This is a repository copy of *Accurate Robot Simulation Through System Identification*.

White Rose Research Online URL for this paper:
http://eprints.whiterose.ac.uk/85406/

Monograph:

Kyriacou, T., Nehmzow, U., Ingelsiais, R. et al. (1 more author) (2008) Accurate Robot Simulation Through System Identification. Research Report. ACSE Research Report 968 . Department of Automatic Control and Systems Engineering

Accurate Robot Simulation through System Identification

T Kyriacou~, U Nehmzow[#], R Inglesias [*], S A Billings

~School of Computing and Mathematics, University of Keele
#Dept Computer Science, University of Essex
*Electronics and Computer Science, University of Santiago de Compostela,
Spain
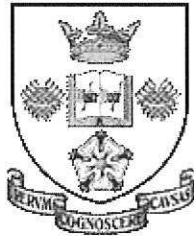
Department of Automatic Control and Systems Engineering
The University of Sheffield, Sheffield, S1 3JD, UK

Research Report No. 968

January 2008

Accurate Robot Simulation through System Identification

Theocharis Kyriacou [a] Ulrich Nehmzow [b],* Roberto Iglesias [c]
S. A. Billings [d]

[a]*School of Computing and Mathematics, Keele University, Keele ST5 5BG, UK*

[b]*Department of Computing and Electronic Systems, University of Essex, Colchester CO4 3SQ, UK*

[c]*Department of Electronics and Computer Science, University of Santiago de Compostela, Spain.*

[d]*Department of Automatic Control and Systems Engineering, University of Sheffield, UK*

Abstract

Robot simulators are useful tools for developing robot behaviours. They provide a fast and efficient means to test robot control code at the convenience of the office desk. In all but the simplest cases though, due to the complexities of the physical systems modelled in the simulator, there are considerable differences between the behaviour of the robot in the simulator and that in the real world environment.

In this paper we present a novel method to create a robot simulator using real sensor data. Logged sensor data is used to construct a mathematically explicit model (in the form of a NARMAX polynomial) of the robot's environment. The advantage of such a transparent model — in contrast to opaque modelling methods such as artificial neural networks — is that it can be analysed to characterise the modelled system, using established mathematical methods

In this paper we compare the behaviour of the robot running a particular task in both the simulator and the real-world using qualitative and quantitative measures including statistical methods to investigate the faithfulness of the simulator.

* Corresponding author, udfn@essex.ac.uk, Fax +44-1206-872788

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 (see figure 1). This results in a highly complex, usually non-linear, system. 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 environment are concerned. Furthermore, even though the robot's programming language provides a very precise means of dictating the task in mind, a reasonably complex program can never be tested fully and is likely to cause some unexpected behaviour at some point during its execution.
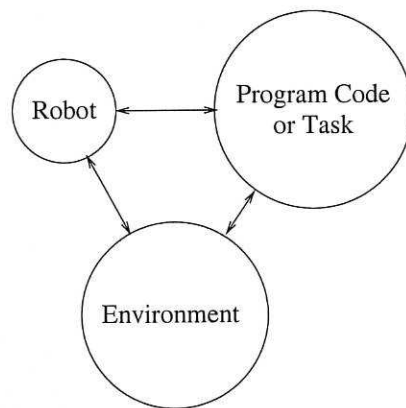

Fig. 1. The behaviour of a robot emerges from the interaction between robot, task and environment.

The lack of a formal design methodology that is 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. This problem is addressed in previous work under the Robot-MODIC project (see [1–3]).

The lack of a theoretical foundation for mobile robotics means that development tools have to be based on assumptions (*e.g.* idealised, simplified models of sensors, simplified environment models *etc*) that commonly result in significant discrepancies between predicted and actually observed behaviour of the physical mobile robot.

The objective of the work described here is to establish a methodology for creating faithful robot simulators with which we can: (i) make accurate predictions about robot behaviour and (ii) methodically investigate how robot behaviour is influenced by the environment so that we can ultimately develop a theoretical foundation of robot-environment interaction.
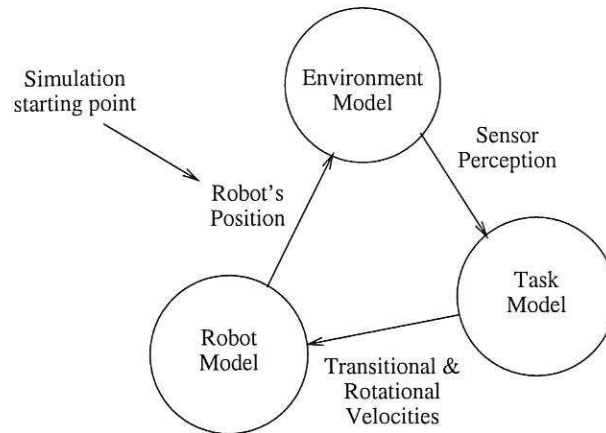
Fig. 2. A diagrammatic representation of the function of the robot simulator. Based on the robot's position and orientation the environment model produces the estimated sensor perception. This is then passed to the task model which computes the transitional and rotational velocities of the robot. Finally the velocities are used by the robot model in order to compute the next position and orientation of the robot in the environment.

Robot simulators are convenient robotics research tools for developing and testing new robot controllers. One of their main advantages is that they save significant time and effort which is otherwise spent in working with real world experimental setups [4]. This is because: (i) time in simulation usually runs faster than real time and therefore the results of experiments are ready quicker and (ii) because less time is spent to setup an environment in simulation than in the real world.

In addition, simulations provide the only absolutely consistent method for experimental repetition. This is particularly helpful, for example, when we need to compare two robot controllers under identical conditions (robot and environment). Finally, with some learning paradigms such as reinforcement learning, it is necessary to allow the robot to err in order to learn. Simulating the robot, in such cases, would save any time required to reset the experimental setup after such errors and will also minimise any damage risk to the robot.

The prediction accuracy of a robot simulator directly depends on the fidelity of the three models that comprise it (see figure 2). By far the most complex of the three, and the one we are most concerned with in this paper, is the environment model. The environment model provides the sensor perception of the robot, given the robot's position and orientation. This model depends on the structure of the environment as well as the properties of the material it is made of.

3

The advantages, then, of mobile robot simulation are obvious, and computer models of robot-environment interaction are clearly desirable as development tools. Consequently, large parts of the robotics research community have adopted simulation as a viable and fundamental research tool, and a number of simulators, such as Player-Stage [5] and Webots [6], as well as those supplied with specific robot platforms[7–9], are widely used.

These general purpose robot simulators, as well as models of specific robots, such as [10,11], have in common that they aim to be *generally* applicable, i.e. generic, and that they are therefore based on general assumptions about kinematics and dynamics of the robot and environment properties. The Webots simulation, for instance, defines mass distribution matrices for robot components, friction coefficients, bounciness etc in order to achieve a certain degree of fidelity, a RoboCup simulation is similar[11]. Player-Stage aims for "good enough fidelity" [5], where "good enough" is not defined (see discussion in section 3.2.1). Some simulations use generic kinematics and dynamics models as well as generic noise models, usually assuming all noise affecting robot operation is Gaussian [10].

Generic robot simulators, based on general assumptions about the robot and its environment, have attracted criticism in the past [12,13] that has, so far, only been addressed to a limited degree by the research community. As we argued in [14,15], and continue to believe, faithful simulation of robot-environment interaction has to be data-driven, and can therefore only model *specific* experimental scenarios. In all but the simplest robotics experiments one finds that certain parts of the environment give rise to extreme perceptions, for example through specular reflections off smooth surfaces — a phenomenon that a generic simulator can not model.

This paper therefore addresses the issue of *faithful* computer simulation of robot-environment interaction, not generic simulation, and for this reason uses a modelling technique that is data-driven. Related work that is therefore much closer to our proposal here is the simulator built by Lund and Miglino [16], which stores logged data in a lookup table and uses interpolation for the prediction of sensory perception at unvisited locations, or the neural network-based simulator presented in [14,15], which trains a multilayer Perceptron to model the relationship between robot location and sensor perception. Both of these approaches result in highly accurate models that are able to predict sensory perceptions, even in areas having perceptual discontinuities.

Besides faithful simulation, there is, however, another motivation behind the work presented in this paper: we would like to *understand*, at least partially,

the relationship between location and perception. An opaque neural network model is unsuitable for this, and an interpolation method is, strictly speaking, not a modelling (i.e. generalising) approach at all.

In one sentence, we would like to obtain a *faithful* computer model of robot-environment interaction that retains, in *simplified* form, the essence of that interaction, and that is *transparent* and therefore amenable to analysis. We achieve these three objectives using system identification[17].

Compared with other modelling approaches, such as artificial neural networks or lookup tables, system identification offers noticeable benefits:

- The obtained models are extremely compact, consisting of one polynomial of typically a few dozen terms.
- The models are transparent, and therefore analysable by standard techniques such as sensitivity analysis. Examples of such analyses are given in [18] and [19].

1.2 *The NARMAX modelling procedure*

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

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

$$y(n-1)^l, y(n-2)^l, \cdots, y(n-N_y)^l)$$

were $y(n)$ and $\boldsymbol{u}(n)$ 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, this is typically taken to be a polynomial or wavelet multi-resolution expansion of the arguments. The degree l of the polynomial is the highest sum of powers in any of its terms.

Noise is always present in physical systems and can be accommodated as part of the NARMAX model. In the present study we have initially assumed that the effects of the noise are small and can be neglected. In later studies the effects of any noise will be accommodated by fitting noise models as part of the identification procedure to ensure that unbiased models are obtained.

The NARMAX methodology breaks the modelling problem into the following steps:

(1) Structure detection,
(2) Parameter estimation,
(3) Model validation,
(4) Prediction and
(5) 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 [20].

A detailed procedure of how structure detection, parameter estimation and model validation is done is presented in [21]. 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). We call the first the *estimation data set* and it is used to determine the model structure and parameters. The remaining data set is called the *validation data set* and it is used to validate the model.

The initial 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. Any signal that influences the output should be assigned as an input. The number of initial 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. Because the models are constructed by adding one term at a time (the most significant term) all the redundant terms are discarded. The final structure of

6

the estimated NARMAX model will indicate any insignificant or redundant inputs.

Before any removal of model terms an equivalent *auxiliary model* is computed from the original NARMAX model. The model terms of the auxiliary model are orthogonal. This allows the computation of their associated parameters to be done in a sequential manner which is computationally more efficient.

The calculation of the auxiliary model parameters and the refinement of the model's structure is an iterative process which allows the model to be built up one term at a time. This is followed by model validation.

After the model validation step, 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 and also to avoid over-fitting the model to the training data.

To determine the contribution of a model term to the output the Error Reduction Ratio (ERR) [21] 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 (usually around 0.05%) are removed from the model polynomial in the last step of each iteration during the refinement process.

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. Finally, the NARMAX model parameters are computed from the auxiliary model.

In the next section we describe how we apply the NARMAX methodology to obtain a model of the robot's environment. The inputs of the system (i.e. the environment) are the robot's position and orientation and output is the robot's sensor perception.

2 Experimental methods

2.1 Experimental setup

The robot used in our experiments is the *Magellan Pro* autonomous mobile robot *Radix* (figure 3). The robot is equipped with 16 sonar, 16 infra-red and 16 tactile sensors distributed uniformly around its circumference. A *SICK* laser range finder is also present which scans the front semi-circle of the robot ([0°, 180°]) with a radial resolution of 1° and a distance resolution of less than 1 centimetre. The robot also incorporates a video camera. In the work presented in this paper primarily the laser sensor is used.
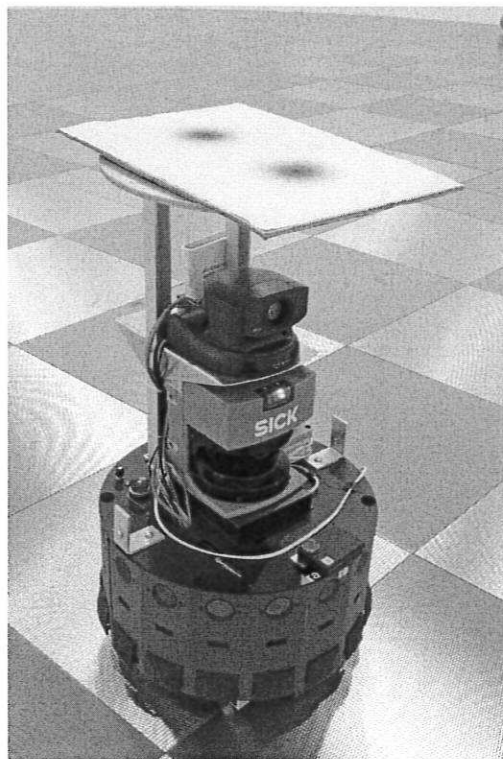

Fig. 3. *Radix*, the *Magellan Pro* mobile robot used in the experiments described in this paper. The visual target visible on the top of the robot was used for vision-based trajectory tracking.

During experiments with *Radix*, the input from all its sensors (apart from the video camera), the robot's position, orientation, transitional and rotational velocities were recorded every 250 ms. Position and orientation of the robot were obtained by placing two point-targets on top of the robot (see figure 3) and using an overhead video camera to track them continuously. The position error of this tracking method was approximately 1 cm. After a logging session the sensor data from the robot and the position/orientation data from the

8

tracking system are aligned using time as the common reference. Finally, due to the high sampling rate used, it is often necessary to subsample the collected data before it is used for modelling and analysis.

Experimental setups of the robot's static environment are built in a dedicated robot arena usually using carton boxes. An example of one such experimental setup is shown in figure 4.
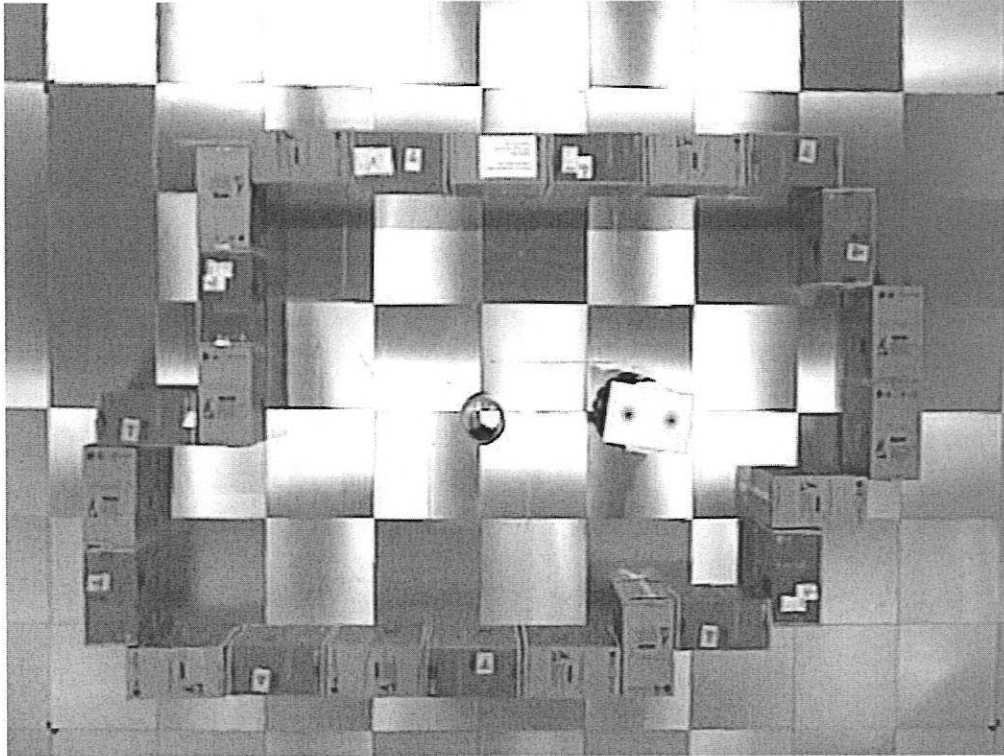

Fig. 4. Bird's eye view of an experimental setup. The longest width and height of the environment is 3.9 and 2.4 metres respectively. The image is taken with the overhead camera used for tracking the position of the robot.

To build the simulator, *Radix* is first used to collect data from the environment that needs to be modelled. This is done using an exploratory task in order to scan the environment using the robot's sensors as thoroughly as possible. Figure 5 shows the trajectory of the robot during one such exploration session.

2.2 Environment modelling

2.2.1 Method 1: Continuous Model building

After collecting sensor data during the exploration phase, a model is computed which can predict the sensor perception given the robot's position and
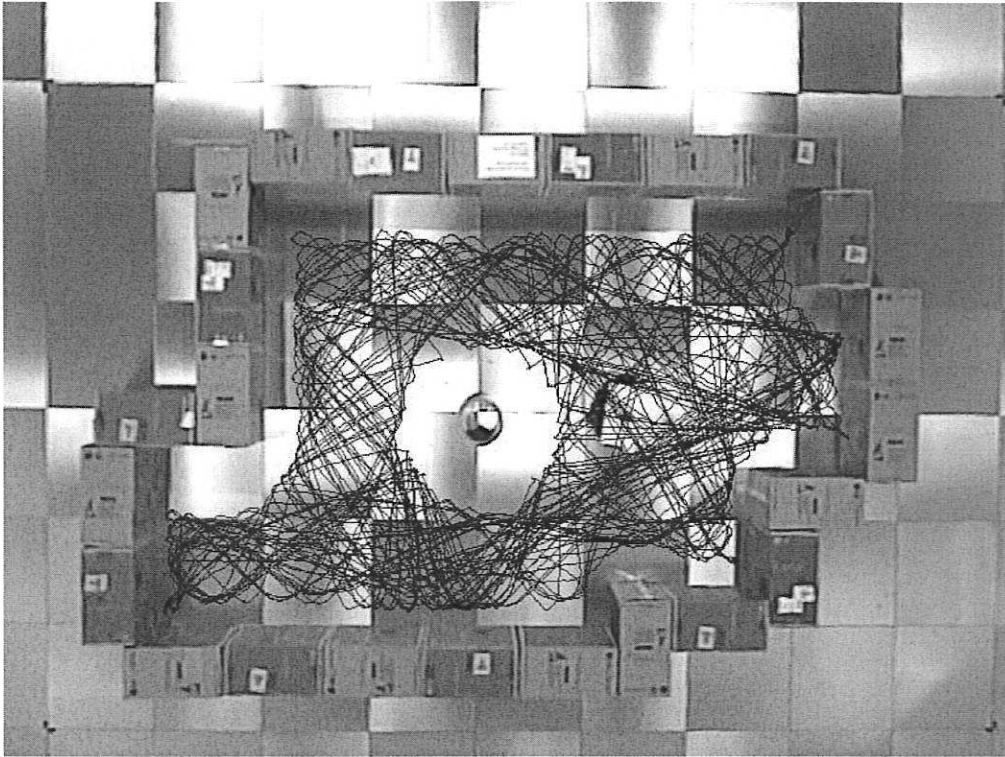
Fig. 5. The trajectory of the robot during an exploration of the environment. The task that was being executed was an obstacle avoidance task using the robot's laser sensor. To produce the shown trajectory, the robot moved in the environment for approximately 2 hours.

orientation in the environment. This is the basis of our robot simulator.

The environment model $\mathcal{M}^{CSM}$ is comprised of a set of functions, each of which is itself a model of the environment but only as perceived by the robot when it is at a particular location:

$$\mathcal{M}^{CSM} = \{S^{mp}\} \tag{1}$$
$$S^{mp}_{(x_m,y_n)} = f(\varphi) \tag{2}$$

$$m = 1, ..., w$$
$$n = 1, ..., h$$
$$\varphi \in [-\pi, +\pi]$$

where $S^{mp}_{(x_m,y_n)}$ is the model-predicted sensor perception when the robot is at location (x_m, y_n). This is a function of the angle φ in which the particular sensor is facing. We shall, from now on, call $S^{mp}_{(x_m,y_n)}$ the *model-predicted sensor signature* of location (x_m, y_n). The set of locations $\{(x, y)\}$ are chosen to be on an equally spaced grid (x^g, y^g) (comprised of $w \times h$ locations) which covers
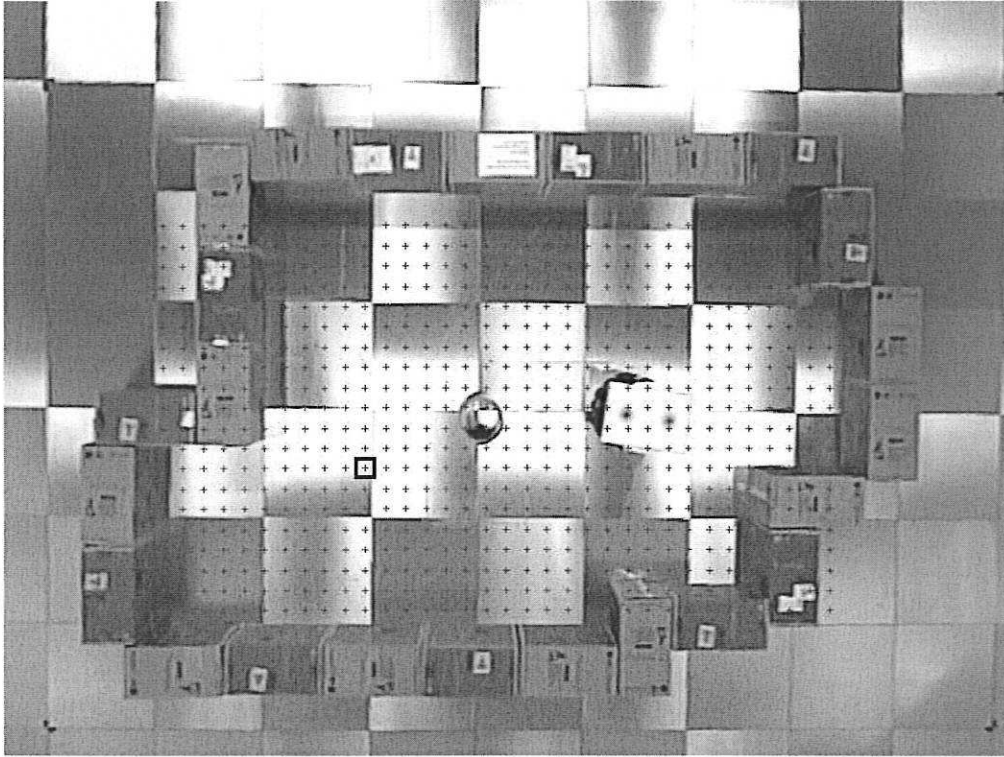
10

Fig. 6. A 10 cm grid superimposed on the image of the environment. Location (1.78, 2.31) is shown in black, the laser perception of the robot at this location is given in figure 7.

the surface area of the environment to be modelled. As an example, the grid locations of a 10 cm spaced grid for the environment shown in figure 4 are shown in figure 6.

Each sensor signature, perceived at a particular grid location, is then modelled using a NARMAX polynomial, using real sensor data obtained during the exploration of the environment. This approximated sensor signature $\{S^a_{(x_m,y_n)}\}$ for location (x^g_m, y^g_n), represented by a non-linear polynomial, contains two elements: the sensor angle φ and the approximated range value of the sensor v_φ corresponding to this angle:

$$S^a_{(x_m,y_n)} = \{(\varphi, v_\varphi)\} \tag{3}$$

$$\varphi \in [-\pi, +\pi]$$

The sensor range value is approximated because it is taken to be the same as that of the nearest value available in the exploration data (*i.e.* the same as that of the nearest robot position and sensor angle, in the exploration data, to the grid position and sensor angle for which the signature is approximated).
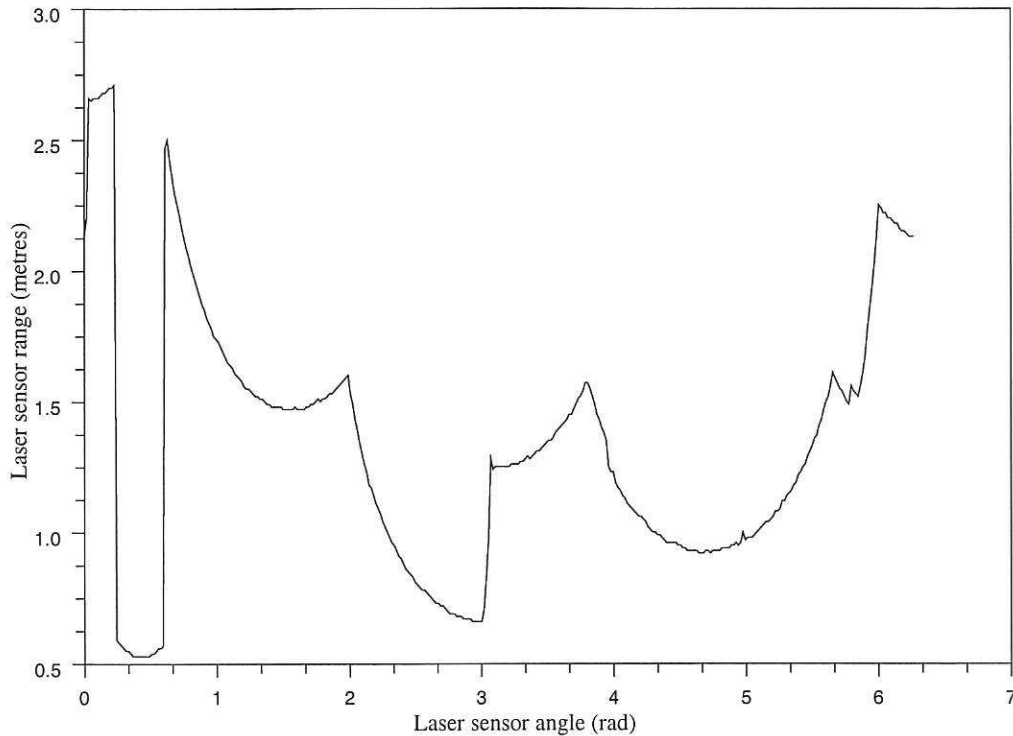
11

Fig. 7. The laser sensor signature at grid location (1.78, 2.31) metres. The sensor values are shown starting from 0 radians (east direction with respect to figure 6) clockwise to 2π radians.
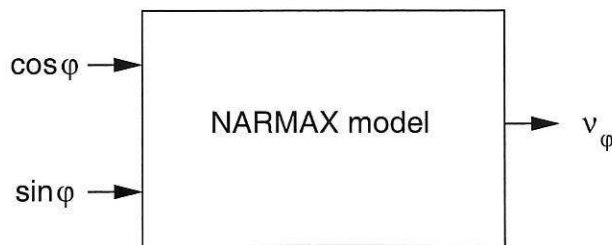

Fig. 8. Inputs and output of the NARMAX model for each sensor signature in the grid of locations chosen to model the environment, were φ is the sensor angle and v_φ the approximated range value perceived by the sensor at this angle.

The resolution of the angle φ is $\pi/180$ rad thus making S^a a signature of 360 samples. Figure 7 shows the approximated signature of the laser sensor at grid location (1.78, 2.31) metres (shown as a black square in figure 6).

After obtaining each approximated grid signature (S^a), the NARMAX modelling methodology is used to estimate the continuous function (S^{mp}) that represents S^a. To avoid the discontinuity at $\varphi = \pm\pi$, we use $sin(\varphi)$ and $cos(\varphi)$ as inputs to the model, rather than φ directly. The output of the model is the sensor range value ν (see figure 8).

The NARMAX models obtained were of degree 5 and no input or output lags

12

were used (i.e. $l = 5$, $N_u = 0$, $N_y = 0$). For the signature shown in figure 7 the following polynomial was obtained:

$$S^{mp}_{(1.78,2.31)} = \tag{4}$$
$$+1.489$$
$$-1.625 * u_1$$
$$-0.766 * u_2$$
$$-0.602 * u_1^2$$
$$+4.726 * u_1^3$$
$$+1.258 * u_2^3$$
$$+0.379 * u_1^4$$
$$-2.860 * u_1^5$$
$$+0.329 * u_1 * u_2(n)$$
$$-0.164 * u_1 * u_2(\varphi)^3$$
$$+1.178 * u_2 * u_1(\varphi)^4$$

where

$$u_1 = sin(\varphi)$$
$$u_2 = cos(\varphi)$$
$$\varphi \in [-\pi, +\pi]$$

Following the example of the grid location (1.78, 2.31) the model-predicted laser signature values at this location produced by evaluating the function above for values of φ in the range $[-\pi, +\pi]$ is shown in figure 9. The approximated sensor signature at location (1.78, 2.31) is also shown for comparison.

In a similar way, the complete environment model is composed by finding the models corresponding to each approximated grid signature. Note that the degree of the model polynomials (here degree 5) is adaptive and is decided based on how well the model fits the experimental data. This can be bigger or smaller depending on the complexity of the sensor signature which is related to the complexity of the environment topology.

When the environment model is used during simulation, the robot's sensor perception is determined using the polynomial of the environment model that corresponds to the grid location nearest to the robot's actual position. This is done by simply evaluating the selected polynomial for the range of angles which correspond to the simulated robot sensor angles.
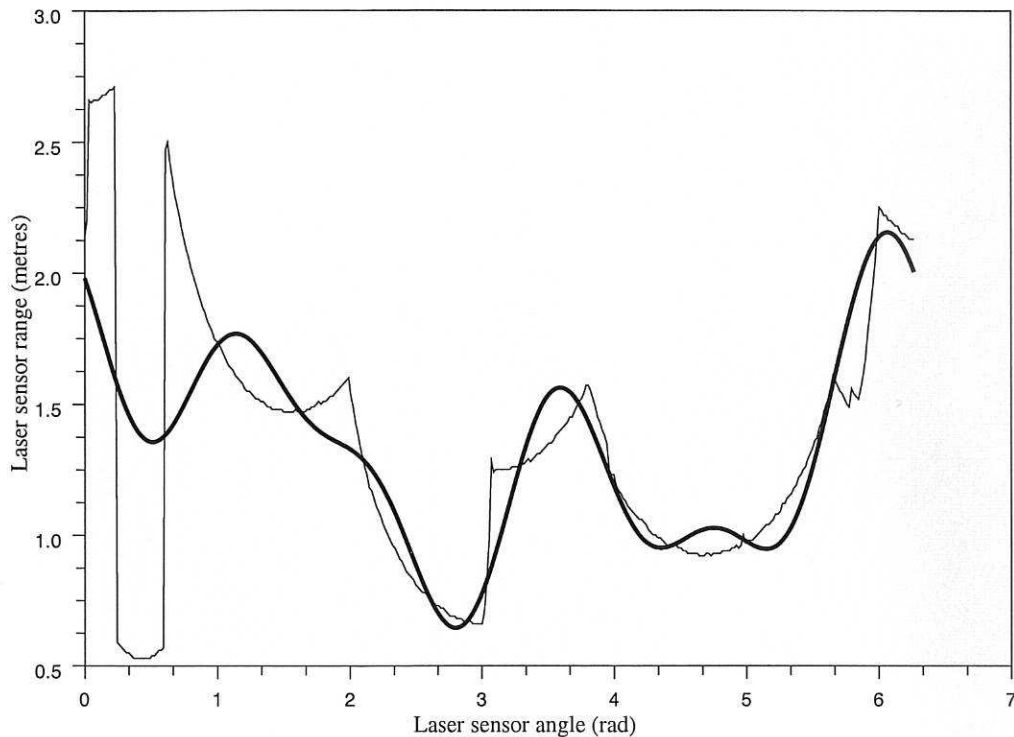
Fig. 9. Approximated (thin line) and model-predicted (thick line) sensor signature at location (1.78, 2.31) metres.

2.2.2 Model validation

In order to validate the environment models obtained by the method described thus far, test data is collected in the real environment while the robot executes a new, different task in the laboratory (the *validation task*) right after the exploratory phase. The validation task is also executed in the robot simulator and the two obtained trajectories are compared. A complete example of environment modelling and validation is presented in section 3.

2.2.3 Method 2: Piecewise signature modelling

We implemented an alternative to the above method that improves the prediction accuracy of the environment model considerably by modelling smaller segments of each grid signature rather than the entire signature. Here, each grid signature is modelled using *a set* of NARMAX polynomial functions instead of only one. We call this the piecewise signature modelling method.

The segments of the signature modelled are chosen to be those between cusps, the points where there is a relatively big change either in the value or the derivative of the signature. In a typical environment this occurs when the sensor faces a surface as the robot moves, either at a different angle compared

14

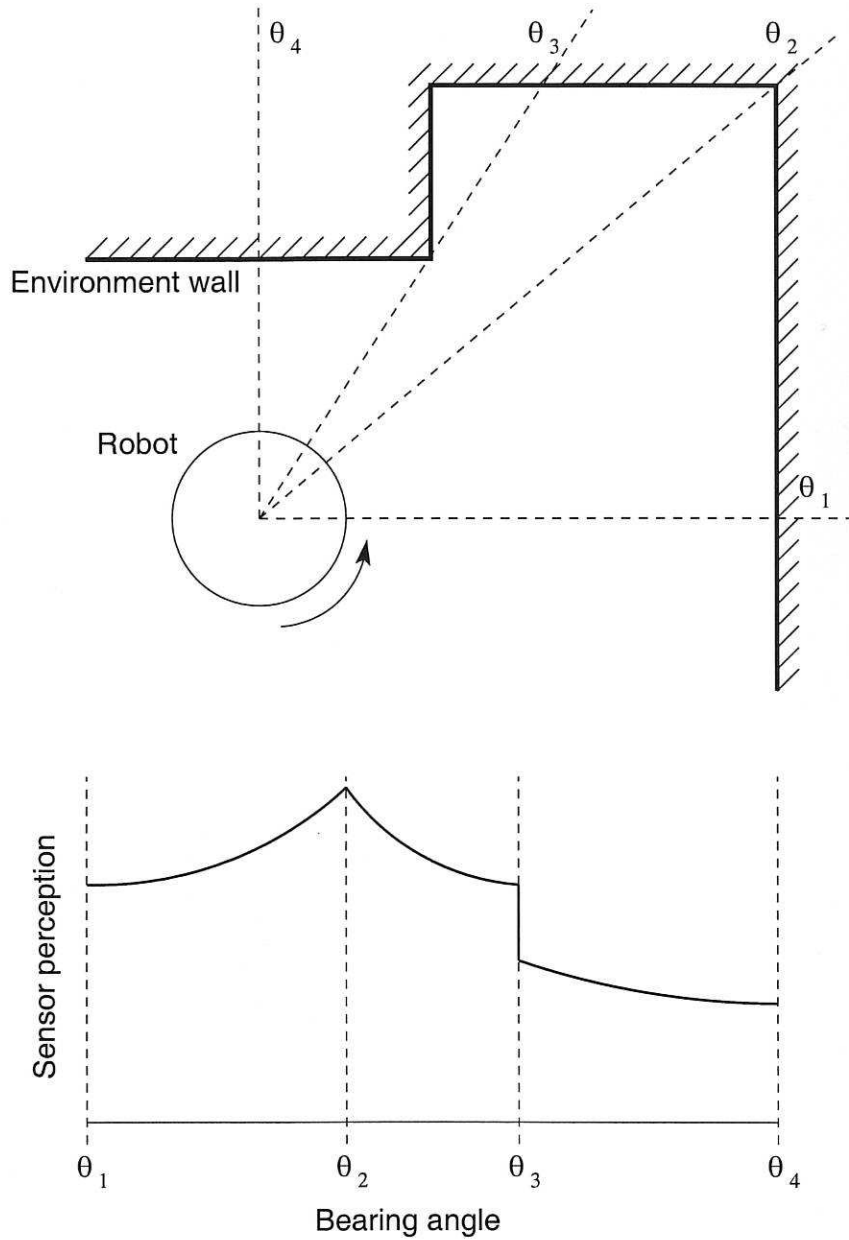with the previous surface or at a different distance (see figure 10).

Fig. 10. Two occasions where cusps appear in the sensor signature. As the robot rotates anti-clockwise at the same location one laser sensor scans the wall surface from φ_1 to φ_4. Two cusps appear in the signature recorded: the first at φ_2 and the other at φ_3. Actually in the latter case two cusps exist, one on top of the other.

Figure 11 shows the location of the cusps of the laser sensor signature shown in figure 7.

The complete environment model $\mathcal{M}^{\mathcal{PSM}}$ is now defined as:
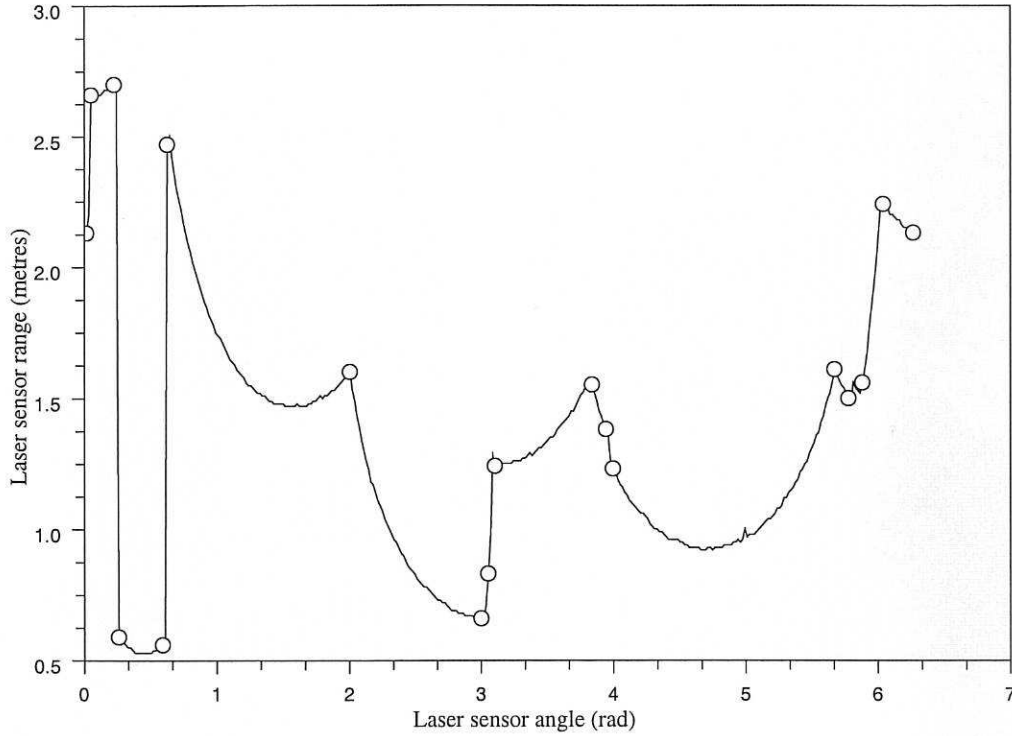
Fig. 11. The circles indicate the location of the cusps. These are determined from the magnitude of the second derivative of the sensor signature.

$$\mathcal{M}^{\mathcal{PSM}} = \{S^{mp}\} \tag{5}$$

$$S^{mp}_{(x_m, y_n)} = \{f^i(\varphi), \; \varphi_i \leqslant \varphi < \varphi_{i+1}\} \tag{6}$$

$$m = 1, ..., w$$
$$n = 1, ..., h$$
$$i = 1, ..., k$$
$$\varphi \in [-\pi, +\pi]$$

where $S^{mp}_{(x_m, y_n)}$ is now a set of k polynomials each of which models a segment of the approximated sensor signature at grid location (x_m, y_n).

As an example, the piecewise model of the signature shown in figure 11 is given in table 1, figure 12 shows the piecewise model-predicted sensor signature and the approximated signature.

where

$$u_1 = sin(\varphi)$$
$$u_2 = cos(\varphi)$$

16

$$
S^{mp}_{(1.78,2.31)} = \begin{cases}
+2.130 \;,\; \varphi = 0.000 \\
+2.065 + 15.186 * u_1 \;,\; 0.000 < \varphi < 0.035 \\
+5.349 - 0.042 * u_1 - 2.695 * u_2 \;,\; 0.035 \leqslant \varphi < 0.209 \\
+15.945 - 61.998 * u_1 \;,\; 0.209 \leqslant \varphi < 0.244 \\
+4.063 - 1.482 * u_1 - 3.210 * u_2 \;,\; 0.244 \leqslant \varphi < 0.576 \\
-35.633 + 65.876 * u_1 \;,\; 0.576 \leqslant \varphi < 0.611 \\
+3.772 - 2.336 * u_1 - 0.051 * u_2 \;,\; 0.611 \leqslant \varphi < 1.990 \\
+3.203 - 0.793 * u_1 + 2.404 * u_2 \;,\; 1.990 \leqslant \varphi < 2.985 \\
+1.138 - 3.238 * u_1 \;,\; 2.985 \leqslant \varphi < 3.037 \\
+1.772 - 8.793 * u_1 \;,\; 3.037 \leqslant \varphi < 3.089 \\
+2.719 - 0.005 * u_1 + 1.471 * u_2 \;,\; 3.089 \leqslant \varphi < 3.822 \\
+2.953 + 2.233 * u_1 \;,\; 3.822 \leqslant \varphi < 3.927 \\
+4.624 + 4.580 * u_1 \;,\; 3.927 \leqslant \varphi < 3.979 \\
+2.283 + 1.375 * u_1 + 0.043 * u_2 \;,\; 3.979 \leqslant \varphi < 5.655 \\
+0.890 - 1.220 * u_1 \;,\; 5.655 \leqslant \varphi < 5.760 \\
+1.747 + 0.481 * u_1 \;,\; 5.760 \leqslant \varphi < 5.864 \\
+7.620 + 6.408 * u_1 - 3.803 * u_2 \;,\; 5.864 \leqslant \varphi < 6.021 \\
+3.546 - 0.265 * u_1 - 1.426 * u_2 \;,\; 6.021 \leqslant \varphi < 6.266 \\
+2.130 \;,\; \varphi = 6.266
\end{cases} \tag{7}
$$

Table 1
Piecewise model of the sensor signature shown in figure 11

In simulation, the environment model obtained using the piecewise approach is used to obtain the robot's sensor perception in a similar fashion as with model $\mathcal{M}^{CSM}$ but in this case there is an additional step: The bearing of the sensor whose value is sought determines which polynomial function of the signature must be used. In the example given above (equation 7), if the robot in simulation is nearest to location $(1.78, 2.31)$ metres so that model $S^{mp}_{(1.78,2.31)}$ is selected and, say, a single laser sensor component is facing 0.75 rad, then the polynomial

$$
S^{mp}_{(1.78,2.31)}|_{\varphi=0.75} = +3.772 - 2.336 * u_1 - 0.051 * u_2
$$

would be used to obtain the model-predicted perception of that laser sensor component (in this case 2.14 metres).

In the following section we present the results of an environment modelling
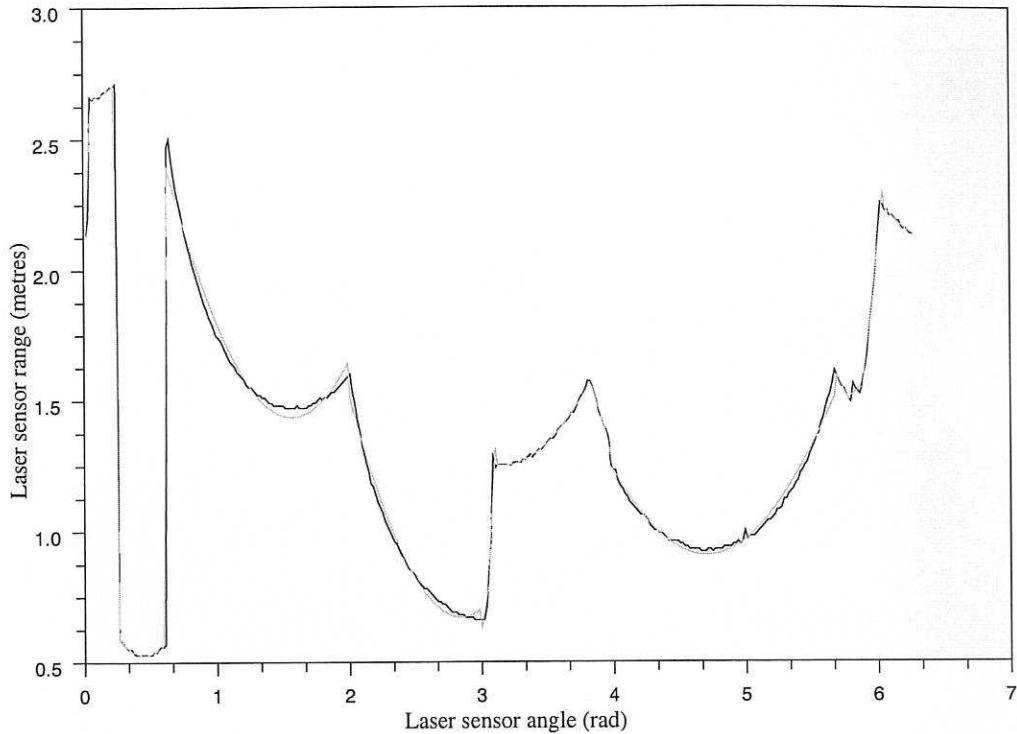
17

Fig. 12. Approximated (dark-coloured line) and model-predicted (light-coloured line), using the piecewise modelling method, sensor signature at location (1.78, 2.31) metres.

example using both modelling approaches explained above. Both models obtained are tested by comparing the robot's behaviour in a simulator and that in the real world for the same robot task.

3 Experimental results

The environment shown in figure 4 was modelled using the two methods described in the previous section. Only the laser sensor was modelled in the following example.

Initially the robot was allowed to execute an exploratory task using its laser sensor. This was essentially an obstacle avoidance task. The aim was to cause the robot to visit the entire environment as thoroughly as possible in order to record, through its sensors, all features in the environment. The trajectory of the robot after executing the exploratory task in the environment for approximately 2 hours is shown in figure 5.

The grid locations whose signatures are approximated using sensor data from the exploration phase are shown in figure 6. The grid spacing in both x and y directions is 0.1 metres.

18

Using the two modelling methods described in section 2.2, two models of the environment were obtained: first, by modelling the entire approximated laser sensor signature of each grid location using *one* NARMAX polynomial (Complete Signature Model $\mathcal{M}^{CSM}$, see section 2.2); second, by modelling the approximated laser sensor signature of each grid location using *multiple* NARMAX polynomials (Piecewise Signature Model $\mathcal{M}^{PSM}$, see section 2.2.3). Model $\mathcal{M}^{CSM}$ resulted in a set of 680 polynomials (one for each grid location), model $\mathcal{M}^{PSM}$ in a set of 14280 polynomials (i.e. an average of 21 polynomials for each grid location).

In order to evaluate each model, the behaviour of the robot while running a particular, new task in the simulated environment (using either of the models) was compared with that of the real robot executing the same task in the real environment. This test data was obtained in the environment that had been modelled during the initial exploration shown in figure 5. The test control program was a wall-following task using the robot's laser sensor. The trajectory of the robot during the collection of the test data is shown in figure 13. The same wall-following task was then implemented in a simulator using, first, model $\mathcal{M}^{CSM}$ and then model $\mathcal{M}^{PSM}$.
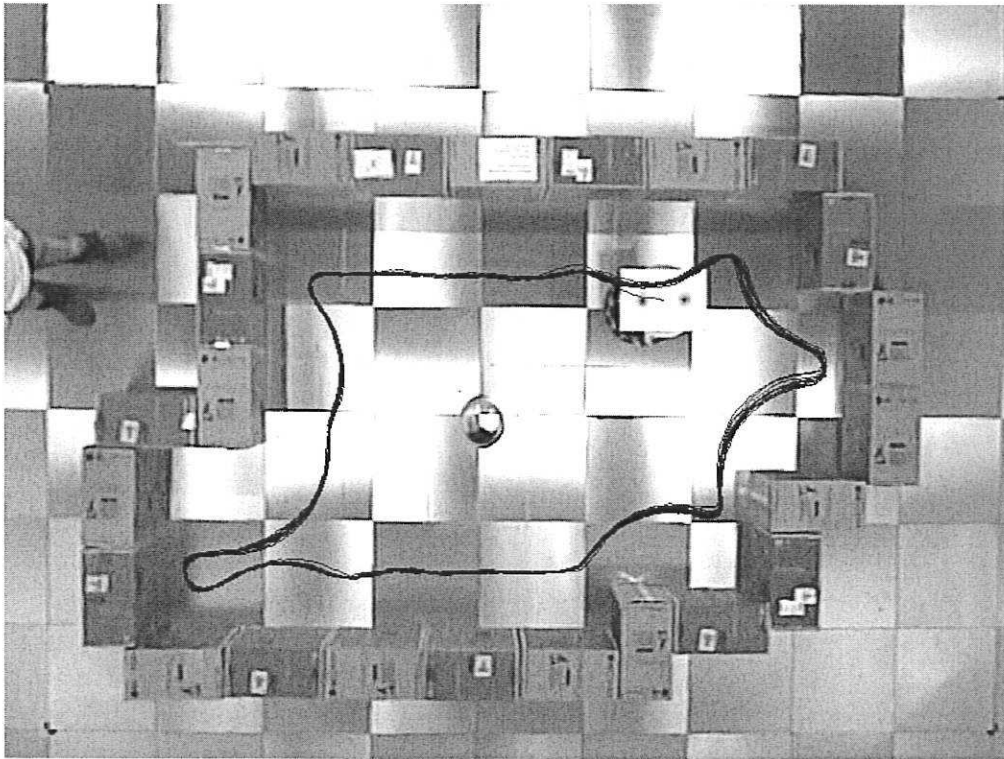

Fig. 13. The trajectory of the robot during the execution of the wall-following task in the environment.

Figures 14 and 15 show the trajectory of the robot in simulation (using models $\mathcal{M}^{CSM}$ and $\mathcal{M}^{PSM}$ respectively) with the trajectory of the robot in the real

19

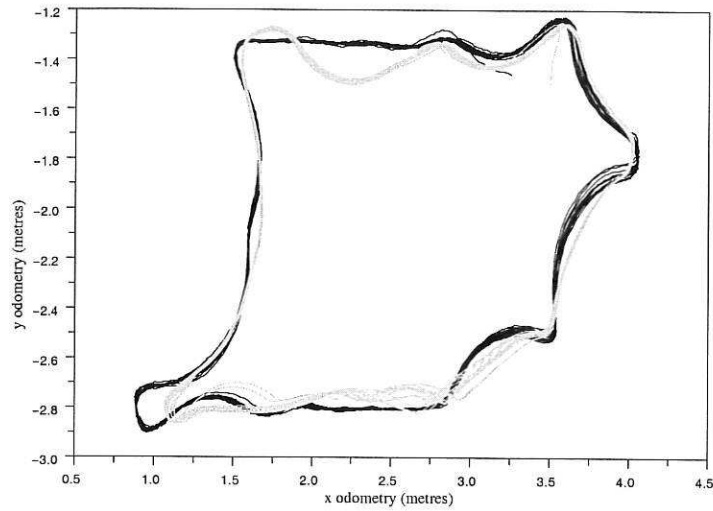world (used as the "ground truth") when executing the wall-following task.

Fig. 14. The trajectory of the robot in the simulator using environment model $\mathcal{M}^{\mathcal{CSM}}$ (light-coloured line). This is the environment model which uses *one* NARMAX polynomial for each grid location. The robot's trajectory obtained during the collection of the test data (ground truth) is also shown (dark-coloured line) for comparison.

Fig. 15. The light-coloured line shows the trajectory of the robot in the simulator using environment model $\mathcal{M}^{\mathcal{PSM}}$ (piecewise signature model). This is the environment model which uses *several* NARMAX polynomial for each grid location. The robot's trajectory obtained during the collection of the test data (ground truth) is also shown (dark-coloured line) for comparison.
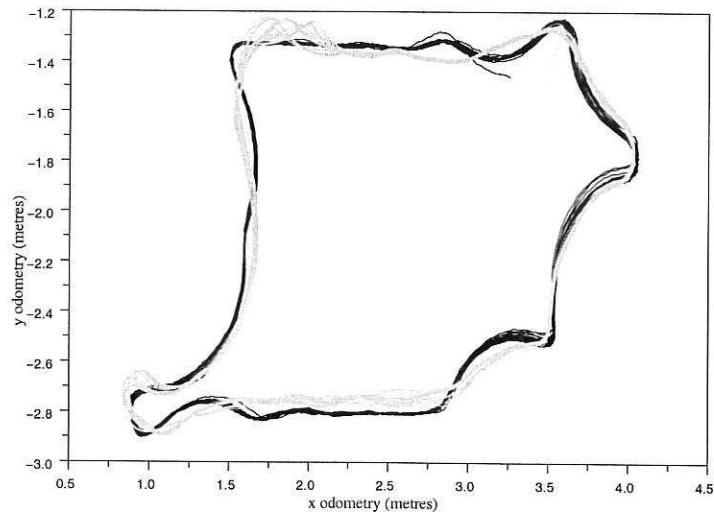
In order to compare the performance of the two environment models with a standard look-up table simulation methods (such as the one used in [16]) a further simulation run was performed using this approach. This was done by using the signature of the nearest grid location to the simulated robot's

location to determine the robot's sensor perception. The trajectory of the simulated robot during this run is shown in figure 16.
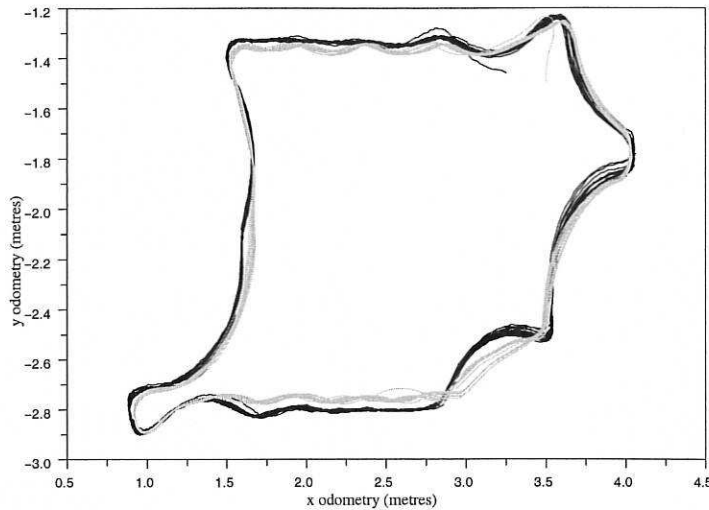

Fig. 16. The trajectory of the robot in the simulator using sensor data directly from the approximated grid signatures (look-up table model) to determine the sensor perception of the simulated robot (light-coloured line). The robot's trajectory obtained during the collection of the test data (ground truth) is also shown (dark-coloured line) for comparison.

To assess the faithfulness of each of the three models obtained we conducted a qualitative and a quantitative comparison between the pairs of trajectories presented in figures 14, 15 and 16, which is discussed in the following section.

3.1   Qualitative assessment of experimental results

Visual comparisons between figures 14 and 15 show that the piecewise environment model $\mathcal{M}^{PSM}$ produces a trajectory that matches the test trajectory more closely than the trajectory obtained with the complete signature model $\mathcal{M}^{CSM}$. This is particularly noticeable at the vicinity of the turn at (1.0, -2.8) metres and the part where the robot follows the straight wall in the environment which appears at the top of image 13. This was expected since $\mathcal{M}^{PSM}$ models the grid signatures more precisely than $\mathcal{M}^{CSM}$.

We also observe qualitatively that the trajectory predicted by the look-up table model (figure 16) appears to follow the actual robot trajectory more closely than those trajectories predicted by models $\mathcal{M}^{CSM}$ and $\mathcal{M}^{PSM}$.

In general, however, we can see that all simulated trajectories display the characteristics of the original wall-following behaviour very well. We believe that some of the differences between the actual and simulated behaviours can be

21

attributed to inevitable changes in the environment and/or the robot properties regardless of our best efforts to keep these constant. The exploration data (with which the models were estimated) and the test data were collected in two consecutive days because the robot had to be recharged in the meantime. Small changes in the positioning of the environment setup, variations of environmental conditions, differences in robot battery voltage level *etc.* could all have contributed to different robot sensor and/or actuator properties during the collection of the two data sets.

3.2 Quantitative assessment of experimental results

3.2.1 What is a faithful robot simulator?

Qualitative comparisons between original and model prediction, such as the ones given in figures 14, 15 and 16 will give an intuitive "feel" for the fidelity of a model, but do not allow any precise, quantitative comparison. They do not, in other words, provide any scientific argument to prefer one model over another.

The method we propose in this paper to compare different models with each other is to i) define a performance criterion that captures the essence of the simulation, and ii) to compare the performance of different models with respect to this criterion, using a statistical analysis.

Kohler and Wehner have recently published a very interesting analysis of trajectories of desert ants *melophorus bagoti* [22] that is relevant to this aspect of this paper. In their experiment, the ants' task was to return home from a distant location. Taking the straight line between release site and home location as a reference, Kohler and Wehner then analyse the deviation to the left or right of each path from that reference line, using an analysis of variance test.

We analysed the model predictions obtained in our experiments in a similar way, taking one circuit of the reference trajectory as a baseline, and statistically analysing the distance of all other trajectories, including different circuits of the reference trajectory, to that baseline. This is shown in figure 17.

3.2.2 Statistical analysis of simulation results

Besides comparing the simulator-predicted trajectories and actually observed trajectories visually (section 3.1), we were interested to *measure* if predictions and actual trajectory differ in a statistically significant way. Quantitative trajectory analysis is still a young research area, and only very few examples of a quantitative comparison between robot trajectories are found in the literature,
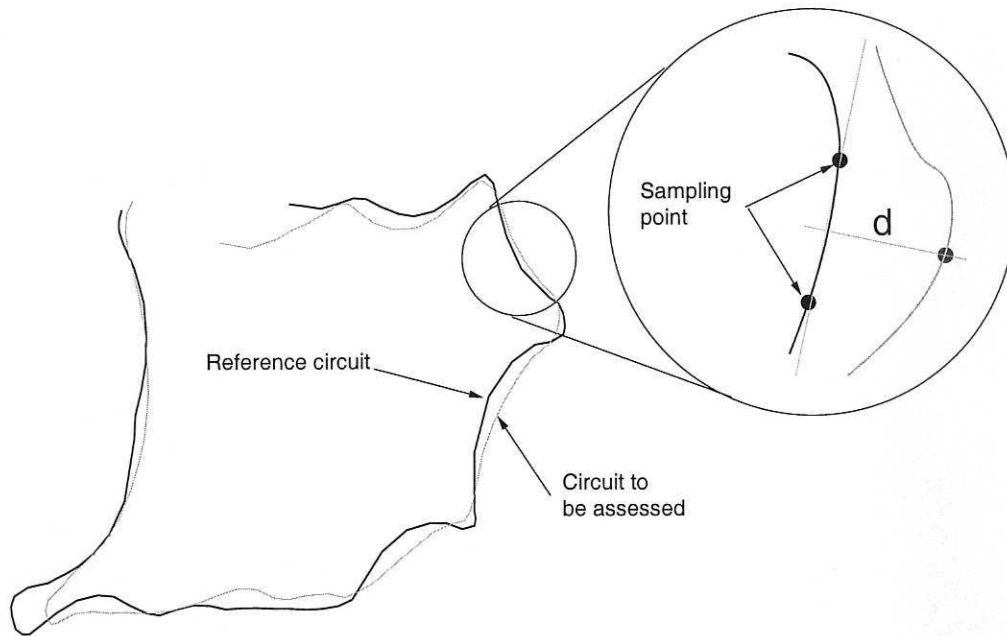
Fig. 17. Trajectories were compared by assessing the distribution of distances to a reference trajectory. The distance d between the reference trajectory (dark line) and the trajectory that is to be assessed (light line) is defined as shown above.

for example [19] and [23]. Here, we are interested to determine if the trajectories shown in figures 14, 15 and 16 resemble the actual trajectory taken by the physical robot *globally*, i.e. whether the predicted trajectories deviate to the left and the right of the reference trajectory (the trajectory of *Radix* in the laboratory) in the same manner that the robot's own trajectory deviates as the robot completes separate rounds of its wall following task.

As stated above, we took one circuit of the reference trajectory as a baseline, and determined the distance of circuits of all other trajectories, including different circuits of the reference trajectory, to that baseline (figure 17).

For each of the four trajectories — continuous model, piecewise model, interpolated model and actual robot trajectory — we took 15 circuits (shown in figure 18), and computed the distribution of distances to the reference circuit (which was an additional circuit taken from the reference trajectory). These distributions are shown in figure 19.

We then established whether there is a statistically significant difference in the deviation from the reference circuit and any of the four trajectories: The four distributions do *not* differ from each other significantly (parametric ANOVA, $p > 0.05$), meaning that all four trajectories deviate from the reference circuit in the same manner.

This means that lookup table model or polynomial model do not differ with
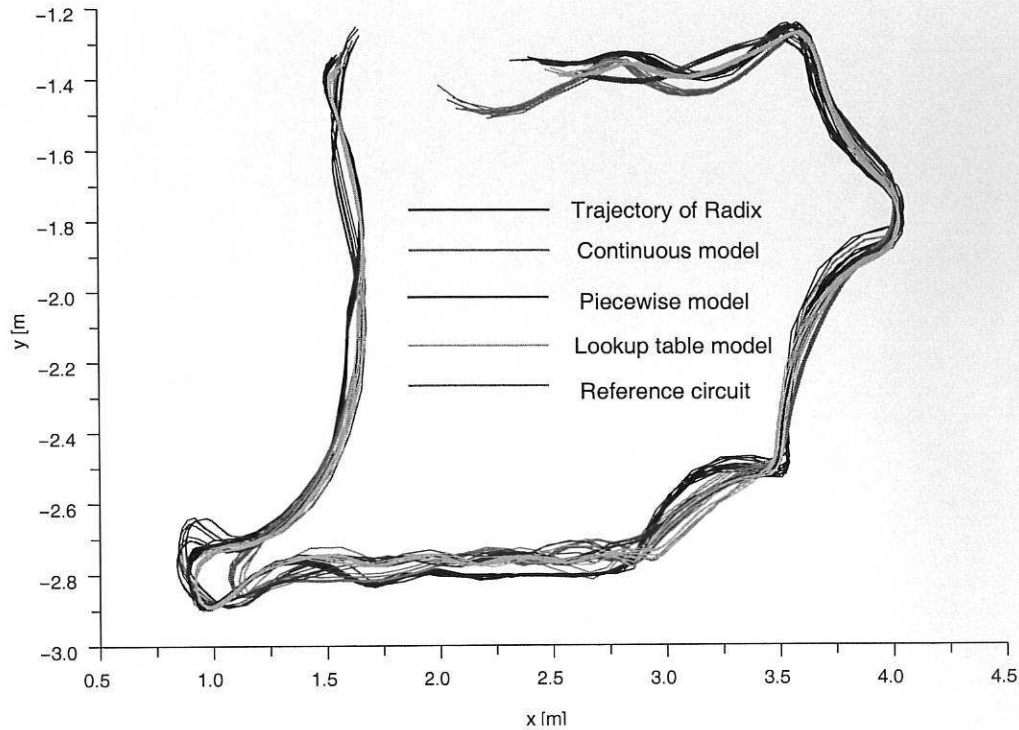
23

Fig. 18. Actual circuits taken by the robot, circuits predicted by the three simulators, and reference circuit used for comparison (see figure 17)

respect to their deviation around the reference circuit, i.e. that they have a similar global fidelity to the original. We argue, however, that a polynomial NARMAX model is preferable for two reasons:

(1) The models obtained are transparent and are thus amenable to analysis using established mathematical tools. Such analysis can lead to the characterisation of the environment and the determination of those important factors that predominantly influence the robot's behaviour. For examples of such analysis of NARMAX models see [3] and [18].

(2) The polynomial models occupy considerably less memory space compared to the look-up table model. In the example presented above, the look-up table model occupied approximately 2.1MB of memory whereas the two NARMAX models occupied 1.3MB ($\mathcal{M}^{\mathcal{CSM}}$) and 0.9MB ($\mathcal{M}^{\mathcal{PSM}}$)[1]. Such saving become important when larger and more complex environments are modelled.

---

[1] This may come as a surprise as $\mathcal{M}^{\mathcal{PSM}}$ comprises of a considerably larger number of polynomials compared to $\mathcal{M}^{\mathcal{CSM}}$, however the polynomials in $\mathcal{M}^{\mathcal{CSM}}$ are of much higher degree than those in $\mathcal{M}^{\mathcal{PSM}}$ which results in them containing more terms and thus taking more memory space
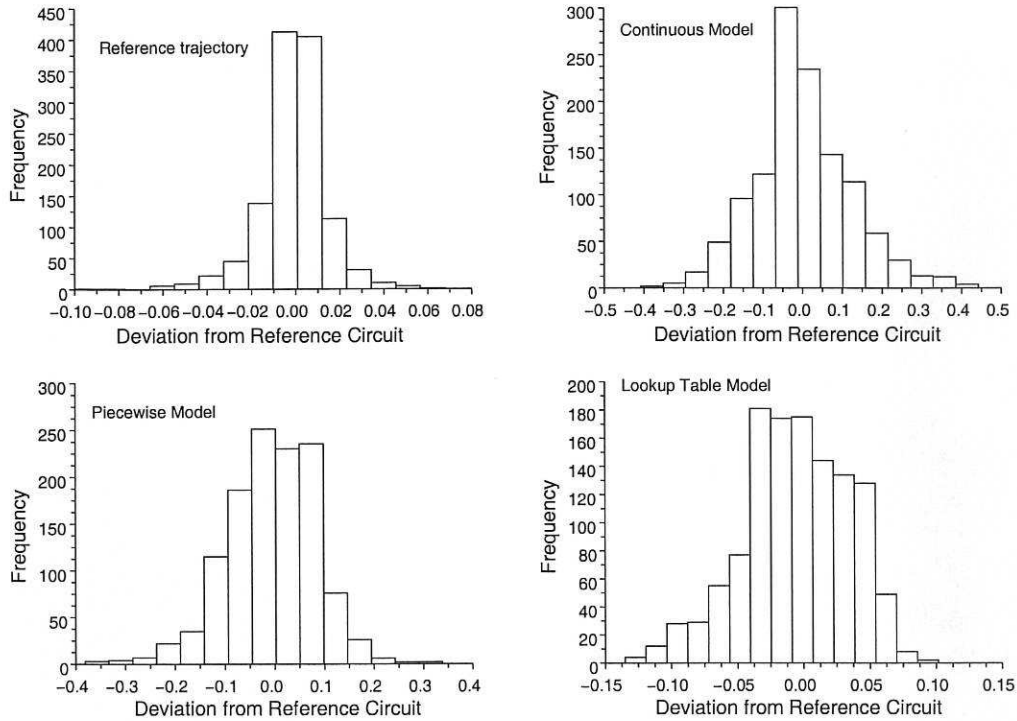
Fig. 19. Distributions of distances measured to the reference circuit for the reference trajectory itself, the continuous model, the piecewise model, and the model based on a lookup table.

4   Discussion

4.1   Summary

We have presented the RobotMODIC (Robot MODelling Identification and Characterisation) approach for robot environment modelling using the NAR-MAX model estimation methodology.

Initially, a real robot is used to sample the environment in order to collect location and corresponding sensor information to be used for the model estimation process. The collected data are used to approximate the sensor perception of the robot at each node of a regular grid of locations which covers the entire environment area. Two methods are presented for modelling the approximated sensor perception signatures at every grid location:

(1) using a *single* NARMAX polynomial to model the signature as a whole, and
(2) using a *set* of NARMAX polynomials to model the signature in a piece-wise fashion.

25

In order to assess the accuracy of each of the estimated models we compared the behaviour of the robot when executing a particular task in the real world with that when executing the same task in simulation.

No statistically significant difference was found between the real-world and simulated robot behaviours in our example.

4.2 Conclusions

Purpose   The purpose of the work presented in this paper is to obtain accurate robot simulators that speed-up the development of robot control programs.

The interaction of a robot with its environment is complex and can be expressed in many ways (trajectory, velocity, acceleration, battery voltage, response to obstacles *etc*). Here we have chosen the trajectory profile of the robot to describe its function (wall-following) because we felt that this would represent the task best. Future work will look into the comparison of different modes of behaviour while the robot is performing the same task or even different tasks (such as obstacle avoidance, route learning *etc*). This will allow a more complete and thorough comparison between the simulated and real environments.

Contrast to SLAM   The method presented in this paper is not to be confused with a SLAM (Simultaneous Localisation And Mapping) method such as the one presented in [24]. The purpose of the work presented here is the generation of a faithful simulator for location-perception mappings, rather than map-building and localisation for navigation purposes.

Contrast to sensor signal interpretation   [25] and [26] describe how to interpret the robot's sensor perception (the sonar sensor in particular) in order to recognise specific features of the environment. Again, this is quite different from what we expect to attain in the work presented here. We do not aim to identify environment features, but would like to be able to reproduce the robot's perception accurately as a function of its position. Different tasks such as SLAM or landmark identification algorithms, for example, can then be compared qualitatively under the exact same environment model.

The most important aspect of our method is that it uses *transparent* mathematical functions to model the environment. This allows the analysis of the models, which in turn will provide a better theoretical understanding of the complex interaction between a robot and its environment.

Future Work This paper addressed the issue of faithful, transparent and abstracted modelling of robot-environment interaction, and did so by introducing a modelling method based on system identification. To assess the fidelity of the obtained models, two identified models and a lookup-table model were compared with each other in a laboratory environment.

We see two possible extensions to this work, both of which are under investigation in our laboratories: i) to conduct experiments in larger, more complex and possibly dynamic environments, and ii) to compare our system identification method with other commonly used modelling methods, using quantitative statistical analysis.

Acknowledgements

References

[1] T. Kyriacou, U. Nehmzow, R. Iglesias, and S. Billings, "Cross-platform programming through system identification," in *Proceedings of TAROS 2005*, London, 2005.

[2] U. Nehmzow, T. Kyriacou, R. Iglesias, and S. Billings, "Self-localisation through system identification," in *Proceedings of the European Conference on Mobile Robotics (ECMR)*, Ancona, Italy, 2005.

[3] R. Iglesias, T. Kyriacou, U. Nehmzow, and S. Billings, "Programming through system identification and training," in *Proceedings of the European Conference on Mobile Robotics (ECMR)*, Ancona, Italy, 2005.

[4] T. Ziemke, "On the role of robot simulations in embodied cognitive science," *AISB Journal*, vol. 1, no. 4, pp. 389–399, 2003.

[5] B. Gerkey, R. Vaughan, and A. Howard, "The player/stage project: Tools for multi-robot and distributed sensor systems," in *Proceedings of the International Conference on Advanced Robotics*, 2003.

[6] O. Michel, "Webots: Professional mobile robot simulation," *Advanced Robotic Systems*.

[7] I. Nomadic Technologies, *Nomad 200 Mobile Robot User's Guide*, Mountain View, CA., 1996.

[8] K. G. Konolige, *Saphira Software Manual*, SRI International, Menlo Park, California, 2001, saphira Version 8, Saphira/Aria Integration.

[9] O. Michel, *Kephera Simulator Version 2.0 User Manual*, 1996.

[10] A. Koestler and T. Braunl, "Mobile robot simulation with realistic error models," in *Proceedings of the International Conference on Autonomous Robots and Agents (ICARA 2004)*, Palmerston North, New Zealand, 2004, pp. 46–51.

[11] A. Kleiner and T. Buchheim, "A plugin-based architecture for simulation in the f2000 league," in *RoboCup Symposium*, 2003.

[12] R. A. Brooks and M. J. Mataric, *Real Robots, Real Learning Problems.* Kluwer Academic Publishers, 1993, ch. 8.

[13] T. Smithers, "On why better robots make it harder," in *From Animals to Animats 3, Proceedings of the 3rd International Conference on Simulation of Adaptive Behaviour, SAB'94*, Brighton, England, 1994.

[14] T.-M. Lee, U. Nehmzow, and R. Hubbold, "Mobile robot simulation by means of aquired neural network models," in *Proc. of the 12th European Simulation Multiconference, Manchester*, 1998.

[15] T.-M. Lee, "A new approach to mobile robot simulation by means of aquired neural network models," Ph.D. dissertation, University of Manchester, 2000.

[16] H. H. Lund and O. Miglino, "From simulated to real robots," in *Proc. IEEE 3rd International Conference on Evolutionary Computation.* IEEE Press, 1996.

[17] S. Chen and S. Billings, "Representations of non-linear systems: The narmax model," *International Journal of Control*, vol. 49, pp. 1013–1032, 1989.

[18] R. Iglesias, U. Nehmzow, T. Kyriacou, and S. Billings, "Modelling and characterisation of a mobile robot's operation," in *Proceedings of CAEPIA 2005, 11th conference of the Spanish association for Artififcial Intelligence*, Santiago de Compostela, Spain, November 2005.

[19] U. Nehmzow, *Scientific Methods in Mobile Robotics: Quantitative Analysis of Agent Behaviour.* Springer Verlag, 2006.

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

[21] M. Korenberg, S. Billings, Y. P. Liu, and P. J. McIlroy, "Orthogonal parameter estimation algorithm for non-linear stochastic systems," *International Journal of Control*, vol. 48, pp. 193–210, 1988.

[22] M. Kohler and R. Wehner, "Idiosyncratic route-based memories in desert ants, melophorus bagoti: How do they interact with path-integration vectors?" *Neurobiology of Learning and Memory*, vol. 83, no. 1, pp. 1–12, 2005.

[23] P. Roduit, A. Martinoli, and J. Jacot, "Behavioral analysis of mobile robot trajectories using a point distribution model," in *Proc. of the 9th Int. Conference on the Simulation of Adaptive Behavior*, vol. 4095, 2006, pp. 819–830.

[24] J. D. Tardos, J. Neira, P. M. Newman, and J. J. Leonard, "Robust mapping and localization in indoor environments using sonar data," *The International Journal of Robotics Research*, vol. 21, no. 4, pp. 311–330, 2002.

[25] R. Kuc, "Recognizing retro-reflectors with an obliquely-oriented multi-point sonar and acoustic flow," *The International Journal of Robotics Research*, vol. 22, no. 2, pp. 129–145, 2003.

[26] W. Gao and M. Hinders, "Mobile sonar backscatter algorithm for automatically distinguishing walls, fences and hedges," *The International Journal of Robotics Research*, vol. 25, no. 2, pp. 135–145, 2006.