TUTA/IOE/PCU

Journal of the Institute of Engineering, Vol. 7, No. 1, pp. 1-18
© TUTA/IOE/PCU
All rights reserved. Printed in Nepal
Fax: 977-1-5525830

# IMPROVEMENT OF ALGORITHM IN THE PARTICLE TRACKING VELOCIMETRY USING SELF-ORGANIZING MAPS

**Er. Shashidhar Ram Joshi**
Prof. Electronics and Computer Engineering Department, Pulchowk Campus, Institute of Engineering, Tribhuvan University
E-mail: sashi@healthnet.org.np

**Abstract:** The neural network techniques are becoming a useful tool for the particle tracking algorithm of the PIV system software and among others, the self-organizing maps (SOM) model seems to have turned out particularly effective for this purpose. This is mainly because of the performance of the particle tracking itself, capacity of dealing with unpaired particles between two frames and no necessity for a priori knowledge on the flow field (*e.g.* maximum flow rate) to be measured. Initially, concept of SOM was applied to PIV by Labonte. It was modified by Ohmi and further modified algorithm is developed using the concept of Delta-Bar-Delta rule. It is a heuristic algorithm for modifying the learning rate as training progresses. Earlier, the treatment of unpaired particles, a specific problem to any type of PIV, is not fully considered and thereby, the tracking goes unsuccessfully for some particles. The present research is to bring about further improvement and practicability in this promising particle tracking algorithm. The computational complexity can be reduced employing modified algorithm compared to other algorithms. The modified algorithm is tested in the light of the synthetic PIV standard image as well as in particle images obtained from visualization experiments.

**Key words:** Delta-Bar Delta, Dynamic Threshold Binarization, HVD Algorithm, Labonte's SOM, Modified Algorithm, Ohmi's SOM, Particle Image Velocimetry(PIV), Particle Tracking Velocimetry(PTV), Self-Organizing Map(SOM), Single Threshold Binarization.

## 1. INTRODUCTION

Particle tracking velocimetry (PTV) is a multi-disciplinary research field. The spectrum of research includes bio-medical, electrical, electronics, flow visualization, heat flow visualization, and nuclear engineering. The PTV quantifies the displacement of individual dynamic particles between two or more image frames. Several experimental engineers involved in the PIV measurement of fluid flows are well aware of the importance of both extraction and tracking algorithms. The extraction algorithm is for identifying individual particles and detecting their respective centers from the images coded in different gray scale levels. The accuracy of particle tracking relies in the extraction of individual particles.

The scope of application of PTV research is a multi-disciplinary matter. The visualization and image processing of flow around an axial flow fan in the outdoor side of an air conditioner [1] is an example of PTV in the field of mechanical and electrical engineering. PIV measurement of turbo-ram combined engine wake-flow with combustion [2] can be example in the field of aeronautical engineering. The bio-medical application of PTV is flow visualization and evaluation of centrifugal blood pumps for artificial heart [3]. Another example in the same field is PIV studies of heart valve prostheses in pulsatile flow [4]. Other application is the technique for quantitative temperature mapping in liquid by measuring the lifetime of Laser Induced

Phosphorescence [5]. Human beings are extremely interested in the observation of nature, as this was and still is of utmost importance for their survival. Human senses are especially well adapted to recognize moving objects as in many cases they mean eventual danger. One can easily imagine how the observation of moving objects has stimulated first simple experiments with set-ups and tools easily available in nature. Today the same primitive behavior becomes obvious, when small children throw little pieces of wood down from a bridge in a river and observe them floating downstream. Even this simple experimental arrangement allows them to make a rough estimate of the velocity of the running water and to detect structures in the flow such as swirls, wakes behind obstacles in the river, water shoots, etc.

A typical PTV system is comprised of two systems: capturing of image and image analysis. In an image capture system, a pulsed laser is generally used for illumination due to the high energy in each laser pulse. Either a special photographic camera or a CCD video camera can be used as recording medium. Images are formed on a photographic film or a video array detector, and the images are subsequently transferred to a computer for automatic analysis [6].

The preprocessing algorithm is for extracting individual particles and detecting their respective centers from densely distributed particle images [7]. The second one is the particle-identifying algorithm and this enables to quantify the displacement of moving particles between two or more image frames [8].

The Moravec operator [9] was originally implemented in the field of automatic motion analysis and intended for marking any interest points systematically. The operation process consists of three computation steps; the first one is a computation of directional variance over the square mask windows, where sums of squares of differences of pixels adjacent in each of four directions. The application of Moravec operator in PTV was applied by Ohmi and Li [10]. The dynamic threshold binarization [11] method is not a direct computation scheme of particle centres, a conventional binarization using a recursive calculation. The concept of the method is to adjust the threshold level particle by particle in accordance to the mean brightness level of the particle image [12]. The HDV algorithm [13] is based on the elimination of floating particles and light streaks. The HDV elimination range and HDV elimination relaxation for horizontal, vertical and diagonal directions are entered and noises are eliminated. The algorithm is comparatively very fast and finally the image is binarized, but it can be applied to the elimination of suspended particles.

The Genetic Algorithm based PTV was proposed by Ohayama. The Spring Model technique [14] is based on the pattern matching of the particle cluster between the two consecutive images. The particles are assumed to be connected by invisible elastic springs. The Velocity Gradient [15] is based on the gradient of velocity. The local correlation value is iteratively arrived at through successive approximations of local displacement using increasingly smaller regions of determination in the Recursive Local-Correlation method [16]. The Relaxation Method [17] and modified by Lee [18] consists of probabilistic calculation. The Improved Relaxation Method by Ohmi and Lam, [19] considers both matching and non matching probabilities. The Wavelet Vector proposed by Hui [20] is the new concept of matching. These methods require long computational time. The computational time can be reduced using modified self-organizing map algorithms and it will be discussed later.

The Cellular neural network had already been applied with some degree of success to the 2-D as well as 3-D Particle Tracking Velocimetry (PTV) is proposed by Achyut and Ohmi, [21]. In the given paper, the new approach in which image frames are divided in to segments before candidate particles go the matching operations. The particle pairing results are tested with the standard images available from Visualization society of Japan. The algorithm has been successful to deal comparatively the higher number of particles within considerable degree of accuracy [22].

If looked over the recent approaches, [23] implemented Hopfield Neural Network to particle pairing process of two dimensional PTV. Hopfield neural network's inherent demerit of long computation time with increase in neuron units (number of particles per frame in case of particle tracking algorithms) makes it reluctant to negotiate those who wish for its implementation to relatively higher number of particles per frame. This fact is evident from the experimental results reported [24] that only around 150 particles per frame were successfully matched within the tolerable computation time. In order to overcome this problem the Ohmi and Sapkota suggested the idea of implementing Cellular Neural Network [25], in which neuron units' connections are limited to units in local neighborhood of individual units.

Cellular Neural Network (CNN) [26] is a massive parallel computing paradigm defined in discrete N-dimensional spaces. Unlike Hopfield neural network, the connections are limited to units in local neighborhood of individual units in this neural network. In other words, any cell is connected only to its neighbor units, *i.e.* adjacent units interact directly with each other. Units not in the immediate neighborhood have indirect effect because of the propagation effects of the dynamics in the network.

Perfect matching of refractive indices between the working fluid and the flow model provides an optical access for imaging and lighting adequate for PIV. Such an approach was reported by one of the authors [27] who used the glycerol solution and a silicone-rubber model for index matching.

Significantly reduces facility test times over conventional measurement techniques incorporating intrusive diagnostic probes [28].

In both optical configurations, the measurement planes were sized to completely capture the fully turbulent jet shear layer growth [29]. The measured three-dimensional mean and turbulent velocity fields, along with computed second-order statistics including axial vorticity and turbulent kinetic energy, were evaluated for all test points [30] [31] [32]. Well-defined streamwise vortex structures in the jet shear layers were measured and documented.

## 2. LABONTE'S SOM ALGORITHM

The neural network techniques are becoming a useful tool for the particle tracking algorithm of the PIV system software and among others, the self-organizing maps (SOM) model seems to have turned out particularly effective for this purpose. This is mainly because of the performance of the particle tracking itself, capacity of dealing with unpaired particles between two frames and no necessity for a priori knowledge on the flow field (*e.g.* maximum flow rate) to be measured.

In this regard, a pioneering work using this SOM network model would be the algorithm proposed by Labonté [33], smart and efficacious with his new ideas of modifying the original model established by Kohonen [34]. However, according to the present author's tests, his account of SOM particle tracking seems to have still some more room for improvement, especially in the case of densely seeded particle images.

In this case, the treatment of unpaired particles, a specific problem to any type of PIV, is not fully considered and thereby, the tracking goes unsuccessfully for some particles. Therefore, the objective of the present study is to bring about further improvement and practicability in this promising particle tracking algorithm. The new improved algorithm will be tested in the light of the (synthetic) PIV standard image as well as in (real) particle images obtained from visualization experiments.

Let ***xi*** (*i*=1,..,*N*) and ***yj*** (*j*=1,..,*M*) be the coordinate vectors of the particles in the first and the second frames respectively. The neural network is composed of two similar sub-networks, each one corresponding to one of the two frames. The first network has *N* neurons situated at ***xi*** and the second one has *M* neurons at ***yj***. Each neuron has two weight vectors, corresponding to the two components of the coordinate vectors ***xi*** and ***yj***, and is denoted by ***vi*** for the first sub-network and by ***wj*** for the second one. These weight vectors are assigned the following initial values:

$$\boldsymbol{v}_i = \boldsymbol{x}_i \quad \left( i = 1,..,N \right), \qquad \boldsymbol{w}_j = \boldsymbol{y}_j \quad \left( j = 1,..,M \right) \qquad (1)$$

The weight vectors are so updated that those of one sub-network should work as stimuli for the other sub-network. Concretely, the stimulus vector ***vi*** from the first sub-network is presented to the second sub-network. Then, a winner neuron is selected from the latter sub-network as the one with the weight vector closest to ***vi***. Let *c* be the index of this neuron and ***wc*** its weight vector, then each neuron of the second sub-network is subjected to the following displacement of weight vectors:

$$\Delta \mathbf{w}_j(c) = \alpha_j(\mathbf{v}_i - \mathbf{w}_c) \qquad \left( j = 1,...,M \right),$$
$$\alpha_j = \begin{cases} \alpha & \text{if neuron } j \in S_c(r) \\ 0 & \text{otherwise} \end{cases} \qquad (2)$$

where $\alpha j$ is a scalar variable between 0 and 1 and *Sc* (*r*) the radius *r* of a closed circle centered on the point ***yc***. The weight increment formula in equation (2) is given an important modification from the original Kohonen [34] network model, in which the right-hand term is expressed as (***vi*** – ***wj***) instead of (***vi*** – ***wc***). Each time the weight vector ***vi*** is presented to the second sub-network, the weight vectors of the latter sub-network are updated according to:

$$\mathbf{w}_j \leftarrow \mathbf{w}_j + \sum_{i=1}^{N} \Delta \mathbf{w}_j(c_i) \left( j = 1,...,M \right) \ (3)$$

In the next step, by contrast, the stimulus vector ***wj*** from the second sub-network is presented to the first sub-network. A winner neuron is selected as the closest one to ***wj***. Each time the weight vector ***wj*** is presented to the first sub-network, the weight vectors of the latter sub-network are updated according to:

$$\mathbf{v}_i \leftarrow \mathbf{v}_i + \sum_{j=1}^{M} \Delta \mathbf{v}_i(c_j) \ \left( i = 1,...,N \right) \ (4)$$

At each step when all the weight vectors from either sub-network are updated, the radius *r* of the circle, within which the neuron weights are changed, is decreased by $r \leftarrow \beta r$ ($0 < \beta < 1$). At the same time, the amplitude $\alpha$ of the weight translation is increased by $\alpha \leftarrow \alpha / \beta$.

These alternate steps are iterated until the radius *r* of the circle reaches a given value of *rf*, which should be small enough to cover only the winner neuron. Since the correspondence between a weight vector and its matching neuron is not always reciprocally identical for the two sub-networks, a final nearest-neighbor check is done with a neighborhood criterion of small radius $\varepsilon$.

## 3. OHMI'S SOM ALGORITHM

The above-mentioned Labonté's algorithm seems to leave some room for improvement in the following two aspects. First, the weight vectors update in equation (2) is probably too simple, because this is a yes-or-no scheme depending on the distance from the winner neuron. Some vectors would be displaced too much and others too little. So, a new update scheme has been implemented by using a distance-dependent Gaussian function, which is as follows:

$$\alpha_j = \begin{cases} \alpha & \text{if } |\mathbf{w}_c - \mathbf{w}_j| \le r \\ \alpha * \exp\left\{-(|\mathbf{w}_c - \mathbf{w}_j| - r)^2 / (2r^2)\right\} & \text{if } |\mathbf{w}_c - \mathbf{w}_j| > r \end{cases} \quad (5)$$

Another improvement would be required for detecting 'unpaired' or 'overlapped' particles more explicitly. Although the Labonté's original algorithm is capable of dealing with these loss-of-pair particles in the final nearest-neighbor check, this is obviously not enough for densely seeded particle images. Therefore, the present author has adopted an idea from the refined TSP (traveling salesman problem) algorithm [35]. In their algorithm, the neurons can be removed or doubled according to the status of the winner nomination from iteration to iteration. More precisely, the neurons not nominated as a winner for consecutive three (ten in the present work) iterations are removed from the network. Likewise, the neurons nominated as a winner by two neurons from the opposite network are doubled at the same position. The removal of neuron stands for loss of paired particles and the duplication for inseparable overlap of particle images. As a matter of face, the removed neurons have no chance to come back to life, whereas the doubled neurons are not necessarily allowed to survive until the final stage. This process of iteration allows better detection then Labonte's SOM.

## 4. MODIFIED SOM ALGORITHM

Among unsupervised learning methods, the Kohonen SOM has been strongly suggested as an ideal candidate for clustering of textual documents. Kohonen based his neural network on the associative neural properties of the brain. The network contains two layers of nodes: an input layer and a mapping layer in the shape of a two dimensional grid. The output layer acts as a distribution layer. The number of nodes in the input layer is equal to the number of features associated with the input. Each node of the mapping layer has as many features as there are input nodes.

Thus, the input layer and each node of the mapping layer can be represented as a vector that contains the number of features of the input. [36] applied the Kohonen SOM to textual analysis in an attempt to detect the logical similarity between words from the statistics of their contexts. [37] developed DISCERN (Distributed Script processing and Episodic memoRy Network) as a prototype of a subsymbolic natural language processing system based on the Kohonen SOM. [38] used the Kohonen SOM for classifying documents for information retrieval.

The algorithm proposed by Ohmi [39] has been modified using Delta Bar Delta Rule. This modification has reduced the computation time. Since the cost surface for multi-layer networks can be complex, choosing a learning rate can be difficult. What works in one location of the cost surface may not work well in another location. Delta-Bar-Delta is a heuristic algorithm for modifying the learning rate as training progresses. Initially, let us discuss about delta rule. In this case, the step function is replaced with a continuous activation function. For normal classification problem, use the step function. In this case, the weights are updated. This is the same algorithm used for regression. All that really differs is how the classes are determined.

Consider the simplest useful model used in several analysis.

$$y = w_1\ x + w_0 \qquad (6)$$

This is a linear model: in a xy-plot, equation (6) describes a straight line with slope $w_1$ and intercept $w_0$ with the y-axis. How do we choose the two parameters $w_0$ and $w_1$ of general model? Clearly, any straight line drawn somehow through the given sets of data could be used as a predictor, but some lines will do a better job than others.

In order to make precise what we mean by being a "good predictor", we define a loss (also called objective or error) function $E$ over the model parameters. A popular choice for $E$ is the sum-squared error:

$$E = \frac{1}{2}\sum_p (t_p - y_p)^2 \qquad (7)$$

where:

*p* = number of points,

*tp* = desired target value associated with the *p*th example,

*yp* = output of network when the *p*th input pattern is presented to network.

In words, it is the sum over all points *p* in our data set of the squared difference between the target value *tp* and the model's prediction *yp*, calculated from the input value *x* by equation (6). For a linear model, the sum-squared error is a quadratic function of the model parameters. The loss function $E$ provides us with an objective measure of predictive error for a specific choice of model parameters. We can thus restate our goal of finding the best (linear) model as finding the values for the model parameters that minimizes $E$.

The gradient of $E$ gives us the direction in which the loss function at the current setting of the w has the steepest slope. In order to decrease $E$, we take a small step in the opposite direction, -G.

By repeating this over and over, we move "downhill" in $E$ until we reach a minimum, where G = 0, so that no further progress is possible.

The linear model of equation (6) can in fact be implemented by the simple neural network. It consists of a bias unit, an input unit, and a linear output unit. The input unit makes external input x available to the network, while the bias unit always has a constant output of 1. The output unit computes the sum:

$$y_2 = y_1\ w_{21} + 1.0\ w_{20} \qquad (8)$$

It is easy to see that this is equivalent to equation (6), with $w_{21}$ implementing the slope of the straight line, and $w_{20}$ its intercept with the y-axis. It is shown that how an optimal linear function for predicting one variable from one other. Suppose now that we are also given one or more additional variables which could be useful as predictors. The simple neural network model can easily be extended to this case by adding more input units.

Similarly, it is possible to predict more than one variable from the data that are given. This can easily be accommodated by adding more output units. The loss function for a network with multiple outputs is obtained simply by adding the loss for each output unit together. The network now has a typical layered structure: a layer of input units (and the bias), connected by a layer of weights to a layer of output units.

In order to train neural networks such as the ones shown above by gradient descent, we need to be able to compute the gradient *G* of the loss function with respect to each weight *wij* of the network. It tells how a small change in that weight will affect the overall error $E$. It can be illustrated by splitting the loss function into separate terms for each point *p* in the training data:

$$E = \sum_p E^p, \qquad E^p = \frac{1}{2}\sum_o (t_o^p - y_o^p)^2 \quad (9)$$

where *o* ranges over the output units of the network. The superscript *p* to denote the training point and this is not an exponentiation. Since differentiation and summation are interchangeable, we can likewise split the gradient into separate components for each training point:

$$G = \frac{\partial E}{\partial w_{ij}} = \frac{\partial}{\partial w_{ij}} \sum_p E^p = \sum_p \frac{\partial E^p}{\partial w_{ij}} \quad (10)$$

The notation can be easier by computation the gradient for a single data point, omitting the superscript *p*. Also applying the chain rule to decompose the gradient into two factors:

$$\frac{\partial E}{\partial w_{oi}} = \frac{\partial E}{\partial y_o}\frac{\partial y_o}{\partial w_{oi}} \quad (11)$$

The first factor can be obtained by differentiating equation (9):

$$\frac{\partial E}{\partial y_o} = -(t_o - y_o) \quad (12)$$

Using, $y_0 = \sum_j w_{oj} y_j$ the second factor becomes:

$$\frac{\partial y_o}{\partial w_{oi}} = \frac{\partial}{\partial w_{oi}} \sum_j w_{oj} y_j = y_i \quad (13)$$

Using the equations 11-13 together, it can be shown that:

$$\frac{\partial E}{\partial w_{oi}} = -(t_o - y_o) y_i \quad (14)$$

To find the gradient *G* for the entire data set, the sum at each weight the contribution given by equation (14) over all the data points. We can then subtract a small proportion μ (called the learning rate) of *G* from the weights to perform gradient descent. The gradient descent algorithm is as follows:

```
1. Initialize all weights to small random values.
2. Repeat until done
2.1 For each weight $wij$ set $\Delta wij = 0$
2.2 For each data point $(x, t)^p$
2.2.1 set input units to x
2.2.2 compute value of output units
2.2.3 For each weight $wij$ set $\Delta wij = \Delta wij + (ti - yi)\, yj$
3. For each weight $wij$ set $wij = wij + \mu\, \Delta wij$
```

The algorithm terminates once *G* is zero, or sufficiently near to, the minimum of the error function, where *G* = 0. Then it is assumed that the algorithm has converged. In any type of neural network an important consideration is the learning rate μ, which determines by how much we change the weights w at each step and it should be assigned carefully. If μ is too small, the algorithm will take a long time to converge (figure 1).Conversely, if μ is too large, we may end up bouncing around the error surface out of control - the algorithm diverges (figure 2). This usually ends with an overflow error in the computer's floating-point arithmetic.

It is seen earlier that the gradient contributions for all data points in the training set before updating the weights. This method is often referred to as batch learning. An alternative approach is online learning, where the weights are updated immediately after seeing each data point. Since the gradient for a single data point can be considered a noisy approximation to the overall gradient *G*, this is also called stochastic (noisy) gradient descent. Online learning has a number of advantages:

- it is often much faster, especially when the training set is redundant (contains many similar data points),

- it can be used when there is no fixed training set (new data keeps coming in),
- it is better at tracking nonstationary environments (where the best model gradually changes over time),
- the noise in the gradient can help to escape from local minima (which are a problem for gradient descent in nonlinear models).

The cost function *E* that measures how well the network has learned is given in equation (15) and is similar to the least square function.

$$E = \frac{1}{2}\sum_{i=1}^{n}(t_i - y_i)^2 \tag{15}$$

where: *n* = number of examples,

*ti* = desired target value associated with the i^th example,

*yi* = output of network when the i^th input pattern is presented to network.

To train the network, we adjust the weights in the network so as to decrease the cost (this is where we require differentiability). This is called gradient descent.

**Algorithm**

- Initialize the weights with some small random value
- Until *E* is within desired tolerance, update the weights according to where *E* is evaluated at W(old), μ is the learning rate, and the gradient is

$$W_{(new)} = W_{(old)} - \mu\frac{\partial E}{\partial W} \tag{16}$$

$$\frac{\partial E}{\partial W} = \sum_{i=1}^{n}(t_i - y_i)\ x_i \tag{17}$$

If there are more than 2 classes we could still use the same network but instead of having a binary target, we can let the target take on discrete values. For example if there are 5 classes, we could have t=1,2,3,4,5 or t= -2,-1,0,1,2. The modified SOM involves the similar condition. It turns out, however, that the network has a much easier time if we have one output for class. We can think of each output node as trying to solve a binary problem.

In the modified SOM, the Delta-Bar-Delta rule is considered and in this algorithm the learning rate is different for each weights and gradient at the present time is compared with the previous time to estimate the idea for learning rate.

- Each weight has its own learning rate.
- For each weight: the gradient at the current time step is compared with the gradient at the previous step (actually, previous gradients are averaged)
- If the gradient is in the same direction the learning rate is increased
- If the gradient is in the opposite direction the learning rate is decreased
- Should be used with batch only.

Let $g_{ij}(t)$ = gradient of *E* wrt $w_{ij}$ at time t

then define

$$\bar{g}_{ij}(t) = (1-\beta)g_{ij}(t) + \beta\bar{g}_{ij}(t-1) \text{ where } 0<\beta<1 \tag{18}$$

Then the learning rate $\mu_{ij}$ for weight $w_{ij}$ at time t+1 is given by

$$\mu_{ij}(t+1) = \begin{cases} \mu_{ij}(t) + \kappa & \bar{g}_{ij}(t-1)g_{ij}(t) > 0 \\ (1-\gamma)\mu_{ij}(t) & \bar{g}_{ij}(t-1)g_{ij}(t) < 0 \\ \mu_{ij}(t) & \text{otherwise} \end{cases} \tag{19}$$

where β, γ, and κ are chosen depending upon the type of image. Normal value for all these parameters is from 0 to 1. The time complexity of the modified SOM is less compared to Ohmi's SOM. Let $t_1$ is time complexity for Ohmi's SOM and $t_2$ for modified SOM and comparison can be illustrated as follows:

$$\alpha_1 = \alpha * \exp\left\{-(|\mathbf{w}_c - \mathbf{w}_j| - r)^2/(2r^2)\right\} \tag{20}$$

The equation (20) shows the learning rate calculation as prposed by Ohmi and it is based on distance-dependent Gaussian function. If there n point iterations, then the computational complexity of Ohmi's SOM is given by:

$$t_1 = O(n^2) \tag{21}$$

The computational complexity and especially time complexity is given by Big $O$ notation. Equation (20) involves quadratic and exponential function, whereas proposed modified SOM involves only simple linear function. According to computational complexity system, computational time requirement for linear system is less than polynomial and transcendental equations. As seen from equation (20), it involves both polynomial and transcendental equation. As mentioned earlier, the learning rate converges faster in the modified SOM for specific values of $\beta = 0.2$, $\gamma = 0.5$, and $\kappa = 0.4$. In this case, the number of required iteration reduces from $n_1$ to $n_2$ and the computational complexity becomes:

$$t_2 = O(n_2) \tag{22}$$

$$n_1 > n_2 \text{ and } t_1 > t_2 \tag{23}$$

As seen from equation (23), the modified SOM is better than Ohmi's SOM.

## 5. RESULTS AND DISCUSSION

### Preprocessing algorithm

The HDV algorithm is based on the elimination of floating particles and light streaks. The HDV elimination range and HDV elimination relaxation for horizontal, vertical and diagonal directions are entered and noises are eliminated. The algorithm is comparatively very fast and finally image is binarized.

The test image originates from the PIV Standard Experiment http://www.vsj.or.jp/piv [40] and [41], one of the latest activities of the PIV Standard Project inaugurated by the Visualization Society of Japan, and the particle motion here represents a periodic water flow in a quasi 2-D reservoir tank undergoing a self-excited oscillation or simply called as sloshing. Two large-scale recirculating flows are existent on the right and left sides of the reservoir and the gross size of these two recirculating flows is switched alternately, giving rise to a periodic standing wave on the water surface. The digital visualization image, shown in figure 4, is composed of 512 by 512 pixels per frame with 8 bit/pixel gray-level resolution but the particles are distributed in a much smaller area, covering approximately 50% of the whole image area. The particle distribution density is definitely smaller than the preceding test images and the number of particles recognizable per frame is 400 to 600. According to the parallel LDV measurement, the maximum flow rate is reported around 6 pixels per frame, but most of the PIV measurements are carried out for every other frame and in that case, the maximum flow rate is 12 pixels per frame.

The comparison for the five different types of particle detection algorithms are performed using the PIV standard experiment image (spg1500). For better visualization all the images are inverted. The original image is illustrated in figure 4. The comparative results of this PIV standard experiment image are given in figure 5.2 to 5.7, from which it is convinced that the particle identification algorithm is a very important factor for determining the accuracy of particle tracking velocimetry. The visualizations of the BIN, PMC, MOR, DTB, HDV, and HDV+BIN are given in figure 5 to 10 respectively. The BIN algorithm falls under the acceptable to good category with a relatively small number of particle extractions. Due to its excessive sensitivity to random noise, the PMC has demonstrated here very poor performance, which is beyond the acceptable range. On the other hand, the

MOR algorithm can extract the particle to acceptable range. The performance of the DTB is in acceptable range but unable to eliminate floating particle properly. It is clearly observed that the HDV algorithm with binarization has demonstrated better results and virtually all floating particles are eliminated. Due to manual preset of mask size, it is difficult to get better result using the Moravec operator. If this mask size is not properly determined, the resultant detection of particle centers may include many errors.

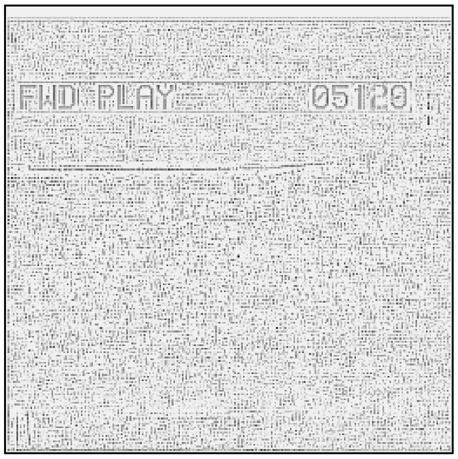


Fig. 6. Particle Mask Correlation (PMC)

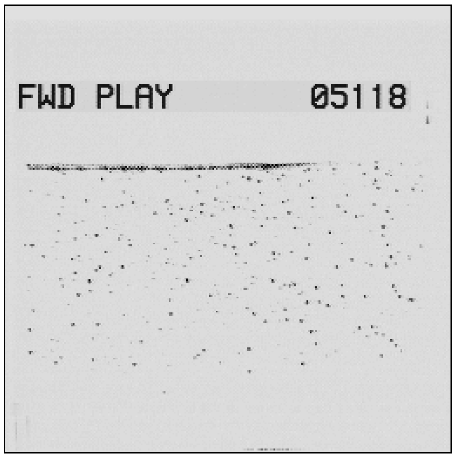


Fig. 4. Standard Experimental Image (Spg1500)

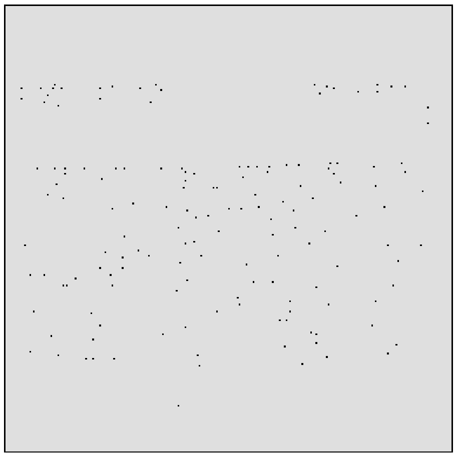


Fig. 7. Moravec Operator (MOR)

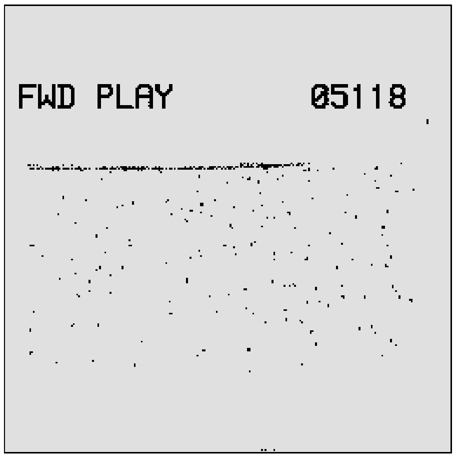


Fig.5. Binarization (BIN)

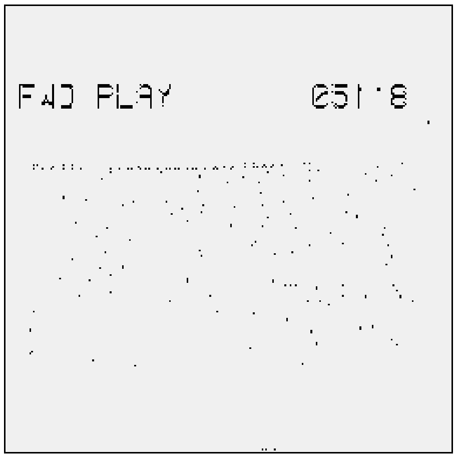


Fig. 8. Dynamic Threshold Binarization (DTB)

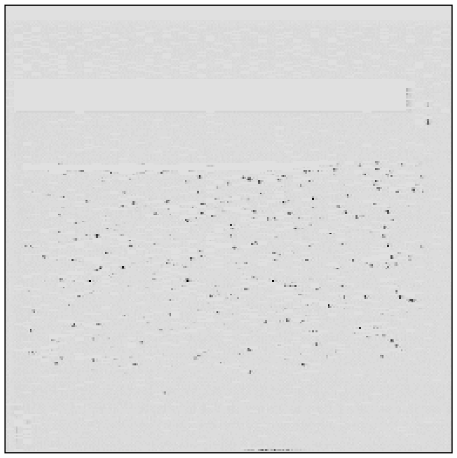
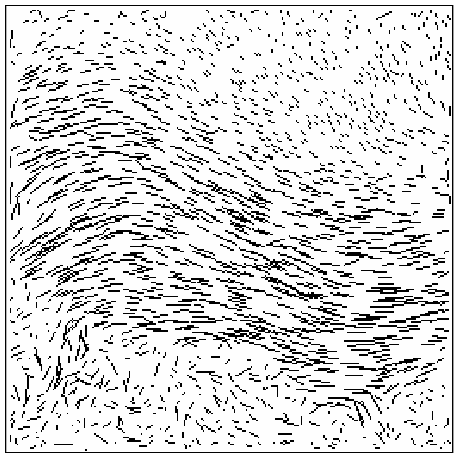


Fig. 9. HDV Extraction (HDV)


b) Labonté's algorithm

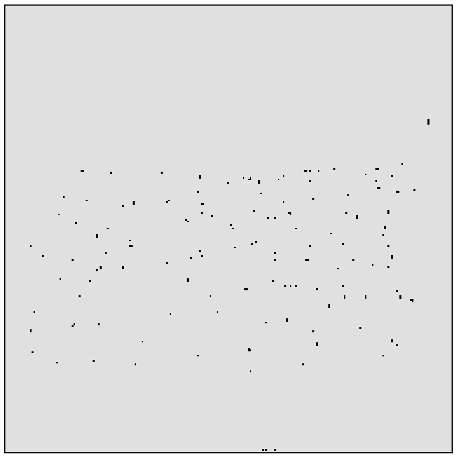
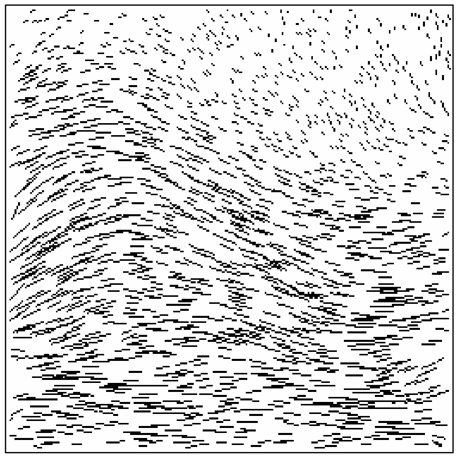
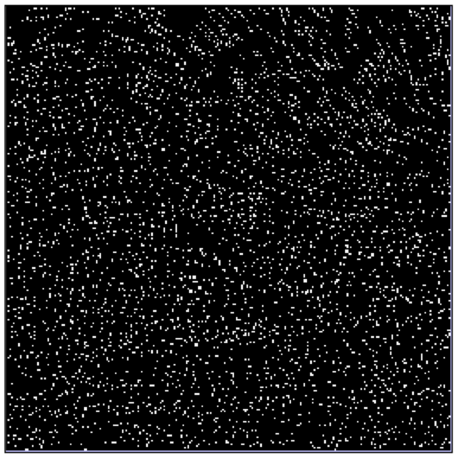
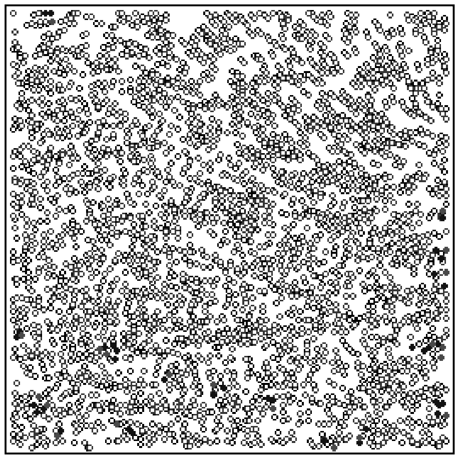


Fig. 10. HDV Extraction + Binarization (HDV+BIN)

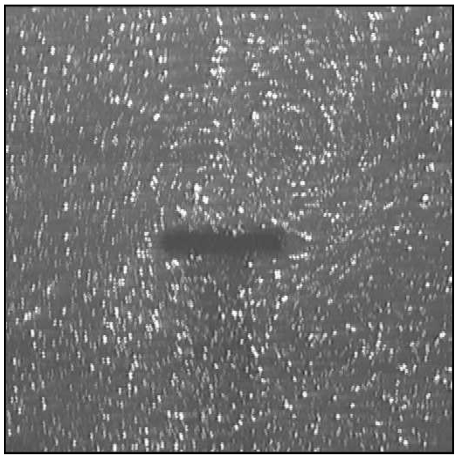

(c) Modified SOM algorithm


(a)Overlap of two original frames

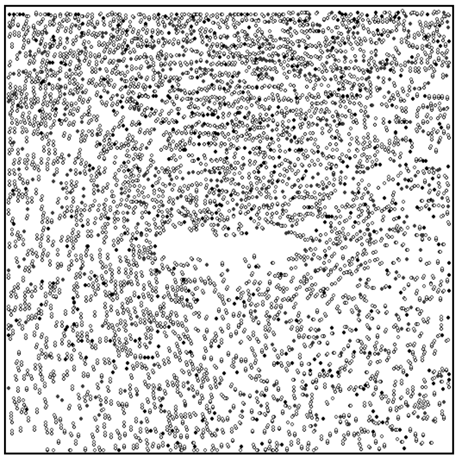

(d) Unpaired particles according to the Labonté

## Particle detection algorithm

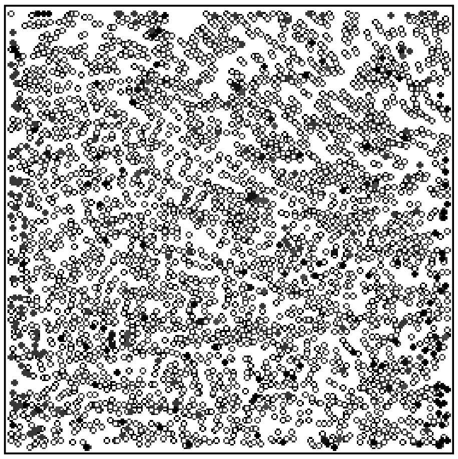
The modified SOM algorithm was tested first using a synthetic particle image from the web site of the Visualization Society of Japan (VSJ). Their PIV standard image (Okamoto *et al*., 1997), showing a portion of a channel flow with an impinging jet, is a good measure for all the PIV/PTV algorithms. The particle tracking result by the modified algorithm is shown in figure 11, where the result of the Labonté's algorithm is also given for comparison. In both cases, the centroid of the individual particles was computed from the original images through the dynamic threshold binarization algorithm developed by Ohmi and Li [42] and it gives acceptable results.



(a) Overlap of two original frames



(b) Paired and unpaired particles resulting from the modified SOM algorithm (unpaired particles are indicated by black and red solid plots).

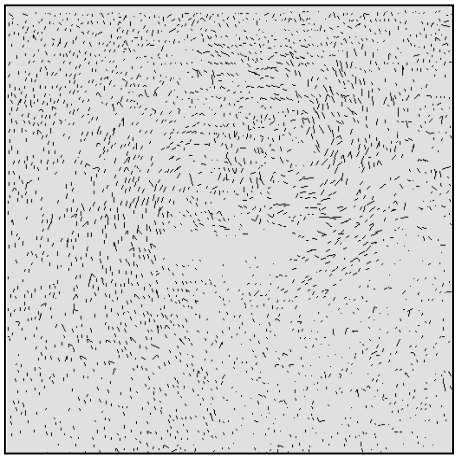


(e) Additional of unpaired particles by the modified SOM

Fig. 11. Particle tracking results of the PIV standard image #01 [41]

The system parameters of the original and modified SOM algorithms are as follows: *r0* (initial radius) = 100.0, *rf* (final radius) = 0.1, $\alpha$ (initial translation rate) = 0.005, $\beta$ (attenuation rate) = 0.9 and $\varepsilon$ (max distance for final pairing) = 0.3. As shown in Figure 11(b) and (c), the performance of particle tracking is much improved with the modified algorithm, especially so near the border of the image as well as in the high-speed flow region.

Both figures, 11(d) and (e) show the positions of individual particles detected by the dynamic threshold binarization method. The black and red hollow plots stand for the paired particles originating respectively from the first and second frames. The blue and pink solid plots in figure 11(d) indicate the unpaired particles in the first and second frames were detected by the Labonté's algorithm. By contrast, figure 11(e) shows an addition of more unpaired particles detected by the modified SOM algorithm. These additional particles are depicted by the black and red solid plots in the figure, the colors still corresponding to their original frames. This can state that more particles can be detected by modified SOM.


(c) Labonte's algorithm

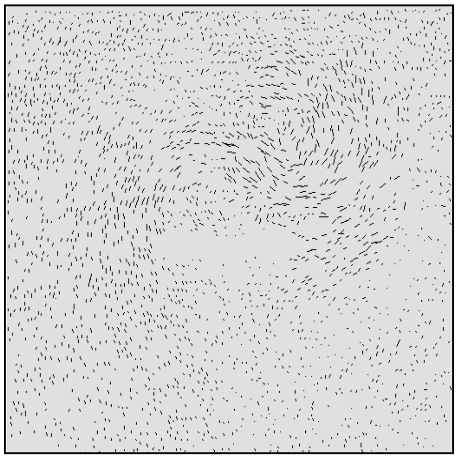

(d) Modified SOM algorithm

Fig. 12. Particle tracking results of the PIV Benchmark Test Image [43]

From these figures, it is recognized that the improvement of the particle tracking result by the modified SOM algorithm is mainly due to much better detection of unpaired particles along the border area of the particle image. The number of the blue and pink solid plots in figure 11(e) is somewhat increased if compared with those of figure 11(d). This is because they do not originate from the Labonté's original algorithm but from the modified version with the new definition of the weight variable *αj*, as expressed by equation (5). The given equation provides different learning rate for different weights and the conversion is faster. Figure 12 shows that the Labonte's algorithm contains spurious vectors in the middle of the flow region but, modified SOM exhibits smooth flow in the same region.

The second test was attempted by using a real experimental image supplied by Hayami [43] and available now at *ftp://bluebonnet.utnl.gen.u-tokyo.ac.jp/pub/piv*. The wake of a slowly oscillating flat plate in still water was visualized with particles and the image was recorded by a simplified image acquisition system consisting of a home video movie and a single-frame image capture board. The resultant image is rather noisy, narrow-banded in histogram and temporally fluctuant in the gray level. This particle image, called as PIV Benchmark Test Image by the authors, was tested by the modified SOM as well as the Labonté's original algorithms and the results are provided comparatively in figure 12. In both cases, the overall movement of the vortex flow is roughly captured but in the Labonté's result, there are quite a few spurious vectors observable. As a result, the shape and the location of the upper-surface vortex, for instance, are less clearly defined.

## Discussion

The experiment image (spg1500) is considered for the comparison of five different types of particle detection algorithms. All the images are inverted for better visualization. The original image is given in figure 4. The visualizations of the BIN, PMC, MOR, DTB, HDV, and HDV+BIN are given in figure 5 to 10 respectively. The BIN algorithm falls under the acceptable to good category with a relatively small number of particle extractions. Due to its excessive sensitivity to random noise, the PMC has demonstrated here very poor performance, which is beyond the acceptable range. On the other hand, the MOR algorithm can extract the particle to acceptable range. The performance of the DTB is in acceptable range but unable to eliminate floating particle properly. The dynamic threshold binarization (DTB) algorithm is more suitably applicable to relatively low-density particle images in which the number of particles does not exceed 400 per 256 by 256 pixels and less suspended particles.

Figure 12 (b) shows the positions of individual particles detected by the dynamic threshold binarization method. The paired particles are indicated by the black and red hollow plots and the unpaired particles by the solid plots. In this case, again, the modified algorithm detects not a few unpaired particles near the border of the image. In addition, the modified algorithm seems more robust against the fluctuant noise components of the tested image, because a number of unpaired particles are detected in rather noisy parts of the original image. In this context, particle tracking by the modified algorithm can contribute for better performance in the particle tracking velocimetry.

The overlap of two original frames is given in figure 11(a). The labonte,s algorithm shown in figure 11(b) contains spurious vectors in the bottom of the flow. This is eliminated in the modified algorithm shown in figure 11(c). The modified SOM can effectively detect unpaired particles in the border area of the particle image.

Figure 13, shows that Ohmi's algorithm can detect more particles than Labonte's algorithm. The particles detected by modified algorithm are similar to Ohmi's algorithm in terms of numbers of particle detection. From figure 14, and table 1, it can be conformed that modified algorithm is computationally better than Ohmi's algorithm.

## 6. CONCLUSION

The flow visualization experiments contain random noise generated by scattering of floating particles and streak of lights. The selection of proper algorithm depends upon the characteristics of the target flow. However, if is necessary to select only one algorithm for pre-processing of the image, the HDV algorithm along with binarization would be the most suitable. It can effectively eliminates the suspended particles and light streaks.

The dynamic threshold binarization (DTB) algorithm is more suitably applicable to relatively low-density particle images in which the number of particles does not exceed 400 per 256 by 256 pixels and less suspended particles.

The SOM neural network is considered as an increasingly promising approach for the use in the particle tracking velocimetry. With the introduction of more new ideas, the performance of the particle tracking algorithm seems to go up to a practical-use level. Further efforts should be made to apply the algorithm to more densely seeded particle images with larger numbers of particles (up to several tens of thousands) and with more dynamic range of velocity.

Comparing Figure 11 (b) and (C), show that modified algorithm can demonstrate better performance in the boarder of the image and also in the high-speed flow region. The performance of modified SOM algorithm shown in figure 12 (d) is better than Labonte's algorithm illustrated in figure 12 (c). Figure 12 (b) shows the positions of individual particles detected by the dynamic threshold binarization method. In addition, the modified algorithm seems more robust against the fluctuant noise components of the tested image, because a number of unpaired particles are detected in rather noisy parts of the original image. Probably, these contribute to the better performance of particle tracking by the modified algorithm.

The comparison of three algorithms: Labonte SOM, Ohmi SOM, and Modified SOM is given in the table 1. It is observed that for lower particle density, the performance is similar for all algorithms up to 100 particles. For more highly dense particles, figure 13 shows that Ohmi's algorithm can detect more particles than Labonte's algorithm. The particles detected by modified algorithm are similar to Ohmi's algorithm. From figure 13, and table 1, it can be conformed that modified algorithm is computationally better than Ohmi's algorithm. It is seen that computational time for 1200 particles is 10 seconds for Ohmi's algorithm and which is reduced to 7 seconds in the modified algorithm.

## 7. RECOMMENDATION

Recently, SOM neural network is considered as an increasingly promising approach for the use in the particle tracking velocimetry. With the introduction of more new ideas, the performance of the particle tracking algorithm seems to go up to a practical-use level. Further efforts should be made to apply the algorithm to more densely seeded particle images with larger numbers of particles (up to several tens of thousands) and with more dynamic range of velocity. The proposed algorithm performance is not so satisfactory beyond 1200 particles.

The proposed algorithm can be modified for the detection of 3D particle tracking. Further, it is not easy to choose the parameters β, γ, and κ of the delta bar delta rule. A technique can be devised to approximate these values for faster conversion of the algorithm. It can enable to reduce the computational complexity much more than the proposed algorithm.

## REFERENCES

[1] Hong-yeol Yoon, In-seop Lee, Oh-sung Kwon, Kwang-hyup An, Dong-ik Rhee, Akikazu Kaga, Katsuhito Yamaguchi, "Visualization of Image Processing of Flow around an Axial Flow Fan in the Outdoor Side of an Air Conditioner", *Proceedings of VSJ-SPIE98*, 1998, p 238.

[2] Guezet Jacky, Youji Kurosawa, Mitsuo Gomi, Kazuo Suzuki, "PIV Measurements of Ram Flameholder Wake-Flow with Combustion", *Proceedings of VSJ-SPIE98*, 1998, p 242.

[3] Asztalos Balazs, Takashi Yamane, Masahiro Nishida, T. Masuzawa, Y. Konishi, "Flow Visualization and Evaluation of Centrifugal Blood Pumps for Artificial Heart", *Proceedings of VSJ-SPIE98*, 1998, p 246

[4] Baumann S., J. R. Kadambi, H. Harasaki, S. Amirthaganesh, M. P. Wernet, "PIV Studies of Heart Valve Prostheses in Pulsatile Flow", *Proceedings of VSJ-SPIE98*, 1998, p 248.

[5] Hui Hu and Manoochehr. M. Koochesfahani, "A Novel Technique for Quantitative Temperature Mapping in Liquid by Measuring the Lifetime of Laser Induced Phosphorescence", *Journal of Visualization,* 2003, Vol. 6, No.2, pp143-153.

[6.] Adrian, R. J., "Particle-Imaging Techniques for Experimental Fluid Mechanics", *Annu. Rev. Fluid Mech.*, 1991, p.261-304.

[7] Ohmi, K., Li Hang Yu, Shashidhar Ram Joshi, "Performance of relaxation method PTV on the basis of the PIV standard images", *CD-ROM Proceedings of the VSJ/SPIE (Optical Technology and Image Processing in Fluid, Thermal and combustion Flow) Symposium*, 1998, AB-122.

[8] Etoh, T., Takehara, K, "The Particle Mask Correlation Method", *CD-ROM Proceedings of the 8th Int. Symp. on Flow Visualization*, Sorrento, Italy, 1998, p.283.

[9] Moravec,H.P., 'Towards automatic visual obstacle avoidance', *Proceedings 5th Int. Joint Conf. Artificial Intelligence*, 1977, p.584.

[10] Ohmi, K., Li Hang Yu, "Particle tracking velocimetry by combined use of the Moravec operator and the relaxation algorithm", *CD-ROM Proceedings 2nd Pacific Symp. on Flow Visualization*, 1999, PF-112.

[11] Ohmi, K., "Image processing of particle path images in two dimensional fluid flows", *J. Flow Visualization Society*, 1986, Vol.6, No.20, p.19.

[12] Ohmi, K., Li Hang Yu, Shashidhar Ram Joshi, "Performance of relaxation method PTV on the basis of the PIV standard images", *CD-ROM Proceedings of the VSJ/SPIE (Optical Technology and Image Processing in Fluid, Thermal and combustion Flow) Symposium*, 1998, AB-122.

[13] Shashidhar Ram Joshi, "HDV Extraction Algorithms in the Particle Tracking Velocimetry", *4th Asia-Pacific Symposium on Information and Telecommunication Technologies*, 2001, p.364.

[14] Okamoto, K., William D. Schmidl, Yassin A. Hassan, "Least Force Technique fotr the Particle Tracking Algorithm", *Proceedings of the seventh International Symposium on Flow Visualization, Flow Visualization IV*, 1995, p.647.

[15] Ishikawa, M., Yamamoto, F., Murai, Y., Iguchi, M., Wada, A., "A Novel PIV Algorithm using Velocity Gradient Tensor", *Proc. PIV-Fukui '97*, 1997, p.51.

[16] Hart Douglas P., "Super-Resolution PIV by Recursive Local-Correlation", *Proceedings of VSJ-SPIE98*, 1998, p 222.

[17] Barnard, S. T., Thompson, W. B., "Disparity Analysis of Images", *IEEE Trans on Pattern Analysis and Machine Intelligence*, Vol.2, No.4, 1980, p.333.

[18] Lee, S. J., Baek, S. J., "Two-Frame PIV and its Application to a Turbulent Channel Flow", *Proc. PIV-Fukui '95*, 1995, p.217.

[19] Ohmi, K., Dao Hai Lam, "New Particle Tracking PIV using an Improved Relaxation Method", *8th International Symposium on Flow Visualization*, 1998, p. 122.

[20] Hui, L., Hui, H., Kobayashi, T., Saga, T., Taniguchi, N., "Application of Wavelet Vector Multi-Resolation Technique to PIV Measurements", *AIAA*, 2001, p.1.

[21] Achyut. Sapkota and K. Ohmi, 2007, "Image segmentation particle tracking: A new approach to cellular neural network based PTV", *PSFVIP-6: The 6th Pacific Symposium on Flow Visualization and Image Processing, Hawaii, USA, May 16th-19th*.

[22] Grant I., Pan X., "The use of neural techniques in PIV and PTV", *Measurement Science Techonology,* 1997, pp. 1399-1405.

[23] Knaak M., Rothlübbers C., Orglmeister R.,1997, "A Hopfield neural network for flow field computation based on Particle Image Velocimetry/ Particle Tracking Velocimetry image sequences", *in Proc. IEEE International Conference on Neural Networks*.

[24] Ohmi K., Sapkota A., 'Particle Tracking Velocimetry using a Hopfield Neural Network', *in Proc. 15th International Symposium on Transport Phenomena,* 2004, *#20*.

[25] Ohmi K., Sapkota A., "Improvement in Hopfield Neural Network PTV", *Proc International conference on Advanced Optical Diagnostics in Fluids, Solids and Combustion, December 4-6, 2004, Tokyo, Japan,* 2004, *#V0010*.

[26] Chua L.O., Yang L., "Cellular Neural Networks: Theory and Applications", IEEE *Trans. on Circuits and Systems, (CAS), 1988, Vol.35,* 1988, *pp.1257-1290*.

[27] Nishino, K., Kawaguchi, D., Kosugi, T and Isoda, H., "Highly Efficient PIV Measurement of Complex Flows Using Refractive Index Matching

Technique", *CD-ROM Proc. 2004 Korea-Japan Joint Seminar on Particle Image Velocimetry, POSTTECH, Korea,* 2004, *Dec. 9-10.*

[28] Opalski, A.; Wernet, M.; and Bridges, J., "Chevron Nozzle Performance Characterization Using Stereoscopic DPIV", 2005, *AIAA-2005-0444.*

[29] Opalski, A.; Paxson, D.; and Wernet, M., "Detonation Driven Ejector Exhaust Flow Characterization Using Planar DPIV", 2005, *AIAA-2005-4379.*

[30] Roth, Don J.; et al., "Approaches for Non-Uniformity Correction and Dynamic Range Extension for Acoustography", *SPIE Int. Soc. Opt. Eng., vol. 5770*, 2005, pp. 124-134.

[31] Thomas,D.G. and Kraus,K.A., "Interaction of Vortex Streets", *J. Applied Physics,* Vol.35, 1964, pp.3458-3459.

[32] Zdravkovich, M. M., "Review of Flow Interference between Two Circular Cylinders in Various Arrangements", *Trans. ASME - J. Fluids Engineering*, 1977, Vol.99, pp.618-633

[33] Labonté G., "A new neural network for particle tracking velocimetry", *Experiments in Fluids.* 26, 1999, pp.340-346.

[34] Kohonen T., "A simple paradigm for the self-organized formation of structured feature maps", *Competition and cooperation in neural nets*, Lecture notes in biomathematics 45, Springer, New York, 1982.

[35] Angéniol B, Vaubois GC, Le Texier JY., "'Self-organizing feature maps and the travelling salesman problem", *Neural Networks.* 1, 1988, pp.289-293

[36] Ritter, H., and Kohonen T., "Self-organizing semantic maps" *Biological Cybernetics, 6),* 1989, 241-254.

[37] Miikkulainen, R,. "Symbolic Natural Language Processing: An Integrated Model of Scripts, Lexicons, and Memory", *Cambridge, MA: MIT Press,* 1993,

[38] Lin X.,; Soergel D.; and Marchionini G., "A self-organizing semantic map for information retrieval", *Proceedings of the Fifteenth Annual lnternational ACM/SIGIR Conference on R&D in Information Retrieval, Copenhagen,* 1992, pp. 37-50.

[39] Ohmi K., "Neural network PIV using a self-organizing maps method", *Proceeding of PSFVIP-4, Chamonix, France,* 2003.

[40] (VSJ) Visualization Society of Japan, 'Essentials in PIV', *CD-ROM publication,.* 1998.

[41] Okamoto, K., Nishio, S., Saga, T., Kobayashi, T., "Standard images for particle imaging velocimetry", *Proc. PIV-Fukui '97*, 1997, p.229.

[42] Ohmi K, Li H.Y., "Particle tracking velocimetry with new algorithms", *Measurement Science and Technology.* 11-6, 2000, pp.603-616.

[43] Hayami H, Oakmoto K, Aramaki S., "A trial of benchmark test for PIV", *Journal of Visualization Society Japan*. 17-S1, 1997, pp.163-166.

**S. R. Joshi** (B.E'84, M.Sc. Engg'92, Research Fellow'98, Ph. D. 07) The author passed his Bachelor of Electrical Engineering from Sardar Ballavbhai Regional College of Engineering, South Gujarat University, Surat, India in 1984 and passed with first class first with distinction. He passed his Master's of Science in Electrical Engineering from University of Calgary, Canada in 1992. He did his Ph. D. from Tribhuvan University in 2007. He was Research Fellow in Osaka Sangyo University, Japan for one year from 1997 to 1998 and published four papers in the peer reviewed international journals. He has joined Institute of Engineering in 1985 and presentlyhe is a professor in the Department of Electronics and Computer Engineering, Institute of Engineering, Pulchowk Campus, Nepal

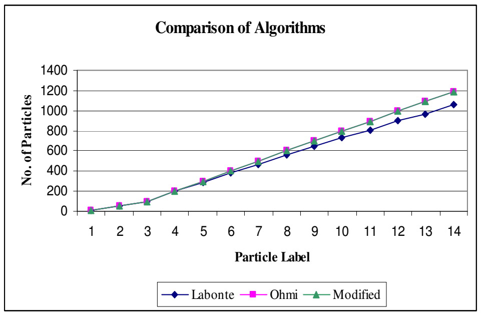


Fig. 13. Comparison of different algorithms

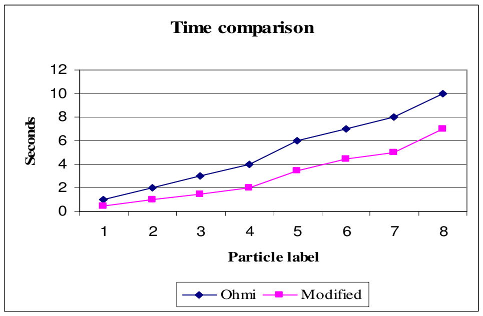


Fig. 14. Computational time comparison of different algorithms

Table 1. Comparison of different algorithms

| S. N. | Number of Particles | Labonte SOM | | Ohmi SOM | | | Modified SOM | | |
|---|---|---|---|---|---|---|---|---|---|
| | | Detect | % | Detect | % | Time (sec) | Detect | % | Time (sec) |
| 1 | 10 | 10 | 100 | 10 | 100 | - | 10 | 100 | - |
| 2 | 50 | 50 | 100 | 50 | 100 | - | 50 | 100 | - |
| 3 | 100 | 100 | 100 | 100 | 100 | - | 100 | 100 | - |
| 4 | 200 | 197 | 98.5 | 200 | 100 | - | 200 | 100 | - |
| 5 | 300 | 291 | 97 | 300 | 100 | - | 300 | 100 | - |
| 6 | 400 | 378 | 94.5 | 400 | 100 | - | 400 | 100 | - |
| 7 | 500 | 468 | 93.5 | 500 | 100 | 1 | 500 | 100 | 0.5 |
| 8 | 600 | 561 | 93.5 | 600 | 100 | 2 | 600 | 100 | 1 |
| 9 | 700 | 650 | 92.9 | 699 | 99.9 | 3 | 699 | 99.9 | 1.5 |
| 10 | 800 | 730 | 91.25 | 793 | 99.1 | 4 | 793 | 99.1 | 2 |
| 11 | 900 | 810 | 90 | 895 | 99.4 | 6 | 895 | 99.4 | 3.5 |
| 12 | 1000 | 899 | 89.9 | 998 | 99.8 | 7 | 998 | 99.8 | 4.5 |
| 13 | 1100 | 970 | 88.18 | 1095 | 99.5 | 8 | 1095 | 99.5 | 5 |
| 14 | 1200 | 1062 | 88.5 | 1193 | 99.4 | 10 | 1193 | 99.4 | 7 |