilot is a more complete, reliable and better performing pilot system than any other currently available.

6 FUTURE WORK

We are currently developing the third version of the X-Pilot, which also enables intelligent control on multihulls. The factor of balance on a multihull is far more important compared to a monohull and, due to the much higher boat speeds, shorter reaction times are needed. When combined with the higher risks of sailing a multihull[13], it is clear this seriously complicates research and development.

One of the most promising aspects of performance gain is not yet covered by the X-Pilot: that of a high level navigation aid. A human skipper collects meteorological and nautical data from a whole range of sources and determines the best strategy to reach a certain goal. To extend the current autopilot with these kinds of information streams will further increase the optimization process of sailing.

Another promising aspect is that of reinforcement learning in high dimensional continuous domains, such as Q-Learning as stated in [17].

In cooperation with Perot Systems Netherlands, a project has been launched to analyze data gathered by the X-Pilot by use of advanced data-mining technology [14].

With the University of Utrecht, work in the area of real-time visual sailshape recognition by means of geometrical pattern recognition techniques has just finished [4].

Also currently underway is a project to apply new theory in the area of vector-based time-series analysis [2], as developed at Delft University of Technology, to multi-dimensional wave prediction.

Finally, with the Intelligent Autonomous Systems Group at the University of Amsterdam, the use of Machine Learning techniques for learning high-level behavior is under research, for use in both *RoboSoccer* as well as RoboSail.

[13] A 60 foot multihull will probably not survive capsizing at a speed of over 20 knots.

The future also holds a series of two-boat tests with the aim of objectively and accurately determining the success achieved by the X-Pilot. Also, for this season, training the pilot by means of supervised learning has been planned.

ACKNOWLEDGEMENTS

We would like to thank all those that helped and inspired us over the last few years. Most notably, we thank Peter van Lith of Lithp Systems BV, University of Amsterdam and Delft University of Technology, for his help and insights. Also, we thank Bram Rooijmans, to whom the authors are greatly indebted for his indispensable work and ideas. Furthermore, we thank Mark de Gids for his splendid work on his sailshape recognition project, as well as Jonatan Samoocha, Erik Hermans and Mans Scholten for insightful discussions and ideas. We would also like to thank Lex Keuning from the department of Maritime Technology of the Delft University of Technology for his extensive knowledge on everything that moves over water in an efficient way. Finally, the authors wish to thank the reviewers of this article for their comments and suggestions.

REFERENCES

[1] F. Bethwaite, *High Performance Sailing*, Waterline Books, 101 Longden Rd., Shrewsbury, England, 1996.
[2] P.M.T. Broersen, 'Facts and fiction in spectral analysis', *IEEE Transactions on Instrumentation and Measurement*, **49**(4), (2000).
[3] R.A. Brooks, 'A robust layered control system for a mobile robot', in *IEEE Journal of Robotics and Automation*, pp. 14–23. MIT AI Lab, (1986). Lab Memo 864.
[4] M. de Gids, *Zeilvorm analyse met geometrische patroonherkenning*, Master's thesis, Utrecht University, 2001.
[5] *FINNatics*, ed., R. Deaves, Fernhurst Books, 2000.
[6] J.R. Deller, J.G. Proakis, and J.H.L. Hansen, *Discrete-time Processing of Speech Signals*, MacMillan Publishers Ltd., New York, 1993.
[7] K. Etschberger, *Control Area Network*, IXXAT Automation GmbH, 2001.
[8] R. Garrett, *The Symmetry of Sailing*, Adlard Coles Nautical, 1996.
[9] S. Harnad, 'The symbol grounding problem', *Physica*, **D 42**, 335–346, (1990).
[10] R.E. Kalman, 'A new approach to linear filtering and prediction problems', in *Transaction of the ASME, Journal of Basic Engineering*, pp. 33–45, (1960).
[11] J.W. Koeman, *Mathematisch model voor het door een windvaanstuurinrichting geregeld stuurgedrag van een zeiljacht*, Ph.D. dissertation, Delft University of Technology, 1973.
[12] M.L. Minsky, *Society of Mind*, William Heinemann Ltd., London, 1987.
[13] C. Sammut, S. Hurst, D. Kedzier, and D. Michie, 'Learning to fly', in *Proceedings of the Ninth International Conference on Machine Learning*, Aberdeen, (1992). Morgan Kaufmann.
[14] J. Samoocha, *Learning to Sail*, Master's thesis, University of Amsterdam and Perot Systems Corp., 2001. In press.
[15] R. Simmons, R. Goodwin, K.Z. Haigh, S. Koenig, J. O'Sullivan, and M.M. Veloso, 'A layered architecture for office delivery robots', in *Proceedings of First International Conference on Autonomous Agents*, ed., W.L. Johnson, pp. 245–252. ACM Press, (1997).
[16] RoboSail systems BV website. http://www.robosail.com.
[17] S. the Hagen, *Continuous State Space Q-Learning for Control of Nonlinear Systems*, Ph.D. dissertation, University of Amsterdam, 2001.
[18] J.K. Uhlmann, S.J. Julier, M. Csorba, and H.F. Durrant-Whyte, 'A culminating advance in the theory and practice of data fusion, filtering and decentralized estimation', Technical report, Covariance Intersection Working Group (CIWG), (1997).
[19] M.L. van Aartrijk and C.P. Tagliola, *Trials and Tribulations of RoboSail: Managing a Complex World*, Master's thesis, University of Amsterdam, 2002.
[20] J.G. Ziegler and N.B. Nichols, 'Optimum settings for automatic controllers', *Trans. ASME*, **64**, 759–768, (1942).