

Memory-Based Navigation

Bruno Crespi, Cesare Furlanello and Luigi Stringa

Istituto per la Ricerca Scientifica e Tecnologica

I-38050 Povo, Trento, ITALY

Phone 39 461 814444; Fax 39 461 810851;

e-mail crespi@irst.it furlan@irst.it stringa@irst.it

Abstract

A memory-based system for mobile robot navigation is presented. The system is implemented as a follow-midline module on a small robot moving autonomously along the corridors of our Institute. The robot learns to estimate its position by comparing the visual input with images contained in its memory. The system memory is formed during a learning phase by encoding a set of labeled images. Encoding is based on the extraction of local directional fields.

During learning, the robot explores limited areas of the environment. During performance the robot moves along the corridors. The system shows a good generalization capability. The robot is able to navigate not only where learning was done, but also in unknown corridors with different illumination conditions. Key aspects of this approach are a flexible representation of information and independence from an explicit use of geometrical representations.

1 Navigation and Memory-Based Learning

Learning to navigate in a structured environment is a relevant test for an intelligent system. A robot moving inside a building has to cope with novel situations, such as changes in illumination, a person moving close to it, or a displacement of objects set along the walls.

Given these complex real-world conditions, it is important to choose an approach based on a flexible representation of information: the system should have the capability of learning from experience, performing in real-time and adapting to a changing environment. This means that information processing should not be based on specific geometrical characteristics of the problem domain, such as, for example, the vanishing point of a corridor.

Real-time performance is a necessary requirement for a robot moving in real-world conditions. At the same time, autonomy must be achieved with reasonable computational requirements and low power consumption. While these constraints are usually considered only as limita-

tions, in this context they can provide a useful indication about the validity of the chosen paradigm.

The purpose of this work is to explore a memory-based approach to information processing. Memory-based processing is a learning schema based on a simple and general principle. The basic operation is the comparison between present sensorial inputs and previously memorized patterns.

Memory-based reasoning, in the form of similarity-based induction, had been advocated as an exciting paradigm for a foundation of machine learning [Stanfill and Waltz, 1986]. Several years ago it seemed that feasibility would depend especially on the development of new parallel architectures. Thanks to technological advancement, standard CPUs and 'big' memories allow the potentials of memory-based processing to be tested even with a compact experimental setup.

Memory-based visual navigation

This paper presents a memory-based visual navigation system developed at IRST. A module implementing a follow-midline behavior was experimented on a small robot that moves autonomously along the corridors of the Institute.

The system is based on the comparison of the visual input with encoded labeled images of the surrounding environment. Inputs are 128×128 gray-level images that are encoded by a preprocessing module that extracts local directional fields. The spatial position of the robot is represented by classes that discretize the lateral displacement and the attitude (angle) of the robot with respect to the centerline of the corridor.

The system memory is formed by associating encoded images with the corresponding spatial class. During learning the robot explores a part of the environment and memorizes images taken at a known attitude and lateral displacement. During performance, the robot has to move along the corridor near the centerline. To perform this task, the robot modifies its driving and steering velocity in accordance with the class estimated from the current image.

The research work has been developed within the framework of MAIA, the general AI project under development at IRST ([Stringa, 1992; Poggio and Stringa, 1992]).

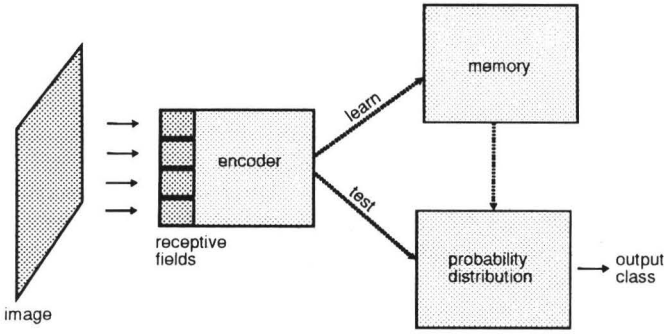


Figure 1: Structure of the system.

More on memory-based learning

Learning schemata related to memory-based processing seem to constitute a successful strategy for coping with a complex and changing environment.

The basic mechanism is simple and general. The system memory is organized using a set of *templates* of the input pattern space. The output for a new sensorial input is computed on the basis of its similarity with the templates previously memorized.

From a general point of view, the solution of many A.I. problems can be regarded either as the construction of a mapping or the validation of a regression model from data. In the *algorithmic* approach, the mapping is a rigid procedure, which is devised to extract from the input the information relevant for a specific problem. Algorithmic procedures can have high performance when the task to be performed and the environmental setting are well defined. However, as they are very specific, every task and environmental condition requires a different algorithm. The more a system is dedicated to a particular solution, the more bias we have to introduce in the model, reducing flexibility and robustness [Geman *et al.*, 1992].

In *memory-based* processing the mapping is synthesized by a general procedure that depends on the content of the system memory. We impose as little bias as possible in pre-processing. By reducing the amount of specialized knowledge in the pre-processing phase, the information stored in memory can be used by different centers of elaboration to perform different tasks with one basic procedure.

Main characteristics of the memory-based approach are *flexibility*, *speed* and *robustness*. Necessarily, such a general approach is sensitive to the characteristics of the training data, which have to be as representative and informative as possible. On the other hand, whenever storage space and the number of comparisons can be limited (for example by the use of vector quantization techniques), information processing based on matching can be very efficient.

2 System Architecture

The navigation module has been developed from the general architecture of memory-based learning systems presented in Ref. [Crespi *et al.*, 1992; Stringa, 1990]. Inputs

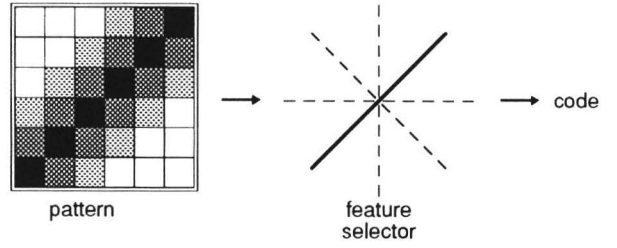


Figure 2: Extracting the local dominant direction from a simple pattern (gray-level)

are gray-level images that are encoded by the receptive fields of the system. The system output is a probability distribution that determines the classification of the input image according to a Bayesian decision rule. The global structure of the system is sketched in Fig. 1.

The input image, described by a matrix $G(x, y)$ of $N_X \times N_Y$ pixels, is covered by N_s subsets S_α of fixed size $K_X \times K_Y$. These subsets may or may not be overlapping and can be considered the receptive fields of the system. The pixels of a subset form a pattern that is mapped into a *code*. In this representation, an image $G = \bigcup_\alpha S_\alpha$ (where $1 \leq \alpha \leq N_s$) is described by a vector of codes $(code(S_\alpha))_\alpha$.

We may think of the encoding procedure as a mapping. The visual information contained in the image is decomposed into elementary patterns and then mapped into the *code space*. The encoding mapping is chosen to quantize and compress information while emphasizing the *dominant directions* contained in the S_α . The hypothesis here is that navigation is such a basic behavior that we do need to take into account large or complex patterns. Relevant information for real time navigation can be obtained by extracting local directional fields from simple patterns as shown in Fig. 2.

During learning a large number of images are associated into several classes. These associations are memorized by constructing the frequency matrices $\phi_\alpha(code_p | class_k)$ which represent the number of times pattern $code_p$ has appeared in images belonging to a certain $class_k$ in subset S_α .

The image G is divided into larger subregions called *windows*. For example, in Fig. 3, the image is divided into four windows, W_1 , W_2 , W_3 , and W_4 . Subsets belonging to the same window are grouped together and the corresponding ϕ_α are summed

$$\Phi_w(code_p | class_k) = \sum_{S_\alpha \in W_w} \phi_\alpha(code_p | class_k) \quad (1)$$

where $w = 1, 2, \dots, N_{windows}$, $k = 1, 2, \dots, N_{classes}$, and p ranges over all the possible patterns. The frequency matrices Φ_w form the memory of the system.

During performance an unlabeled image is classified

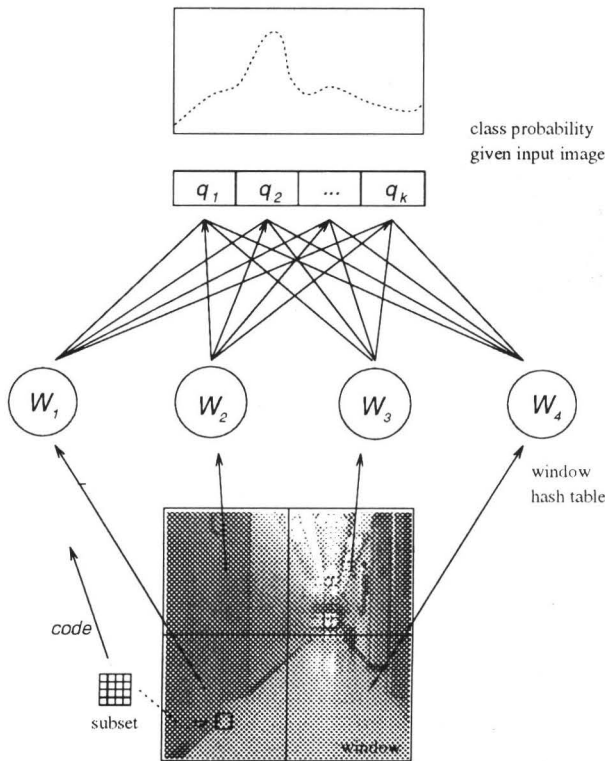


Figure 3: The coding procedure. The input image is covered by subsets of fixed size and a *code* is generated from each subset. Subsets belonging to the same window are grouped together.

by computing the Bayesian probabilities

$$P_w(class_k | code_p) = \frac{\Phi_w(code_p | class_k) \cdot P(class_k)}{\sum_j \Phi_w(code_p | class_j) \cdot P(class_j)} \quad (2)$$

for every different pattern that appears in the image. The *a priori* class probabilities $P(class_i)$ are assumed to be equal. If we suppose, for simplicity, that patterns are mutually independent, the image G is assigned to the class that maximizes the quantity

$$\wp(class_k | G) = \prod_w \prod_{p \in W_w} P_w(class_k | code_p) \quad (3)$$

where $k = 1, 2, \dots, N_{classes}$ and p ranges over all the patterns that appear in window W_w of the test image.

The probability distribution $\wp(class | image)$ is computed from the elementary distributions $P_w(class | code)$. It is not known *a priori* how these probabilities are correlated. The simplest choice, that of considering individual patterns as mutually independent, turns out to be good enough for our purposes.

The encoding is based on the division of visual information between pattern space (type of pattern) and position space (position of the pattern in the image): the pattern is the elementary concept, or content, at position (x_s, y_s) . We reduce the large amount of information contained in images giving more consideration to content than to locality. This means grouping together subsets

belonging to larger subregions (windows) of the image. A total independence of locality would be obtained by summing over all the locations of the subsets, i.e. by setting to one the number of windows.

For navigation along corridors, the relevant information is contained in the slopes of the edges. If the subsets are large enough, they will be able to represent segments of different directions, long enough to give a good estimation of the mean slope in a window of the image. Since the exact location of the segments is not important, the loss of locality will not affect the classification of the image.

Issues on preprocessing

As explained in the previous section, the basic operation during learning is the construction of the associative memory in terms of matrices that relate patterns and classes.

Even for a small receptive field, say 5×5 subregion, the number of possible patterns, $N_{patterns} = 256^{5 \times 5}$, is too large to be explicitly addressed. Therefore it is necessary to reduce drastically the size of the pattern space without losing important pieces of information.

This can be done by choosing a set of feature selectors. Every pattern appearing in a subset is filtered by the feature selectors and the resulting activation vector, $v_1, v_2, \dots, v_{N_{features}}$ is quantized in $[0, Q]$. The quantized vector is the code associated with the pattern. The number of possible codes is a function of the number of features and of the quantization parameter Q , $N = Q^{N_{features}}$.

The net result of feature extraction is to project the original pattern space into a smaller code space. Therefore it is important to choose a set of uncorrelated features that span the region of the pattern space that contains most information. It is natural to choose primary features that emphasize local directions, i.e. the principal directions of the pattern contained in a subset. In this work we use features segments of 8 directions, one every 22.5 degrees [Stringa, 93].

The feature extractors are analogous to the simple cells of the neo cognitron [Fukushima *et al.*, 1983]. These are cells that respond to particular local features. In our case we have 8 different cells responding to 8 directions. The activation function of a cell can be defined in different ways, but the choice is not crucial. Two methods were tried. One is convolution with 5×5 masks of segment-like Laplacian operators. In the second we discretize the direction of the gradient in $[0, 7]$, and average the result in each 5×5 subset. In this application the gradient method gives better results.

The level of activation of a cell is quantized in $[0, Q]$. The resulting vector is the code associated with the given pattern. For $Q = 1$ the number of possible codes is 2^8 , if $Q = 2$ we have $3^8 = 6651$ codes. With $Q = 0$ we indicate a *winner takes all strategy* in which only the cell with maximum activation generates a code.

We tested the classification capability of the system with real images, real preprocessed images and virtual images.

Artificial images (128×128) are composed by edges.

The virtual environment is a corridor, learning and testing is done in a strip 10-meters long and 2-meters wide. The angle varies in the range $[-30, +30]$ degrees. Spatial positions are discretized in 5×5 classes. Since virtual images are simple and noiseless, the performance of the system is very good for any reasonable choice of parameters. The number of correct answers is around 98%.

The data base of real images consists of 450 images (128×128) of a corridor of our institute. Images are randomly distributed in a strip 10-meters long and 1-meter wide, the angle is in the range $[-20, +20]$ degrees. In typical tests 400 images are used for learning and 50 for testing. The data base is randomly subdivided between learning set and test set.

The number of spatial classes used is nine, three classes for attitude and three for lateral displacement. With $Q = 1$, the system classifies correctly 79% of the images; in the 20% of the cases only angle or displacement is correct, in 1% of the cases both classes are wrong. For $Q=0$ we have (76%, 23%, 13%) respectively. The performance improves if the code space is enlarged ($Q=2$) and the number of patterns is increased, for example by using overlapping subsets. In this case the performance reaches $\sim 85\%$ of right answers but the processing time increases. A more complicated preprocessing, such as the use of an "edge follower" does not improve performance.

In the version implemented on the robot we use 15 classes, 3 classes for position and 5 classes for angle. More classes are needed for attitude because the range is wider than in the data base. In the Fourier method the features are the orthogonal Fourier basis functions with wavelength of 2-4 pixels and the code is obtained by a quantization of the absolute value of the corresponding coefficients.

In ref.[Crespi *et al.*, 1992] we tested a processing method based on local 2-D Fourier transform. The method uses a much larger code space, $N_{codes} \sim 2^{30}$. Since the matrices are too large to be explicitly stored, memory addressing is based on hash table techniques.

The 2-D Fourier coefficients

$$C(p, q) = \sum_{x, y=0}^{N-1} S(x, y) e^{\frac{2\pi}{N} i (p x + q y)} \quad p, q \in \left[-\frac{N}{2}, \frac{N}{2} - 1\right] \quad (4)$$

represent directions. The direction corresponding to coefficient $C(p, q)$ is given by $(q, -p)$. For example, $C(3, -3)$ represents the 45° direction.

In every subset we compute the absolute values of coefficients with $\theta_{max} \geq p^2 + q^2 \geq \theta_{min}$, where θ_{max} and θ_{min} are chosen thresholds. Values corresponding to the same slope are added together. The resulting values represent the directional spectrum of the subset and are quantized in the interval $[0, 3]$. For receptive fields of size 6×6 we consider 16 different directions corresponding to $\theta_{min} = 5$, $\theta_{max} = 18$. The number of possible patterns is 4^{16} .

In the Fourier method the features are the orthogonal Fourier basis functions with wavelength of 2-4 pixels and the code is obtained by a quantization of the absolute value of the corresponding coefficients.



Figure 4: The robot.

The performance of the Fourier method on the same data base is similar, but the processing time is longer. This is due to the larger size of the space code. For navigation purposes only a small number of features is sufficient. However in a more complex environment information other than simple slope could become important.

Connections with similar paradigms

The simplest instance of the memory-based approach is the *nearest-neighbor* regression schema: the response for a new input $\mathbf{x}$ is obtained by averaging the responses of the reference patterns contained in a neighborhood of the input pattern.

More generally, in the *Parzen-window* schema non-negative *kernel* functions are evaluated for each new input. Local estimation is then obtained with kernel functions usually having a maximum at prototype pattern $\mathbf{x}$ and decreasing value away from $\mathbf{x}$.

Many learning schemata usually placed in the neural networks paradigm are a further extension of this idea. A major example is given by networks implementing the approximation schema $f(\mathbf{x}) = \sum_{\alpha=0}^N c_{\alpha} G_{\alpha}(\|\mathbf{x} - \mathbf{t}_{\alpha}\|)$ where the basis functions $G(\|\mathbf{x} - \mathbf{t}_{\alpha}\|)$ exhibit a *radial* symmetry with respect to the *centers* $\mathbf{t}_{\alpha}$. Position of centers and parameters of the expansion are learned via presentation of exemplar pairs (*input, output*) and optimized with respect to a cost functional. A well founded extension of this schema is the HyperBF network (see [Poggio, 1990] and references therein). A HyperBF network is applied in [Aste and Caprile, 1992] to address the same navigation task for a robot moving in a synthetic world and with a more elaborated preprocessing phase.

We have mentioned above how the use of memory-based reasoning on massively parallel computer was advocated as a possible foundation of machine-reasoning in [Stanfill and Waltz, 1986]. Although we are not explicitly using a distance metric between templates and new inputs, our approach is consistent with the *similarity-based* induction criterion proposed in that paper. Our navigation module implemented on a PC board indicates that memory-based reasoning is a valid approach to information processing even on standard computers.

Experimental Setting

The robot used for the experiments is a small platform (see Fig. 4) developed at IRST [Bellutta, 1991]. The motion is governed by two stepping motors programmable in terms of displacement or motor speed and acceleration. The robot measures 40×40 cm and weights 30 Kg. The robot "brain" is a 486 CPU (25 MHz) equipped with a visual system consisting of a Matrox PIP frame grabber and a gray-level camera.

In the navigation experiment, spatial positions (d, a) of the robot in a corridor are represented by 15 classes, 5 classes for attitude angle a in the interval $[-45^\circ, +45^\circ]$ degrees, and 3 classes for lateral displacement d from the centerline in the interval $[-0.60 \text{ m}, 0.60 \text{ m}]$. To better control the system performance we have chosen to label images with their spatial position and not to associate images and commands directly as done in [Pomerlau, 1991].

3 Results and Discussion

In Fig. 5 we show some typical trajectories of the robot. The system implemented on the robot is based on the 8-direction feature extractors. Similar results were obtained with Fourier preprocessing in [Crespi *et al.*, 1992]. In Fig. 6 some trajectories obtained with the two methods are compared. The robot is trained in the gray areas. In each area the robot memorizes 225 images, 3 images for each spatial class. During performance the robot changes its steering and driving velocities according to the image present. The driving speed is about 0.5 m/sec and the processing speed is 2.5 images per second.

To test the generalization capability of the system, learning is limited to a small part of the corridor. The robot is able to move along the entire corridor and avoid obstacles that narrow the perspective of the corridor, such as air conditioners and plants set along the walls (see Fig. 5.b,d). Furthermore it is able to navigate in other corridors of different geometrical shapes, see for example Fig. 5.d. System performance does not depend on illumination conditions. Learning and testing were done at different times and under very different lighting conditions.

Deviations of the trajectories from the centerline are usually due to the presence of windows, reflections on the walls, openings, etc. The deviation is usually confined to a 50 cm band around the "optimal" trajectory.

The experiments show that a memory-based architecture can provide an efficient and flexible method for processing sensorial inputs even using standard sequential computers. The basic system can be easily adapted to a specific application and has a good generalization capability.

Our results indicate that the extraction of local directional fields is an important technique in processing visual information. For navigation purposes only a small number of local features and a single layer network is sufficient in the analysis of the visual input. For more complex tasks the system could be easily expanded to include several layers in which the local features of the first layer are combined into higher order structures.

References

- [Aste and Caprile, 1992] M. Aste and B. Caprile. Learning Autonomous Navigation Abilities Using Radial Basis Functions Networks. In *Proceedings of the Intelligent Vehicles 92 Symposium*, pages 241–246, 1992.
- [Bellutta, 1991] P. Bellutta. Basic robot utility technical reference manual. IRST-TR #9112-16, IRST, 1991.
- [Crespi *et al.*, 1992] B. Crespi, C. Furlanello, and L. Stringa. A memory-based approach to navigation. *Biological Cybernetics*, (accepted), 1992.
- [Fukushima *et al.*, 1983] K. Fukushima, S. Miyake, and T. Ito. Neocognitron: a neural network model for a mechanism of visual pattern recognition. *IEEE Trans. on Systems, Man and Cybernetics*, SMC-13(5):826–834, 1983.
- [Geman *et al.*, 1992] S. Geman, E. Bienenstok, and R. Doursat. Neural networks and the bias/variance dilemma. *Neural Computation*, 4:1–58, 1992.
- [Poggio and Stringa, 1992] T. Poggio and L. Stringa. A project for an intelligent system: Vision and learning. *International Journal of Quantum Chemistry*, 42(727), 1992.
- [Poggio, 1990] T. Poggio. A theory of how the brain might work. *Cold Spring Harbor Symp. Quant. Biol.*, pages 899–910, 1990.
- [Pomerlau, 1991] D. A. Pomerlau. Efficient training of artificial neural networks for autonomous navigation. *Neural Computation*, 3(88), 1991.
- [Stanfill and Waltz, 1986] C. Stanfill and D. Waltz. Towards memory-based reasoning. *Comm. of the ACM*, 29:1213–1228, 1986.
- [Stringa, 1990] L. Stringa. S-nets: A short presentation. In E. R. Caianiello, editor, *Proceedings of the Second Italian Workshop on Parallel Architectures and Neural Networks*, pages 177–184. World Scientific Publishing Co., May 1990.
- [Stringa, 1992] L. Stringa. Il progetto Maia. *Le Scienze (Italian ed. of Scientific American)*, (290):86–97, 1992.
- [Stringa, 93] L. Stringa. A visual model for pattern recognition. *Int. J. of Neural Systems (in press)*, 93.