LEARNING THE VISUOMOTOR COORDINATION OF A MOBILE ROBOT BY USING THE INVERTIBLE KOHONEN MAP

C. Versino[*], L.M. Gambardella
IDSIA, Corso Elvezia 36, 6900 Lugano,Switzerland
cristina@idsia.ch, luca@idsia.ch, http://www.idsia.ch

Abstract

In this paper we present an experiment of the *visuomotor coordination* of a simple mobile robot. Our approach to sensorimotor modelling belongs to the category of *learning by doing*. First, $< perception, action >$ pairs are collected by observing the robot behavior during operation. Then, a learning by examples method is used to estimate the parameters of the model. In our experiment the learning machine is an *Extended Kohonen Map*. We have observed that this neural network model has the property of being naturally *invertible*. Given an input pattern, the network output value is retrieved by competition on the neuron fan-in weight vectors. This is the standard use of the input-output mapping that we call *forward* mode. Viceversa, given an output value, a corresponding input pattern can be obtained by competition on the neuron fan-out weight vectors. We call this use of the network *backward* mode. The invertibility property makes the Extended Kohonen Map worth considering for sensorimotor modeling. Our experiment shows that by training the network on the robot direct kinematics (the forward mode), one obtains at the same time a solution to the inverse kinematics problem (the backward mode). The experiment has been performed both in a computer simulation and by using a real robot.

1 Introduction

Robotics research devotes considerable attention to the study of sensorimotor coordination [BDM94, BGG93, M94, RMS91, ZGLC94]. This term refers to the association of signals coming from various sensory modalities to motor commands in view of a given task. A widely studied instance of sensorimotor association is the *visuomotor coordination* of a robot arm [RMS91, BGG93]. The task is to automatically position the end effector of a robot arm at an arbitrary target location (τ_1, τ_2, τ_3) in a 3D environment (Figure 1, left). The information about the location of the target is provided to the robot system by two cameras which observe the workspace. The robot arm moves when a valid set of angles (in Figure 1, $(\vartheta_1, \vartheta_2, \vartheta_3)$) is selected and transmitted to the joint motors. The end effector positioning task requires visuomotor coordination because one needs to relate the visual information about the target location, as it is perceived by the camera system, to the movements of the arm.

Studies on sensorimotor coordination deal with the *bidirectional association* existing between sensory perception and motor action. On one side, there is a transformation from the space of motor actions to the space of resulting sensory situations: this part is commonly referred to as the *forward* association. On the other side, one needs to know how to generate motor commands to reach a target sensory perception: this part is referred to as the *inverse* association. In the robot arm example, the forward relationship corresponds to the ability of predicting the position of the end effector when a set of joint angles is transmitted to the arm motors. On the other side, the inverse association maps positions in the workspace into joint angles configurations that allow the robot to reach these positions.

From a methodological point of view, the study of robot sensorimotor coordination (both forward and inverse) may be addressed in two different ways. The classical approach consists of assuming a *a priori* model of the interaction between sensors and actuators. By *a priori model*, we mean, for instance, a model which is built on the basis of physical laws. In this way, the forward association, which predicts the robot arm position for a joint configuration, can be expressed as a direct kinematics equation, while the end effector positioning task is viewed as the solution to an inverse kinematics equation. The alternative approach consists of building a *a posteriori* description of the interaction between sensors and actuators. By the term *a posteriori* description, we mean a model built on data which implicitly

[*]Supported by the No. 21-36365.92 project of the Fonds National de la Recherche Scientifique, Berne, Suisse.

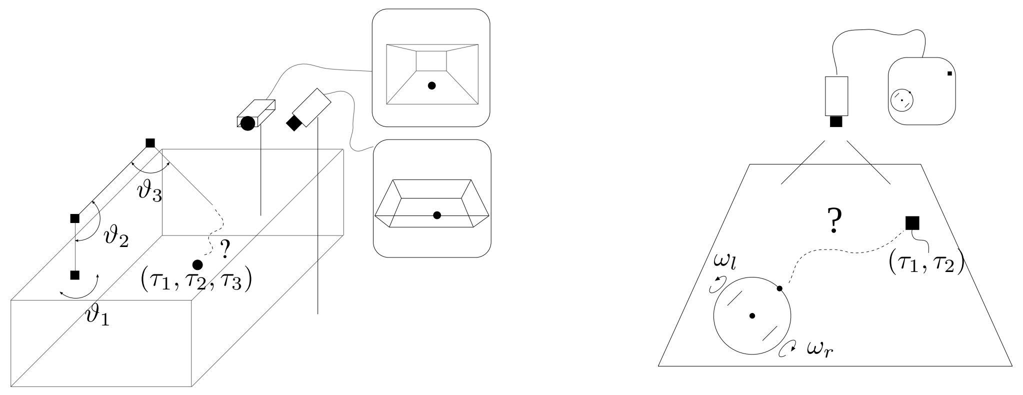


Figure 1: Experimental setup for the visuomotor coordination of a robot arm (left) and of a mobile robot (right).

describes the association between sensory perceptions and motor commands. First, $< perception, action >$ pairs are collected by observing the robot behavior during operation. Then, a *learning by examples* method is used to estimate the parameters of the model. Advocates of the *a posteriori* methodology stress the realism of their approach, in that no fixed model is assumed to be valid before experimenting with the real robot; on the contrary, the *a priori* model describes ideal interactions which are met in simulated environments only[1] [BDM94].

In this paper we study the *visuomotor coordination for a simple mobile robot by using the learning by examples scheme*. The task is to automatically guide a mobile robot to an arbitrary target location (τ_1, τ_2) in a 2D environment (Figure 1, right). As in the arm example, the target location with respect to the robot's current position is observed by a camera located above the workspace. The robot movement is controlled by choosing the angular velocities (ω_l, ω_r) for its left and right driving wheels. Hence, the task is to find a sequence of wheel velocities that would eventually bring the robot from any initial position in the workspace to any target location; in order to do that, the visual information provided by the camera is used. Although the main focus is put on a typical inverse task (the inverse kinematics of a mobile robot), we show that both forward and inverse associations can be addressed at the same time by an Artificial Neural Network (ANN) model, which has the property of being *invertible*.

The paper is organized as follows. In the second section, we review related work on visuomotor coordination addressed by the learning by examples approach, in particular those based on ANN. In the third section, we present the ANN model we have chosen for our experiment, namely an Extended Kohonen Map (EKM). We motivate the choice of this model by observing an interesting property of its input-output behavior, namely the invertibility, which makes it worth considering for the study of forward and inverse associations. In the fourth section, we present the experiment on the visuomotor coordination of a mobile robot addressed by the previously introduced ANN. We describe the $< perception, action >$ data collection, the ANN training and the performance phase, when the trained network guides the mobile robot to target locations in the workspace. Solution paths obtained in a computer simulation will be presented, as well as a description of an experiment performed on a real robot. Finally, we will draw some conclusion.

2 Related work on visuomotor coordination

A widely studied case of visuomotor coordination is the automatic positioning of the end effector of a robot arm in the experimental setup depicted in Figure 1. Following the learning by examples approach, Ritter, Martinetz and Schulten have realized an ANN, typically an EKM, which learns to gradually solve the arm inverse kinematics problem [RMS91]. Their methodology is referred to as *direct inverse modeling* and can be summarized as follows. During the network training phase, target positions are chosen at random in the workspace. Each target point is supplied as input data to the ANN, which decides a joint configuration. This is transmitted to the joint motors to move the end effector to a new position, which is recognized by the camera system. The difference between the target position and the one actually reached by the robot allows the computation of an "improved" joint angle configuration which can be taught to the network as target output value. Strictly speaking, the learning rule is *supervised*, because the network weights are modified to reproduce the improved joint configuration. However, one should notice that there is no *a priori* decided configuration for any given target position in the workspace: the improved joint angle configuration is derived by letting the robot operate and by observing the results of its action. Therefore, the authors claim that the robot represents an autonomous system which learns in a closed-loop mode: it receives all the information needed for adaptation from the cameras. This style of learning is correctly called *learning by doing*.

Arm trajectory generation has also been studied by Bullock, Grossberg and Guenther [BGG93]. They have used a neural architecture called DIRECT, which learns a transformation between spatial and velocity coordinates through

[1]While we essentialy agree with this statement, we wish to note that *a priori* knowledge is also present in the *a posteriori* approach. The decision of which variables are to be observed, the choice of a particular class of models as a suitable skeleton for sensorimotor description are all examples of *a priori* knowledge currently used to construct *a posteriori* models.

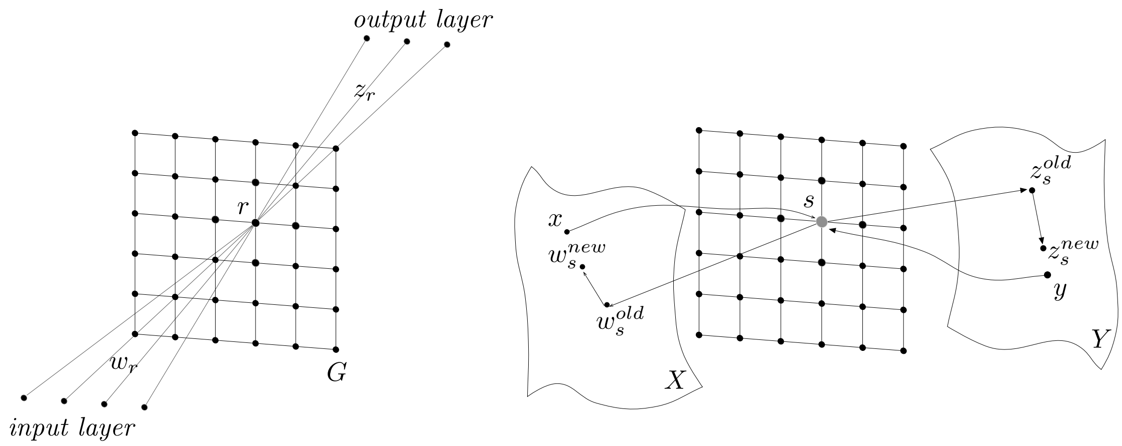


Figure 2: Extended Kohonen Map architecture (left) and learning step (right).

a sequence of spontaneously generated random movements.

Recently, following the same approach, Zalama, Gaudiano and López Coronado have proposed a neural network model (NNETMORC) for the control of a mobile robot in a 2D environment [ZGLC94]. Their model combines VAM learning and associative learning within an architecture similar to the DIRECT model.

In this paper we study the visuomotor coordination problem for a mobile robot as it is addressed in [ZGLC94], i.e. in the experimental setup of Figure 1. We have found that a very straightforward solution can be expressed in terms of an EKM. Although our work takes inspiration from the experiment on end effector positioning realized by Ritter, Martinetz and Schulten (in that we use the same neural model), *we don't use the direct inverse modeling* approach. Instead, our departure point is the observation of an interesting property of the EKM that greatly eases our task.

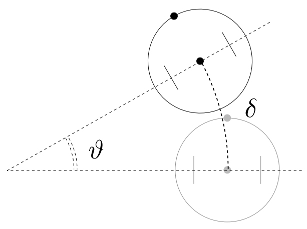
3 An interesting property of the Extended Kohonen Map

We assume that the reader is familiar with the Kohonen Map (KM) neural model [K84]. To describe our experiment we briefly introduce the required extension to the basic KM algorithm (EKM) proposed by Ritter and Schulten in [RS87], which enables us to train on output values the KM network by supervised learning. From the architecture point of view, the KM network is augmented by adding to each neuron r on the competitive grid G a fan-out weight vector z_r to store the neuron output value (Figure 2, left). The computation in the EKM network proceeds as follows. When an input pattern x is presented to the input layer, the neurons on G compete to respond to it. The competition involves the neurons fan-in weight vectors w_r, and consists of the computation of the *distance* [2] between x and each w_r. The neuron s, whose fan-in vector w_s is the closest to x, is declared the winner of the competition, and its fan-out vector z_s is taken as the network output answer to x.

During the training phase, both the input pattern x and the desired output value y proposed by a teacher are learned by the winning neuron and by its neighbors on the grid. The learning step consists of moving the fan-in weight vectors of the selected neurons closer to x, and their fan-out weight vectors closer to y (Figure 2, right). This learning style has been described as a *competitive-cooperative* training rule [RMS91]. It is *competitive* because the neurons compete through their fan-in weight vectors to respond to the presented input pattern. As a consequence, only that part of the network [3] which is relevant to deal with the current input data undergoes the learning process. Moreover, neighboring locations on the grid will correspond to fan-in weight vectors that are close to each other in input data space: this property is referred to as the *data topology-conserving* character of the network. The rule is also *cooperative* in that the output value learned by the winning neuron is partially associated to the fan-out weight vectors of its neighbors. The authors claim that if the input-output function to be learned is a continuous relationship, spreading the effect of learning an output value to the neighborhood of the winner represents a form of generalization which accelerates the course of learning [RMS91].

We are now in a position to make an observation about an interesting property of the input-output mapping realized by the EKM. We have noticed that this mapping is naturally *invertible*. To explain this point, suppose you have trained an EKM on a set of examples $< x, y >$, where x is a point in the input data space X and y is a point in the output data space Y. We see two possibile ways of using the map. The first use, which we call *forward*, consists of presenting an input pattern x to the network input layer to retrieve the corresponding output value, that will be an approximation to y. This is accomplished in the usual way, namely by letting the fan-in weight vectors w_r compete on x, and by taking the fan-out weight vector z_s of the winner s as the network output value. The second use, which we call *backward*, consists of presenting an output pattern y to the network output layer to retrieve a corresponding input value, that will be an approximation to x. This is achieved by letting the fan-out weight vectors z_r compete on

[2]The distance measure may be, for instance, the Euclidean distance.
[3]As defined by the neighborhood function.



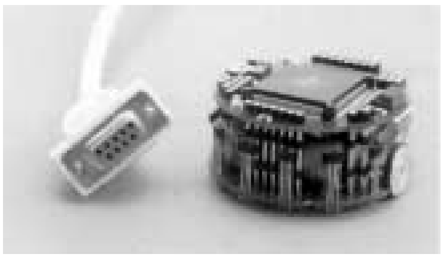


Figure 3: Displacement values for the robot (left side), the Khepera robot (right).

y, and by taking the fan-in weight vector w_s of the winner s as the network input value. The possibility of retrieving output values, through forward mode, and input values, through backward mode, is due to the fact that the EKM is completely symmetrical with respect to its input-output behavior, as the neuron activation is based on a distance measure [4]. Moreover, input and output patterns are learned by the network in exactly the same way (Figure 2).

Why is this property so interesting? Remember that our objective is to solve an inverse problem, namely the inverse kinematics of a mobile robot. Inverse tasks are often characterized by the existence of a well-defined forward task. In our case, the forward task corresponds to the robot direct kinematics. On the contrary, inverse tasks may be ill-posed, or may have multiple solutions. The observation above provides us with a rudimentary but effective approach to tackle forward and inverse problems at the same time. It suggests that, first, a model for the forward task may be constructed by training an EKM on examples of the forward association; these examples should be easily available. Second, by using the trained network in backward mode, we are provided with a solution to the inverse task. Thus, to apply this idea to our instance of inverse problem, it will be sufficient to train the network on the robot direct kinematics. The backward use of the network will be the solution to the inverse kinematics problem. In the next section we describe this procedure in detail.

4 Visuomotor coordination of a mobile robot by the invertible EKM

4.1 Example collection

To model the robot direct kinematics according to the *a posteriori* sensorimotor modeling approach, one has to generate a set of examples $< action, perception >$. In our case the relevant variables to record are the robot wheel velocities, for the action side, and the corresponding robot displacement, for the perception side. To be more specific, an example is a pair $< (\omega_l, \omega_r), (\vartheta, \delta) >$. (ω_l, ω_r) are the angular velocities of the left and right wheel, while (ϑ, δ) are, respectively, the robot heading change and the distance travelled by its axle mid-point during a fixed Δt time (Figure 3, left).

The example generation phase amounts to repeatedly selecting a velocity pair at random[5], applying it to the wheels for a fixed Δt time, and measuring the corresponding robot displacement by using the camera system. These examples, which implicitly describe the real robot kinematics, are the starting point for modeling the forward association between the space of motor commands and the space of resulting sensory situations. The advantage of the learning by doing approach to sensorimotor modeling lies in the fact that if the real robot em consistently deviates from the ideal behavior as predicted by a kinematics equation (if the robot is “malfunctioning”), this deviation will be included in the model[6].

In our experiment, examples of direct kinematics have been generated by using Khepera, a miniature robot [MFI93] (Figure 3, right). Khepera is equipped with two wheels, each controlled by an independent motor. The motor speed is expressed in integer units, one unit corresponding to 8 millimeter of advancement per second. With Khepera we generated a training set consisting of 1000 $< velocity, displacement >$ pairs, with velocity chosen in the range $[0, 5]$, and applied to the wheels for approximately one second.

4.2 Network training

Having collected a set of examples, the EKM training can take place. The network architecture is a 6×6 competitive grid, with two input units and two output units (Figure 4, left). The input units are used to encode the velocity pair, while the output units represent the robot displacement. The competitive grid dimension (6×6) was chosen to reflect the fact that each wheel may assume six speed values only. The network was trained as explained in the third section.

The result of the training phase is graphically depicted in Figure 4, right side. Each square is a neuron on the competitive grid. For each neuron, the learned fan-in and fan-out weight vectors have been represented. The fan-in weight vector is a velocity pair indicated by the position of the neuron on the (ω_l, ω_r) reference system. The fan-out

[4]This would not be possible, for instance, in a feed-forward network.
[5]With uniform probability in the space of valid speed range.
[6]Infact it is hard to talk about a deviation as there is no *a priori* expectation on how the robot should behave.

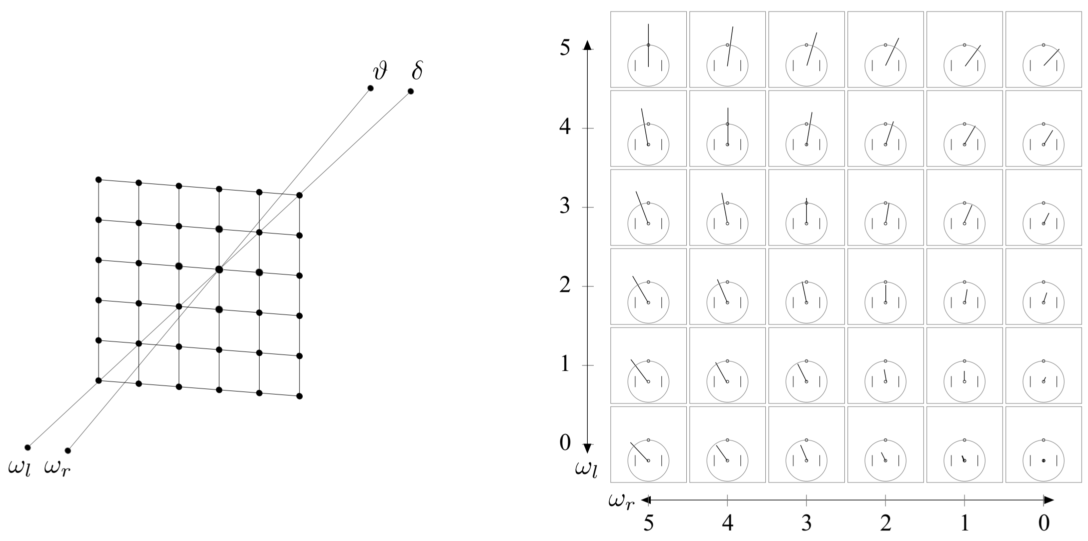


Figure 4: The network architecture of the experiment (left), representation of the network weights after training (right).

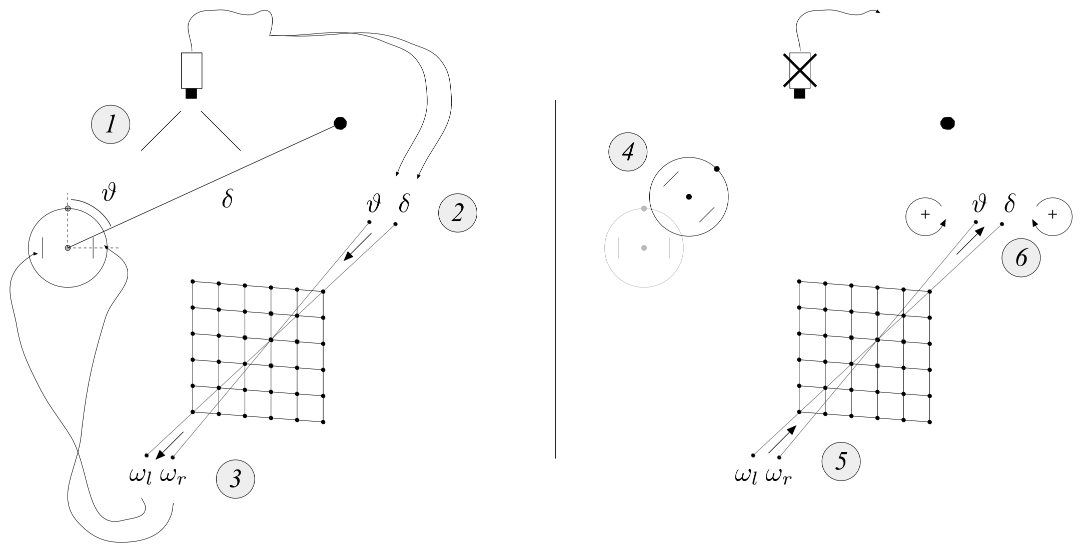


Figure 5: The way the trained network is used to guide the mobile robot to a target location.

weight vector is the robot displacement and has been represented as a vector: the vector orientation indicates the robot heading change, while the vector length is the distance travelled by the its axle mid-point. By looking at the overall aspect of the grid, one may appreciate the *data topology-conserving* character of the network. Velocity pairs are ordered along the grid horizontal and vertical axes, so similar velocity pairs are mapped onto neighboring locations on the grid. A continuous variation is also present in the displacement values. So similar velocity pairs will lead to a similar displacement. We argue that the EKM model is especially well-suited to learn the robot kinematics, thanks to the cooperative aspect the learning rule, which naturally tends to associate similar output values to similar inputs. This turns out to be a desirable generalization property, as the function to be learned is a continuous relationship between velocity and displacement.

4.3 Network performance

The EKM has been trained on the forward mode, namely on a transformation from the space of motor commands to the space of visual perceptions. We will now explain how the trained network is used in backward mode to compute the inverse function, which transforms a visual perception into a motor command. The task is to guide the robot to a target location placed at arbitrary angle ϑ and distance δ in the workspace (Figure 5). The angle and distance information is provided by the camera system, which observes both the robot and the target (step 1). To be more specific, ϑ is now defined as the angle between the robot heading direction and the vector connecting the robot axle mid-point and the target, while δ is the Euclidean distance between the robot axle mid-point and the target location.

The observed ϑ and δ values are supplied to the EKM in backward mode to retrieve a velocity pair (steps 2 – 3). For this particular application, the competition on the pattern (ϑ, δ) has been designed to consider its components *in sequence*. First, ϑ is processed: the competition is restricted to the weight vector component of the neuron which

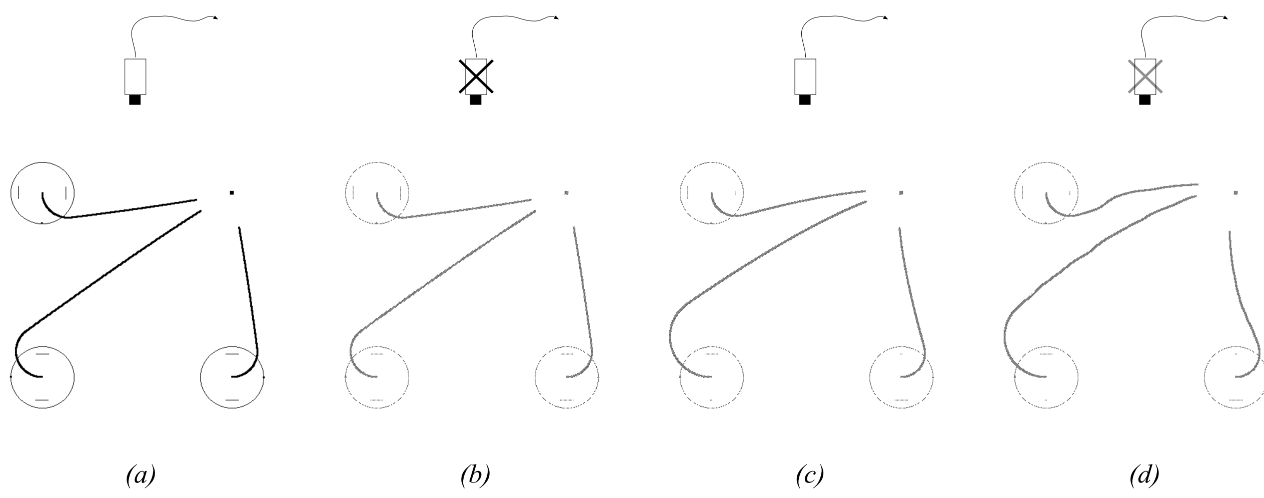


Figure 6: Trajectories performed by the robot with the camera information (a), without camera feedback (b), with left wheel radius reduced and visual information (c), with left wheel radius reduced and sporadic visual feedback (d).

stores the angle information. As the result of this preliminary step, a subset of grid neurons are selected: these are those neurons which match ϑ equally well. Second, δ is processed, but only those neurons selected at the previous step are allowed to compete; among these, the competition is restricted to the weight vector component of the neuron which stores the distance information. The overall competition process leads to the selection of a velocity pair for the robot wheels, namely the fan-in weight vector of the winning neuron. The rationale for splitting the competition in two steps is twofold. First, consider that, while the angle varies in a limited range of values, the distance value can be arbitrarily large. Therefore, if both data were processed by the network in one step, the distance component would have had more importance in the determination of the winning neuron. This can lead to undesirable effects, such as always selecting the neuron corresponding to the highest speed for both wheels[7]. Second, it seems natural to give priority to the angle information, so that the robot's first goal is to turn to face the target. The distance information becomes really relevant when the robot is close to the target so as to gradually reduce the wheel speed[8].

After having applied the speed pair to the wheels for a fixed time Δt, the robot arrives to a new position (step 4). Its new position with respect to the target is again derived by the camera system, and the whole procedure is repeated until the robot reaches the target. However, it is interesting to note that, as an alternative to employing the camera system to update the robot positional information, the network direct kinematics model can be used instead. This means that, after having applied a velocity pair to the robot wheels (step 4), the same pair is processed by the network in forward mode to retrieve a displacement pair (steps 5–6) and to update the new position of the robot with respect to the target. So "in principle" the robot is able to reach the target position blindly, i.e. without the visual feedback provided by the camera.

4.4 Results on network performance

The network performance has been tested both in a computer simulation and on the Khepera robot.

Figure 6 refers to the computer simulation. It depicts the trajectories performed by the robot to reach a target starting from several initial positions and under different experimental conditions. In Figure 6a the robot reaches the target by using the visual feedback from the camera system to update its positional information during motion. In other words, the EKM is just employed in backward mode to retrieve velocity pairs when the positional information is supplied by the camera system (steps 1 to 4 of Figure 5). Figure 6b shows the paths followed by the robot when the network kinematic model is used to update the robot position. In other words, the network is supplied with the target position at the beginning of the path only, then steps 2 to 6 are repeated until the target is reached. So the network is able to reach the target position blindly by using both forward and backward modes. No significant deviation from the previous trajectories is observed. Finally, Figures 6c–d show the trajectories of the robot after having reduced its left wheel radius to half of the original size. In this way, the robot kinematics dramatically changes. Nevertheless, when supplied with the camera visual feedback continuously (Figure 6c) or at least sporadically (Figure 6d), the robot is still able to reach the target, although the trajectories are now different. This last experiment suggests that the system exhibits a kind of "noise resistance". However, it should be made clear that the compensation ability to perturbations (in this case, the reduction of a wheel radius), is due to the fact that the robot is provided with the information of how it is moving with respect to the target. This information gives the robot a signal that something is going wrong. In other words, at each step of its trajectory, the robot can compare the actual position to the goal to the one calculated by the network kinematic model. Eventually, if the difference between the two is too large, the

[7]This is the neuron that machtes the longest distance, which corresponds to the speed pair $(5,5)$.

[8]For example, by selecting the speed sequence $[(5,5),(4,4),(3,3),(2,2),(1,1),(0,0)]$, which corresponds to one diagonal of the competitive grid.

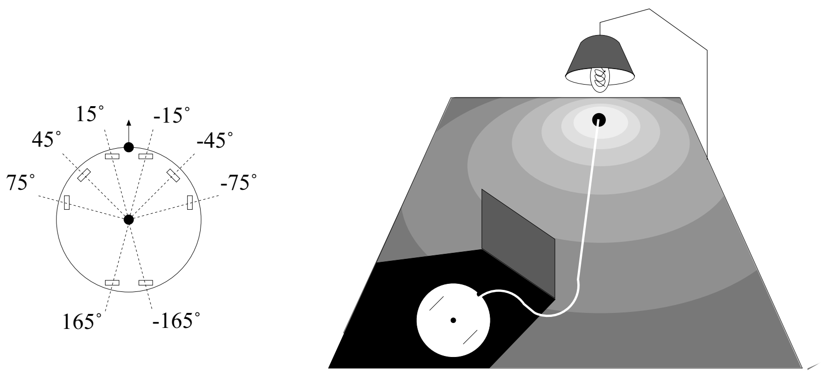


Figure 7: The Khepera infrared proximity sensors (left), the robot moving towards the light (right).

robot may decide that the model is no longer accurate enough and start a new training phase. We want to stress once again that this decision is induced by the camera system. If the robot is not given the visual information, at least sporadically, there is no way to compensate for noise. A real robot which moves blindly will get lost, sooner or later, as noise is always present and unpredictable.

We conclude by briefly describing the experiment with the Khepera robot. The experiment was organized by taking advantage of the robot's sensing capabilities. Khepera is provided with eight infrared proximity sensors (Figure 7, left). Each sensor contains a light receiver which allows the measurement of normal ambient light along a particular direction. We identified the target position with a light source (Figure 7, right). In this way, the angle between the robot forward direction and the target can be defined, for example, as the weighted mean of the angles corresponding to the two most activated sensors. This gives a rough approximation of the ϑ value. As far as the distance value δ is concerned, we kept it fixed to a chosen value[9]. With ϑ and δ defined in this way, Khepera is able to reach a fixed source of light or to track a moving light. The behavior of the robot clearly depends on the value to which δ is fixed. It the distance value is high[10], the network selects high velocities and the robot moves fast. If a small distance value is selected, smaller velocities are preferred. As δ is always set to a value other than 0, the robot never stops; rather, after having reached the area of maximal light, it keeps moving in that area. Finally, a nice effect we observed, is the robot's ability of avoiding obstacles while moving towards the light (Figure 7). As the obstacles project shadows, the robot, which always moves in the direction of the strongest light, takes the shadow as an indication of which areas are to be avoided to prevent collisions. In a certain sense, this experiment can be regarded as the physical implementation of a robot moving in a potential field generated by an attraction force towards the goal (the light) and by repulsion forces exerted by the obstacles (the shadows).

5 Conclusions

We have presented an experiment on the *visuomotor coordination* of a mobile robot. Our approach to sensorimotor modeling belongs to the category of *learning by doing*. First, $< perception, action >$ pairs are collected by observing the robot during operation. These data are the starting point to describe the real robot behavior. Second, a learning by examples method is used to estimate the parameters of the model. The advantage of the *a posteriori* approach to sensorimotor modeling lies in the fact that, in the case where the robot consistently deviates from the ideal behavior as predicted by its kinematics equation, this deviation will been included in the model.

In our experiment the learning machine is an *Extended Kohonen Map*. We have observed that this neural network model has the property of being *invertible*. Given an input pattern, the network retrieves the output value by competition on the fan-in weight vectors of the neurons. We have called this use of the input-output mapping the network *forward* mode. Viceversa, given an output pattern, a corresponding input data is retrieved by competition on the neuron fan-out weight vectors. We have called this use of the map *backward* mode.

We argue that the invertibility property makes the Extended Kohonen Map a suitable architecture for sensorimotor modelling.. The experiment of the visuomotor coordination of a mobile robot shows that by training such a network on the robot direct kinematics (the forward mode), one obtains at the same time a solution to the inverse kinematics problem (the backward mode). The trained network can be used to guide the robot to any target location in a 2D workspace. To update its position with respect to the target during motion, the robot may take advantage either of the visual feedback provided by a camera system, or of the network direct kinematics model. However, experimentation in the presence of noise has stressed the importance of providing the robot with at least some sporadic visual information about the target position. In this way, the system exhibits a kind of noise resistance. In our opinion, this issue becomes really relevant in path finding problems, where the robot is expected to keep track

[9]A sensible definition of distance is hard to give.

[10]Here "high" means high with respect to the distance range learned by the network.

of its position with respect to the goal autonomously. A real robot which moves blindly will get lost sooner or later. Rather, the definition and recognition of landmarks on the path could greatly help the robot task. This will be one of the subjects of our future research.

Acknowledgements

We would like to thank *Neuristique* (France) for having provided the SN neural network simulator to support the experimental part of the work. Thanks to Vicente Ruiz de Angulo, Thomas Barbas and Domenico Perrotta for their comments on early drafts of this paper.

References

[BDM94] Béssiere, P., Dedieu, E., Mazer, E. *Representing Robot/Environment Interactions Using Probabilities: the "Beam in the Bin" Experiment.* Proc. "From Perception to Action", Lausanne, Switzerland.

[BGG93] Bullock, D., Grossberg, S., Guenther, F. *A Self-Organizing Neural Network Model for Redundant Sensory-Motor Control, Motor Equivalence and Tool Use.* Journal of Cognitive Neuroscience, 5, 408-435.

[K84] Kohonen, T. *Self-Organization and Associative Memory.* Springer Series in Information Sciences, 8, Heidelberg.

[M94] Massone, L. L. E. *Sensorimotor Learning.* To appear in "The Handbook of Brain Theory and Neural Networks", M. A. Arbib Editor, MIT Press.

[MFI93] Mondada, F., Franzi, E., Ienne, P. *Mobile Robot Miniaturization: a Tool for Investigation for Control Algorithms.* Proc. of the Third International Symposium on Experimental Robotics, Kyoto, Japan.

[RMS91] Ritter, H., Martinetz, T, Schulten, K. *Neural Computation and Self-Organizing Maps. An Introduction.* Addison-Wesley.

[RS87] Ritter, H., Schulten, K. *Extending Kohonen's Self-Organizing Mapping Algorithm to Learn Ballistic Movements.* Neural Computers, R. Eckmiller and E. von der Marlsburg (eds.), Springer, Heidelberg.

[ZGLC94] Zalama, E., Gaudiano, P., López Coronado, J. L. *A Real-Time, Unsupervised Neural Network Model for the Control of a Mobile Robot in a Nonstationary Environment.* Boston University Technical Report CAS/CNS-94-002.