This is a repository copy of *An application of lyapunov stability analysis to improve the performance of NARMAX models*.

White Rose Research Online URL for this paper:
http://eprints.whiterose.ac.uk/74647/

Monograph:
Akanyeti, O., Rano, I., Nehmzow, U. et al. (1 more author) (2009) An application of lyapunov stability analysis to improve the performance of NARMAX models. Research Report. ACSE Research Report no. 994 . Automatic Control and Systems Engineering, University of Sheffield

Reuse
Unless indicated otherwise, fulltext items are protected by copyright with all rights reserved. The copyright exception in section 29 of the Copyright, Designs and Patents Act 1988 allows the making of a single copy solely for the purpose of non-commercial research or private study within the limits of fair dealing. The publisher or other rights-holder may allow further reproduction and re-use of this version - refer to the White Rose Research Online record for this item. Where records identify the publisher as the copyright holder, users can verify any specific terms of use on the publisher's website.

Takedown
If you consider content in White Rose Research Online to be in breach of UK law, please notify us by emailing eprints@whiterose.ac.uk including the URL of the record and the reason for the withdrawal request.

eprints@whiterose.ac.uk
https://eprints.whiterose.ac.uk/

An Application of Lyapunov Stability Analysis to Improve the Performance of NARMAX Models

O Akanyeti[1], I Rano[1], Ulrich Nehmzow[1], and S A Billings[2]
[1]Dept Computer Science, University of Ulster, Ireland
[2]Dept Automatic Control and Systems Engineering, University of Sheffield

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

Research Report No. 994

June 2009

An Application of Lyapunov Stability Analysis to Improve the Performance of NARMAX Models

Otar Akanyeti[a], Iñaki Rañó[b], Ulrich Nehmzow[c], S. A. Billings[d]

[a]*School of Computer Science and Electronic Engineering, University of Essex, UK.*
[b]*Dept. of Computer Science and Systems Engineering, University of Zaragoza, Spain.*
[c]*School of Computing and Intelligent Systems, University of Ulster, UK.*
[d]*Department of Automatic Control and Systems Engineering, University of Sheffield, UK.*

Abstract

Previously we presented a novel approach to program a robot controller based on system identification and robot training techniques. The proposed method works in two stages: first, the programmer demonstrates the desired behaviour to the robot by driving it manually in the target environment. During this run, the sensory perception and the desired velocity commands of the robot are logged. Having thus obtained training data we model the relationship between sensory readings and the motor commands of the robot using ARMAX/NARMAX models and system identification techniques. These produce linear or non-linear polynomials which can be formally analysed, as well as used in place of "traditional robot" control code.

In this paper we focus our attention on how the mathematical analysis of NARMAX models can be used to understand the robot's control actions, to formulate hypotheses and to improve the robot's behaviour. One main objective behind this approach is to avoid trial-and-error refinement of robot code. Instead, we seek to obtain a reliable design process, where program design decisions are based on the mathematical analysis of the model describing how the robot interacts with its environment to achieve the desired behaviour. We demonstrate this procedure through the analysis of a particular task in mobile robotics: door traversal.

1. Introduction

Sensor-motor couplings form the backbone of most mobile robot control tasks, and often need to be implemented fast, efficiently and reliably. Robot training is a fast and efficient method of obtaining robot control code [1, 2, 3]. It is also an alternative to manual programming based on empirical trial-and-error processes. This process involves the programmer writing the program, testing it on the robot and refining the code constantly until the behaviour of the robot resembles the desired behaviour within the desired degree of accuracy. Machine learning techniques, such as artificial neural networks are often used to obtain the desired sensor-motor competences [4, 5, 6, 7, 8]. However, representing the relationship between perception and action using neural networks has the disadvantage of being an opaque mechanism, which does not reveal how the desired behaviour is achieved using the robot's perception, in other words: opaque models such as artificial neural network are not easily comprehensible to humans. Therefore, we are not able to analyze the generated controllers from a mathematical point of view and learn more about how the robot interacts with its environment.

Our understanding of robot-environment interaction is so limited that so far we are not able to find answers to questions like "Which sensors are the most important for the desired behaviour?", "What would happen if one of the robot sensors broke?" or "Is it possible to predict the behaviour of the robot beforehand in untested environments?". Furthermore, when the obtained controllers are not comprehensible by humans, or they cannot be expressed mathematically, they neither can be analysed using mathematical tools. The only way of evaluation is to test them in the target environment and measure their performance based on some performance metrics [9]. For example if the task under investigation is to follow a wall from a certain distance, the perpendicular distance between the robot and the wall is one such performance metric [10]. If the desired task is to traverse through door-like openings, one test option is to start the robot from many different initial positions and see if it passes through the door successfully in each attempt [11].

However, these kind of performance measures do not validate the obtained controllers from a theoretical point of view. For instance, 20 successful attempts of door traversing do not guarantee that the 21st will be successful, too. Moreover, when the robot fails to achieve the desired task, the analysis of the controllers is even more complex, and so far there is no formal method of error-debugging or controller optimization. The more robots are to be used in close interaction with humans, the more important will be the safe operation of the robot. One way to address the safety issue is to develop transparent and analysable controllers, so that they can be evaluated in terms of safety, efficiency and robustness, using mathematical tools.

In [11, 12] we presented a novel approach to program a robot controller, which produces transparent, human comprehensible models using mathematical functions. The proposed method

Email addresses: oakany@essex.ac.uk (Otar Akanyeti), irano@unizar.es (Iñaki Rañó), u.nehmzow@ulster.ac.uk (Ulrich Nehmzow), s.billings@sheffield.ac.uk (S. A. Billings)

works in two stages: first, the robot is driven manually, demonstrating the desired behaviour. During this run, sensor readings and the actions of the robot are logged. Then we model the relationship between sensory readings and the motor commands of the robot using ARMAX/NARMAX system identification techniques [13, 14]. These produce linear or non-linear polynomials, which can be used in place of "traditional" robot control code, but have the further advantage that they can be formally analysed.

In this paper we focus our attention on how the mathematical analysis of NARMAX models can be used to understand the robot's control actions, to formulate hypotheses and to improve the robot's behaviour. In particular, we demonstrate that the performance of the polynomial models obtained for the episodic task of "door traversal" can be improved using Lyapunov stability analysis.

The rest of the paper is organised as follows: Section 2 briefly introduces the NARMAX models and training algorithm used in our methodology. The methodology is applied to the door traversal task in Section 3 using raw sensor readings, and the performance of the robot traversing the door is also evaluated. A new controller is obtained for a set of human readable variables as inputs in Section 4. The NARMAX model is validated and a simple stability analysis is performed over it. Finally, conclusions and future research are presented in Section 5.

2. NARMAX System Identification Methodology

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:

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

where $y(n)$ and $\mathbf{u}(n) = (u_1(n), \cdots, u_d(n))$ are the sampled output and input signals at time n respectively, n_y and n_u are the maximum regression orders of the output and input respectively and d is the dimension of the input vector. The function $f(\cdot)$ is a non-linear function of its arguments, in our case a polynomial expansion. Other expansions such as multi-resolution wavelets or Bernstein coefficients could be used as an alternative to the polynomial expansions considered in this study.

2.1. Polynomial NARMAX Models

The polynomial representation of a NARMAX model is the set of monomials of x. For example, a 2-input, second order polynomial with no input and output lag ($n_y = n_u = 0$) takes the form:

$$\begin{aligned} y(n) \;=\; & C_0 + C_1u_1(n) + C_2u_2(n) \\ & + C_3u_1(n)u_2(n) + C_4u_1^2(n) + C_5u_2^2(n), \end{aligned} \quad (1)$$

where $C_0, \cdots, C_5$ are the coefficients of each monomial term. They must be estimated using an estimation algorithm like the Orthogonal Parameter Estimation (described in section 2.2.1) [15]. It has been shown in [16] that any continuous function in a closed interval can be approximated by a polynomial. In fact, any continuous real function can be approximated within an arbitrary degree of accuracy by a polynomial of sufficient order (Stone-Weierstrass theorem) .

2.2. Identification of Polynomial NARMAX Models

The first step towards modelling a particular system using a NARMAX model structure is to select appropriate inputs $\vec{u(n)}$ and output $y(n)$. The general rule in choosing suitable inputs and outputs is that there must be a causal relationship between the input signals and the output response.

After the choice of suitable inputs and outputs, the NARMAX method breaks the modelling problem into the following steps: i) polynomial model structure detection — determining input lag n_u and output lag n_y and the order of the polynomial l, ii) model parameter estimation and iii) model validation.

For the first step it is useful to have some knowledge about the system to identify, or at least a proper guess of the polynomial order and the regression orders. The last two steps are performed iteratively using two sets of collected data: (a) the estimation and (b) the validation data set. Usually a single data set collected in one long session is split in half, and the second half is used for this purpose.

2.2.1. Orthogonal Parameter Estimation Algorithm

The orthogonal parameter estimation algorithm (OPE) [15] is a technique which allows each parameter in a polynomial NARMAX model to be estimated sequentially and independently of the other parameters in the model. The main advantage of this technique, compared to classical least square learning algorithms, is that it provides an indication of the contribution that each term in the model makes to the desired output variance. This assists the user to detect the structure of the system under investigation and yields a parsimonious system model [14].

Having parsimonious models is vital in polynomial system identification since increasing the order of the dynamic terms n_y and n_u and the order of the polynomial expansion (l) to achieve the desired prediction accuracy will result in an excessively complex model and possible ill-conditioned computations [17].

Once the structure of the polynomial model is determined, the coefficients of each term in the polynomial are determined as follows:

1. An auxiliary model is defined such that the terms in the model are orthogonal over the training data set.
2. The coefficients of the auxiliary model are estimated in the least square manner.

3. The individual contribution of each term in the auxiliary model to the desired output is measured using the error reduction ratio.
4. The terms having minimal or no contributions are deleted from the model.
5. Steps (2) to (4) are repeated until the model convergences to the desired degree of accuracy.
6. Finally the coefficients of the NARMAX model are estimated from the remaining auxiliary model.

The OPE algorithm is a very well established technique and is being widely used in many control applications. More detailed discussion about parameter estimation and model validation can be found in [15] [18] and [19].

3. Experiment: Door Traversal

In mobile robotics applications, the NARMAX system identification method has been applied to generate various sensor-motor tasks such as wall following [10], following a moving object [20], and route learning [12]. The task investigated in this paper is to model the sensor-motor couplings of a hard-wired controller designed to achieve the episodic task of "door traversal" where each episode comprises the movement of the robot from a starting position to a final position, traversing through a door-like opening.

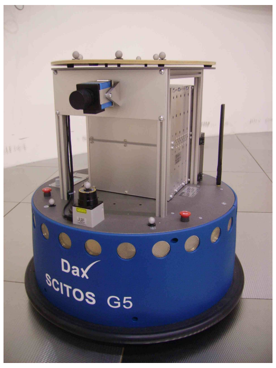
3.1. Experimental Scenario and Acquisition of the Training Data

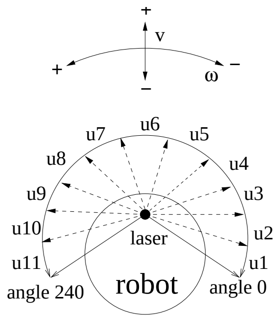
The experiments were conducted in the 100 square meter circular robotics arena of the University of Essex. We used a *Scitos G5* mobile robot called DAX (Figure 1(a)). The robot is equipped with a Hokuyo laser range finder which can deliver distance readings up to 4 meters ($0 \leq d \leq 4$). This range sensor has a wide angular range (240°) with a radial resolution of 0.36° and distance accuracy of better than 1 cm. The base of the robot is circular, its diameter is 60 cm.

The experimental scenario is presented in Figure 2, where the width of the opening is 1 m. First we designed a hard-wired door traversal control program manually, such that the controller computes the linear and angular velocities of the robot to make it approach to the door and pass through it safely. This control program relies only on the current laser scan.

Instead of driving the robot towards the door, the controller was designed to make it approach a point slightly in front of the door with an angle orthogonal to the door. Both speeds were controlled to reach the configured point and then cross the door following a straight line trajectory.

We then let the hard-wired controller drive the robot to collect the training data. The robot was started from 23 different initial positions, so that the door was always visible to the robot. In each run, laser readings and the angular velocity of the robot were logged every 250 ms, forming the training data set. Figure 3 shows the trajectories of the robot driven by the hard-wired controller.


(a) Scitos G5 mobile robot Dax



(b) Sensor coding and motion configuration

Figure 1: The robot has two degrees of freedom and equipped with a Hokuyo laser sensor. The range finder has a wide angular range (240°) with a radial resolution of 0.36° and distance accuracy of better than 1 cm. During the experiments, in order to decrease the dimensionality of the input space, we coarse-coded the laser readings into 11 sectors (u_1 to u_{11}) by averaging 62 readings for each 22 degree intervals.

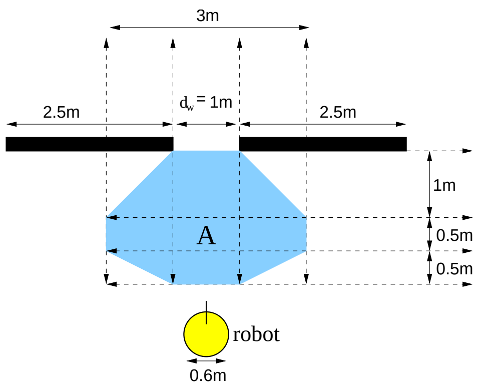


Figure 2: The experimental scenario used in the door traversal experiments. The region A indicates possible starting positions of DAX. The width of the opening is 1 m and the base of the robot is 0.6 m. The length of the walls on each side of the opening is 2.5 m.

3.2. Modelling the Door Traversal Controller

Having obtained the training data set, we coarse-coded the laser readings into 11 sectors by averaging 62 readings for each 22° interval in order to decrease the dimensionality of the input space (Figure 1(b)). We then used the coarse-coded laser readings to model the rotational velocity of the robot using NARMAX system identification method.

In the experimental setup, the relative position and orientation of the robot with respect to the door was such that the opening was always detected by the robot's laser scanner (if the door were occluded at any time, some internal state would be necessary to maintain the robot's movement towards the door, but since the laser scanner always perceives the door at each position, the robot always has the necessary sensor information to drive through the passage, and there is no need of including state or regression).

Therefore, the polynomial model was chosen to be

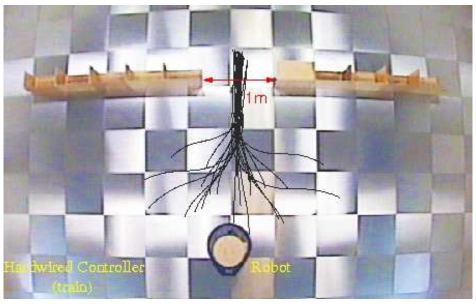


Figure 3: The trajectories of the robot under the control of hard-wired door traversal controller. The robot was started from 23 different starting positions facing the direction of the door. In each run, the laser readings and the angular velocity of the robot were logged every 250 ms in order to form the training data set.

regression-less and a second degree expansion was used in the inputs. The order of the polynomial was automatically selected by the Orthogonal Parameter Estimation algorithm such that the model error did not improve significantly when including new terms. The result is a non-linear polynomial model with 49 terms, given in Table 1.

$$
\begin{aligned}
\omega_{pol}(n) \quad = \quad & -6.698 + 0.670 \cdot u_1(n) - 0.113 \cdot u_2(n) + 0.372 \cdot u_3(n) \\
& +0.515 \cdot u_4(n) + 0.049 \cdot u_5(n) + 0.344 \cdot u_6(n) + 0.032 \cdot u_7(n) \\
& +0.013 \cdot u_8(n) + 0.310 \cdot u_9(n) + 0.251 \cdot u_{10}(n) + 0.829 \cdot u_{11}(n) \\
& +0.014 \cdot u_2^2(n) + 0.004 \cdot u_3^2(n) - 0.001 \cdot u_4^2(n) + 0.008 \cdot u_5^2(n) \\
& +0.001 \cdot u_7^2(n) - 0.003 \cdot u_{10}^2(n) + 0.012 \cdot u_1(n) \cdot u_2(n) \\
& +0.001 \cdot u_1(n) \cdot u_3(n) - 0.012 \cdot u_1(n) \cdot u_4(n) - 0.039 \cdot u_1(n) \cdot u_5(n) \\
& -0.010 \cdot u_1(n) \cdot u_6(n) + 0.008 \cdot u_1(n) \cdot u_7(n) - 0.089 \cdot u_1(n) \cdot u_9(n) \\
& -0.032 \cdot u_1(n) \cdot u_{10} + 0.025 \cdot u_2(n) \cdot u_3(n) + 0.002 \cdot u_2(n) \cdot u_4(n) \\
& -0.032 \cdot u_2(n) \cdot u_6(n) + 0.009 \cdot u_2(n) \cdot u_9(n) + 0.008 \cdot u_3(n) \cdot u_4(n) \\
& -0.021 \cdot u_3(n) \cdot u_6(n) - 0.009 \cdot u_3(n) \cdot u_7(n) + 0.005 \cdot u_3(n) \cdot u_9(n) \\
& +0.112 \cdot u_3(n) \cdot u_{11}(n) - 0.031 \cdot u_4(n) \cdot u_6(n) - 0.011 \cdot u_4(n) \cdot u_7(n) \\
& +0.001 \cdot u_4(n) \cdot u_8(n) - 0.094 \cdot u_4(n) \cdot u_{11}(n) - 0.009 \cdot u_5(n) \cdot u_7(n) \\
& +0.012 \cdot u_5(n) \cdot u_8(n) + 0.009 \cdot u_5(n) \cdot u_9(n) - 0.004 \cdot u_5(n) \cdot u_{11}(n) \\
& -0.007 \cdot u_6(n) \cdot u_7(n) + 0.007 \cdot u_6(n) \cdot u_9(n) - 0.005 \cdot u_6(n) \cdot u_{11}(n) \\
& +0.009 \cdot u_7(n) \cdot u_{10}(n) + 0.020 \cdot u_7 n \cdot u_{11}(n) - 0.029 \cdot u_{10}(n) \cdot u_{11}(n)
\end{aligned}
$$

Table 1: Polynomial model for door traversal behaviour where $\omega_{pol}(n)$ is the angular velocity of the robot and $u_1(n)$ to $u_{11}(n)$ are the coarse-coded laser readings at time step n.

3.3. Model Validation

Once we obtained the turning speed model ω_{pol}, we let the model drive the robot in the training environment. The linear velocity of the robot was kept the same as the linear velocity of the hard-wired door traversal controller. The robot was started from 20 different positions always facing the direction of the door. Figure 4 shows the trajectories of the robot under the control of ω_{pol}. The model was always successful driving the robot through the door.

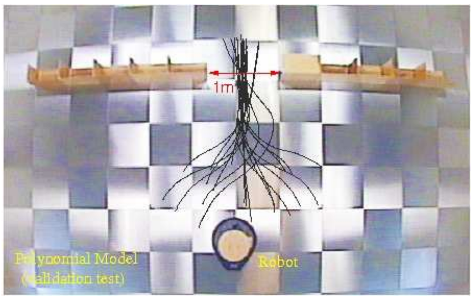


Figure 4: The performance of the robot driven by the polynomial NARMAX model ω_{pol} of Table 1. The robot was started from 20 different positions and the model was successful to drive the robot through the gap in all experiments.

3.4. Quantitative Analysis of the Model ω_{pol}

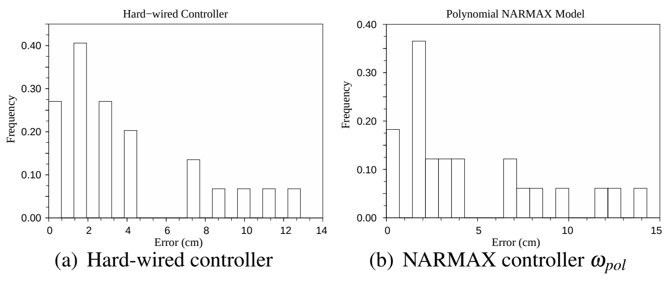
Besides the experimental validation of the controller model in the real world, using the robot, we measured some quantitative features to assess its performance. We compared the ω_{pol} model with the hand-coded door traversal program.

3.4.1. Imitating the Hard-wired Controller

In order to quantify the performance of the controller given in Table 1, we measured how well ω_{pol} was at imitating the hard-wired door traversal control program in the training region. We fed the model and the hard-wired controller with identical sensory data, and then compared the corresponding angular speed outputs quantitatively in order to measure how similar the model and the original controller are. The data logged along 60 different trajectories were used to compute the Spearman rank correlation coefficient between turning speeds generated by the hand-coded controller and the modelled robot controller (0.937, $p < 5\%$), showing a highly significant correlation.

3.4.2. Door centralization

We also measured how much the robot driven by the model ω_{pol} deviates from the center of the door while passing through it. This measure was compared with the original, hand-coded program. For each test run, we computed the horizontal distance between the center point and the trajectory of the robot during crossing the door. The histograms for both the hard-wired controller and the model ω_{pol}, showing how much the robot deviates from the center of the door (in cm) during experiments, are given in Figure 5.



(a) Hard-wired controller (b) NARMAX controller ω_{pol}

Figure 5: Distribution of deviations from the center of the door.

The median deviation from the door center was 3.024 cm (confidence interval at 5% significance level $[1.559, 7.143]$) for the ω_{pol} model, and, slightly better, 2.857 cm ($[1.429, 4.286]$) for the hardwired controller.

3.5. Investigating the Generalization Ability of the Model ω_{pol}

After assessing and comparing the model with the hard-wired controller, we evaluated the generalization ability of the model ω_{pol}. To do so, we started the robot from a region where the model was not trained before (region B in Figure 6): the hard-wired controller still drives the robot safely through the gap when the robot was started from this area.

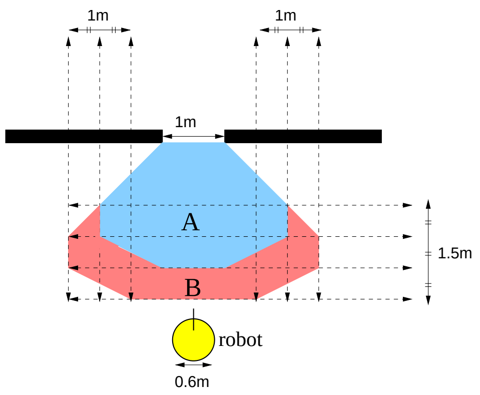


Figure 6: The area labelled A shows the region where the training data set was obtained to model the controllers. The area labelled B shows the region where the obtained model was tested to evaluate its generalization ability of the door traversal behaviour.

We started the robot from 16 different initial positions in area B. Figure 7 shows the trajectories of the robot under the control of the model ω_{pol}. The results show that the performance of the model deteriorates as the robot starts further from the known training region. Nevertheless, the robot traversed the gap successful in all but one experiment.

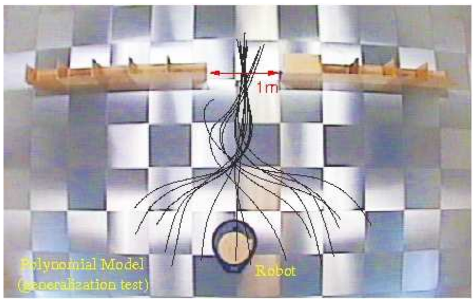


Figure 7: The generalization performance of the robot driven by the polynomial NARMAX model ω_{pol}. The robot was started from 16 different initial positions and the model managed to drive the robot through the gap 15 times out of 16 attempts.

4. Mathematical Analysis of the Trained Polynomial Model

One of the advantages of identifying unknown systems using polynomial NARMAX models is that the obtained expressions are transparent mathematical functions which can be directly related to the task. Polynomials give us the additional advantage of analysing models from a mathematical point of view in order to identify and understand the underlying rules governing the robot's interaction with the environment. In this section we will demonstrate how this advantage can be utilized to improve the performance of the obtained polynomial.

4.1. Simplifying the Model

Since the second order polynomial model ω_{pol} of Table 1 (with 49 terms) is a large polynomial, its mathematical analysis is difficult. We therefore decided to obtain simpler models. To do this, we did not use the raw sensor input as the input to our model, but selected a smaller set of input variables computed from them. At each time-stamp the laser scan contains in its values the information of the robot's relative position and orientation to the door, the minimal information the robot needs to perform the task. Therefore, the selected variables should also contain sufficient information for the robot. This does not only help reducing the dimensionality of the input and produce a simpler model, but, as will be seen, allows us to get human-comprehensible models in terms of the input variables.

4.1.1. Sensor data processing

We processed the raw laser readings to extract three input variables which contain enough information to determine the linear and angular speeds of the robot to drive the robot successfully through the door-like openings. These inputs are (Figure 8):

d: The distance from the robot to the midpoint of the door, where d is a positive value smaller than 4 m, since the robot's laser scanner range is limited to 4 m.

α: The angle between the heading of the robot and the direction to the center of the door. An obvious restriction on this parameter is $-90° < \alpha < 90°$, since outside this range the door cannot be in the angular field of view of the robot (the actual range in the experiments is even smaller; if α becomes too big, one of the sides of the opening is likely to be out of view.

β: The angle of the line from the robot's position to the door centre and the normal to the door. The value of this input variable falls in the range $(-90°, 90°)$.

The computation of the input vector (d, α, β) as input to the NARMAX model is relatively straightforward. First, the gaps generated by the two door jambs are detected and the distances d_1 and d_2 are computed, using a simple edge detection algorithm. Figure 9 illustrates a sample laser scan of the robot during door traversal.

These distances, jointly with the angular distance between the two corresponding beams, are then used to compute d, α and β.

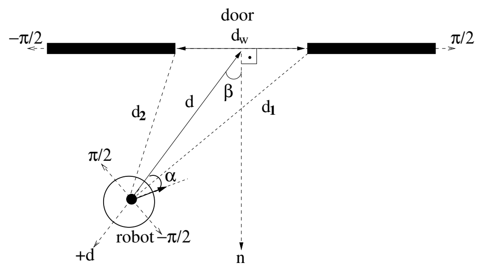


Figure 8: The three inputs (α, β and d) used as inputs to the simplified model.

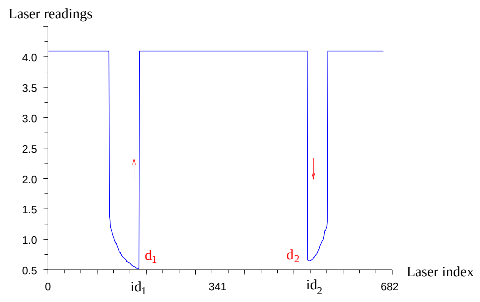


Figure 9: A sample laser scan of the robot during door traversal, where d_1 and d_2 are the distances between the robot and the left and right door jambs respectively, and id_1 and id_2 are the laser indexes of the two door jambs.

4.1.2. *Obtaining angular and linear velocity models*

Once d, α and β are computed, we used the NARMAX system identification technique to model the relationship between the input vector (d, α, β) and the angular and linear velocities of the robot. The resultant models were ARMAX models with no lags in the inputs and output, both models contained only 2 terms (Table 2). The obtained models proportional controllers, they are easily readable in terms of the variables d, α and β: The linear velocity equation computes a velocity proportional to the distance from the robot to the door.

The angular velocity depends on both angles α and β, and it can be seen that it makes the robot point to the center of the door. For example, if the robot is on the normal to the door (i.e. $\beta = 0$), but does not face the center, the angular velocity controller makes the robot turn in the correct direction. A similar analysis can be performed for positions outside this line (this type of analysis is much harder to perform for the previously obtained controller ω_{pos}).

$$\begin{aligned} v(n) &= \frac{d(n)}{40} + \frac{1}{10} \\ \omega(n) &= \frac{\beta(n)}{2} - \frac{\alpha(n)}{4} \end{aligned}$$

Table 2: The simplified polynomial models to achieve the door traversal behaviour.

4.1.3. *Model validation*

Having obtained the models given in Table 2, we tested them on the robot in the training environment, starting the robot from 23 different locations. In all episodes, the models were successful in driving the robot through the gap (Figure 10). We also evaluated the performance of the simplified models on imitating the hard-wired controller as in Section 3.4. The results revealed that the performance of the simplified models were far better than the previously obtained ω_{pol}, the new models had modelled the hard-wired controller perfectly (correlation 1.0 ($p < 5\%$) for both models).

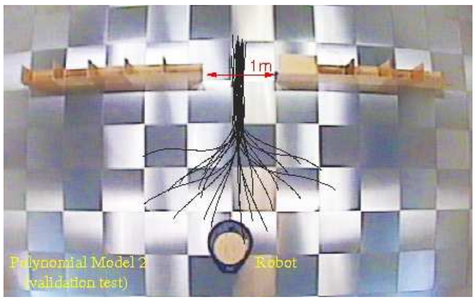


Figure 10: The trajectories of the robot under the control of simplified polynomial NARMAX models given in Table 2.

4.2. *Stability Analysis and Improving the NARMAX Controller*

The analysis of the robot trajectories shown in Figure 10 demonstrate that the robot traversed the door successfully, albeit not centrally. During the crossing, the median deviation of the robot from the center of the door was 3.284 cm (confidence interval $[1.635, 4.475]$, $p = 0.05$). In this section we will demonstrate how having transparent models allows us to optimize the models systematically so that the performance of the robot improves (passing through the gap more centrally).

To do so, we analysed the obtained models using Lyapunov stability analysis [21]. The Lyapunov stability theorems check if all the solutions of a dynamical system starting out near an equilibrium point x_c converge or stay close to it. In order to prove the stability of an equilibrium point, the theorem requires finding a function which must fulfil two conditions. Let $V(x) : \Re^n \rightarrow \Re$ be a Lyapunov candidate function which describing some energy of the dynamical system, the system of differential equations $\dot{x} = F(x)$ has a stability point at x_c if for all $x \in \Re$:

1. $V(x) \geq 0$, and $V(x) = 0 \iff x = x_c$ (positive definite function).
2. $\dfrac{dV(x)}{dt} = \nabla V(x) \cdot \dot{x} < 0$, (negative definite function).

where $\nabla V(x)$ is the gradient of the Lyapunov function $V(x)$, and $\dot{x} = F(x)$. For the door traversal behaviour, the state vector x includes the three elements d, α and β. This is the set of variables needed to completely characterise the system we are considering.

We defined the equilibrium point x_c as the center of the door, where $d = 0$, $\alpha = 0$ and $\beta = 0$ (see Figure 11) since we want the robot to be centralized in the doorway during crossing with a heading orthogonal to the door.

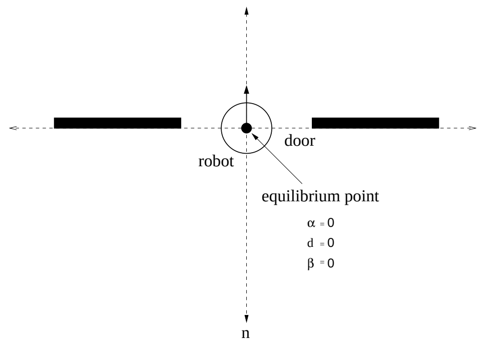


Figure 11: The equilibrium point or desired position and the orientation of the robot while passing through the door where $d = 0$, $\alpha = 0$ and $\beta = 0$.

The stability theorem implies that if the robot is near the equilibrium point, the controllers for v and ω will drive the robot to the origin and make the robot stay there forever. This means that when the robot reaches the equilibrium point, both angular and linear velocities of the robot should be zero to keep the robot on the equilibrium point. We find, however, when $d = 0$, $\alpha = 0$ and $\beta = 0$, the linear and angular speed models of Table 2 result in $v = 0.1$ and, correctly, $\omega = 0$.

Obviously, having a non-zero linear velocity on the equilibrium point violates the stability of the controllers. We therefore decided to modify the structure of the v controller, in such a way that $v = 0$ at the equilibrium point ($d = 0$, $\alpha = 0$ and $\beta = 0$). We removed the constant term from the model and changed the denominator of the second term to K so that new K parameter compensates the effect of removing the constant term. The new candidate v and ω controllers are given in Table 3.

$v(n)$	$=$	$\frac{d(n)}{K}$
$\omega(n)$	$=$	$\frac{\beta(n)}{2} - \frac{\alpha(n)}{4}$

Table 3: The candidate polynomial models to improve the door centralization.

4.2.1. *Estimating the value of K*

The value of the K was determined such that Lyapunov stability conditions hold for the controllers and therefore drive the robot safely to the equilibrium point. For the stability analysis, we derived the three differential equations in terms of the defined state ($\dot{d}$, $\dot{\alpha}$ and $\dot{\beta}$) defining the dynamical system of door traversal behaviour. These equations describe how the three inputs (d, α and β) change according to the actions of the robot. These equations are, in fact, a unicycle kinematic model expressed in polar coordinates [22]:

$$\dot{d} = -vcos(\alpha), \tag{2}$$

$$\dot{\beta} = \frac{-vsin(\alpha)}{d}, \tag{3}$$

$$\dot{\alpha} = \omega, \tag{4}$$

where v and ω are the robot velocities. The derivative of α with respect to time is just the angular velocity of the robot, while the change in d is proportional to the linear velocity component of the robot parallel to the $-d$ direction and change in β is proportional to the linear velocity component perpendicular to d direction and also inversely proportional with the distance d (see Figure 12).

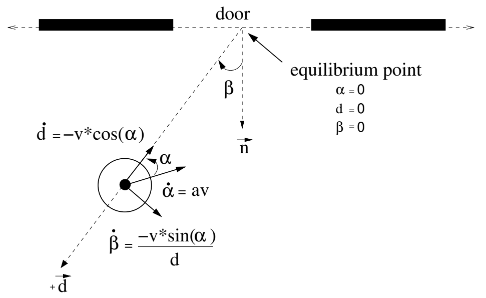


Figure 12: Three differential equations which describes how the perception of the robot (d, α, β) changes according to the robot's actions.

Substituting the candidate controller models given in Table 3 into Equations 4, 2 and 3, the resultant differential non-linear equations modeling the dynamic behaviour of the robot become:

$$\dot{\alpha} = \frac{\beta}{2} - \frac{\alpha}{4}, \tag{5}$$

$$\dot{d} = \frac{-d\cos\alpha}{K}, \tag{6}$$

$$\dot{\beta} = \frac{-\sin\alpha}{K}. \tag{7}$$

Now we can use the following Lyapunov candidate function $V(d, \alpha, \beta)$:

$$V = \frac{1}{2}(\dot{\alpha}^2 + \dot{d}^2 + \dot{\beta}^2). \tag{8}$$

This function is one of the most common choices among Lyapunov candidate functions [21] which satisfies the requirement of the first Lyapunov stability condition where $V \geq 0$ for all the values of d, α and β.

The derivative of V with respect to time can now be written in terms of partial derivatives of the differential equations of d, α and β.

$$\frac{dV}{dt} = \frac{1}{2}\left(\frac{\partial V}{\partial \alpha}\dot{\alpha} + \frac{\partial V}{\partial d}\dot{d} + \frac{\partial V}{\partial \beta}\dot{\beta}\right), \tag{9}$$

where

$$\frac{\partial V}{\partial \alpha} = \frac{1}{2}\left(\frac{\alpha - 2\beta}{8} + (d^2-1)\frac{\sin 2\alpha}{K^2}\right), \tag{10}$$

$$\frac{\partial V}{\partial d} = \frac{d\cos^2\alpha}{K^2}, \tag{11}$$

$$\frac{\partial V}{\partial \beta} = \frac{1}{8}(-\alpha + 2\beta). \tag{12}$$

Combining Equations 5-7 with Equations 10-12 we get the individual terms of $\dot{V}(d,\alpha,\beta)$

$$\frac{\partial V}{\partial \alpha}\dot{\alpha} = -\frac{1}{64}(\alpha-2\beta)^2 + \frac{1}{8}\frac{(d^2-1)(\alpha-2\beta)\sin 2\alpha}{K^2}, \tag{13}$$

$$\frac{\partial V}{\partial d}\dot{d} = -\frac{d^2\cos^3\alpha}{K^3}, \tag{14}$$

$$\frac{\partial V}{\partial \beta}\dot{\beta} = \frac{(\alpha-2\beta)\sin\alpha}{8K}. \tag{15}$$

Having identified $\dot{V} = \frac{dV}{dt}$ as a function of d, α and β, we evaluated the equation on the computer to see the response of the system for the input vector (d, α, β) varying in the the range of $0 \le d \le 4$, $-90^\circ \le \alpha \le 90^\circ$, and $-90^\circ \le \beta \le 90^\circ$, corresponding to the possible positions of the robot. Different values were tested for the parameter K within the range of $0 \le K \le 20$. The results revealed that to keep the system stable ($\dot{V} < 0$) (Figure 13), the value of K must be bigger than or equal to 10. As the value of K gets smaller the stability of the system cannot be guaranteed ($\dot{V} > 0$).

From a theoretical point of view, having a lower bound for K was expected, since K is the parameter affecting only the linear speed of the robot. It is obvious that the v and ω models are linked to each other for the generation of the proper behaviour of the robot. When K gets smaller, v increases and the ω value does not increase enough to make the robot turn to desired direction, i.e. the curvature of the trajectories ($\kappa = \omega/v$) is smaller and the robot cannot turn fast enough to head towards the door.

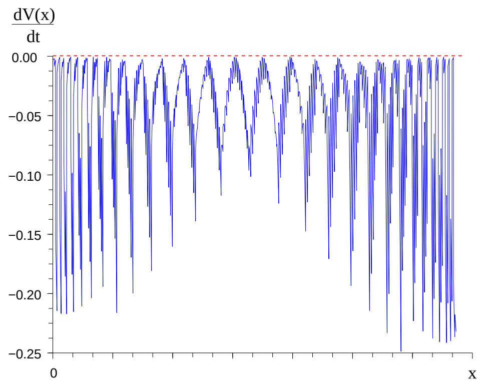


Figure 13: The plot of $\frac{dV(x)}{dt}$ function where $x = [d,\alpha,\beta]$ and $K = 10$.

On the other hand, when K increases, v gets smaller and the overall response of the system slows down, i.e. the robot spends more time to achieve the desired task. We therefore chose K as small as possible ($K = 10$) within the stability region. The final, optimized controllers are given in Table 4.

$$\begin{aligned} v(n) &= \frac{d(n)}{10}, \\ \omega(n) &= \frac{\beta(n)}{2} - \frac{\alpha(n)}{4}. \end{aligned}$$

Table 4: The optimized polynomial models to achieve better door centralization.

4.2.2. *Validation of the optimized model*

Having obtained the new controller models, we tested them on the real robot by starting from 32 different locations in the training environment (note that the new controllers will drive the robot to the equilibrium point, but will not make the robot pass through the door since v and ω become 0 at the equilibrium point). Therefore, just for demonstration purposes, we modified the program in such a way that when the robot reaches the equilibrium point, we set the linear velocity to 0.1 so that the robot would carry on and actually pass through the door.

The results (see Figure 14) demonstrated that in all experiments the models drove the robot successfully through the gap and, more importantly, the robot traversed the door more centrally (median deviation of the robot from the center of the door is 1.254 cm, confidence interval = $[0.567, 1.345]$, $p = 0.05$), significantly better than the previous models (3.284 cm, confidence interval = $[1.635, 4.475]$) at the 5% significance level (U-test).

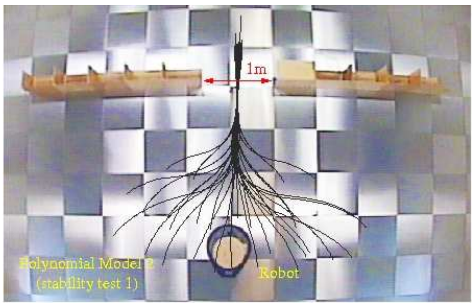


Figure 14: The trajectories of the robot under the control of the optimized polynomial NARMAX models given in Table 4.

4.2.3. *Measuring the generalization performance of the model*

Having demonstrated that the obtained controllers are stable, driving the robot to the center point of the door, we also postulate that the models would drive the robot successfully through *any* opening, as long as the width of the gap is bigger than the width of the robot .

We tested this hypothesis by running the robot in an environment composed of two consecutive door-like openings with different widths, 70 cm and 80 cm. These are very small openings, leaving just 5 cm manoeuvering space either side for a robot of 60 cm diameter! Figure 15 shows the trajectories of the robot, they confirm that the model indeed achieves this difficult, high-precision task. The median deviations of the robot from the center point for the first and the second gap are (1.389 cm, confidence limits= $[0.455, 1.548]$) and 1.456 cm, confidence limits= $[0.300, 1.889]$, respectively. This controller performs better than the non-optimized one, even for smaller door openings.

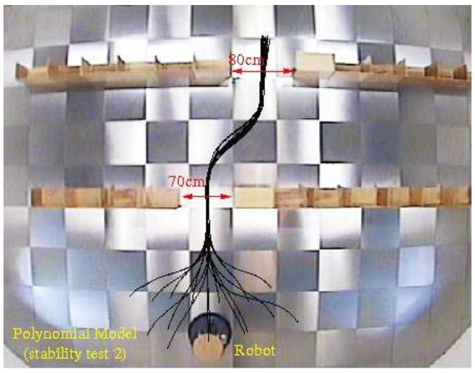


Figure 15: The trajectories of the robot under the control of the optimized polynomial NARMAX models of Table 4, passing through two consecutive openings with 70 cm and 80 cm. The base width of the robot is 60 cm.

4.3. Obtaining Empirical Perception Models

In the previous section we defined three differential equations ($\dot{d}$, $\dot{\alpha}$ and $\dot{\beta}$), modelling the dynamical system of door traversal behaviour. These equations describe how the three inputs (d, α and β) change, as a result of the actions of the robot. We assumed these functions to be known beforehand.

However, in real robot applications these equations are rarely known *a priori*, and have to be *estimated* by looking at the training data.

To obtain such models from real data, rather than from theoretical models, we again used the the NARMAX system identification method to obtain perception models describing how the controller input vector (d_k, α_k, β_k) changes in relation to the actions of the robot, performing door traversal, at each time stamp k. We also assume that the current state of the environment perceived by the robot (d_k, α_k, β_k) can be computed from the previous environment state vector $(d_{k-1}, \alpha_{k-1}, \beta_{k-1})$ and the previous velocity commands of the robot (v_{k-1}, ω_{k-1}).

We trained three polynomial NARMAX models to estimate the controller inputs (d_k, α_k, β_k) based on the previous input values $(d_{k-1}, \alpha_{k-1}, \beta_{k-1})$ and the previous action commands (v_{k-1}, ω_{k-1}). The obtained perception model can be stated as a set of ARMAX models with no lag in the inputs and the outputs (Table 5). This means that for our episodic task considering only a linear polynomial is enough to make a good prediction.

$$\begin{aligned} d_k &= 0.21 + 1.05 d_{k-1} - 2.25 v_{k-1} \\ \alpha_k &= 1.03 \alpha_{k-1} + 0.3 \omega_{k-1} \\ \beta_k &= 0.96 \beta_{k-1} - 0.1 \alpha_{k-1} \end{aligned}$$

Table 5: The perception models estimating position and orientation of the door with respect to robot position, as a consequence of the actions of the robot.

The performance of the obtained perception models were tested using a validation data set obtained during the training session. Figure 16 presents the real and predicted values of d, α and β along the whole set of testing trajectories chained one after the other.

While computing d, α and β, the estimation models were allowed to get real sensor input only at the starting position of each trajectory, then predicting the rest using the real motor commands and the previously estimated input values. Jumps at the beginning of the sequences are the effect of chaining the data of different trajectories.

To evaluate the performance of the models we computed the Spearman rank correlation coefficient between the real and predicted values for d, β and α, given in Table 6, indicating quantitatively that the estimates of the prediction models closely match the real values of d, β and α.

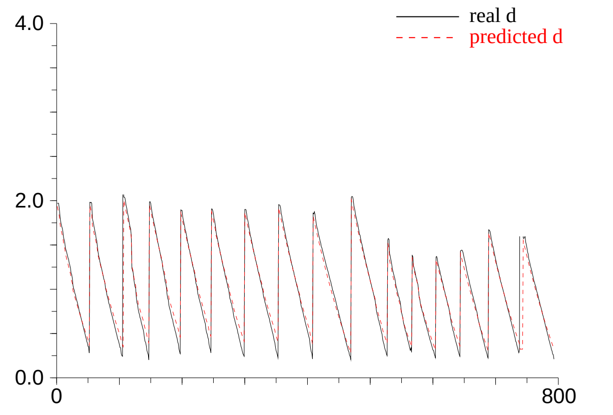
	Mean Absolute Error	*Spearman Rank Correlation*
d	0.081 ± 0.010 (m)	0.951, $(p < \%5)$
β	0.117 ± 0.003 (rad)	0.757, $(p < \%5)$
α	0.095 ± 0.003 (rad)	0.963, $(p < \%5)$

Table 6: We evaluated the performance of the perception models by computing the mean absolute error and the Spearman rank correlation coefficients between the real and predicted values for d, β and $alpha$ respectively.

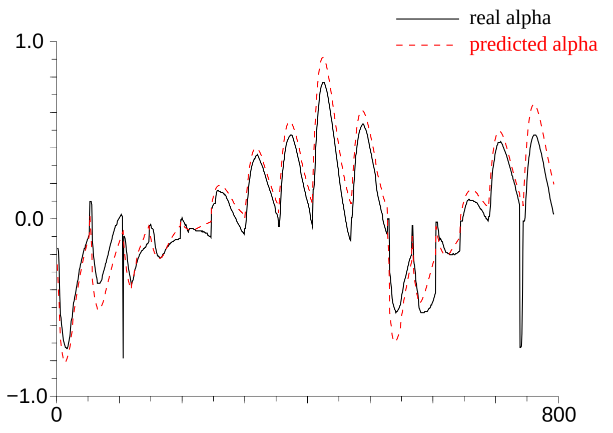
4.3.1. Validation of the perception models

Having obtained both perception and controller models, we then validated the performance of the models by testing them on the real robot in a real door traversal application. We used the controller model to drive the robot, but rather than using real sensory inputs (d, α, β), the estimated values computed by the perception models were fed to the controller. The robot was allowed to get real sensor input only at the starting position, and then the whole set of motion commands was generated by the controller using the estimate of d, α and β at each step.

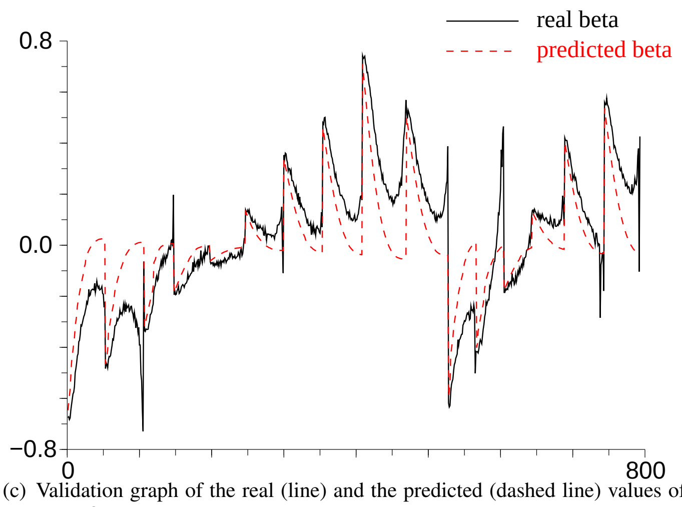
Figure 17 shows the resulting trajectories of the robot. The results show the accuracy of the models, since the robot is able to traverse the door successfully even with a single snapshot of the environment at the starting position. Obviously the performance of the robot is not as good as when the robot uses real sensor signals to compute the speed outputs where the median



(a) Validation graph of the real (line) and the predicted (dashed line) values of the distance d to the centre point of the door



(b) Validation graph of the real (line) and the predicted (dashed line) values of the robot heading α relative to the centre point of the door



(c) Validation graph of the real (line) and the predicted (dashed line) values of the angle β between robot position and centre point of the door

Figure 16: Real values and predictions of the robot-environment variables over the test trajectories.

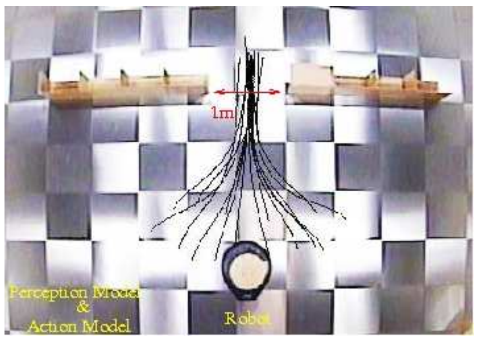


Figure 17: The trajectories of the robot under the control of the controller and perception models of Table 5. The robot was able to traverse through the door almost blindly without using any real sensor inputs. The robot was allowed to get real sensor input only at the starting position, and then the whole set of motion commands was generated by the controller based on the estimate values of d, α and β computed by the prediction models.

deviation of the robot from the mid-point of the door was 5.456 cm, $[0.300, 1.889]$. However, this approach can be very useful under unexpected circumstances; for instance if at some step of the door traversal behaviour the door detection algorithm fails to detect the door accurately, or if the laser sensor of the robot breaks down for some reason. It can also be used for abnormality detection since the robot has an expectation about how its perception changes according to its actions.

5. Conclusions and Further Work

5.1. *Summary*

This paper discusses the application of the NARMAX system identification method to generate sensor-motor couplings in mobile robotics applications. The main advantage of modelling sensor-motor relationship of the robot to achieve a desired task using polynomial NARMAX models is that polynomial models are transparent mathematical functions which can be directly related to the task under investigation. This allows an analysis of how each input entering the model affects the overall behaviour of the robot but also an analysis to test whether the obtained models are stable in the sense that they guarantee to drive the robot to the desired end point.

In the example presented (Section 4.2), we demonstrated that Lyapunov stability analysis can be used for analysing polynomial NARMAX models. We computed the optimum value for the parameter K in order to optimize the door traversal controllers such that they drive the robot to the center point of the door safely. Having demonstrated that the robot traverses the gap well centered, we then postulated that the models would drive the robot through *any* opening, as long as the width of the gap is bigger than the width of the robot, and the door is detected correctly. Testing our hypothesis by letting the optimized models drive the robot in a test environment, the results confirmed that the optimized models are able to traverse through the openings safely.

Furthermore we demonstrated that NARMAX system identification method can be used to obtain perception models that identify how the perception of the robot changes according to its actions, in the context of the desired behaviour. In particular we obtained models d_k, α_k, and β_k that estimate how the position and the orientation of the door relative to the robot alters according to robot's actions.

As demonstrated in Section 4.3, we validated the performance of the models by testing them on the robot in a real door-traversal application. Using the combination of controller and perception models enabled the robot to cross door-like opening almost blindly, using sensory perception only once right at the beginning, after that computing motion commands *en route*, based on the model-predicted values of d, β and α.

5.2. *Future Work*

Having controller and perception models identified in a transparent form using mathematical models enables us to have a complete mathematical description of the robot-environment interaction for the desired behaviour. We believe that this will give us the opportunity for further analysis of robot-environment interaction from a mathematical point of view so that we can have a better understanding of how the robot is interacting with the environment to accomplish the desired behaviour. Therefore, we are currently investigating formal methods of analysing the obtained models to address scientific questions such as identifying if the generated controllers are stable or not, or to predict the behaviour of the robot in untested environments.

Overall, the work already carried out and the proposed future work are part of ongoing research to develop a theory of robot-environment interaction.

Acknowledgments

Iñaki Rañó was supported by the Spanish Ministry of Science and Education under grant JC2007-00293.

References

[1] A. Billard, G. Hayes, Learning to communicate through imitation in autonomous robots, in: In 7th International Conference on Artificial Neural Networks, Springer-Verlag, 1997, pp. 763–768.

[2] S. Lauria, G. Bugmann, T. Kyriacou, J. Bos, E. Klein, Training personal robots using natural language instruction, IEEE Intelligent System 16 (2001) 38–45.

[3] A. Dearden, Y. Demiris, Learning forward models for robots, in: Proceedings of the International Joint Conference on Artificial Intelligence, Edinburgh, UK, 2005, pp. 1440–1445.

[4] D. Nguyen, B. Widrow, The truck-backer upper: An example of self-learning in neural networks, IEEE Control Systems Magazine 10(2) (1990) 18–23.

[5] D. Pomerleau, Knowledge-based training of artificial neural networks for autonomous robot driving, in: J. Connell, S. Mahadevan (Eds.), Robot Learning, Kluwer Academic Publisher, 1993, pp. 19–44.

[6] N. E. Sharkey, Learning from innate behaviors: a quantitative evaluation of neural network controllers, Machine Learning (1998) 115–139.

[7] L. Zhao, C. E. Thorpe, Stereo- and neural network-based pedestrian detection, IEEE Transactions on Intelligent Transportation Systems 1(3) (2000) 148–154.

[8] U. Nehmzow, Applications of robot training: Clearing, cleaning, surveillance, in: Proc. of International Workshop on Advanced Robotics and Intelligent Machines, Salford, UK, 1995.

[9] U. Nehmzow, Mobile Robotics: A practical introduction, 2nd Edition, Springer Verlag, 2003.

[10] T. Kyriacou, U. Nehmzow, R. Iglesias, S. A. Billings, Task characterization and cross-platform programming through system identification., International Journal of Advanced Robotic Systems 2 (2005) 317–324.

[11] R. Iglesias, T. Kyriacou, U. Nehmzow, S. A. Billings, Robot programming through a combination of manual training and system identification, in: Proc. of ECMR 05 - European Conference on Mobile Robots 2005., Springer Verlag, 2005.

[12] O. Akanyeti, U. Nehmzow, S. A. Billings, Robot training using system identification, Robotics and Autonomous Systems 56 (2008) 1027–1041.

[13] P. Eykhoff, Trends and Progress in System Identification, Pergamon Press, 1981.

[14] S. Chen, S. A. Billings, Representations of non-linear systems: The narmax model, International Journal of Control 49 (1989) 1013–1032.

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

[16] K. Weierstrass, Über die analytische Darstellbarkeit sogenannter willkürlicher Funktionen einer reellen Veränderlichen, in: Sitzungsberichte der Königlich Preußischen Akademie der Wissenschaften, Berlin, 1885.

[17] S. Chen, S. A. Billings, C. F. N. Cowan, P. M. Grant, Practical identification of narmax models using radial basis functions, International Journal of Control 56(6) (1990) 1327–1350.

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

[19] S. A. Billings, S. Chen, The determination of multivariable nonlinear models for dynamical systems, in: C. Leonides (Ed.), Neural Network Systems, Techniques and Applications, Academic press, 1998, pp. 231–278.

[20] O. Akanyeti, T. Kyriacou, U. Nehmzow, R. Iglesias, S. A. Billings, Visual task identification and characterization using polynomial models, Int. J. Robotics and Autonomous Systems 55 (2007) 711–719.

[21] R. C. Dorf, Modern Control Systems, 9th Edition, Prentice Hall, 2000.

[22] M. Alcaradi, G. Casalino, A. Bicchi, A. Balestino, Closed loop steering of uniclycle-like vehicles via Lyapunov techniques, IEEE Robotics and Automation Magazine 2 (1) (1995) 27–35.