Strojniški vestnik - Journal of Mechanical Engineering 60(2014)7-8, 483-494
DOI:10.5545/sv-jme.2013.1614
Original Scientific Paper
Received for review: 2013-12-18
Received revised form: 2014-03-06
Accepted for publication: 2014-03-11

Flow Image Velocimetry Method Based on Advection-Diffusion Equation

Benjamin Bizjan[1,2,*] – Alen Orbanić[2,3] – Brane Širok[1] – Tom Bajcar[1] – Lovrenc Novak[1] – Boštjan Kovač[2,3]

[1] University of Ljubljana, Faculty of Mechanical Engineering, Slovenia
[2] Abelium d.o.o. – Research and Development, Slovenia
[3] University of Ljubljana, Faculty of Mathematics and Physics, Slovenia

This paper presents a non-contact method for velocity field calculation from a series of fluid flow images with illuminated planar layer of the flow and a mixed-in pollutant for flow visualization. The velocity field is calculated using a model similar to optical flow that is based on the advection-diffusion equation. The model is evaluated using a set of synthetic airfoil flow visualization images generated by a combination of computational fluid dynamics and inverse advection-diffusion equation approaches. The calculated velocity fields are in a good agreement with the true velocity fields.

Keywords: computer-aided visualization, image velocimetry, dense flow field calculation, optical flow, advection-diffusion equation, computational fluid dynamics

0 INTRODUCTION

Flow visualization is a method for observation and monitoring of fluid flow structures in which we make flow structures visible by adding a pollutant such as a dye or a smoke to the fluid, unless the structures are already visible by themselves. Using high-speed cameras one is able to collect a large amount of spatial flow structure data in a very short time. Image data is further processed to determine velocity fields and other flow properties.

The main advantage of flow visualization methods is their very low impact on the flow dynamics as they are mostly non-contact methods. This is especially advantageous in processes with harsh measurement conditions (hot or corrosive fluids, etc.) where the choice of conventional measurement methods is very limited. Širok et al. [1] used the visualization principle to measure melt mass flow in mineral wool production. Another field of application includes large hydraulic structures such as irrigation and flood management systems where local flow velocities are of interest and would not be practical to be measured with conventional flow metering devices or simulated adequately enough due to potential complex flow phenomena [2]. Recently, Novak et al. [3] and [4] used flow visualization to study flow phenomena in side weirs. Furthermore, visualization methods are also preferred for multiphase flow analysis, especially in biological and chemical engineering, where other methods are too intrusive or do not allow measurements on the micro scale. For example, Bajcar et al. [5] and [6] used visualization techniques to study sedimentation efficiency in circular settling tanks.

There are many principles of flow visualization and a number of flow properties that can be assessed. In this paper however we will focus on flow velocity measurement in incompressible flows. Local fluid velocities manifest themselves indirectly through movement of the brightness patterns in the image, known as the optical flow.

Numerous different approaches for optical flow computation have evolved so far. Horn and Schunck [7] developed an early optical flow calculation method based on image brightness analysis and an assumption of smooth velocity variations using a quadratic penalization scheme. Through the years, optical flow computation methods have been improved significantly in both accuracy and robustness. For example, Black and Anandan [8] improved the penalization term to make the algorithm less sensitive to noise and occlusions. Optical flow algorithms have been expanded further by the introduction of a temporal in addition to a spatial smoothing term, which allowed for use on image sequences longer than two frames. Flow-driven spatiotemporal smoothing term was used successfully by Brox et al. [9] and Bruhn and Weickert [10]. A detailed overview of optical flow analysis methods and their performance was provided by Barron et al. [11] and, more recently, by Zimmer et al. [12].

While the majority of the earlier algorithms were primarily intended to be used for analysis of quasi-rigid motion with low divergence and practically no diffusion, in recent years significant research has been carried out on the use of optical flow for fluid flow visualization. Corpetti et al. [13] analyzed fluid flows using a minimization-based motion estimator based on a second order div-curl regularization and

*Corr. Author's Address: Abelium d.o.o., Kajuhova 90, 1000 Ljubljana, Slovenia, benjamin.bizjan@abelium.eu

on optical intensity continuity equation rather than the assumption of intensity conservation. Bajcar et al. [14] presented a fluid flow visualization method based on the advection-diffusion equation. This method was used on experimental fluid flows with added passive tracer (smoke). All fluid flow analysis methods we have listed so far are two-dimensional. Recently however, three-dimensional approaches have become increasingly popular. For instance, Regart et al. [15] extended the 2D method by Corpetti et al. [13] to be used on 3D image sequences.

The majority of the models discussed lack the dedicated software solutions that could be used more generally. The purpose of our work was to develop a robust, software-supported visualization method that could be used on a wide variety of real lab- and industrial environment phenomena with a reasonable accuracy.

This paper is organized as follows. In Section 1 a theoretical formulation of our approach to velocity field calculation is presented. Section 2 describes the setup for generating a synthetic image sequence to be used for the evaluation of our method. Section 3 presents the method evaluation procedure on the generated image set while Section 4 provides calculation results and optimal calculation settings are determined. In Section 5 we list our conclusions and indicate the possibilities for further developments of our method.

1 THEORETICAL BACKGROUND

Our method is intended for approximate velocity field calculation from an image sequence of an incompressible fluid flow and has been implemented in our software package *ADMflow* (http://admflow.net). The method is based on the assumption that the concentration N of the pollutant at time t is proportional to the image brightness. The pixel at position (i,j) for image A at time t is characterized by its brightness (grayscale level) $A(t,i,j)$ ranging from 0 (black) to 255 (white).

Our system of equations for velocity field calculation is derived from the advection-diffusion equation:

$$\frac{\partial N}{\partial t}+\frac{\partial (Nv_x)}{\partial x}+\frac{\partial (Nv_y)}{\partial y}=D\left(\frac{\partial^2 N}{\partial x^2}+\frac{\partial^2 N}{\partial y^2}\right). \quad (1)$$

In Eq. (1) D is the diffusivity coefficient, and v_x and v_y are the components of the velocity vector $\mathbf{v} = (v_x, v_y)$. Since we are dealing with incompressible fluids, hence div($\mathbf{v}$) = 0, we can further simplify Eq. (1):

$$\frac{\partial N}{\partial t}+\frac{\partial N}{\partial x}v_x+\frac{\partial N}{\partial y}v_y=D\left(\frac{\partial^2 N}{\partial x^2}+\frac{\partial^2 N}{\partial y^2}\right). \quad (2)$$

Note that in this form with the right side set to 0, the equation is identical to the one of the optical flow as proposed by Horn and Schunck [6]. Derivatives are discretized using the central difference method:

$$\frac{\partial f}{\partial x_i}(x)\approx\frac{f(x+\Delta x_i)-f(x-\Delta x_i)}{2\Delta x_i}, \quad (3)$$

$$\frac{\partial^2 f}{\partial x_i^{\,2}}(x)\approx\frac{f(x+\Delta x_i)-2f(x)+f(x-\Delta x_i)}{\Delta x_i^{\,2}}. \quad (4)$$

Let $v_x(t,i,j)$ and $v_y(t,i,j)$ denote velocity components at pixel (i,j) of the fluid at time t. Time t is advanced in discrete steps, $t = \{1,2,3,\ldots,R\}/f$, where R is the number of images in the image sequence and f the number of frames per second. The integers Δt and Δxy represent the time and space displacements in the number of images in the series and the number of pixels in an image, respectively. The pixel size in meters will be denoted by s.

$$\begin{aligned}&\frac{A(t+\Delta t,i,j)-A(t-\Delta t,i,j)}{2D\cdot\Delta t/f}+\\&+\frac{A(t,i+\Delta xy,j)-A(t,i-\Delta xy,j)}{2D\cdot\Delta xy\cdot s}v_x(t,i,j)+\\&+\frac{A(t,i,j+\Delta xy)-A(t,i,j-\Delta xy)}{2D\cdot\Delta xy\cdot s}v_y(t,i,j)=\\&=\frac{A(t,i+\Delta xy,j)-2A(t,i,j)+A(t,i-\Delta xy,j)}{(\Delta xy\cdot s)^2}+\\&+\frac{A(t,i,j+\Delta xy)-2A(t,i,j)+A(t,i,j-\Delta xy)}{(\Delta xy\cdot s)^2}. \quad (5)\end{aligned}$$

Using Eq. (5) we obtain one linear equation and two unknowns, namely $v_x(t,i,j)$ and $v_y(t,i,j)$, for each particular image t at the particular pixel (i,j). To differentiate between the actual velocities and the calculated ones, we shall use as the unknowns in the equations $u_x(t,i,j)$ and $u_y(t,i,j)$, respectively, and u will denote the calculated velocity. Equations can be obtained for all $t<R-1$ and all the pixels in the images which are not within the Δxy margin from the image borders. Clearly we get more unknowns than the equations, which is not surprising as our method basically solves Eq. (1), but without boundary

conditions. To increase the number of equations in the model we use two approaches which can yield additional equations and the solution of the system is then obtained using the least squares method.

In the first approach called *temporal smoothing*, each triplet of images (t–$\Delta t, t, t$+Δt) introduces a set of equations with unknowns $u_x(t,i,j)$ and $u_y(t,i,j)$. Taking the series of triplets on images t–Δt–$|l/2|$, t–Δt–$|l/2|$ +1, …, t+Δt+$|l/2|$–1, we yield l sets of equations where all $u_x(t+r,i,j)$ and $u_y(t+r,i,j)$ for $r \in \{-\lfloor \frac{l}{2} \rfloor, -\lfloor \frac{l}{2} \rfloor + 1, ..., \lceil \frac{l}{2} \rceil - 1\}$ are replaced by $u_x(t,i,j)$ and $u_y(t,i,j)$, respectively. The number of image triplets l used in temporal smoothing will be given by the parameter *IIE* (*images in equations*).

The second approach focuses on the penalization of rapid changes in the velocity field and has a similar effect as the regularization functional in the Horn and Schunck algorithm [7]. For each velocity vector $\mathbf{u} = (u_x(t,i,j), u_y(t,i,j))$, we add the linear equations of the form $\beta \cdot (\text{grad } u_x) = 0$ and $\beta \cdot (\text{grad } u_y) = 0$ where we use Eq. (3) for the discretization with the displacement of 1 pixel (note: this may be different from the Δxy used in discretization). The coefficient $\beta \geq 0$ determines the amount of penalization of the spatial changes of the velocity field in the image - the larger the value, the more the change is penalized. This approach is called *spatial smoothing*.

Similarly to the velocity gradient penalization, other flow field properties can be penalized as well. Flow divergence is regulated by equations $\gamma \cdot (\text{grad } u_x) = 0$ and $\gamma \cdot (\text{grad } u_y) = 0$ and flow curl by eqs. $\delta \cdot (\text{curl } u_x) = 0$, $\delta \cdot (\text{curl } u_y) = 0$, $\theta \cdot (\text{grad}(\text{curl } u_x)) = 0$ and $\theta \cdot (\text{grad}(\text{curl } u_y)) = 0$.

In an ideal case, two sequential images would be infinitesimally separated in the time domain and the spatial changes in the image would also be infinitesimal. To achieve as fine a discretization as possible, Δt and Δxy would therefore have to be 1. However, when the spatial movements between two sequential images are large, this choice does not yield adequate results. Improved results are achieved by downsampling images by a coefficient k, which first smooths the image by filters and then takes every k^{th} pixel in horizontal and vertical direction. A direct downsampling would yield a much sparser velocity field, resulting in k^2 times fewer vectors. In the proposed model, we carry out downsampling implicitly, by first smoothing the image A by box filter and then, instead of the direct downsampling, we use $\Delta xy = k$, thus still obtaining dense velocity fields and better results with respect to the ground truth, i.e. the real velocity fields. We use box filters with integer parameter *SWS (smoothing window size)*, where for each image pixel the average of the pixels in the square of size $2 \cdot SWS+1$ centered in the pixel is calculated. The *SWS* parameter regulates image gray level smoothing and is not to be confused with β, which smoothes (penalizes) velocity gradients. Optimal Δxy and *SWS* settings depend on the average flow feature displacement between consecutive images. If ζ_x and ζ_y are the horizontal and vertical displacement components, respectively, then we assume that the relevant displacement to be taken in consideration is the maximum value of both, $\zeta = \max(\zeta_x, \zeta_y)$. We intend to find the optimal ratio for $\Delta xy/\zeta$ and SWS/ζ.

2 SYNTHETIC IMAGE GENERATION

For the purpose of our velocimetry method evaluation, we numerically generated a synthetic image sequence of smoke-traced air flow over a scaled-down NACA4421 airfoil in a low-speed wind tunnel. Due to low speeds, air flow can be treated as incompressible, allowing us to use our velocimetry method without modifications. We previously used experimentally obtained wind tunnel visualization images with hotwire-measured velocities to test the performance of our method. While such experimental images are a typical example of an industrial application, there is one significant drawback, namely the fact that the true velocity fields (the ground truth) are unknown as they can never be measured with total precision regardless of the choice of measurement method.

To overcome this limitation and thus ensure that the deviation of our method's results from the ground truth is entirely a consequence of the method calculation error, a synthetic image sequence was produced. First, the airfoil was modeled in 3d modeling software, then meshed and imported into the *Ansys Fluent* computational fluid dynamics (CFD) software. A simulation of air flow over the airfoil was performed and the obtained velocity fields were exported. These velocity fields were then used in our software *ADEsolver* where smoke flow was visualized using an inverse advection-diffusion equation approach.

2.1 CFD Simulation

The geometry of the numerical model was made to represent the actual wind tunnel where experiments had previously been conducted. Numerical simulations were performed for the airfoil NACA4421 with a chord of length 30 mm and oriented at a 3° angle of attack. The three-dimensional computational

domain (Fig. 1) covered the entire height of the wind tunnel (100 mm) in the direction normal to the airfoil (y coordinate) and had a length of 200 mm in the streamwise direction (x coordinate) with the airfoil leading edge at x = 50 mm.

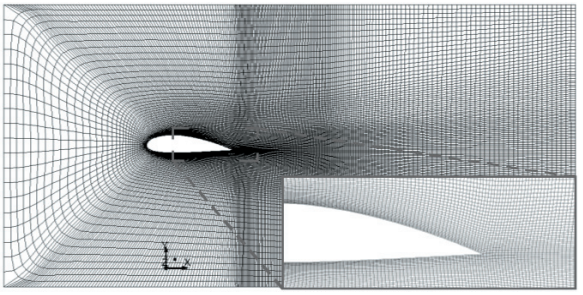

Fig. 1. *Computational grid for the numerical simulation in Fluent*

Top and bottom of the domain were defined as no slip walls since they represented the actual wind tunnel walls. Periodic boundary conditions were imposed on the spanwise boundaries. Inlet velocity profile was set accordingly to the experimentally measured data, obtained with a hot-wire anemometer at 18 equally spaced locations across the channel height (average inlet velocity was 4.945 m/s). Turbulence intensity of 1.5% was computed from the hotwire measurements and set as inlet turbulence, while the inlet turbulent length scale of 5 mm was estimated based on the upstream flow straightener geometry. Outlet was defined as a pressure outlet with constant relative pressure of 0 Pa. A time step of $4.11 \cdot 10^{-5}$ s was chosen taking in consideration both CFD calculation stability (namely, the CFL number which ranged 0.3 and 1.8 in the wake region, close to the recommended value of unity) and the flow feature displacement between images (ζ) which ranged between 1 and 5 pixels, assuring a good performance of our velocimetry method in all flow regions.

A combination of two turbulence models was used in the simulation. Initial conditions were provided by the steady state solution using the Shear stress transport (SST) model with the low Reynolds number correction enabled. Then, a transient simulation was run using the SST-SAS (SST-Scale-adaptive simulation) model. Both models were used with the default coefficients as set in *Ansys Fluent 13* [16]. The SST model is a two-equation eddy viscosity turbulence model with improved prediction of separation compared to earlier models such as the k-ε model, and is one of the most frequently used RANS (Reynolds-averaged Navier-Stokes) models for aerodynamics simulations [17].

While the SST model provides a good agreement with measured data for mean flow at reasonable computational expense, it fails to give sufficient information on turbulent structures. For this purpose, there are several different approaches that can yield more detailed results. A good compromise between calculation time and accuracy is to use one of the hybrid RANS-LES models such as the scale-adaptive simulation (SAS). The SAS model was developed by Menter and Egorov [18] and is in fact an improved unsteady-RANS (URANS) model with LES capability in unstable flow regions. The SAS model is based on introduction of the von Karman length scale into the turbulent length scale equation of a two-equation turbulence model, allowing for local detection of unsteadiness and automatic balancing between contributions of modeled and resolved turbulence stresses.

Air flow velocity fields were exported for a zoomed-in view defined by a planar cross-section of the computational domain in the streamwise direction (spanwise coordinate z = –0.05 m) Fig. 2. A sequence of 100 consecutive grayscale contour plots sized 960×720 pixels were exported for both x and y velocity and the physical size of one pixel was s = 45.5 μm. These velocity contour plots were later imported into the *ADEsolver* software as the ground truth data (v_x, v_y) for flow tracer (smoke) visualization.

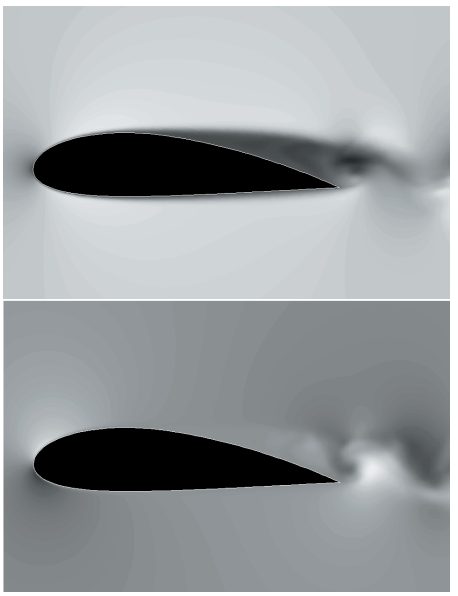

Fig. 2. *An example of CFD-calculated x- (upper image) and y- (lower image) velocity contour plots around the airfoil; colormap range (black to white color) is [–3, 8] m/s for v_x and [–6, 5] m/s for v_y*

2.2 Inverse Advection-Diffusion Equation for Pollutant Simulation

In this step, the *x* and *y* velocity fields calculated in *Fluent* were imported into our *ADEsolver* software along with the pollutant (smoke) sources defined as matrices of pollutant concentration (gray level) *A* with a range from 0 (zero source intensity) to 255 (maximum source intensity) in integer increments. There was one source matrix for every time step with three sources (nonzero concentration areas) placed at different locations near the airfoil upper and lower surfaces. To ensure that the advection-diffusion equation based velocimetry method that was to be tested on generated synthetic images could function properly, gray level spatial and temporal gradients had to be sufficiently large. For this reason, the source intensity A_S was not constant, but varied between $A_{S,\min} = 100$ and $A_{S,\max} = 255$, accordingly to Eq. (6) for all three sources.

$$A_S(n) = A_{S,\min} + \frac{1}{2}\left(A_{S,\max} - A_{S,\min}\right)\cdot\left(1+\sin\left(\frac{2\pi}{t_0}n\right)\right). \quad (6)$$

In Eq. (6), t_0 is the sine wave period given in number of frames (images) between consequent concentration maxima (t_0 = 20 was used). The independent variable n is the discrete time which is equal to the number of the current frame (n = 1, 2, 3, ..., R).

Once all the v_x, v_y and A_S data were imported in *ADEsolver*, the pollutant concentration A was calculated for every pixel in the calculation area (960×720 pixels, the same as velocity calculation area in *Fluent*) using the simplified advection-diffusion equation given by Eq. (2). This differential equation was solved iteratively using the iterative equation (Eq. (7)), also known as the Jacobi iteration. With this method, the calculation using Eq. (7) is repeated several times using the results from the previous iteration until the solution has converged sufficiently. In each iteration, the current time t is advanced by the time step dt and the new pollutant concentration is calculated for every pixel in the calculation domain. The time step dt is typically much smaller than the time between two consecutive frames ($\delta t = 1/f$) in order to ensure stable convergence and the total number of iterations before the next frame is $\delta t/dt$.

$$\begin{aligned} A(t+dt,i,j) = A(t,i,j) + dt\cdot D\cdot[A(t,i+1,j) + \\ +A(t,i,j+1) + A(t,i-1,j) + A(t,i,j-1) - \\ -4A(t,i,j)]/s^2 - adx(t,i,j) - ady(t,i,j). \end{aligned} \quad (7)$$

Diffusivity D was estimated to 10^{-6} m^2/s by visual comparison of simulated pollutant flow to the smoke flow from the actual experiment, meaning advection rather than diffusion was the predominant mechanism of pollutant spreading. The terms *adx* and *ady* represent the pollutant concentration change due to convection in *x* and *y* direction, respectively, and are defined using the upwind advection scheme [19] – Eqs. (8) and (9).

$$\begin{aligned} adx(t,i,j;v_x>0) &= v_x(n\delta t,i,j)\cdot dt\cdot(A(t,i,j) - \\ &\quad - A(t,i-1,j))/s, \\ adx(t,i,j;v_x<0) &= v_x(n\delta t,i,j)\cdot dt\cdot(A(t,i+1,j) - \\ &\quad - A(t,i,j))/s. \end{aligned} \quad (8)$$

$$\begin{aligned} ady(t,i,j;v_y>0) &= v_y(n\delta t,i,j)\cdot dt\cdot(A(t,i,j) - \\ &\quad - A(t,i,j+1))/s, \\ ady(t,i,j;v_y<0) &= v_y(n\delta t,i,j)\cdot dt\cdot(A(t,i,j) - \\ &\quad - A(t,i,j-1))/s. \end{aligned} \quad (9)$$

Pollutant concentration field images were generated using the following procedure. First, we ran 200 time steps δt of simulation for constant ground truth v_x and v_y (the first from the 100 frame sequence acquired in *Fluent*) to obtain a steady state solution. Then, a simulation of another 100 time steps in length was calculated using variable ground truth fields, with n^{th} v_x and v_y fields being used for the calculation of n^{th} time step pollutant concentration. For each time step, the pollutant concentration field was saved to an image of the same size as the *Fluent* velocity field images (960×720 pixels) to represent a synthetic flow visualization image that was to be used in *ADMflow*. A section of a typical image is shown in Fig. 3.

3 METHOD EVALUATION PROCEDURE

For evaluation of our velocimetry method, a sequence of 10 images was selected from the 100-image synthetic flow visualization sequence generated in *ADEsolver*, along with the corresponding ground truth velocity fields calculated in *Fluent*. The main selection criterion was to attain a good match in position of turbulent structures between *Fluent* and *ADEsolver* at the beginning of the chosen sequence and to find as many characteristic flow regions as possible. We identified three main regions of interest (Fig. 3). Region 1 was placed just behind the airfoil's trailing edge where a small, slowly moving vortex was observed. In region 2, vortex shedding, namely the von Karmann vortex street occurred as a result of a low Reynolds number and the shape of the airfoil

and manifested itself as an oscillating texture pattern. Region 3 represents the laminar flow over the airfoil above the boundary layer zone where the highest velocities occurred. Windows shown by white squares were used as testing areas for our velocimetry method implemented in *ADMflow*. The sizing of the windows 1, 2 and 3 was 50×50, 50×50 and 40×40 pixels, respectively.

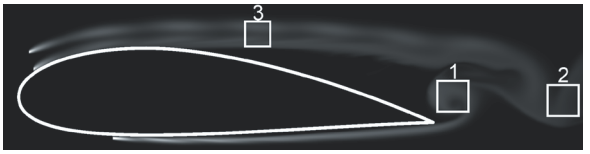

Fig. 3. *A section of a synthetic flow visualization image with the three distinctive flow regions and corresponding windows where our velocity calculation method was tested*

Using the synthetic flow visualization sequence, velocity fields (u_x and u_y) were calculated in *ADMflow* in windows 1, 2 and 3 and compared to the actual velocity fields (ground truth) of the set, using the [m/s] unit for both. In order to evaluate our method's accuracy, calculation errors are defined by equations (10) to (16) and an overview of the evaluation methodology is given in Fig. 4.

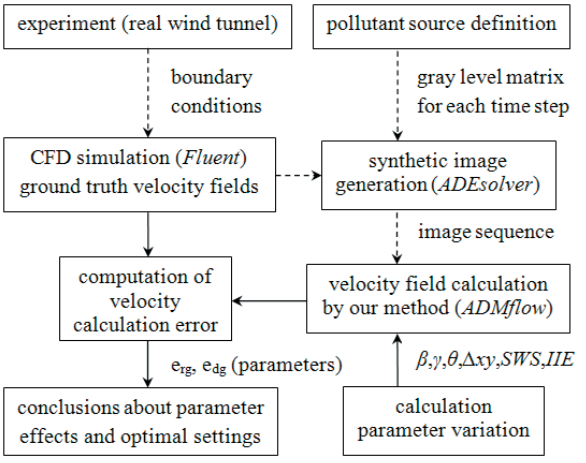

Fig. 4. *Flow chart of our velocimetry method evaluation procedure*

The velocity field calculation error vector is introduced as the difference between calculated and true velocity fields, *u* and *v,* respectively. Local absolute error vector and magnitude at the pixel location (i, j) and time t are given by Eq. (10) and (11), respectively:

$$\mathbf{e}_a(t,i,j) = (e_{a,x}(t,i,j), e_{a,y}(t,i,j)) = \\ = (u_x(t,i,j) - v_x(t,i,j), u_y(t,i,j) - v_y(t,i,j)), \quad (10)$$

$$e_a(t,i,j) = \sqrt{e_{a,x}(t,i,j)^2 + e_{a,y}(t,i,j)^2}. \quad (11)$$

To make a comparison between the results obtained in different flow regions easier, local relative error e_{rl} is introduced as a ratio between the absolute error magnitude and the average wind tunnel inlet velocity ($V = 4.945$ m/s):

$$e_{rl}(t,i,j) = \frac{e_a(t,i,j)}{V}. \quad (12)$$

Spatial relative error average of $e_r(t,i,j)$ over the calculation domain Ω and at time t is given as:

$$e_{rs}(t) = \frac{1}{n} \sum_{(i,j)\in\Omega} |e_{rl}(t,i,j)|. \quad (13)$$

In Eq. (14), n is the number of points in the calculation domain Ω. Finally, we define the global relative error e_{rg} as a spatiotemporal average of the local relative error:

$$e_{rg} = \frac{1}{n \cdot T} \sum_{t=1/f}^{T=R/f} \sum_{(i,j)\in\Omega} |e_{rl}(t,i,j)| = \frac{1}{T} \sum_{t=1/f}^{T=R/f} e_{rs}(t). \quad (14)$$

Analogously to the velocity magnitude relative error e_{rg}, the x and y velocity relative errors e_{rgx} and e_{rgy} can be defined.

In addition to the velocity magnitude error, the velocity directional error is introduced as the angle φ between the calculated and true velocity vector ($-\pi<\varphi<\pi$), as given by Eq. (15) and (16). The local directional error at location (i, j) and time t is denoted by e_{dl}.

$$\begin{aligned} e_{dl}(t,i,j;|a\tan 2(u_y,u_x) - a\tan 2(v_y,v_x)| < \pi) = \\ = a\tan 2(u_y,u_x) - a\tan 2(v_y,v_x), \\ e_{dl}(t,i,j;|a\tan 2(u_y,u_x) - a\tan 2(v_y,v_x)| > \pi) = \\ = a\tan 2(u_y,u_x) - a\tan 2(v_y,v_x) - 2\pi, \\ e_{dl}(t,i,j;|a\tan 2(u_y,u_x) - a\tan 2(v_y,v_x)| < -\pi) = \\ = a\tan 2(u_y,u_x) - a\tan 2(v_y,v_x) + 2\pi. \end{aligned} \quad (15)$$

In Eq. (15), atan*2*(*y*,*x*) is the four-quadrant inverse tangent defined by Eq. (16):

$$a\tan 2(y,x) = \begin{cases} \arctan(y/x); & x > 0 \\ \arctan(y/x) + \pi; & y \geq 0,\ x < 0 \\ \arctan(y/x) - \pi; & y < 0,\ x < 0 \\ \pi/2; & y > 0,\ x = 0 \\ -\pi/2; & y < 0,\ x = 0 \\ \text{undefined}; & y = 0,\ x = 0 \end{cases}. \quad (16)$$

The spatial and global (spatiotemporal) averages of directional error (e_{ds} and e_{dg}, respectively) are defined analogously to the magnitude error definitions in Eq. (13) and (14).

4 RESULTS AND DISCUSSIONS

In *ADMflow*, velocity fields in windows 1 to 3 were calculated for a number of different calculation parameter settings and compared to the ground truth fields using the magnitude and directional error definitions from the previous section. For every window, the optimal combination of calculation parameters was determined as the one where the calculation error was the lowest (Table 1).

Table 1. *Optimal parameter settings for velocity calculation windows and the corresponding calculation errors*

win.	β	γ	θ	Δxy	*SWS*	*IIE*	e_{rg} [%]	e_{dg} [°]
1	.002	.01	10^{-6}	2	2	2	13.8	15.2
2	.004	.01	10^{-6}	2	3	4	12.1	5.8
3	10^{-4}	.03	10^{-7}	3	3	8	4.1	0.5

Other calculation parameters were identical for all the regions of interest presented in Table 1. Diffusivity was set to the same value as in *ADEsolver* for pollutant simulation ($D = 10^{-6}$). The calculation time step was set to $\Delta t = 1$ as larger values resulted in excessive calculation errors. In Fig. 5, the velocity fields calculated by optimal settings given in Table 1 are compared to the true velocity fields of the image set.

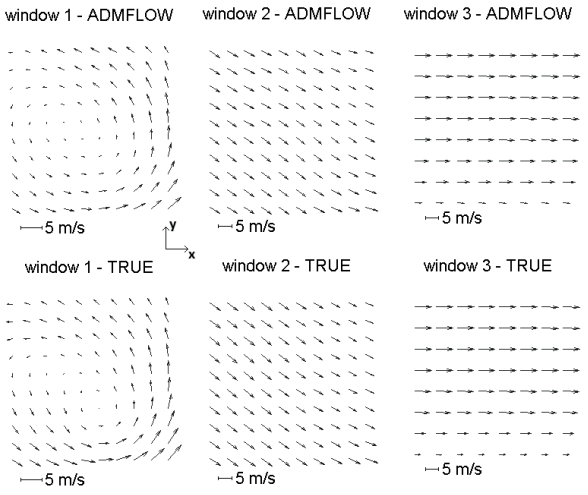

Fig. 5. *Comparison between calculated and true velocity fields for the first frame of the selected 10-image sequence*

From Fig. 5, a good agreement between the calculated and true velocity fields can be observed in terms of velocity magnitude as well as flow direction. In window 1, our calculation method was able to detect the vortex flow structure well, while at the bottom of window 3, a velocity drop near the boundary layer separation zone was also correctly identified. As we can see from Table 1, the largest velocity calculation error (about 14% in velocity magnitude and 13° in velocity direction) occurs in window 1. This can be largely attributed to a shift in the detected position of the vortex center (lowest velocity area) of the otherwise well reproduced vortex shape. The shift most likely occurs due to the fact that calculation parameters could only be optimized for window 1 as a whole and not separately for different parts of the vortex. Optimal calculation parameter values are largely dependent on the flow velocity magnitude, which is in fact much larger at the edge of the observed vortex than at its center as the vortex translatory motion is very slow and rotational motion is predominant.

In window 2, which is located further downstream in the von Karmann vortex street zone, the calculation error is already smaller (12% in velocity magnitude and 6° in velocity direction), especially the directional error. The error here largely depends on the flow visualization quality and may rise significantly if the pollutant concentration gradient becomes too low, especially if the concentration drops to near zero values.

The lowest calculation error was observed in window 3 both in terms of velocity magnitude (4%) and especially in the flow direction (only 0.5°). The flow in the window is almost steady, but periodic oscillations in pollutant concentration allow the advection-diffusion algorithm to perform properly. This is true even for the area near the boundary layer separation zone where flow velocity is much lower but still accurately calculated.

Now, let us assess the effect of individual calculation parameters. In Fig. 6a and b, velocity magnitude and directional errors are shown as a function of parameters *IIE* and β. Other calculation parameters were set to the values given in Table 1.

From Fig. 6a and b it can be observed that the effect of temporal smoothing (parameter *IIE*) is predominant while for $IIE \geq 2$, temporal smoothing (parameter β) has a relatively low impact on the calculation error. For windows 1 and 2, the lowest velocity magnitude and directional errors occur at $IIE = 2$ to 3 and $\beta = 0.001$ to 0.004 and for window 3 at $IIE = 5$ to 8 and $\beta = 0$ to $3 \cdot 10^{-4}$. Due to the higher error sensitivity to β at $IIE = 1$, only values of $IIE \geq 2$ should be used. On the other hand, using too high *IIE*

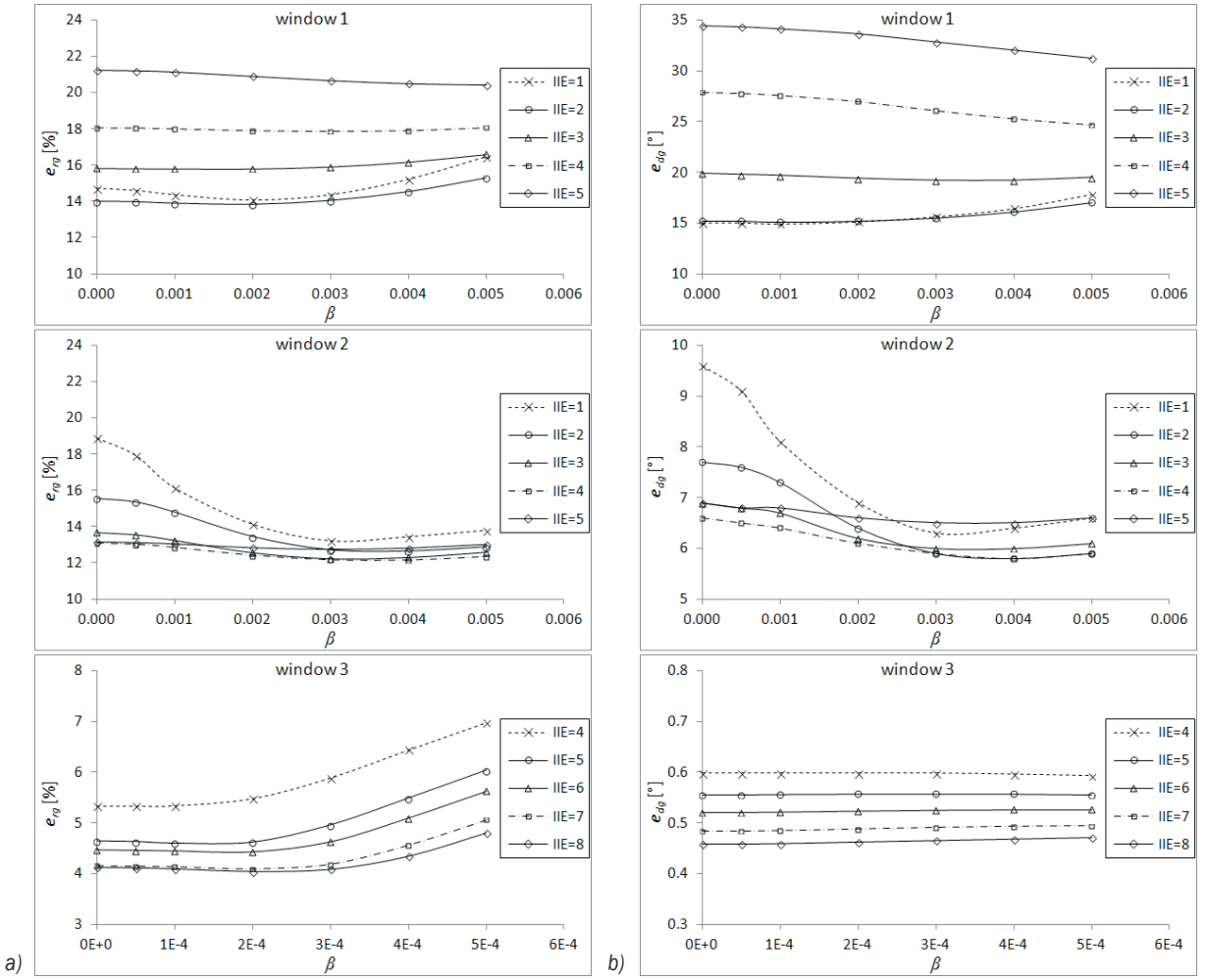

Fig. 6. *Effect of parameters IIE and β on a) velocity magnitude error; and b) velocity directional error*

values for laminar flows such as the flow in window 3 is not advisable as the instantaneous velocity information is lost due to excessive time averaging and a good choice would therefore be $IIE \approx 5$. While the calculation error at $IIE = 8$ is slightly lower than at $IIE = 5$, the difference is marginal (0.5% for velocity magnitude and 0.1° for direction, respectively).

Regarding parameter β, a nonzero setting is advisable as some flow image sets may produce excessive local errors for $\beta = 0$ but perform well above 10^{-3} to 10^{-2}, with smooth, almost steady flows demanding lower β than the more unsteady flows (e.g. vortex shedding). The minimum required β also depends on the image signal-to-noise ratio - higher the ratio, lower the needed β. Signal-to-noise ratio (*SNR*) in our synthetic image set is high due to the methodology used to visualize smoke - image gray level is defined by the calculated pollutant concentration, while for real experiments, images are recorded with a camera and a higher level of noise may be present due to the non-ideal lighting, lenses and smoke generation. Regardless of the *SNR*, values of $\beta >> 0.01$ may cause the spatial smoothing effect to become too pronounced, filtering out not only the noise, but also the relevant flow structures.

In addition to *IIE* and β, another two very important parameters are Δxy and the smoothing window size (*SWS*). The impact of these two parameters on velocity calculation error is shown in Figs. 7a and 7b.

From Figs. 7a and b we can see that the effect of smoothing window size (parameter *SWS*) is predominant while, for $SWS \geq 2$, downsampling (parameter Δxy) has a relatively low impact on the calculation error. For windows 1 and 2, the lowest velocity magnitude and directional errors occur at SWS and Δxy in range of approximately 2 to 4. For window 3, the optimal parameter range is similar with

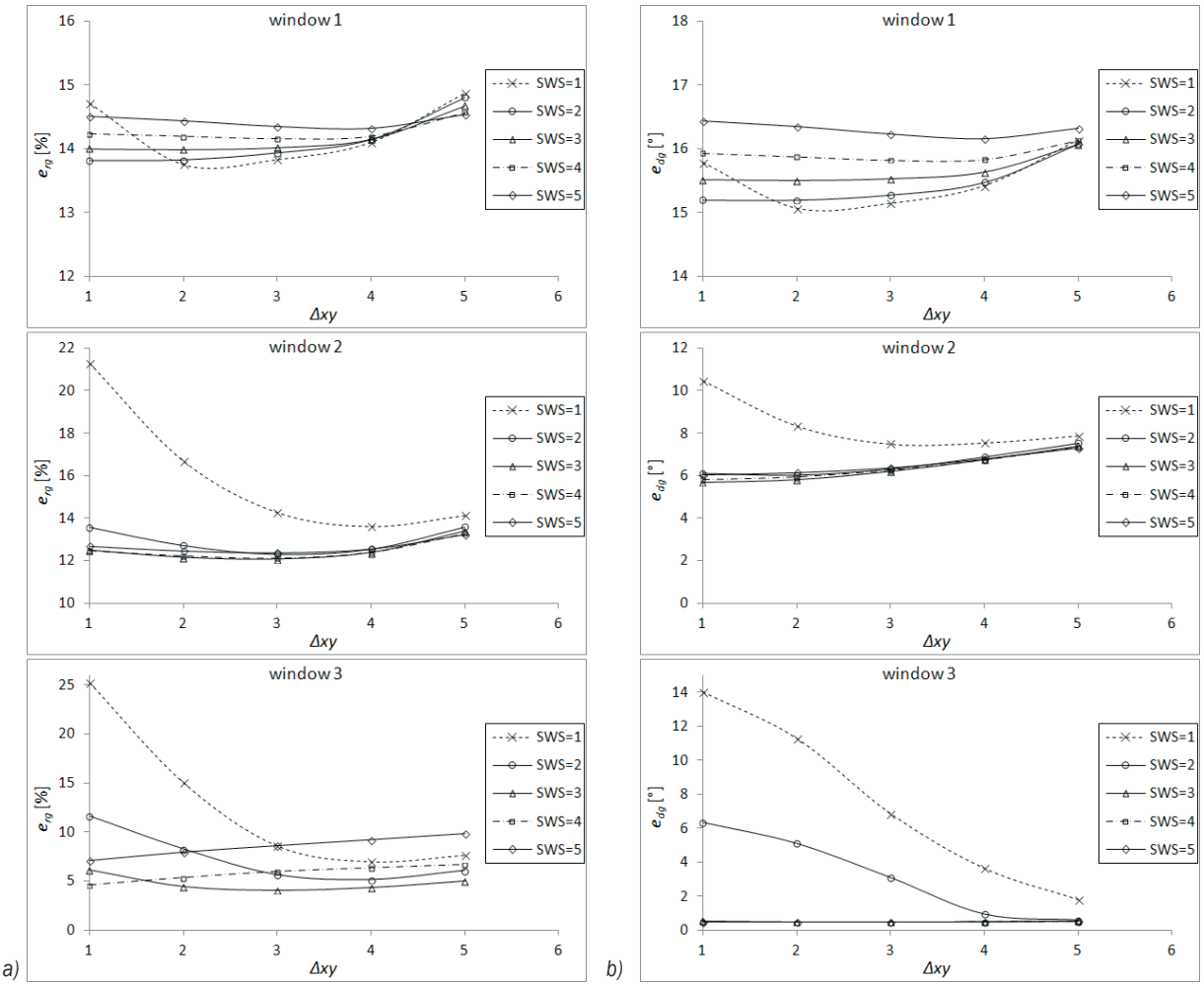

Fig. 7. *Effect of parameters Δxy and SWS on a) velocity magnitude error; and b) velocity directional error*

the exception that the velocity directional error is large for $SWS < 3$ and becomes very low and practically independent of the smoothing window size for $SWS \geq 3$. Due to the higher error sensitivity to *SWS* at $IIE = 1$ and vice versa, only values of $IIE \geq 2$ and $SWS \geq 2$ should be used (regardless of displacement ζ). On the other hand, if the value of these two parameters is too high, calculated velocity fields become scaled up in value, exceeding true values and causing the calculation error to increase again.

Similar to the coefficient β, *SWS* may improve the signal-to-noise noise ratio of the images, but with a different approach. β penalizes excessive local velocity gradients whereas *SWS* smoothes the gray level (and its gradient) by non-weighed arithmetic averaging, both methods effectively acting as low-pass filters [20]. The coefficient β should be set accordingly to the velocity gradient magnitude for proper penalization, while the primary criterion for choosing *SWS* should be the flow feature displacement ζ to ensure downsampling (with factor Δxy) is performed on a smooth enough pollutant concentration field.

To determine a more general rule for the selection of Δxy and *SWS*, an optimal range of these two parameters can be analyzed as a function of ζ. In Table 2, the range of Δxy and *SWS* that gives best calculation results is given for the existing image sequence as well as for the sequence where every second image is taken for calculation (subscript 2ζ), increasing ζ two-fold.

From Table 2 we can see that the optimal Δxy and *SWS* range is proportional to the displacement ζ. An increase in ζ caused by increased flow velocity (e.g. window 3 compared to windows 1 and 2) or reduced frame rate (e.g. image set with one half of the original sampling rate, subscript 2 ζ) results in the need for higher Δxy and *SWS* settings. As seen in Table 1, frame rate reduction causes an increase in calculation errors, especially when $\zeta > 5$. Calculations on image sets with

$\zeta > 10$ should generally be avoided to prevent issues due to the lack of temporal information.

Table 2. *Optimal Δxy and SWS range depending on flow feature displacement ζ. Minimum velocity calculation errors are also listed*

win.	ζ	Δxy	*SWS*	erg [%]	edg [°]
1	1.12	2 to 3	2 to 3	13.8	15.2
$1_{2\zeta}$	2.24	3 to 5	2 to 6	13.1	19.3
2	3.30	2 to 4	2 to 5	12.1	5.8
$2_{2\zeta}$	6.60	5 to 7	3 to 7	14.3	7.7
3	5.01	3 to 5	3 to 4	4.1	0.5
$3_{2\zeta}$	10.0	3 to 8	3 to 5	6.8	0.5

In Fig. 8., the optimal Δxy and *SWS* parameter range is shown bounded with solid lines according to the values from Table 2.

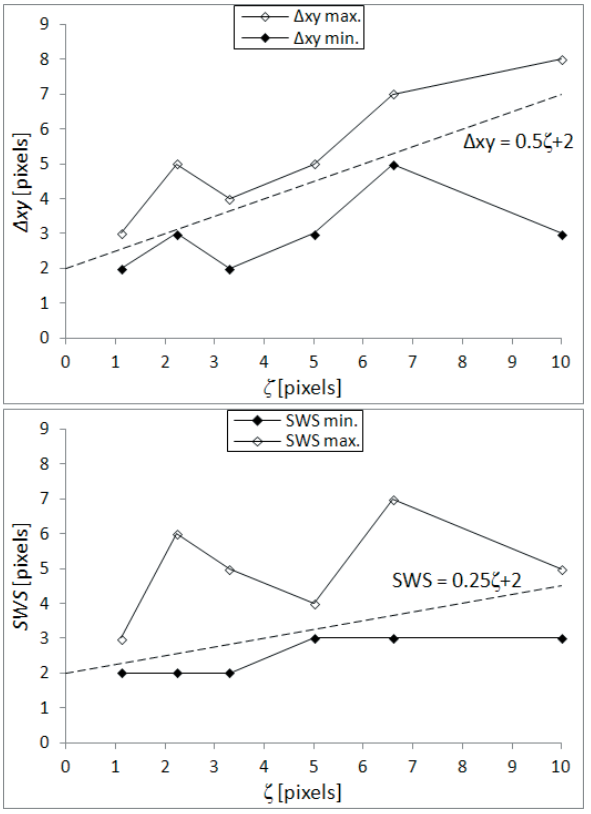

Fig. 8. *Parameters Δxy (upper panel) and SWS (lower panel) as a function of displacement ζ*

We propose a simple linear model for optimal Δxy and *SWS* values (dashed line in Fig. 8) – Eqs. (17) and (18). The model predicts values within the optimal parameter range and fulfills the requirement *IIE, SWS* ≥ 2.

$$\Delta xy = \frac{\zeta}{2} + 2, \tag{17}$$

$$SWS = \frac{\zeta}{4} + 2. \tag{18}$$

In addition to the calculation parameters studied so far, the effect of coefficients γ and θ was investigated as well. Coefficient γ (Fig. 9) regulates flow velocity divergence and reduces calculation error when set between 0.01 and 0.1. Velocity magnitude error is reduced by about 10%, in terms of the error's own magnitude, for window 1 and as much as 40% for window 3 when compared to calculations with $\gamma = 0$. Directional error is only affected by γ in window 1 (2° reduction), while in windows 2 and 3 the error reduction is negligible. However, as γ exceeds 0.1, velocity magnitude and directional error start to grow very rapidly, making calculation less accurate than for $\gamma = 0$.

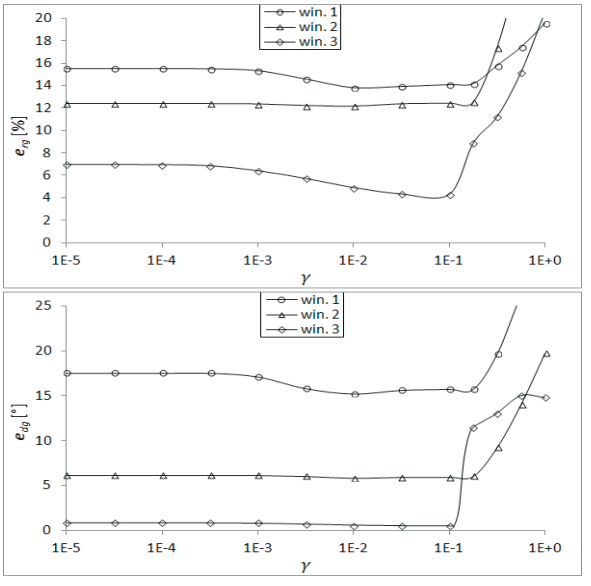

Fig. 9. *Effect of γ on velocity magnitude error (upper panel) and directional error (lower panel)*

Coefficient θ (Fig. 10) regulates gradient of flow velocity curl and reduces calculation error when set between approximately 10^{-7} and 10^{-6}. Velocity magnitude error is reduced by almost 15%, in terms of the error's own magnitude, for window 1 and up to 30% for window 3 when compared to calculations with $\theta = 0$. Directional error exhibits a similar dependence on θ, dropping to almost zero for window three in the optimal θ range. As with coefficient γ, calculation errors increase drastically if θ is raised above its optimal range.

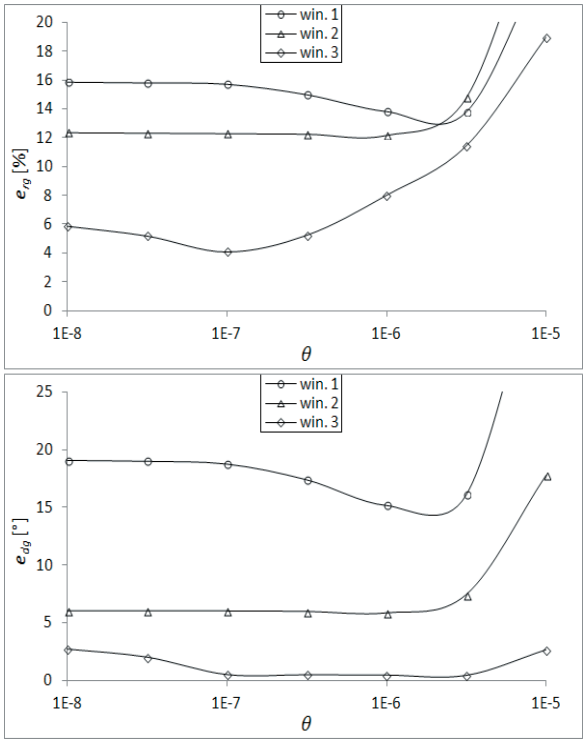

Fig. 10. *Effect of θ on velocity magnitude error (above) and directional error (below)*

In the end, we can summarize our findings regarding the calculation parameters of our velocimetry method into some recommendations:

- $\Delta xy = \zeta/2+2$ (rounded to nearest integer),
- $SWS = \zeta/4+2$ (rounded to nearest integer),
- $\Delta t = 1$,
- $IIE = 2$ to 5 (lower for more unsteady flows),
- $\beta = 10^{-4}$ to 10^{-2} (higher for more unsteady flows),
- $\gamma = 0.01$ to 0.1,
- $\theta = 10^{-7}$ to 10^{-6}.

Of course, these recommendations are quite general and may not be optimal for all real-life flow problems. One is advised to consider the suggested parameter values as initial settings and then critically asses the results (velocity fields) by comparing them to the flow characteristics knowledgebase of the process determined by other methods (theoretical, experimental and/or numerical), if available.

5 CONCLUSIONS

A non-contact, computer-aided velocimetry method was developed for the quantification of flow kinematic properties. The method takes recorded grayscale images of an observed process as input and calculates two-dimensional flow velocity fields. The calculation algorithm is based on the advection diffusion equation, which couples the velocity field with the concentration field, and the image downsampling process, which compensates for the feature displacement between consecutive frames. The method was evaluated on a synthetic image set of airfoil flow visualization in a wind tunnel generated by a combination of CFD software and inverse advection-diffusion equation solver. This way, we simulated a quite complex flow with several characteristic flow types and a known ground truth.

Our velocimetry method was evaluated by comparison of the calculated velocity fields to their ground truth counterparts in three different regions. Multiple calculation parameters were varied, calculating the velocity field error for each set. Calculation errors proved to be reasonably small for the optimal parameter settings – between 4% and 14% of mean inlet velocity for velocity magnitude and 0.5 to 15° for velocity direction. It is important to note that a lower error was attained in the higher flow velocity (thus more critical) regions above the airfoil and in the von Karmann vortex street behind it, compared to a higher error on the slowly moving vortex just behind the airfoil. Furthermore, visual comparison of the calculated and true velocity fields showed a good agreement between both, especially in terms of flow structures that were well preserved by our algorithm. In addition, our method proved to be quite robust in terms of the calculation parameter range that provides good results, although caution should be taken not to use excessively high parameter values, especially with penalization coefficients, as the error may surge. Based on the parameter variation results, some general conclusions regarding optimal parameter settings were provided, including a model for the downsampling coefficient and the smoothing window size.

While the conventional measurement methods may still be more accurate, they face other limitations that are overcome by our method. For example, the PIV method requires an expensive and rigid experimental setup and would perform poorly on smoke-type visualization. On the other hand, point velocimetry methods such as hotwire anemometry (HWA) and laser Doppler anemometry (LDV) are time consuming and only give average flow velocity fields. Therefore, the potential range of our method's applications is much wider, including, but not limited to, the visualization and control of many industrial processes for which other velocimetry methods are not well suited. Further development of our velocimetry method should include numerical

calculation algorithm refinements to improve the accuracy and introduce a certain level of automation in choosing the proper calculation parameters. The use of additional synthetic and real image sets for various flow types could also allow for a statistically verified knowledgebase of the method's optimal employment. At some point, an expansion of our method from 2D to 3D may be considered as an alternative to the other 3D velocimetry methods. Of course, development of the method must also be followed by improvements in flow visualization techniques, namely in a variable pollutant concentration to ensure optimal utilization of the method for areas of more stationary and laminar flows.

6 ACKNOWLEDGMENTS

This work was supported in part by the Slovenian Research Agency (ARRS), grants P2-0167, L2-4270. Operations were also financed in part by the European Union, European Social Fund and the Ministry of Economic Development and Technology, Republic of Slovenia, project KROP 2011 at Abelium d.o.o.

7 REFERENCES

[1] Širok, B., Bajcar, T., Orbanić, A., Eberlinc, M. (2011). Melt mass flow measurement in mineral wool production. *Glass Technology*, vol. 52, no. 5, p. 161-168.

[2] Tic, V., Lovrec D. (2012). Design of modern hydraulic tank using fluid flow simulation. *International Journal of Simulation Modelling*, vol. 11, no. 2, p. 77-88, DOI:10.2507/IJSIMM11(2)2.202.

[3] Novak, G., Kozelj, D., Steinman, F., Bajcar, T. (2013). Study of flow at side weir in narrow flume using visualization techniques. *Flow Measurement and Instrumentation*, vol. 49, p. 45-51, DOI:10.1016/j.flowmeasinst.2012.10.008.

[4] Novak, G., Steinman, F., Müller, M., Bajcar, T. (2012). Study of velocity field at model sideweir using visualization method. *Journal of Hydraulic Research*, vol. 50, no. 1, p. 129-133, DOI:10.1080/00221686.2011.648766.

[5] Bajcar, T., Gosar, L., Širok, B., Steinman, F., Rak, G. (2010). Influence of flow field on sedimentation efficiency in a circular settling tank with peripheral inflow and central effluent. *Chemical Engineering and Processing: Process Intensification*, vol. 49, no. 5, p. 514-522, DOI:10.1016/j.cep.2010.03.019.

[6] Bajcar, T., Steinman, F., Širok, B., Prešeren, T. (2011). Sedimentation efficiency of two continuously operating circular settling tanks with different inlet- and outlet arrangements. *Chemical Engineering Journal*, vol. 178, p. 217-224, DOI:10.1016/j.cej.2011.10.054.

[7] Horn, B. K. R., Schunck, B. G. (1981). Determining optical flow. *Artificial Intelligence*, vol. 17, p. 185-204, DOI:10.1016/0004-3702(81)90024-2.

[8] Black, M.J., Anandan, P. (1996). The robust estimation of multiple motions: parametric and piecewise smooth flow fields. *Computer Vision and Image Understanding,* vol. 63, no. 1, p. 75-104, DOI:10.1006/cviu.1996.0006.

[9] Brox, T., Bruhn, A., Papenberg, N., Weickert, J. (2004). High accuracy optical flow estimation based on a theory for warping. *8th European Conference on Computer Vision*, Prague.

[10] Bruhn, A., Weickert, J. (2005). Towards ultimate motion estimation: combining highest accuracy with real-time performance. *10th International Conference on Computer Vision*, Beijing.

[11] Barron, J.L., Fleet, D.J., Beauchemin, S. (1994). Performance of optical flow techniques. *International Journal of Computer Vision*, vol. 12, no. 1, p. 43-77, DOI:10.1007/BF01420984.

[12] Zimmer, H., Bruhn, A., Joachim, W. (2011). Optic flow in harmony. *International Journal of Computer Vision*, vol. 93, no. 3, p. 368-388, DOI:10.1007/s11263-011-0422-6.

[13] Corpetti, T., Heitz, D., Arroyo, G., Memin, E., Santa-Cruz, A. (2005). Fluid experimental flow estimation based on an optical-flow scheme. *Experiments in Fluids*, vol. 40, no. 1, p. 80-97, DOI:10.1007/s00348-005-0048-y.

[14] Bajcar, T., Širok, B., Eberlinc, M. (2009). Quantification of flow kinematics using computer-aided visualization. *Strojniški vestnik – Journal of Mechanical Engineering,* vol. 55, no. 4, p. 215-223.

[15] Regert, T., Tremblais, D., Laurent, D. (2010). Parallelized 3D optical flow method for fluid mechanics applications. *Paper presented at the 5th International Symposium on 3D Data Processing, Visualization and Transmission*, Paris.

[16] ANSYS 13.0 Documentation (2010). Ansys, Pittsburgh.

[17] Sekavčnik, M., Mori, M., Novak, L., Smrekar, J., Tuma, M. (2008). Heat transfer evaluation method in complex rotating environments employing IR thermography and CFD. *Experimental Heat Transfer*, vol. 21, no. 2, p. 155-168, DOI:10.1080/08916150701815770.

[18] Menter, F.R., Egorov, Y. (2010). The Scale-Adaptive Simulation Method for Unsteady Turbulent Flow Predictions. Part 1: Theory and Model Description. *Flow, Turbulence and Combustion*, vol. 85, no. 1, p. 113-138, DOI:10.1007/s10494-010-9264-5.

[19] Ngo, Q. A. (2010). Explicit one-step schemes for the advection equation: The upwind scheme. Retrieved on 19. 11. 2013 from *http://anhngq.wordpress.com/2010/04/25/explicit-one-step-schemes-for-the-advection-equation-the-upwind-scheme/*.

[20] Brüllmann, D. D., d'Hoedt, B. (2011). The modulation transfer function and signal-to-noise ratio of different digital filters: a technical approach. *Dentomaxillofacial Radiology,* vol. 40, p. 222-229, DOI:10.1259/dmfr/33029984.