Learning to Sail

M.L. van Aartrijk* J. Samoocha†

Abstract

This article describes a fully functional, already deployed intelligent adaptive system, called the Odys Intelligent Pilot, built by RoboSail systems [11]. It is an intelligent sailboat autopilot and diagnostics system, built upon a hybrid architecture with use of machine learning methods. These methods are used in a knowledge discovery process that assists the incremental build-up of a rule-base that describes higher-level knowledge about the activity of sailing.

KEYWORDS: Intelligent autonomous system, knowledge discovery, machine learning, sailing

1 Introduction

[1] This paper describes the RoboSail Odys Intelligent Pilot system that has been built to be the ultimate in semi-autonomous control of sailing yachts. A description is given of the overall system, its architecture and its hardware- and software implementations. The second part of the paper describes a set of knowledge discovery modules, as developed by Perot Systems Netherlands. These modules were aimed at learning new 'rules of sailing' from the databases collected during actual sailing sessions.
The overall goal of the RoboSail project, that was also described in [12], is to create an intelligent autopilot system, that steers a ship at least as good as a human helmsman can and also optimally support short-handed sailors[2]. Restrictions in this include the fact that no automatic handling and trimming of sails is usually possible, i.e. the sails cannot be adjusted automatically. Only direct control of the rudder is possible. However, in combination with an intelligent Advisor component, man and machine can effectively operate together. First, the hybrid, agent-based *Avalon* architecture that underlies the Odys Pilot

*martijn@robosail.com; RoboSail systems BV, Waterlandlaan 120, 1441 RW, Purmerend, The Netherlands

†Jonatan.Samoocha@ps.net; Perot Systems NL BV, Hoefseweg 1, 3800 GG, Amersfoort, The Netherlands

[1] Information contained within this document is confidential and proprietary to Perot Systems and/or RoboSail systems and should not be disclosed to anyone other than the intended recipient(s) of this paper

[2] A 'short handed sailor' is a sailor or team of sailors that race a sailing boat with less crew than is normally deployed.

Figure 1: The Open40 *Syllogic* sailing lab

is explained. This architecture brings together elements from reactive behavior as well as from higher-level reasoning, and various levels in between. Hidden, but well defined knowledge about the activity of sailing exists in the form of sailing *jargon*. This jargon, that can be seen as a condensation of many centuries of human sailing experience [6], contains many clues as to how to optimally implement an intelligent system that can actually assist a human sailor.
Next, the physical implementation of this system's hardware and software is treated. This encompasses both the *Avalon Network*, the physically robust realization of the Avalon Architecture, as well as the software suite which extends the hardware and completes the architecture to form the Odys Pilot.
The second part of this article is devoted to the knowledge discovery process, which is used to uncover patterns in the data collected in the system's databases. The methodology is presented, as is the structure of the knowledge that is looked for; and finally some results. Conclusions will finish this article.

2 The Avalon Hybrid Architecture

Sailing is a complex activity, containing elements from low-level 'feeling' to high-level reasoning activity. Controlling a rudder is largely a matter of 'feeling' and 'experience'. On the other hand, navigation is largely high-level reasoning. In between these two extremes, a lot of other activities exist as well. Judging, for example, whether or not a wind gust occurs, is a fuzzy judgement. The notion of what constitutes a wave, is not only fuzzy, it is also partly dependent on the ship one is standing on; heavy ships don't experience waves the way lighter ships do.

The hybrid Avalon Architecture is sketched in figure 2. It is based on a merge of the Subsumption Architecture by Brooks [2] and the Xavier-architecture by Simmons [10], combined with agent technology, in the form of an agent pool. Within this pool, each agent instantiates one or more concept from sailing jargon. This *pool of agents* forms a context of physically grounded concepts. The

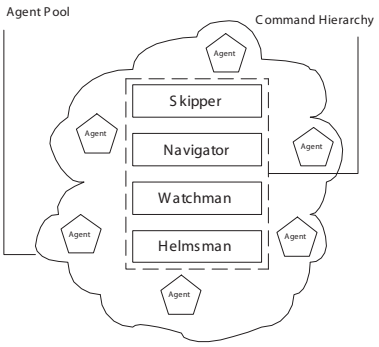

Figure 2: Overview of the Avalon Architecture

functional architecture also defines the existence of four so-called 'main agents', which have the end-responsibility for carrying out certain behavior. These four main agents are modelled after the abstracted command hierarchy that is actually used on sailing vessels: Skipper and Navigator do long-term en mid-term planning and routing, respectively. The Watchman is responsible for short-term optimization, like using waves and wind gusts. The Helmsman only focusses on efficiently controlling the rudder. These four agents can use the context provided by the agent pool to reason about the world.
Within the Avalon architecture, for each of the separate terms in sailing jargon, agents have been created. For example, there is an agent which identifies if the ship is indeed sailing close hauled[3]. There is also an agent that judges if there is a wind gust or not. There are agents that form an opinion about the regularities in wave patterns. These beliefs are then used in the rule-based reasoning component. This means the Odys Pilot is capable of using events from the world to locally optimize its behavior, for example by luffing in a gust, or use waves to surf off.

3 The Odys Pilot System

This section briefly describes the physical implementation of the Avalon Architecture. The basis is called the Avalon Network, on which the complete Odys Pilot system has been built.

3.1 The Hardware

The Avalon Architecture, as described above, forms the basis for the RoboSail Odys Intelligent Pilot system. It not only implements the advanced intelligent features provided by the Avalon Architecture, it has been specifically built for

[3]Close-hauled: close to the wind, i.e. sailing into the wind at an angle of roughly 45 degrees

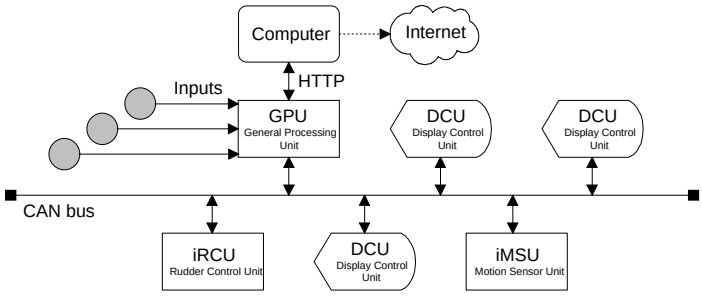

Figure 3: Drawing of CAN based Avalon Network

Figure 4: The Odys Pilot: MSU, IRCU, DCU and GPU (left to right)

extreme reliability and ease of use: something that is absolutely mandatory in the demanding environment of the high seas. The practical implementation of the Odys Pilot system starts with the design and implementation of hardware. The communication hardware involved is built on the Controller Area Network, or CAN bus, the de-facto standard in automotive applications. The CAN bus provides an extremely reliable and robust medium for data transport. As depicted in figure 3.1, the four main components of the Odys network are:

IRCU: The Intelligent Rudder Control Unit is RoboSail's advanced 100 Ampere pilot drive unit, making it the most powerful rudder drive unit in existence. It has been developed in close cooperation with NIKHEF, the Dutch Institute for Nuclear- and High Energy Physics. The IRCU also includes a 40MHz microprocessor which implements most of the autopilot logic. Being equipped with internal temperature- and current sensors, the IRCU has everything to steer and guard the ship optimally.

MSU: The Motion Sensor Unit contains sensors to measure the motion of the ship in all 6 Degrees Of Freedom (DOF), as well as a high-speed and very accurate self-calibrating digital compass.

GPU: 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 information, external sensor networks and every other device that uses the de facto nautical interfacing standard NMEA-0183. The GPU contains two microprocessors, at 40MHz and 50MHz respectively. This embedded power ensures the system's integrity and constant performance optimization, by means of a system-wide watchdog which keeps an eye on sensor data integrity and efficiency of the overall system, in terms of both energy consumption and sailing performance.

DCU: The Display & Control Units have been developed to allow the human sailor to easily interface with the system, even in the harshest conditions. There can be more than one DCU installed on any Odys network.
All hardware has been designed to be extremely robust: operating temperatures range from -40^oC to $+85^oC$, the PCB's have been coated with a salt- and chemical proof layer, and all equipment has been stored in fully watertight and shock resistant cases.

3.2 The Software

The hardware for the Odys platform is complemented by an extensive software suite, that provides both an easy-to-ease interface to the network, as well as a high-performance addition to the hardware based pilot system. This enables the use of complex analysis- and control algorithms to optimize rudder control, without being confined to embedded hardware. If, for any reason, communications between PC and Odys Pilot should fail, the Odys immediately regains control; true to the original subsumption principles.
The software suite also includes various tools, including an incrementally learnt model of the ship (the *Polars*), a *SailCharter* module that advises on optimal sail trim and ballast distribution, a wave predictor, based on a high-dimensional Auto-Regressive Moving Average (ARMA) model, called the *WaveRider*, and various interfacing tools. A detailed discussion of all these tools is however outside of the scope of this paper. In addition, the system includes a database that logs all data flow through the system. A fully connected Odys Pilot has more than 50 data streams, all of which are logged. This data is used online in a suite of analysis- and Machine Learning algorithms and serves as input for the Knowledge Discovery process.

4 Knowledge Discovery on Odys Data

In order to extend the rule base of the Odys pilot, knowledge discovery methods have been applied to the data collected by the system. A variety of these methods and processes are described in [5] or [3]. The goal of applying these knowledge discovery methods was to learn a static model of the boat's performance, which is defined as follows:

Let an *attribute* $a \in A$ be some measurable property of the Odys pilot. An *attribute value* $v_t(a_i)$ represents information about the system itself, or its environment, where i stands for the i-th attribute and t is a given time-stamp. If the system has n attributes $a_1 \ldots a_n$, an n-tuple $\langle v_t(a_1), \ldots, v_t(a_n) \rangle$ describes an *observation* of the system at time t. A *configuration* C of the system is given as $\langle v(a_1) \in X_1, \ldots, v(a_m) \in X_m \rangle$, where $\{a_1, \ldots, a_m\} \subseteq \{a_1, \ldots, a_n\}$ and $\forall i : X_i \subseteq A_i$ if A_i is the domain of attribute a_i. The attribute describing the performance will be seen as a special attribute of the system, called a_0, with domain A_0.This domain is assumed to represent a discrete set.
The required output from the knowledge discovery process is a set of rules with the following form:

$$a_0 = v(a_0) \leftarrow C_1 \vee \ldots \vee C_k \tag{1}$$

The inputs to this process, as well as the specific application of knowledge discovery methods are described in the remainder of this section.

4.1 Input Data

The data as used in this exercise was collected between March and July 2000, when a prototype of the Odys pilot was installed on the yacht *Syllogic*. The data was collected during a wide range of activities varying from software tests to an actual Trans Atlantic sailing race.
The data was stored using *sessions*. With a frequency of 10 Hz records were written to a session file until (i) manually aborted or (ii) expiry of a given time limit. In the last case, a new session was created that was used to store the data. During the five months mentioned above more than 500 sessions were collected, containing over 13 million records of raw data.
The attributes describe three categories of data:(i) literal sensor values, (ii) manual input from an end user, and (iii) other output from the Odys pilot. For the knowledge discovery, only attributes of the first category were used. Examples of these attributes are wind angle, wind speed, boat speed, boat course, boat motion (acceleration in six degrees of freedom) and rudder angle.
The collected (raw) data can be qualified as very noisy: attribute values were missing for various reasons such as sensor breakdowns, end users not manually logging changes to the boat's configuration and having multiple sensors with multiple frequencies for delivering data. In addition, some data was meaningless because the boat was lying in a harbor or "sailing" on its inboard motor.

4.2 Data Extension

In order to enrich the raw data, new attributes were derived from the existing ones. Multiple methods were used to extend the data:

Regular Extensions: These extensions include regularly used functions such as derivative, local average or mod (which is used to convert absolute angles to a $[-180, 180]$ domain). In addition, a *local extreme* function was used to find

the difference between minimum and maximum values of an attribute within a given time window.

Fuzzy Extensions: Besides the numeric extensions, new attributes were created to express various concepts as used in sailing jargon. Examples of such jargon are "a wind gust" or "sailing close hauled". These concepts were modeled as fuzzy sets.

Clustering: Motion sensors measured the boat's acceleration in the direction of its X -, Y - and Z axes. These attributes were used to derive a new attribute describing the sea state. This sea state attribute was created as follows:

1. A clustering algorithm (k-means) is applied to the three motion attributes, resulting in 5 clusters.
2. Each cluster describes a combination of 3 motion attributes's values. These clusters each represent a (relative) sea state.
3. Each record is then classified according to the clusters.

Performance Attribute The performance attribute has been derived from the boat's speed relative to its maximum possible speed given the current wind speed and wind angle. Usually, the combination of wind speed, wind angle and maximum possible boat speed is expressed in a *polar diagram* [1].This performance measure is then defined as the fraction $CUR - SPEED/MAX - SPEED$, which is discretized to 5 possible values (0-20%,..., 80+%).

4.3 Experiment

The process of creating samples is shown in figure 5. After extending the data, all the reliable sessions were randomly distributed among a training - and validation set. The session structure was maintained in this process. The next step involved selecting records from both sets using a selection query.
Creation of the final training and validation samples was driven by different goals: where the training sample needed to be as diverse as possible the validation sample was required to be as independent as possible. Diversity of the training sample was achieved by picking random records out of the record set satisfying the selection query. Independence of the validation sample was achieved by (i) ensuring that no records were selected from sessions used for training, and (ii) ensuring that the sample was created using a different process than used for the training sample. This other process consisted of choosing chronologically ordered sets of records (satisfying the selection query) from the sessions. In this way, the learnt model could be confronted with (simulated) new sailing situations. The generated training sample has been used to learn decision trees, using the C4.5[7] algorithm. All trees were generated with a (default) pruning severity of 0.75. The experiment consisted of increasing the

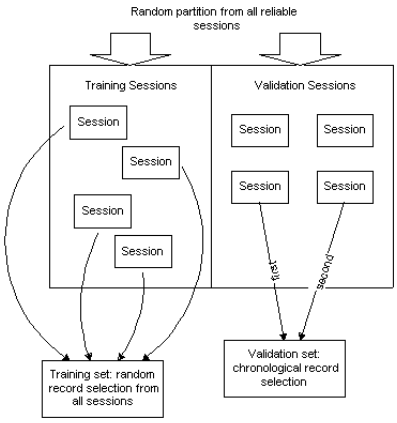

Figure 5: The sampling mechanism

minimal number of records covered by a branch (which equals the minimal generalization represented by the tree) in steps of 500. Finally, the generated trees were converted to rule sets. For each model, its predictive accuracy was tested on the validation sample as well as on the training sample itself.

4.4 Knowledge Discovery Results

Performance	Training	Validation
0-20%	0,01	0,04
20-40%	0,04	0,02
40-60%	0,26	0,19
60-80%	0,50	0,50
80+%	0,18	0,25
Examples in set:	353613	49210

Table 1: Distribution of Performance Attribute

The results of creating sample sets are shown in table 1. This table shows the distribution of the performance attribute in the generated samples. The bottom row shows the total number of records in each sample. The distributions show that there is a bias towards the area of 60-80% of the maximum performance. Further, distributions of the validation sets (especially Validation2 and Validation4) show the required independence in relation to the training set. The (high-level) results of learning the decision trees and testing their predictive accuracy are shown in figure 6. It represents the percentage of correctly classified records in relation to the minimal generalization of the learnt model. The plot

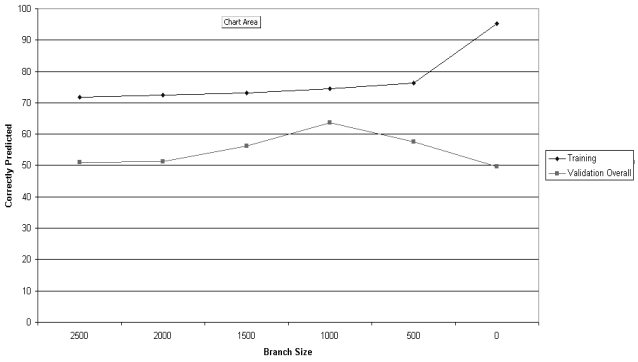

Figure 6: Model Accuracy

shows a maximum predictive accuracy for the model generated with a minimum branch size of 1000 records. When the models are allowed to be more specific, predictive performance drops.
When looking at the models concerning content, the decision trees seem to reflect an experienced sailor's common sense knowledge[4]. One example of the rules resulting from the knowledge discovery process looks as follows:

```
if  md_TWA =< 50.959
and  fuz_nowind_md_TWA =< 0
and  abs_RA > 14.985
then -> 0-20%_PolarSpeed
```

The rule shows that boat performance is bad when sailing close-hauled and at the same time having a deflection of the rudder.
In addition, the models showed that the fuzzy attributes describing sailing jargon sometimes appeared on high levels in the decision trees, which supposes a significance for classification. However, discussing detailed content of the models is beyond the scope of this paper. More results are detailed in [9].

4.5 Partially Automated Knowledge Discovery

To be able to easily integrate the used knowledge discovery process with the Odys pilot, a substantial part of this process is automated. This automation is mainly driven by metadata, which is collected for (i) the complete data set and (ii) each session. For each attribute, properties such as the fraction of missing values or mean value were compared between a session and the complete data set. Based on this comparison, the decision on which sessions to choose for data extension or sample creation is automated.
The data extension process is automated as well: it is possible to define chains

[4] As judged (subjectively) by the owner/skipper of the *Syllogic*.

of derivations in order to create n-th order derived attributes. These definitions are saved as scripts that can be re-used.

5 Conclusions

The Odys pilot has been successfully deployed in the real world. It is, amongst others, installed on the *Syllogic*, a high-tech Open40 racing yacht. This yacht is managed by the *Syllogic Sailing Team* [13]. The yacht has participated in internationally acclaimed sailing races, of which it has won the 2002 Dual Round Britain & Ireland.
Results as described in section 4.4 show that it was possible to learn non-trivial models from very noisy data. The models are qualified as being non-trivial by objective (predictive accuracy) as well as subjective (judgement of an experienced sailor) measures.
Having automated large parts of the knowledge discovery process creates the possibility to integrate it with the existing architecture. Future research should focus on the actual integration of these two components. More information about current research activities can be found at the RoboSail website[11].

References

[1] F. Bethwaite. *High Performance Sailing*. Airlife Publishing Ltd., 1996.

[2] R.A. Brooks. A robust layered control system for a mobile robot. In *IEEE Journal of Robotics and Automation*, pages 14–23. MIT AI Lab, 1986. Lab Memo 864.

[3] P. Chapman, R. Kerber, J. Clinton, T. Khabaza, T. Reinartz, and R. Wirth. The crisp-dm process model, 1999.

[4] Robert Engels and C. Theusinger. Using a data metric for preprocessing advice for data mining applications. In *European Conference on Artificial Intelligence*, pages 430–434, 1998.

[5] Usama M. Fayyad, Gregory Piatetsky-Shapiro, and Padhraic Smyth. From data mining to knowledge discovery: An overview. In *Advances in Knowledge Discovery and Data Mining*, pages 1–34. 1996.

[6] Edwin Hutchins. *Cognition in the Wild*. The MIT Press, 1995.

[7] J. Ross Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann, 1993.

[8] 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.

[9] Jonatan Samoocha. Empirical determination of sailing boat performance. Master's thesis, University of Amsterdam, 2003. To appear.

[10] Reid Simmons, Richard Goodwin, Karen Zita Haigh, Sven Koenig, and Joseph O'Sullivan. A layered architecture for office delivery robots. In W. Lewis Johnson and Barbara Hayes-Roth, editors, *Proceedings of the First International Conference on Autonomous Agents (Agents'97)*, pages 245–252, New York, 5–8, 1997. ACM Press.

[11] RoboSail systems BV website. http://www.robosail.com.

[12] Martijn van Aartrijk, Claudio Tagliola, and Pieter Adriaans. AI on the Ocean: the RoboSail project. In Frank van Harmelen, editor, *Proceedings of the 15th European Conference on Artificial Intelligence*, pages 653–657. IOS Press, 2002.

[13] Syllogic Sailing Team Website. http://www.robosail.com/syllogic.