Experiments in Fluids 19 (1995) 225–232 © Springer-Verlag 1995

Recognition of partially overlapped particle images using the kohonen neural network

F. Carosone, A. Cenedese, G. Querzoli

Abstract A neural network is proposed for the recognition of partially overlapped particle images in the analysis of Particle Tracking Velocimetry (PTV) frames. The Kohonen neural network is an approximation to an *optimum* classifier. In this work it allows single particle images to be distinguished from overlapped particle images by shape analysis: it classifies 99.1% of the spots correctly (in test images). If a spot has an almost circular shape, the barycenter co-ordinates are extracted. If the spot shape is far from being circular, it is believed to be a particle overlap, and a procedure to find more centroids is activated.

The particle recognizer based on the Kohonen neural network is tested on both multi-exposed and single-exposure images at high particle density, and compared to a particle recognizer that did not consider the partial overlap. The management of overlapped particles causes the neural network to produce a big improvement in the number of barycenters that can be extracted from these images. The practical consequence is that the seeding density in PTV can be increased, so as to improve the spatial resolution of the technique in the velocity field calculation.

1
The overlapping particle problem

In PTV images, particles usually look like bright rounded spots on a dark background. In order to calculate the velocity field, the barycenters of every particle image must be found.

Figure 1 shows examples of PTV images: not all bright spots are always rounded. Sometimes, more complex shapes appear, produced by the partial overlap of two or more particle images. This happens mainly in two situations:

a) *2D multi-exposed frames* (Fig. 1a). The flow field is illuminated by a light sheet at regular time intervals, and a number of consequent exposures are superimposed on the same recording frame. Here, two spots represent likely subsequent positions of one particle. Hence, overlaps appear whenever a particle has a small displacement (in the time interval between two consecutive light shots).

b) *stereoscopic PTV*. The whole volume is illuminated, and viewed through multiple cameras. In single-exposure frames (Fig. 1b), every bright spot represents a different particle. However, overlaps may still occur. As a matter of fact, every camera records a projection of the field onto a plane. Thus, for high seeding density (required for good spatial resolution), particles belonging to different parallel planes, orthogonal to the projection plane, overlap their images if they have similar co-ordinates in the plane. If the viewing angle between the cameras is low, there are less overlaps, but information about the third velocity component (in the depth direction) is reduced.

The partial overlap of particle images constitutes a problem for a particle recognition software, since more than one centroid has to be looked for in the spot (i.e. two centroids for a "double overlap", three centroids for a "triple" overlap). In fact, most particle recognition softwares look for connected sets of lighted pixels. A partial overlap of two or more particle images would be taken as one particle by mistake: one centroid would be produced. One would not find the right centroids, and would also introduce a false result. Guezennec and Kiritsis (1990), and Sata and Kasagi (1992) both emphasized the particle overlap as a vital problem for 3-D PTV at high particle concentrations.

Various authors proposed different solutions to the problem [Mass et al. (1993), Guezennec et al. (1994)]. Performing standard techniques of digital image processing on all spots (such as boundary erosion and dilation, or template matching) is not necessary, since only overlaps have to be processed this way. It is more convenient to compute some simple features for every spot (e.g. area, perimeter, aspect ratio), which allow to discriminate single particles from overlaps, and then perform e.g. erosion on the last only.

In order to overcome some gaps of the current state-of-the-art:

1. a large number of spot features were methodically examined and compared on a rigorous basis, in order to select the best ones;
2. a minimum error classifier (the Kohonen neural network) distinguished single particles from overlaps on the basis of the selected features. It is to be noted that a classification based on a combined use of more features produces less mistakes than using one feature, but requires much more computational

Received: 5 July 1994/Accepted: 8 March 1995

F. Carosone, A. Cenedese, G. Querzoli
Mechanics and Aeronautics Department,
University of Rome "La Sapienza"
V. Eudossiana 18, I-00184 Rome, Italy

Marcello Sallusti and Paolo Monti produced the synthetic PTV images that were used in this project. Gianni Leuzzi helped clearing the statistic properties of the neural network through infinite discussions. We would like to thank them all, since we are aware that they all gave a decisive contribute to the project.

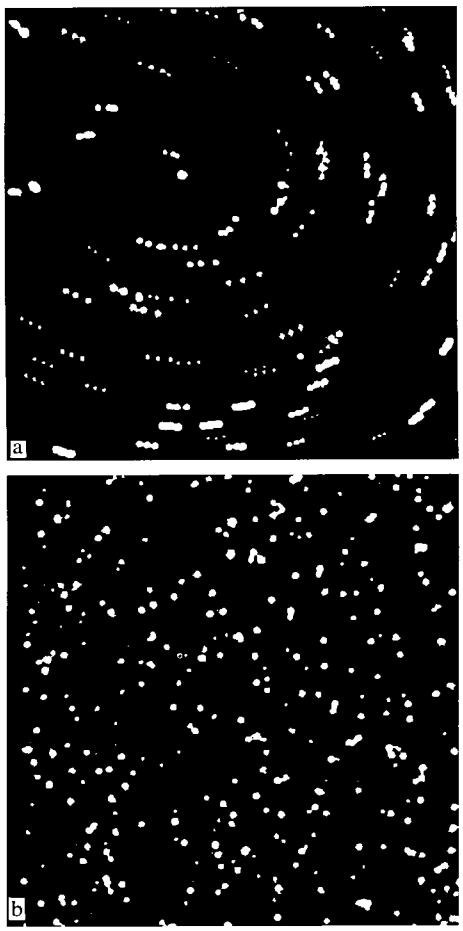

Fig. 1a, b. Multi-exposed PTV image a and single exposure image at high concentration b

effort to comply with Bayes' decision theory [Papoulis (1965)]. The Kohonen neural network is a very fast approximation of the ideal *Bayes' classifier* [Kosko (1992)] and provides an automatic tuning to optimal classification;
3. boundary erosion, controlled by the neural classifier, was applied to the spots classified as overlaps.

The final product is a particle recognition software that outputs all particle barycenters in a PTV frame, regardless of partial overlaps.

2
The Kohonen neural network

2.1
Unsupervised neural networks for signal processing

Artificial neural networks (ANN) are a branch of Artificial Intelligence (A.I.). They have been applied successfully to signal processing, pattern recognition and computer vision, during the last years [Kabrisky and Rogers (1991)]. Neural networks have an inherently parallel conception, hence their use can be optimized on a parallel machine. In these cases ANN may lead to better performances, in terms of computing speed, with respect to other algorithms.

The inspiration for neural architectures comes from the brain of higher animals. Neural networks consist of a number of relatively simple processing units (called 'neurons' like the biological prototype), with a high degree of connectivity among them. A flexible connection (called 'synapse') is established among the units, modelled through a variable 'weight' associated to it. The basic consequence of the synaptic flexibility is that a neural network is eventually able to *learn from examples*. Some neurons work as sensors that receive the external signals to be processed: as a response to the arriving inputs, the linking weights of all synapses in the net are modified according to the so called *learning rule*. Starting from a random weight initialization, the net 'learns' to act as a signal processor, whose transfer function is obtained through modification of its inner structure.

Neural networks are usually divided in two categories [Wassermann (1989)]: *supervised* and *unsupervised* networks. The backpropagation networks and the Hopfield network are supervised, while the ART network and the Kohonen network, used in this work, are unsupervised. Input data to a supervised network have to be provided with a label indicating the correct output, while input data to an unsupervised network are unlabelled: this makes the learning of unsupervised networks much easier and faster for the operator. Moreover, unsupervised networks are self-adapting, in that they change their behavior with time if input signals are functions of time: this makes them suitable for control [Hecht-Nielsen (1990)].

2.2
Network architecture, neuron model and learning rule

A Kohonen network with the Desieno correction was used in this work: a detailed explanation of the Kohonen network may be found in many texts on ANN [Lippmann (1987, 1989), Hecht-Nielsen (1990)]. For the Desieno correction, Kabrisky and Rogers (1991) were followed.

The network architecture is shown in Fig. 2. The N circles below ($N=2$ in Fig. 2) represent the sensory nodes through which a N-component vector $X(X_1, \ldots, X_n, \ldots, X_N)$ feeds the network. The sensors constitute an 'input layer', that receives information in parallel and distributes it to the neurons above. Let m be a neuron in the second layer ($m=1, \ldots, M$), and W_{nm} denote the synaptic weight of the link from sensor n to neuron m. Weights have a random initialization, as in all ANN. In this work, both W_{mn} and X_n were continuous real values in the range [0, 1].

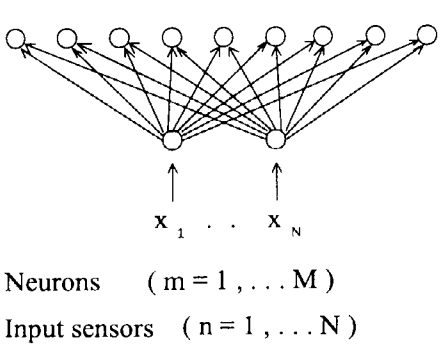

Fig. 2. Network architecture

The neuron output value OUT_m, called the neuron "activity", is given by:

$$OUT_m = \sqrt{\sum_n (W_{nm} - X_n)^2} \qquad [1]$$

Then a competition occurs among all neurons, to select the one p with the minimum activity:

$$p: \quad OUT_p = \min_m \{OUT_m\} \qquad [2]$$

The synaptic weights from all input sensors towards the selected neuron and its neighbors are modified according to the following learning rule:

$$W_{nm,\,\text{new}} = W_{nm,\,\text{old}} + \eta \cdot (X_n - W_{nm,\,\text{old}}) \quad n = 1, \ldots, N \qquad [3]$$

where η is a scalar named "learning parameter". As more examples are presented to the net, η slowly decreased from its initial value (usually in the range [0.1, 1]) and vanishes. In fact, η controls the net flexibility. Note that no indication about the correct output appears in Eq. [3], since the Kohonen learning rule, as above mentioned, is unsupervised.

2.3
Geometric interpretation

A weight vector $\boldsymbol{W}_m$ can be defined for every neuron, whose components are the synaptic weights from all input sensors to that neuron:

$$\boldsymbol{W}_m := (W_{1m}, \ldots, W_{nm}, \ldots, W_{Nrr}). \qquad [4]$$

When $N=2$, both the input vector $\boldsymbol{X}$ and the weight vectors $\boldsymbol{W}_m$ are two-dimensional vectors, represented by points in a plane. Thus, a position in this plane can be associated to every neuron, corresponding to the co-ordinates of its weight vector (Fig. 3). In particular, in this work, neurons were represented by points in the square $D = [0, 1] \times [0, 1]$.

The black segments between the neurons on the plane in Fig. 3 do not correspond to any synapse in the net; they just indicate neighboring neurons in the net topology shown in Fig. 2.

When an input $\boldsymbol{X}$ is presented to the net, a point $\boldsymbol{X}$ may be thought to 'fall' on the plane: Eqs. [1] and [2] select the neuron whose Euclidean distance from $\boldsymbol{X}$ is minimum (*nearest neighbor*, Fig. 4a). Then Eq. [3] moves the selected neuron and its neighbors towards the input point along the vector difference $(\boldsymbol{X} - \boldsymbol{W}_m)$ of a fraction η (Fig. 4b).

In this geometric representation, neurons look like a "snake" (the *Kohonen "snake"*) moving around in the square D, towards the input data.

2.4
Representation of multivariate probability density functions

It was shown [Kohonen (1990)] that the net is able to calculate the multivariate probability density function (pdf) of the input data.

Let a set of input vectors $\boldsymbol{X}$ be characterized by an underlying pdf. If they are offered, one by one, as input examples to the net, the Kohonen snake moves around at each example. After presentation of the whole set, weight modification in the net is not allowed any longer: the position of the snake in the square is 'frozen'. Then the input vectors are presented again to the net: for each one, the closest neuron

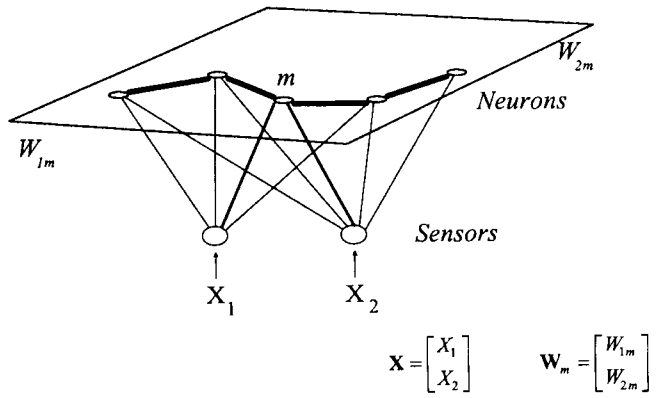

Fig. 3. Geometric interpretation for $N=2$

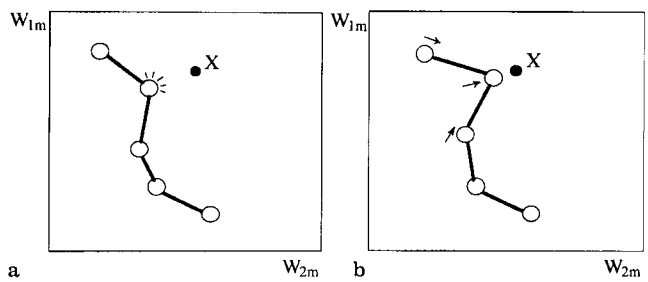

Fig. 4a,b. Geometric interpretation of learning rule

in the plane is selected by Eqs. [1] and [2]. The basic property of the Kohonen network (with the Desieno correction) is that the frozen position of the snake is such that all neurons in the net tend to be selected with the same frequency.

Let us denote by D_m the region of points in the square for which neuron m is closest than any other neuron: if an input vector $\boldsymbol{X}$ falls in D_m, it causes neuron m to be selected. For any given position of the snake, D turns out to be divided into M regions $(D_1, \ldots, D_m, \ldots, D_M)$ of different size and shape, each corresponding to one neuron. The above mentioned property can be reformulated as follows: every region 'captures' input vectors with the same frequency, hence, conversely, the probability of the input data must be equal in every region. The equal probability method of dividing the input domain is one of the ways to code a multivariate probability density function.

Figure 5 shows some demonstrative examples. A random initialization of the synaptic weights for a 16 neurons network took place with a uniform density in the square $[0.45, 0.55] \times [0.45, 0.55]$ (Fig. 5a). A set of 1000 vectors, with a uniform pdf in the square D, fed the network. The position of the snake after these 1000 examples is illustrated in Fig. 5b: the snake tends to approximate the Peano curve, that would divide D in 16 equiprobable regions (identical in this case). After that, another set of 1000 vectors were presented to the net. This time the underlying pdf was the sum of two Gaussian functions, whose mean values were respectively $M_1(0.2, 0.8)$ and $M_2(0.8, 0.2)$, and all deviations equal to 0.1. The final position of the snake is in Fig. 5c it tends to approximate two ellipses (circles in this case) whose centers are M_1 and M_2, and

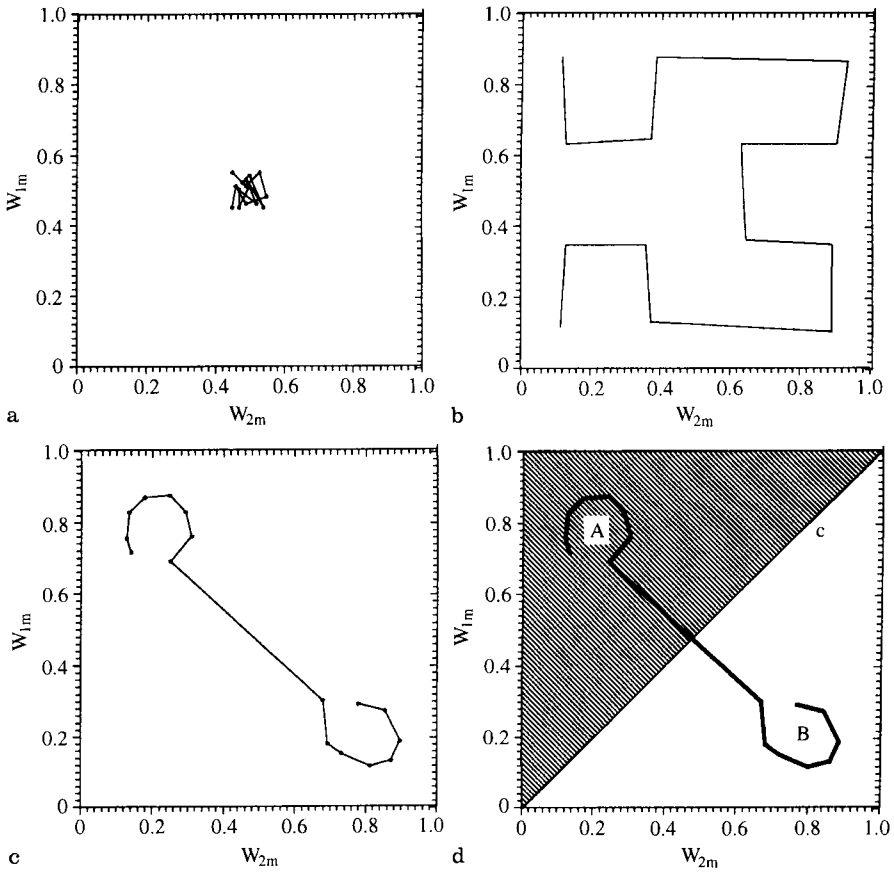

Fig. 5a–d. Demonstrations of the network mathematical properties

whose axes are related to the deviations. These results show that the net is a flexible architecture that adapts itself to the arriving input data.

2.5
Pattern recognition

The net capability of forming an accurate, yet compact, model of the multivariate pdf underlying an input vector set, is of great value for pattern recognition tasks. A pattern is usually described by a few suitable features [Cotterill and Oja (1990)]. The measured values of these features constitute a vector. Similarity between patterns is thus reduced to Euclidean distance in a vector space: two patterns are said to be similar if the corresponding feature vectors are close in this space. Recognition is a classification: a group of similar vectors form a *cluster*, representative of one class; and a new is said to be recognized when the corresponding feature vector is close to the cluster. Once recognized, the new pattern is incorporated into the cluster.

The Kohonen network may be regarded as a clustering technique [Kohonen (1990)] suitable for automatic pattern recognition. Suppose we have a two-class problem, where each class is represented by a multivariate Gaussian probability density function. This happens, for instance, when two ideal patterns (A, B) should have been produced, but only a large number of imperfect samples of both are available. Let all feature vectors feed a 16 neuron Kohonen neural network: in the end, the position of the Kohonen snake is of the same kind as in Fig. 5c. The net is then 'frozen'. The first eight neurons are associated to class A, while the last eight neurons are associated to class B. When a new feature vector X is given as an input to the net, representing an imperfect sample of one of the two ideal patterns (say, A), the activity of every neuron is calculated according to Eq. [1], and the neuron with the minimum activity is selected as in Eq. [2]. The feature vector is presumed to belong to the class to which the selected neuron was associated, and the new sample pattern is assumed to be an imperfect version of the ideal pattern corresponding to that class (i.e. A).

This procedure, though simple and fast, is an approximation of a complex theory for pattern classification, that is the Bayes' decision theory [Papoulis (1965), Kosko (1992)]. A geometrical interpretation may help (Fig. 5d). The grouping of eight neurons, representing one class, yields a subdivision of the input domain D in two regions, each corresponding to a class, and formed by the union of eight equiprobable domains D_m. A *decision boundary* is produced between the two classes. If one had to find the Bayes' decision border by an algorithm, the two expected class prior pdf should be modelled as Gaussian functions, and an iterative least-squares approximation should be performed to find the Gaussian parameters (μ, σ for each class) for fitting the input data probability density. Moreover, the decision border calculation would lead to a non-linear set of equations of nasty solution. Using the Kohonen neural network, pattern recognition has two main advantages:

1. modelling the densities through Gaussian functions, though reasonable in many cases, is arbitrary. The neural network, on

the contrary, is non-parametric (that is, independent of any assumption on data distribution);
2. the neural equations (Eqs. [1], [2], and [3]) are very light, hence the decision boundary is obtained at a computational cost which is much lower than an algorithm of Bayesian approximation.

3
A neural particle recognizer

In the present project, the input to the particle recognizer was a 512×512 pixel image, where each pixel had a grey level in the range 0–255. The output was the co-ordinates of all particle barycenters, including those of partially overlapped particle images (named '*overlaps*' hereafter). The input image is first threshold. The binarized image shows a number of white spots on a black background. Every spot, defined as a 8-connected domain of white pixels, is detected and sent to a (neural) shape recognition module.

The basic idea is that a spot may represent either a particle or an overlap. If it is recognized to be a particle image, its barycenter co-ordinates are calculated and produced as outputs. If it is recognized to be an overlap, the spot is eroded until it splits into more component particles: the barycenter calculations are then reduced to the previous case. The spot shrinking is performed through boundary erosion controlled by the shape recognition module. In fact erosion tends to separate the single-particle images that constitute an overlap.

The neural particle recognizer works with shape, and is therefore insensitive to the size of particle images in a PTV frame. The Kohonen neural network is the shape recognition module, that distinguishes the rounded spots (single-particle images) from differently shaped spots (overlaps). For each spot, a set of geometrical features is calculated, which describe its shape [Parker (1994)]. The measured values of these features constitute a vector, that is the actual input to the neural network. Two vector classes are defined: "particles" and "overlaps". A particle (*overlap*) is said to be recognized when the corresponding shape feature vector matches the "particle" (*overlap*) class.

4
Tuning the shape recognizer

Six synthetic images were prepared to constitute the examples for the neural network. They contained single particles and overlaps in equal number. One of these is shown in Fig. 6. Only double overlaps (i.e. two overlapping particle images) were considered: extension to multiple overlaps will be discussed below. The total number of spots in all test images was 228: this set will be named 'test set'. The overlapping particle images in the test set were chosen to represent a wide distribution of different degrees of overlap, orientation and size. Good care must be taken in choosing the examples of the test set: if the test set is representative of all possible situations in PTV images, then if the neural network recognizes these, it should be able to distinguish them also in images never seen.

4.1
Shape features

A set of candidate features were calculated and actually examined for every spot of the test set. Shape features must be

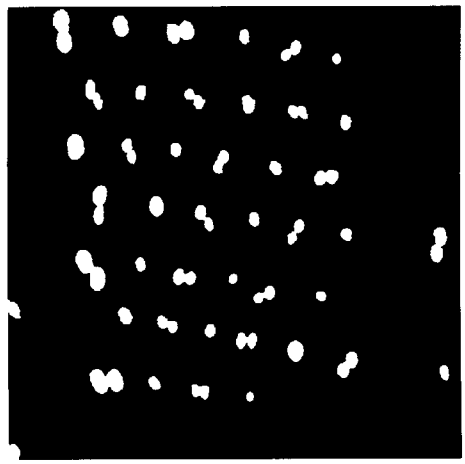

Fig. 6. Test image

invariant on translation, rotation and scaling. Besides, for convenience, all features were defined in the range [0, 1]. A complete list would be too long, therefore only the best five candidate features are reported here:

1. First circularity measure (crf_1). A direct circularity measure comes from the polar equation $r(\varphi)$ of the spot boundary, where r is the distance from the barycenter to the boundary, and φ is the polar angle. A measure of distance from a circumference (for which $r(\varphi) = r_{\text{med}}, \forall\varphi$) is given by:

$$crf_1 = \frac{\int_0^{2\pi} (r(\varphi) - r_{\text{med}})^2 \, d\varphi}{\int_0^{2\pi} r_{\text{med}}^2 \, d\varphi} \qquad [5]$$

where r_{med} is the average radius. $crf_1 = 0$ for a circular shape.

2. Second circularity measure (crf_2). It is the ratio between spot area and the smallest external circumference (whose radius is $r_{\max} = \max\{r(\varphi)\}$):

$$crf_2 = \frac{Area}{\pi \cdot r_{\max}^2} \qquad [6]$$

that yields $crf_2 = 1$ if the spot is a circle.

3. Third circularity measure (crf_3). In a circle, the ratio between area and squared perimeter p^2 is $1/(4\pi)$. This value is greater than any other plane figure, hence

$$crf_3 = \frac{4\pi \cdot Area}{p^2} \qquad [7]$$

is in [0, 1] and equals 1 for a circle.

4. Aspect Ratio (AR). The ratio between the minor and major axis length of an equivalent ellipse, (i.e. an ellipse having the same second moments as the spot):

$$AR = \frac{minor\ axis\ length}{major\ axis\ length} \qquad [8]$$

Eq. [8] produces $AR = 1$ for a circle.

5. Convexity (cnv). The smallest convex region enclosing the spot is first found. Then cnv is defined as the ratio between

Table 1

crf_1 and AR	88.1%
crf_1 and crf_2	92.1%
crf_1 and crf_3	94.6%
crf_3 and cnv	99.1%

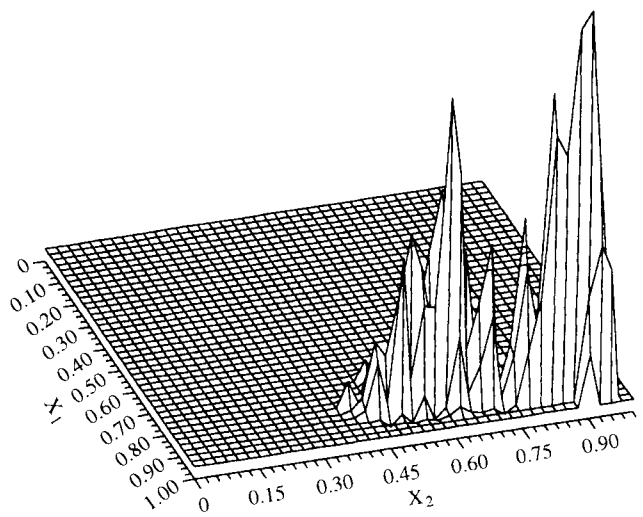

Fig. 7. Joint-probability of the test set

the two perimeters:

$$cnv = \frac{smallest\ enclosing\ convex\ perimeter}{spot\ perimeter} \quad [9]$$

$cnv = 1$ for any convex figure, e.g. a circle.

4.2
Single particle and overlap recognition

The feature values, measured for every spot in the test set, fed a sixteen neuron Kohonen network. Then the first eight neurons were associated to the single particle class, and the last eight to the overlap class. The synaptic weights were frozen, and the test set was presented again to the net for classification. Since the right classification was known for the test images, the percentage of correct recognition could be computed to compare the discriminant power of different features.

First, the candidate features were tested one by one. No shape feature reaches optimal results alone. The best feature was found to be crf_3, i.e. the circularity measure based on area and perimeter. Classification over the test set, based on this feature, achieved 94.6% of correct recognition.

Then different combinations of more features were tested. Combining the above five features two by two, ten couples were formed. Once a couple of features had been selected, the values of this couple were measured for all the spots in the test set, and given to the network. The number of correct classifications was then found for every couple: the percentages are listed in Table 1 for four couples of increasing accuracy.

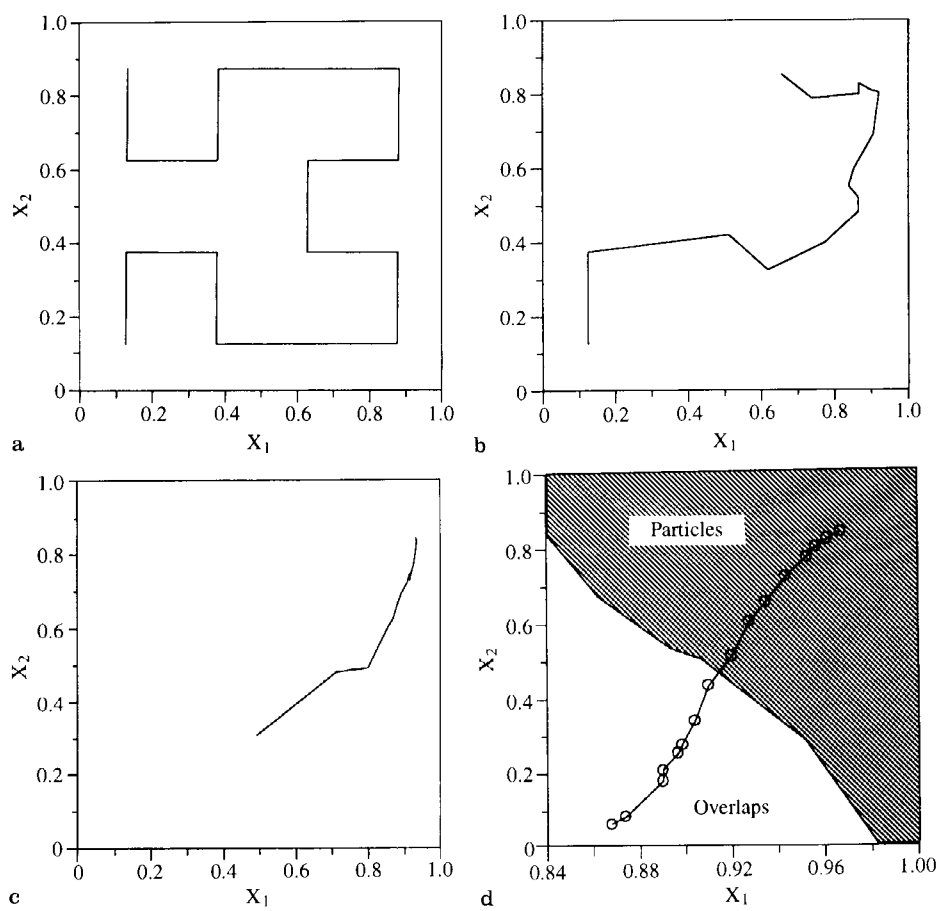

Fig. 8a–d. Kohonen *snake* after 0 **a**, 10 **b**, 70 **c** and detail after 456 **d** examples

Sometimes, adding a second feature, no better performance was reached (see crf_1 and crf_3 in Table 1): however, two joint features were more effective in most cases. The best couple was crf_3 and *cnv*, i.e., respectively, the circularity measure based on area and perimeter, and the convexity measure.

Figure 7 shows the joint-probability density $p(X_1, X_2)$ for the test set (calculated for check), with the two shape features X_1, X_2 being respectively *cnv* and crf_3. The test set was repeated twice, so the total number of examples was 456. The initial value of the learning parameter in [3] was $\eta = 0.3$. The weight initialization was the Peano curve with 16 nodes for the square D (Fig. 8a). The Kohonen snake moved in the square as in Fig. 8. The final position and the decision boundary are zoomed in Fig. 8d.

Since the neural network achieved, with the selected features, 99.1% correct recognition over the test set, and joint use of three features was tried with no better results, the problem of discriminating single particles from overlapping particle images was considered to have been given an optimal solution.

4.3
Outline of sensitivity analysis

The number of neurons was first varied (all the rest held constant). Good results (97.6%) are obtainable with two neurons only; the correct recognition percentage was monotonically increasing with the number of neurons and reached the optimal behavior with sixteen neurons.

In general, the more examples are presented to a neural network, the better it 'learns' how to perform a task. Yet, some neural networks are faster than others, and this property gives a measure of the network flexibility. The learning of the decision boundary, for the overlapping particle problem, was monitored stopping the synaptic weight change after a number *nex* of examples and testing the recognition accuracy over the whole test set. At $nex = 70$ already 93.3% correct recognition was reached, which witnesses the big adaptivity of the Kohonen network (compared to other neural networks, that need thousands of examples to get similar accuracies). After $nex = 100$ the curve of correct recognition versus *nex* was almost flat, yet 456 examples were needed to get the optimal performance.

The dependence of many neural networks on some obscure internal parameters has often been criticized. Kohonen himself stressed that the choice of the initial learning parameter η (in Eq. [3]) is not critical [Kohonen (1990)]. In fact, in our study, the network behavior was quite insensitive to changes in η over a rather wide range. The percentage of correct recognition versus η, is flat for $\eta \in (0.1, 0.8)$.

5
Extension to multiple overlaps

Multiple overlaps can be successfully treated by applying the recognizing procedure recursively to the eroded spots that have not been classified as particles. Doubts may arise about the network capability to manage triple overlaps (i.e. composed of three partially overlapped particle images) or more, since it was trained by means of images that presented double overlaps only (Fig. 6). When a multiple overlap is presented to the net, the two chosen features give values that make it classified as an overlap anyway (because it is still more different from a circular and convex shape than a double overlap is).

6
Results and limits of the neural particle recognizer

Synthetic images are necessary to test and compare different particle recognition softwares, since barycenters are known in advance, and their position can be used as an exact reference. The recognition performance of the neural particle recognizer was compared to that of a previous non neural version, that did not extract barycenters from overlaps. The previous version considered every 8-connected set of white pixels to be a particle image, hence discarded all spots having area larger than a threshold, suspected to be overlaps, because the barycenter calculation of these would have brought errors to the code. As a visual demonstration, the particle recognition process, performed on the image of Fig. 1a by the previous recognizer (Fig. 9a) and by the neural recognizer (Fig. 9b), is shown as a cross put on the barycenter of every recognized particle.

Two sets of 512×512 synthetic PTV images were produced. The first set consisted of multi-exposed images containing particles with increasing average area. Three instants of time

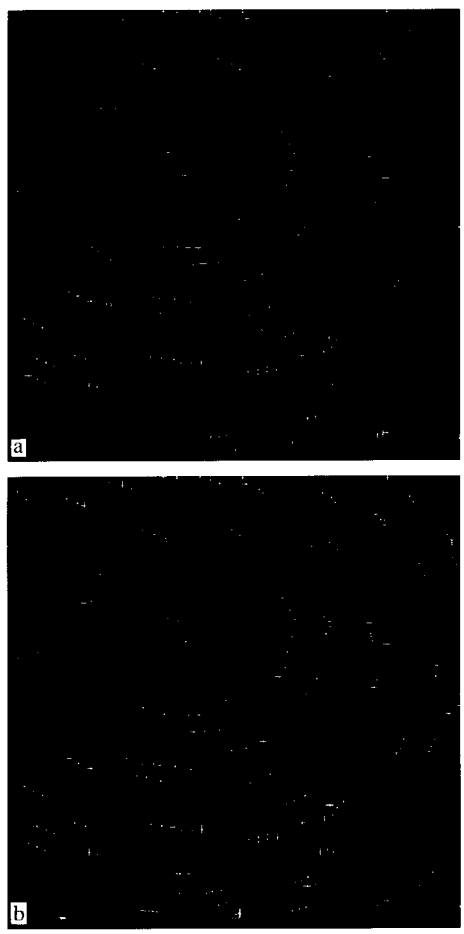

Fig. 9a, b. Previous **a** and new **b** particle recognition over the image of Fig. 1a

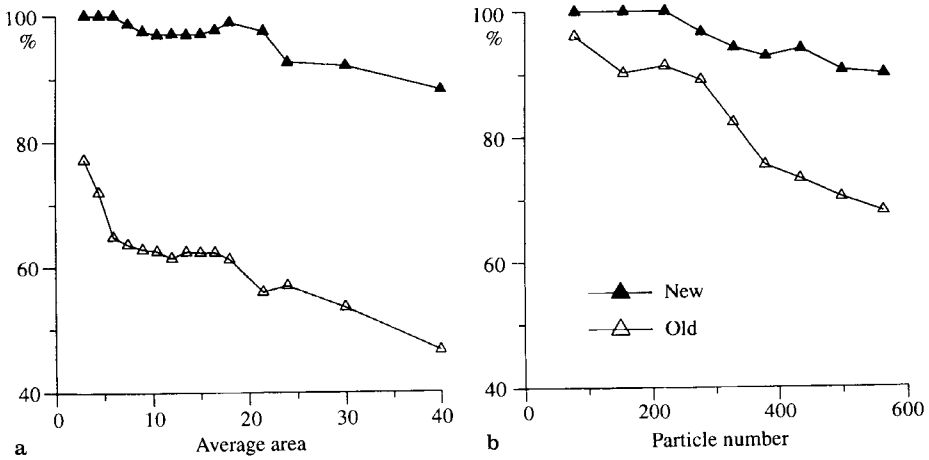

Fig. 10a, b. Correct barycenter recognition %v. average area **a** and number of particles **b**

were superimposed. The number of particles ($\sim 200 \times 3$) and their mean displacement (7 pixels) were kept constant. One of these images was shown in Fig. 1a. As area increases, the number of overlaps is expected to increase as well. Although area is an indirect parameter to observe overlaps, it is widely used experimentally [see also Guezennec and Kiritsis (1990)]. Recognition accuracy, expressed in percent of identified barycenters, is plotted versus average particle area in Fig. 10a for the previous and the new particle recognizer.

The second set consisted of single-exposure images at increasing particle concentration. Average area was kept constant and equal to 12 pixels. One of these images was shown in Fig. 1b. The number of overlaps increases as the number of particles. The percent of barycenters identified by both particle recognizers is plotted in Fig. 10b versus the number of particles.

In all cases, the management of overlapped particle causes the neural recognizer to be more accurate and far more robust than the previous recognizer.

The neural particle recognizer is not recommended in the following cases:

1. images containing a small number of overlaps. In fact, even if the neural network is extremely light computationally, the feature calculations for every spot slow down the particle recognizer, hence its use on images with few overlaps would not be worthwhile.

2. images with low average particle area. In fact, the shape feature extraction is not accurate for every small spots: the numerical error is such that these spots are considered as particle images anyway. Recognition might be uncorrect if the average particle area is less than 7 pixels.

7 Conclusion

The neural-based particle recognizer is able to manage partial overlaps of particle images in the PTV technique. The software produces a sharp increase in the number of identified barycenters in PTV images with many overlaps.

The practical significance is a big benefit to the spatial resolution of the technique, since the seeding density can be increased. This property may be of great help in the analysis of both multi-exposed and stereoscopic PTV images. The time interval between two consequent shots in the multi-exposed technique can be made shorter, and thus the dynamic range be increased. In the stereoscopic technique, an orthogonal view can be adopted, so that full information about a 3-D velocity field can be recovered with no need of redundancies in the number of video cameras.

References

Cotterill R; Oja E (1990) Neural computing and pattern recognition, Courses in advanced technology, CEI-Europe/Elsevier

Guezennec YG; Kiritsis N (1990) Statistical investigation of errors in Particle Image Velocimetry, Exp Fluids 10: 138–146

Guezennec YG; Brodkey RS; Trigui N; Kent JC (1994) Algorithms for fully automated three-dimensional Particle Tracking Velocimetry, Exp Fluids 17: 209–219

Hecht-Nielsen (1990) Neurocomputing, Addison-Wesley

Kabrisky M; Rogers S (1991) An introduction to neural networks for pattern recognition, SPIE press

Kohonen T (1990) The Self-Organizing Map, Proc. IEEE on Neural Networks, September

Kosko B (1992) Neural networks for signal processing, Prentice Hall

Lippmann RP (1987) An introduction to computing with neural nets, IEEE ASSP Magazine, April

Lippmann RP (1989) Pattern Classification using neural networks, IEEE Comm. Magazine, November

Mass HG; Gruen A; Papantoniou D (1993) Particle Tracking Velocimetry in three-dimensional flows – part 1. Photogrammetric determination of particle coordinates. Exp Fluids 15: 133–146

Papoulis A (1965) Probability, random variables, and stochastic processes, McGraw-Hill

Parker J (1994) Practical computer vision using C, Wiley and Sons

Sata Y; Kasagi N (1992) Improvement toward high measurement resolution in three-dimensional particle tracking velocimetry, Proc 6th Int Symp on Flow Visualization, Yokohama, Springer-Verlag

Wassermann PD (1989) Neural Computing, Van Nostrand-Reinhold