Monograph:
Kyriacou, T., Nehmzow, U., Igelsiais, R. et al. (1 more author) (2005) Cross-Platform Programming Through System Identification. Research Report. ACSE Research Report 895 . Department of Automatic Control and Systems Engineering

---

Cross-Platform Programming Through System Identification

T Kyriacou[#], U Nehmzow[#], R Iglesias[#], S A Billings

#Dept Computer Science, University of Essex

Department of Automatic Control and Systems Engineering
The University of Sheffield, Sheffield, S1 3JD, UK

Research Report No. 895

May 2005

Cross-platform programming through system identification

Theocharis Kyriacou[1], Ulrich Nehmzow[1], Roberto Iglesias[2], and Steve Billings[3]

[1]Department of Computer Science, University of Essex, United Kingdom.
[2]Department of Electronics and Computer Science,
University of Santiago de Compostela, Spain.
[3]Department of Automatic Control and Systems Engineering,
University of Sheffield, United Kingdom.

Abstract

In this paper we present the RobotMODIC procedure (robot modelling, identification and characterisation), as a process which lets us identify the robot's motion through a non-linear polynomial function (Narmax). This procedure represents an important step towards a science of mobile robots, because the Narmax function can be analysed to understand the underlying phenomena governing the robot's behaviour, and it also can be used to subsequently control the movement of the robot.

As we'll see through this paper, the RobotMODIC procedure has another important benefit, it allows very easy and fast cross-platform transfer of robot code. The mathematical description of robot's behaviour (in the form of a polynomial function), can be easily exchanged between different robot platforms, thus essentially "programming" mobile robots not through the explicit definition of a control mechanism, but through a model derived from the observation of robot's motion. In this way, the RobotMODIC procedure produces an extremely compact code which contributes towards saving in memory space and reduces processing time on the host robot during execution.

1. Introduction

Fundamentally, the behaviour of a robot is influenced by three components: i) the robot's hardware, ii) the program it is executing and iii) the environment it is operating in. A robot program written with a specific task in mind almost never produces the desired robot behaviour right from the beginning. This is because many idealistic assumptions are made as far as the the robot's hardware and the environment are concerned.

We believe that there are currently two major problems in mobile robotics research: i) the lack of a theory-based design methodology for mobile robot control programs, and ii) the lack of accurate mobile robot models as a design tool.

The lack of a formal design methodology, based on a theory of robot-environment interaction (which would allow the methodical design of mobile robot control programs) means that control programs have to be developed through an empirical trial-and-error process. This is costly, time-consuming and error prone. In addition, the lack of a theoretical foundation for mobile robotics means that development tools have to be based on general assumptions (e.g. idealised, simplified models of sensors) that commonly result in significant discrepancies between predicted and actually observed behaviour of the physical mobile robot.

As part of the RobotMODIC project conducted at the universities of Essex and Sheffield, in this paper we propose a novel procedure to exchange control programs between different robot platforms, based on system identification techniques. System identification lets us identify and model the behaviour of a robot driven manually (Iglesias et al., 2005), or using a control program (Iglesias et al., 2004), through a non-linear polynomial function (NARMAX). This function identifies the relevant input-output parameters of the robot control process, and can be used afterwards, during autonomous operation, to determine the translational and rotational velocities that are suitable for the desired behaviour. Given that this function is amenable to mathematical analysis, it can be used to understand better some important properties of the robot's behaviour. This is an important step towards the development of a theory of robot-environment interaction that supports the off-line design of robot controllers, and that makes testable and falsifiable hypotheses about that interaction.

On the other hand, as the NARMAX function encodes the control of the robot, it can be directly imple-

mented in any computer language, resulting in a very compact code, which is useful for applications where memory and processing speed matter. As we'll see in this article, in many cases the generated code is platform-independent and can be used without modifications on different robots with similar configuration and kind of sensors.

As the developed code is based on a mathematical function which relates the motor responses of the robot with its sensor perceptions, it can be more easily shared between different research groups. This makes the design of mobile robot controllers more transparent, as the control strategy is clearly expressed in closed form. Furthermore, unlike opaque models (such as for instance artificial neural networks, or fuzzy controllers), transparent models have obvious benefits when reasons for any failures need to be established.

In this paper we explain how we used the NARMAX process to obtain a model of a wall-following task. The task was originally executed by a Magellan Pro mobile robot running in a real environment setup in the robotics lab at the University of Essex, UK. We have then sent the task model to colleagues in the Department of Electronics and Computer Science of the University of Santiago de Compostela, Spain, to run it on their Nomad 200 mobile robot. The Nomad robot was operated in the corridors of the department.

2. Experimental procedure and methods

2.1 *Experimental procedure*

Our first goal was to identify the behaviour of a Magellan Pro mobile robot following a wall. The experimental scenario is the one shown in figure 1. The Magellan Pro is equipped with front-facing laser, sonar, infrared, tactile and vision sensors (figure 2).

During execution of the task under investigation we log information every 250ms that define the state of the robot. The logged data includes all sensor and actuator values of the robot. We also log the position and orientation of the robot using an overhead camera in order to avoid drift errors associated with the robot's odometry measurements. Figure 1 shows an overhead image of one such experimental setup.

After the collection of the experimental data a NARMAX model is estimated that relates robot sensor values to robot actuator signals. The choice of which sensor modalities to use as inputs and which actuators to use as outputs depends on the nature of the task we intend to model. For example, to model a wall-following task, the robot's sonar distance sensors can be used as inputs and the robot's rotational velocity can be used as the output. The general rule in choosing suitable inputs and outputs is that at least some of the inputs should be causing the output.

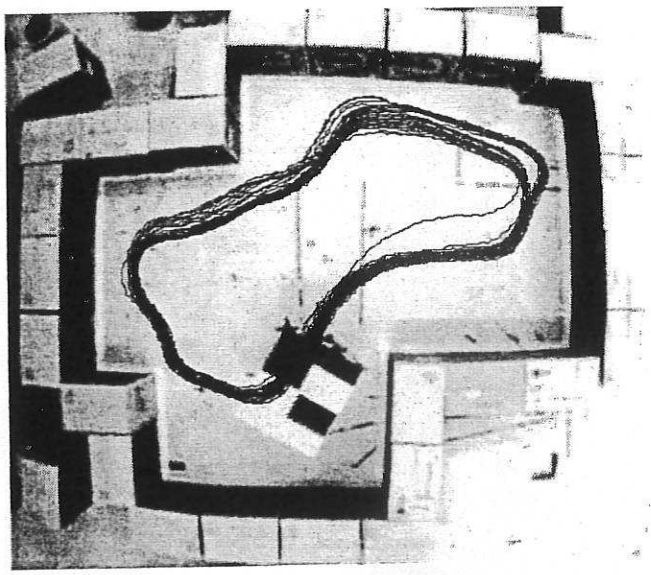

Figure 1: The environment in which experiments were conducted and the robot's trajectory. The robot is visible in the bottom of the image. The dimensions of the navigable space in the environment are approximately 3.3m x 2.8m. The robot is executing a wall-following behaviour (following the right wall), i.e. the direction of motion is counter-clockwise with respect to the image.

The NARMAX model estimation methodology is explained in the next section.

2.2 *NARMAX Modelling*

The NARMAX modelling approach is a parameter estimation methodology for identifying both the important model terms and the parameters of unknown nonlinear dynamic systems. For multiple input, single output noiseless systems this model takes the form:

$$
\begin{aligned}
y(n) \;=\; & f(u_1(n), u_1(n-1), u_1(n-2), \cdots, u_1(n-N_u), \\
& u_1(n)^2, u_1(n-1)^2, u_1(n-2)^2, \cdots, u_1(n-N_u)^2, \\
& \cdots, \\
& u_1(n)^l, u_1(n-1)^l, u_1(n-2)^l, \cdots, u_1(n-N_u)^l, \\
& u_2(n), u_2(n-1), u_2(n-2), \cdots, u_2(n-N_u), \\
& u_2(n)^2, u_2(n-1)^2, u_2(n-2)^2, \cdots, u_2(n-N_u)^2, \\
& \cdots, \\
& u_2(n)^l, u_2(n-1)^l, u_2(n-2)^l, \cdots, u_2(n-N_u)^l, \\
& \cdots, \\
& \cdots, \\
& u_d(n), u_d(n-1), u_d(n-2), \cdots, u_d(n-N_u), \\
& u_d(n)^2, u_d(n-1)^2, u_d(n-2)^2, \cdots, u_d(n-N_u)^2, \\
& \cdots,
\end{aligned}
$$

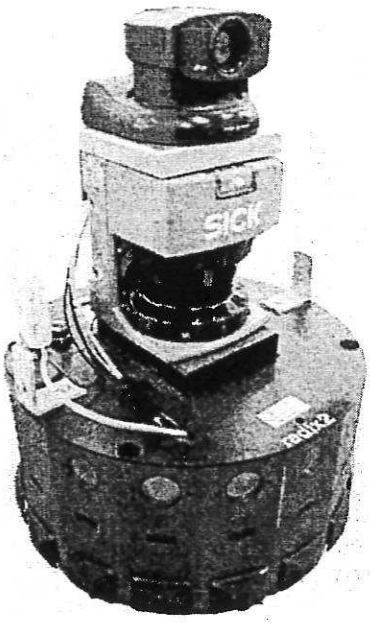

Figure 2: *Radix*, the Magellan Pro mobile robot used in the experiments described in this paper.

$$
\begin{aligned}
&u_d(n)^l, u_d(n-1)^l, u_d(n-2)^l, \cdots, u_d(n-N_u)^l, \\
&y(n-1), y(n-2), \cdots, y(n-N_y), \\
&y(n-1)^2, y(n-2)^2, \cdots, y(n-N_y)^2, \\
&\cdots, \\
&y(n-1)^l, y(n-2)^l, \cdots, y(n-N_y)^l)
\end{aligned}
$$

were $y(n)$ and $u(n)$ are the sampled output and input signals at time n respectively, N_y and N_u are the regression orders of the output and input respectively and d is the input dimension. $f()$ is a non-linear function and it is typically taken to be a polynomial or wavelet multiresolution expansion of the arguments. The degree l of the polynomial is the highest sum of powers in any of its terms.

The NARMAX methodology breaks the modelling problem into the following steps: i) Structure detection, ii) parameter estimation, iii) model validation, iv) prediction, and v) analysis. These steps form an estimation toolkit that allows the user to build a concise mathematical description of the system. These procedures are now well established and have been used in many modelling domains (Billings and Chen, 1998).

A detailed procedure of how structure detection, parameter estimation and model validation is done is presented in (Korenberg et al., 1988, Billings and Voon, 1986). A brief explanation of these steps is given below.

Any data set that we intend to model is first split in two sets (usually of equal size). The first is called the estimation data set and it is used to calculate the model parameters. The remaining data set is called the validation set and it is used to test and evaluate the model.

The structure of the NARMAX polynomial is determined by the inputs $\boldsymbol{u}$, the output y, the input and output orders N_u and N_y respectively and the degree l of the polynomial. The number of terms of the NARMAX model polynomial can be very large depending on these variables, but not all of them are significant contributors to the computation of the output. In fact most terms can be safely removed from the model equation without this introducing any significant errors.

The calculation of the NARMAX model parameters is an iterative process. Each iteration involves three steps: i) estimation of model parameters, ii) model validation and iii) removal of non-contributing terms.

In the first step the NARMAX model is used to compute an equivalent auxiliary model whose terms are orthogonal, thus allowing their associated parameters to be calculated sequentially. This increases the speed of computation of the model parameters. Once the parameters of the auxiliary model are obtained the NARMAX model parameters are computed from the auxiliary model.

The NARMAX model is then tested using the validation data set. If there is no significant error between the model predicted output and the actual output, non-contributing model terms are removed in order to reduce the size of the polynomial as much as possible. This is primarily done to increase the speed of computation of the model output during its use.

To determine the contribution of a model term to the output the Error Reduction Ratio (ERR) (Korenberg et al., 1988) is computed for each term. The ERR of a term is the percentage reduction in the total mean-squared error (i.e. the difference between model predicted and true system output) as a result of including (in the model equation) the term under consideration. The bigger the ERR is, the more significant the term. Model terms with ERR under a certain threshold are removed from the model polynomial.

In the following iteration, if the error is higher as a result of the last removal of model terms then these are re-inserted back into the model equation and the model is considered as final.

3. Experimental results

This section presents an example of model exchanging between different robot platforms. First, the task under investigation (wall-following) and the data collection procedure, using the Magellan Pro mobile robot, is explained. After that, the NARMAX modelling process which led to the task model is described. In order to verify that the model obtained is a faithful representation of the task, suitable statistical tests are chosen to compare the behaviour of the robot when executing the

task and that when executing the NARMAX model of the task. Finally, the model was sent to colleagues in the Department of Electronics and Computer Science of the University of Santiago de Compostela in Spain, to run on their Nomad 200 mobile robot. The Nomad robot executed the task in the corridors of the department while continuously recording its sensor data. For comparison purposes the Nomad 200 was also run in the same environment with the original ANN controller.

3.1 *The wall-following task*

An artificial neural network (ANN) based wall-following task similar to the one described in (Iglesias et al., 1997) and (Iglesias et al., 1998) was used in this experiment to drive the robot. An overhead image of the lab environment used to collect data using the Magellan Pro robot and the resulting trajectory of the robot is shown in figure 1.

The program uses as input the information coming from the sonar sensors facing the frontal obstacles and the wall being followed, and as output the appropriate robot rotational velocity ω to effect the desired wall-following behaviour (see figure 3). The transitional velocity v of the robot is kept constant at 0.08m/s.

Briefly, the wall following behaviour is achieved as follows:

To reduce the dimensionality of the input space a self-organising map (SOM) (Kohonen, 1997) is used. This reduces the input to a six-component vector $\boldsymbol{u}_c$ which is subsequently thresholded using a simple low-pass filter (LPF) to produce a binary vector $\boldsymbol{u}_d$. This thresholding is done to further reduce the input space to 64 (2^6) possible states. The binary vector is then fed into a multilayer perceptron (MLP) which outputs the appropriate rotational velocity for the robot. The MLP is trained using data generated using a robot driven by a human operator.

At executing time, because the number of possible input states is relatively small, a lookup table is used instead of the MLP to provide the output. This is done in order to minimise computational load during the real-time execution of the code.

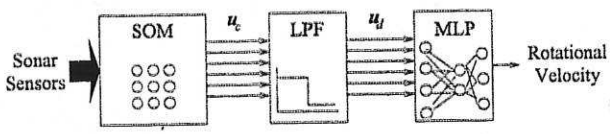

Figure 3: Diagrammatic representation of the ANN wall-following program used in the experiments conducted in this paper.

$$
\begin{aligned}
&\omega_{model}(t) = \\
&-0.282 \\
&-0.129 * (1/s_1(t)) \\
&-0.039 * (1/s_1(t-1)) \\
&-0.076 * (1/s_1(t-2)) \\
&-0.017 * (1/s_3(t)) \\
&+0.007 * (1/s_5(t)) \\
&+0.017 * (1/s_9(t)) \\
&+0.009 * (1/s_{10}(t)) \\
&-0.007 * (1/s_{12}(t-1)) \\
&+0.165 * (1/s_{13}(t)) \\
&-0.019 * (1/s_{13}(t-1)) \\
&+0.079 * (1/s_{14}(t)) \\
&-0.051 * (1/s_{15}(t)) \\
&-0.072 * (1/s_{16}(t)) \\
&+0.134 * (1/(s_1(t))^2) \\
&+0.017 * (1/(s_1(t-1))^2) \\
&+0.096 * (1/(s_1(t-2))^2) \\
&+0.001 * (1/(s_2(t))^2) \\
&+0.018 * (1/(s_7(t-2))^2) \\
&-0.019 * (1/(s_{13}(t))^2) \\
&+0.056 * (1/(s_{15}(t))^2) \\
&+0.099 * (1/s_{16}(t))^2 \\
&+0.063 * (1/(s_1(t-1) * s_{16}(t-1)) \\
&-0.071 * (1/(s_1(t-2) * s_9(t-2)) \\
&+0.039 * (1/(s_2(t) * s_{14}(t))) \\
&-0.038 * (1/(s_2(t-1) * s_6(t) \\
&+0.059 * (1/(s_3(t-1) * s_{15}(t))) \\
&+0.003 * (1/(s_{13}(t) * s_{13}(t-1))) \\
&-0.027 * (1/(s_{13}(t) * s_{14}(t)))
\end{aligned}
$$

Table 1: The NARMAX model of the ANN wall-following task, showing the rotational velocity ω as a function of the sonar sensor values s_i, $\forall i = 1, ..., 16$.

3.2 *Task identification*

Using the procedure described in section 2.2, a model of the ANN wall-following task was obtained. In order to avoid making assumptions about the relevance of specific sonar sensors, all the ultrasound measurements were taken into account. The model structure was of input order $N_u = 2$, output order $N_y = 0$ and degree $l = 2$. This initially produced a polynomial of 1225 terms of which, after iterative refinement, only 29 remained. The final model equation is shown in table 1.

Note that, although this was not essential for the modelling process, the sonar sensor values were inverted before they were used to obtain the model. This was done as a matter of convenience in order to avoid having to deal with the sonar sensor range limitation.

3.3 *Original and model behaviour comparison*

Figure 4 compares the actual and model-predicted rotational velocities for part of the validation data set.

To validate the model further, we computed the Pearson and Spearman rank correlation coefficients r and r_S between the original rotational velocity ω_{orig} and the model-predicted rotational velocity ω_{model}. These were found to be $r = 0.81$ and $r_S = 0.74$ (both significant, $p < 0.05$), thus indicating no significant difference between ω_{orig} and ω_{model}.

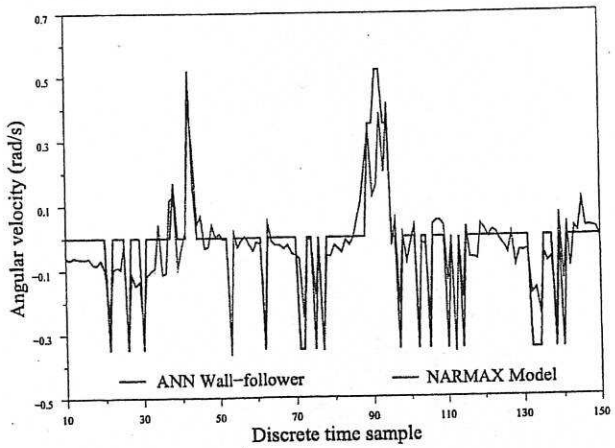

Figure 4: A plot of the actual and model-predicted rotational velocities for part of the validation data set for the wall-following behaviour.

Of course, the final and most important validation of any model of robot-environment interaction is to actually execute the model on a real robot in the target environment. We therefore used the model given in table 1 to actually drive the Magellan Pro *Radix*. The behaviour of the robot was then compared to that obtained when the original ANN wall-following program was being run. Figure 5 shows the two trajectories superimposed for the purposes of this comparison.

3.3.1 *Qualitative comparison*

By merely looking at the two robot behaviours in figure 5 we can subjectively say whether one is a faithful model of the other. We believe that this is the case here, since the model trajectory follows the original trajectory well, with few deviations. The model certainly displays a wall-following behaviour especially at the most challenging aspects of the environment, which are the convex and concave corners.

3.3.2 *Quantitative comparison*

We also performed numerical tests to determine the degree of similarity between the two robot behaviours in figure 5. Since this is a wall-following behaviour in a static environment and the robot moves with a constant

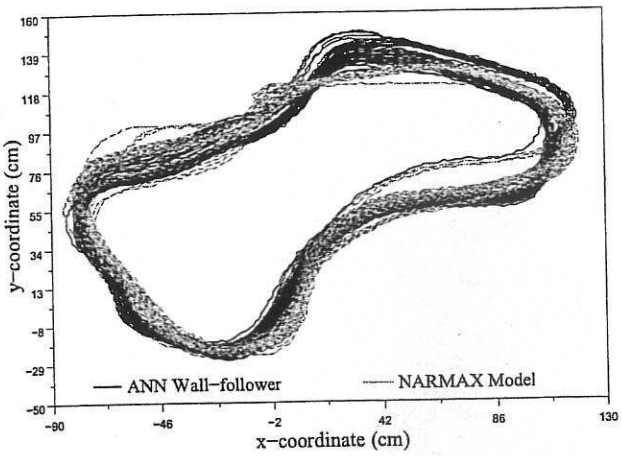

Figure 5: A comparison between the trajectory of the robot when executing the ANN wall-following program and when executing the estimated NARMAX model of this task.

transitional velocity, there are two distributions we can test for similarity for each of the two behaviours: i) the distribution of minimum robot distance to the wall and ii) the distribution of the robot's rotational velocity. Each of these statistical measures is plotted for both trajectories in figures 6 and 7 respectively.

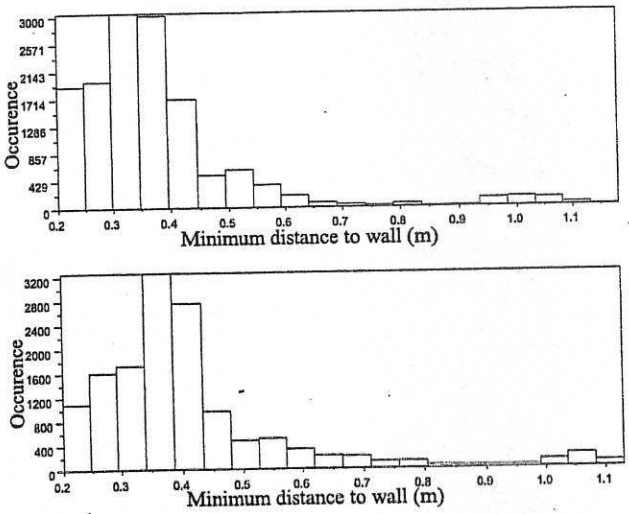

Figure 6: The distributions of the minimum robot distance to the wall for the ANN wall-following program (top) and the NARMAX model (bottom).

The Mann-Whitney U-test (Barnard et al., 1993, Snedecor and Cochran, 1989) was performed on both pairs of distributions in figures 6 and 7 in order to check whether they are significantly different or not. Both tests indicated that the distributions are not different at the 5% significance level.

Another test was also used to compare the two be-

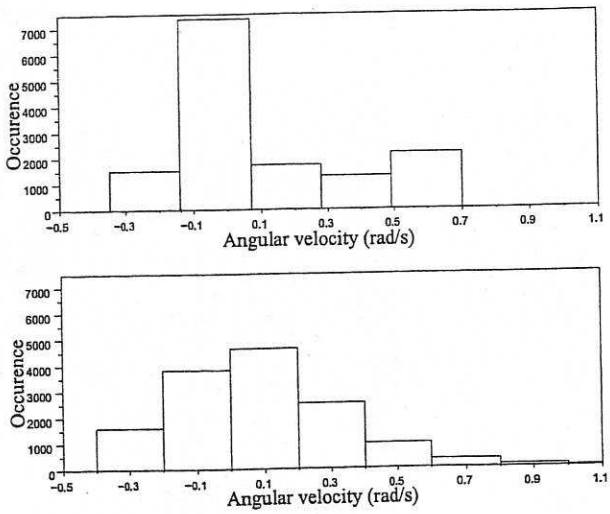

Figure 7: The distributions of the robot's rotational velocity for the ANN wall-following program (top) and the NARMAX model (bottom).

haviours in figure 5. This test compared the space occupancy of each trajectory, thus essentially treating the two trajectories as static monochromatic images. To do this the space occupancy of each trajectory along the x and y axes was compared. Figure 8 shows the distributions of the robot's position along the x-axis for both the ANN wall-follower and the NARMAX model. Figure 9 shows the same distributions for the y-axis.

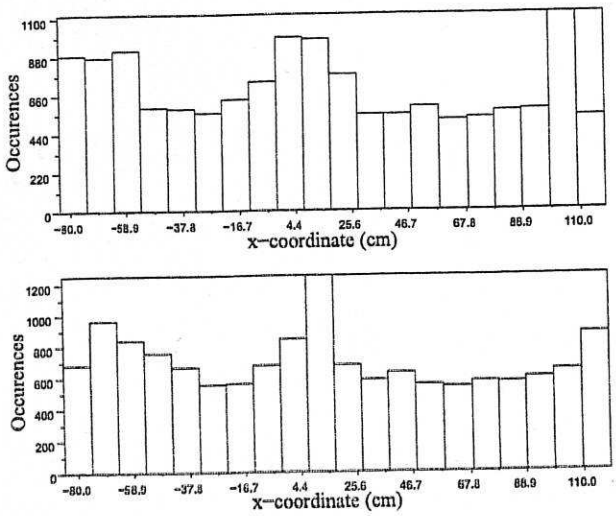

Figure 8: The distributions of the robot's x-coordinate for the ANN wall-following program (top) and the NARMAX model (bottom)

Again the Mann-Whitney U-test was used to compare each pair of distributions. Both tests indicated that the distributions are not significantly different ($p < 0.05$).

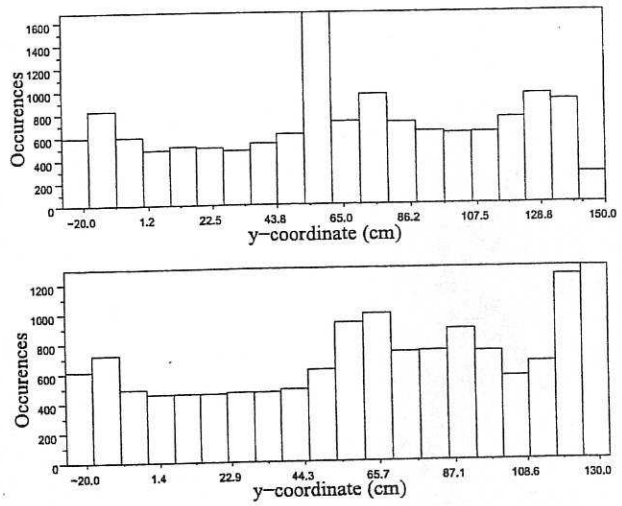

Figure 9: The distributions of the robot's y-coordinate for the ANN wall-following program (top) and the NARMAX model (bottom)

3.4 *Platform-independent programming*

Several advantages become apparent as a consequence of compressing the ANN wall-following algorithm into a single NARMAX polynomial shown in table 1. These relate to the ease of transferring the wall-following task to different programming platforms within the same robot or even different robot platforms with the same sensor configuration.

In order to demonstrate this, we have used the NARMAX model in table 1 to run on a Nomad 200 robot (see figure 10), moving in the corridors of the department of Electronics and Computer Science of the University of Santiago de Compostela, Spain. Like the Magellan Pro, the Nomad 200 robot uses 16 sonar sensors distributed around it (one per 22.5 degrees).

To verify that the Nomad robot behaves in a similar fashion as if the ANN wall-following code was executed, sensor data was also collected with the original task code. It is important to note that the environment in both cases was unchanged, (figure 11).

As with the Magellan Pro robot, in order to compare the behaviour of the Nomad robot when running the original task and the task model, we used the Mann-Whitney U-test to compare the minimum distance to the wall and the rotational velocity of the robot. Both tests revealed no significant difference ($p < 0.05$) between the two pairs of distributions shown in figures 12 and 13.

A comparison of the trajectory of the Nomad robot when executing the original code and the model could not be made in this case because there were no reliable means to accurately record the position of the robot during its operation.

It is important to note that, even though the same task

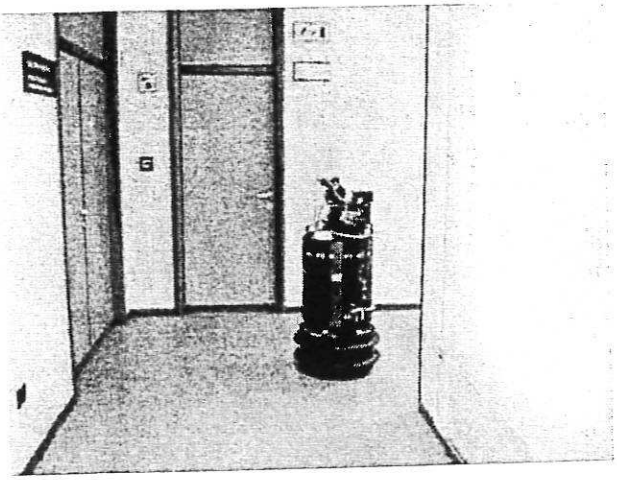

Figure 10: The Nomad 200 robot while executing the wall-following task.

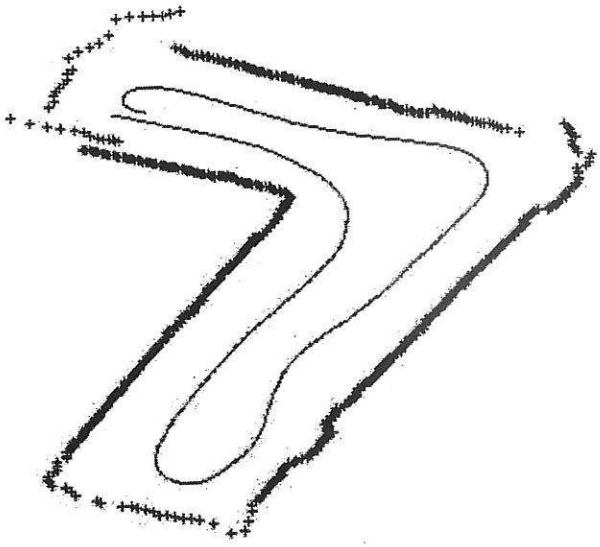

Figure 11: Trajectory followed by the Nomad robot in its first lap in a L-shaped environment, and when the Narmax model was used to control it. The symbol '+' reflects the sensor measurements provided by one of the lateral sensors during this first lap.

was executed in two robots with very similar sonar sensor specifications, when a comparison is made between the behaviour of the Magellan Pro robot and the Nomad robot, a significant difference is observed. This can be seen when comparing figures 6 and 12 or figures 7 and 13. This difference is also confirmed with the Mann-Whitney U-test. This is because the environments where the experiments were carried out and the robots were different. In the first case, the Magellan Pro robot was run in a closed environment build out of carton boxes (see figure 1) whereas in the second case, the Nomad robot was operated in a corridor with open or closed doors (see figure 10). The fact that the robots are different may also affect in the trajectories observed. In this case the difference between the command sent to the robot every 250 milliseconds and the real velocities achieved by the motors may differ between the Magellan and the Nomad.

4. Summary and Conclusions

In this paper we have shown how the NARMAX modelling approach can be used to obtain a non-linear polynomial model of the robot's task in a real robot-environment-task system. We have also shown how these models can be validated in order to prove that they are faithful representations of the task program codes and so they can used to make realistic predictions of the robot's behaviour. A transparent model of the task has several advantages over opaque models (such as those produced using artificial neural networks). Perhaps the most important advantage is that it can be directly analysed using established mathematical methods.

As an example, an artificial neural network based wall-following task was modelled and validated in this paper. This has demonstrated the applicability of the NARMAX approach in situations where a polynomial representation cannot be derived from the original code.

Both qualitative and quantitative approaches were followed in order to validate the estimated model of the wall-following task. No attempt has been made in the past to analyse robotic behaviour quantitatively. Here we do this by selecting to compare statistical measures which are relevant to the task being modelled. In the case of the wall-following task these were: i) the distance of the robot to the wall and ii) the robot's rotational velocity.

Several advantages become apparent as a consequence of compressing the ANN wall-following algorithm into a single NARMAX polynomial shown in table 1. These relate to the ease of transferring the wall-following task to different programming platforms within the same robot or even different robot platforms with the same sensor configuration. As an example of one such cross-platform implementation, we have used the NARMAX model in table 1 to run on a Nomad 200 robot, moving in the corridors of the department of Electronics and Computer Science of the University of Santiago de Compostela in Spain. The robot performed identically when running the original ANN code and the NARMAX model code. The most interesting aspect of this experiment was the small amount of time spent, between us and our colleagues in Spain, in communicating the task and setting up the robot to run using the NARMAX model.

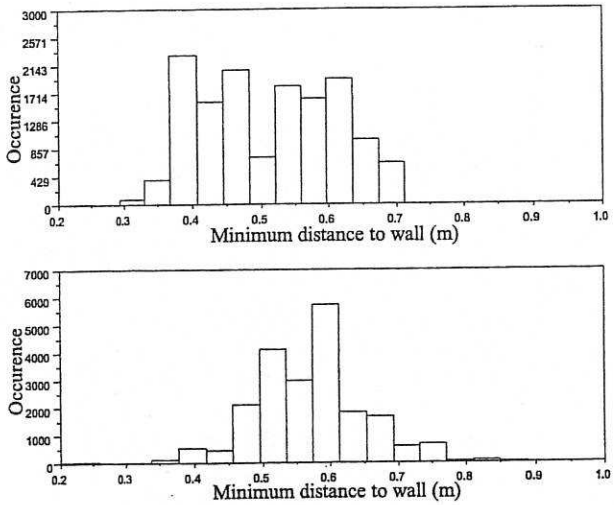

Figure 12: The distributions of the minimum robot distance to the wall for the ANN wall-following program (top) and the NARMAX model (bottom) for the Nomad 200 robot.

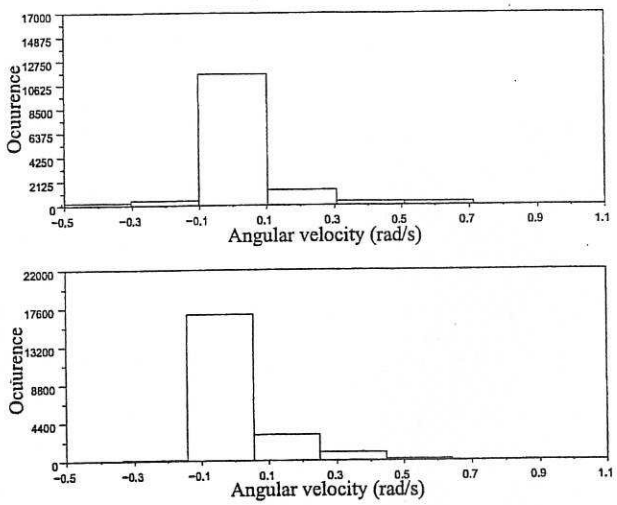

Figure 13: The distributions of the robot's rotational velocity for the ANN wall-following program (top) and the NARMAX model (bottom) for the Nomad 200 robot.

Acknowledgement

The authors would like to thank the following institutions for their support:
The RobotMODIC Project is supported by the Engineering and Physical Sciences Research Council (EPSRC) grand GR/S30955/01 under the Mathfit initiative. Roberto Iglesias is supported by research grants PGIDIT04TIC206011PR and TIC2003-09400-C04-03.

References

Barnard, C., Gilber, F., and McGregor, P. (1993). *Asking questions in Biology. Design, analysis & presentation in practical Work.* Longman Sicientific & Technical.

Billings, S. and Chen, S. (1998). The determination of multivariable nonlinear models for dynamical systems. In Leonides, C., (Ed.), *Neural Network Systems, Techniques and Applications*, pages 231–278. Academic press.

Billings, S. and Voon, W. S. F. (1986). Correlation based model validity tests for non-linear models. *International Journal of Control*, 44:235–244.

Iglesias, R., Kyriacou, T., Nehmzow, U., and Billings, S. (2004). Task identification and characterisation in mobile robotics. In *Proceedings of TAROS 2004, "Towards Autonomous Robotic Systems"*. Springer Verlag.

Iglesias, R., Kyriacou, T., Nehmzow, U., and Billings, S. (2005). Robot programming through a combination of manual training and system identification. In *ECMR 05 - European Conference on Mobile Robots 2005. Submitted.* Springer Verlag.

Iglesias, R., Regueiro, C. V., Correa, J., and Barro, S. (1997). Implementation of a basic reactive behaviour in mobile robotics through artificial neural networks. In *Biological and Artificial Computation: From Neuroscience to Technology.* Springer.

Iglesias, R., Regueiro, C. V., Correa, J., Schez, E., and Barro, S. (1998). Improving wall following behaviour in a mobile robot using reinforcement learning. In *Proc. of the International ICSC Symposium on Engineering of Intelligent Systems.* ICSC Academic Press.

Kohonen, T. (1997). *Self-Organizing Maps.* Springer, second edition.

Korenberg, M., Billings, S., Liu, Y. P., and McIlroy, P. J. (1988). Orthogonal parameter estimation algorithm for non-linear stochastic systems. *International Journal of Control*, 48:193–210.

Snedecor, G. W. and Cochran, W. G. (1989). *Statistical Methods.* Iowa State Press, 8th edition.