AN APPROACH TO BEACONS DETECTION FOR A MOBILE ROBOT USING A NEURAL NETWORK MODEL

Reçu le 02/01/2005 – Accepté le 28/06/2005

Résumé

Dans cet article nous proposons une technique neuromimétique relative à la détection de balises en robotique mobile. L'objectif est d'amener un robot se déplaçant dans un environnement quelconque. à acquérir des attributs en vue d'une reconnaissance. Nous développons en amont une approche pratique par la segmentation d'images d'objets d'une scène et en évaluer les performances en temps réel. Le classifieur neuronal utilisé est une fenêtre d'un réseau MLP (9-6-3-1) utilisant l'algorithme de rétro-propagation du gradient, où le pixel central distribué utilise l'information en niveau de gris. L'originalité de notre travail réside dans l'utilisation d'une technique hybride coopérative associant les RN et une méthode de paramétrisation qu'est la Transformée de Hough généralisée pour mieux cerner et lever la problématique de la reconnaissance de formes. Les résultats obtenus avec un momentum de 0.3 et un coefficient d'apprentissage égal à 0.02 montrent que notre système est robuste avec un temps de calcul fort appréciable.

Mots clés *:Classifieur neuronal, image de procédé, rétro-propagation, détection et extraction d'objets, transformée de Hough.*

Abstract

In this paper we propose a neuro-mimetic technique relating to the detection of beacons in mobile robotics. The objective is to bring a robot moving in an unspecified environment to acquire attributes for recognition. We develop a practical approach for the segmentation of images of objects of a scene and evaluate the performances in real time of them. The neuronal classifier used is a window of a network Multi-layer Perceptron MLP (9-6-3-1) using the algorithm of retro-propagation of the gradient, where the distributed central pixel uses information in gray level. The originality of the work lies in the use of the association of an enhanced neural network configuration and Standard Hough Transform. The results obtained with a momentum of 0.3 and one coefficient of training equal to 0.02 shows that our system is robust with an extremely appreciable computing time.

Keywords: *Neural network classifier, image processing, back-propagation network, detection and object extraction, Hough Transformer.*

A BOUTARFA
N. BOUGUECHAL
Y. ABDESSEMED

Advanced Electronics Laboratory (LEA), University of Batna, Algeria,

ملخص

نقترح في هذه المقالة موضوعة في مجال الآلي المتحرك و ذلك بواسطة الشبكة النورونية المتعددة الطبقات تقنية خاصة لكشف و إظهار روا سم الإرشاد "معالم خاصة". إن هدفنا الأساسي يكمن في تطوير المقاربة التجريبية و التطبيقية التي تعتمد أساسا على الشبكة النورونية. إن مصنف الشبكة النورونية و الكاشف للأجسام في مشهد يخضع لآلية التطور معتمدا في ذلك على نافذة كاسحة لمجال المشهد المعني بالدراسة التجريبية. بواسطة خوارزم الانتشار الرجعي تتكون الشبكة من نماء و انتشار ذو خصائص إيحائية خلفية و كذا من حيوية مختارة بكيفية جد ملائمة، عدة عوامل و وسائط تحوي في طياتها ثوابت قيمة، و كذا جملة من الممارسات الشاهدة على العمليات التجريبية و أيضا معايير ابتدائية محللة بمنهجية و ترتيب عالي الفعالية. و نشير أيضا أن هنالك اربع طوابق " 9-6-3-1 " واضحة و متناسبة. إن أحسن النتائج و أدقها من ناحية الفعالية والصلبة تحصلنها بواسطة نسبة متوسطة الميل تقارب 0،5 و معامل التعليم 0،05 .

<u>الكلمات المفتاحية</u> *: المصنف النوروني, خوارزم الانتشار الرجعي , شبكة متعددة الطوابق , تحول Hough*

Under the words "neural networks", we group today a number of models, hich attempt to imitate some human brain functions in order to try to implement some of its basic structures. Neural network models are specified by the net topology, the node characteristics and the learning rules. The rules specify an initial set of weights and indicate how weights should be adapted during use to improve performance. Different types of neural networks are proposed in the literature by several authors [1, 2]. Some of the most important models are those of Hopfield, Kohonen, single layer and Multi-layer Perceptrons. The Hough Transform (HT) has long been recognized as a robust technique for detecting multi-dimensional features in an image and estimating their parameters. It has many applications, as most manufactured parts contain feature boundaries, which can be described by regular curves or straight lines.

Its main advantage is that it is tolerant of gaps in feature boundary descriptions and is relatively unaffected by noisy image. Fig.1 illustrates this transformation.

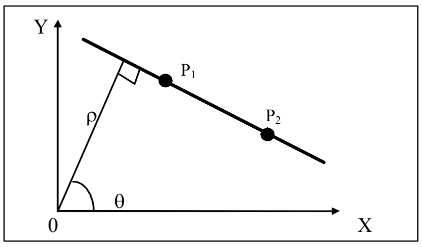

Figure 1: The straight line polar parameter.

A straight-line in a two dimensional space can be represented by two parameters, r and y using the following equation:

$$f(x, y, \rho, \theta) = \rho - x\cos\theta - y\sin\theta = 0 \qquad \textbf{(0)}$$

Where **r** is the normal distance between the line and the origin, and **y** is the angle of the normal with respect to the positive x-axis.

II. PREVIOUS RELATED WORK

A serial algorithm for computing HT has complexity *O (P x n $_\theta$),* where *n $_\theta$* is the number of quantization in the *θ* space, representing a measure of the desired accuracy computational in an *N x N* image. *P* is a fraction of N^2 and represents the number of points of an image where edges exist (or where the pixel value is 1 in a binary image). However, each edge point is independent of any other points for the HT computation. This makes the HT problem suitable for parallel computing which would considerably reduce the processing time. Different aspects of the HT have been investigated and reported in the literature. The computation-bound nature of the HT has inspired the development of efficient algorithms [3–5] and implementations [6–8] of the transform in multi-processor systems. All these algorithms are developed to solve most of the Standard Hough Transform (SHT) application problems whether it is intended to reduce the computation time or it is intended to optimize the memory usage. There have been some attempts [9–11], to use neural nets for pattern recognition, natural language processing and image processing in order to have an output in real time. Most present algorithms consider the number of neurons equal to the number of pixels in the input image.

The paper of Tuytelaars and L.Van Gool [12] consists of two main contributions. First, one starts from corners and uses the nearby edges, while the second one is purely intensity-based. As a matter of fact, the goal is to build an opportunistic system that exploits several types of invariant neighbourhoods as it sees fit. This yields more correspondences and a system that can deal with a wider range of images. In this paper, we present a novel algorithm for computing of the association of an enhanced neural network configuration and Standard Hough Transform especially adapted for digital device implementation such as Field Programmable Gate Arrays (FPGA), ASICs, etc. This paper is organized as follows: First, the design of the neural network classifier is described. Next, a systematic analysis of different network configurations is presented. Then, the novel algorithm of object extraction is formulated.

Finally, we bring this paper to a conclusion and present the perspectives of this research.

III. ARCHITECTURE OF THE NEURAL CLASSIFIER

To detect and extract an object from a scene, we have developed a multi-layer neural network which classifies the entire object. It is based on a network window classifier, which classifies the central pixel of a relatively small area in the image. To detect this object, the window must slide pixel by pixel over the entire image. The multi-layer feed forward network is used as the window classifier. Its input units receive the gray scale values from a square region of the image. The generalized delta rule with a momentum is used as the learning paradigm to train the network [1, 2, 10]. When the training converges and the system error decreases under an acceptable value (a small enough value), the window classifier is considered as trained and then applied over the entire object from the image. Figure 2 illustrates the architecture of the window classifier and how it is applied to the image.

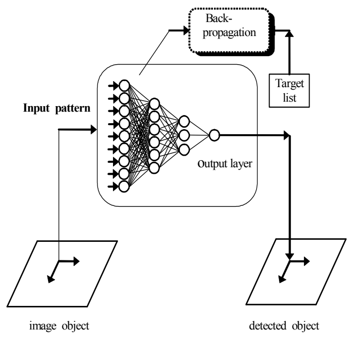

Figure 2: Architecture of the neural network window classifier

IV. ANALYSIS OF THE NETWORK CONFIGURATION

Contrarily to the mathematical theory for network training based on back-propagation, there is no general guideline for choosing the appropriate network configuration for a given problem. The results only lead to the selection of the optimal network configuration but also give us an insight to the sensitivity of adjusting network configuration.

A-Network topology

Let N be the number of weights for a given network topology. By definition, any neuron in the hidden layer must receive an input from all the neurons in the previous layer and feeds all the neurons in the next layer. Let n be the number of layers in the network and L_i be the number of neurons in the ith layer in the network. Hence, the total number of weights is given by:

$$N = \sum_{i=1}^{n-1} L_i * L_{i+1} \qquad \textbf{(1)}$$

A preliminary experiment was performed with a four layers network (two hidden layers) whose input window size varies from 3x3 to 7x7 pixels. The number of neurons in the input layer is equal to the size of the input window. 3x3 and 5x5 windows (respectively $L_i=9$ and $L_i=25$) were generally selected for the remaining experiments.

B-Rate parameters

The network is trained by the delta rule with a momentum term which was introduced by Rumelhart [9]. At the (t+1)th iteration of the training, the network weight w_{ij} is updated with two components. The first component is proportional to the error signal e_j on the output $X_{ij,}$ for the neuron sending the activation signal. The second component is proportional to the amount of weight changes in the previous iteration (the momentum term).

$$\Delta w_{ij}(t+1) = w_{ij}(t) + C1.e_j .X_{ij} + C2.\Delta w_{ij}(t) \qquad \textbf{(2)}$$

Where **C1** is the learning rate and **C2** the momentum term.

C1 controls the rate of convergence for the training process and C2 prevents the oscillation problem near the solution point. These two parameters are the major factors which affect learning [10]. They also control the tradeoff between system's stability and classification quality. The learning rate decreases generally starting from 0.5 and the momentum term varies between 0 and 1.

C- The transfer function:

The transfer function must be continuous and monotonous [2]. Two types of transfer functions generally used are given below:

- The sigmoid function:

$$f(z) = \frac{1}{1.0 + e^{-z}} \qquad \textbf{(3)}$$

Where **z** varies between $-\infty$ and $+\infty$ and **f (z)** varies between 0 and 1. The derivative of f (z) is:

$$f'(z) = f(z).(1.0 - f(z)) \qquad \textbf{(4)}$$

- The hyperbolic tangent:

$$f(a) = \frac{exp(a) - exp(-a)}{exp(a) + exp(-a)} \qquad \textbf{(5)}$$

Where a varies between $-\infty$ and $+\infty$ and **f (a)** varies between -1 and +1. The derivative of f (a) is:

$$f'(a) = (1.0 + f(a)).(1.0 - f(a)) \qquad \textbf{(6)}$$

The sigmoid function has been used in our application because it yields the best results.

D- Network initializing weights

Learning by back-propagation has been successful in many applications. However, the possibility of being trapped in a local minimum of the error function during the training exists. The components of the weight vectors are chosen as a real random number in the interval [0, 1].

E-The generalized delta rule

The generalized delta rule is the description of the multi-layer back-propagation network [2, 11].

The algorithm is given as follows:

Step 1: apply an input vector to the neural network and calculate the output values.

Step 2: compare current outputs with real outputs, and then determine the error.

Step 3: back-propagate the error in all the hidden layers and calculate errors for all units of these layers.

Step 4: determine the weights variations which will adjust each weight connection of the neural network.

Step 5: apply corrections to the connection weights of the network.

F-Selection of the Optimal Threshold T

The threshold T divides the neurons set into two groups (thereby the image space into two regions: the object and the background). The weights of only one group are updated. For a particular threshold, the updating procedure continues until it converges. When the network converges, the image space is divided into two groups, one group having d (w.U)>T (the object pixels, see equations 8 & 9), and the other with d (w.U) <T (the background pixels). The threshold T is calculated by an arithmetical method, using the average of the values present at the output layer for each pixel of the image at the entry of the neural network.

V. FORMULATION OF THE OBJECT EXTRACTION ALGORITHM

Extraction of an object is often done either by gray level thresholding or by pixel classification (region growing). In this section an object extraction algorithm will be formulated using a neural network. The weights will be assigned as random numbers in [0, 1]. A neural network classifier for detecting objects from a scene is thus developed. The implemented algorithm is given below.

1. Algorithm 1

Step 1: Assign input U(i,j) to the neuron y(i,j) using the following equation :

$$U(i,j) = (x(i,j) - l_{min}) / (l_{max} - l_{min}) \qquad \textbf{(7)}$$

where x(i,j) is the gray level of the (i,j)th pixel, and l_{max}, l_{min} are the maximum and minimum available gray levels of the image. Note that the value of U(i,j) lies in the range [0, 1].

Step 2: Initialize the weights of the neurons by a random real number lying in the interval [0, 1].

Step 3: *C*ompute for all neurons the product:

$$d(i,j) = U(i,j).\, w(i,j) \qquad \textbf{(8)}$$

Step 4: Apply the back-propagation learning algorithm (algorithm2).

Step 5: Select the neurons verifying the condition:

$$d(i,j) > T \qquad \textbf{(9)}$$

Step 6: Repeat steps 5 and 6 until convergence.

Step 7: Vary T, iterate steps 2 and 5 and find the optimum one.

The back-propagation algorithm learning can be written in algorithmic form as follows [10, 13]:

First, note that i, k, l, j are the layers of the neural network, respectively the input layer, the first hidden layer, the second hidden layer and the output layer for the multi-layer neural network classifier.

2. Algorithm 2:

Step 1: Calculate the output y_k for the first hidden layer :

$$y_k = fct1(a_k^{[c]}) \qquad \textbf{(10)}$$

with a_k the weighted sums of the entries:

$$a_k = \sum_k w_{ik} . x_i \qquad \textbf{(11)}$$

fct1 is the sigmoid function given by equation (3).

Step 2: Calculate the output z_l for the second hidden layer:

$$z_l = fct1(b_l) \qquad \textbf{(12)}$$

with b_l the weighted sum of the entries :

$$b_l = \sum_l w1_{kl} . y_k \qquad \textbf{(13)}$$

Step 3: Calculate the output u_j of the output layer:

$$u_j = fct1(c_j) \qquad \textbf{(14)}$$

with c_j is the weighted sum of the entries:

$$c_j = \sum_j w2_{lj} . z_l \qquad \textbf{(15)}$$

Step 4: Calculate the error of the output layer:

$$e3_j = (d_j - u_j).fct2(c_j) \qquad \textbf{(16)}$$

where d_j is the desired output and *fct2* which is given by equation (4) is the derivative of function *fct1.*

Step 5: Calculate the error of the second hidden layer using the output error:

$$e2_l = e3_j . w2_{lj} . fct2(b_l) \qquad \textbf{(17)}$$

Step 6: Calculate the error of the first hidden layer using the error of the second hidden layer

$$e1_k = e2_l . w1_{kl} . fct2(a_k) \qquad \textbf{(18)}$$

Step 7: Calculate the weight variation for all the layers using the learning rate C1 and the momentum rate C2:

$$\Delta w_{ik} = C1.e1_k.x_i.C2.(C1.e1_k.x_i)$$

$$\Delta w1_{kl} = C1.e2_l.y_k.C2.(C1.e2_l.y_k) \qquad \textbf{(19)}$$

$$\Delta w2_{lj} = C1.e3_j.z_l.C2.(C1.e3_j.z_l)$$

Step 8: Update the weights:

$$w_{ik}(t+1) = w_{ik}(t) + 0.9\Delta w_{ik}$$

$$w1_{kl}(t+1) = w1_{kl}(t) + 0.9\Delta w1_{kl} \qquad \textbf{(20)}$$

$$w2_{kj}(t+1) = w2_{kj}(t) + 0.9\Delta w2_{kj}$$

Step 9: if the algorithm converges (e3 inferior to a predefined threshold) stop the process, if not, return to step 1.

Note

It is essential to improve the rule of the modification of the weights in the algorithm of the retro-propagation by adding the following equation:

$$\Delta W(t+1) = \Delta W(t) + 0.9W$$

This major change enhances considerably the reliability of the algorithm.

COMPUTER SIMULATIONS AND EXPERIMENTAL RESULTS

In all the following computer simulations, we use various types of beacons (polyhedral, cylindrical, and rectangular). The neural network classifier takes the image data of specific beacons directly as an input and then back propagates the signal errors in order to adapt the weighing (synaptic) coefficients.

The configuration of the network is crucial to obtain a satisfactory performance within an acceptable training time. The proposed algorithm was tested on several images. The used images are digitized on 8 bits (256 gray levels scale). Tests with different sizes of input windows (3x3, 5x5, 7x7, and 11x11) were done. The best results were obtained particularly with the input size 3x3.

The four-layer network (9-6-3-1) generalizes well for either artificial or real test images.

The proposed algorithm was applied to the images shown in figures 3a and 4a. The extracted objects are shown respectively in figures 3b and 4b. The best results are obtained using an input window of 3x3 sizes.

The optimal threshold values are 0.220 and 0.125 respectively for figures 3b and 4b.

Figures 5 and 6 show also the detection of beacons in the scene for the other shapes..

From the output and the previous discussion, it is evident that the best detection is obtained using the input window 3x3. Increasing the number of hidden layers implies training difficulties and fails to gain in classification performance. Thus, we conclude that the four-layer network (two hidden layers) is sufficient for the segmentation problem. By increasing the input window size, the computing time also increases and the resulting output images are noisier. The rate parameters for the back-propagation learning affect not only the learning rate but also the classification performance. The appropriate choice for our problem is a small learning rate (C1=0.02) combined with a medium momentum rate (C2=0.3).

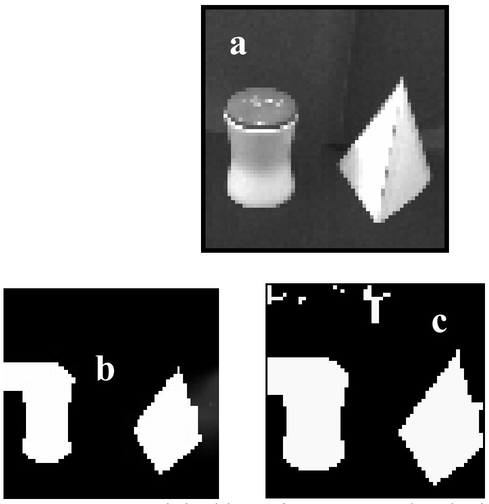

Figure 3: (a): original input image; **(b)**: The obtained result with a 3x3 window size; **(c)**: The obtained result with a 5x5 window size

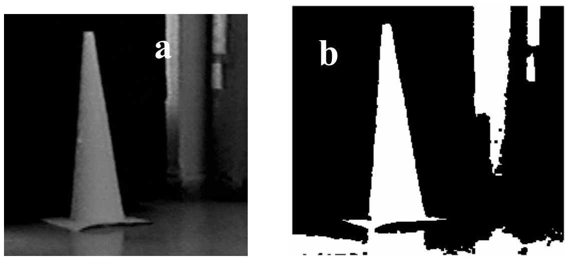

Figure 4: Polyhedral beacon. **(a):** original input image, gray level scale image; **(b):** The obtained result with a 3x3 window size

CONCLUSION

In this paper, we have presented a neural network based application in image processing. Our contribution was to extract objects present in an image of a scene and to distinguish them from the background by a neural network classifier. Its originality lies in the fact that it is based on a hybrid parametric technique which uses the neural network configuration and the generalized Hough transform for the detection of a beacon. It consists of combining the techniques of training and description to elaborate work out samples for the objects recognition. The classifier includes a multi-layer feed-forward network window in which the central pixel is classified using gray- scale information within the window. The mobile robot has been made familiar with the environment during an initial training phase. The artificial neural learning system is made to detect only the beacons from other items such as the blurs shown in figures 3 and 4, although the mobile robot is moving in the scene. The visual information (the input data to the NN) is computed using a stochastic parametric activation function of the neuron. The network is trained using the back-propagation algorithm with a momentum term. Factors including rate parameters, training sample sets, and initial weights are systematically analyzed. The output computation can be done in real time. The performance is higher than other research work carried out by other authors [10],[11],[14] because the learning is of the supervised type, and the modification of the attributes is carried out by a probabilistic rule. The experiments of figures 3 and 4 have been carried out in a noisy environment and background. The blur (the remaining noise in the background of the scene) is subtracted in the correspondence stage, which uses a predefined data base.

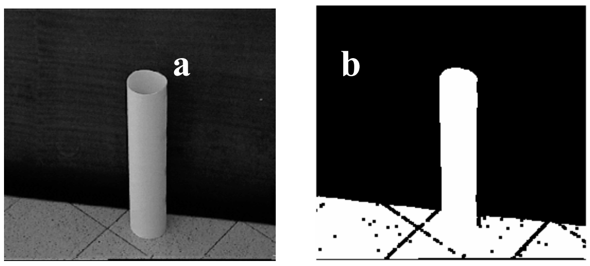

Figure 5: Cylindrical beacon. **(a):** original input image, gray level scale image; **(b):** The obtained result with a 3x3 window size.

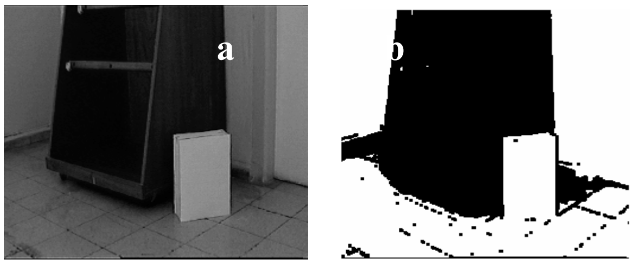

Figure 6: Rectangular beacon. (a): original input image, gray level scale image; (b): The obtained result with a 3x3 window size.

This data base includes internal models of the beacons (or artificial landmarks). The correspondence between the object of interest (the beacon) and the filtered image is found by matching the two-dimensional projections of the beacon by considering all possible sets of the available beacons in the internal environment of the mobile robot. While joining HT (technical multi-resolution, decision, fusion), our system remains robust at the price of an extremely appreciable computing time. The time requirements to process a frame, after the learning stage, with an IBM-PC having a simple Pentium processor is about a few milliseconds during the operation of the mobile robot, which seems better than other published results obtained in the same conditions of work [14, 15, 16]. Our experiments have been carried out on the mobile robot (Automated Guided Vehicle AGV), shown in figure 7, which is equipped with a camera. The results obtained by our algorithm are very satisfying since the objects are clearly extracted and detected from the original images. Also, the obtained results from this study give us a valuable insight into the role of network topology, rate parameters, training sample set and initial weights.

The FPGA device will be included in the visual navigation system of our laboratory mobile robot. After edge restoration, the device's results will be used either as the basic data to determine the depth of an object in three dimensions by geometric reasoning, or as input data to other algorithms such as the matching algorithms developed in [17] and the navigation algorithm developed in [18–20] to endow the mobile robot with the intelligence required to perceive its environment.

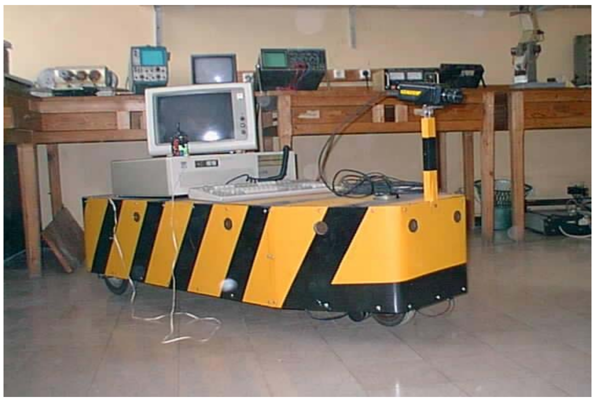

Figure 7: AGV with camera.

To increase the robustness of the system, two semi-local constraints on combinations of neighbourhood correspondences are derived (one geometric, the other photometric). They allow testing the consistency of correspondences and hence to reject falsely matched neighbourhoods. Experiments on images of real-world scenes taken from substantially different viewpoints demonstrate the feasibility of the approach.

REFERENCES

[1] A. Ghosh and K. P. Sankar, "Neural Network, self organization, an object extraction", Pattern Recognition Letters, volume 13, n°5, May (1992).

[2] Resa Nekovei and Ying Sun, "Back-propagation network audits configuration for blood vessel detection in angiograms" .IEEE Transactions on Neural Networks, Volume 6, N°1, January (1995).

[3] Atiquzzaman M. Multiresolution Hough transforms—an efficient method of detecting pattern in images. IEEE Transactions on Pattern Analysis and Machine Intelligence (1992); 14(11):1090-5.

[4] Koshimizu H, Numada M. FIHT2 algorithm: a fast incremental Hough transform. IEICE Transactions (1991); E74 (10).

[5] Philip KP, Dove EL, McPherson DD, Gotteiner NL, Stanford W, Chandran KB. The fuzzy Hough transforms feature extraction in medical images. IEEE Transactions on Medical Imaging (1994); 13(2):235-40.

[6] Choudhary AN, Ponnusarry R. Implementation and evaluation of Hough transform algorithm on a shared—memory multiprocessor. Journal of Parallel and Distributed Computing (1991); 12:178–88.

7] Lotufo RA, Dagless EL, Milford DJ, Morgan AD, Morrissey JF, Thomas BT. Hough transform for transputer arrays. In: Proceedings of the third international conference on image processing and its applications, IEEE proceedings, London, (1994), pp. 122–33.

[8] Tagzout S, Achour K, Djekoune O. Hough transform for FPGA implementation. Elsevier Journal, Signal Processing (2001); 81(6):1295–301.

[9] Rumelhart D. E., J. McClelland and PDP Research group, "Parallel Distributed Processing", Explorations in the Microstructure of recognition, volume 1, MIT Press, Cambridge, MA (1986).

[10] Fnaiech F., M. Sayadi, et M. Najim, "Factored and fast algorithms for training feed forward neural networks", ESST de Tunis (1997).

[11] Caplier A., F. Luthon et C. Dumantier, "Real time implementations of an mrf-based motion detection algorithm, special issue on real-time motion analysis", Journal of real time imaging, vol. 4, n°1, February (1998), pp. 41-54.

[12] T.Tuytelaars and L.Van Gool, “Matching Widely Separated Views based on affinity Invariant Neighbourhoods”, International Journal on Computer Vision, July (2003).

[13] Kohonen T, "An introduction to neural networks", Neural Networks 1, 3-16, (1988).

[14] Qing Song, Jizhong Xiao and Yeng Chai Soh, "Robust back propagation training algorithm for multilayered neural tracking controller", IEEE Transactions on Robotics and Automation, vol. 10, n°5, September (1999).

[15] S.Baluja, , Evolution of an artificial neural network based autonomous land vehicle controller, ALVINN, IEEE Transactions on Systems, Man and Cybernetics, Part B, Vol. 26, No. 3, June (1996), pp. 450-463

[**16**] L. Paletta, E Rome and A. Pinz, "Visual object detection for autonomous sewer robots", IROS'99, Proceeding of the 1999 IEEE/RSJ, International Conference on Intelligent Robots and automated Systems, Kyougju, South Korea, October 17-21, (1999), pp. 1087-1093.

[17]- Oualid A. Djekoune AO, Achour K, Zoubiri H. Segments matching using a neural network approach. ACS/IEEE International conference on computer systems and applications, AICCSA’ 01, June 25-29, (2001), Beirut, Lebanon. pp. 103-105.

[**18]** K. Achour, O. Djekoune, “Localisation and guidance with an embarked camera on a mobile robot”. Advanced Robotics, (2002), (16:1).

[19] Passold F., M.R Stemmer, "Feedback error learning neural network applied to a Scara robot", RoMoCo’04, Proceedings of the fourth international workshop on robot motion and control, June 17-20 (2004), Puszczykowo, Poland, pp. 197-202.

[**20]-** K. Achour, O. Djekoune, “Incremental Hough transform: an improved algorithm for digital device implementation” Real Time Imaging, Elsevier, (2004).