Unsupervised Complexity Reduction of Sensor Data for Robot Learning and Adaptation

T. Kyriacou, R. Iglesias, M. Rodríguez, P. Quintía

***Abstract*— In this paper an unsupervised method is presented for decreasing the amount of data a robot uses from its sensors without destroying the useful information that the robot needs to be successful in its learning task. This reduction is done by selecting a subset of all the sensors available to the robot but also by segmenting the measurement range of each sensor into intervals. It is shown here that this reduction in sensor data results in a significant reduction in the learning time of the robot.**

I. INTRODUCTION

Every year an increasing number of robots are entering the domestic environment. These robots are intended to help with household chores, act as home health aids, and serve as companions and entertainers for people. However most of these robots are still very limited, or look and behave like they belong in a factory. There is still an important barrier between the latest developments on research robots and the commercial ones. Partly to blame for this is the uncertainty inherent to any robot behaviour: what any mobile robot does is the result of the properties of the robot itself, the control program the robot is executing and the robot's environment [1]. This problem becomes worse for domestic and service robots due to an important lack of predictability (neither the user's behaviour nor the physical environment can be known before a robot is placed in a home). Therefore, we must accept that we will be buying imperfect robots and that at some point these robots will make mistakes when moving in our home. Therefore we need robots that are able to adapt their behaviour and learn from their experiences when emulating people or exploring their environment. As is the case with humans, the mistakes and successes a robot makes should influence its future behaviour rather than relying only on predefined rules, models or hard-wired controllers.

To achieve real and effective on-line robot adaptation we consider a three-stage process: First we must achieve fast learning procedures. Second, the robot needs to learn simultaneously how to perceive and how to act. Finally, the learning procedure must be transferred into the real robot. This paper concentrates on the second stage.

We need robots that are able to simultaneously learn how to perceive and how to act from their interaction with the environment. Robot learning and adaptation in realistic environments requires novel algorithms able to identify important events in the stream of sensor data. Nevertheless, this is getting more and more complex, due to the increasing number and resolution of sensors that today's robots use. Technology advances provide robots continuously increasing sensor modalities that make data collection easier and faster. This usually results in high dimensional and complex data sets in which many of the dimensions are often irrelevant to what the robot's learning task. These irrelevant dimensions confuse clustering techniques since when the number of dimensions increases, distance measures become increasingly meaningless [2]. Additional dimensions spread out the points until, in very high dimensions, they are almost equidistant from each other, this is known as the *curse of dimensionality*. Obviously, one solution to reduce the complexity of data is (for the user) to determine, heuristically, which subset of sensors are relevant for the task the robot is trying to solve, but this contradicts the idea of an autonomous robot that is changing its behaviour under the working circumstances. We need unsupervised techniques able to reduce the sensor data complexity.

In this paper mutual information and entropy measurements are used to determine not only the subset of sensors that are really meaningful for what the robot is doing, but also a suitable partition of every sensor's range of values into a finite set of intervals, thus reducing the complexity of the robot's sensor data.

II. EXPERIMENTAL SETUP

The aim if this paper is to propose and evaluate a method to determine the most relevant sensors and their partition in intervals. The experimental scenario is shown in figure 1. A Pioneer 3DX robot is used to learn the wall following behaviour in a simulated environment. The robot is equipped with a front-facing laser scanner and 16 ultrasonic sensors (figure 2). To evaluate the unsupervised sensor selection and the partitioning methods, the robot's actions and sensor perceptions were logged with a sampling period of 250 milliseconds while the robot was learning its task. The data collected was then used to determine the most relevant sensors and the partition of the sensors ranges into finite sets of intervals. This data complexity reduction was evaluated using subsequent attempts to learn the same task. In other words, during the evaluation phase, the robot was made to use less sensors (those deemed most relevant) and sensors whose ranges were discretised (again, into intervals that were deemed to be most relevant to the task). In this way we

T. Kyriacou is with the School of Computing and Mathematics, Keele University, Keele, Staffordshire ST5 5BG, UK. t.kyriacou@keele.ac.uk

R. Iglesias, M. Rodríguez and P. Quintía are with the Department of Electronics and Computer Science, University of Santiago de Compostela, Campus Vida, 15782 Santiago de Compostela, Spain. {roberto.iglesias.rodriguez; miguel.rodriguez.gonzalez; pablo.quintia}@usc.es

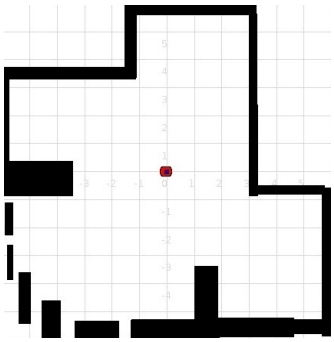

Fig. 1: Simulated environment where the Pioneer 3DX will learn how to follow a wall located on its right.

Fig. 2: P3DX robot, a simulated model of which was used in our experiments. This robot is equipped with a SICK laser scanner and 16 ultrasound sensors.

evaluated the performance of our method and whether the simplified sensor data still contained enough information to learn the task.

The set of data that was used to determine the most relevant sensors and the partition of every sensor's range readings into intervals was that logged during the early stages of the robot's learning, while its movement was still erratic and showed many mistakes. The reason why we decided to do this is because as part of our future research we plan to use our method in parallel with the learning process itself.

III. Determination of the most relevant sensors

To determine the most relevant sensor inputs we estimated the sensitivity of the robot actions, A, with respect to each input S_i, $\forall i$ by using the average mutual information I_m [3].

$$I_m(S_i,A) = \sum_{a_i,s_j} P_{S,A}(a_i,s_j) log_2[\frac{P_{S,A}(a_i,s_j)}{P_A(a_i).P_S(s_j)}] \qquad (1)$$

$P_{S,A}(a_i,s_j)$ is the joint probability density with which the robot performs action a_i when sensor S_i takes the value s_j. $P_S(s_j)$ and $P_A(a_i)$ are the individual probability densities for S_i and A resulting in values s_j and a_i, respectively.

Mutual information measures the information that S_i and A share; i.e. how much we can say about one of the two variables (robot action or sensor value) when we know the value of the other. If S_i and A are completely unrelated (i.e. independent), $P_{S_i,A}(s,a) = P_{S_i}(s).P_A(a)$ and, therefore, the mutual information is zero. In the other extreme, if the two variables (S_i, A in our case) are identical, all the information conveyed by one of them is shared with the other; the value of one of the variables can be used to determine the value of the other. In this case the mutual information is the uncertainty contained in any of the two variables, i.e. the entropy H of either of the two since they are identical.

IV. Determination of the most meaningful sensor intervals

Splitting sensor values in intervals instead of working with the full range of sensor values (i.e. the full resolution of each sensor's range) can be beneficial for many robot behaviours. Working with intervals might help to focus the robot's attention on those aspects that are relevant for the task at hand and thus cause it to avoid details that might confuse the robot or even alter its behaviour erroneously. Examples of such problems occur during a door traversal behaviour or during convoy formation behaviour. In the case of the door traversal task, all the robot needs to know is the location of the door at every instant so that it can move towards the door's centre. Details such as the objects encountered after the robot has traversed the door (detected with the front sensors) will make the learning more difficult and the robot's behaviour less robust. A very crude descretisation of the robot's range sensors using only two intervals ("1" showing close obstacles such as the door frame, and "0" representing distant obstacles) would almost suffice in order to learn the task. Something similar happens with the convoy formation. In this case each robot in the formation needs to follow closely the robot in front. The observation of close and frontal obstacles is much more relevant than distant features of the environment in which the convoy moves.

The descretisation of each sensor range in intervals needs to be done automatically and in an unsupervised way. We must avoid error-prone heuristic criteria dependent on the designer's considerations. On the other hand we must use an automatic way of determining a partition that is dependent on the sensor itself and the behaviour exhibited by the robot ("the task"). In this paper we suggest a combination of mutual information and evolutionary strategies [4] to find this partition: we plan to find the partition to those intervals, for every sensor, that maximizes the mutual information between the sensor and the action. Therefore, the partition will be dependant on both, the sensor and the robot behaviour:

- *Computation of the mutual information using intervals*: If we divide the range of possible sensor values S_i, $\forall i$ in a set of P intervals:

$$
\begin{aligned}
&S_i = ... \\
&... = \{S_i(1) = [0, s_1), S_i(2) = [s_1, s_2), ..., \\
&..., S_i(p) = [s_{p-1}, s_p = max\_possible\_value\_of\_S_i)\}
\end{aligned} \tag{2}
$$

we can compute the mutual information between the action A, and S_i when this partition is used:

$$
I_m(S_i, A, P) = \sum_{a_i, l} P_{S,A}(a_i, S_i(l)) log_2 [\frac{P_{S,A}(a_i, S_i(l))}{P_A(a_i).P_S(S_i(l))}] \tag{3}
$$

$P_{S,A}(a_i, S_i(l))$ is the probability that the value of S_i falls into the l-interval and the action A takes the value a_i. $P_S(S_i(l))$ is the probability that the value of S_i falls into l-interval.

- *Search for the best partition*: Given a partition of S_i in P-intervals, we can shift the boundaries of the intervals trying to maximize the mutual information between S_i and A. To do this we have chosen to use evolutionary strategies (ES), since they are suitable and common for continuous parameter optimisation [4]. An evolutionary strategy is an optimization technique based on ideas of adaptation and evolution: given a population of individuals within some environment that has limited resources, competition for those resources causes natural selection (survival of the fittest). Therefore, evolutionary strategies (ES) need a fitness function that will be maximized. In fact, on the basis of this fitness function some of the best candidates are chosen to seed the next generation. In this paper we have applied one of the simplest ES algorithms, (1+1)-ES. This algorithm will find the (P-1) upper boundaries of the intervals into which we shall split S_i values ($s_1, s_2, ..., s_{p-1}$ in equation 2). These boundaries will be found by trying to maximize the mutual information between the robot actions and the sensor readings. To do this, the (1+1)-ES will operate on a population of size two: the current point (parent), and the result of its mutation (offspring). Only if the offspring's fitness is at least as good as the parent's one, the offspring becomes the parent for the next generation. Next we present a brief description of the basic algorithm:

BEGIN

1) Set $t = 0$
2) Create an initial point with the boundaries of the intervals $x^t = (s_1, s_2, ..., s_{p-1}) \in \mathbb{R}^{p-1}$
3) REPEAT UNTIL (TERMINATION CONDITION IS SATISFIED) DO
   a) draw a new random point $z^t \in \mathbb{R}^{p-1}$ from a normal distribution $N(0, \sigma)$
   b) create a new candidate (offspring) by adding the random vector z to the initial *parent* point $y^t = x^t + z^t$
   c) Survivor selection: compare the *fitness* of the parent and the new candidate.
      If $Im(A, S_{i,x}) > Im(A, S_{i,y})$ then
        $x^{t+1} = x^t$
      else
        $x^{t+1} = y^t$
   d) $t = t + 1$

END

$Im(A, S_{i,x})$ and $Im(A, S_{i,y})$ are the mutual information amongst A and S_i, when the interval boundaries are the ones in vector x, and in vector y,respectively. A Gaussian, or normal, distribution with zero mean and standard deviation σ is used for drawing random vectors and thus generate new candidates. Thus, the σ value is a parameter that determines the extent to which the parent values (x) are perturbed. For this reason σ is often called *mutation step size*. Usually σ's value is altered on-line using the *1/5 success rule*:

$$
\sigma = \begin{cases} \sigma/c & \text{if } p_s > 1/5 \\ \sigma.c & \text{if } p_s < 1/5 \\ \sigma & \text{if } p_s = 1/5 \end{cases}
$$

where p_s is the relative frequency of succesful mutations measured over a number of trials. The parameter c is a constant value usually within the interval $[0.8, 1]$. This *1/5 rule* is applied at periodic intervals, for instance, after k iterations. The more the success of mutation the higher the value of σ in an attempt of making a wider search for the solution in $\mathbb{R}^{p-1}$. On the contrary if the success ratio is less that $1/5$ then σ should be decreased to concentrate the search around the current solution.

- *Search for the best number of intervals*: There is a problem that still needs to be solved: according to the previous two items we know how to search for the best partition of S_i in P-intervals. This partition will be the one that maximizes the mutual information amongst S_i and the robot actions, A. Nevertheless, we still don't know how many intervals we should use. Because of this, we still need to apply a straightforward interative process that, once again, tries to maximize the mutual information:

1) P=1
2) DO
   a) Apply (1,1)-ES to get the best partion of S_i in P-intervals. $I_m(A, S_i, P)$ is the mutual information achieved when P-intevals are used. This mutual information is calculated using the method described in the previous item.
   b) If (P=1) then
      $max\_I_m = I_m(A, S_i, P)$

$best\_number\_intervals = 1$

Else If $max\_I_m < I_m(A, S_i, P)$

$max\_I_m = I_m(A, S_i, P)$

$best\_number\_intervals = P$

$P = P + 1$

UNTIL (P =MAXIMUM NUMBER OF PARTITIONS)

3) END THE SEARCH PROCESS, *best_number_intervals* IS THE BEST PARTITION FOR S_i.

V. Simultaneous learning of perception and action

We now describe the strategy we used to learn a mapping "sensor-state-action" through trial and error robot-environment interactions. In order for a robot to learn from its own experiences, it needs to be able to build a representation of the environment that increases dynamically to include new situations that have not been seen before. We shall call to these new and distinguishable situations, detected in the stream of sensor inputs, *states*. A dynamic creation of the state representation will avoid the classic, error-prone and cyclic process of designing and testing ad hoc representations. On the other hand, the dynamic representation of the environment will be combined with a learning strategy able to adapt the robot's behaviour to those relevant situations (states) that are being identified. Therefore, our system will learn simultaneously the state space and the action to execute on each state.

Despite the fact that we will describe the learning procedure so that the contents of this paper are clear, we also want to highlight that this learning procedure plays a secondary role in this paper. We want to determine whether the complexity reduction of the sensor data helps the robot to achieve learning faster than when it is using all sensors available to it.

A. Dynamic representation of the environment

If we really want a robot able to adapt to its workspace without human help, we must avoid representations of the environment that are created off-line and, instead, use strategies that enable the identification of important events in the stream of sensor inputs in a largely unsupervised way. We used a Fuzzy ART artificial neural network to obtain this dynamic creation of the set of states [5] .

The input of the Fuzzy ART network will be an M-dimensional vector containing sensor readings. Each one of this components will be translated into the interval [0,1]. To prevent the Fuzzy ART from creating too many states, we will normalize the inputs using *complement coding*. The complement coded input I to the recognition system is a 2M-dimensional vector:

$$\mathbf{I} = (\mathbf{a}, \mathbf{a}^c) \equiv (a_1, ..., a_M, a_1^c, ..., a_M^c), \tag{4}$$

where $a_n^c = 1 - a_n$. Using complement coding, the norm of the input vector will always equal the dimension of the original vector.

The prototype that identifies each state learnt by the Fuzzy ART network, will be codified as an array of 2M dimensions with values in $[0,1]$: $W_j = (w_{j1}, ..., w_{jM}, w_{j1}^c, ..., w_{jM}^c)$, where the sub index j refers to the state. The behaviour of the Fuzzy ART is determined by two parameters: learning rate $\beta \in [0,1]$; and a vigilance parameter $\rho \in [0,1]$. The way the Fuzzy ART network operates can be summarized in the following steps (there are some important differences in comparison with the general proposal described in [5]):

1) After presenting an input $\mathbf{I}$ to the network, there will be a competitive process after which the categories will be sorted from the lowest activation to the highest. For each input $\mathbf{I}$ and each state (often called category) j, the activation function T_j is defined as

$$T_j(\mathbf{I}) = \frac{|\mathbf{I} \wedge w_j|}{|w_j|} \tag{5}$$

the fuzzy operator AND $\wedge$ is $(x \wedge y)_i \equiv min(x_i, y_i)$ and the norm $|\cdot|$ is defined as

$$|x| \equiv \sum_{i=1}^{M} |x_i|. \tag{6}$$

2) The state with the maximum activation value will be selected to see if it resonates with the input pattern $\mathbf{I}$

$$J = arg\_max_j\{T_j : j = 1...N\}. \tag{7}$$

3) The Fuzzy ART network will enter in resonance if the matching between the input $\mathbf{I}$ and the winning state $\mathbf{J}$ is greater or equal than the vigilance parameter ρ:

$$|\mathbf{I} \wedge w_J| \geq \rho |\mathbf{I}| \tag{8}$$

If this relation is not satisfied, a new state will be created and the new prototype vector will be equal to the input $\mathbf{I}$.

4) When the network enters in resonance with one input, the prototype vector w_J is updated:

$$w_J^{(new)} = \beta \mathbf{I} + (1 - \beta) w_J^{(old)}. \tag{9}$$

B. Increasing Time before failure

According to psychology theories, learning is strengthened if it is followed by positive reinforcement – pleasure – and weakened if it is followed by punishment – pain [6]. Inspired by these psychological theories, Sutton and Barto developed reinforcement learning as a machine learning paradigm that determines how an agent ought to take actions in an environment so as to maximise some notion of long-term reward [7]. What makes this learning paradigm appealing is that the system learns on its own, through trial and error, relying only on its own experiences and a feedback – reward signal

– that encourages or discourages the execution of different sequences of actions.

We have developed a new reinforcement-learning based algorithm that is almost parameterless, and which allows an easy interpretation of what the robot is learning [8], [9]. Our proposal will provide a prediction of how long the robot will be able to move before it makes a mistake and hence, it receives negative reinforcements. This will make it easier to assess the evolution of the learning process as a high discrepancy between the time before failure predicted and what is actually observed on the real robot is a clear sign of an erroneous learning.

During the learning process, our robot will learn a utility function of states and actions, termed Q-function, $Q(S \times A)$. This function estimates the expected time interval before a robot failure when the robot starts moving in s, performs action a, and follows an optimal policy thereafter:

$$Q(s,a) = E[-e^{(-Tbf(s_0=s,a_0=a)/50T)}], \tag{10}$$

where $Tbf(s,a)$ represents the expected time interval (in seconds) before the robot does something wrong, when it performs action a in s, and then follows the best possible control policy. T is the control period of the robot (expressed in seconds). The term $-e^{-Tbf/50T}$ in Eq. 10 is a continuous function that takes values in the interval $[-1,0]$, and varies smoothly as the expected time before failure increases. If we are able to predict the consequences of performing a particular action in a state (time that will elapse before the robot makes a mistake), it is straightforward that we can achieve the best control policy by simply selecting the action with the highest Q-value for every state. This will give us the control policy that maximizes the time interval before any robot failure; this is called greedy policy π^*:

$$\pi^*(s) = arg\_max_a\{Q(s,a)\} \tag{11}$$

Since $Q(s,a)$ and $Tbf(s,a)$ are not known, we can only refer to their current estimations $Q_t(s,a)$ and $Tbf_t(s,a)$:

$$Tbf_t(s,a) = -50 * T * Ln(-Q_t(s,a)), \tag{12}$$

The definition of $Q(s,a)$, Tbf, and the greedy policy, determine the relationship between the Q-values corresponding to consecutive states:

$$Q_{t+1}(s_t,a) = \begin{cases} -e^{-1/50} & \text{if } r_t < 0 \\ Q_t(s_t,a_t) + \delta & \text{otherwise} \end{cases} \tag{13}$$

where,

$$\delta = \beta_L(e^{\frac{-1}{50}} * Q_{max}(s_{t+1}) - Q(s_t,a_t)). \tag{14}$$

r_t is the reinforcement the robot receives when it executes action a_t in state s_t, $\beta_L \in [0,1]$ is a learning rate, and it is the only parameter whose value has to be set by the user.

To obtain the utility values $Q(s,a)$, the robot begins with an initial set of random values, $Q(s,a) \in [-1,-0.95]$, $\forall s,a$, and then it initiates a stochastic exploration of its environment. The robot will move and collect data during a maximum period of time or until it makes an error. The data collected will later be used to update the Q-values:

First stage, collecting data

1) $m = 0$
2) At each instant t and while the robot does not receive a negative reinforcement or moves for a maximum period of time do:
   a) Observe the current state, s(t): $s[m] = s_t$.
   b) Select an action a_t for s_t: $a[m] = a_t$.
   c) Perform action a_t, observe new state s_{t+1} and reinforcement value r_t, $r[m] = r_t$,
   d) $m \leftarrow m+1$.

Second stage, updating the Q-values

1) for $k = m-1, m-2, \ldots, 0$ do:
   a) Update *time before failure*:
      if $k = m-1$ then:
      $$Tbf = \begin{cases} T & \text{if } r[m-1] < 0 \\ Tbf(s[m-1]) & \text{otherwise} \end{cases}$$
      else $Tbf \leftarrow Tbf + T$.
   b) Update the Q-values:
      - $\Delta Q_t(s[k],a[k]) =$
        $\ldots = \beta_L(-e^{-Tbf/50T} - Q_t(s[k],a[k]))$

To speed up the learning procedure, we have used an ensemble of learning individuals working in parallel. Since the $Q-values$ predict the time interval before a robot failure, we call "predictor" to each learning individual. The main idea is that each predictor will have to learn the best action interval for every state. Therefore the best action to be executed at every state will be inferred from the joint participation of all the predictors. Basically each predictor will "vote" for a set of actions, so that the action with the maximum number of votes will be the one the robot finally executes. Another nice aspect of our proposal is that it allows the parallel learning of the same behaviour from different robots simultaneously operating in the same or different environments. In all the experiments we present in this paper there were two robots moving at the same time. Further details can be found in [8], [9], and there is a journal article in preparation.

The reason why we used this learning strategy instead of other classic reinforcement learning algorithms, is because it really shortens the amount of time that the robot requires to learn a behaviour. Table I shows the performance of the learning algorithm just described (I_Tbf) in comparison with a classic reinforcement learning algorithm that that has been widely used in the past (Naive $Q(\lambda)$ [7]). Clearly I_Tbf outperforms Naive $Q(\lambda)$.

VI. Experimental results

We applied mutual information and evolutionary strategies to search the most relevant sensors and their partitioning intervals for a particular behaviour ("wall following"). The robot we used was a simulated version of a Pioneer P3DX. This robot is equipped with a SICK laser scanner and 16

TABLE I: Time required to learn the wall following behaviour when two different algorithms were used. Average learning time is in minutes. Each one of these learning times have been obtained after 15 experiments

Learning algorithm	**Average learning time**	**learning parameters**
Parallel I_Tbf	13.85 ± 15.26	$\rho = 0.91005$, $\beta = 0.0015$ $\beta_L = 0.288282$
Naive Q(λ)	23.12 ± 15.33	$\rho = 0.91005$, $\beta = 0.0015$ $\beta_L = 0.288282$ $trace\_decay\_\lambda = 0.869965$ $decay\_factor\_\gamma = 0.9$

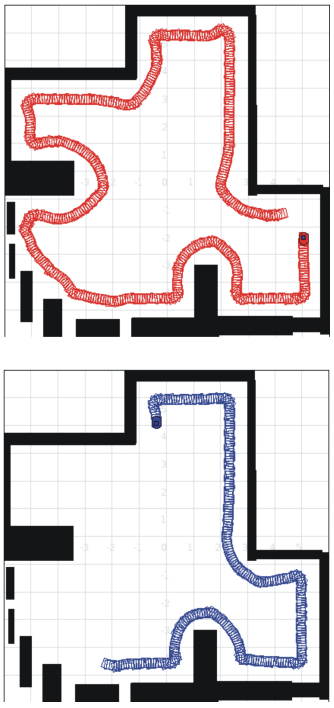

Fig. 3: Two robots learning the wall following behaviour using the algorithm described in section V. This graph has been obtained at the last stages of the learning process

ultrasonic sensors. Because of this, we log 193 data every 250 milliseconds, 177 of these inputs are laser readings, while the 16 remaining inputs are ultrasound sensors. These data were logged while the robot was learning the wall following task in a simulated environment (figure 3), and using Player/Stage software [10].

A. Determination of the relevant sensors

We calculated the mutual information amongst every sensor and the actions performed by the robot while it was learning the task. The results achieved are shown in figures 4 and 5. According to figure 4 the most relevant laser inputs are those corresponding to the front and right side of the robot. This agrees with what we would expect from a right-hand wall follower, but we still need to prove if this is really enough to learn the task. Regarding the ultrasound sensors,

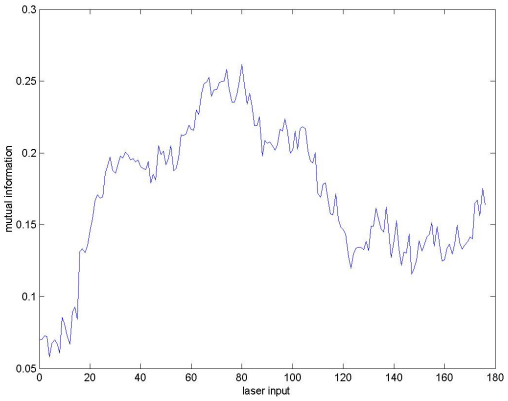

Fig. 4: Mutual information amongst each laser beam and the robot actions. This mutual information was calculated using data logged at the initial stage of the learning process.

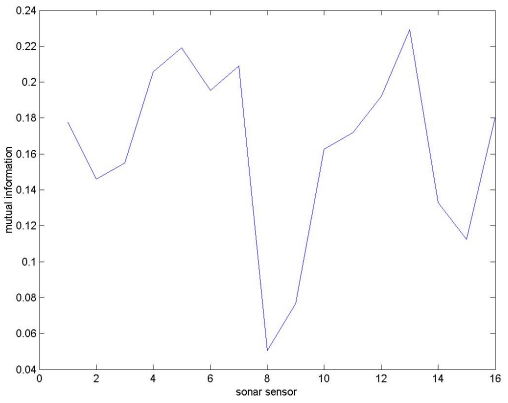

Fig. 5: Mutual information amongst each sonar and the robot actions. This mutual information was calculated using data logged at the initial stage of the learning process.

the most relevant ones appear to be the ones located not only at the front, but also at the right and rear sides of the robot.

To determine whether this information is meaningful we decided to learn the behaviour using only the subset of sensors that seem to be the most relevant. To determine these relevant sensors we first added up all the mutual information values:

$$Total\_I_m = \sum_{i=0}^{i=192} I_m(S_i, A)$$

We then sorted out the sensors by mutual information, from large to small values and, finally, we selected one sensor at a time until the sum of their mutual information reached 50% of the $Total\_I_m$.

Using this criterion, the relevant sensors are those 76 sensors (laser and sonar) shown in figures 6 and 7. When only these sensors were used to learn the task, we got an important reduction in the learning time (table II)), in particular the robot only needed 65% of the time that was necessary to learn the same behaviour using all sensors.

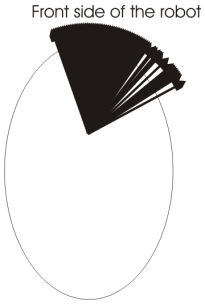

Fig. 6: Location of the most relevant laser beams.

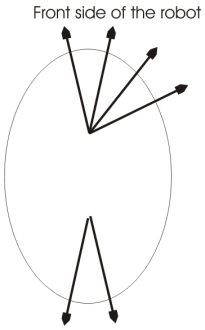

Fig. 7: Location of the most relevant sonar sensors

B. Determination of meaningful intervals

We now used the same data collected for the determination of the relevant sensors, but this time to find the best partition of every sensor readings into a finite set of intervals. The combination of 1+1 ES and mutual information, described in section IV, allowed us to explore the partition of each sensor readings into a number of intervals that ranged from 1 to 10. In most cases the mutual information obtained for the partition of the sensor values in 1, 2, ...,10 intervals approaches a logarithm curve (figure 8). The mutual information increases very fast with the first intervals, but this growth slows down when a higher number of intervals is used.

For every sensor we took the partition which gave raise to the highest mutual information (8, 9 or 10 intervals for most of the laser sensors (figure 9), while in the case of the sonar this partition was in 10 intervals in all cases but one (figure 10)).

Once again our robot learnt the right wall following behaviour using these partitions of the sensor readings. We considered two situations: a) learning of the task using all sensors and their partitions, b) learning of the task using only the most relevant sensors and their partition intervals. In both cases the ART network didn't receive the sensor readings at each instant, but the centroid of the intervals that contained those sensor values. We achieved very important and representative reductions in the learning time in both cases (table II). The use of intervals reduces very significantly the amount of time the robot requires to learn the task, and this reduction is even higher when only the relevant sensors and their partition is involed in the learning procedure.

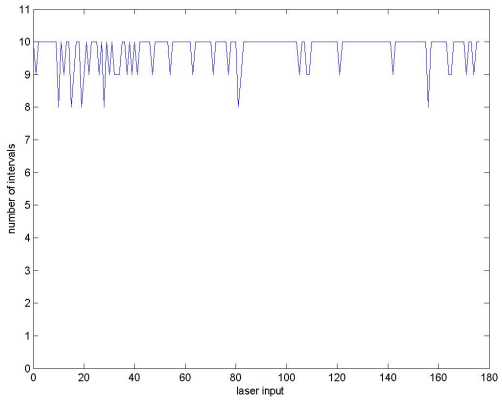

Fig. 9: Number of intervals into which the values of each laser beam are projected.

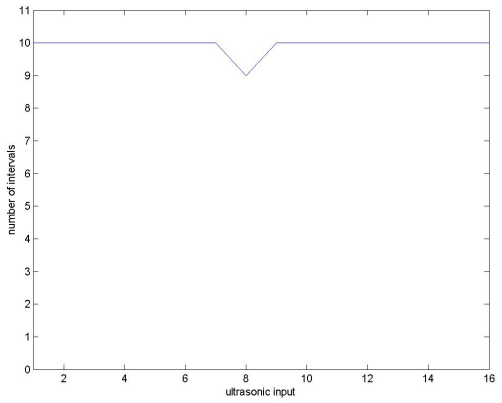

Fig. 10: Number of intervals into which the values of each sonar are projected.

VII. Conclusion

Successful applications of mobile robots have so far been restricted to well defined, fairly narrow application scenarios in which boundary conditions are known a priori. Nevertheless, this opposes the idea of robots operating in our everyday workplaces or our homes. Robots need to be able to adapt and learn on their during their interaction with their environment. This adaptation does not affect only the robot's actions but also how they should perceive the environment: data acquisition and processing will be dependent on the current goal or the task the robot is trying to achieve.

The constant development of new and more accurate sensors opens new challenges and possibilities in robotics. Robots will now be able to operate in more complex scenarios. In turn, this produces an enormous increase in data dimensionality which makes necessary the development of unsupervised methods that are able to reduce the complexity of sensor data. We have addressed this problem by developing a method able to search the subset of sensors and distance intervals which seem to be directly connected with what the robot is trying to achieve (i.e. the robot's motor actions). This "connection" is determined by means of the

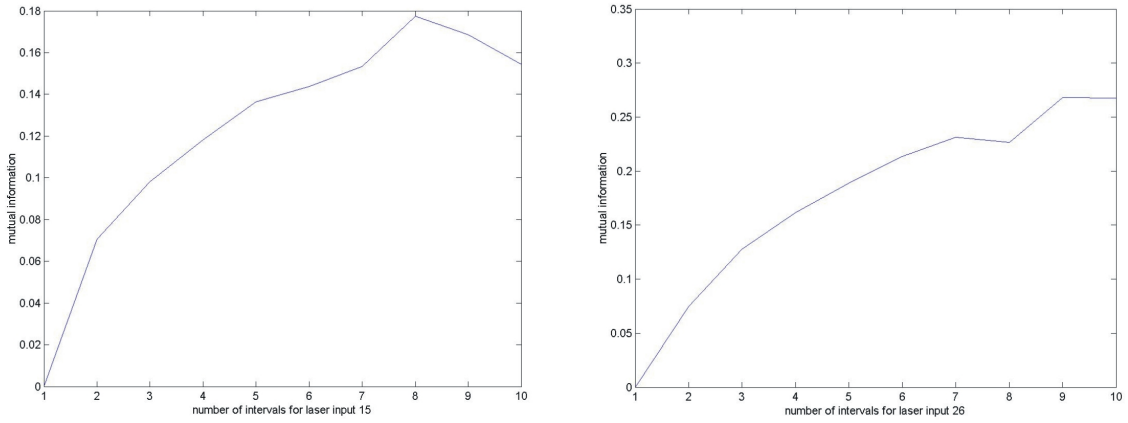

Fig. 8: Mutual information amongst two different laser beams and robot actions. These graphs show the growth of the mutual information as the number of intervals increases. This growth is faster at the beginning thus approaching a logaritmic curve. For the rest of the laser beams and sonar sensors the variance of the mutual information with the robot actions, as the number of intervals increases, is very similar to these two cases.

TABLE II: Time required to learn the wall following behaviour when different levels of sensor data complexity reduction were used. All the learning experiments where run using the parallel I_Tbf described in section V. In all experiments $\beta_L = 0.288282$. The average learning time is in minutes. Each one of these learning times have been obtained after 15 experiments

Learning Characteristics	**Average learning time**	**Fuzzy ART parameters**
all sensors, no intervals	13.85 ± 15.26	$\rho = 0.91005$, $\beta = 0.0015$
relevant sensors, no intervals	8.97 ± 5.51	$\rho = 0.91005$, $\beta = 0.001$
all sensors and intervals	7.66 ± 4.36	$\rho = 0.91005$, $\beta = 0.0015$
relevant sensors and intervals	7.09 ± 3.54	$\rho = 0.926$ $\beta = 0.0027$

mutual information measure. Our method was applied on a data set collected when the robot was starting to learn a wall following behaviour, the robot was learning from its mistakes and successes when interacting with the environment. As part of our future work we plan to use this method in order to combine the reduction of sensor data complexity with the learning process itself. We shall apply this on-line complexity reduction to the learning of wider robot tasks.

VIII. ACKNOWLEDGMENTS

This work has been funded by the research grants TIN2009-07737, INCITE08PXIB262202PR, and TIN2008-04008/TSI.

REFERENCES

[1] U. Nehmzow, R. Iglesias, T. Kyriacou, S. Billings, Robot learning through task identification, International Journal on Robotics and Autonomous Systems 54 (2006) 766–778.

[2] L. Parsons, E. Haque, H. Liu, Subspace clustering for high dimensional data: a review, SIGKDD Explor. Newsl. 6 (2004) 90–105.

[3] H. D. I. Abarbanel, Analysis of Observed Chaotic Data, Springer, 1996.

[4] A. E. Eiben, J. E. Smith, Introduction to Evolutionary Computing, Springer, 1998.

[5] G. A. Carpenter, S. Grossberg, D. B. Rosen, Fuzzy ART: Fast stable learning and categorization of analog patterns by an adaptive resonance system, Neural Netw. 4 (1991) 759–771.

[6] M. A. Bozarth, Pleasure: The politics and the reality, Springer Netherlands, 1994, pp. 5–14.

[7] R. S. Sutton, A. G. Barto, Reinforcement learning: An introduction, MIT Press, 1998.

[8] P. Quintia, R. Iglesias, M. Rodriguez,C. V. Regueiro, Fast Robot Learning through an Ensemble of Predictors Able to Forecast the Time Interval Before Failure, in: In Proceedings of Towards Autonomous Robotic Systems, TAROS 2009.

[9] M. Rodriguez, R. Iglesias, P. Quintia,C. V. Regueiro, Parallel robot learning through an ensemble of predictors able to forecast the time interval before a robot failure, in: In Proceedings of XI Workshop of Physical Agents, WAF 2010.

[10] B. P. Gerkey, R. T. Vaughan, A. Howard, The Player/Stage project: Tools for multi-robot and distributed sensor systems, in: In Proceedings of the 11th International Conference on Advanced Robotics, 2003, pp. 317–323.