A Low Cost Adaptive Autopilot For Inland Ships

J. van Amerongen
H. Duetz
T. Okawa

This paper was presented at the symposium on
Low Cost Automation, 26-29 November 1986, Valencia, Spain

A LOW COST ADAPTIVE AUTOPILOT FOR INLAND SHIPS

J. van Amerongen, H. Duetz, T. Okawa

Control Laboratory, Department of Electrical Engineering, Delft University of Technology, P.O. box 5031, 2600 GA Delft, The Netherlands

Abstract. This paper discusses the design of an autopilot for inland ships. The autopilot makes use of adaptive strategies which can be implemented in inexpensive microcomputers. As the paper makes clear, both the adaptive features and the possibility to implement these in inexpensive hardware are essential for realizing an acceptably priced autopilot.
As an example of these, the adaptive controller of the ship's hydraulic steering machine is described. This controller computes on-off signals for the relays which control two hydraulic valves.
For the realization of the autopilot an 8-bit microprocessor system is chosen which can be programmed in a high-level language. The reasons for selecting this system are given along with some properties of the system, which have to be taken into consideration in the software design process.
The results of simulation and of full-scale trials are described.

Keywords: Relay control; Adaptive control; Ships; Computer selection & evaluation ; Low Cost automation.

1. INTRODUCTION

An autopilot for inland ships aids a skipper in sailing curves and straight tracks in canals and rivers. Unlike an autopilot for sea-going ships it controls the ship's rate of turn rather than the heading. With an autopilot, the skipper can more easily control the motions of the ship, so that he can concentrate on the other traffic and the path of the river. Another advantage of a good autopilot is the reduction in fuel consumption achieved by steering more accurately with fewer rudder moves.

Inland ships are usually owned and sailed by single-owner companies. By working hard the skipper and his family usually earn a marginal income. Obviously, they cannot afford expensive instruments.
The price of an autopilot for inland ships depends on various factors:

- cost of the hardware
- development cost
- installation cost

The total cost has to be considered in relation to the operational cost which depends on:

- maintenance cost
- operational performance (fuel cost)

There are inland ships of many different sizes, with a wide range of engine power and different types of rudders. Therefore, there are big differences in the dynamics of inland ships. This has some implications for the autopilot. The present autopilots can be used on a wide range of inland ships, but they have to be tuned carefully before they work well on a specific ship. This tuning is often difficult and time consuming. An experienced and costly engineer may need more than one day to install and tune the autopilot. The installation and tuning cost alone forms an important part of the price of the present autopilots for inland ships.

The sailing conditions of an inland ship change often. The important factors which influence the dynamics of the ship are its load, its speed and the depth and width of the river. The present autopilots have one or more knobs to adjust the controller settings for changing sailing conditions. Especially when the knobs are not clearly related to the sailing behaviour of the ship, it is very hard for a skipper to find good settings.

The Control Laboratory is working on a project to reduce the installation cost and improve the performance of an autopilot for inland ships by introducing adaptive algorithms. These algorithms will eliminate the most important part of the tuning. It makes the installation easier, so that it does not necessarily have to be done by a costly engineer. The adaptive algorithms will also guarantee good performance under varying sailing circumstances. This will decrease the fuel consumption and will contribute to the safety and reliability of the autopilot.

Although adaptive algorithms can reduce the installation cost and the fuel consumption, most of the adaptive controllers which are in use at the moment are still based on quite expensive hardware. For the autopilot which is being developed now, the adaptive algorithms to be used will be able to be implemented in an inexpensive microcomputer.
The use of adaptive strategies in inexpensive hardware increases the development cost of the autopilot. However, it can be apportioned among many autopilots if the design is a success.

The Control Laboratory is designing the adaptive autopilot for inland ships in close cooperation with two small companies; Kuipers Electronic Engineering B.V. will be the manufacturer of the hardware and Egas Scheepsdiesel- en Elektrotechniek B.V. will sell and install the autopilots. The project is supported by the Netherlands Foundation for Technical Sciences. Although much research on autopilots for ships had been done before at the laboratory (e.g. Van Amerongen 1982, Bouman 1981), the project officially started in December 1985. The first full-scale trials were done in August 1986.

Figure 1 shows the structure of the autopilot for inland ships. The skipper defines the track to be sailed. He translates it into the desired rate of turn, which is the

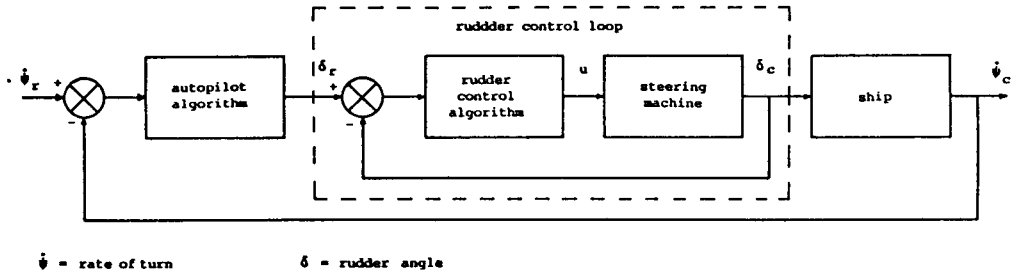

Fig. 1. Structure of an autopilot for inland ships

setpoint of the autopilot. The autopilot algorithm calculates the setpoint of the rudder angle to be realized by the rudder control algorithm, which is implemented in the autopilot as well.
As an example of the use of adaptive strategies in inexpensive hardware, the design of the adaptive rudder controller will be described in this paper. It controls the inner loop of Fig. 1.
Section 2 of this paper deals with the modelling of the steering machine. Section 3 describes the adaptive rudder control algorithm. The choice of the hardware and its implications for the software design are explained in section 4. Section 5 shows the results of experiments with an analogue simulation model as well as during full-scale trials. Finally section 6 summarizes the conclusions and indicates the next possible steps in the development of the adaptive autopilot for inland ships.

2. MODELLING

The process considered in this paper can be represented by the block diagram of Fig. 2.

The steering machine of an inland ship consists of a hydraulic system, controlled by two valves. The rudder control algorithm has to compute on-off signals for the relays of these valves in order to make the rudder angle equal to the desired rudder angle, which is calculated by the adaptive autopilot algorithm.

The block diagram of Fig. 2 is a simplification of the reality, which is less ideal. There are delays in the responses of the hydraulic valves. Another complicating factor is that the linear part of the system may better be represented by an underdamped second-order system than by the first order-model of Fig. 2.

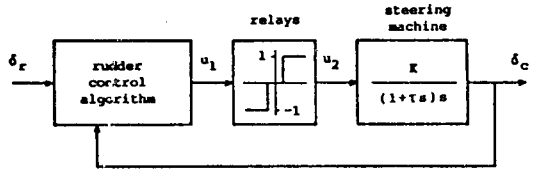

Fig. 2. Model of the rudder control loop.

The parameters of the model, especially the maximum rudder speed (K), may vary over a wide range in different steering machines. Table 1 shows some typical data of steering machines of inland ships.

TABLE 1 Typical Steering Machine Parameters

K (max. rudder speed)	2.5 deg/s	- 25 deg/s
T (time constant)	0.05 s	- 0.15 s

Many new ships have a steering machine of which the maximum rudder speed can be controlled by a proportional valve. These proportional steering machines are easier control, but they are more expensive. In this paper only steering machines without a proportional valve will be considered.

3. ADAPTIVE CONTROL OF THE STEERING MACHINE

To achieve accurate steering, the rudder controller should position the rudder within less than 1.0 degree from the desired rudder angle. It should do this with as few steering pulses as possible in order to get a fast response and to reduce the wear on the steering mechanism. The rudder controller should also be adaptive because the tuning procedure must be short and simple and the parameters of the steering machine may vary. Last but not least, it should be possible to implement the algorithm in inexpensive hardware (section 1).

Hartman and Bailey (1981) proposed an algorithm which can meet all these demands. It is based on the following philosophy.
If the dynamics of a relay-actuated process are known, it is possible to generate a table which contains a switching schedule of the relays, in order to move from one position to another. When the table is filled with the correct data, the move will be performed correctly as far as the limited discretization of the table allows.
When the move is completed and the final position is not correct, the move provides the following information which can be used in an adaptation process:

1. The move corresponding with the pulse width used is now in principle known exactly.
2. Information on how to correct the pulse width of the intended move is obtained.

In this way the adaptation mechanism adjusts the table to improve the accuracy of the next rudder move. This algorithm is well suited to use in inexpensive microcomputers since it substitutes complex computations by table lookup.

The algorithm has been applied to the control system of Fig. 2. Simulation results were reported earlier by Van Amerongen and Honderd (1983). The algorithm is implemented in a microcomputer with a sampling rate of about 0.1s. In order to obtain higher accuracy, pulse-width modulation of the output pulses is applied with a discretization of 0.01s. The lookup table only contains pulse lengths for rudder displacements below a certain limit (e.g. 5 or 10 degrees). Larger displacements are simply started by switching the relay on. The range of the table is split into discrete intervals. The number of intervals depends on the required accuracy. The time required for a certain displacement is stored at the corresponding position in the table. The system is supposed to be symmetrical, so the same table is used for both port and starboard rudder moves.

When a new rudder angle is ordered, an error signal is calculated. If the error is within the specified range of the table, the nearest value in the table provides information about the time interval that the relay has to be switched on in order to reach the desired position. The sign of the error signal tells which one of the relays should be switched on for that interval. The adaptation procedure can be explained by an example:

Example
The lookup table has a range of 5 degrees with a discretization of 0.5 degrees. Suppose that the desired change of the rudder angle is 4.2 degrees. The nearest position in the table lists 1.5s for a move of 4 degrees. When the move has been completed, it appears that the actual move has been only 3 instead of 4 degrees. Then 1.5 is filled in the table for a rudder move of 3 degrees. The new pulse width for a move of 4 degrees is calculated under the (incorrect) assumption that the dynamics are linear:

$$\frac{4}{3} * 1.5 = 2 \qquad (1)$$

At the startup of the system the table is filled with pulse lengths which correspond with the maximum possible rudder speed:

$$T(\Delta\delta) = \frac{\Delta\delta}{K} \qquad (2)$$

where $\Delta\delta$ is the move to be made.

When the adaptation mechanism works properly, after a few different rudder moves, the data in the table will converge to the correct values, even in the case of non-linear dynamics, or when not only a time constant but also a time delay or other dynamics are present.

For rudder moves larger than the range of the table, the algorithm is slightly extended. First the relay is switched on until the position error is within the range of the table. Then the table is consulted to get the remaining length of the steering pulse. However, at this moment the rudder has a certain speed. The pulse length has to be corrected for this initial speed. For this correction the following formula is used:

$$T = T(\varepsilon_0) - C * \hat{\dot{\delta}}_0 \qquad (3)$$

where:

$$\hat{\dot{\delta}}_0 = \frac{\varepsilon_0 - \varepsilon_{-1}}{T_s} \qquad (4)$$

and:

$T(\varepsilon_0)$ = pulse width stored in the table
C = constant
ε_0 = position error
ε_{-1} = position error one sample earlier
T_s = sampling interval

Assuming the linear first-order dynamics of the steering machine of Fig. 2, a pulse width T and an initial rudder speed $\dot{\delta}_0$ cause a rudder move of:

$$\Delta\delta = K.T + \tau.\dot{\delta}_0 \qquad (5)$$

This implies that instead of eqn. (2) for zero initial speed, a pulse width

$$T = \frac{\Delta\delta}{K} - \frac{\tau}{K} * \dot{\delta}_0 \qquad (6)$$

or

$$T = T(\varepsilon_0) - \frac{\tau}{K} * \dot{\delta}_0 \qquad (7)$$

can be applied in order to make the desired move.

Supposing that $\hat{\dot{\delta}}_0$ is a good approximation of $\dot{\delta}_0$, the constant C follows from eqns. (3) and (7).

$$C = \frac{\tau}{K} \qquad (8)$$

Because K and τ may vary and are unknown at the moment of installation, the value of C is adapted in a similar way to the lookup table.

4. REALIZATION OF THE CONTROLLER

There can be a trade-off between hardware cost and development cost. Obviously if the hardware is not a source of limitations, the development will be easier than if the limitations of the hardware have to be taken into account in the design of the software. An optimum in the hardware cost has to be found. The cost of the autopilot may be calculated as follows:

$$P = \frac{C_{dev} + C_{dev,hw}}{n} + P_{hardw} + P_{inst}$$

where:

P = cost price per unit
C_{dev} = development cost (hours)
$C_{dev,hw}$ = dev. cost (hardware)
n = number of units to be sold
P_{hardw} = hardware cost per unit
P_{inst} = installation cost per unit

Although the number of autopilots to be sold will hopefully be large, the development cost needs further consideration.

To reduce the development effort, some restrictions on the choice of the computer system were formulated at the beginning of the project:

- Programming should be done in a high level language combined with a real-time operating system.
- It should be easy to transfer the software developed to the final commercial product.

The first restriction sets a lower limit on the computer system but it considerably reduces the effort to implement complicated algorithms and it improves the maintainability of the software. The second restriction can be satisfied in two ways:

1. using the same hardware for the development and the final commercial product.
2. using different hardware but the same programming language and operating system.

In case 2 it is possible to use a more powerful computer system during the development of the autopilot. It could reduce the development effort. On the other hand, it will be more difficult to transfer the software to the final product and it is uncertain whether the hardware of the final product is powerful enough for that software.

The cost of the computer system is roughly determined by the selection of the microprocessor. From the development point of view a 16-bit processor system based on, say, the 68000 processor would be preferable. Although 16-bit microprocessors have become much less expensive in the last few years, 16-bit microcomputer systems are still considerably more expensive than 8-bit systems. Therefore, it was decided that a Kempac computer system, which is manufactured by Kuipers B.V., would be tested. It is based on the Motorola 6809 processor with a clock

frequency of 2 Mhz. The 6809 is a powerful 8-bit processor. By choosing a proper operating system and compiler, the option to use a 68000 system for the development remains open.

The operating system chosen is OS9. It is available for both the 6809 and the 68000 microprocessor. OS9 is a Unix-like multiuser/multitasking real-time operating system and it is compact and efficient. The operating system can also be used in the final product, because it is possible to put the necessary OS9 modules in ROM.

For the OS9/6809 operating system a few compilers are available. One of them is a C compiler which is also available for OS9/68000. It was selected because C is one of the most popular programming languages at the moment and it is highly suited to this project; the C compiler produces compact size object modules which have a fast speed of execution.

In order to evaluate the absolute and relative computation speeds of the 6809 and the 68000 systems, some benchmarks were executed. One of the benchmarks is the well-known Sieve prime number program. The other benchmarks tested the speed of integer and floating point multiplications (Table 2).

TABLE 2. Results Benchmarks

	6809/2Mhz	68000
Sieve	9.0 s	6.2 s
int. mult.	0.13 ms	0.025 ms
f.p. mult.	1.6 ms	0.13 ms

The Sieve program, which consists of mainly simple integer operations, gives good results for the 6809 system. A floating point multiplication, however, takes 1.6ms. The conclusion drawn from the benchmarks is that the 6809 might be a good microprocessor for the autopilot as long as floating point operations are avoided as much as possible.

The overall performance appeared to be satisfactory. The rudder control algorithm was implemented and it takes less than half of the available processing time. Although compilation is not fast compared with a 68000 system and the size of a source module has some limitation, the development and debugging of the software is not very time consuming. The necessity of avoiding floating point operations, however, may cause programming errors (overflow, too little accuracy etc.). It also reduces the readability of the source programs.
The OS9 operating system could be modified in an elegant way for I/O with DA and AD converters and the pulse-width modulator.

5. EXPERIMENTAL RESULTS

The first tests with the adaptive rudder controller were done at the Control Laboratory. The steering machine was replaced by an analogue simulation model. A pen recorder was used to record the desired and the actual rudder angle during the experiments. These signals as well as the output to the steering machine could also be stored on floppy disc for later evaluation.

The rudder controller appeared to work well. Figure 3 shows the performance of the adaptive rudder controller with initial table values for a steering machine with a speed of 20 deg/s. The analogue model of the steering machine had a speed of 10 deg/s. After a few changes of the desired rudder angle, the table is filled with correct data and the number of necessary steering pulses reduces to one per setpoint change.

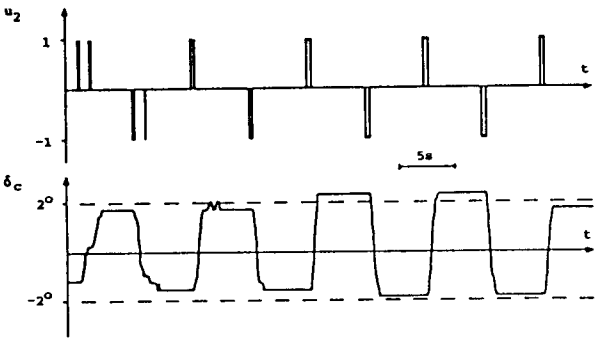

Fig. 3. Performance of the adaptive rudder controller (over-estimated rudder speed)

Figure 4 shows the robustness of the adaptive rudder controller when the speed of the model is changed from 10 deg/s to 20 deg/s.

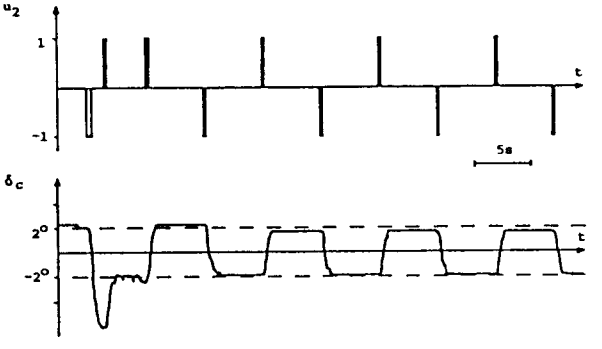

Fig. 4. Performance rudder controller after speed increase

A full-scale trial was held on board the "Arno" (Fig. 6), a small inland tanker. While sailing on the river Merwede, the controller appeared to work well and to be very robust. Without making any modifications and without tuning for this specific ship, it was possible to take the controller from the laboratory to the inland ship and have it work properly almost immediately. With this adaptive controller it will be possible to reduce the installation cost considerably.

Figure 5 shows the performance of the adaptive rudder controller on the "Arno". The table was initially filled with data corresponding to a rudder speed of 20 deg/s. The steering machine of the "Arno" appeared to have a speed of about 16 deg/s, so the results of the adaptation are less spectacular than in Figs. 3 and 4.

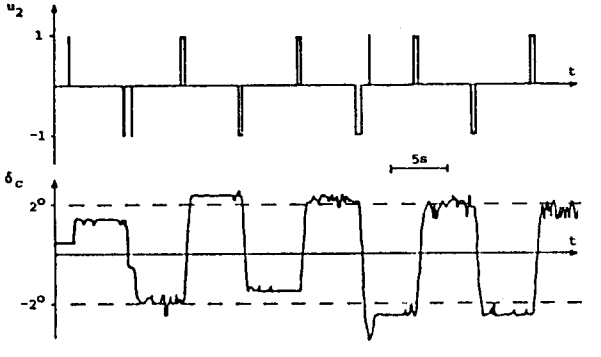

Fig. 5. Performance of the Adaptive Rudder Controller on board the "Arno"

The rudder could be positioned within 1 degree of the desired rudder angle. Higher accuracies were not possible on this particular ship due to the backlash in the steering gear. Figure 5 indicates that the measurements of the rudder angle were quite noisy, but this hardly influenced the performance. Only occasionally was more than one steering pulse necessary to position the rudder accurately when the setpoint changes were small. Additional experiments, however, indicated that the control for large moves was not as desired. This is subject to further research. These experiments also indicated that the behaviour may be not equal for port and starboard moves. In this case the table size should be doubled. Because it is only a small table, this will cause no additional problems. Nor will it require extra computation time.

6. CONCLUSIONS AND SUGGESTIONS

It has been shown that it is possible to implement an adaptive controller for the positioning of the rudder of an inland ship in an inexpensive microcomputer. The adaptive algorithm will reduce the installation cost to a minimum. Although the microcomputer is inexpensive it is possible, with some restrictions, to use a high-level programming language and a real-time operating system, which reduces the development cost. During full-scale trials accurate rudder control with a minimum number of relay switchings, and consequently reduction of wear on the steering machine were achieved.
It can be concluded that an adaptive controller which uses table lookup is well suited to use in inexpensive hardware.

The design of the adaptive rudder controller has been the first stage in the development of the adaptive autopilot for inland ships. An adaptive autopilot algorithm still has to be implemented. This is now being developed. The autopilot algorithm is less time critical than the rudder controller because it can have a much slower sampling rate.

ACKNOWLEDGEMENTS

These investigations were supported (in part) by the Netherlands Foundation for Technical Sciences (STW), future Technical Science Branch of the Netherlands Organization for the Advancement of Pure Research (ZWO).

The authors wish to acknowledge STW, Egas Scheepsdiesel- en Elektrotechniek B.V., Kuipers Electronic Engineering B.V. and the skipper of the "Arno" for their support to obtain the results described in this paper.

REFERENCES

Amerongen, J. van, G. Honderd (1983), A robust adaptive controller for relay-actuated processes. Yale Workshop on Applications of Adaptive Systems Theory, 1983, pp. 234-238.

Amerongen, J. van (1982), Adaptive Steering of Ships: a model-reference approach to improved manoeuvring and economical course keeping, Ph. D. thesis, Delft University of Technology.

Bouman, R.J. (1981), An Autopilot for Inland Ships (Dutch), Delft University of Technology.

Hartman, J.L., F.N. Bailey (1981). Adaptive position controller avoids complex algorithms. Control Engineering, May 1981, pp. 71-74.

Fig. 6. The "Arno"