SoftwareX 12 (2020) 100585

Contents lists available at ScienceDirect

SoftwareX

journal homepage: www.elsevier.com/locate/softx

Original software publication

OpenPIV-MATLAB — An open-source software for particle image velocimetry; test case: Birds' aerodynamics

Hadar Ben-Gida [a,*], Roi Gurka [b], Alex Liberzon [c]

[a] Aeronautical Engineering Branch, Israeli Air Force, 6473428 Tel-Aviv, Israel
[b] Department of Physics and Engineering Science, Coastal Carolina University, Conway, SC 29528, USA
[c] School of Mechanical Engineering, Tel-Aviv University, Tel-Aviv, 6997801, Israel

ARTICLE INFO

Article history:
Received 25 July 2020
Received in revised form 25 August 2020
Accepted 31 August 2020

Keywords:
Particle Image Velocimetry
MATLAB
Fluid mechanics
Wake analysis

ABSTRACT

We present an open-source MATLAB package, entitled OpenPIV-MATLAB, for analyzing particle image velocimetry (PIV) data. We extend the PIV analysis with additional tools for post-processing the PIV results including the estimation of aero/hydrodynamic forces from the PIV data of a wake behind an immersed (bluff or streamlined) body. The paper presents a detailed description of the packages, covering the three main parts: generating two-dimensional two component velocity fields from pairs of images (OpenPIV-MATLAB), spatial and temporal flow analysis based on the velocity fields (Spatial and Temporal Analysis Toolbox), and wake flow analysis along with the force estimates (getWAKE Toolbox). A complete analysis with a variety of post-processing capabilities is demonstrated using time-resolved PIV wake data of a freely flying European starling (*Sturnus vulgaris*) in a wind tunnel.

Code metadata

Current code version	v1.7
Permanent link to code/repository used for this code version	https://github.com/ElsevierSoftwareX/SOFTX-D-20-00014
Code Ocean compute capsule	
Legal Code License	MIT
Code versioning system used	git
Software code languages, tools, and services used	MATLAB
Compilation requirements, operating environments & dependencies	Linux, macOS, Windows, MATLAB (with Image Processing Toolbox, Statistics and Machine Learning Toolbox, Curve Fitting Toolbox and Signal Processing Toolbox)
If available Link to developer documentation/manual	https://github.com/OpenPIV/openpiv-matlab/blob/master/docs/Tutorial_OpenPIV/Tutorial_OpenPIV.pdf, https://github.com/OpenPIV/openpiv-spatial-analysis-toolbox/blob/master/docs/tutorial.rst, https://github.com/OpenPIV/getWAKE/blob/master/docs/%E2%80%8F%E2%80%8FgetWAKE-UsersManual.pdf
Support email for questions	openpiv2008@gmail.com

1. Motivation and significance

Particle Image Velocimetry (PIV) is a state-of-the-art optical flow measurement technique. Using PIV, two- and three-dimensional (2D/3D) velocity fields are obtained with high spatial and temporal resolution, in a non-intrusive manner. PIV is widely used in research and industrial applications. For a detailed description of PIV principles and methodology, please refer to the book by Raffel et al. [1].

OpenPIV is an international scientific community that develops free and open-source software that performs PIV analysis to obtain the velocity fields from images and a variety of post-processing tools to elucidate the physics of the investigated flow. OpenPIV houses several software packages, schematically shown in Fig. 1. Originally, OpenPIV was developed to analyze PIV images

* Corresponding author.
E-mail address: bengida1989@gmail.com (H. Ben-Gida).

https://doi.org/10.1016/j.softx.2020.100585

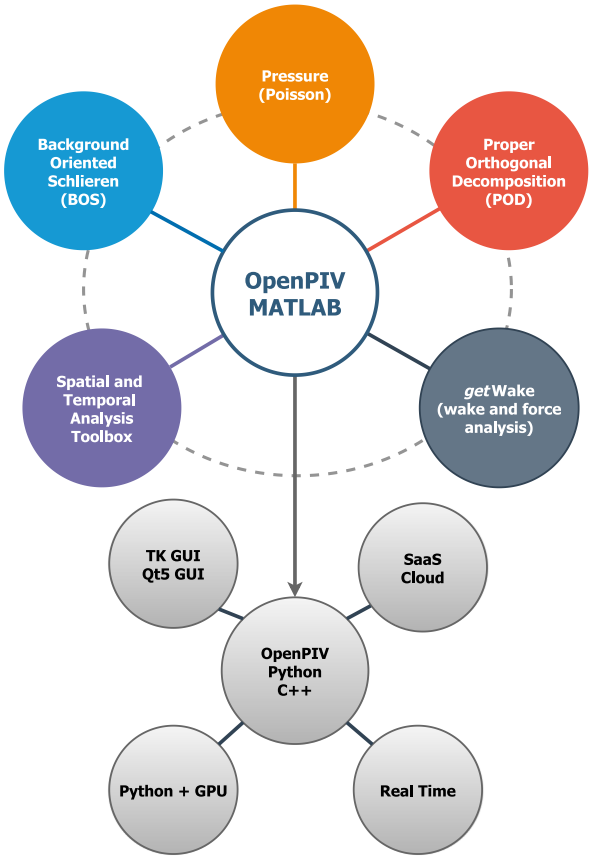

Fig. 1. OpenPIV community software structure overview.

to yield a 2D velocity field using MATLAB platform [2]. Later on, the spatial and temporal analysis toolbox was developed for the characterization of the measured flow field as well as provide turbulence characteristics and some features of spectral analysis based on PIV data. Two additional toolboxes were later added: Pressure and Proper Orthogonal Decomposition (POD) [2,3]. The pressure toolbox uses Poisson formulation of the flow equation to estimate the pressure field, assuming the boundary conditions of the field is known; using an iterative technique [2]. The POD toolbox was developed to extract energetic modes from velocity or vorticity fields to study coherent motions in turbulent flows [3,4]. Another OpenPIV toolbox extended the post-processing capabilities with calculation of the density fields using the background-oriented schlieren technique [5]. Most recently, we added a toolbox to estimate aero/hydrodynamic loads based on PIV data acquired in a body wake, entitled getWAKE [6].

In parallel to MATLAB toolboxes, OpenPIV team developed also C++ and Python branches (see Fig. 1). Detailed description of the C++ version can be found in Taylor et al. [7]. The Python branch has additional features, including adaptive meshing, correlation of 3D-PIV images, real-time processing, among others. A paper of the OpenPIV-Python version is out of scope of this work and will be released separately.

The set of tools described hereinafter, provides an open-source "full-stack" package for analyzing and post-processing PIV data measured in the body wake. The significance is in the variety of tools, together providing a complete solution supporting new PIV users and experts that study aero/hydrodynamics based on wakes of dragged or self-propelled bodies.

We demonstrate OpenPIV-MATLAB capabilities using the PIV near wake flow data measured behind a freely flying European starling (*Sturnus vulgaris*) in a wind tunnel [8] and present how we can shed light on the role of unsteady aerodynamics in natural flyers locomotion. The wake signature behind birds for estimating their aerodynamic performance is a very active field of research due to rapid technological advancement of high-speed lasers, cameras and data transfer solutions [9–12].

The OpenPIV-MATLAB stands-out due to the advanced analysis toolboxes with its capabilities beyond the other open-source PIV software (e.g., PIVlab [13], MATPIV, among others).

2. Software description

2.1. Software architecture

OpenPIV-MATLAB uses the MATLAB (Mathworks Inc.) language to provide fast advanced PIV processing and post-processing tools and straightforward development process.

We present here the three tools: OpenPIV (invoked using `opevpivgui.m`), Spatial and Temporal Analysis Toolbox (`spatialbox.m`) and getWAKE Toolbox (`wake.m`).

- `opevpivgui.m` is a GUI comprised of several subroutines that allow to import PIV images, pre-process them, analyze using fast Fourier transform-based cross-correlation algorithm, filter and interpolate the flow field, and export the velocity vector maps as ASCII files.
- `spatialbox.m` is a GUI that allows the user to load a series of velocity maps created by `openpivgui.m` and calculate various flow characteristics (mean and turbulent, velocity derivatives, energy terms, auto-correlation, etc.), plot them as contours, vector fields and spatial distribution profiles and storing the processed data in MATLAB MAT format.
- `wake.m` is a GUI which loads the MAT file exported by `spatialbox.m`. For the case of time-resolved wake data, it enables reconstruction of a complete wake signature behind the body using a cross-correlation algorithm that overlaps consecutive velocity maps. Based on the reconstructed wake data, `wake.m` also estimates aerodynamic body forces such as profile drag and cumulative circulatory lift.

In the following we describe the functionality of each toolbox.

2.2. Software functionalities

An example of a flowchart of analysis of the body wake PIV data is depicted in Fig. 2.

2.2.1. OpenPIV

The PIV raw images acquired at the wake downstream of a body are imported into MATLAB GUI using `File->Load`. The user defines the main PIV parameters, including: magnification or scale (i.e., pixels/meter), time interval between laser pulses, (Δt), image pre-processing function (e.g., contrast enhancement or inversion of shadowgraphy images), ROI (region-of-interest) for the PIV analysis, interrogation window size and spacing/overlap size (pixels), and filters: signal-to-noise ratio (S/N) type and threshold and an outlier velocity threshold.

The OpenPIV utilizes a cross-correlation algorithm that yields a displacement vector map by correlating two consecutive PIV images. The output data in pixels of displacement is converted to physical units (meters/second) using the Δt and magnification (scale).

The main `openpivgui.m` loop consists of the following steps:

- Cross-correlation algorithm is applied to sub-image square or rectangular interrogation windows. OpenPIV utilizes an

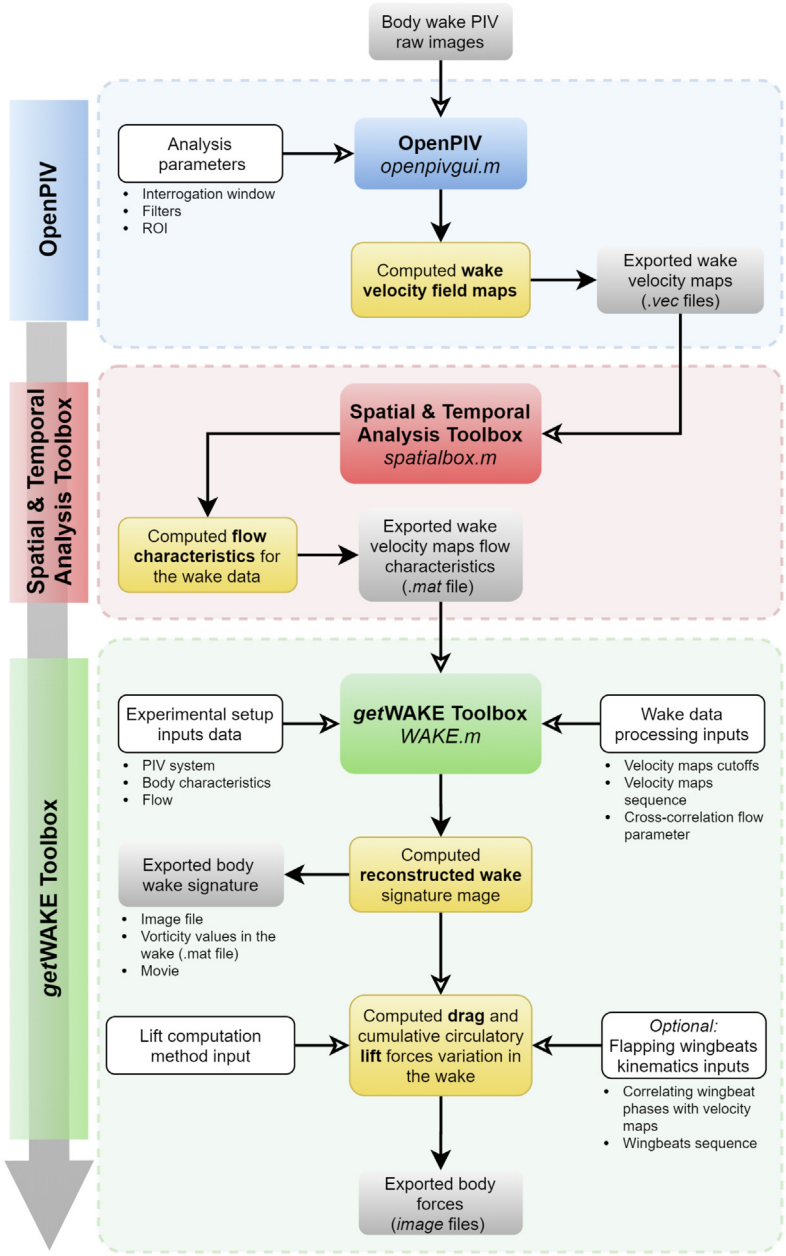

Fig. 2. Schematic overview of the body PIV wake data analysis and post-processing routine in OpenPIV-MATLAB.

FFT-based cross-correlation algorithm to process pairs of PIV images to yield the velocity field maps [1], therefore sizes are typically of $2^n \times 2^n$ size (32×32, 16×64, etc.). The spacing/overlap value controls the spatial resolution of the grid x, y at which we estimate horizontal and vertical velocity components (u, v). Larger interrogation window size reduces resolution but less affected by the background noise.

- The ratio of the maximum cross-correlation peak to the average cross-correlation value, or the ratio of the peak to the second highest peak are used as a measure of the signal-to-noise ratio (S/N) [14–16]. Selection of S/N type is based on the data quality (i.e., strong contrast images). The choice of S/N threshold value that marks erroneous vector is obtained through a trial and error procedure, based on manual assessment of the vector field.
- After the cross-correlation, additional filters are applied for validation and removal of outliers marked by the S/N ratio or removed based on statistics of the flow field. A so-called global filter removes vectors with length that are larger than the mean of the flow field plus N times its standard deviation. The outlier filter parameter in the GUI indicates the value of N.
- Local filter is performed on small neighborhoods of vectors using the 3×3 kernel, removing vectors that are more than 3 times local standard deviation distant from the local mean of the 8 nearest neighbor vectors.
- It is desired to complete the analysis with less than 5%–10% of erroneous vectors. After removing the outliers, the missing values are filled using iterative interpolation, based on the valid neighborhood vectors.

The velocity field result is stored in three ASCII files: the raw vector results (`dataName_noflt.txt`), filtered results (`dataName_flt.txt`) and interpolated data (`dataName.vec`). Output files have headers that define the list of variables, the

units (e.g., pixels/dt or m/s) and the size of the field in terms of rows and columns, followed by the 5 columns of data: x, y, u, v, and S/N.

2.2.2. Spatial and temporal analysis toolbox

The Spatial and Temporal Analysis Toolbox GUI (`spatialbox.m`) loads series of `dataName.vec` files into a 3D flow velocity array where the 3rd dimension is the number of the flow field in the ensemble (or time for the time-resolved case). The data is automatically decomposed (using the Reynolds decomposition) into mean and turbulent fluctuations and provides both qualitative and quantitative visualization tools of a large variety of flow properties. Qualitatively, the toolbox offers several options to show the calculated flow properties in the form of colored contour maps, colored contour lines and vector representation. Quantitatively, it offers the user to plot the properties in a 2D profile format, by selecting regions of interests, or cross-sections. The flow properties include the spatial velocity derivatives and the properties such as vorticity, rate of strain, along with the turbulent parameters such as turbulent intensity, Reynolds stress, turbulent kinetic energy, production, dissipation and enstrophy. Furthermore, the toolbox includes an additional feature allowing to perform some basic spectral analysis using auto-correlation functions applied to the velocity field. The flow characteristics computed in this toolbox for the velocity maps are then exported as a binary MATLAB MAT file.

For more details, the reader is referred to the following illustrative example and documentation listed in Code metadata.

2.2.3. getWAKE toolbox

The getWAKE toolbox is designed for wake data analysis, and specifically for the case where several consecutive flow velocity maps contain time evolution of the flow in the wake or motion of the same vortices as they shed behind the body. The toolbox allows defining multiple experimental setup related parameters, required for the reconstruction (i.e., combination) of the wake signature behind a body, followed by the forces estimate procedure. These parameters include characteristics of the PIV system (time interval between two consecutive laser pulses, scaling ratio in pixels to centimeters and time interval between two consecutive velocity vector maps), body (physical dimensions and weight) and incoming flow (free stream velocity, fluid density and dynamic viscosity) in SI units, and the motion type (stationary or cyclic flapping motion, etc.). For the flapping case, for instance, the parameters relate to three phases of downstroke, transition and upstroke. In some cases, a sub-region from the velocity map is selected mainly to remove noise at the edges. For the wake reconstruction, velocity maps sequence parameters and the cross-correlation flow parameter (different then the one used in the OpenPIV analysis, applied to the instantaneous or fluctuating velocity fields) are defined. Further details regarding the wake reconstruction scheme are available in Appendix A.

The reconstructed wake signature is presented using the velocity fluctuations vector fields (u', v') and colored with the normalized spanwise vorticity field $\omega_z c/U_\infty$, where $\omega_z = \partial v/\partial x - \partial u/\partial y$, c is the characteristic length scale of the body and U_∞ is the freestream velocity. Multiple visualization options are available, including contour threshold, vorticity or swirl strength [17] contours, with or without Gaussian smoothing, etc. The reconstructed wake can be shown and exported as a static image or as an animated movie.

The wake data enables the estimation of the profile drag and cumulative circulatory lift coefficients. The lift is calculated using either (i) Panda and Zaman method [18] or (ii) a direct summation of the circulation values. Further details regarding the forces estimation are given in Appendix B.

Forces can be presented as a function of the normalized streamwise wake distance (x/c, where c is the body size, or chord length of the wing) or time (t). If the wake is generated due to a flapping wing cyclic motion, one can correlate the wingbeat phases (downstroke, upstroke) with the velocity maps in the wake by setting the flapping wingbeats kinematics inputs. Thus, the force coefficients can be presented for a specific wingbeat or a sequence of wingbeats that correspond to the reconstructed wake signature, where the different wingbeat phases are highlighted.

3. Illustrative example

We exemplify the analysis of PIV wake data measurements obtained in the wake of a freely flying European starling (*Sturnus vulgaris*) in a wind tunnel; see Kirchhefer et al. [8] and Ben-Gida et al. [11] for details. The experimental setup utilized for PIV measurements behind the bird is depicted in Fig. 3.

Experiments were conducted in a climatic closed-loop wind tunnel at the Advanced Facility for Avian Research (AFAR), at Western University (Canada). The wind tunnel test section with cross-sectional area of 1.2 m^2 is specifically designed for simulating the flight conditions experienced by birds during long distance migratory travel. The flight conditions reported in this work correspond to atmospheric static pressure, a temperature of 15 °C, and relative humidity of 80%.

A European starling (*Sturnus vulgaris*) had been trained to fly in flapping flight mode in the wind tunnel. At the time the experiments were performed the bird had a mass of 76 g, where its wings had an average chord of $c = 6$ cm, a maximum wingspan of $b = 38$ cm and an aspect ratio (wingspan squared divided by the wings lifting area) of AR = 6.4. A typical cruising flight speed of $U_\infty = 13.5$ m/s was chosen for the experiments [8].

Flow measurements were taken using a long-duration time-resolved PIV system, developed by Taylor et al. [7], consisting of a 80 W double-head diode-pumped Q-switched Nd:YLF laser at a wavelength of 527 nm and two CMOS cameras (Photron FASTCAM-1024PCI) with a spatial resolution of 1 MP at a sampling rate of 1 kHz (see Fig. 3). Olive oil aerosol particles with an average size of 1 μm [19] were introduced into the wind tunnel using a Laskin nozzle from the downstream end of the test section. One camera was used for measuring the PIV wake data (in the streamwise-normal plane) whilst a second one was used for recording the bird kinematics simultaneously with the PIV, where the PIV wake measurements were taken 2 chord lengths behind the starling. The fields of view of the PIV the bird kinematics cameras were $2c \times 2c$ and $9c \times 9c$, respectively. The PIV system sampled pairs of wake images at a rate of 500 Hz (2 ms intervals), which allows sufficient resolution for temporally resolving the wake of the bird that flap its wings at a frequency of 15 Hz.

The distance between the bird wing and the location of the laser sheet was determined by placing a set of IR detectors that trigger the laser once the bird passed the laser location about 3–4 chord length upstream (see Fig. 3).

3.1. OpenPIV analysis

The velocity fields were computed from the PIV wake data behind the bird with 32 $\times$ 32 $pixel^2$ interrogation windows and 50% overlap (16 $\times$ 16 $pixel^2$), thus yielding a spatial resolution of 32 vectors per average chord of the bird (c), equal to 1.8 vectors per millimeter. Here, the second type of the S/N type was used (with a threshold value of 1) and an outlier filter value of 100 was chosen. An example of velocity vector maps computed by the OpenPIV is depicted in Fig. 4. The raw velocity map (non-filtered) is depicted in Fig. 4(a), where the outlier vectors are colored in red. The filtered velocity map (after applying global

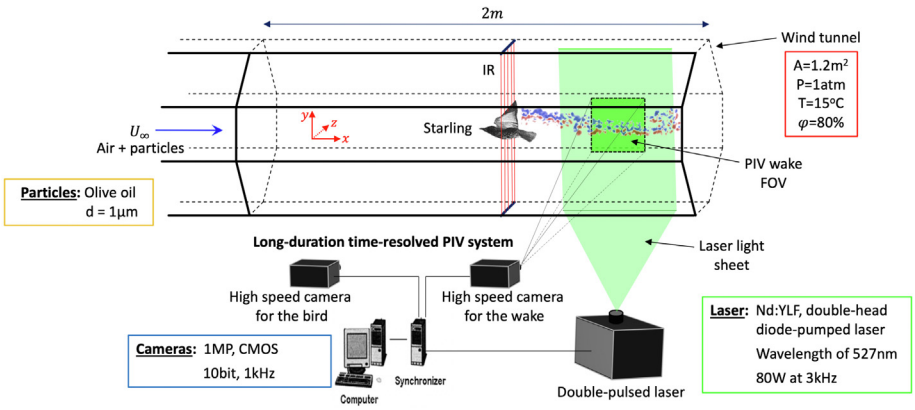

Fig. 3. Illustrative scheme of the avian wind tunnel and the experimental setup system.
Source: Reproduced from [11].

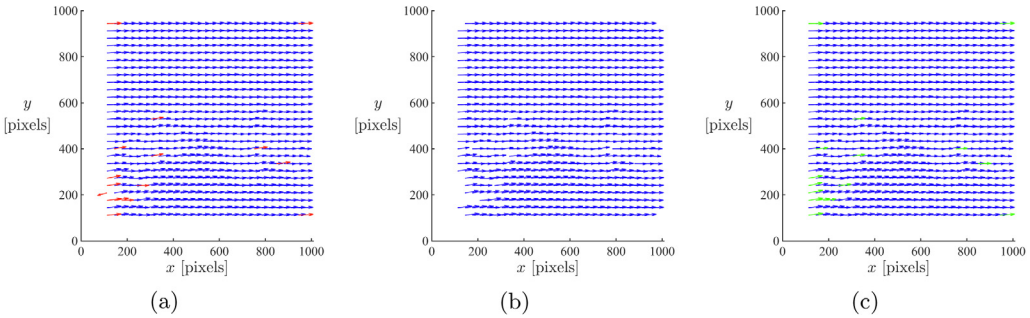

Fig. 4. An example of the output velocity vector maps files from the OpenPIV analysis in MATLAB: (a) raw, (b) filtered, and (c) interpolated velocity vector maps, respectively. Outlier velocity vectors are colored in red, whereas interpolated vectors are colored in green. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

and local filters) is depicted in Fig. 4(b). The final velocity map is shown in Fig. 4(c), where the interpolated vectors are colored in green.

Fig. 5 depicts an example of instantaneous velocity maps computed using the OpenPIV (`opevpivgui.m`) at the near wake behind the starling, while it initiates the downstroke and upstroke phases. The bright light on the right side of the image is the laser light sheet illuminating the flow field behind the bird.

The final velocity vector maps in the wake, which were computed using the OpenPIV (`opevpivgui.m`) and exported as .vec files, are then imported into the Spatial and Temporal Analysis Toolbox (`spatialbox.m`) for the computation of the required flow characteristics. Fig. 6 depicts an example of the instantaneous spanwise vorticity fields computed using the Spatial and Temporal Analysis Toolbox, from the velocity vector maps depicted in Fig. 5, in the near wake behind the starling during flight.

The flow characteristics and the velocity vector maps are then exported from the Spatial and Temporal Analysis Toolbox as a binary MAT file, and imported into the getWAKE Toolbox GUI (`wake.m`). The characteristics of the experimental setup (PIV system, bird and freestream flow) are set in the getWAKE GUI. Using these inputs, the wake signature behind the starling is reconstructed for a sequence of velocity maps, which corresponded to flapping wingbeat phases: downstroke and upstroke, as depicted in Fig. 7. The bird appears to fly from right to left; thus, the downstream wake essentially occurred earlier (corresponding to the downstroke phase), while the upstream wake occurred later (corresponding to the upstroke phase). Here, we utilized the velocity fluctuations (u', v') for cross-correlating the velocity maps.

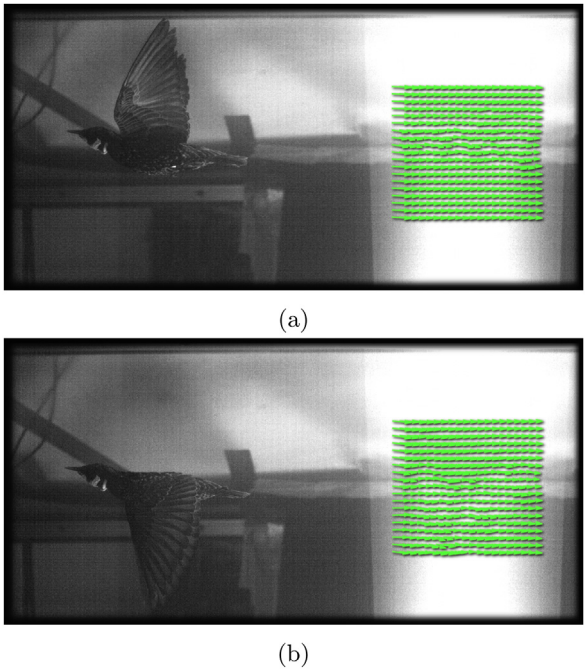

Fig. 5. Example instantaneous velocity vector maps, as computed from the OpenPIV (`opevpivgui.m`) at the near wake behind the flapping starling, while it initiates (a) the downstroke phase and (b) the upstroke phase.

The time variation of the profile drag and cumulative circulatory lift forces exerted on the bird is depicted in Fig. 8. We

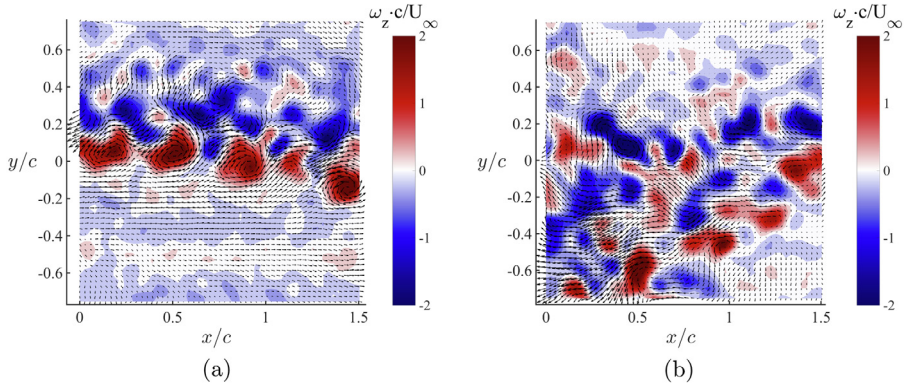

Fig. 6. Example instantaneous spanwise vorticity contour fields, as computed from the Spatial and Temporal Toolbox (`spatialbox.m`) at the near wake behind the flapping starling, while it initiates (a) the downstroke phase and (b) the upstroke phase. The air flows from left to right and the vectors displayed are the velocity fluctuations.

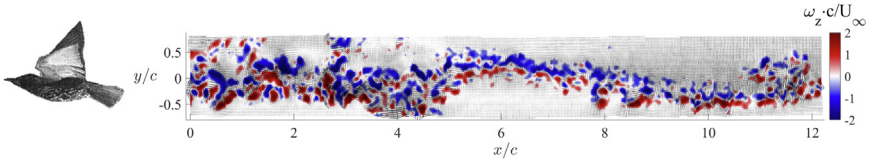

Fig. 7. Example of a wake reconstructed by the getWAKE Toolbox from PIV wake measurements taken behind a freely flying starling during a complete flapping wingbeat. The contour is the normalized spanwise vorticity in the wake, $\omega_z c/U_\infty$. The bird flew from right to left; therefore, the downstream distance is measured as positive chord lengths, x/c. What appears as downstream essentially happened earlier, while what appears as upstream happened later. The vectors displayed are the velocity fluctuations. A threshold of 8% from the global maximum absolute vorticity values in the wake field was applied.

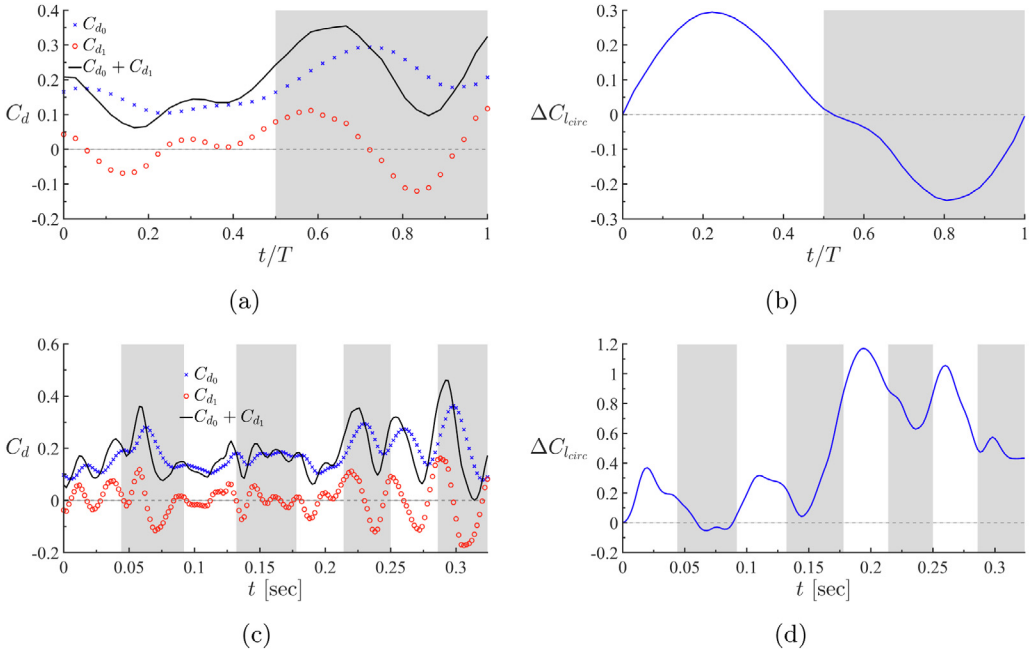

Fig. 8. Time variation of the drag coefficient C_d and the cumulative circulatory lift coefficient $\Delta C_{l_{circ}}$ for (a–b) a single flapping wingbeat, and (c–d) four consecutive flapping wingbeats of the starling. The white shaded regions correspond to the downstroke phases, whereas the gray shaded regions correspond to the upstroke phases.

demonstrate the forces calculations over a single and multiple wingbeat phases. The time variation of the forces for a single wingbeat are given as a function of the non-dimensional time, t/T; where T is the time period of the flapping wingbeat cycle. The white shaded regions in Fig. 8 correspond to the downstroke phases of each flapping wingbeat, whereas the gray shaded region corresponds to the upstroke phases. Herein, C_d, total drag coefficient is the summation of the C_{d_0} and C_{d_1} components, which are the steady and unsteady components of the drag coefficient, respectively. The cumulative circulatory lift coefficient was computed based on Panda and Zaman method [18], see the procedure in Appendix B).

4. Impact

The main contribution of this work is to provide an open source code for PIV data analysis and post-analysis. OpenPIV-MATLAB is a complete 2D PIV data analysis software package that can handle PIV and its extensions (i.e., time-resolved, stereo) images and perform a detailed flow analysis. To the best of our knowledge, OpenPIV-MATLAB is the most complete set of tools for the flow analysis based on data acquired using optical measurement techniques in fluid dynamics, providing thousands of lines of code as an open source. The toolbox is used in a variety of fields associated with fluid dynamics, such as mechanical [20], aerospace [21], civil [22], and environmental engineering [23], medicine [24], biology [25], earth sciences [26] and so forth. The package presented herein offers the only known open-source software for advanced post-processing of time-resolved PIV wake data behind immersed bodies, enabling visual description of the wake evolution over time and space estimation of the aero/hydrodynamic forces.

5. Conclusions

We present a MATLAB (Mathworks Inc.) software package for analysis and post-analysis of PIV data. OpenPIV is comprised of three toolboxes that can analyze PIV images to yield a 2D2C (two-dimensional, two-component) velocity vector fields using cross-correlation technique, post-analysis the velocity fields calculating various flow properties including turbulence and spectral analysis and estimation of the aero/hydrodynamic forces exerted over immersed bodies in fluids. As test-case study of the toolbox applicability to PIV time-resolved data, we present flow analysis of the near wake region behind a freely flying bird in a closed-loop wind tunnel. We demonstrate the functionality and usefulness of the toolboxes as applied to a data collected in a wake region that is unsteady and turbulent. The main contribution is a free, flexible, extendable and validated platform for both basic and advanced PIV related analysis and post-processing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

The authors want to acknowledge all the co-authors of the OpenPIV-MATLAB toolboxes, listed on the respective Github pages, as listed in Code metadata. The open source software could not exist without the vibrant community of users and developers that contribute by their test cases, verification and validation studies, and of course software and documentation development.

Appendix A. Wake reconstruction

The wake reconstruction (see an example wake measured behind a flapping starling in Fig. 7) is based on time-resolved PIV images taken from a stationary camera yielding Eulerian observation of the flow field behind an immersed body (either stationary or undergoing flapping wing motion). The wake reconstruction method described herein was originally developed by the first author and later on certain aspects of the methods appeared in [27]. Thus, there are similarities in terms of its core principles, however, the utilization and output of the reconstruction scheme detailed below are different.

For the wake reconstruction, we assume the body's position did not change much relative to the measurement plane. Based on Taylor's frozen turbulence hypothesis [28], we assume that the turbulent flow remains relatively unchanged as it passes through the measurement plane. This hypothesis implies that there is no significant variation of a spatial velocity distribution over the timescale required for observation, supported by previous studies [29].

As a minimal requirement, sufficient temporal resolution is needed to track the flow patterns as they propagate from one velocity map to another (see Fig. A.1). For example, if the wake is measured behind a bird in flapping flight with a flapping frequency of 10Hz, the PIV velocity vector maps must be sampled at a rate of 100Hz or higher ($\geq$ 200 Hz PIV raw images sampling rate); thus, enabling a given flow structure to be tracked by at least three consecutive velocity maps.

The wake composite image is generated by offsetting each consecutive PIV velocity map with a calculated instantaneous convection velocity and then overlap the images, while keeping the mid region of each instantaneous PIV velocity map (to avoid overlapping 'noisy' data from the velocity map edges). The instantaneous convection velocity, which determines the offset if each PIV velocity map, is calculated based on a cross-correlation algorithm that examine the match of the velocity vector fields (u, v) or the fluctuating velocity vector fields (u', v') of two consecutive velocity maps. The cross-correlation coefficient, which determines the match of two consecutive PIV velocity maps, is calculated as follows (with respect to a given flow property s):

$$C_s(X,Y,T) = \sum_{i=1,j=1}^{I,J} \frac{\left[s(x_i,y_i,t)-\overline{s(t)}\right]\left[s(x_i+X,y_i+Y,t+T)-\overline{s(t+T)}\right]}{IJ\sigma_s(t)\sigma_s(t+T)} \tag{A.1}$$

If $C = 1$, the two velocity maps are identical (perfectly matched), whereas if the two images are different, then $C < 1$. (I, J) is the size of the overlapping area and σ_s is the standard deviation of the flow property s. Here, s is to be replaced with (u, v) for computing the cross-correlation coefficients with respect to the velocity vector components (C_u, C_v), or with (u', v') for computing the cross-correlation coefficients with respect to the velocity fluctuations vector components $(C_{u'}, C_{v'})$. The spatial shift (X, Y) of any instantaneous PIV velocity map is computed based on overlapping that achieves the maximum correlation coefficient; i.e., $\max[C_u, C_v]$ or $\max[C_{u'}, C_{v'}]$.

If the cross-correlation failed during the wake reconstruction process (e.g., when the PIV data is low in quality or if the computed spatial shift is larger than the velocity map boundaries), the spatial shift of each instantaneous PIV map is determined from the freestream advection velocity U_∞ and the time difference Δt; i.e., $X = U_\infty \Delta t$.

Appendix B. Forces estimation

The variation of the drag force coefficient C_d can be estimated from the PIV wake data as shown in Ben-Gida et al. [11]. The analysis is based on the flow momentum equations that with appropriate assumptions leads to the following formulation of the drag coefficient for the two-dimensional wake case:

$$C_d = \underbrace{\frac{2}{cU_\infty}\int_0^h u\left(1-\frac{u}{U_\infty}\right)dy}_{C_{d_0}\text{ - Steady part}} - \underbrace{\frac{2}{cU_\infty}\frac{\partial}{\partial t}\int_0^h\int_0^l \frac{u}{U_\infty}dxdy}_{C_{d_1}\text{ - Unsteady part}} \tag{B.1}$$

where (x, y) is the Cartesian coordinates system used for the streamwise-normal plane in the wake; the x-axis indicates the downstream direction and the y-axis is the normal direction.

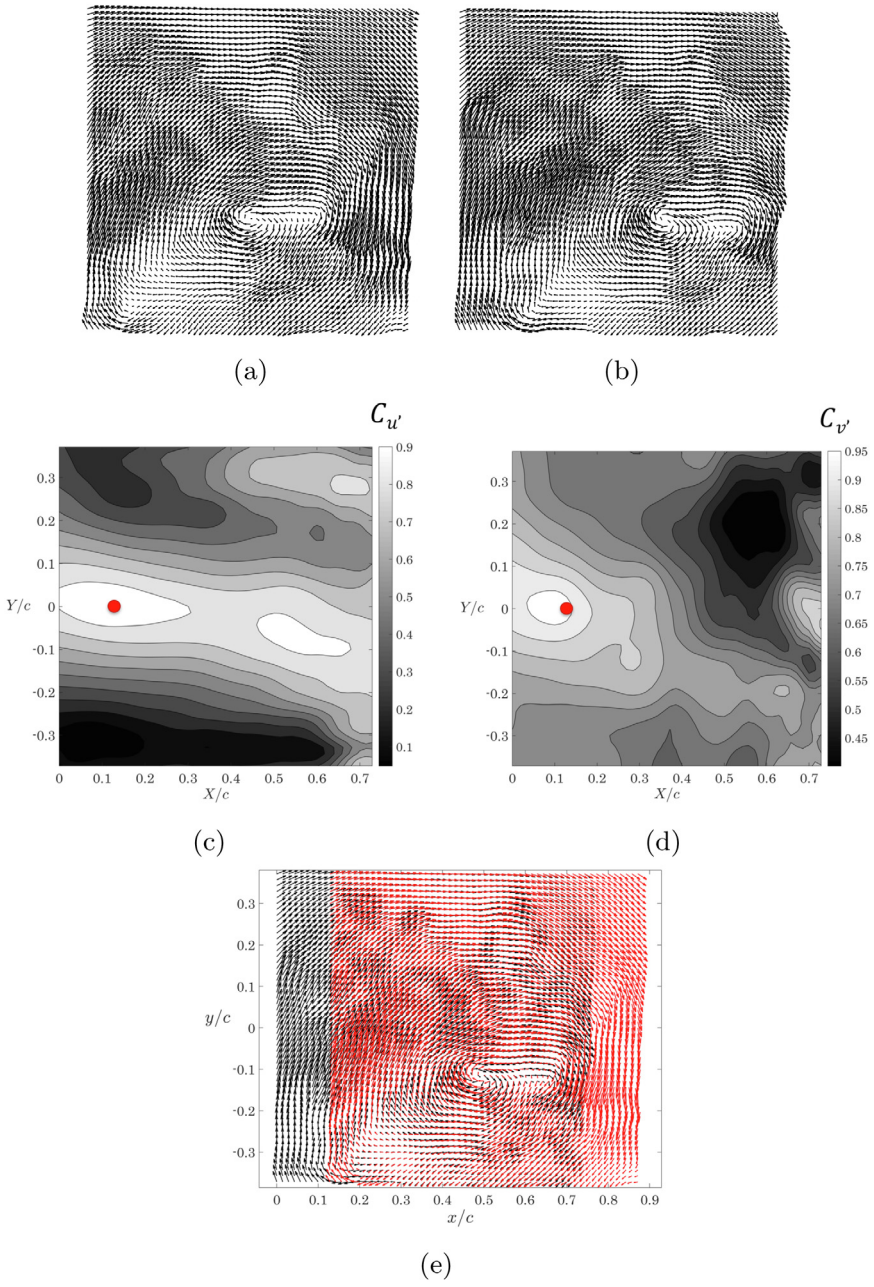

Fig. A.1. An example of the spatial (X, Y) shift, as computed from the cross-correlation procedure, of two consecutive velocity vector maps in the near wake behind a freely flying European starling. The red dots mark the location of the maximum cross-correlation coefficient. (a) velocity vector map at time t_1 (the flow is from left to right); (b) consecutive velocity vector map at time $t_2 = t_1 + 2$ ms; (c) cross-correlation map of the $C_{u'}$ coefficient; (d) cross-correlation map of the $C_{v'}$ coefficient; (e) spatial shift visualization of the two consecutive velocity vector maps.

Conceptually, the steady drag coefficient component C_{d_0} is proportional to the velocity deficit at the wake, whereas the unsteady drag coefficient component C_{d_1} is associated with the unsteady flow motion. While the steady drag term can be obtained from the near wake velocity field, the unsteady drag term requires information regarding the entire control volume (surface in the case of 2D PIV where we assume volume per unit length) surrounding the body over time. We assume most of the unsteady disturbances generated by the unsteady motion are obtained from the velocity field at the near wake where both unsteady contribution and viscous effects have not dissipated yet. Therefore, we approximate the full surface integral of the unsteady term to include only the velocity field obtained from the PIV experiments in the body near wake. Here, U_∞ is the mean undisturbed streamwise velocity, and h and l are the vertical and horizontal extent of the computed velocity field in the wake, respectively. For the steady drag coefficient, we average the various profiles along the streamwise extent of each PIV map, thus yielding a single C_{d_0} value to represent each PIV map. It is noteworthy that drag coefficient is computed for a sequence of velocity maps, and therefore it does not require *a priori* information of the reconstructed wake.

The variation of the cumulative circulatory lift coefficient $\Delta C_{l_{circ}}$ can be estimated from the PIV velocity vector fields based on Stalnov et al. [30], Ben-Gida et al. [31] and Nafi et al. [32]. Any fluid motion around a body is accompanied by the shedding of vortices into the wake. By analyzing these vortical patterns in the near wake, one can estimate the unsteady lift exerted on the body. Herein, the estimation of increment in the time-dependent lift throughout the wake is evaluated from the PIV

velocity fields by utilizing Wu's viscous flow approach [33], which was later expressed by Panda and Zaman [18]. This method can only be applied for near wake flow measurements behind bodies, where it is assumed that the wake has not deformed yet and interactions between the vortices (which results from the tip and root regions) shed into the wake are not significant. Assuming two-dimensional, incompressible flow and neglecting added mass effects, the time-dependent circulatory lift force exerted on a body can be evaluated from the near wake flow field, as follows [18]:

$$L_{circ}(t) = \underbrace{\rho \frac{\mathrm{d}}{\mathrm{d}t} \iint_A x\omega_z(t)\mathrm{d}x\mathrm{d}y}_{x\text{-moment of the vorticity field}} + \underbrace{\rho U_\infty \int_0^t \int_0^h \nu \left(\frac{\partial^2 u(t)}{\partial x^2} + \frac{\partial^2 u(t)}{\partial y^2} \right) \mathrm{d}y\mathrm{d}t}_{\text{Diffusion contribution}} \tag{B.2}$$

In the above equation, the first integral from left is the first x-moment of the vorticity field (A), with $\omega_z(t)$ as the instantaneous spanwise vorticity field. The second integral from left is the contribution from the viscous term (diffusion), where ν is the kinematic viscosity and ρ is the fluid density.

Applying Taylor's hypothesis (as introduced above for the wake reconstruction), $\mathrm{d}x = U_\infty \mathrm{d}t$, one can transform the spatial derivative in the left-side integral of Eq. (B.2) into a temporal one. Moreover, by interchanging the left-side integral in Eq. (B.2) with the time derivative (using Leibniz integral rule), one can re-write Eq. (B.2) as follows:

$$L_{circ}(t) = \rho U_\infty \int_0^t \left[\int_0^h u\omega_z(t)\mathrm{d}y + \int_0^h \nu \left(\frac{\partial^2 u(t)}{\partial x^2} + \frac{\partial^2 u(t)}{\partial y^2} \right) \mathrm{d}y \right] \mathrm{d}t \tag{B.3}$$

Therefore, the change in the lift in time $\delta\tau$ can be expressed accordingly:

$$\delta L_{circ} = \rho U_\infty \delta\Gamma \tag{B.4}$$

where the corresponding change in the circulation $\delta\Gamma$ is given by:

$$\delta\Gamma = \int_0^h u\omega_z(t)\mathrm{d}y + \int_0^h \nu \left(\frac{\partial^2 u(t)}{\partial x^2} + \frac{\partial^2 u(t)}{\partial y^2} \right) \mathrm{d}y \tag{B.5}$$

Since at the beginning of the unsteady motion the lift is unknown, we shall refer to the estimated lift component as an increment in the circulatory lift [30,31] that is generated from the beginning of the motion. Based on Eq. (B.3), the cumulative circulatory lift at time t, $\Delta L_{circ}(t)$, is computed accordingly:

$$\Delta L_{circ}(t) = \rho U_\infty \int_0^t \zeta(t)\mathrm{d}t = \rho U_\infty \Gamma(t) \tag{B.6}$$

with the vorticity flux term $\zeta(t)$ being expressed as follows:

$$\zeta(t) = \int_0^h u_c\omega_z(t)\mathrm{d}y + \int_0^h \nu \left(\frac{\partial^2 u(t)}{\partial x^2} + \frac{\partial^2 u(t)}{\partial y^2} \right) \mathrm{d}y \tag{B.7}$$

Here, u_c is the advection velocity at which the characteristics of the wake collectively travel downstream. The cumulative circulatory lift coefficient at time t is therefore expressed as:

$$\Delta C_{l_{circ}}(t) = \frac{2}{cU_\infty} \int_0^t \zeta(t)\mathrm{d}t = \frac{2\Gamma(t)}{cU_\infty} \tag{B.8}$$

As detailed in Section 2.2.3, the getWAKE Toolbox suggests two options for computing the cumulative circulatory lift coefficient in the wake. The first option is based on Panda and Zaman method [18], where the vorticity flux $\zeta(t)$ defined by Eq. (B.7) is computed for each individual PIV velocity map obtained in the wake behind the body (as function of time), after applying a threshold on the vorticity contours. For each velocity map (or time t), the advection velocity is defined as the mean streamwise velocity component along the x-direction, $u_c \approx \langle u \rangle_x(y)$, and the spanwise vorticity field is estimated as the local mean spanwise vorticity along the x-direction, $\omega_z(x,y,t) \approx \langle \omega_z \rangle_x(y,t)$. Moreover, for each velocity map, the second order derivatives of u are computed using a least squares differentiation scheme and then approximated with their local mean value along the x-direction; $\partial^2 u(x,y,t)/\partial x^2 \approx \langle \partial^2 u/\partial x^2 \rangle_x(y,t)$ and $\partial^2 u(x,y,t)/\partial y^2 \approx \langle \partial^2 u/\partial y^2 \rangle_x(y,t)$. All the above vectors, representing each instantaneous velocity map, are then integrated over the y-direction to yield the vorticity flux $\zeta(t)$, as presented in Eq. (B.7), and the time variation of the cumulative circulatory lift coefficient in the wake (see Eq. (B.8)). Using the option described above, the computation of the cumulative circulatory lift coefficient for a sequence of velocity maps does not require *a priori* information of the reconstructed wake. Nevertheless, if needed, in the getWAKE Toolbox, the user can choose to compute the vorticity flux $\zeta(t)$ directly from the reconstructed wake signature and not from individual velocity maps; with or without a threshold applied on the vorticity contours. In doing so, the advection velocity is approximated as the freestream velocity, $u_c \approx U_\infty$ and the spanwise vorticity field array is taken directly from the reconstructed wake, $\omega_z(x,y)$, where the x-direction and time t are interchangeable by applying $\mathrm{d}t = U_\infty \mathrm{d}x$. The second order derivatives of u are computed using a least squares differentiation scheme directly from the reconstructed wake; $\partial^2 u(x,y)/\partial x^2$ and $\partial^2 u(x,y)/\partial y^2$. All the above arrays, representing the full reconstructed wake map, are then integrated over the y-direction (at each x location) to yield the vorticity flux $\zeta(t)$, as presented in Eq. (B.7).

In the second option, the cumulative circulatory lift coefficient in Eq. (B.8) is computed based on a direct summation of the circulation values throughout the reconstructed wake signature. Thus, $\zeta(t)$ is computed as the integral over the spanwise vorticity field of each individual PIV velocity map, accordingly:

$$\zeta(t) = \int_0^l \int_0^h \omega_z(x,y,t)\mathrm{d}x\mathrm{d}y \tag{B.9}$$

References

[1] Raffel M, Willert CE, Scarano F, Kähler C, Wereley ST, Kompenhans J. Particle image velocimetry: A practical guide. 3rd ed Springer; 2018.

[2] Gurka R, Liberzon A, Hefetz D, Rubinstein D, Shavit U. Computation of pressure distribution using PIV velocity data. In: Third international workshop on particle image velocimetry. Santa Barbara, California: 1999. p. 1–5.

[3] Gurka R, Liberzon A, Hetsroni G. POD of vorticity fields: A method for spatial characterization of coherent structures. Int J Heat Fluid Flow 2006.

[4] Liberzon A, Gurka R, Hetsroni G. Comparison between two and three-dimensional POD in a turbulent boundary layer using multi-plane stereoscopic PIV. J Phys Conf Ser 2011;318(2):22010.

[5] Verso L, Liberzon A. Background oriented schlieren in a density stratified fluid. Rev Sci Instrum 2015;86(10):103705.

[6] Ben-Gida H, Gurka R, Liberzon A. OpenPIV - getWAKE Matlab toolbox. Figshare - Softw 2020;6. http://dx.doi.org/10.6084/m9.figshare.12331007.v5.

[7] Taylor ZJ, Gurka R, Kopp GA, Liberzon A. Long-duration time-resolved PIV to study unsteady aerodynamics. IEEE Trans Instrum Meas 2010;59(12):3262–9.

[8] Kirchhefer AJ, Kopp GA, Gurka R. The near wake of a freely flying European starling. Phys Fluids 2013;25(5):051902.

[9] Spedding GR, Rosén M, Hedenström A. A family of vortex wakes generated by a thrush nightingale in free flight in a wind tunnel over its entire natural range of flight speeds. J Exp Biol 2003;206(14):2313–44.

[10] Johansson LC, Hedenström A. The vortex wake of blackcaps (Sylvia atricapilla L.) measured using high-speed digital particle image velocimetry (DPIV). J Exp Biol 2009;212(20):3365–76.

[11] Ben-Gida H, Kirchhefer AJ, Taylor ZJ, Bezner-Kerr W, Guglielmo CG, Kopp GA, et al. Estimation of unsteady aerodynamics in the wake of a freely flying European starling (Sturnus vulgaris). PLoS One 2013;8(11):e80086.
[12] Gurka R, Krishnan K, Ben-Gida H, Kirchhefer AJ, Kopp GA, Guglielmo CG. Flow pattern similarities in the near wake of three bird species suggest a common role for unsteady aerodynamic effects in lift generation. Interface Focus 2017;7.
[13] Thielicke W, Stamhuis EJ. PIVlab – Towards user-friendly, affordable and accurate digital particle image velocimetry in MATLAB. J Open Res Softw 2014;2.
[14] Huang H, Dabiri D, Gharib M. On errors of digital particle image velocimetry. Meas Sci Technol 1997;8:1427–40.
[15] Huang HT, Fiedler HE, Wang JJ. Limitation and improvement of PIV; Part I: Limitation of conventional techniques due to deformation of particle image patterns. Exp Fluids 1993;15:168–74.
[16] Huang HT, Fiedler HE, Wang JJ. Limitation and improvement of PIV; Part II: Particle image distortion, a novel technique. Exp Fluids 1993;15:263–73.
[17] Zhou J, Adrian RJ, Balachandar S, Kendall TM. Mechanisms for generating coherent packets of hairpin vortices. J Fluid Mech 1999;387:353–96.
[18] Panda J, Zaman KBMQ. Experimental investigation of the flow field of an oscillating airfoil and estimation of lift from wake surveys. J Fluid Mech 1994;265:65–95.
[19] Echols WH, Young JA. Studies of portable air-operated aerosol generators. US Naval Research Laboratory; 1963, Report No. 5929.
[20] Gravish N, Peters JM, Combes SA, Wood RJ. Collective flow enhancement by tandem flapping wings. Phys Rev Lett 2015;115(18):188101.
[21] Mangini D, Mameli M, Fioriti D, Filippeschi S, Araneo L, Marengo M. Hybrid pulsating heat pipe for space applications with non-uniform heating patterns: ground and microgravity experiments. Appl Therm Eng 2017;126:1029–43.
[22] Stanier S, Dijkstra J, Leśniewska D, Hambleton J, White D, Wood DM. Vermiculate artefacts in image analysis of granular materials. Comput Geotech 2016;72:100–13.
[23] Yoneda J, Jin Y, Katagiri J, Tenma N. Strengthening mechanism of cemented hydrate-bearing sand at microscales. Geophys Res Lett 2016;43(14):7442–50.
[24] Wong AD, Searson PC. Live-cell imaging of invasion and intravasation in an artificial microvessel platform. Cancer Res 2014;74(17):4937–45.
[25] Masuzzo P, Van Troys M, Ampe C, Martens L. Taking aim at moving targets in computational cell migration. Trends Cell Biol 2016;26(2):88–110.
[26] Sutherland DA, Jackson RH, Kienholz C, Amundson JM, Dryer WP, Duncan D, et al. Direct observations of submarine melt and subsurface geometry at a tidewater glacier. Science 2019;365(6451):369–74.
[27] Kirchhefer AJ. The near wake of a European starling [Ph.D. thesis], London, Canada: Western Ontario University; 2012.
[28] Taylor GI. The spectrum of turbulence. Proc R Soc Lond Ser A Math Phys Eng Sci 1938;164:476–90.
[29] Zaman KBMQ, Hussain AKMF. Taylor hypothesis and large-scale coherent structures. J Fluid Mech 1981;112:379–96.
[30] Stalnov O, Ben-Gida H, Kirchhefer AJ, Guglielmo CG, Kopp GA, Liberzon A, et al. On the estimation of time dependent lift of a European starling (Sturnus vulgaris) during flapping flight. PLoS One 2015;10(9):e0134582.
[31] Ben-Gida H, Stalnov O, Guglielmo CG, Kopp GA, Gurka R. Unsteady aerodynamics loads during flapping flight of birds; case study: Starling and Sandpiper. In: 12th international conference on heat transfer, fluid mechanics and thermodynamics. 2016. p. 121–7.
[32] Nafi A, Ben-Gida H, Guglielmo CG, Gurka R. Aerodynamic forces acting on birds during flight: A comparative study of a shorebird, songbird and a strigiform. Exp Therm Fluid Sci 2020;113:110018.
[33] Wu JC. Theory for aerodynamic force and moment in viscous flows. AIAA J 1981;19(4):432–41.