

RECOVERING 3-D FORM FEATURES BY A CONNECTIONIST ARCHITECTURE

E. Ardizzone, A. Chella*, R. Pirrone, F.Sorbello

DIE - Dipartimento di Ingegneria Elettrica Universita' di Palermo
Viale delle Scienze 90128 Palermo Italy
E-mail: ardizzone@vlsipa.cres.it

*DIE - Dipartimento di Ingegneria Elettrica Universita' di Palermo
and
CRES - Centro per la Ricerca Elettronica in Sicilia
Monreale (Palermo) Italy.

Abstract

An original approach to the recovery of 3-D shape information from 2-D images is described. The approach is based on an architecture made up by two cascaded neural networks. The first network is an implementation of the Boundary Contour System aimed to extract a brightness gradient map from the image. The second is a backpropagation network that estimates the geometric parameters of the object parts present in the acquired scene.

A description of simulations of the implemented architecture and the quite satisfactory experimental results are reported as well as the comparisons with some classic approaches to the problem.

Key words: Artificial Vision, Connectionism, Shape from Shading.

Please address future correspondence to:
Dr. Edoardo Ardizzone
DIE - Dipartimento di Ingegneria Elettrica
Universita' di Palermo
Viale delle Scienze
90128 Palermo Italy

1. Introduction

Classical approaches to tridimensional perception are generally based on several information processing stages, starting from features extracted from the 2-D image, e.g. to obtain the volumetric representation of the object [Marr, Horn]. According to these approaches, functional and structural relations among objects may not be obtained by simple local analysis because heavy, global assumptions like regularity and isotropy would be necessary. Assumptions of this kind generally make the model not well suitable to describe the large variety of objects of the real world.

Instead, the general framework followed by the authors to 3-D description of a scene is based on the direct recognition of simple 3-D geometric primitives by which the objects in the scene may be decomposed; the objects are therefore described as simple combinations of those recognised parts acting as “building blocks” [Pentland, 1986]. The geometric description of 3-D primitive parts is then the starting point for the conceptual description of the scene, as pointed out in [Ardizzone et al, 1989].

An original approach aimed to the direct extraction of 3-D geometric parameters of the parts describing a scene is the object of the present work. The approach is based on an artificial neural network which is able to estimate the form parameters of the geometric primitives best fitting the objects in the scene from the information coming from the shading of the objects' shape.

Neural networks are perhaps a very suitable way to face this class of problems because of their ability to derive good solutions in a non analytic form for all the elements of the domain of interest being trained with a small set of meaningful examples belonging to this domain.

Relative to the presented system, two neural networks learn how to estimate the geometric parameters from examples presented during the training phase of the architecture; they are capable of generalisation in order to estimate novel input not presented during the learning phase. The approach based on neural networks overcomes the shortcomings of the classical shape from shading algorithms: the operation of the neural network, which may be regarded as a non-linear regression operation [Hecht-Nielsen, 1990]

allows neural networks to be robust with respect to noise, change of illumination and environmental conditions; these features make them most suitable for real world application. Moreover, after the network is trained, the operation is very quickly and it not presents the heavy computational load typical of classic algorithms. The network operation is intrinsically parallel and it therefore well suited for implementation in parallel hardware architectures [Ardizzone et al, 1991b].

2. Theoretical remarks

2.1 The adopted geometric primitives

In order to make a 3-D reconstruction of the objects present in the perceived scene, a Constructive Solid Geometry (CSG) [Ardizzone et al, 1989] model has been adopted whose primitives are superquadrics. Each object is therefore described by means of simple Boolean operations among superquadrics.

Superquadrics are geometric shapes derived from the quadrics' parametric equation rising the trigonometric functions to two real exponents [Barr, 1981]:

$$\vec{x}(\eta, \omega) = \begin{pmatrix} a_1 \cos \eta^{\varepsilon_1} \cos \omega^{\varepsilon_2} \\ a_2 \cos \eta^{\varepsilon_1} \sin \omega^{\varepsilon_2} \\ a_3 \sin \eta^{\varepsilon_1} \end{pmatrix} \quad -\pi/2 \leq \eta \leq \pi/2; \quad -\pi \leq \omega \leq \pi$$

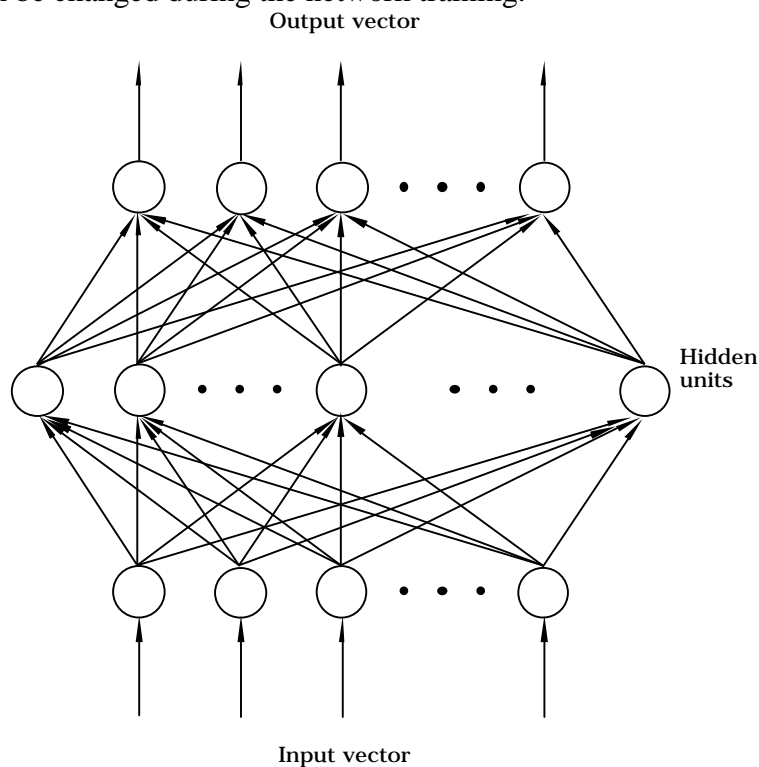
where η and ω are the latitude and longitude parameters. The a-parameters are the superquadric semiaxes lengths; the exponents ε_1 and ε_2 are the form factors and affect the appearance of the surface of the superquadric respectively in latitude and longitude. If these factors assume values less than 1 the shape is somewhat squared. If the value is nearly 1 the shape is rounded; if the exponents approach 2 the shape is locally flat with some sharp edges. Finally, if the values of these factors are greater than 2 the surface exhibits cusps. The previous equation is the canonical form parametric equation of a superellipsoid characterised by 5 parameters. Three orientation parameters and the centre coordinates are to be added to these quantities to completely describe a generically displaced superquadric.

It is possible to deform a superquadric surface in several ways, modifying its shape factors. This characteristic makes the superquadrics very suitable primitives for a geometric modelling system because they can easily approximate the shapes of the objects in the real world: as an example, fig. 1 shows examples of superquadric when varying the form factors.

FIGURA SUPERQUADRICHE

2.2 Main features of artificial neural networks

A neural network is a parallel device made up by several elementary processors usually called units: in general each unit receives different inputs and sends its output to a certain number of other units. Generally the units are organised in layers: all the units of a layer receive their input from the lower layers and send their output to the above layers; normally the units of a same layer are not connected to each other (see fig. 2). Connections between units can be more or less strong: this aim is obtained making use of modifiable weights that can be changed during the network training.



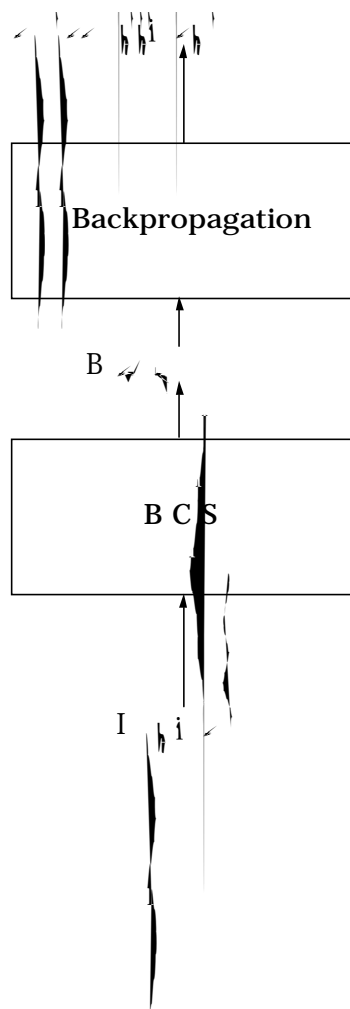
Each unit performs its processing task through a firing rule and a transfer function: the input of each unit is made up by the sum of the inputs from the other units each weighted by the strength of its connection. This global input is processed using the firing rule and, if the unit results activated, it provides the output using its transfer function.

On the whole, this kind of device is able to perform a mapping between the inputs' and the outputs' space by means of a training phase during which some input-output couples are presented to the network. These couples make the weights of the connections to change according to a particular learning rule in order to obtain a good correspondence between the input and the output vectors. If the training examples are chosen to be very meaningful, the network exhibits a good generalisation ability being able to associate the right output even to inputs not presented during the learning phase.

3. Description of the architecture

3.1 *The Boundary Contour System block*

The outline of the proposed architecture is shown in fig. 3.



The first block is an original implementation the Boundary Contour System (BCS) model [Grossberg et al, 1987]. The Boundary Contour System is a physiological model of low-level visual perception as an explanation of processes of contour extraction, texture grouping, shape from shading, etc. Several competitive-cooperative processes between units are at the basic operation of the model. The general output function of the unit i is:

$$\frac{d}{dt}x_i = -Ax_i + (B - x_i)I_i - x_i \sum_{k \neq i} I_k$$

where A is the spontaneous decay rate, B is the maximum activity level, $\sum_{k \neq i} I_k$ is the total input from the other units to unit i [Grossberg et al, 1987].

Each unit of the model is sensitive to a brightness gradient in a small window of the image; units in the same position locally compete in order to find the most probable direction of the brightness gradient. After this competition stage, an on-centre off-surround filtering is performed in order to ensure end-cut and allowing the system to correctly operate near contours, where the gradient sharply changes. The last stage performs the cooperation process among the aligned and same-oriented units, in order to preserve smooth variations of the brightness gradient. This stage is moduled by the previous stage: near the contours the end-cut process is prevalent with respect to the cooperation process, while far from contours the cooperation process is prevalent.

The BCS is not an adaptive model and it does not learn from examples but the internal parameters are fixed from outside. The units of the BCS are sensitive to the brightness variations due to the surface shading: the BCS traces contour lines not only for the more sharp brightness variations near the external contours of the object, but also for the less sharp brightness variations growing up from the shading. The contours so generated are called *boundary webs* allowing to determine the shape information deriving from the image shading.

The BCS then transforms a grey-level image in a boundary web map, thus allowing for a substantial data compression without loss of information about the shapes of the

objects. The BCS operation is invariant with respect to the direction of illumination and it shows a good behaviour with respect to the noise.

Output examples of the BCS system together with the corresponding input superquadrics are shown in fig. 2 a) and b). Fig. 3 shows the behaviour of the BCS when a scene made up by more than a single object is presented, allowing for the BCS segmentation capabilities. The BCS individuates two separate objects along with the occluding boundary.

3.2 The backpropagation block

The second block of the proposed architecture is cascaded to the BCS and it performs an estimate of the form parameters of the superquadrics best fitting the objects in the scene. The block is made up by a feed forward architecture implementing the backpropagation learning algorithm, because of its interesting characteristics of adaptivity, associativity and noise robustness [Rumelhart 1986].

This block is made up by three layers of units; the input unit receives input from the outside and it supplies output only to the hidden layer of units; this layer in turn supplies output to the output layer of unit which supplies output to the outside. In this kind of architecture there are no connections among units of the same layer.

The transfer function of each unit is the sigmoid function:

$$o_{pj} = \frac{1}{1 + \exp\left(-\sum_i w_{ji} o_{pi} + \vartheta_j\right)}$$

where o_{pj} is the output of the j -th unit when pattern p is presented to the network, w_{ji} is the weight connecting unit i to unit j , o_{pi} is the output of the i -th unit connected to unit j via the weight w_{ji} and ϑ_j is the bias for unit j .

The weights among units are determined during the learning phase in which the training set, made up by several couples of input and related output pattern, is presented to the network. When the input pattern p is presented to the network at time t , the activation of the input units are forward propagated to hidden and then to the output units; here the actual output of the network is compared to the target output and the error dp_j for the

generic output unit j is measured. The error for the output units is the backpropagated to all the hidden units in the previous layer, until an error is assigned to all the units of the network.

The weight between units j and i then changes at the time t according to the following equation [Rumelhart 1986]:

$$\Delta w_{ji}(t+1) = \alpha(\delta_{pj}o_{pi}) + \eta\Delta w_{ji}(t)$$

where α is the learning rate, representing the amount of change of the weight, η is the momentum term representing the inertia of the weight to change, δ_{pj} is the error of the unit j and o_{pi} is the output of the unit i on pattern p .

A learning epoch is therefore the complete presentation of all the input-output couples of the training set. The learning phase stops when the network has reached convergence.

In [Rumelhart 1986] it is shown that the backpropagation algorithm performs a gradient descent with respect to the Total Sum of Squared root error function (TSS):

$$TSS = \sum_p PSS_p = \sum_p \sum_j (t_{pj} - o_{pj})^2$$

where t_{pj} and o_{pj} are respectively the target output and the actual output of the j -th output unit of the network. The TSS is the squared sum of the differences between the actual output pattern and the target pattern summed for all patterns in the training set and it is a measure of how the network has learned the training set.

Another used measure is the Pattern Sum of Squared error (PSS):

$$PSS_p = \sum_j (t_{pj} - o_{pj})^2$$

is the squared sum of the differences between the actual output pattern and the target pattern when a single input pattern is presented to the network and it is used in order to test the operation of the network on input pattern not belonging to the training set.

Well known problems of the backpropagation architecture are the choice the internal parameters of the backpropagation architecture, such as the learning rate, the momentum, of

the number of hidden units and the number of learning epochs, which are generally resolved by trial and error processes.

About the number of unit, it should be noted that the number of input and output units are fixed from the statements of the problem, while the number of hidden units is fixed by a trial and error process. When the number of hidden units is low, the network is not able to learn the training set; when the number of hidden units is high the network tends to operate as a lookup table: it well learns the training set, but it does not generalise, i.e. it does not well operate when patterns not belonging to the training set are presented. The same problem arise with respect to the learning epochs of training: a low number of learning epoch make the network to not correctly operate, an high number the overlearning problem: the network correctly recognises the training set but it is not able to generalise.

3. Implementation of the proposed architecture

A first implementation of the proposed architecture has been developed; the operation of the first implementation of the architecture is to estimate the two form parameters of the input superquadric by letting the other superquadric parameters fixed. The architecture has been implemented in C language on a system HP9000/825SRX with HP-UX, by using the HP Starbase routines for the graphic interface.

The BCS block receives as input a 512x512x8 grey-level image; its output is the boundary webs map of the input images made up by a 32x32 units map. Each BCS output unit is characterised by the vector $[\alpha \ \sigma]$, where α is the medium orientation angle measure of the local boundary contour and σ is the strength of the activation of the unit.

As previously stated, the BCS internal parameters have been fixed from outside in order to make the generated boundaries well follow the surface shape of the superquadric. The shape of a superquadric is in fact generally smooth but near the edges may present some sharp cusps. A precise description of the mathematical details of the implemented BCS along with the adopted choice of internal parameters may be found in [Pirrone].

The output of the BCS stage feeds input to the cascaded stage, which is the backpropagation neural network. More in detail, a 10x10 window of the boundary webs map is presented as input of the backpropagation architecture. The backpropagation input stage is then made up by 10x10x2 units and the output stage is made up by 2 units, one for each form parameter to estimate.

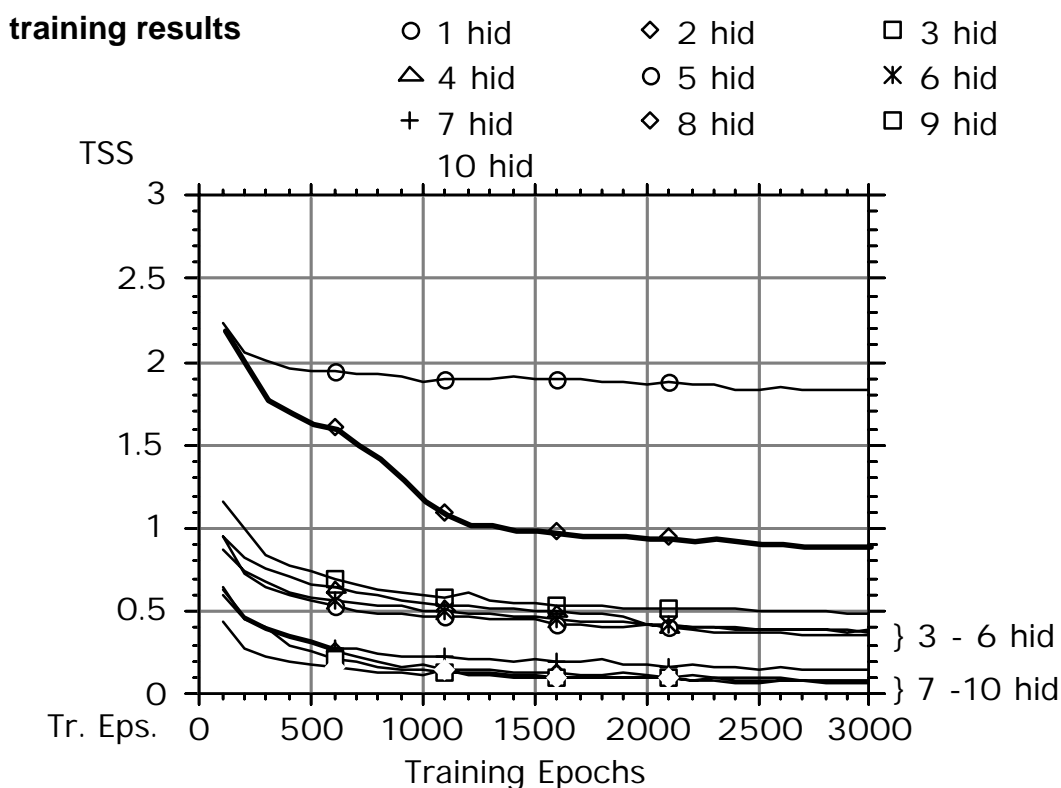
A training set and a test set were built up in order to find a good choice of the internal parameters of the network, such as the number of hidden units, the learning rate, the momentum, and the number of learning epochs. Both the training set and the test set were built up from several synthesised images representing shaded superquadrics of several forms and from several point of view, generated in a Cartesian reference system with origin at the screen centre and x-axis pointing to the outside. The generated superquadrics are centred in [0,0,0], their normalised sides along x,y,z are [0.5 1 0.7], the latitude of the point of view is fixed at 60° while the longitude is in the range [0°–300°]; both the form factors to recover are in the range [0.2 – 2].

The training set is built by selecting for each superquadric image 6 regions of the corresponding boundary webs map: two region roughly cover the centre of the image, while the other ones cover the corners; in fact the corners are generally characterised by a strong brightness gradient, corresponding to a sharp change of curvature and on the contrary the centre of the image is characterised by a smooth change of curvature. The training set size was then made up by 1700 elements. The test set was on the contrary built up by selecting for each superquadric image a random window of the boundary contour map and it was made up by 300 elements not belonging to the training set.

The TSS (Total Sum of Squared Errors) has been used as a measure of the performance of the network. First simulations were carried out with a smaller training set in order to find a preliminary value of the learning rate and the momentum value. According to [Rumelhart 1986], all the best results have been found by using the learning rate value of 0.2 and the momentum value of 0.9.

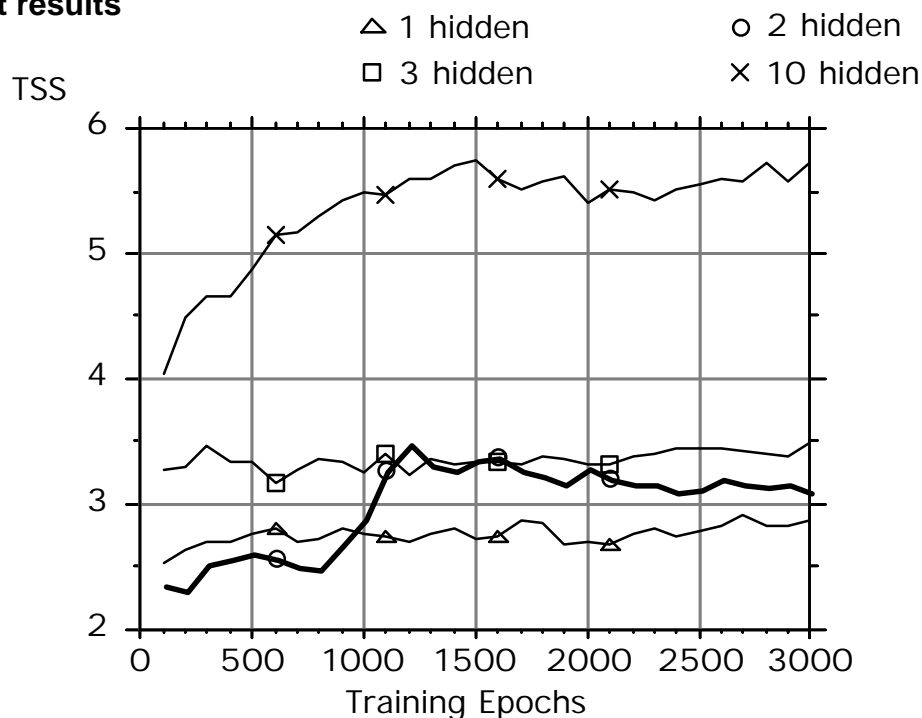
Further simulations have been performed by using the whole training set in order to find the best number of hidden units and learning epochs. Fig. 4 shows the TSS measure for

the training set vs. the number of learning epochs when the hidden units of the architecture varies in the range 1–10.



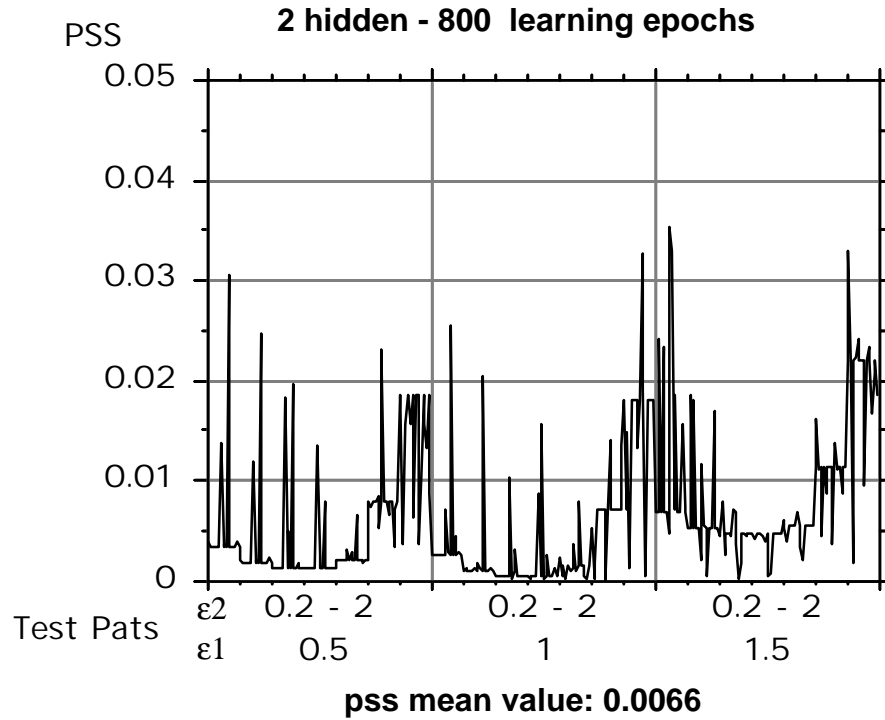
It should be noted ,according to theory, the longer the training time and the more the hidden units, the better are the results, i.e. the better the network learns the training set.

Fig. 5 shows the performances of the architectures made up by 1,2,3 and 10 hidden units with respect to the test set.

test results

It should be noted the well-known overlearning problem: when the number of hidden units and the learning time of the network increase, the network improves its performances with respect to the training set, but it is less able to generalise: i.e. to correctly operate on patterns not previously presented during the learning phase. The same problem arise when augmenting the hidden units of the network, as it should be noted in the figure. For these reasons the architecture made up by 2 hidden units trained with 800 learning epochs has been adopted; fig 5 shows also the training and test behaviour of the chosen architecture as bolded lines: it should be noted that the minimum TSS of the test set is reached at about 800 learning epoches.

Fig. 6 shows the PSS vs. the form factor ϵ_2 , when the form factor ϵ_1 is equal to 0.5, 1, 1.5.



The fig. explains how the error is distributed along the training set: the network better operates when both the form factors are round 1, while a sufficient learning level is achieved for very low ϵ_2 values and the worst learning is obtained when ϵ_2 reaches values more greater than 1. This is due to the fact that the BCS better operates when the superquadric shape is smooth. When the surface present cusps the BCS is less able to follow the sharp brightness gradient, allowing for some degradation of the performances of the whole architecture. Anyway it should be noted that the PSS mean value is very small, thus allowing for a satisfactory form estimate in the whole range of form factor variation.

#####FIGURE DELLE SUP. RICONOSCIUTE#####

Fig. 7 a) b) c) show examples of the reconstruction ability of the architecture when test superquadrics not belonging to the training set are presented to the architecture. In all the figures the left superquadric is the input while the right superquadric is the reconstruction performed by the architecture. It should be noted the satisfactory estimate

operation of the architecture quite independent from the point of view and the form of the starting superquadric.

In order to compare the results of the proposed architecture, the method proposed by Pentland [Pentland 1986] has been implemented in order to recover the form factors from images of shaded superquadrics. Briefly, the method is based on the extraction of the needle diagram from the shaded image; then the superquadric best fitting the needle diagram is found by performing a linear regression operation. In order to obtain the needle diagram of the image the shape from shading method proposed by Horn [Horn 1986] has been implemented; a linear regression method starting from the result of this shape from shading estimates the form parameters of the superquadric.

Main results are that the implemented neural network architecture shows good estimates in the whole variation range of the form factors while results of the Pentland's method are very good only when form factors are near 1 and poor when the form factors are far from 1. The proposed architecture also shows better invariance when noise is added to the image; the BCS is in fact more invariant with respect to the environmental conditions than the needle diagram. Moreover the backpropagation architecture performs a non-linear regression, as stated in [Hecht-Nielsen 1990], thus allowing for a better estimate than the linear regression which is a too weak approximation.

Conclusions

Only simple scenes made up by a single superquadric have been taken into account in the experimental paradigm. This is a not too serious limit, because the BCS gives a local estimation of an object's surface curvature, so that even in the case of a complex scene, made up by several objects, one has to carry out a series of local estimations, by using various portions of the whole boundary webs, corresponding to various portions of the surface of the objects present in the scene.

The application of the system to the recognition of complex scenes is, at present, in an advanced experimental phase: the proposed architecture will be here integrated with

other neural architectures capable to extract geometric information from the scene by other processes, e.g. by the visual motion process[Ardizzone et al. 1991a], in order to obtain an optimum estimate of the geometric parameters of the objects in the perceived scene.

The results we are obtaining are satisfactory and in agreement with our expectations.

References

- Ardizzone, E., Gaglio, S., Sorbello, F. (1989). Geometric and Conceptual Knowledge Representation within a Generative Model of Visual Perception, *Journal of Intelligent and Robotic Systems*, 2, 381-409.
- Ardizzone, E., Chella, A., Compagno, G., Pirrone, R. (1992). An Efficient Neural Architecture Implementing the Boundary Contour System, *Proc. of ICANN-92*.
- Ardizzone, E., Chella, A., Gaglio, S., Sorbello, F. (1991a), Motion Analysis using the Novelty Filter, *Pattern Recognition Letters* 12, 177-182.
- Ardizzone, E., Chella, A., Gaglio, S., Pirrone, R., Sorbello, F. (1991b), A Neural Architecture for the Estimate of 3-D Shape Parameters, in: Caianiello, E. (ed.): *Parallel Architectures and Neural Networks - Fourth Italian Workshop*, World Scientific Publishers, Singapore (in press).
- Ardizzone, E., Chella, A., Pirrone, R., Sorbello, F. (1991c), A System Based on Neural Architectures for the Reconstruction of 3-D Shapes from Images, in: Ardizzone, E., Gaglio, S., Sorbello, F. (eds.): *Trends in Artificial Intelligence*, Springer Verlag, Berlin (in press).
- Barr, A.H. (1981). Superquadrics and Angle-Preserving Transformations, *IEEE Computer Graphics and Applications*, 1, 11-23.
- Callari, F., Chella, A., Gaglio, S., Pirrone, R. (1992). A New Hybrid Approach to Robot Vision. *Proc. of ICANN-92*.
- Grossberg, S. (1973). Contour enhancement, short-term memory, and constancies in reverberating neural networks, *Studies in Applied Mathematics*, 52, 213-257.
- Grossberg, S., Mingolla, E., (1987). Neural Dynamics of Surface Perception: Boundary Webs, Illuminants, and Shape-from-Shading. *Computer Vision, Graphics and Image Processing*, 37, 116-165.
- Hecht-Nielsen, R. (1990) *Neurocomputing*, Addison-Wesley, Reading, MA, USA.
- Horn, B.K.P. (1986). *Robot Vision*. MIT Press, Cambridge, MA, USA.
- Pentland, A.P., Perceptual Organization And The Representation Of Natural Form, *Artificial Intelligence*, 28, 293-331, 1986.
- Rumelhart, D.E., Hinton, G.E. & Williams, R.J., Learning Internal Representations by Error Propagation, in: Rumelhart, D. E., McClelland, J. L. (ed.s) & PDP Research Group (1986), *Parallel Distributed Processing*, Vol 1, MIT Press, Cambridge, MA, USA.

Figure Captions

FIGURE 1: Typical superquadrics shapes. varying the value of the form factors it is possible to obtain different shapes: low values correspond to squared ones, while high values (more than 2) correspond to pinched objects; the spherical shape corresponds to setting both parameters to 1.

FIGURE 2: A general neural network architecture: units in each layer are fully connected with the above and the lower layers and they are not connected to each other in the same layer.

FIGURE 3: The proposed system structure. The BCS filters the input image in order to obtain a brightness gradient map (boundary web) that is supplied as input to the backpropagation architecture which extracts an estimate of the shape parameters of the objects displayed in the scene.

FIGURE 4: The TSS distribution for the training set over 3000 epochs and ranging from 0 to 10 hidden units.

FIGURE 5: The TSS distribution for the test set over 3000 epochs relative to the architectures with 1, 2, 3 and 10 hidden units.

FIGURE 6: The PSS over the test set by varying both e_1 and e_2 . relative to the architecture with 2 hidden units trained for 800 epochs.

FIGURE 7 a) b) c) Examples of the operations of the proposed architecture: in all the pictures the top left superquadric is the input to the network while the bottom right superquadric is the reconstruction performed by the architecture.

