SoftwareX 19 (2022) 101173

Contents lists available at ScienceDirect

SoftwareX

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

Original software publication

VISION: VIdeo StabilisatION using automatic features selection for image velocimetry analysis in rivers

Alonso Pizarro [a,*], Silvano F. Dal Sasso [b], Salvatore Manfreda [c]

[a] Escuela de Ingeniería en Obras Civiles, Universidad Diego Portales, 8370109 Santiago, Chile
[b] Department of European and Mediterranean Cultures: Architecture, Environment and Cultural Heritage (DICEM), University of Basilicata, 75100 Matera, Italy
[c] Department of Civil, Architectural and Environmental Engineering, University of Naples Federico II, 80125 Naples, Italy

ARTICLE INFO

Article history:
Received 12 April 2021
Received in revised form 1 March 2022
Accepted 22 July 2022

Dataset link: https://doi.org/10.17605/OSF.IO/HBRF2

Keywords:
Video stabilisation
River monitoring
Image velocimetry
UAS
LSPIV

ABSTRACT

VISION is open-source software written in MATLAB for video stabilisation using automatic features detection. It can be applied for any use, but it has been developed mainly for image velocimetry applications in rivers. It includes a number of options that can be set depending on the user's needs and intended application: 1) selection of different feature detection algorithms (seven to be selected with the flexibility to choose two simultaneously), 2) definition of the percentual value of the strongest features detected to be considered for stabilisation, 3) geometric transformation type, 4) definition of a region of interest on which the analysis can be performed, and 5) visualisation in real-time of stabilised frames. One case study was deemed to illustrate VISION stabilisation capabilities on an image velocimetry experiment. In particular, the stabilisation impact was quantified in terms of velocity errors with respect to field measurements obtaining a significant error reduction of velocities. VISION is an easy-to-use software that may support research operating in image processing, but it can also be adopted for educational purposes.

Code metadata

Current code version	*V0.0.3*
Permanent link to code/repository used of this code version	https://github.com/ElsevierSoftwareX/SOFTX-D-21-00066
Code Ocean compute capsule	*none*
Legal Code License	BSD 3-clause
Code versioning system used	*none*
Software code languages, tools, and services used	*MATLAB*
Compilation requirements, operating environments & dependencies	*Computer Vision Toolbox is required*
If available Link to developer documentation/manual	*none*
Support email for questions	alonso.pizarro@mail.udp.cl

1. Introduction

Image velocimetry analysis is becoming an affordable and accurate alternative for flow monitoring in rivers. Typically, fixed cameras, unmanned aerial systems (UASs; often referred to as unoccupied or uncrewed aerial vehicles, UAVs, drones, and remotely piloted aircraft systems, RPASs), or even smartphones can be used to get short videos to compute instantaneous and time-averaged flow characteristics. Although this modern approach is increasing at an unprecedented rate, errors and uncertainty persist, which in turn are not always embraced explicitly [1–6]. The image velocimetry analysis workflow starts with data acquisition, following a pre-processing phase to account for image distortions, undesired movements removal and image enhancement. Afterwards, the application of image velocimetry algorithms can be made. Finally, a post-processing phase might be considered to remove outliers. A number of issues arise, though, at the first stage termed data acquisition. Worth mentioning is that these issues propagate to the entire velocimetry workflow, influencing

* Corresponding author.
E-mail addresses: alonso.pizarro@mail.udp.cl (Alonso Pizarro), silvano.dalsasso@unibas.it (Silvano F. Dal Sasso), salvatore.manfreda@unina.it (Salvatore Mafreda).

https://doi.org/10.1016/j.softx.2022.101173

results significantly [7]. Among them, stabilisations problems are regularly met when the device used to acquire the footage is unstable due to windy or traffic conditions, UAS operator experience or GPS inaccuracies. Non stabilised sequences of images can result in unreal movements in parts where there are none, affecting velocity estimations directly [8].

Many image velocimetry software tools are available to perform velocimetry analyses as well as video acquisition, image pre-processing, and post-processing of results. Among them, KLT-IV [8,9] is an easy-to-use software package for sensing flow velocities and discharge in rivers. Flexibility within the analysis is provided by the fact that surface flow velocities can be determined using either mobile camera platforms or fixed monitoring stations. Different modules are implemented, giving endless possibilities to carry out image velocimetry and pre-processing tasks. The stabilisation process is performed taking as reference the first frame, which defines the coordinate system, and subsequent frames are referred to it. The stabilisation is not fully automated, requiring user involvement by drawing a zone to be excluded due to evident motion (mask). This zone can be, for instance, the area where the water flows or other regions with clear movements. Afterwards, the field of view is divided into four quadrants, and the strongest 10% of detected features – using the minimum eigenvalue algorithm – are kept. Features are matched using the Kanade–Lucas–Tomasi (KLT) tracking algorithm with five pyramid levels, and frames are stabilised using a similarity transform. FlowVeloTool [10] is an open-source velocimetry suite that can automatically extract the water surface area as well as features detection, filtering, and tracking. Furthermore, compensation in case of undesired camera movements is obtained using an automatic feature detection approach, using the Accelerated KAZE (AKAZE) algorithm exploiting brute force matching and a perspective transform. FUDAA [11] is a stand-alone software package, providing an easy way to perform image velocimetry analyses in rivers. The software package has a friendly interface and has been under development for more than a decade. It is free software and well documented both at practitioner and research levels. The latter has led to applications to different natural contexts with successful results (see e.g. [11,12]). FUDAA has a stabilisation module that can be executed separately from orthorectification or other software units. Similarly to KLT-IV, FUDAA needs user involvement by providing a mask over moving zones on the first frame. This zone is used over all the frames to exclude this area in the stabilisation process. Features are detected and matched by using the SURF operator [13], and stabilisation is carried out using a perspective transform. The Rectification of Image Velocity Results (RIVeR) [14] is also a stand-alone software that encompasses PIV, PTV, and STIV velocimetry algorithms (RIvER uses PIVLab and PTVLab software packages). Stabilisation is optional, and the previous frame is taken as a reference. The FAST operator is used to detect and match features, whereas stabilisation is performed with an affine transform. Finally, PIVLab [15] and PTVLab [16] are general-purpose application software tools that are not necessarily restricted to run under field conditions. Indeed, they are often used for PIV and PTV analyses in laboratory-based experiments on which ideal conditions are normally assured. Therefore, no stabilisation modules are provided.

After this short description of the most used available software packages, it is clear that no multipurpose stabilisation module exists at present. It is common to find studies where the authors develop their own codes to deal with this issue (which are not always available), limiting intercomparison and the development of standard methods and tools (Ljubičić et al. [17]). The latter leads to increasing the uncertainty and errors associated with flow characteristic estimates using image velocimetry frameworks.

VISION aims to overcome the aforementioned issues by providing an easy-to-use and open-code MATLAB command-line function following an automatic feature selection approach. The latter detects features automatically on a frame-by-frame basis and match them to stabilise the video. Hence, stabilisation can be carried out with and without ground control points (GCPs), giving more flexibility to the user and case studies under natural conditions. The user's involvement is, therefore, little required being a good option for both experienced and non-experienced ones. Seven feature detection algorithms are implemented, and in order to exploit as maximum as possible the different and own algorithm capabilities, the stabilisation analysis can be carried out with two of them chosen simultaneously. This choice can be beneficial by keeping the best algorithms capabilities but at the expenses of higher computational loads. Afterwards, the strongest and uniformly distributed features are kept and used to stabilise. VISION also gives the possibility to adopt a region of interest (ROI) – on which the stabilisation analysis is carried out – and visualise stabilisation performances in real-time.

This manuscript is organised as follows: Section 2 presents a general overview of VISION and its different running options, while Section 3 presents the Belgrade case study taken for illustration purposes. Section 4 shows the performance of VISION and related stabilisation effects on image velocimetry analysis. Section 5 presents the impacts, challenges, and future developments of VISION. Conclusions are provided at the end.

2. VISION background

VISION is a MATLAB function aiming at stabilising videos for image velocimetry analyses in rivers [18]. It is a simple and easy-to-use command-line function written in MATLAB R2020a. It is necessary to clarify that it exploits the Computer Vision Toolbox, and as a consequence, it is required for proper implementation. Fig. 1 shows the flowchart of VISION, encompassing inputs and data preparation, automatic features detection and matching, and video frame transformation. The tool requires only one input (video to be stabilised) and has six optional totally configurable settings: (i) Stabilised video name; (ii) Selection of the feature detection algorithm(s); (iii) Definition of the percentual strongest detected features for stabilisation purposes; (iv) Definition of the geometric transform type; (v) Definition of an ROI, with the possibility to draw it or use a vector which defines the corners of a rectangle; and, (vi) Stabilisation viewer decision to assess stabilisation analysis visually and on a frame-by-frame basis. These six configurable settings can be classified into three different categories (Footage name; Features detection and Geometric transformation; and, Extras) whose discussions are presented more in detail below. In the first step – VISION inputs and Data preparations – all the variables and algorithms are declared. In the second one – Automatic feature detection and matching - features have been automatically detected and feature matching is performed. For each feature detected in the reference image, the corresponding neighbourhood in a subsequent image is examined in order to find a matching feature, and vice versa. Since automatic detection algorithms usually detect a relatively high number of feature pairs when compared to the methods with manual selection of features, a stronger features selection filter can be applied that define the number of points with strongest metrics. In the following, a uniform filter can be applied that takes the originally detected features and applies a filter to have them as uniform as possible in space. Afterwards, an outlier detection is applied in order to improve the transformation accuracy. In this research, the M-estimator Sample Consensus (MSAC) outlier detection method to exclude them was used to find unacceptable feature correspondences. In the final step – Frame transformation

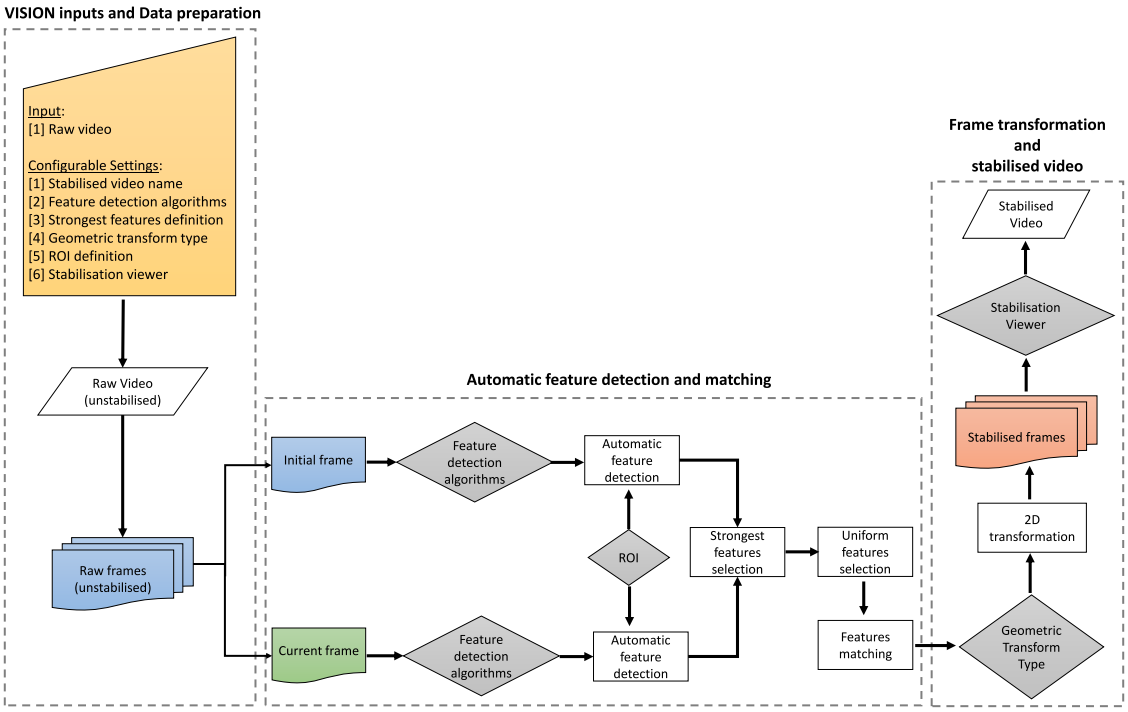

Fig. 1. Flowchart of VISION for video stabilisation. It encompasses VISION inputs and data preparation, automatic features detection and matching, frame transformation and video stabilisation.

and stabilised video - affine or projective transformation matrix between the two images can be determined with any least-square-based procedure and stabilised video is assembled. All the algorithms used for extracting and matching features, and estimating geometric transform are based on MATLAB built-in functions.

a. *Footage name:* The name of the stabilised video can be decided by the user. At the end of the video name, VISION adds the automatic feature detection algorithm used to stabilise. The latter to avoid overwriting the video file if the user decides to run VISION several times. The stabilised video is saved by default in the AVI format keeping the original frame rate. VISION uses as a default setting the following name for the stabilised video: 'Stabilised-Video_{Feature detection algorithm}'.
b. *Features detection and Geometric transformation:* Seven automatic feature detection algorithms are implemented within VISION with the flexibility to choose a maximum of two simultaneously. This with the goal to exploit as maximum as possible the different and own capabilities of each algorithm, giving more robustness but at expenses of a higher computation load. These algorithms have the capacity to automatically distinguish corners, blobs, and regions with uniform intensity. Depending on the selected algorithm, single and multi-scale changes as well as rotation can also be detected. The seven automatic feature detection algorithms are:

    1. Features from accelerated segment test (FAST) [19]
    2. Minimum eigenvalue algorithm (MINEIGEN) [20]
    3. Harris–Stephens algorithm (HARRIS) [21]
    4. Binary Robust Invariant Scalable Keypoints (BRISK) [22]
    5. Speeded-up robust features (SURF) [13]
    6. Oriented FAST and Rotated BRIEF (ORB) [23]
    7. KAZE [24]

Worthy of mentioning, SURF and KAZE are recommended for natural environments since they can better detect features in scenes of no human origin, such as rivers. On the contrary, FAST, MINEIGEN, HARRIS, BRISK, and ORB better detect features in scenes with a clear anthropised intervention. Therefore, a combination of two detection algorithms (one from each category mentioned above) can potentially give robustness to the stabilisation process. At the moment of calling the function, the algorithms must be written within parentheses, e.g.: {'FAST','SURF'} – if two algorithms are wanted to be applied – or {'FAST'}, otherwise. Based on a sensitivity analysis performed by the authors (see Fig. 2), VISION uses SURF as the default algorithm. This decision relies on the maximisation of undesired movement reduction and minimisation of computing time (see Section 3, and the Data and Code Availability Statement for stabilised videos and sensitivity analysis). The analyses were performed on an Intel(R) Core(TM) i7-8700 CPU 3.20 GHz processor with 32 GB RAM.

Detected features are also quantified in terms of how strong they are. Hence, extra customisation can be set to use the strongest features detected only. This is a common practice to keep the best information available in each video frame (see e.g. Perks [8]). To this aim, a percentual value ranging between 0 and 100 of the strongest features can be defined (where 100 means all the features to be considered). Afterwards, a uniform filter is applied to remain with 50% of the percentual value previously entered by the user. This filter gives as uniform as possible features distributed spatially. VISION uses 100% as the default value. Geometric transformation type can be easily defined among similarity, affine, and projective [25]. Note that better accuracy of the estimated transformation is achieved when greater is the number of matched points. Similarity is the transform type that requires the minimum number of matched points, whereas projective the maximum one. VISION uses similarity as the default transform type.

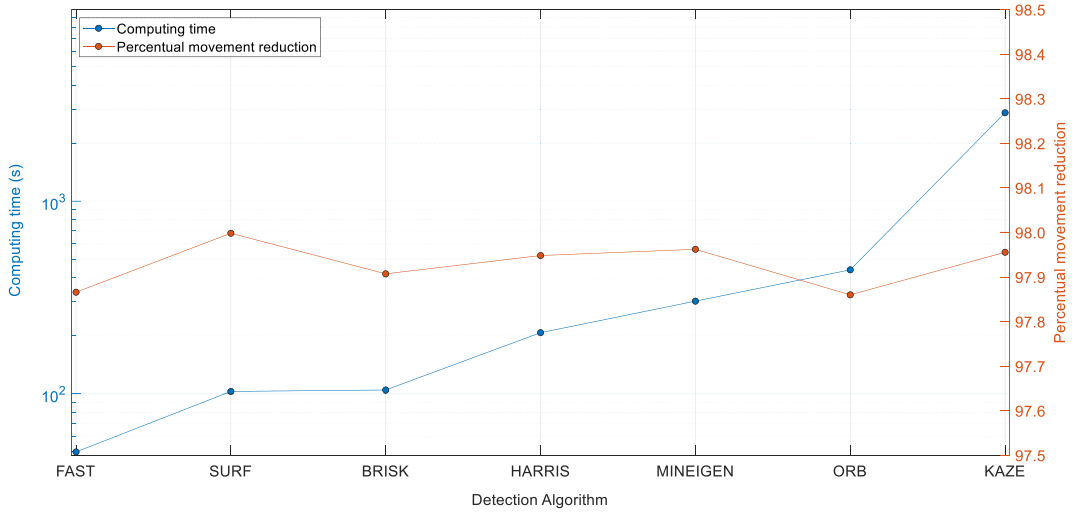

Fig. 2. Sensitivity analysis of the seven different automatic feature detection algorithms to decide the default algorithm used by VISION (SURF). This decision relies on the computing time and undesired movement reductions. Case study: Belgrade (see Section 3, and the Data and Code Availability Statement). Settings: (i) Percentual strongest features: 25%, (ii) Stabilisation Viewer: false, (iii) others: default.

c. *Extras:* The last two configurable settings give the possibility to define an ROI and whether or not the user wants to assess the stabilisation visually. The user has the option to draw an ROI or introduce it as a vector. The latter must keep the following format: [x y Δx Δy], where x and y are the initial coordinates of a desired point (the upper left corner of a rectangle), and Δx and Δy are the extents to the rectangle in two dimensions. When the user chooses to assess the stabilisation in real-time, a viewer (new window) is open, showing the stabilisation process on a frame-by-frame basis. VISION uses all field of view (i.e., no ROI) and the viewer as default settings.
Fig. 3A shows an example to illustrate the ROI drawing process, which is the red rectangle drawn by the user. Fig. 3B presents the stabilisation viewer opened during the real-time stabilisation assessment (default case). Different information is displayed in the video player, such as whether or not MATLAB is processing the footage ('processing' lower left), resolution ('RGB: 2160 x 4096 px'), and the number of the frame under analysis ('10' lower right). In addition, moving objects between consecutive frames are identified with green and magenta colours to identify them visually. For instance, the movement of two operators can be clearly distinguished in Fig. 3C.
After the features are detected and matched, a transformation matrix is estimated. The latter does not include outliers and uses the M-estimator Sample Consensus (MSAC) algorithm to exclude them. Worthy mentioning is that MSAC is a variant of the Random Sample Consensus (RANSAC) algorithm [26,27].

3. Belgrade case study

The Belgrade case study is taken as an example to illustrate how to run VISION. It has six ground control points (GCPs) equally distributed along both riverbanks, a relatively constant river width (~22.30 m), and ADCP measurements (S1 and S2 are the ADCP-measured Sections 1 and 2, respectively). At the acquisition moment, low flow conditions were presented with a river discharge of about 3.4 m^3/s. ADCP data were acquired using a SonTek RiverSurveyor M9 [28], and velocity extrapolation to the water surface was derived by Pearce et al. [29]. The footage was acquired with a standard fitted DJI Phantom 4 Pro camera, hovering at 36 metres and for approximately 30 s (frames rate: 23.98 frames per second (FPS). 740 frames in total). The UAS camera was positioned at nadir position, and the ground sampling distance (GSD) was 0.011 m/px. Fig. 4 shows the Belgrade case study with its six GCPs and ADCP-measured sections. More information about this case study can be found elsewhere [29].

For stabilisation purposes, the original frame rate was reduced and set to 4 FPS following Pearce et al. [29]'s sensitivity analysis and recommendations for LSPIV applications (120 frames in total, see below for stabilisation effects on LSPIV velocity estimates). GCPs are essential for stabilisation as they can be used to compare before and after removing undesired movements. Stabilisation with VISION was performed using two feature detection algorithms (FAST and SURF) and keeping 25% of the strongest detected ones for further analysis. Afterwards, a uniform filter was applied to remain with half of this percentage to have as uniform as possible spatially distributed features. The decision to use two different detection algorithms relies on illustration purposes but also on the idea of exploiting as much as possible different algorithm capabilities (FAST works better in scenes with a clear anthropised intervention, whereas SURF is recommended for natural environments). All the image field of view was used and therefore, no ROI was deemed.

To the aim of comparing raw and stabilised videos objectively, the open-source software Blender (https://www.blender.org) was chosen to compute trajectories. Blender uses the parallax method to reconstruct the camera movement in a 3D scene. For this purpose, six markers in correspondence with the GCPs were identified. For each one, the pattern size (large enough to cover the entire marker) and the search area (to capture the target movements on a frame-to-frame basis) were defined. Results were easily exported to a spreadsheet file which is also provided within this work (see Code and Data availability statement).

Fig. 5 shows the trajectories of the six GCPs considering raw and stabilised-with-VISION footage. The graphs show GCPs movement respect to the first frame and allow to appreciate the strong impact that the tool produces on the relative position of GCPs in the image. The impact of VISION is summarised in Table 1, which presents the minimum, maximum, and average values for GCPs movement estimated respect to the first frame as reference (configuration 1 - default configuration of the tool) as well as frame-by-frame (configuration 2). Overall, stabilisation results

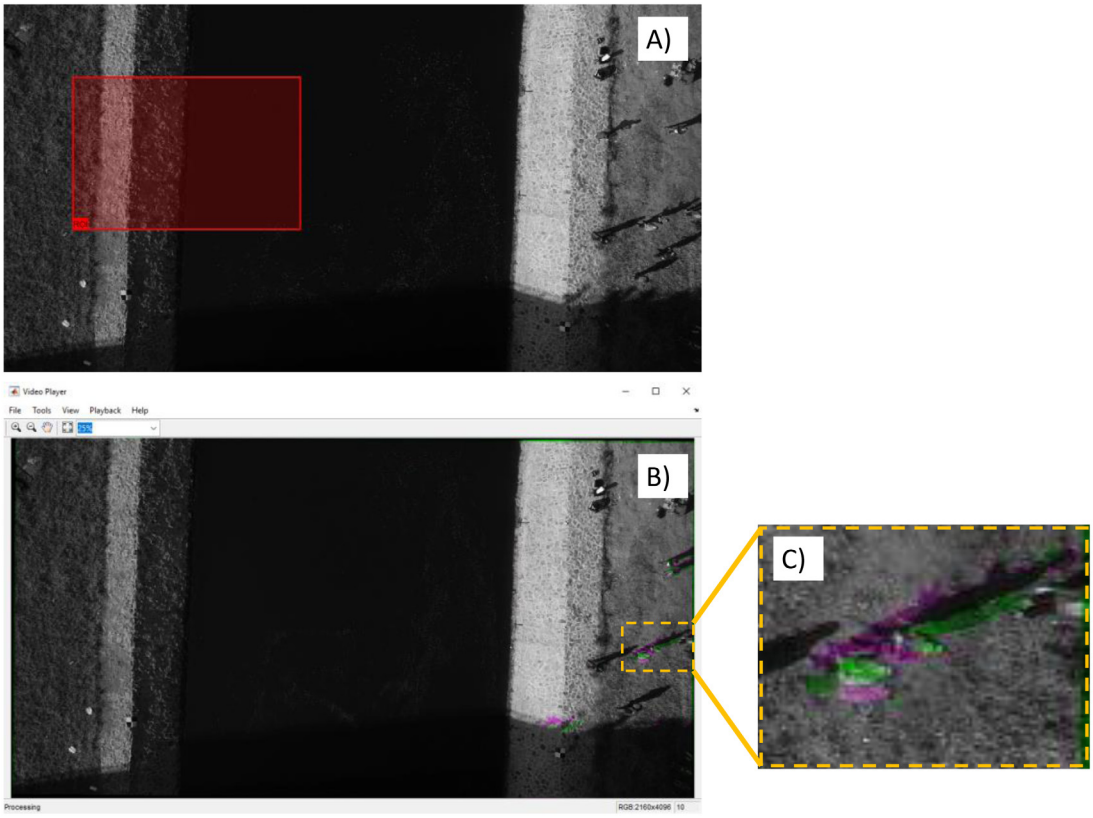

Fig. 3. **(A)** Red rectangle is an example of ROI definition when the user decides to draw it. **(B)** Stabilisation viewer to assess visually in real-time the stabilisation process. **(C)** Magenta and green colours show the movements between the reference and the current frame. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

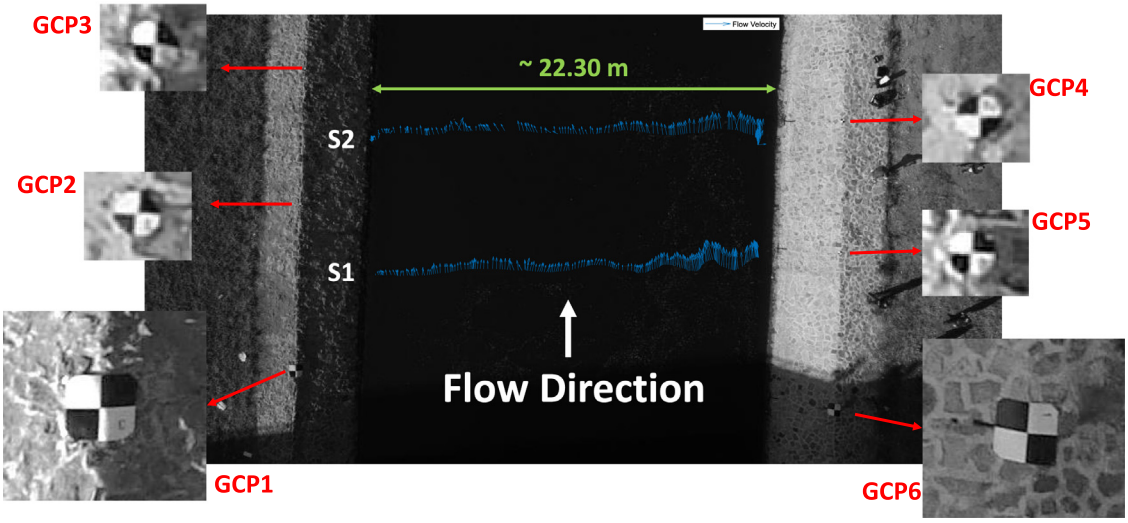

Fig. 4. Belgrade case study with its six GCPs. Blue arrows across the cross-section are the flow velocity vectors measured with the ADCP. S1 and S2 are ADCP-measured Sections 1 and 2, respectively. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

are considered very good, showing average undesired movement reductions greater than 97 and 72% for configuration 1 and 2, respectively.

Configuration 2 is of great importance for image velocimetry analysis as velocimetry algorithms take consecutive frames to compute velocities. In order to explore the role of stabilisation on LSPIV estimates, PIVLab [15] was used to compute surface flow velocities on the raw and stabilised footage. The LSPIV algorithm was applied using the Fast Fourier Transform (FFT) with a one-pass standard correlation method (search and interrogation areas of 128 and 64 px, respectively). Additionally, the 2 × 3-point Gaussian fit was employed to estimate the sub-pixel displacement peak. The images were analysed adopting the sequencing style 1–2, 2–3. Moreover, no post-processing method was applied to filter erroneous velocity results. The mentioned PIVLab settings were the best ones used by Pearce et al. [29], who performed a sensitivity analysis.

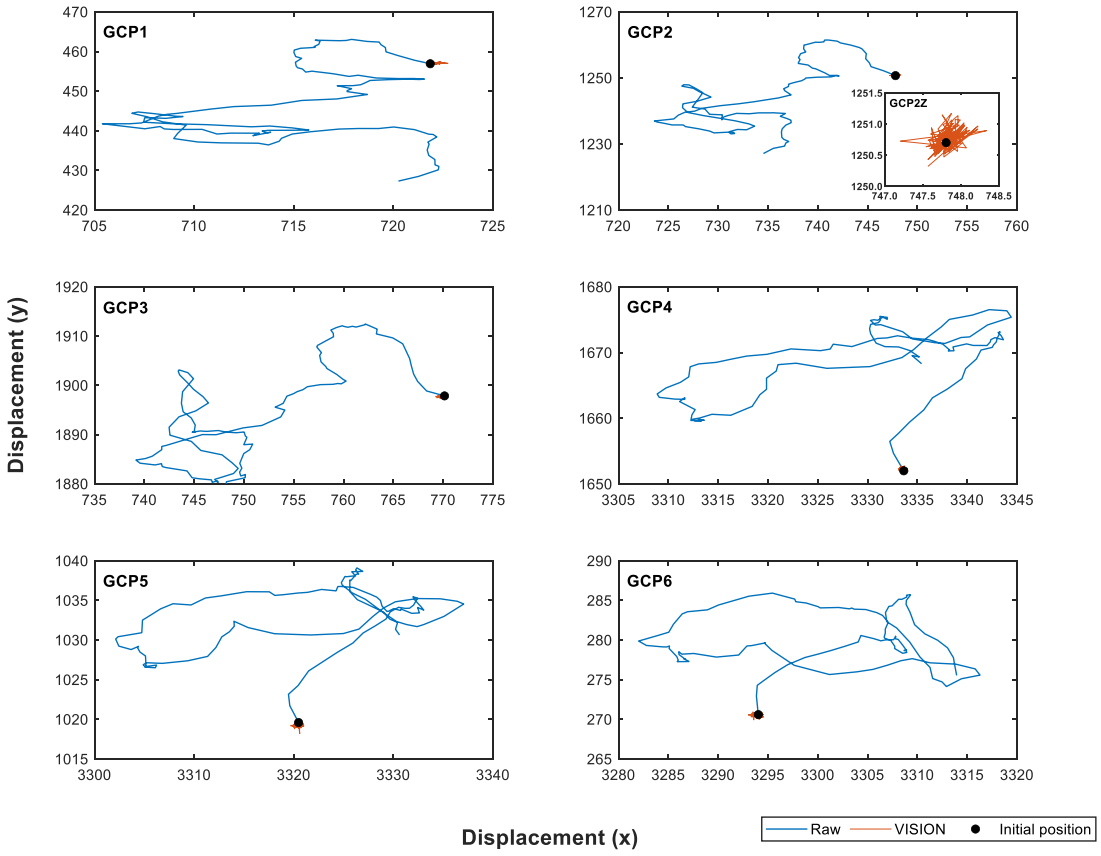

Fig. 5. Trajectories of the GCPs movements throughout the footage using non-stabilised (raw) and stabilised (VISION) data. The black dot is the initial position, whereas the blue and orange lines are the raw and stabilised trajectories. GCP2Z is a close-up considering VISION movements in GCP2. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

Table 1
Displacement magnitude of GCPs for raw and stabilised footage. Minimum, maximum, and average values are reported for the two configurations, i.e. (i) the first frame taken as reference, and (ii) frame-by-frame analysis.

		Displacement magnitude (px)											
		GCP1		GCP2		GCP3		GCP4		GCP5		GCP6	
		Raw	Vision	Raw	Vision	Raw	Vision	Raw	Vision	Raw	Vision	Raw	Vision
1st frame taken as reference	MAX	29.65	0.89	27.82	0.60	33.61	0.90	27.38	0.79	22.31	1.41	22.81	1.01
	AVERAGE	14.26	0.35	16.92	0.21	21.34	0.38	20.41	0.33	16.46	0.43	14.80	0.37
Frame by frame	MAX	2.30	0.71	2.77	0.87	3.05	0.74	4.34	0.61	3.66	1.25	3.72	1.01
	AVERAGE	0.95	0.25	1.01	0.28	1.14	0.26	1.25	0.27	1.14	0.31	1.07	0.29

Fig. 6 shows the velocity magnitude as a function of the distance from the left bank. The black line represents the ADCP data, the blue line is the computed-with-LSPIV surface flow velocities using the raw footage (unstabilised), whereas the orange line is the computed-with-LSPIV surface flow velocities using the footage stabilised with VISION. Velocimetry results are overall in agreement with reference values (RMSE values for raw and stabilised data: 6.79 and 6.30 cm/s for S1, whereas 6.98 and 6.61 cm/s for S2). Therefore, a reduction of 7.14 and 5.37% was met using VISION at S1 and S2. Note that movement reductions for the GCPs placed on the river's left bank are on average less stabilised than the GCPs on the right bank, leading to larger discrepancies between LSPIV estimates and ADCP measurements. On average, the left riverbank GCPs reduced 71.15 and 74.38% for the maximum and average movements, and the right riverbank GCPs reduced 74.88 and 74.70% respectively.

4. Impact, challenges, and future developments

VISION aims to stabilise videos for image velocimetry analysis in rivers using an automatic features selection approach. This approach is ideal for non-experienced users but can also be beneficial for experienced ones. We believe VISION will ease the intercomparison of stabilisation algorithms and methods by providing a simple, parsimonious, and easy-to-use software package. Furthermore, VISION can easily be incorporated in image velocimetry tools written in MATLAB such as KLT-IV, RIVeR, PIVLab, and PTVLab if desired. VISION satisfactorily stabilised the Belgrade footage; however, we must acknowledge that only one field case study was analysed; although promising, further investigations on additional and complex datasets should be carried out to evaluate the uncertainty of the algorithms and software. VISION aims to be an active project, and in consequence, further development is planned: (i) development of a graphical user interface (GUI) to facilitate its use; (ii) provide additional feature detection approaches to increase flexibility; (iii) provide the option of analysing multiple case studies simultaneously; and, (iv) embed a suite to quantify uncertainty within the stabilisation process.

It is worthy to mention that VISION has been also tested on additional two case studies (Basento and Alpine river) in Italy

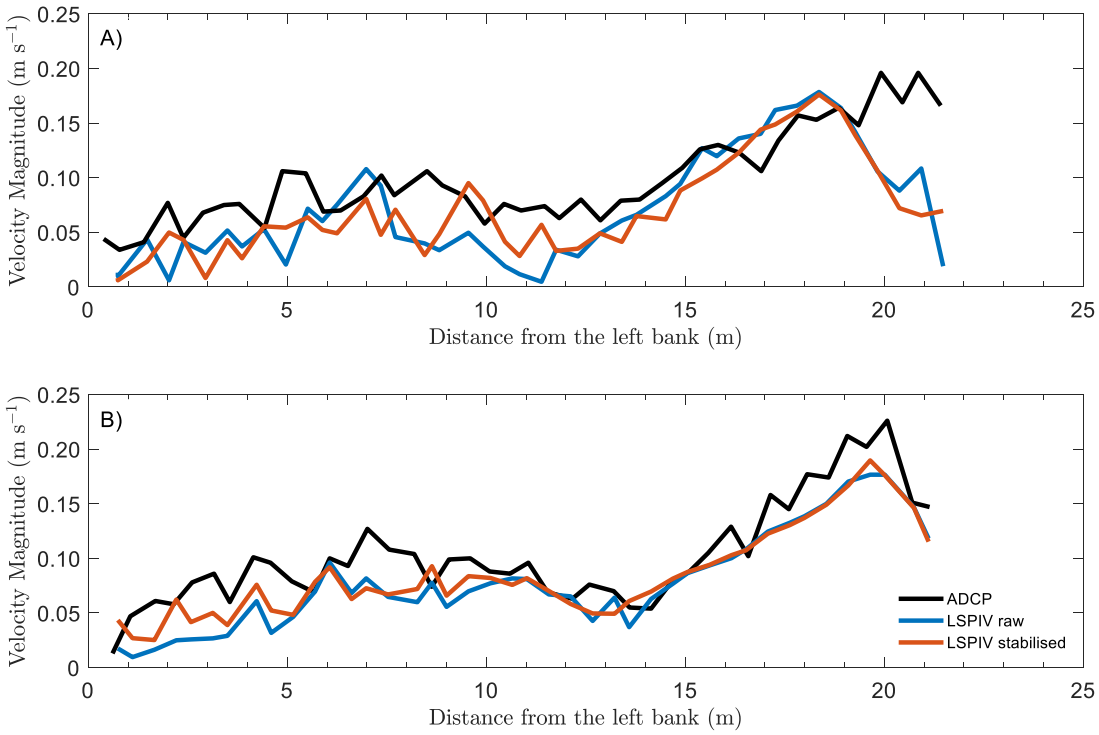

Fig. 6. Cross-sectional analysis of Belgrade case study processed with LSPIV using non-stabilised (raw) and stabilised-with-VISION footage against ADCP data. **(A)** Section S2, **(B)** Section S1.

and Austria in a recent work by Ljubičić et al. [17]. In this study, it is possible to appreciate the good performances obtained by the code in different hydraulic and environmental conditions. In addition, Ljubičić et al. [17] also carried out an intercomparison among different open-access tools highlighting the good performances of VISION. Even though VISION has good performances, it is not always able to stabilise the footage completely. In some cases, it can generate severe jitter artificially or residual motion due to the loss of detectable and matchable features. Nevertheless, it is always expected a reduction of motion.

5. Conclusions

VISION is presented for video stabilisation using automatic features selection aiming at image velocimetry analyses in rivers. VISION is open-source software with a number of implemented options to run under different field conditions depending on the user's needs. Seven feature detection algorithms (with the flexibility to choose two simultaneously) are available, and possibilities to define and use the strongest detected features for stabilisation purposes are also implemented. Additionally, a region of interest and visualisation in real-time of stabilised frames can be set. The Belgrade case study was analysed to show VISION capabilities, and stabilisation performances were presented and discussed. Stabilisation results are considered very good, showing undesired movement reductions greater than 97 and 72% by taking the first frame as a reference and frame-by-frame analysis, respectively. In order to explore the role of stabilisation on image velocimetry estimates, LSPIV was used to compute surface flow velocities on the raw and stabilised footage. A reduction of 7.14 and 5.37% – in terms of RMSE of velocities – was met using VISION at Sections 1 and 2.

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.

Data and code availability statement

The data and code that support the findings of this study are openly available at https://doi.org/10.17605/OSF.IO/HBRF2 [18].

Acknowledgements

We would like to thank Sophie Pearce and Robert Ljubičić for their support and ADCP data over the Belgrade case study. This research was funded by COST Action CA16219, "HARMONIOUS-Harmonization of UAS techniques for agricultural and natural ecosystems monitoring".

References

[1] Pizarro A, Dal Sasso SF, Manfreda S. Refining image-velocimetry performances for streamflow monitoring: Seeding metrics to errors minimization. Hydrol Process 2020. http://dx.doi.org/10.1002/hyp.13919.
[2] Pizarro A, Dal Sasso SF, Perks MT, Manfreda S. Identifying the optimal spatial distribution of tracers for optical sensing of stream surface flow (version 0.1). 2020, http://dx.doi.org/10.17605/OSF.IO/8EGQW, [Codes] OSF.
[3] Dal Sasso SF, Pizarro A, Manfreda S. Metrics for the quantification of seeding characteristics to enhance image velocimetry performance in rivers. Remote Sens 2020. http://dx.doi.org/10.3390/rs12111789.
[4] Muste M, Fujita I, Hauet A. Large-scale particle image velocimetry for measurements in riverine environments. Water Resour Res 2008;44.
[5] Tauro F, Pagano C, Phamduy P, Grimaldi S, Porfiri M. Large-scale particle image velocimetry from an unmanned aerial vehicle. IEEE/ASME Trans Mechatronics 2015;20:3269–75.
[6] Manfreda S, McCabe MF, Miller PE, Lucas R, Madrigal VP, Mallinis G, et al. On the use of unmanned aerial systems for environmental monitoring. Remote Sens 2018. http://dx.doi.org/10.3390/rs10040641.
[7] Detert M. How to avoid and correct biased riverine surface image velocimetry. Water Resour Res 2021. http://dx.doi.org/10.1029/2020wr027833.
[8] Perks MT. KLT-IV v1.0: Image velocimetry software for use with fixed and mobile platforms. Geosci Model Dev 2020. http://dx.doi.org/10.5194/gmd-13-6111-2020.
[9] Perks MT, Russell AJ, Large ARG. Technical note: Advances in flash flood monitoring using unmanned aerial vehicles (UAVs). Hydrol Earth Syst Sci 2016. http://dx.doi.org/10.5194/hess-20-4005-2016.

[10] Eltner A, Sardemann H, Grundmann J. Technical Note: Flow velocity and discharge measurement in rivers using terrestrial and unmanned-aerial-vehicle imagery. Hydrol Earth Syst Sci 2020;24:1429–45. http://dx.doi.org/10.5194/hess-24-1429-2020.

[11] Le Coz J, Jodeau M, Hauet A, Marchand B, Le Boursicaud R. Image-based velocity and discharge measurements in field and laboratory river engineering studies using the free Fudaa-LSPIV software. In: Proc. int. conf. fluv. hydraul. RIVER FLOW. 2014, p. 1961–7.

[12] Le Boursicaud R, Pénard L, Hauet A, Thollet F, Le Coz J. Gauging extreme floods on YouTube: Application of LSPIV to home movies for the post-event determination of stream discharges. Hydrol Process 2016. http://dx.doi.org/10.1002/hyp.10532.

[13] Bay H, Ess A, Tuytelaars T, Van Gool L. Speeded-Up Robust Features (SURF). Comput Vis Image Underst 2008. http://dx.doi.org/10.1016/j.cviu.2007.09.014.

[14] Patalano A, García CM, Rodríguez A. Rectification of Image Velocity Results (RIVeR): A simple and user-friendly toolbox for large scale water surface Particle Image Velocimetry (PIV) and Particle Tracking Velocimetry (PTV). Comput Geosci 2017;109:323–30. http://dx.doi.org/10.1016/J.CAGEO.2017.07.009.

[15] Thielicke W, Stamhuis EJ. PIVlab – Towards user-friendly, affordable and accurate digital particle image velocimetry in MATLAB. J Open Res Softw 2014. http://dx.doi.org/10.5334/jors.bl.

[16] Brevis W, Niño Y, Jirka GH. Integrating cross-correlation and relaxation algorithms for particle tracking velocimetry. Exp Fluids 2011;50:135–47.

[17] Ljubičić R, Strelnikova D, Perks MT, Eltner A, Peña-Haro S, Pizarro A, et al. A comparison of tools and techniques for stabilising unmanned aerial system (UAS) imagery for surface flow observations. Hydrol Earth Syst Sci Discuss 2021;25:5105–32. http://dx.doi.org/10.5194/hess-25-5105-2021.

[18] Pizarro A, Dal Sasso SF, Manfreda S. VISION: Video stabilisation using automatic features selection. OFS. 2021, http://dx.doi.org/10.17605/OSF.IO/HBRF2.

[19] Rosten E, Drummond T. Machine learning for high-speed corner detection. In: Lect. notes comput. sci. (including subser. lect. notes artif. intell. lect. notes bioinformatics), 2006, http://dx.doi.org/10.1007/11744023_34.

[20] Shi J, Tomasi C. Good features to track. In: Proc. IEEE comput. soc. conf. comput. vis. pattern recognit.. 1994, http://dx.doi.org/10.1109/cvpr.1994.323794.

[21] Harris C, Stephens M. A combined edge and corner detector. In: Proc. fourth alvey vis. conf.. 1988.

[22] Leutenegger S, Chli M, Siegwart RY. BRISK: Binary robust invariant scalable keypoints. In: Proc. IEEE int. conf. comput. vis.. 2011, http://dx.doi.org/10.1109/ICCV.2011.6126542.

[23] Rublee E, Rabaud V, Konolige K, Bradski G. ORB: An efficient alternative to SIFT or SURF. In: Proc. IEEE int. conf. comput. vis.. 2011, http://dx.doi.org/10.1109/ICCV.2011.6126544.

[24] Alcantarilla PF, Bartoli A, Davison AJ. KAZE features. In: Lect. notes comput. sci. (including subser. lect. notes artif. intell. lect. notes bioinformatics), 2012, http://dx.doi.org/10.1007/978-3-642-33783-3_16.

[25] Hartley R, Zisserman A. Multiple view geometry in computer vision. 2004, http://dx.doi.org/10.1017/cbo9780511811685.

[26] Fischler MA, Bolles RC. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. Commun ACM 1981;24:381–95.

[27] Torr PHS, Zisserman A. MLESAC: A new robust estimator with application to estimating image geometry. Comput Vis Image Underst 2000. http://dx.doi.org/10.1006/cviu.1999.0832.

[28] Sontek RiverSurveyor S5/M9. Discharge bathymetry and current profiling (brochure). San Diego, CA, USA: Sontek; 2015.

[29] Pearce S, Ljubičić R, Peña-Haro S, Perks M, Tauro F, Pizarro A, et al. An evaluation of image velocimetry techniques under low flow conditions and high seeding densities using unmanned aerial systems. Remote Sens 2020. http://dx.doi.org/10.3390/rs12020232.