Synthetic River Flow Videos for Evaluating Image-Based Velocimetry Methods

G Bodart, Jérôme Le Coz, M Jodeau, A Hauet

Water Resources Research

METHOD
10.1029/2022WR032251

Synthetic River Flow Videos for Evaluating Image-Based Velocimetry Methods

G. Bodart[1,2], J. Le Coz[2], M. Jodeau[1], and A. Hauet[3]

[1]EDF R&D, LHSV, Chatou, France, [2]INRAE, UR RiverLy, Villeurbanne, France, [3]EDF-DTG, St Martin-le-Vinoux, France

Key Points:

- An original method is proposed to generate synthetic videos of realistic river flow scenes with spatially-distributed velocity reference
- The synthetic videos are more suited than available datasets to study environmental error sources of outdoor image-based surface velocimetry
- The strengths and limitations of the method are illustrated through case studies

Correspondence to:
G. Bodart,
guillaume.bodart@inrae.fr

Citation:
Bodart, G., Le Coz, J., Jodeau, M., & Hauet, A. (2022). Synthetic river flow videos for evaluating image-based velocimetry methods. *Water Resources Research*, *58*, e2022WR032251. https://doi.org/10.1029/2022WR032251

Received 24 FEB 2022
Accepted 9 DEC 2022

Abstract Various image-based velocimetry methods have been developed and increasingly used for surface velocity and discharge measurements in rivers. Their evaluation is challenging as in situ comparisons are limited and affected by uncertainties, and synthetic flow images used to test laboratory velocimetry methods are not representative of outdoor applications. A novel method is proposed to generate synthetic river flow videos with known surface velocities. The method is based on open-source computer graphics tools: Blender and Mantaflow. The synthetic videos represent realistic situations where the scene configuration, for example, riverbed and riparian areas, water aspect, camera location and orientation or lighting can be easily modified. Two cases studies emphasize the strengths and the limitations of the proposed method. The synthetic datasets generated will allow the improvement of different image-based velocimetry methods, notably through collaborative evaluations or intercomparisons.

1. Introduction

Over the last decades, digital cameras have become simple and accessible tools impulsing a huge interest from the river science community for image-based velocimetry solutions (Muste et al., 2008; Pearce et al., 2020; Perks et al., 2020). Image-based methods allow a non-intrusive measurement of the two-dimensional surface velocity field over large areas, with interesting spatial and temporal resolutions. These methods derived from the well-known *Particle Image Velocimetry* (PIV) method (Keane & Adrian, 1992) used in laboratory experiments. In PIV applications, seeding particles are illuminated by a laser sheet and they scatter light toward a camera perpendicular to the flow. The resulting images show highly contrasted groups of particles with virtually no perspective or lens distortion (cf. Figure 1a). Displacements between consecutive frames are measured with sub-pixel accuracy thanks to a pattern matching algorithm. They are converted to velocity in physical units using the camera calibration matrix and knowing the elapsed time between frames. In the late 90's, Fujita et al. (1998) adapted this technique to determine the surface velocities of open channel flows, renaming it as *Large Scale Particle Image Velocimetry* (LSPIV). Compared to PIV experiments, usual LSPIV tracers are fickle singularities (e.g., turbulence, foam, boils) or floating objects (e.g., debris, artificial seeding) most of the time illuminated by sun light (cf. Figure 1b). To cope with oblique points of view, LSPIV encompasses an ortho-rectification step prior to velocity estimation. The camera calibration matrix is first determined, based on Ground Control Points (GCPs) and then used to create physically scaled ortho-images corrected from perspective distortion.

Several other image-based methods have been proposed recently. Space-Time Image Velocimetry (Fujita et al., 2007) and Space-Time Volume Velocimetry (Tsuji et al., 2018) use spatio-temporal images to determine the velocity. Variational methods as used by Khalid et al. (2019) determine the displacement based on a specific formulation of the spatio-temporal derivatives of the image intensity. Particle Tracking Velocimetry use either pattern matching (Brevis et al., 2011; Eltner et al., 2020) or variational methods (Lin et al., 2019; Perks, 2020) in a Lagrangian approach. Various applications of these techniques can be found such as real-time discharge measurement at fixed positions, for example, (Hauet, Kruger, et al., 2008; Peña-Haro et al., 2021), airborne discharge measurements, for example, (Detert et al., 2017; Eltner et al., 2020; Fujita & Kunita, 2011; Perks et al., 2016), or urban runoff measurement, for example, (Legout et al., 2012; Leitão et al., 2018). Reviews can be found, either on the methods (Muste et al., 2008; Pearce et al., 2020) or on application cases (Perks et al., 2020).

Image-based methods use camera calibration and optical flow measurement to obtain surface velocities. Camera calibration consists of solving a particular formulation of the pinhole model to determine the camera intrinsic parameters (principal points, focal length, skewness factors) and extrinsic parameters (position and orientation). These parameters are used to link the image coordinate system in pixel to the scene coordinate system in physical

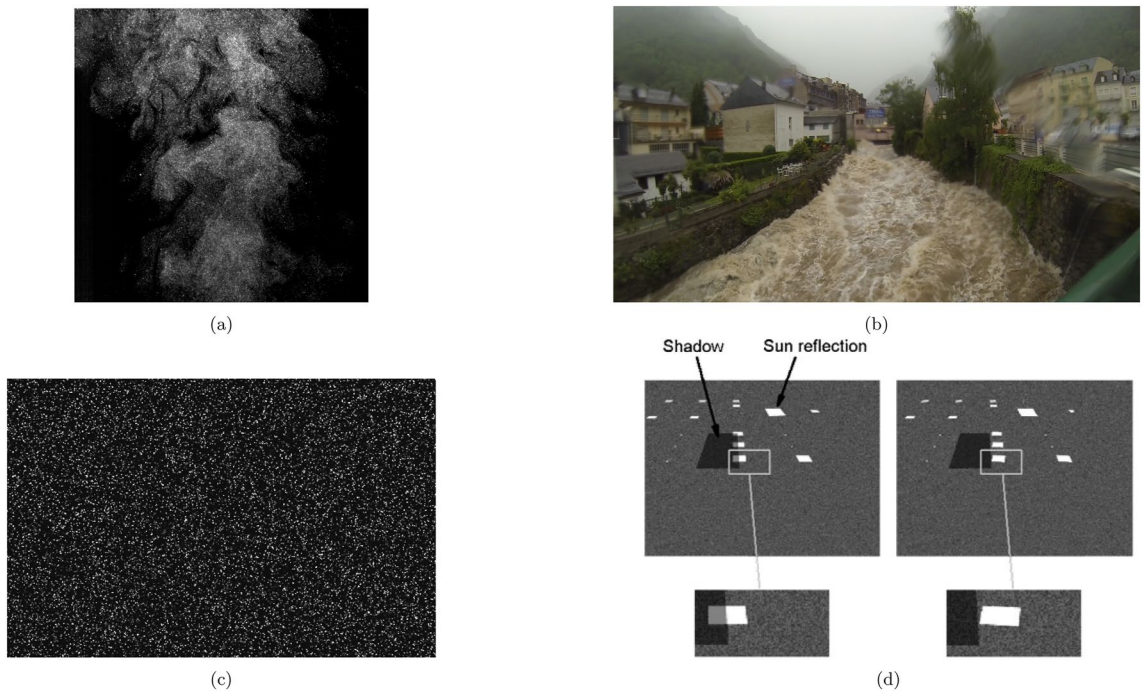


Figure 1. Example of (a, b) real and (c, d) synthetic images used in (a, c) laboratory and (b, d) river image velocimetry. Images are extracted from (a) Stanislas et al. (2008) (b) Le Coz et al. (2014) (c) Kähler et al. (2016), and (d) Hauet, Creutin, and Belleudy (2008).

units. Several camera calibration procedures can be found with their own strengths and limitations (Zhang, 2000). Implicit calibration based on GCPs (Jodeau et al., 2008) is often used in field measurements. The number of GCPs, their spatial distribution and the precision of their real-world and pixel coordinates directly affects the calibration and thus the velocity measurement (Le Coz et al., 2021). Camera calibration must be adapted to the camera specificity for example, fish-eye lenses (Detert, 2021). Other sources of error such as low viewing angles or camera vibrations may also affect the image-based velocity measurement. Optical flow corresponds to the variations of light intensities from one frame to another in a given image sequence. Horn (1986) formulated the *brightness constancy assumption* which states conditions to ensure equality between the optical flow and the real motion field on the imaged scene. The intensity and position of the light source have to be constant through time, and the scene objects have to reflect some of the incident light (i.e., objects are neither transparent nor a black body). Verri and Poggio (1989) considered that the imaged surface has to be sufficiently textured and Lambertian, that is, perfectly matte, so that the luminance is the same regardless of the viewing angle. Such ideal conditions can be difficult to meet in fluvial environments where the surface appearance can strongly vary. Specular and transparent water can be mixed with diffuse particles such as foam at the surface or suspended sediments within the water column. Various waves (e.g., boils, gravity capillary waves) or floating objects (e.g., debris, foam) can texture the water surface. Their spatial and temporal distribution may fluctuate. Surface waves can also propagate with different celerity and direction than those of the overall flow velocity, for example, gravity capillary waves (Dolcetti et al., 2020). Several artifacts such as shadows, glares and occlusions can adversely affect the optical flow measurement. Filtering optical flow measurements is crucial to face up these conditions, as pointed out by Detert (2021).

It is of paramount importance to better understand the impact of these error sources in order to promote the operational deployment of image-based velocimetry techniques. However, the in situ evaluation processes are actually limited. On the one hand, the measurement conditions for example, lighting, water aspect, camera location and shooting angle, are often imposed and cannot be varied easily. On the other hand, the reference data may come with large uncertainties due to the measurement process and the transformations required before comparison. If discharge is the reference for comparing the results, the uncertainty typically ranges from 5% up to 20% for medium or high in-bank flows according to McMillan et al. (2012). Discharge must be computed from surface velocities measured with image-based velocimetry which adds other sources of error, for example,

surface coefficient, water level and bathymetric profile. If a surface velocity field is the reference for comparing the results, it should be derived from velocity measurements. Various tools can be used: radars capture the velocity at the surface, current meters at the near-surface and Acoustic Doppler Current Profilers (ADCP) under the surface. Radar measurements must be made from a bridge under sufficient velocity conditions. ADCP velocity measurements must be extrapolated to get the surface velocities. Extrapolation errors can occur if the vertical velocity profile is poorly estimated or due to the wind shear or flow non-uniformity for instance (Mueller, 2013).

Synthetic imaging is an interesting solution to overcome these limitations. The reference motion field and the scene properties are known precisely and can be methodically controlled and tested. In the field of laboratory PIV, synthetic imaging has been used intensively to evaluate optimal design rules (Keane & Adrian, 1992), to quantify the precision of the methods (Willert, 1996) or to challenge alternative algorithms, cf. the four "PIV challenges" (Kähler et al., 2016; Stanislas et al., 2003, 2005, 2008). The optical aspect of laboratory PIV images, that is, white particles on dark background, is easy to reproduce synthetically (cf. Figure 1c). Several parameters such as the tracers spatial distribution, the amount of in-field and out-of-field particles, the velocity distribution and the particle diameters can be varied to simulate a broad range of measurement conditions, cf. for example, (Lecordier & Westerweel, 2004). More complex situations can be reproduced including optical aberration, for example, defocusing and astigmatism (Rossi, 2019) or complex velocity profiles, for example, shear flows, vortices (Mendes et al., 2020). Several contributions in outdoor image-based surface velocimetry were based on such synthetic images, that is, white particles on dark background. Pizarro et al. (2020) analyzed the impact of the tracers spatial distribution on the accuracy of the velocity measurement. Pumo et al. (2021) investigated the optimal set-up in terms of parameters depending on various measurement conditions such as tracers size and spatial distribution or velocity profile. Hauet, Creutin, and Belleudy (2008) proposed a synthetic image generator that reproduces the image formation through a virtual camera placed along the river. They used a simple hydraulic model coupled to a vertical logarithmic velocity profile to get the surface flow velocities based on given channel geometry and discharge. Camera location, orientation and parameters are used to create the synthetic images. On the images, tracers and shadows are represented as white and dark rectangles, respectively (Figure 1d).

These contributions helped to analyze the individual effect of various error sources on the image-based velocimetry measurement, for example, tracers spatial distribution, velocity profile or camera tilt angle. Several important error sources mentioned earlier are still missing in these studies: optical flow errors (i.e., light-related errors) and camera calibration errors. For instance, how does the surface appearance impact the velocity measurement? How do errors in GCPs pointing affect the measurement with different camera tilt angles? The interactions of optical flow errors and camera calibration errors with the measurement conditions may also be investigated. How does a given water aspect influence the measurement depending on the lighting conditions and the shooting location? The synthetic image sequences currently available are limited to study such questions as several aspects of the scene are not reproduced such as lighting conditions, scene occlusions (e.g., trees, buildings) or water appearance. This motivated the development of a new method for generating realistic videos of synthetic river scenes. The aim is to reproduce a broad range of measurement conditions to study the aforementioned issues. This method is based on open-source computer graphics solutions, namely Blender (Blender Community, 2020) and Mantaflow (Thuerey & Pfaff, 2018). Blender is a 3D creation suite, freely available and open source (GNU GPL license), developed by the "Blender Community." The software supports the entire 3D pipeline: the modeling of the scene objects, their animation through time, the simulation of physical processes (e.g., forces, fluids, wind) and the rendering of the scene through virtual cameras. Tutorials and guidelines can be found on-line thanks to a big user community. Mantaflow was initially developed by the Computer Graphics Laboratory of ETH Zurich and is now maintained by the Thuerey Group from the Technical University of Munich. It is an open-source framework targeted at fluid simulation research in Computer Graphics. It consists of a parallelized C++ solver core and a Python scene definition interface. Various Navier-Stokes solvers allow to simulate either fluids or gazes. Mantaflow has served as the simulation engine in Blender since version 2.80. In computer graphics, fluid simulation methods are designed to limit numerical dissipation (i.e., ensure stability) to provide a correct appearance of the fluid even in degraded simulation scenarios, for example, large time steps (Lentine et al., 2012). This concern differs from that of computational fluid dynamics tools which focus more on physical accuracy. In the scope of our work, the surface appearance is of greater concern than the physical accuracy of the simulated flow. In other words, the fluid simulation is acceptable when it produces a realistic flow appearance and water surface texture, even if the flow simulation is not fully accurate. The exact knowledge of the simulated velocities is more

Table 1
Overview of the Proposed Methodology

Creation of the scene	Creation of the three-dimensional objects
	Layout of the objects on the scene
Fluid simulation	Simulation setup
	Generation of water particles
	Construction of fluid mesh
Rendering	Scene setup
	Lighting setup
	Water aspect setup
	Creation of synthetic image sequence
Reference velocities extraction	Interpolation of velocities at fluid mesh vertices
	Filtering
	Conversion to physical units
	Interpolation on a regular grid

important for comparison and evaluation of image-based velocimetry techniques than the physical accuracy of the simulated flow.

The aim of this paper is to present our method and discuss its strengths and limitations in evaluating image-based velocimetry techniques. Section 2 describes the methodology for creating a synthetic image sequence of a river scene with Blender and Mantaflow and for extracting surface reference velocities. The step-by-step procedure is detailed and illustrated through an example. Recommendations are made on several parameters to obtain a realistic river scene. Two application cases are proposed in Section 3, based on the fluvial environment built in Section 2. LSPIV measurements are carried out in both case studies and compared with the reference velocities. The aim of the first case study is to evaluate the accuracy of the reference velocity extraction procedure. The second case study emphasizes the possibilities of the proposed method to capture environmental error sources quantitatively. The strengths, limitations and possible improvements of the method are discussed in Section 4. Conclusions and perspectives are presented in Section 5.

2. Generation of Synthetic River Flow Images

2.1. Overview

In this section, the method is described step-by-step and illustrated with the creation a synthetic video of a simple fluvial environment. Blender 2.90 with its included version of Mantaflow 0.13 are used to build up the proposed example. It is important to mention that only a subset of Mantaflow features are exposed to Blender. This means that several tools, for example, K-epsilon turbulence modeling, cannot be used in Blender. The overall step-by-step procedure is presented in Table 1. The four steps: (a) creation of the scene, (b) fluid simulation, (c) rendering, and (d) reference velocities extraction are detailed hereafter. The Fluid Implicit Particles method (Brackbill & Ruppel, 1986) is used for the fluid simulation. It consists of a Navier-Stokes solver using a semi-Lagrangian scheme. Particles represent the fluid at each time-step and grids are used to compute and store several quantities such as velocities for instance. In Blender, only wall boundary conditions can be configured. The initial conditions are defined by the shape and location of an inflow object on the scene and the three-dimensional inlet velocities. Fluid particles are generated from the volume of the inflow object at each time-step. Such conditions differ from hydraulic modeling tools where stage, discharge, bed roughness and stage-discharge relation may be used. The boundary conditions, the initial conditions and the simulation parameters have an impact on the generated fluid dynamics. They must be configured through a trial-and-error approach to obtain the desired surface texture and flow regime. Once the fluid is simulated, the camera, the lighting and the water aspect are set. The rendering step consists of building the image sequence of the scene through the virtual camera. Finally, the reference velocities are extracted from the simulation data, converted to conventional units and interpolated on a regular grid.

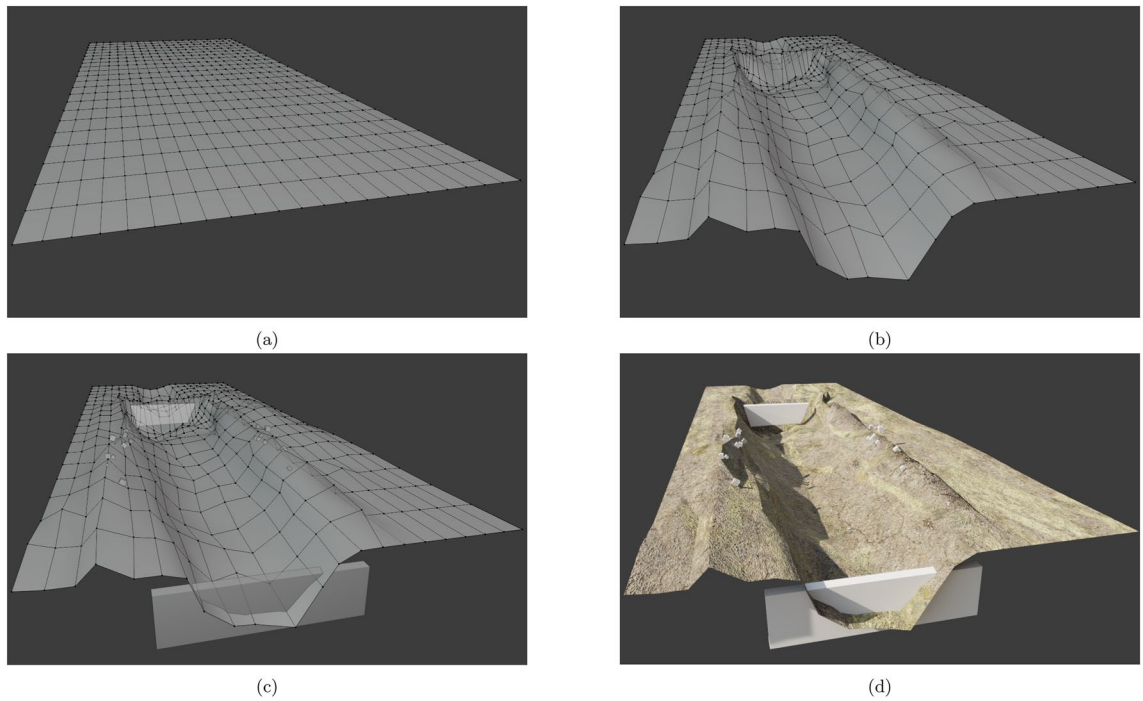


Figure 2. Screenshots from Blender while creating a simple river scene: (a,b) the creation of the river bed with (a) the initial rectangular mesh and (b) the digitally sculpted channel and (c) the layout of the scene objects (inflow object, weir and cubes used as GCPs) and (d) the final appearance of the scene with textures and light.

2.2. Creation of the Scene

The first step is to build up and layout the scene elements. Three dimensional meshes or objects can be imported in Blender with various format. This means that predefined riverbed geometry could be used for instance. In the scope of this article, the objects are shaped under Blender as it offers various possibilities such as modeling (e.g., extrude, bevel, shear) or 3D sculpting (e.g., crease, inflate, scrape). By default, scene units are in meters and degrees. In the proposed example, the simplest fluvial environment consists of a channel and cubic objects used as ground control points (GCPs). The channel is initially defined as a 35 m-long and 15 m-wide plane with a slope of 0.436% (0.25°). This plane is first sampled to a rectangular mesh of twenty by twenty faces (cf. Figure 2a) and then digitally sculpted (cf. Figure 2b) in a dedicated Blender tab. During the 3D sculpting process, the mesh vertices are displaced following a given transformation to reproduce typical actions like sculpting in clay, for example, pinch, scrape or grab. To create the riverbed, the central vertices are moved mainly in the negative vertical direction. To create the river banks, vertices around the riverbed are moved mainly in the positive vertical direction. The sampling resolution of the plane mesh is low but sufficient to define the geometry of a simple channel as shown in Figure 2b.

Objects can be added to the channel as wall boundary conditions to influence the fluid simulation. In the example, a weir of about 1 m height has been placed downstream (cf. Figure 2c). The inflow object placed upstream consists of an extruded trapeze which almost fits the river cross-section (cf. Figure 2c). The inlet velocity is 4 m/s in the stream-wise direction. A high inlet velocity and a rougher riverbed geometry near the inflow are used to promote turbulence generation. Turbulence is useful to texture the water surface. Different inlet velocities could be used to generate slower or faster, smoother or rougher water surface. Mantaflow's features embedded in Blender do not allow for additional parameters such as discharge, vertical velocity profile, bed roughness or turbulence generation to be added to control the flow characteristics. In the example, the objects representing the GCPs are simple cubes with various sizes: 0.15 m, 0.2 and 0.25 m. They are distributed on each side of the flow (cf. Figure 2c). Targets are printed on each face of the cubes to identify their centers. Each center can then be used as a GCP as its exact location can be extracted from Blender. A soil texture is applied to the riverbed and riparian areas. Figure 2d shows the appearance of the created scene.

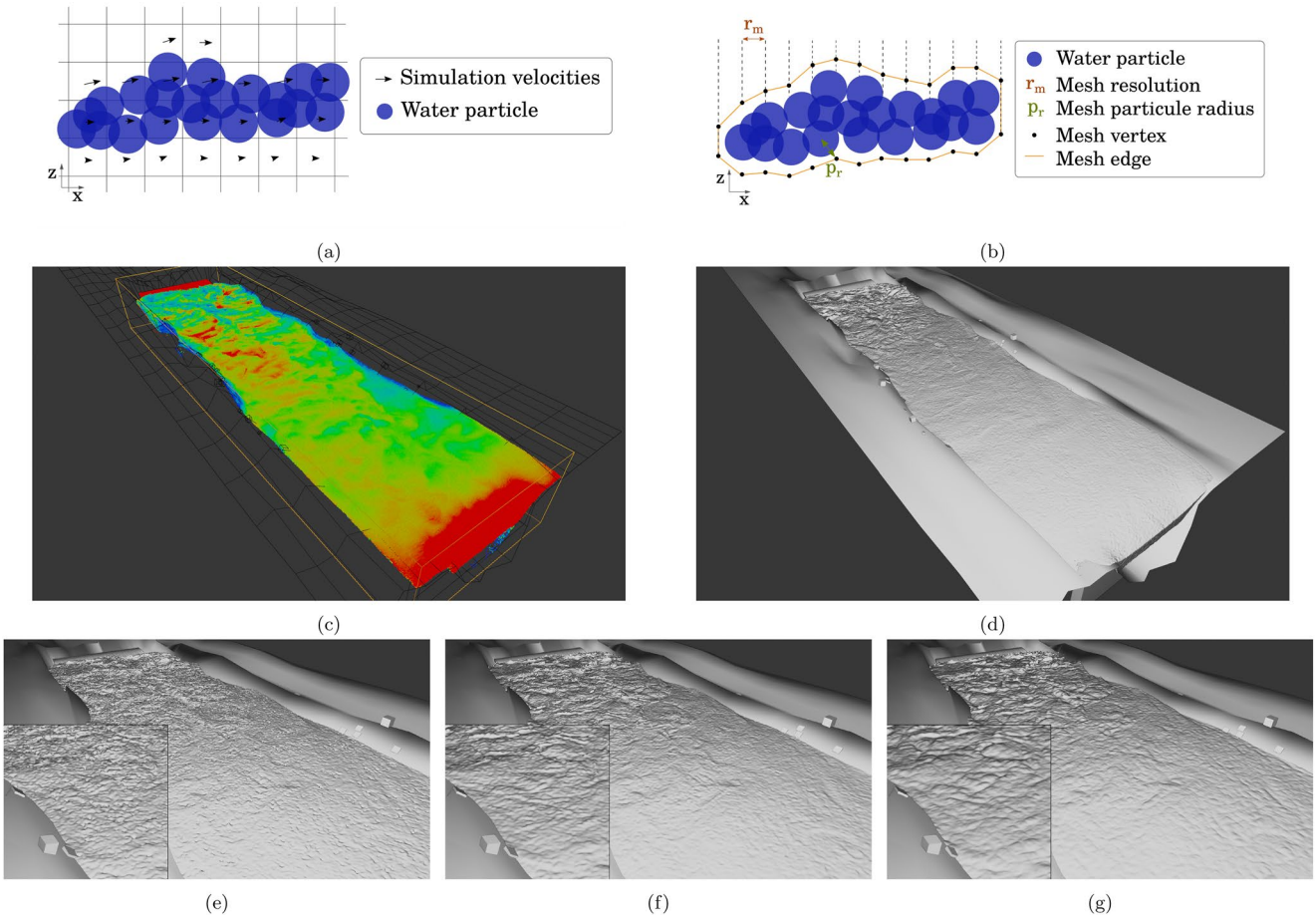


Figure 3. Illustrations of the fluid simulation steps: (a) the generation of the fluid particles and (b) the creation of the fluid mesh around the particle cloud. The illustrations represent a vertical slice of the simulation domain, that is, XZ axes. Screenshot of the fluid simulation results of the example case: (c) fluid particles (colored spheres) within the fluid domain (yellow bounding box) and (d) fluid mesh (gray surface representing the water). Examples of mesh generated with various mesh particle radius p_r: (e) $p_r = 1$, (f) $p_r = 2$, and (g) $p_r = 4$.

2.3. Fluid Simulation

The fluid simulation process is made of two major steps: the simulation of the fluid particles (cf. Figure 3a) and the creation of a mesh around these fluid particles (cf. Figure 3b). For simplicity, we detail only the most important parameters that have to be set for fluid simulation. Information on the other parameters can be found in the Blender documentation.

The simulation area, or fluid domain, is defined on the scene by a bounding box (Figure 3c) containing the channel and the inflow and weir objects. To optimize the fluid simulation time, this fluid domain should be as tight as possible around the flow area. The spatial sampling of the three-dimensional grid is set through the resolution division parameter. This parameter defines the number of cells along the longest side of the domain. Higher resolution allows for finer velocity grids hence finer surface description but at higher computational costs. A sufficiently high resolution is mandatory to obtain a realistic surface appearance. In the proposed example, the fluid domain size is about 30 m-long, 8 m-wide and 3 m-deep. A resolution division parameter of 768 is used which leads to a grid cell size of 0.04 m.

A minimum and a maximum number of particles per cell can be set. These parameters constrain the spatial density of particles along the fluid domain. It also defines the maximum number of particles. A larger amount of particles allows for finer surface deformations at the cost of a larger memory footprint. In the example, the minimum and maximum limits are 1 and 4 particles per cell respectively. Low thresholds are used as the resolution

of the simulation is already high (0.04 m/cell). The total amount of particles generated per frame of interest is about 13.5×10^9.

A particle radius, expressed in simulation cells, that defines the volume covered by a fluid particle can be configured. It determines how much volume around the particles can be considered as liquid. The particle radius can be adjusted if the simulation appears to leak or gain volume in an undesired way. For instance, if too much liquid is being produced the particle radius can be decreased. This parameter directly impacts the fluid volume and thus the discharge. In the example the particle radius has the default value of 1. The water particles generated can be seen in Figure 3b.

To get the final fluid object, a closed mesh have to be built around the particle cloud at each time-step. The mesh generation parameters are the resolution factor, the mesh particle radius, the smoothing coefficients and the concavity bounds. Their configuration may ensure a correct visualization of the water particle cloud. In other words, the generated mesh have to reproduce correctly the particles motion at the free surface. In case of an inappropriate configuration, the mesh may show artifacts such as holes, too sharp or too smooth edges that are not realistic. The mesh generation parameters are interdependent and should be set carefully. The mesh resolution is determined as the product of the resolution factor and the resolution of the simulation grid (cf. Figure 3b). This spatial refinement is useful to enhance the level of details at the water surface but it increases the memory footprint. A resolution factor of 2 is used in the example which leads to a mesh resolution of 0.02 m. The mesh particle radius corresponds to the area covered by a water particle and is expressed in mesh cell units. The mesh cell size differs from the simulation cell size if the resolution factor is greater than 1. A too large particle radius tends to smoothen the surface and thus reduce the amount of details (cf. Figure 3g). A too short particle radius can creates holes or artifacts in the mesh (cf. Figure 3e). A balance has to be found with the other mesh generation parameters to obtain a correct visual aspect of the surface (cf. Figure 3f). The mesh generation configuration used in the example is detailed in Table 2 in Section 3.1. The generated mesh is shown Figure 3d.

In addition to the flow, three groups of diffuse particles can be generated: bubbles within the water column, foam at the free surface and spray above the surface. They are generated from the data of the particle-based fluid simulation as a post-processing step following the method of Ihmsen et al. (2012). Only foam particles are purely advected by the flow. Diffuse particles make the flow aspect more realistic, however their generation is expensive in time and memory space: they increase the simulation execution time and memory space by up to 10 and 15 times respectively. For this reason, no diffuse particles are used in the proposed example.

2.4. Rendering

The rendering step consists of building the image of the scene through a virtual camera. It is based on the rendering parameters, the objects geometry and position, the materials aspects and the camera and lighting parameters. The materials aspects are defined in Blender through block shaders (Abram & Whitted, 1990; Cook, 1984). A shader represents a particular aspect such as metallic, glass, diffuse or transparent, with several parameters, for example, refraction index, color, roughness. Shaders can be combined to create complex optical aspects. The lighting can be set by placing lights on the scene or by using a background image. The background image is usually a High Dynamical Range Image (HDRI) with a field of view (FoV) of 360° (cf. Figure 4a). A large variety of such HDRIs are freely available online. HDRIs are usually stored in 32-bit files. The high dynamical range of the pixels intensity allows to have higher contrast than traditional 8-bit images which is useful to correctly represent strong light variations, for example, sun light and shadows. The scene is initially placed in the center of the HDRI. The pixel colors and intensities of the background HDRI are interpreted as light colors and intensities on the Blender scene. HDRIs are very useful to create various outdoor scenarios (cf. Figure 5.

Blender proposes two kinds of virtual cameras: perspective and orthographic cameras. Orthographic cameras generate images without perspective which means that all light rays reaching the sensor are parallel to the optical axis (cf. Figure 4d). Orthographic cameras could be used to study the performances of velocimetry methods regardless of the ortho-rectification step. For such cameras, the FoV is defined in meters as the longest side of the image. For traditional perspective cameras, the FoV is specified in degrees or through the focal length (f) in millimeters. The camera location and orientation can be configured to reproduce different recording viewpoints (cf. Figures 4b and 4c). These settings can also be animated over time to simulate camera movements, which can be useful to assess the effect of handheld camera shaking for instance. The calibration matrix (i.e., intrinsic and

Table 2
A Summary of the Parameters Used for the Two Case Studies

Simulation parameters		Mesh parameters	
Domain size -X	30 m	Resolution factor	2
-Y	8 m	Obtained resolution	0.02 m/cell
-Z	3 m	Mesh particle radius	1.2
Resolution division	768	Smoothing positive	2
Resolution obtained	0.04 m/cell	Smoothing negative	2
Frame rate	30 fps	Concavity upper	2.7
Number of frames	4,500	Concavity lower	0.2
Particle radius	1		
Particle per cell min.	1		
Particle per cell max.	4		
Inflow velocity	4 m/s		
Weir height	1 m		
Scene parameters	**Case 1**	**Scene parameters**	**Case 2**
Image size	3,120 × 2,640	Image size	1,920 × 1,080
Lighting	Homogeneous	Lighting	HDRIs
Camera type	Orthographic	Camera type	Perspective
Camera FoV	10.6 m	Camera FoV	90°
Camera location	Center above water	Camera location	Left bank by the water
Camera orientation	Nadir	Camera orientation	Pointing upstream
LSPIV parameters	**Case 1**	**Scene parameters**	**Case 2**
Frames analyzed	200	Frames analyzed	200
Frame rate	30 fps	Frame rate	15 fps
Orthorectification	Scaling	Orthorectification	GCP based
Resolution	0.0034 m/pix	Resolution	0.0115 m/pix
IA	60 pix	IA	32 pix
SA_{y-}	10 pix	SA_{y-}	5 pix
SA_{y+}	10 pix	SA_{y+}	5 pix
SA_{x-}	0 pix	SA_{x-}	0 pix
SA_{x+}	25 pix	SA_{x+}	14 pix

extrinsic parameters) of the virtual camera can be obtained swith a simple Python script executed in Blender. These parameters could be used to compute the exact position of the GCPs on the image, based on their locations in the scene. Such data may be of interest to evaluate various ortho-rectification errors, especially GCP pointing errors.

To build up the synthetic images, two kinds of rendering engines can be found: real-time and physically-based engines. Real-time engines, for example, (Markosian et al., 1997), are mainly used in video games as they are much faster than physically-based rendering engines. Real-time engines are non-photorealistic which means that they have difficulties in reproducing complex optical behaviors like water refractions and particular shadows. Physically-based engines are based on ray tracing algorithms, cf. (Cook et al., 1984; Whitted, 1980). They accurately reproduce these complex situations at the cost of longer computational time. Our objective is to study the impact of various water aspects and adverse situations (e.g., glares, shadows) on the image-based measurement, without specific considerations about the simulation time. Thus we only consider the physically-based rendering engine to ensure photorealistic results. Figure 5 shows various images rendered using Cycle, the physically-based rendering engine of Blender 2.90. Different water aspects and background images are used to simulate various

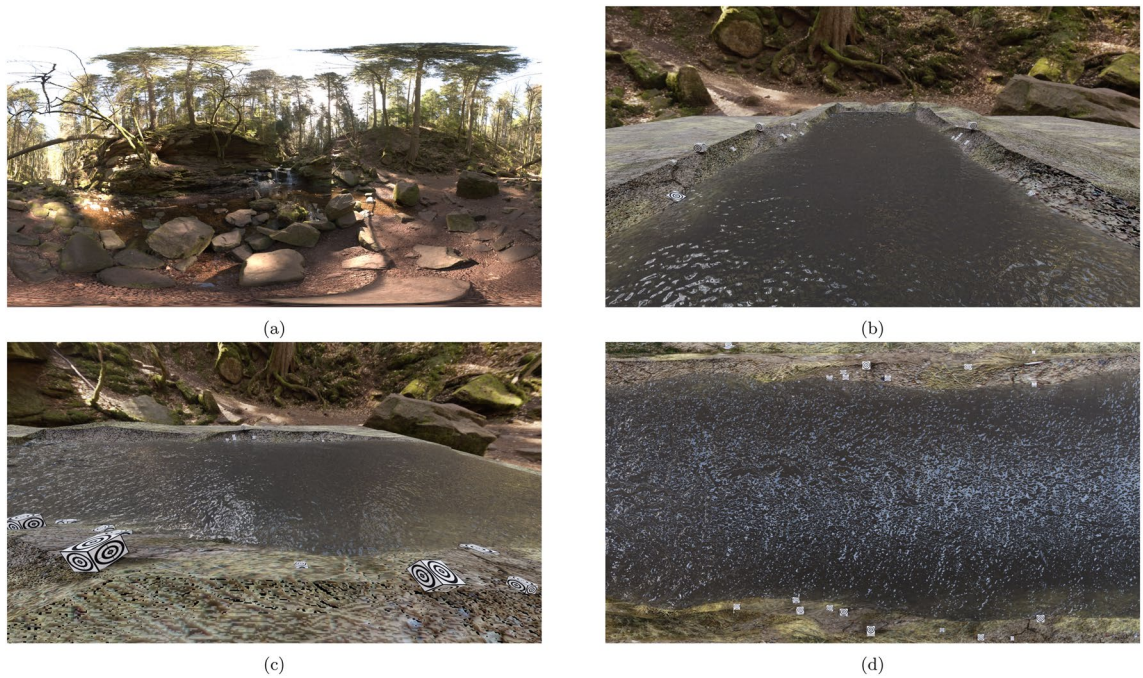


Figure 4. Renders of the proposed example (same fluid simulation) with (a) the HDRI used as the scene background and various cameras: (b) a perspective camera with $f = 18$ mm (FoV = 90°) placed on a bridge pointing upstream, (c) a perspective camera with $f = 8$ mm (FoV = 132°) placed on a bank pointing to the other bank and (d) an orthographic camera with FoV = 16.2 m recording from sky.

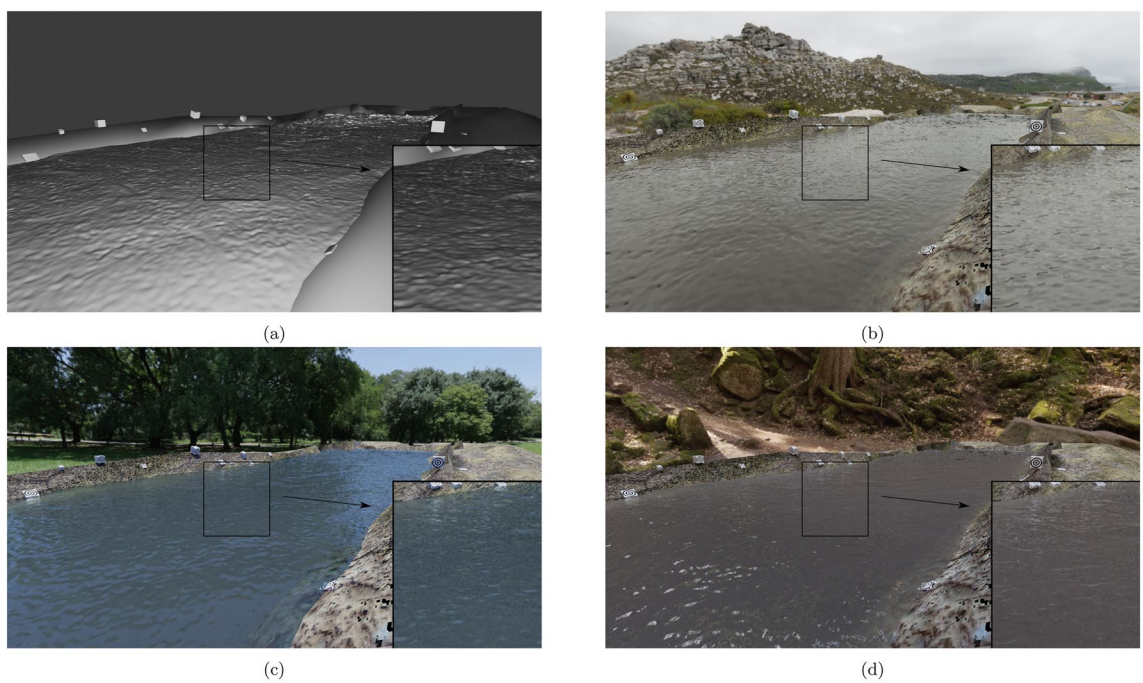


Figure 5. Renders of the example scene with various water aspects and scene environment. (a) The Blender "mesh-view," a non photo-realistic view with an opaque an highly textured water aspect. (b) A cloudy environment with a brownish and diffuse water aspect. (c) A sunny environment with a smooth shadow due to a tree and light scattering effect in the water (i.e., noise at the water surface). (d) A forest environment with light occlusions due to trees and direct sun light in several areas. A zoom on the area delimited by a dark rectangle is added in the lower right corner of each figure.

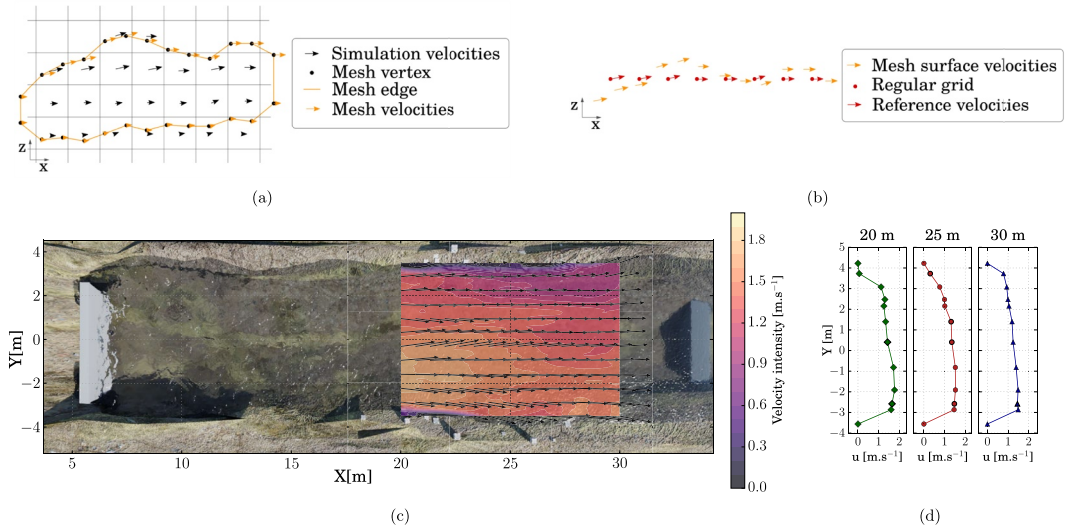


Figure 6. Illustration of the reference velocities extraction process: (a) simulation velocities interpolated on the fluid mesh and (b) filtering and interpolation of the mesh surface velocities on a regular grid. The illustrations represent a vertical slice of the simulation domain, that is, XZ axes. (c,d) Representations of the reference velocities extracted with (c) the time-averaged surface velocity field in the area of interest and (d) the lateral time-averaged surface velocity distributions upstream, midway and downstream of the area of interest.

measurement conditions. These images highlight the capabilities of Blender to generate realistic water aspects. Glares, shadows, occlusions of light and noise on the water surface are well reproduced. Different tracer shapes and spatial distribution are obtained from the same fluid simulation. This shows that various surface appearances can be obtained even with the same three-dimensional surface texture. Other features such as motion blur could be used to make sequences even more realistic.

2.5. Extraction of the Reference Velocities

To evaluate the image-based velocimetry methods, the reference velocities at the water surface have to be retrieved, in physical units, that is, m/s or pix/frame. They are extracted from the simulation data. The extraction process can be summarized as follows: (a) interpolation of the simulation velocities at the fluid mesh vertices, (b) filtering to retain only the surface velocities, (c) conversion to physical units and (d) interpolation on a regular grid. Such framework is necessary as the surface velocities are not computed during the simulation. For the extraction process a modified version of Mantaflow and a dedicated Python library have been built. A function for reference extraction has been added to the original version of Mantaflow. The Python library is used to filter, convert, interpolate and analyze the reference data. These tools are attached with this article, cf. Section 7.

During the simulation, the fluid particles are moved based on the simulation velocities. A closed mesh is then built around the fluid particle cloud to represent the fluid. For the extraction step, the simulation velocities are interpolated on the mesh vertices (cf. Figure 6a). These vertices are located either above or below the fluid. A filtering step is required to keep only surface velocities. The filter is based on the vertical coordinates of the normal vectors at the mesh vertices. Vectors with negative vertical component belong to the bottom of the fluid mesh while those with positive vertical component belong to the flow surface. The vertices with a positive vertical component of the normal vector are retained (cf. Figure 6b).

The mesh vertex positions and velocities are expressed in the simulation coordinate system, that is, cell and cell/time-step. A conversion factor can be extracted from the Mantaflow library to convert velocities to physical units, that is, m/s. The positions can be converted from simulation cells to meters with the information of location, position and spatial sampling of the simulation domain. The positions of the surface vertices provide an information on the water level. In the proposed example, the water level in the area of interest varies between 0.407 and 0.447 m with a mean value of 0.428 m and a standard deviation of 0.0095 m.

As the mesh moves, the positions of the mesh vertices vary through time. So the mesh velocities have to be interpolated on a regular bi-dimensional grid defined on the horizontal XY plane (cf. Figure 6b). A nearest neighbor interpolation scheme is used because the spatial density of mesh velocities is high. For instance, in the proposed example the mesh resolution is 0.02 m so the spatial density of mesh vertices is about 2,500 points/m^2. With a high density of data to be interpolated, the nearest neighbor scheme limits interpolations errors. The camera information (i.e., calibration matrix, frame rate) are used to convert the reference data to the image coordinate space, for example, pixels or pix/frame.

The reference velocities are the simulation velocities used to move the particles. The particles are represented by a mesh which is built independently at each time-step. This means that the reference velocities are not directly the velocities that move the mesh from one frame to another. There may be a difference between the apparent velocities (i.e., mesh velocities) and the reference velocities (i.e., particle velocities) if the mesh does not represent correctly the particles movements. It is of paramount importance to correctly set the mesh generation parameters. Such issue is studied and discussed hereafter.

3. Case Studies

3.1. Overview of the Case Studies

Two case studies are proposed in this section. The first one is an evaluation of the reference velocities extracted. The second one is an example of evaluation of the LSPIV method in various measurement conditions. Both case studies are based on the scene and simulation presented previously as example. The fluid simulation was run on 4,500 frames with a frame rate of 30 fps which corresponds to a duration of 2.5 min. Such duration is used to ensure the establishment of steady-flow conditions in the area of interest. The parameters are listed in Table 2. The simulation took almost 3 days (72 hr) to compute on an Intel Core i9@3.7Gh with 12 cores. The time of execution is high as a river about 30 m-long by 8 m-wide is simulated with a resolution of 0.04 m. Only the last 200 frames are used for the velocity measurement which represents a total length of 6.66 s. This duration is adequate for the first case as the density of surface tracers is high. It is also representative of a typical duration used in field measurement so it is also relevant for the second case study. The simulation data of these 200 frames weight almost 73 GB and the mesh data weight almost 9 GB. Again, the high memory footprint is due to the highly resolved simulation. Figure 6c shows the layout of the simulation with the inflow upstream, the weir downstream and the study area where the averaged reference velocities are depicted. The scene parameters will be detailed for each case in the following sections. The images are rendered with Cycle, the physically-based rendering engine of Blender 2.90. The rendering time varies from 2 to 3 hr for the 200 frames, depending on the image size and the water aspect (e.g., light scattering within the water volume increases the rendering time). The renders are done on a GPU Nvidia Quadro RTX 4000.

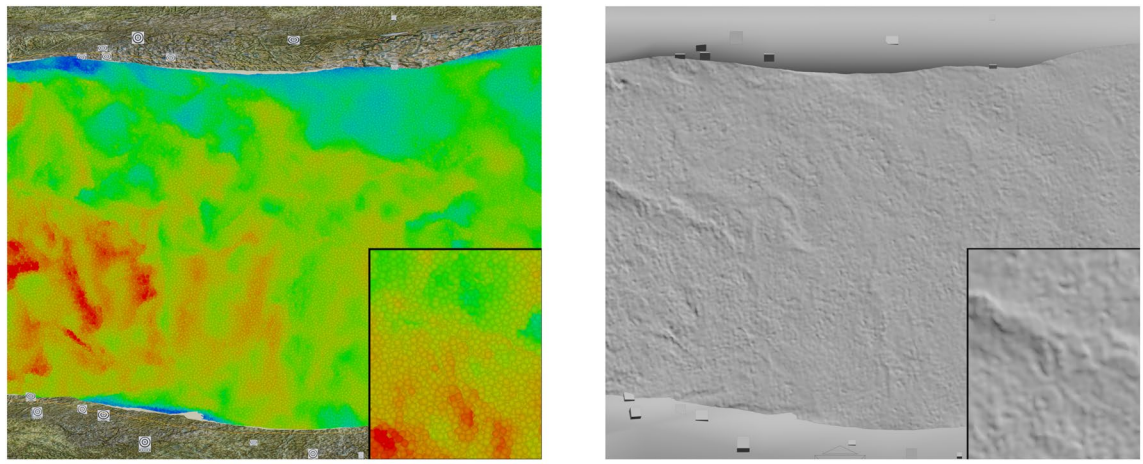
The Fudaa-LSPIV software (Le Coz et al., 2014) is used to analyze the image sequences with the LSPIV method. Several parameters have to be specified: the Interrogation Area (IA) that defines the size of the blocks for pattern matching, the Search Area (SA) that defines the scanned zone (i.e., possible displacements to be measured) and the velocity filters. These parameters are listed in Table 2 and will be detailed in the next sections.

Three metrics are used to compare the measured velocities with the reference. The Magnitude Error (ME) is the difference between the measured velocities and the reference velocities at each grid point i, with (u, v) the components of the measured velocity and (u_{ref}, v_{ref}) the components of the reference velocity.

$$ME_i = \sqrt{u_i^2 + v_i^2} - \sqrt{u_{ref_i}^2 + v_{ref_i}^2} \tag{1}$$

The Relative Magnitude Error (RME) is the same velocity ME expressed in percentage of the reference velocity:

$$RME_i = \frac{\sqrt{u_i^2 + v_i^2} - \sqrt{u_{ref_i}^2 + v_{ref_i}^2}}{\sqrt{u_{ref_i}^2 + v_{ref_i}^2}} \tag{2}$$



(a) (b)

Figure 7. Illustration of the two image sets used for the first case study: (a) render of the fluid particles and (b) of the fluid mesh with an orthographic camera.

The Endpoint Error (EE) (Otte & Nagel, 1994) is the distance in pixels between the endpoint of the measured velocity vector and the endpoint of the reference velocity vector:

$$EE_i = \sqrt{\left(u_i - u_{ref_i}\right)^2 + \left(v_i - v_{ref_i}\right)^2} \tag{3}$$

The analysis is carried out on the raw velocities, the filtered velocities and the velocities averaged over the all sequence. The RME are reported through box-plots. The box extends from the first quartile (Q1) to the third quartile (Q3) with a red line showing the median. The whiskers extend from the box by 1.5 times the inter-quartile range. The cumulative distribution of the ME is represented. The amount of EE above specific values are reported following the idea of Baker et al. (2011). The following classes are used: 0 pix, 0.25 pix, 0.5 pix, 1 pix and 2 pix.

3.2. Case 1: The Evaluation of the Reference Velocities Extracted

The objective of this case study is to evaluate the errors of the reference velocities quantitatively as they are extracted with a specific framework (cf. Section 2.5). The reference velocities are obtained from the simulation velocities that move the water particles between consecutive frames. The surface velocities are extracted and interpolated on a regular grid. So the reference velocities should correspond to particle movements at the surface. Interpolation errors could occur and generate deviations between the reference velocities and the real particle velocities. The particle movements at the surface are measured with LSPIV and compared with the reference velocities. With this verification procedure, both reference velocity errors and LSPIV measurement errors are contained in the observed deviations. Therefore, we design video generation settings to minimize LSPIV errors. The water particles are represented by spheres with a color scale corresponding to their velocity and illuminated by diffuse lighting (cf. Figure 7a). The rendered images show matte and highly textured surfaces which reduces optical flow errors. A high image size (i.e., 3,160 × 2,160) is used to obtain a good resolution of the patterns. It also ensures displacements of more than 10 pixels between consecutive frames which reduces the impact of the LSPIV sub-pixel resolution errors. An orthographic camera placed at nadir is used for the renders. This guarantees that no perspective distortion and calibration errors will impact the LSPIV measurement.

The LSPIV parameters are listed in Table 2. The IA size is 60 pixels which corresponds to a metric size of 0.20 m. Such size ensures that the IA contains significant patterns. The video framerate of 30 fps is used for the LSPIV analyses because the displacements are sufficiently high between consecutive frames, that is, larger than 10 pixels. It also ensures low deformations of the patterns to optimize the LSPIV performances. The SA is set based on a preliminary manual analysis of the surface displacements between consecutive frames. A grid of 30 × 25, that is, 750 points, is used. The raw measurements have a mean correlation of 0.91 with standard deviation 0.053. A minimum correlation filter of 0.85 is set to remove a few spurious vectors. The remaining velocity vectors are

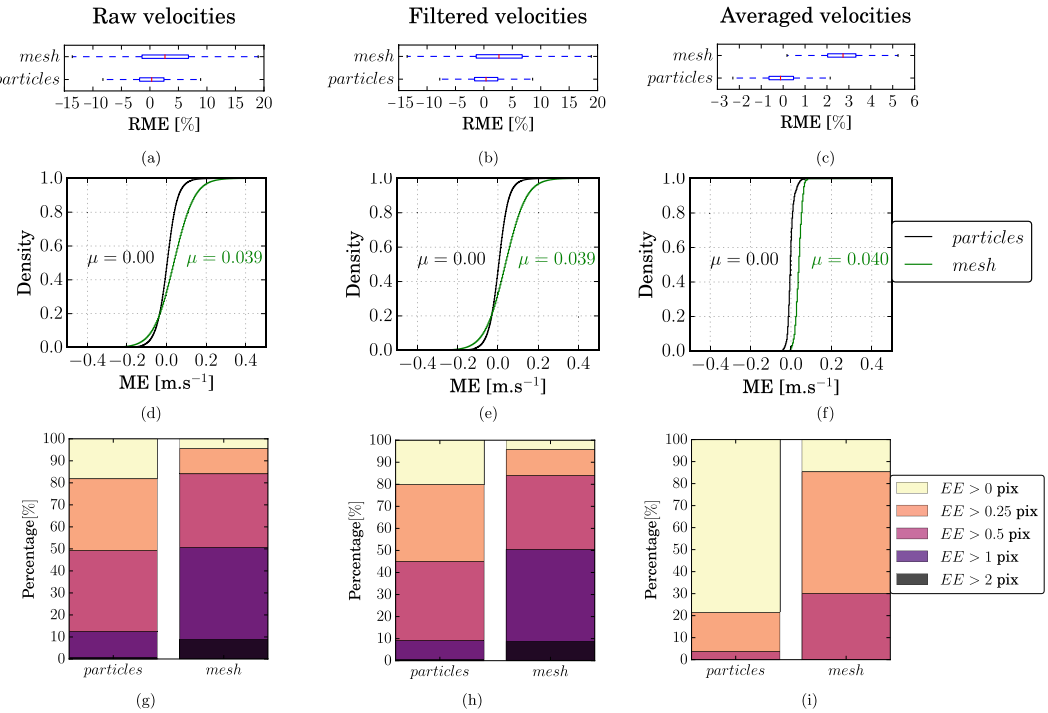


Figure 8. Comparison between LSPIV measurements and reference velocities for two image sequences: "particles" and "mesh" (cf. Figure 7). The LSPIV analyses are carried out on 750 points and 200 frames. The deviations between the measured and the reference velocities are reported with (a–c) RME boxplots, (d–f) ME cumulative distributions and (g–i) EE distribution for (a, d, g) the raw velocities, (b, e ,h) the filtered velocities and (c, f, i) the time-averaged velocities.

averaged over the entire sequence (200 frames) to obtain the time-averaged surface velocities. The deviations between the LSPIV measurement and the reference velocities are presented in Figure 8 as the *particles* results.

For the raw velocities, the RME first and third quartiles are −1.67% and 2.44% with a median at 0.33% (cf. Figure 8a). Almost 90% of the EE are below the pixel size and 50% below 0.5 pixel (cf. Figure 8g). Only a few vectors are erroneous thanks to the good measurement conditions. For this reason, the filtering step has almost no effect on the measurement errors (cf. Figures 8a–8g and Figures 8b–8h). The errors are reduced with averaging which confirms that errors at each time-step are statistically independent. For the averaged velocities, 79% of the EE are below 0.25 pixel (cf. Figure 8i). The RME first and third quartiles are −0.64% and 0.50% with a median at −0.1% (cf. Figure 8c). The errors on the averaged velocities are mainly below 1%, or 0.25 pixel, and non-biased (cf. Figure 8f). The obtained results show that the reference velocities are in good agreement with the particle velocities, within the LSPIV uncertainty. This means that interpolation errors of the reference velocities can be neglected if larger errors than sub-pixel errors are to be studied.

On the synthetic images the water particle cloud is represented by a mesh. The mesh is built around the particle cloud according to the defined mesh generation parameters. The surface appearance can vary depending on the parameters used, as shown in Figures 3e–3g with the impact of the mesh particle radius. This may affect the apparent displacements of the surface patterns between consecutive frames. In this second part, the apparent surface velocities are measured with LSPIV and compared to the reference velocities to verify their agreement. This analysis is the same as the previous one, but with a mesh generated around the fluid particles. The fluid simulation, the scene configuration (i.e., camera and lighting) and the LSPIV parameters are the same. The Blender "mesh-view" is used to obtain a matte and textured surface aspect (cf. Figure 7b). The raw measurements have a mean correlation of 0.939 with standard deviation 0.034. The same minimum correlation filter of 0.85 is set. The remaining velocity vectors are averaged over the entire sequence (200 frames) to obtain the time-averaged surface velocities. The deviations between the LSPIV measurement and the reference velocities are presented in Figure 8 as the *mesh* results.

For the raw velocities, the RME first and third quartiles are −1.17% and 6.85% with a median at 2.74% (cf. Figure 8a). As previously, the filter have almost no effect on the measurement errors and the errors are reduced with averaging. For the averaged velocities, the RME fist and third quartiles are 2.04% and 3.33% with a median at 2.8% (cf. Figure 8c). The slightly larger dispersion of the RME observed in the mesh sequence can be due to measurement errors as the surface aspect that is less textured than the one of the particle sequence. But the results of the mesh sequence also show a constant bias of about or 0.04 m/s in magnitude (Figure 8d–8f) which corresponds to 2.8% relatively to the reference velocities (cf. Figures 8a–8c). The bias corresponds to 0.36 pixel with the camera resolution used in this case study which explains the higher EE errors observed for the mesh case (cf. Figures 8g–8i).

The observed differences between the apparent surface velocities and the reference velocities are due to the mesh generation step. In fact, the surface aspect can vary depending on the mesh generation parameters, as shown in Figure 3. This can impact the surface apparent velocities. In this case study, the bias is about 0.36 pixel. This order of magnitude can be acceptable to investigate errors larger than the pixel size. The bias on the pixel displacements can be reduced with a lower ortho-image resolution for instance. To cope with these reference errors, comparative studies can be conducted. The errors observed in scenes with environmental perturbations are compared to those of an ideal scene without perturbation. With this procedure the bias on the reference velocities is canceled as it is contained in both perturbed scene and the ideal scene. The next case study propose an example of such procedure.

3.3. Case 2: A Typical Evaluation of Optical Flow Errors in LSPIV Measurements

In this case study, synthetic image sequences are used to investigate the impact of optical flow errors in LSPIV measurements. The LSPIV measurement errors of an ideal scene in terms of optical flow analysis are compared to errors of scenes with environmental perturbations. The ideal scene shows a Lambertian (i.e., matte) and textured water surface with homogeneous lighting. The scenes with perturbations represent typical field environments with various lighting conditions and water aspects. The image sequences presented in Figure 5 are used. The Blender "mesh-view" (cf. Figure 5a) is used as the ideal scene. The three typical on-site cases are named: "cloudy," a cloudy environment with brownish and diffuse water aspect (cf. Figure 5b), "sunny," a sunny environment with a large and smooth shadow on the water surface (cf. Figure 5c) and "forest," a forest environment with less light reaching the water surface and several direct sun reflections (cf. Figure 5d). For the "sunny" and "cloudy" image sets, some light scattering effect into the water volume is configured. This results in a noisy surface appearance.

The four image sets are based on the same fluid simulation and virtual camera with parameters listed in Table 2. The image size is representative of typical cases with 1,920 × 1,080 pixels, that is, Full HD size. The LSPIV parameters are the same for the four cases. The video frame rate is divided by two for the measurement, that is, 15 fps. This ensures a sufficiently high displacement in pixels between consecutive frames and thus reduces the impact of the LSPIV sub-pixel resolution errors. The GCP locations and their exact positions in the images are extracted from Blender. They are used to orthorectify the images. The ortho-images are built with a resolution of 0.0115 m/pixel. The velocities are computed from the sequence of 100 frames, on a grid of 19 × 23 points, that is, 437 points. The IA size is 32 pixels which corresponds to a metric size of about 0.4 m. This size allows to contain recognizable patterns within the IA. The SA parameters are listed in Table 2. They are defined based on a preliminary manual velocity estimation. In the ideal case, that is, "mesh" case, the raw measurements have a mean correlation of 0.8 with standard deviation 0.1. In the degraded cases the mean correlations of the raw measurement are lower with a mean value between 0.5 and 0.6 with standard deviation 0.15. A minimum correlation filter of 0.4 is applied to retain only significant correlations. In the degraded cases near zero velocities can occur in poorly textured areas. Several spurious vectors with high magnitudes are also observed in all the cases. A minimal magnitude filter of 0.2 m/s and a maximal magnitude filter of 2 m/s are set to remove them. The time-averaged velocity results are finally computed by averaging the results at each of the 437 grid points over the 100 frames. The deviations of the raw, filtered and averaged velocities are presented for each case in Figure 9.

For the raw and filtered velocities of the "mesh" case, the first, second and third quartiles of the RME distribution are about −5%, 1%, and 10%, respectively (cf. Figures 9a and 9b). The filters have almost no effect on the measurement errors as the measured correlations were globally high and no large magnitude errors occurred. After averaging, the first, second and third quartiles are reduced to −1.7%, 0.7%, and 2.8% respectively (cf. Figure 9c). These errors should be considered as baseline errors for the remainder of the study.

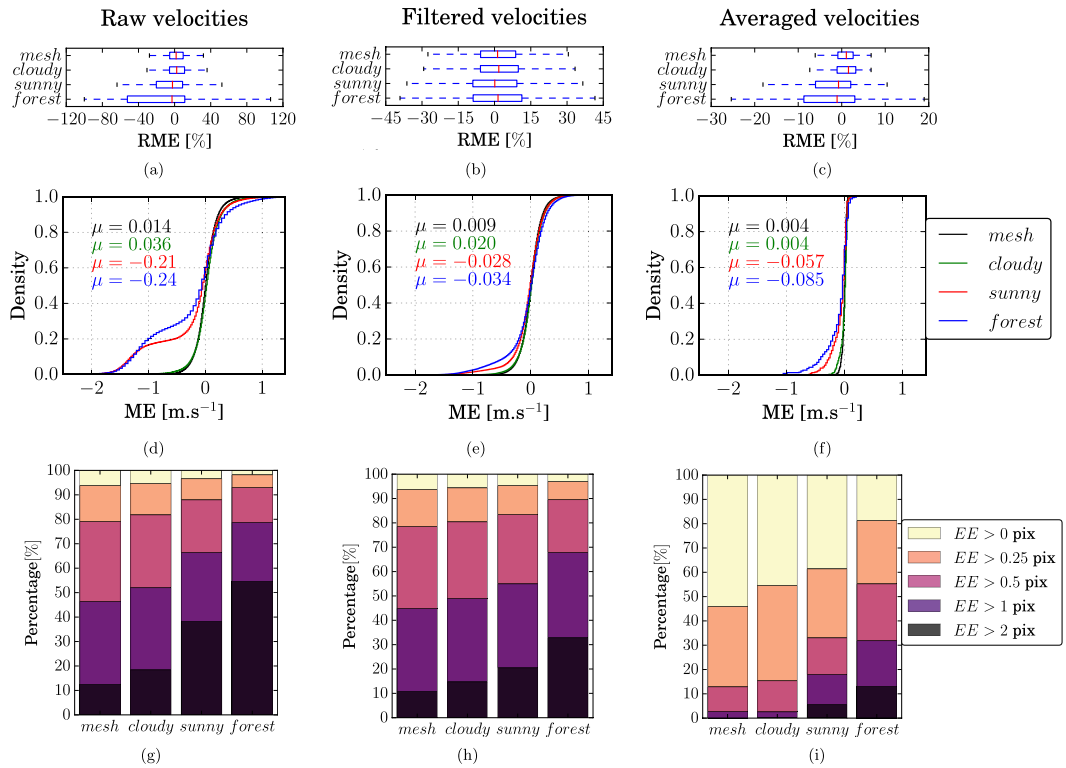


Figure 9. Comparison between LSPIV measurements and reference velocities for four image sets: "mesh," "cloudy," "sunny," "forest" (cf. Figure 5). The LSPIV analyses are carried out on 437 points and 100 frames. The deviations between the measured and the reference velocities are reported with (a–c) RME boxplots, (d,e,f) ME cumulative distributions and (g–i) EE distribution for (a, d, g) the raw velocities, (b, e, h) the filtered velocities and (c, f, i) the time-averaged velocities.

The "cloudy" sequence yields results similar to those of the "mesh" sequence. This could be explained by the good environmental conditions with almost homogeneous lighting and without specific artifacts on the water surface such as glares, noise, shadows or water transparency.

A tendency to underestimate the velocities is observed for the two other sequences "sunny" and "forest." The cumulative distributions of the ME on the raw velocities show that a non negligible part of the results can be strongly underestimated (cf. Figure 9d). 38% of the EE are above 2 pixels for the "sunny" case, and 56% for the "forest" case (cf. Figure 9g). These errors are due to the homogeneous (i.e., non-textured) and noisy surface appearance found in some areas. The filters help to remove several spurious vectors in these sequences and reduce the strong errors (cf. Figure 9b, 9e and 9h). After averaging almost a third of the measured velocities are still underestimated for the "sunny" and "forest" sequences, as shown by the ME cumulative distributions (cf. Figure 9f). The "forest" sequence shows the highest errors with RME going up to more than −10% and almost 30% of the EE larger than a pixel (i.e., 0.35 m/s).

The results presented show that the environmental conditions can impact the image-based velocimetry results, regardless of the water surface undulations. The objective of the case study is to show the possibilities offered by realistic synthetic images to study such optical flow errors. It also proposes typical evaluation criteria to support the comparison analysis. Further analysis could be made with for instance various sets of LSPIV parameters, filters or with a longer video duration.

4. Discussion

4.1. The Advantages of the Proposed Method

The impact of the lighting conditions, the scene environment, the water aspect and the camera parameters, location and orientation can strongly affect the image-based measurement. However, these optical flow and camera

calibration errors remain poorly treated due to the limits of the outdoor evaluation processes and the limits of the available synthetic data set listed in introduction. The main advantage of the method proposed in this paper is the ability to generate realistic river scene where various environmental parameters can be configured. Thanks to this realism, many measurement situations can be reproduced and studied.

The method relies on open-source and freely available software, Blender and Mantaflow. The large user community of Blender is relevant as it provides support, pertinent tutorials and a large variety of plugins. Blender software often evolves with new features for a better realism of the rendered images. Many environmental perturbations such as rain, fog, wind or electric sensor noise can be reproduced. A wide range of water aspects can also be obtained with more complex settings. With Blender, every instance of the scene can be animated. This means that the scene configuration, for example, lighting, water aspect or camera location and orientation can be varied through time. Python scripts can also be used either to extract the scene information such as GCP locations and camera calibration matrix for instance or to automatically launch renders and set parameters.

All these possibilities can be used to evaluate the various outdoor image-based velocimetry methods in given environmental conditions. Comparisons can be conducted with the generated datasets, either to evaluate the inter-user measurement variability or to challenge methods. The datasets can also be used to validate new tools such as filters, ortho-rectification procedures, or new velocimetry methods for instance. They could be interesting for educational purposes as various practical on-field situations can be investigated.

4.2. The Limits of the Proposed Method

The principal limit of the proposed method is the possible difference between the apparent mesh velocities and the reference velocities extracted from the simulation. A bias may be induced by the mesh generation step, depending on the mesh generation parameters. Thus the precision of the reference velocities is affected and the uncertainty cannot be known precisely. An estimation of the reference errors can be obtained with an image-based velocimetry analysis, as illustrated in this study. With this procedure the estimated reference errors are determined within the image-based velocimetry uncertainty. Thus the measurement errors should be minimized to have a more precise estimation of the reference errors. An orthographic camera placed at nadir can be used to remove ortho-rectification errors. The Blender "mesh-view" can be used to reduce optical flow errors as the surfaces of the rendered images are matte, textured and lit by diffuse lighting. The obtained velocity errors give an estimate of the maximum reference errors. In the example presented in this paper, this method suggests that the apparent velocities are biased by approximately 0.040 m/s at most, compared to the reference velocities. In the case of a typical image-based velocimetry analysis with frame rate 15 fps and ortho-rectified image resolution 0.0115 m/pix (cf. Section 3.3), this bias on the reference displacement from one frame to another is about 0.23 pixels. Such estimated reference errors appear to be sufficiently low to study more impacting effects (i.e., inducing errors larger than the pixel size) such as outdoor environmental perturbations and settings errors including operator choices.

The aspect of the fluid simulated in the example look very realistic with various waves and eddies on the surface. This confirms that Mantaflow is well suited for our application as the surface appearance was the principal objective of the simulations. However, Mantaflow requires a large set of parameters and cannot implement several boundary conditions traditionally used in hydraulic simulations, for example, stage, stage-discharge relation, lateral or vertical velocity profile, bed roughness. To obtain the desired hydrodynamics, a trial-and-error procedure has to be used to set up the simulation environment and parameters. The surface velocity distribution cannot be imposed directly. However, imposing an exact reference velocity field may not be necessary to evaluate image velocimetry methods as a looser experimental design may suffice to explore a range of flow conditions. And that does not increase the uncertainty of the reference velocities, still low enough for studying environmental perturbations and calibration errors.

With Mantaflow it is not possible to add floating solid objects to the fluid simulation. Only diffuse particles can be generated. In the examples we do not consider the diffuse particles generator as it is very costly in terms of computational time and memory footprint. The overall fluid simulation was already long to compute with 3.5 days on a single computer. The memory footprint was also important with almost 81 GB for 200 frames at 30 fps (i.e., 6.66 s). These computational needs are due to the fine resolution used: 0.04 cm/cell over a fluid simulation domain 30 m-long, 8 m-wide and 3 m-deep. However, such fine simulation resolution is mandatory to obtain

realistic results and to reduce the bias between the apparent velocities and the reference velocities. In previous simulations with lowest resolutions, we observed a more viscous water appearance and larger reference errors. Even if the simulation time is long, many scenarios with different rendering options can be generated from the same simulation, as shown in the second case study. The rendering process is also time consuming with almost 2–3 hr for 200 frames rendered with Full HD resolution (i.e., 1,920 × 1,080) on a high-performance graphics card. However, these computational times remain fully manageable for the establishment of long-term reference videos that may be used for many applications afterward.

Synthetic imaging is a powerful tool to evaluate image-based methods but it represents an idealized version of in situ cases. As pointed out by Barron et al. (1994) "*the measure of performance should be taken as an optimistic bound on the expected errors with real image sequences*." Several instances of image formation are difficult to reproduce: sensor saturation, pixel asymmetry, optical aberrations, etc. Complex optical behavior of materials (e.g., water) can also be difficult to configure. As shown in this paper, the progress in computer graphics makes it possible to reproduce physical phenomena with increasing precision. Still, the synthetic images will never reach the complexity of natural scenes. These limits should be kept in mind while drawing conclusions from synthetic images.

4.3. Improvements of the Method

The proposed framework is not fully packaged yet and it still requires some manual operations in sequence. The fluid simulation script is generated and executed automatically with Blender. This script has to be modified and executed again with the Mantaflow API, out of Blender, to extract the reference velocities from the simulation. First, the simulation script is exported from Blender with a specific debug code. Then the instructions for data extraction in text files are added to the script. We modified the Mantaflow source code to add the function for data extraction in text files as it was not proposed initially. Other formats than text files could be considered to optimize the memory footprint of the reference data. This framework for reference extraction could be improved to make it simpler. Ideally, it could be formalized in a Blender plugin to make it easier to use.

Only a subset of Mantaflow features are exposed to Blender. Several unexposed tools could be interesting to control the surface aspect such as surface turbulence generation, cf. (Mercier et al., 2015), or K-epsilon turbulence modeling. It would be interesting to investigate how to use them. We found out that guiding objects can be used in Blender to impose velocities in a specific area of the fluid simulation. We do not used this feature but relay it here as a possible improvement to have more control on the surface velocities. Floating solid objects could be added to the Blender scene. It is not possible to move them directly with the Mantaflow fluid simulation, but their positions could be varied through time in post-processing. The Mantaflow velocities could be extracted first and then used to move these floating objects with a Python script executed in Blender. The bed roughness cannot be directly configured in the fluid simulation but it is possible to add rocks on the riverbed and set them as wall boundaries to simulate a rough bed. In the example, a rough bed was obtained near the inflow by playing on the riverbed geometry (cf. Figure 2). The vertical velocity profile can be retrieved from the simulation velocities computed by Mantaflow. This could be useful to compute the reference discharge. The vertical velocity profile was not considered in the scope of this article as only the surface velocities were to be studied.

The main advantages of the proposed method are provided by Blender with the scene construction (i.e., scene layout, lighting, water aspect) and the rendering capabilities (i.e., realism). The main limits are due to Mantaflow with the fluid simulation and the reference velocity extraction. With the proposed framework, it would be possible to change the fluid simulation engine as long as the new simulation engine is able to export a mesh of the fluid. This mesh could be imported in Blender and incorporated in a scene before the scene configuration and rendering steps. For instance, the NEPTUNE_CFD code (Archambeau et al., 2004) offers such possibility. However, high-resolution fluid simulation will always be computationally expensive, regardless of the fluid simulation engine.

To reduce the computational time, the fluid simulation and rendering step can be done on a computer cluster. The latest version of Blender embeds a new physically-based rendering engine: Cycle X. This rendering engine has better performance and may help to reduce the rendering time.

5. Conclusion and Perspectives

The methodology presented in this paper allows the creation of synthetic videos of realistic river flow scenes. It uses open-source computer graphics tools, namely Blender and Mantaflow, to build up the scene, simulate the river flow and generate the synthetic images through a virtual camera. For given channel geometry and fluid simulation, the scene and the flow surface aspect can be varied by playing with the camera parameters and the environmental conditions, such as lighting, environment, water aspect, etc. This is useful for evaluating image-based velocimetry methods over a range of imaging conditions. Also, varying the flow conditions through other fluid simulations with different channel geometry and different hydraulic boundary conditions would be useful for such benchmarking.

A limit of the method is the precision of the reference velocities. A bias may appear between the apparent fluid velocities and the reference velocities due to the fluid simulation procedure and the reference extraction framework. To limit this bias, a fine simulation resolution should be used and a specific attention has to be paid on the mesh generation parameters of the fluid. An evaluation procedure has been proposed to estimate the reference errors, based on an image-based measurement. The obtained velocity errors give an estimate of the maximum reference errors. No accurate evaluation of the bias can be guaranteed. In the case studies, a bias of several percents was estimated which corresponds to displacement errors below half pixel. These errors could be acceptable as larger errors are to be investigated. The recommended way to deal with the reference errors is to conduct comparative studies between an ideal case and perturbed cases, as shown in the second case study.

This paper has shown that computer graphics tools are powerful tools that can be used to generate realistic image sequences useful for quantifying and investigating the errors of outdoor image-based velocimetry methods such as optical flow errors (e.g., water aspect, shadows, glares, non uniform lighting) and ortho-rectification errors (e.g., oblique point of views, camera shaking, spatial distribution of GCPs). A list of synthetic videos in various conditions is attached to this article (cf. Section 7). The generated synthetic sequences complement the currently available datasets by enabling to study the impact of optical flow errors and ortho-rectification errors on the measurement. The validation of new methods or guidelines and various evaluation procedures could be conducted with these sequences (e.g., inter-comparisons, challenges or sensitivity analysis). Such evaluations would help to better quantify the strengths and limitations of image-based velocimetry methods toward their operational deployment.

Data Availability Statement

A data set is available at Zenodo repository via https://doi.org/10.5281/zenodo.6257391 with open access (Bodart et al., 2022). It contains the data used in the two case studies, the modified version of Mantaflow source code with the Python library used for the manipulation of the reference data and a data set with various synthetic videos generated with the fluid simulation presented in the article. The data set comes with reference velocities already interpolated on a regular grid.

Acknowledgments
The authors wish to thank the three anonymous reviewers and the handling editor for their detailed comments, which greatly improved the quality of this manuscript. This work and the PhD grant of the first author were funded by the French National Association of Research and Technology (ANRT) and EDF R&D with the Industrial Conventions for Training through Research (CIFRE grant agreement 2019/0277).

References

Abram, G. D., & Whitted, T. (1990). Building block shaders. *ACM SIGGRAPH Computer Graphics*, *24*(4), 283–288. https://doi.org/10.1145/97880.97910

Archambeau, F., Méchitoua, N., & Sakiz, M. (2004). Code Saturne: A finite volume code for the computation of turbulent incompressible flows—Industrial applications. *International Journal on Finite Volumes*, *1*(1). http://www.latp.univ-mrs.fr/IJFV/spip.php?article3

Baker, S., Scharstein, D., Lewis, J. P., Roth, S., Black, M. J., & Szeliski, R. (2011). A database and evaluation methodology for optical flow. *International Journal of Computer Vision*, *92*(1), 1–31. https://doi.org/10.1007/s11263-010-0390-2

Barron, J. L., Fleet, D. J., & Beauchemin, S. S. (1994). Performance of optical flow techniques. *International Journal of Computer Vision*, *12*(1), 43–77. https://doi.org/10.1007/BF01420984

Blender Community, B. (2020). Blender—A 3D modelling and rendering package. Retrieved from www.blender.org

Bodart, G., Coz, J. L., Jodeau, M., & Hauet, A. (2022). *Synthetic river flow videos dataset*. Zenodo. https://doi.org/10.5281/zenodo.6257391

Brackbill, J. U., & Ruppel, H. M. (1986). FLIP: A method for adaptively zoned, particle-in-cell calculations of fluid flows in two dimensions. *Journal of Computational Physics*, *65*(2), 314–343. https://doi.org/10.1016/0021-9991(86)90211-1

Brevis, W., Niño, Y., & Jirka, G. H. (2011). Integrating cross-correlation and relaxation algorithms for particle tracking velocimetry. *Experiments in Fluids*, *50*(1), 135–147. https://doi.org/10.1007/s00348-010-0907-z

Cook, R. L. (1984). Shade trees. *ACM SIGGRAPH Computer Graphics*, *18*(3), 223–231. https://doi.org/10.1145/964965.808602

Cook, R. L., Porter, T., & Carpenter, L. (1984). Distributed ray tracing. In *Proceedings of the 11th annual conference on Computer graphics and interactive techniques* (pp. 137–145). Association for Computing Machinery. https://doi.org/10.1145/800031.808590

Detert, M. (2021). How to avoid and correct biased riverine surface image velocimetry. *Water Resources Research*, *57*(2), e2020WR027833. https://doi.org/10.1029/2020WR027833

Detert, M., Johnson, E. D., & Weitbrecht, V. (2017). Proof-of-concept for low-cost and non-contact synoptic airborne river flow measurements. *International Journal of Remote Sensing*, *38*(8–10), 2780–2807. https://doi.org/10.1080/01431161.2017.1294782

Dolcetti, G., Hortobágyi, B., Perks, M., & Tait, S. (2020). The effect of surface gravity waves on the measurement of river surface velocity. In *River Flow 2020: 10th conference on fluvial hydraulics*. CRC Press (Taylor & Francis). Retrieved from https://eprints.whiterose.ac.uk/165812/

Eltner, A., Sardemann, H., & Grundmann, J. (2020). Technical Note: Flow velocity and discharge measurement in rivers using terrestrial and unmanned-aerial-vehicle imagery. *Hydrology and Earth System Sciences*, *24*(3), 1429–1445. https://doi.org/10.5194/hess-24-1429-2020

Fujita, I., & Kunita, Y. (2011). Application of aerial LSPIV to the 2002 flood of the Yodo River using a helicopter mounted high density video camera. *Journal of Hydro-environment Research*, *5*(4), 323–331. https://doi.org/10.1016/j.jher.2011.05.003

Fujita, I., Muste, M., & Kruger, A. (1998). Large-scale particle image velocimetry for flow analysis in hydraulic engineering applications. *Journal of Hydraulic Research*, *36*(3), 397–414. https://doi.org/10.1080/00221689809498626

Fujita, I., Watanabe, H., & Tsubaki, R. (2007). Development of a non-intrusive and efficient flow monitoring technique: The space-time image velocimetry (STIV). *International Journal of River Basin Management*, *5*(2), 105–114. https://doi.org/10.1080/15715124.2007.9635310

Hauet, A., Creutin, J.-D., & Belleudy, P. (2008). Sensitivity study of large-scale particle image velocimetry measurement of river discharge using numerical simulation. *Journal of Hydrology*, *349*(1), 178–190. https://doi.org/10.1016/j.jhydrol.2007.10.062

Hauet, A., Kruger, A., Krajewski, W., Bradley, A., Muste, M., Creutin, J.-D., & Wilson, M. (2008). Experimental system for real-time discharge estimation using an image-based method. *Journal of Hydrologic Engineering*, *13*(2), 105–110. https://doi.org/10.1061/(ASCE)1084-0699(2008)13:2(105)

Horn, B. (1986). Robot vision.

Ihmsen, M., Akinci, N., Akinci, G., & Teschner, M. (2012). Unified spray, foam and air bubbles for particle-based fluids. *The Visual Computer*, *28*(6–8), 669–677. https://doi.org/10.1007/s00371-012-0697-9

Jodeau, M., Hauet, A., Paquier, A., Coz, J., & Dramais, G. (2008). Application and evaluation of LS-PIV technique for the monitoring of river surface velocities in high flow conditions. *Flow Measurement and Instrumentation*, *19*(2), 117–127. https://doi.org/10.1016/j.flowmeasinst.2007.11.004

Kähler, C. J., Astarita, T., Vlachos, P. P., Sakakibara, J., Hain, R., Discetti, S., et al. (2016). Main results of the 4th international PIV challenge. *Experiments in Fluids*, *57*(6), 97. https://doi.org/10.1007/s00348-016-2173-1

Keane, R. D., & Adrian, R. J. (1992). Theory of cross-correlation analysis of PIV images. *Applied Scientific Research*, *49*(3), 191–215. https://doi.org/10.1007/BF00384623

Khalid, M., Pénard, L., & Mémin, E. (2019). Optical flow for image-based river velocity estimation. *Flow Measurement and Instrumentation*, *65*, 110–121. https://doi.org/10.1016/j.flowmeasinst.2018.11.009

Lecordier, B., & Westerweel, J. (2004). *The EUROPIV synthetic image generator (SIG)*. Springer. Retrieved from https://hal.archives-ouvertes.fr/hal-00638768

Le Coz, J., Jodeau, M., Hauet, A., Marchand, B., & Le Boursicaud, R. (2014). Image-based velocity and discharge measurements in field and laboratory river engineering studies using the free Fudaa-LSPIV software. In *River flow 2014* (pp. 1961–1967). https://doi.org/10.1201/b17133-262

Le Coz, J., Renard, B., Vansuyt, V., Jodeau, M., & Hauet, A. (2021). Estimating the uncertainty of video-based flow velocity and discharge measurements due to the conversion of field to image coordinates. *Hydrological Processes*, *35*(5), e14169. https://doi.org/10.1002/hyp.14169

Legout, C., Darboux, F., Nédélec, Y., Hauet, A., Esteves, M., Renaux, B., et al. (2012). High spatial resolution mapping of surface velocities and depths for shallow overland flow. *Earth Surface Processes and Landforms*, *37*(9), 984–993. https://doi.org/10.1002/esp.3220

Leitão, J. P., Peña-Haro, S., Lüthi, B., Scheidegger, A., & Moy de Vitry, M. (2018). Urban overland runoff velocity measurement with consumer-grade surveillance cameras and surface structure image velocimetry. *Journal of Hydrology*, *565*, 791–804. https://doi.org/10.1016/j.jhydrol.2018.09.001

Lentine, M., Cong, M., Patkar, S., & Fedkiw, R. (2012). Simulating free surface flow with very large time steps. In *Proceedings of the ACM SIGGRAPH/eurographics symposium on computer animation* (pp. 107–116). Eurographics Association.

Lin, D., Grundmann, J., & Eltner, A. (2019). Evaluating image tracking approaches for surface velocimetry with thermal tracers. *Water Resources Research*, *55*(4), 3122–3136. https://doi.org/10.1029/2018WR024507

Markosian, L., Kowalski, M. A., Goldstein, D., Trychin, S. J., Hughes, J. F., & Bourdev, L. D. (1997). Real-time nonphotorealistic rendering. In *Proceedings of the 24th annual conference on Computer graphics and interactive techniques* (pp. 415–420). ACM Press/Addison-Wesley Publishing Co. https://doi.org/10.1145/258734.258894

McMillan, H., Krueger, T., & Freer, J. (2012). Benchmarking observational uncertainties for hydrology: Rainfall, river discharge and water quality. *Hydrological Processes*, *26*(26), 4078–4111. https://doi.org/10.1002/hyp.9384

Mendes, L., Bernardino, A., & Ferreira, R. M. L. (2020). Piv-image-generator: An image generating software package for planar PIV and Optical Flow benchmarking. *SoftwareX*, *12*, 100537. https://doi.org/10.1016/j.softx.2020.100537

Mercier, O., Beauchemin, C., Thuerey, N., Kim, T., & Nowrouzezahrai, D. (2015). Surface turbulence for particle-based liquid simulations. *ACM Transactions on Graphics*, *34*(6), 1–10. https://doi.org/10.1145/2816795.2818115

Mueller, D. S. (2013). extrap: Software to assist the selection of extrapolation methods for moving-boat ADCP streamflow measurements. *Computers & Geosciences*, *54*, 211–218. https://doi.org/10.1016/j.cageo.2013.02.001

Muste, M., Fujita, I., & Hauet, A. (2008). Large-scale particle image velocimetry for measurements in riverine environments. *Water Resources Research*, *44*(4). https://doi.org/10.1029/2008WR006950

Otte, M., & Nagel, H. H. (1994). Optical flow estimation: Advances and comparisons. In J.-O. Eklundh (Ed.), *Computer vision—ECCV '94* (pp. 49–60). Springer. https://doi.org/10.1007/3-540-57956-7_5

Pearce, S., Ljubičić, R., Peña-Haro, S., Perks, M., Tauro, F., Pizarro, A., et al. (2020). An evaluation of image velocimetry techniques under low flow conditions and high seeding densities using unmanned aerial systems. *Remote Sensing*, *12*(2), 232. https://doi.org/10.3390/rs12020232

Peña-Haro, S., Carrel, M., Lüthi, B., Hansen, I., & Lukes, R. (2021). Robust image-based streamflow measurements for real-time continuous monitoring. *Frontiers in Water*, *3*. https://doi.org/10.3389/frwa.2021.766918

Perks, M. T. (2020). KLT-IV v1.0: Image velocimetry software for use with fixed and mobile platforms. *Geoscientific Model Development*, *13*(12), 6111–6130. https://doi.org/10.5194/gmd-13-6111-2020

Perks, M. T., Dal Sasso, S. F., Hauet, A., Jamieson, E., Le Coz, J., Pearce, S., et al. (2020). Towards harmonisation of image velocimetry techniques for river surface velocity observations. *Earth System Science Data*, *12*(3), 1545–1559. https://doi.org/10.5194/essd-12-1545-2020

Perks, M. T., Russell, A. J., & Large, A. R. G. (2016). Technical Note: Advances in flash flood monitoring using unmanned aerial vehicles (UAVs). *Hydrology and Earth System Sciences*, *20*(10), 4005–4015. https://doi.org/10.5194/hess-20-4005-2016

Pizarro, A., Dal Sasso, S. F., Perks, M. T., & Manfreda, S. (2020). Identifying the optimal spatial distribution of tracers for optical sensing of stream surface flow. *Hydrology and Earth System Sciences*, *24*(11), 5173–5185. https://doi.org/10.5194/hess-24-5173-2020

Pumo, D., Alongi, F., Ciraolo, G., & Noto, L. V. (2021). Optical methods for river monitoring: A simulation-based approach to explore optimal experimental setup for LSPIV. *Water*, *13*(3), 247. https://doi.org/10.3390/w13030247

Rossi, M. (2019). Synthetic image generator for defocusing and astigmatic PIV/PTV. *Measurement Science and Technology*, *31*(1), 017003. https://doi.org/10.1088/1361-6501/ab42bb

Stanislas, M., Okamoto, K., & Kähler, C. (2003). Main results of the first international PIV challenge. *Measurement Science and Technology*, *14*(10), R63–R89. https://doi.org/10.1088/0957-0233/14/10/201

Stanislas, M., Okamoto, K., Kähler, C. J., & Westerweel, J. (2005). Main results of the second international PIV challenge. *Experiments in Fluids*, *39*(2), 170–191. https://doi.org/10.1007/s00348-005-0951-2

Stanislas, M., Okamoto, K., Kähler, C. J., Westerweel, J., & Scarano, F. (2008). Main results of the third international PIV Challenge. *Experiments in Fluids*, *45*(1), 27–71. https://doi.org/10.1007/s00348-008-0462-z

Thuerey, N., & Pfaff, T. (2018). MantaFlow. Retrieved from http://mantaflow.com

Tsuji, I., Tani, K., Fujita, I., & Notoya, Y. (2018). Development of aerial space time volume velocimetry for measuring surface velocity vector distribution from UAV. In *E3S web of conferences* (Vol. *40*). https://doi.org/10.1051/e3sconf/20184006011

Verri, A., & Poggio, T. (1989). Motion field and optical flow: Qualitative properties. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *11*(5), 490–498. https://doi.org/10.1109/34.24781

Whitted, T. (1980). An improved illumination model for shaded display. *Communications of the ACM*, *23*(6), 343–349. https://doi.org/10.1145/358876.358882

Willert, C. (1996). The fully digital evaluation of photographic PIV recordings. *Applied Scientific Research*, *56*(2), 79–102. https://doi.org/10.1007/BF02249375

Zhang, Z. (2000). A flexible new technique for camera calibration. In *IEEE transactions on pattern analysis and machine intelligence* (Vol. *22*, pp. 1330–1334). https://doi.org/10.1109/34.888718