Applying Feature Tracking to Particle Image Velocimetry

Dmitry Chetverikov and Marcell Nagy
Computer and Automation Research Institute
Budapest, Kende u.13-17, H-1111 Hungary
csetverikov@sztaki.hu

Abstract

Particle Image Velocimetry (PIV) is a popular approach to flow visualisation and measurement in hydro- and aerodynamic studies and applications [7]. The fluid is seeded with particles that follow the flow and make it visible. Traditionally, correlation techniques have been used to estimate the displacements of the particles in a digital PIV sequence. Recently, we have successfully applied to PIV two feature tracking algorithms proposed in computer vision. Promising results were presented in our pilot experimental study [4]. In this paper the algorithmic solutions of the application are described. We address methodological issues which arise when a local motion estimation technique is applied to a complex flow containing discontinuities. In particular, algorithms for coherence filtering and interpolation of a velocity field in the presence of flow discontinuity are proposed. Some new quantitative results of flow estimation are shown.

Keywords. Flow Estimation, Visulalisation, Particle Image Velocimetry, Feature Tracking.

1 Introduction

In this study we apply feature based tracking algorithms to flow measurement with *particle image velocimetry* [7]. Flow visualisation and measurement appear in many scientific and industrial tasks, including the studies of combustion processes, hydrodynamic and aeronautical phenomena, flame propagation and heat exchange problems. PIV refers to a particular method of flow visualisation, when the flow is seeded with particles that reflect light and make the motion visible. Digital PIV, or DPIV, is the recently emerged technique of using high-performance CCD cameras and frame-grabbers to store and process the digitised PIV sequences by computer. Cross-correlation methods implemented via the Fast Fourier Transform have been conventionally used to estimate the flow velocity [12].

Motion estimation and tracking have a long history in machine vision, where numerous efficient optical flow based [2, 8] and feature based [5] algorithms have been developed. However, most of these algorithms have never been tested in DPIV. The reason is twofold. On one hand, researchers and engineers working in flow mechanics are often unaware of the developments in machine vision. On the other hand, flow analysis poses specific problems which should be addressed explicitly if a motion estimation technique developed in machine vision is to be applied to a digital flow sequence. Otherwise, the outcome will be disappointing, as the assumptions typical for computer vision — motion of a few large objects having smooth surfaces — are not applicable to flows. In PIV, precise motion estimation of thousands of tiny, poorly visible and often disappearing particles is required, rendering most of the classical feature tracking (and optical flow) approaches inapplicable [13]. Turbulent flows further complicate the task of velocity estimation, as the assumption of locally coherent motion does not hold in some areas.

Because of these difficulties, it is only recently that attempts have been made to adapt machine vision techniques to PIV. In particular, Quénot et al. [9] designed a dynamic programming (DP) based optical flow algorithm and customised it to PIV. The optimal matching is searched that minimises a distance between two images. This is achieved by the Orthogonal Dynamic Programming technique that slices each image into two orthogonal sets of parallel overlapping strips. The strips are then matched as one-dimensional signals. The number of operations required for an $N \times N$ image is $O(N^3 \log N)$.

In a series of modifications, the algorithm has been enhanced to yield high accuracy of flow velocity estimation and robustness to noise. Tests with standard synthetic and real DPIV sequences [9] show that the DP techniques compare favourably to the classical correlation-based methods in all aspects except the execution time: two of the three versions proposed in [9] run for hours. (For the same sequences, typical running time of the correlation-based techniques is 10 minutes.) This limits the potential application area, as online processing, flow monitoring and analysis of time-varying, non-stationary flows are not feasible.

Our search for alternatives to both the conventional and the DP techniques is driven by two issues: the speed and the tractable complexity of flow. We aim at developing a fast technique that would give reasonable accuracy when applied to complex, possibly time-varying flows. Our basic assumption is that a particle-seeded flow should only be measured (tracked) in the locations which provide sufficient information for the measurement. This is in line with the feature tracking philosophy adopted in the well-known Kanade-Lucas-Tomasi (KLT) Tracker [11] which only tracks local areas of sufficient intensity variation in both x and y directions. These are the areas where the analytic estimation of the displacement is precise. When applied to PIV, this strategy means that the flow should be measured for the particles, since it is the particles that make the motion visible.

Most of the existing approaches to DPIV, including the conventional and the DP based ones, measure the flow either in each pixel or on a dense regular grid of locations, independently from the actual distribution and visibility of the particles. The number of operations per measurement is usually higher than in the feature trackers. Given that the number of particles is much less than the number of pixels, it is reasonable to hope that the feature tracking based approaches will be much faster — and our experience shows that they indeed are.

The price to be paid for the less dense measurement is the necessity to interpolate the obtained velocity field to a regular grid, as traditionally required for visualisation and comparison. However, such interpolation is implicitly done in the existing approaches as well, when motion is estimated in a position where it cannot be observed because of the absence of particles.

Recently, we have successfully applied to DPIV two feature based tracking techniques: the KLT Tracker [11] and our algorithm called IPAN Tracker [5]. (IPAN stands for Image and Pattern Analysis group.) The initial ideas of our study were proposed in [13]. Experimental results demonstrating the PIV-efficiency of the track-

ers are presented in the forthcoming paper [4], in which the algorithmic solutions are not described. This is done in the current paper whose contribution is as follows. First, we present the feature tracking approach to PIV and discuss methodological issues that arise when feature tracking is applied to a complex flow. Then, algorithms are proposed for coherence filtering and interpolation of a velocity field in the presence of flow discontinuity. Finally, quantitative results of flow estimation are presented.

2 Feature tracking techniques applied to DPIV

Feature tracking techniques extract local regions of interest (features) and identify the corresponding features in each image of the sequence. In this section, we outline two particular techniques, the KLT Tracker [11] and the IPAN Tracker [5], which we apply to particle image velocimetry. But let us first discuss some basic problems that should be addressed when a feature tracking (and, in fact, any other motion estimation) technique is used to compute a dense velocity field of an inhomogeneous flow.

Particle flow is usually a coherent motion: spatially close particles tend to have similar velocity vectors. A velocity vector field is typically quite smooth, allowing for detection and correction of a wrong measurement based on a sound and coherent neighbourhood. Resampling non-uniform measurements to a regular grid is also more or less straightforward. Such resampling is usually needed for better visualisation of a flow and for comparison of velocity fields obtained in different ways.

However, in complex flows the coherence assumption may be violated: high variation, or even discontinuity may exist in some parts of the fluid, for instance, when two flows meet. A good solution to flow estimation should therefore: 1. be based on informative and reliable image features; 2. handle distortion of local pattern (e.g., by using affine matching); 3. use the coherence in an adaptive way; 4. detect and handle flow inhomogeneity and discontinuity.

Condition 2 means that patterns formed by the particles are not rigid. These patterns may undergo considerable distortion within a PIV sequence. Condition 3 states that the coherence constraint is very productive, but care should be taken to adapt it to the local behaviour of the flow: sometimes, it might be necessary to relax the constraint. In section 3 we propose algorithms that apply the coherence in a flexible way.

2.1 The KLT Tracker

The KLT Tracker selects features which are optimal for tracking, and keeps track of these features. A good feature is a textured patch with high intensity variation in both x and y directions, such as a corner. The algorithm defines a measure of dissimilarity that quantifies the change of appearance of a feature between the first and the current image frame, allowing for affine distortions. At the same time, a pure translational model of motion is used to estimate the position in the next frame.

More specifically, consider the intensity function $g(x, y)$. Assuming small displacements and minimising the linearised dissimilarity, the displacement vector $\mathbf{d} = (d_x, d_y)^T$ is computed as the solution of the linear system [11]

$$\begin{bmatrix} g_x^2 & g_x g_y \\ g_x g_y & g_y^2 \end{bmatrix} \begin{bmatrix} d_x \\ d_y \end{bmatrix} = \begin{bmatrix} g_x \\ g_y \end{bmatrix}, \tag{1}$$

or, in matrix notation, $Z\mathbf{d} = \mathbf{e}$. (g_x is a partial derivative of $g(x, y)$.) It is implied that Z and $\mathbf{e} = (g_x, g_y)^T$ are integrated over a feature window W. The size of W is normally set to 25×25. A patch defined by the window is accepted as a candidate feature if both eigenvalues of Z, λ_1 and λ_2, exceed a predefined threshold λ: $\min(\lambda_1, \lambda_2) > \lambda$. This ensures that the matrix Z is well-conditioned and the solution of (1) is accurate. When applied to a PIV image, the KLT selects individual particles as feature centres. To cope with relatively large motion, the algorithm is implemented in a multiresolution way.

The number of the features to be tracked, N_f, is specified by the user. A separate parameter sets the minimum distance between the features. In the first frame of a sequence, the candidate features are ranked according to their strength defined by $\min(\lambda_1, \lambda_2)$, then the N_f strongest features are selected. Note that in the first frame feature coordinates are integers, while in the subsequent frames subpixel precision is used.

Each feature being tracked is monitored to determine if its current appearance is still similar, under affine distortion, to the initial appearance observed in the first frame. For this purpose, a larger (6×6) linear system is solved. When the dissimilarity exceeds a predefined threshold, the feature is considered to have been lost and will no longer be tracked. At least two frames are needed for the operation of the algorithm.

Since the KLT algorithm incorporates an analytical solution to motion estimation, it is much faster than the methods that use explicit region matching, such as the conventional cross-correlation and the DP based techniques. The processing speed depends on the image size N and the number of features N_f. The source code of the tracker can be downloaded from the web site [3]. A few parameters are to be set before the algorithm can be applied to a PIV sequence. In particular, the number of features must be significantly increased, allowing to track less prominent features. (The default settings assume very sparse features, which is reasonable when a few large objects are observed.)

2.2 The IPAN Tracker

The IPAN Tracker is a non-iterative, competitive feature point linking algorithm. Its original, application-independent version tracks a moving point set, tolerating point entry, exit and false and missing measurements. Position is the only information assigned to a point. The algorithm is based on a repetitive hypothesis testing procedure that switches between three consecutive image frames and maximises the smoothness of the evolving trajectories.

The application-independent version of the tracker is described in full detail in our recent paper [5]. When applied to PIV, the operation of the algorithm remains basically the same. The modifications necessary for the PIV application are as follows: 1. a PIV-specific feature selection mechanism is added; 2. the cost function is modified to include feature appearance. These modifications are described below.

A PIV image $g(x, y)$ is smoothed by a 3×3 mean filter and the (real-valued) maxima of the smoothed image $s(x, y)$ are selected as the features. Each bright particle is represented by a maximum of $s(x, y)$. For more precise motion estimation, the position of each maximum is corrected by parabolic interpolation in x and y directions separately. Consider a maximum $s_2 = s(x', y')$ and its neighbours in x direction, $s_1 = s(x' - 1, y')$ and $s_3 = s(x' + 1, y')$, $s_2 > \max(s_1, s_3)$. Then the corrected coordinate x_f is given by

$$x_f = x' - 1 - \frac{t_2}{2t_1}, \tag{2}$$

where the vector $\mathbf{t} = (t_1, t_2, t_3)^T$ is the solution of the 3×3 linear system

$$\begin{bmatrix} 0 & 1 & 4 \\ 0 & 1 & 2 \\ 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} t_1 \\ t_2 \\ t_3 \end{bmatrix} = \begin{bmatrix} s_1 \\ s_2 \\ s_3 \end{bmatrix} \tag{3}$$

The same procedure is applied in y direction to obtain the corrected coordinate y_f.

A feature point $P(x_f, y_f)$ is then assigned a feature value

$$f(x_f, y_f) = \sqrt{\frac{\hat{s}(x', y')}{g_{max}} \cdot \frac{\hat{s}(x', y')}{s(x', y')}} \tag{4}$$

where $\hat{s}(x', y')$ is the mean grayvalue in the 3×3 neighbourhood of (x', y'), g_{max} the maximum possible grayvalue (e.g., 255).

Since $s(x', y')$ is a local maximum, $\hat{s}(x', y') \leq s(x', y')$. $f(x_f, y_f)$ characterises the 'strength' (dominance) of a feature so that large and bright particles have high strength. In (4), the first multiplier reflects the brightness, the second the size of a particle: when $\hat{s}(x', y')$ is close to $s(x', y')$, the top of the intensity surface is almost flat, which means a large particle. The features (particles) are ranked according to their dominance and the N_f most dominant features are selected for tracking.

The IPAN Tracker processes sequentially each three consecutive frames F_{n-1}, F_n and F_{n+1}, where n is the current frame. Consider a feature point in each of the 3 frames, $A \in F_{n-1}$, $B \in F_n$ and $C \in F_{n+1}$. When applied to PIV, the tracker minimises the cost function $\delta(A, B, C)$ which accounts for changes in velocity and appearance of a feature:

$$\delta(A, B, C) = w_1\Theta(A, B, C) + w_2\Lambda(A, B, C) + w_3\Phi(A, B, C),$$

where

$$\begin{aligned} \Theta(A, B, C) &= 1 - \frac{\overline{AB} \cdot \overline{BC}}{\left\|\overline{AB}\right\| \cdot \left\|\overline{BC}\right\|}, \\ \Lambda(A, B, C) &= 1 - \frac{2\left[\left\|\overline{AB}\right\| \cdot \left\|\overline{BC}\right\|\right]^{\frac{1}{2}}}{\left\|\overline{AB}\right\| + \left\|\overline{BC}\right\|} \end{aligned}$$

$\overline{AB}$ and $\overline{BC}$ are the vectors pointing from A to B and from B to C, respectively, $\overline{AB} \cdot \overline{BC}$ their scalar product. Θ and Λ, the trajectory smoothness terms, penalise changes in the direction and the magnitude of the velocity vector.

$\Phi(A, B, C)$, the appearance term of $\delta(A, B, C)$, accounts for changes in feature appearance in a 3×3 neighbourhood. It is defined as

$$\Phi(A, B, C) = \frac{1}{T}\sum_{k=1}^{9}\left[|s(A_k) - s(B_k)| + |s(B_k) - s(C_k)|\right],$$

where $s(P_k)$ are the grayvalues of the feature point P and its 8 neighbours, and $T = 18g_{max}$.

The constant weights balancing the components of the cost function are set as $w_1 = 0.1$, $w_2 = w_3 = 0.45$. An additional weighting function can be introduced to moderate the cost of small displacements. (See [5] for details.)

The cost function $\delta(A, B, C)$ is minimised using the repetitive hypothesis testing procedure presented in [5]. The number of operations is $O(N_f \cdot n_f^2)$, where n_f is the average number of features in the search area defined by the maximal possible displacement. At least three frames are needed for the operation of the algorithm.

3 Post-processing of velocity vector field

3.1 Coherence filtering and resampling

Feature trackers may occasionally yield completely wrong velocity vectors. To enhance the result of measurement, coherence based post-processing is applied to the 'raw' velocity field obtained by the trackers. The *coherence filter* modifies a velocity vector if it is inconsistent with the dominant surrounding vectors. The solution we use is a modified version of the vector median filter [1]. The algorithm operates as follows.

Given a feature point P_c with the velocity vector $\mathbf{v}_c$, consider all points P_i, $i = 1, 2, \ldots, p$, lying within a distance S from P_c, including P_c itself. Let their velocities be $\mathbf{v}_i$. Assuming that a majority of the vectors are similar, that is, form a cluster, we find the median $\mathbf{v}_{med}$ as the innermost point of the cluster. The innermost point is the point whose mean distance from the rest of the points is minimal. More formally, introduce

$$\hat{d}_i = \frac{\sum_{j \neq i} \|\mathbf{v}_i - \mathbf{v}_j\|}{p - 1} \tag{5}$$

Then the median index $med = \arg\min_i d_i$. $\hat{d}_{med}$, the mean distance from the median to the other points, characterises the size of the cluster. $\mathbf{v}_c$ is substituted by the median if the difference between $\mathbf{v}_c$ and $\mathbf{v}_{med}$ is significant:

$$\|\mathbf{v}_c - \mathbf{v}_{med}\| > \hat{d}_{med} \tag{6}$$

The standard median filter [1] tends to modify most of the measurements and introduce an additional error. The conditional median filter we use only modifies the vectors that are likely to be erroneous measurements. Our tests show that, as far as the overall accuracy is concerned, the conditional median is superior to the standard version.

The size of the neighbourhood, S, is adaptively set so as to have $p \geq 8$. Starting from a relatively small size, we locally increase S until the required number of neighbours is picked. The implementation of this procedure is discussed in [13].

The above coherence filter is applied iteratively until any of the stopping conditions is fulfilled. Denote by V_k the number of vectors modified at the k^{th} iteration. The conditions for stopping after the k^{th} iteration are as follows:

1. $V_k = 0$ OR
2. $V_k < V_{min}$ and $V_k > V_{k-1}$ OR
3. $k = k_{max}$

We use $V_{min} = 20$ and $k_{max} = 30$.

Uniform sampling of the measured velocity field is often required. A number of techniques [6] are available for *resampling* results of measurements. However, most of them perform resampling from one regular grid to another. We use the following procedure [13]. Given a point, G, on the required regular grid, consider all feature points P_i, $i = 1, 2, \ldots, p$, lying within a certain distance S from G. (S is selected adaptively in the way already discussed.) Let their velocity vectors be $\mathbf{v}_i$. Denote by d_i^2 the squared distance from P_i to G and by $\widehat{d^2}$ the mean of d_i^2 over all i. Introduce

$$\begin{aligned} \alpha_i &= \exp\Big(-\frac{d_i^2}{\widehat{d^2}}\Big), \\ \beta_i &= \frac{\alpha_i}{\sum_{i=1}^{k}\alpha_i}, \end{aligned}$$

where $0 < \alpha_i, \beta_i \leq 1$ and $\sum_{i=1}^{k}\beta_i = 1$. The interpolated velocity vector in G is calculated as

$$\mathbf{v}_G = \sum_{i=1}^{k}\beta_i\mathbf{v}_i \tag{7}$$

3.2 Handling flow discontinuities

The coherence assumption is violated in the vicinity of a flow discontinuity or any other significant inhomogeneity. Consider the hypothetic case of two interacting flows that move in opposite directions, as shown in figure 1. Near the border, the vector field is not coherent. The coherence filtering and the resampling are not applicable in the form just presented.

As a possible solution, we propose to apply the flexible neighbourhood assignment whose idea is sometimes used in texture segmentation. Instead of the traditional window centred on the point being considered, a five-window configuration is used which is illustrated in figure 1. (The index of a window is shown in the upper-left corner of the window.) Depending on the local conditions, the algorithm adaptively decides which of the 5 windows to assign to the point. The idea is that if the curvature of the border is not too high, at least one of the windows will be homogeneous. The coherence filtering and the resampling are based on that window, but the result is assigned to the central point.

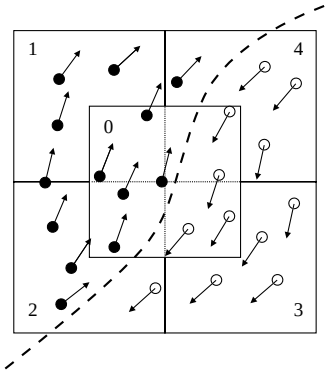


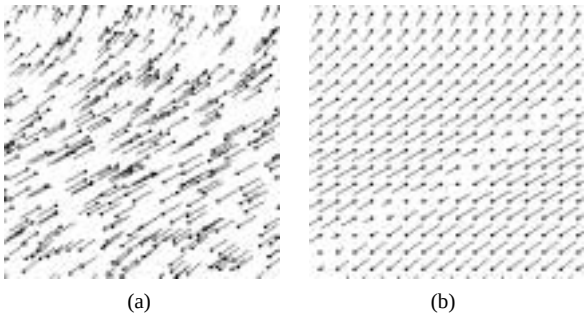
Figure 1: A hypothetic flow discontinuity and the five-window configuration.

Let W_n, $n = 0, 1, \ldots, 4$, denote the five windows. The central window W_0 will be called the *symmetric window*, the other four the *asymmetric* windows. The procedure of window selection starts with calculating $\mathbf{v}_{med}$ and $\hat{d}_{med}$ in each of the 5 windows. (See (5).) Denote the resulting 5 vectors by $\mathbf{v}_n$, the corresponding 5 differences by $\hat{d}_n$. An asymmetric window should only be used if W_0 is definitely inhomogeneous, since in normal circumstances the use of the symmetric window is preferable. To test the windows, the following measure of inhomogeneity is introduced:

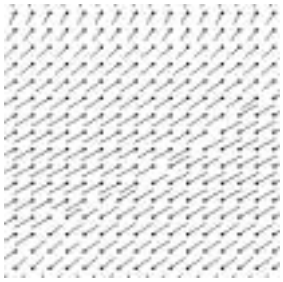
$$I_n = \frac{\hat{d}_n}{(\|\mathbf{v}_n\| + 1)} \tag{8}$$

If I_0 does not exceed a conservative threshold I_{thr}, W_0 is used. Otherwise, the five inhomogeneity measures I_n are compared and and the window with the least inhomogeneity is selected. (That is, W_0 may be selected in this case as well.)

The efficiency of the flexible neighbourhood assignment in velocity interpolation is illustrated in figure 2. We have synthesised a PIV sequence DSC with a flow discontinuity, whose velocity field is shown in figure 2a. When the original interpolation procedure is applied, the opposite vectors lying close to the border nullify each other and the result is distorted. The adaptive use of asymmetric neighbourhoods improves the quality of the interpolation.



(a) (b)



(c)

Figure 2: Resampling a vector field. (a) The original field. (b) Results of interpolation using only symmetric windows. (c) Results with asymmetric neighbourhoods.

4 Tests and discussion

Comparative experimental results of flow estimation are presented in the forthcoming paper [4], where the conventional symmetric window is applied in all cases. Standard PIV sequences which are commonly used to test the accuracy of the methods are not really suitable for testing the flexible neighbourhood assignment as they show relatively smooth flows without discontinuities.

Tables 1 and 2 compare the IPAN, the KLT, Quénot's enhanced algorithm ODP2 and a conventional correlation-based method DPIV32 for the synthetic flow sequence collection CYLINDER provided by G. Quénot [10]. N5 and N10 are noisy versions of the original noise-free flow N0 visualised in figure 3. The degrees of noise is 5% and 10%, respectively. The mean velocity of the flow is 7.58 pixel/frame. The mean absolute deviation from the ground truth displacement is given in pixels.

Table 1: Displacement errors for CYLINDER: trackers

	IPAN-NR	IPAN	KLT-NR	KLT
N0	0.36 ± 0.5	0.42 ± 0.5	0.28 ± 0.7	0.34 ± 0.6
N5	0.43 ± 0.7	0.49 ± 0.6	0.30 ± 0.7	0.35 ± 0.6
N10	0.52 ± 0.8	0.55 ± 0.7	0.29 ± 0.4	0.32 ± 0.4

Experimental data for ODP2 and DPIV32 are reproduced from [9]. ODP2 is the fastest of the 3 enhanced DP based optical flow algorithms discussed in [9]. DPIV32 is a conventional 32×32 window size correlator. For the feature trackers, results are given without (NR) and with resampling. The proposed controlled window selection mechanism was used.

For the CYLINDER sequences, only a minor improvement has been observed compared to the original results [4]. This is not sur-

Table 2: Displacement errors for CYLINDER: ODP2, DPIV32

	ODP2	DPIV32
N0	0.13 ± 0.1	0.55 ± 1.0
N5	0.21 ± 0.5	0.61 ± 1.2
N10	0.53 ± 1.4	0.77 ± 1.6

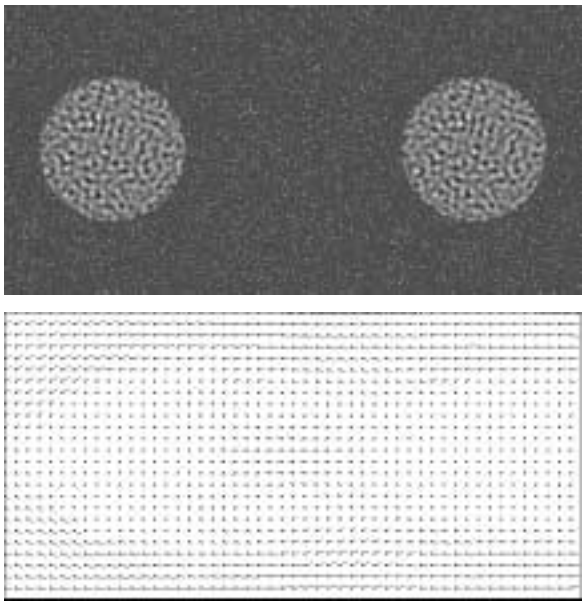


Figure 3: First frame and flow visualisation of sequence CYLINDER N0.

prising since these sequences contain only small vortices and no discontinuities. Results with and without resampling (NR) are reasonably close, which supports the proposed interpolation approach.

Results for the inhomogeneous synthetic flow DSC (figure 2a) are presented in table 3. The mean velocity of this flow is 7.61 pixel/frame. 'Conv' denotes the results obtained with the conventional symmetric assignment. Here, a noticeable improvement has been achieved by using the flexible window assignment.

Table 3: Displacement errors for DSC

	IPAN-NR	IPAN	KLT-NR	KLT
Conv	0.74 ± 2.7	0.91 ± 2.3	0.41 ± 1.1	0.67 ± 1.6
Flex	0.49 ± 1.7	0.74 ± 1.7	0.37 ± 0.3	0.62 ± 1.5

We have presented a novel approach to particle image velocimetry based on feature tracking algorithms developed in computer vision. This approach is a useful alternative to both the correlation-based and the DP-based techniques. It is prepared to cope with flow discontinuities. Feature trackers provide higher flow estimation accuracy and are faster than the conventional correlation-based techniques. (Typical running times are 20–40 sec for the trackers, 10 min for DPIV32 and 20 min CPU for ODP2.) For noisy images, the feature tracking PIV has been shown to compete in accuracy with the most precise DP based algorithms while requiring much less computation.

Online demonstration of the feature tracking PIV algorithms is available on the Internet at the web site of the IPAN group: `http://visual.ipan.sztaki.hu`. One can select an algorithm, set the parameters and run the algorithm on a short PIV sequence.

Acknowledgement. This work is partially supported by the Hungarian National Science Foundation under the grant OTKA T026592.

References

[1] J. Astola, P. Haavisto, and Y. Neuvo. Vector Median Filters. *Proceedings of the IEEE*, 78:678–689, 1990.

[2] J.L. Barron, D.J. Fleet, and S.S. Beauchemin. Performance of optical flow techniques. *International Journal of Computer Vision*, 12:1:43–77, 1994.

[3] Stan Birchfield. KLT: An Implementation of the Kanade-Lucas-Tomasi Feature Tracker. `http://vision.stanford.edu/~birch/klt/`.

[4] D. Chetverikov, M. Nagy, and J. Verestóy. Comparison of Tracking Techniques Applied to Digital PIV. In *Proc. International Conf. on Pattern Recognition*, 2000, to appear.

[5] D. Chetverikov and J. Verestóy. Feature Point Tracking for Incomplete Trajectories. *Computing*, 62:233–242, 2000.

[6] A.S. Glassner. *Principles of Digital Image Synthesis*. Morgan Kaufmann, 1995.

[7] I. Grant. Particle image velocimetry: a review. *Proc. Institution of Mechanical Engineers*, 211 Part C:55–76, 1997.

[8] B. Jähne. *Digital Image Processing*. Springer, 1997.

[9] G. Quénot, J. Pakleza, and T. Kowalewski. Particle image velocimetry with optical flow. *Experiments in Fluids*, 25:177–189, 1998.

[10] G. Quénot. Data and procedures for development and testing of PIV applications. `ftp://ftp.limsi.fr/pub/quenot/opflow/`.

[11] J. Shi and C. Tomasi. Good features to track. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR94), Seattle*, June 1994.

[12] P.T. Tokumaru and P.E. Dimotakis. Image correlation velocimetry. *Experiments in Fluids*, 19:1–15, 1995.

[13] J. Verestóy, D. Chetverikov, and M. Nagy. Digital particle image velocimetry: a challenge for feature based tracking. *Machine Graphics & Vision*, 8:553–570, 1999.