POLITECNICO
MILANO 1863

SCUOLA DI INGEGNERIA INDUSTRIALE
E DELL'INFORMAZIONE

Enhanced BOS optimization and experimentation on an hydrogen flame

Tesi di Laurea Magistrale in
Aeronautic Engineering - Ingegneria Aeronautica

Author: **Mauro Rossi**

Student ID: 944770
Advisor: Prof. Fabio Cozzi
Academic Year: 2020-21

Abstract

The Background Oriented Schlieren (BOS) is an optical technique able to visualize and measure the optical gradients in a transparent media.
Its main principle is the same as the more consolidated Schlieren technique: the density gradient variation of a fluid causes a variation in its refractive index, and thus a deflection of the light rays that pass through it. By measuring the deflection of the light rays on a background (usually white particles over a black background) it is possible to measure the density gradient of the fluid.
For the BOS, a camera acquires two images of the background, one of which is distorted by the presence of the schlieren object; by comparing these two images it is possible to measure the deflection of the light rays and thus the density gradient of the fluid. The Enhanced BOS (EBOS) uses n backgrounds with slight variations from the original to compare to the single image of the distorted background; the n analyses are then averaged to obtain a more accurate result than the one obtained by the BOS.
For this work, a monitor-displayed background is used; this allows to define a tool for the measure of the quality of a BOS and EBOS analysis, based on the creation of synthetic displacement fields.

The first part of this work is dedicated to the optimization of some parameters of the background, by exploiting the synthetic displacements: the optimal shape, dimension and density of the background particles are defined.
Then, two different software for the image analysis are compared, and the pros and cons of each one are determined, up to the advice of the most suited one for the BOS.
Different methods for the definition of the EBOS n backgrounds are proposed, and it is explained which techniques generates the best results.
Eventually, BOS and EBOS techniques are applied over an experimental case, represented by and hydrogen flame. It is shown how both BOS and EBOS are accurate techniques for the flame visualization, and how the EBOS, optimized as previously demonstrated, is a more accurate and reliable measure instrument than the BOS.

Keywords: optical techniques, BOS, EBOS, optimization, hydrogen flame

Sommario

La Background Oriented Schlieren (BOS) è una tecnica ottica che permette di visualizzare e misurare gradienti di densità in fluidi trasparenti.
Il principio su cui si basa è lo stesso della più consolidata tecnica Schlieren: una variazione del gradiente di densità in un fluido comporta una variazione nell'indice di rifrazione, e di conseguenza la deflessione dei raggi luminosi che lo attraversano. Misurando la deflessione dei raggi su di uno sfondo (generalmente composto da puntini bianchi su uno sfondo nero) è possibile risalire al gradiente di densità del fluido.
Nella BOS, una fotocamera acquisisce due immagini dello sfondo, una indeformata e una deformata dall'oggetto schlieren, confrontando queste due immagini si può misurare la deflessione dei raggi luminosi e quindi il gradiente di densità. La Enhanced BOS (EBOS) utilizza invece n sfondi leggermente diversi da confrontare con la singola immagine dello sfondo deformato, per poi mediare le analisi ed ottenere un risultato più preciso rispetto alla BOS.

In questo lavoro, lo sfondo è visualizzato su un monitor; questo permette la creazione di uno strumento di misura della qualità di una analisi BOS o EBOS che si basa sulla creazione di campi di spostamento sintetici.
La prima parte di questo lavoro di tesi si occupa dell'ottimizzazione di alcuni parametri di creazione dello sfondo, sfruttando gli spostamenti sintetici: sono definite la forma, la dimensione e la densità ottimale delle particelle che compongono lo sfondo.
Dopodiché si confrontano due diversi programmi di analisi delle immagini, stabilendo i vantaggi e gli svantaggi di ciascuno e suggerendo il più adatto per le analisi BOS.
Si propongono poi diversi metodi per la realizzazione degli n sfondi della EBOS, e si spiega quale tecnica generi i risultati più accurati.
Infine, si applicano la BOS e la EBOS ad un caso reale, composto da una fiamma di idrogeno. Si mostra come sia la BOS che la EBOS siano strumenti molto precisi nella rappresentazione della fiamma, e di come la EBOS, ottimizzata come dimostrato prima, rappresenti uno strumento di misura più preciso e affidabile rispetto alla BOS.

Parole chiave: tecniche ottiche, BOS, EBOS, ottimizzazione, fiamma di idrogeno

Contents

Introduction

The Background Oriented Schlieren (BOS) is an optical technique that allows to quantitatively measure density gradients in transparent flows.
The BOS belongs to the Schlieren techniques family, and it is based on the deflection encountered by the light rays as they pass through a medium with non-constant density. The light deflection can be appreciated by the use of a background, where a pattern (usually white dots on a black background) is present.
An image of the background is compared to a different image of the same background, acquired while the phenomenon to observe is present between a camera and the background: by comparing the two images, it is possible to measure the deflection of the light rays, and thus, thanks to the Gladston-Dale equation 1.2, to measure the density gradient field.

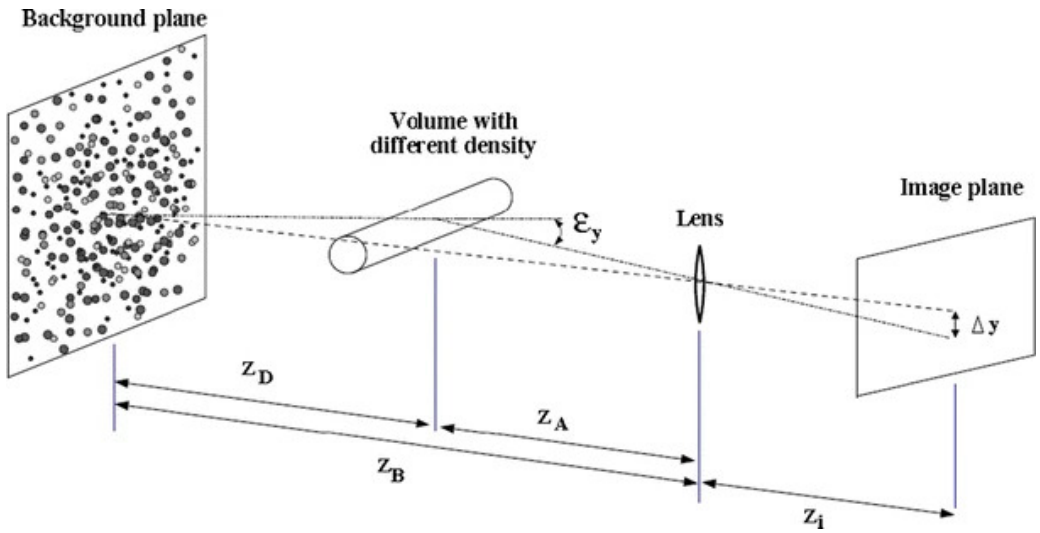


Figure 1: Typical BOS setup, [21].

As shown in figure 1, the setup is usually composed by a camera and a background, which are located at the two different sides of the phenomenon to measure.

Motivations and Objectives

The simplicity of the setup used and the non invasiveness of this technique, make the BOS an optical technique that could be easily used in many fields, from the aerodynamics to the energy production. Moreover, the capability of the BOS to produce non only qualitative but also quantitative measures of the density gradients allow this optical technique to be used for the state-of-the-art of the scientific research. The combination of high precision, low cost, and simplicity of use, make the BOS to be potentially a reliable, wide-spread optical technique.
However, the BOS is a recent technique which is not consolidated yet, and a wide number of studies are made every year to propose improvements in both the simplicity or the accuracy of the BOS.
One recent improvement in this technique is the use of a monitor-displayed background, which allows the fast generation of different backgrounds. A different recent improvement is the Enhanced BOS or EBOS [4], which represents an improvement in the accuracy of the measure, obtained by the use of multiple backgrounds as reference images. This work combines the advantages of these two recent improvements in the BOS by using a monitor-displayed background to easily display the multiple backgrounds needed for the EBOS.

Thanks to the use of the monitor, it is also possible to define a practical method to measure the accuracy of the technique, as explained in chapter 3, which is based on simulated density gradients between the camera and the monitor, for which the displacement field is known.
The simplicity and complete customization of the background generation granted by the monitor, together with the use of a reliable instrument to measure the quality of an analysis, allow to study the EBOS performances to the variation of those parameters.

The main objective to this study will thus be to choose some parameters for the background generation, and to optimize the BOS and the EBOS techniques with respect to those parameters, trying to make the EBOS the most accurate as possible in the measure of a density gradient field.
Then, the two techniques as previously optimized will be applied to an experimental case, represented by a hydrogen flame, to test the overall accuracy of the techniques and to confirm if the EBOS represents an improvement to the BOS in terms of accuracy over an experimental case.

1 | Optical techniques

The insight into a physical process is always improved if a pattern produced by or related to this process can be observed by visual inspection [17].
The aim of this chapter is to introduce the techniques that can provide the visual inspection necessary to the understanding of many physical processes, and in particular one technique that has been studied in this thesis.
First it will be given a short review to the optical techniques in general, then the focus will be posed on the Schlieren techniques and the Background Oriented Schlieren (BOS). Finally, the *Enhanced Background Oriented Schlieren*, as it was defined by Cozzi and Göttlich in [4], will be explained into detail, and the main differences from the traditional BOS will be analyzed.

1.1. Flow-visualization techniques

The optical techniques are often applied to fluids as, since most of them are transparent, the direct observation through naked eye or standard cameras doesn't give enough information about the fluid properties. When the optical techniques are applied to a flow, they are called "Flow-visualization techniques".
These techniques offer significant advantages when compared to traditional measurement techniques:

- firstly, they are minimally intrusive, as no measure instrument is inserted into the flow. A Pitot tube, for example, offers precise information about the local velocity of the flow field, but it can cause turbulence or, if the Mach number is high enough, flow separation.
  Flow visualization techniques, on the other hand, exploit light properties to give information about the flow field, without altering its velocity, temperature, pressure or density.

- Secondly, a flow-visualization technique offers information about every point of the flow field at the same time, whereas a thermocouple, for example, gives the temper-

ature only of a single point.

In order to measure it, *the flow field can be made visible only indirectly and therefore one of the following techniques is needed:* [23]

- foreign material added to the fluid;
  the visualization is made on the foreign material, and not on the fluid itself, but if the parts composing the material are small enough, one can assume that they move together with the flow without altering it.
  The material can be totally immersed in the fluid or can be put on a surface that is, in turn, immersed in the fluid, usually as a side of the test chamber.
- Energy addiction to the fluid;
  in this case fluid elements with increased energy level are introduced in the flow. In some cases *the energy released is so high that the marked fluid elements become self luminous* [17], whereas in other cases they are made visible with the help of some optical techniques.
- Changes of the refractive index;
  since the refractive index of a fluid has a straight dependence on its density, compressible fluids can be observed by the variation of the refractive index, with the use of optical methods that make the refractive index variation clear.
  With the help of the correct mathematical formula, one can then trace back the density of the fluid.

The latter techniques are the ones of interest in this thesis work, and they will be treated into detail in the next section.

1.2. Density-based techniques

As mentioned before, the density-based techniques exploit the relationship between the density of a fluid and its refractive index to measure the latter and get the fluid density. Depending on the experimental case, the density can then be utilized to measure, for example, the fluid velocity in cases of high-speed flows, the fluid temperature in cases of combustion or heating, the mixing of different density fluids, and so on.

Starting from the electrical and physical properties of a fluid, as well explained in [17], it is possible to get an equation that expresses the refractive index as a function of the density of the fluid, this equation is often called the Clausius-Mosotti relation.

$$\frac{n^2-1}{n^2+2} = \frac{\rho\, L\, e^2}{3\pi\, m_e\, Mm} \sum_i \frac{f_i}{v_i^2 - v^2} \tag{1.1}$$

Where n is the refractive index, ρ the density and Mm the molar weight of the fluid, L is the Loschmidth's number, m_e the mass of the electron, f the oscillation strength and v the resonance frequency.

In the case of gases, the above equation can be simplified to the Gladstone-Dale relation.

$$n - 1 = K\rho \tag{1.2}$$

Where K is called the "Gladstone-Dale constant" and it depends both on the gas physical properties and on the frequency of the light that passes through the gas.
Values for the Gladstone-Dale constant has been determined experimentally for the most common gases and for different wavelengths; some are reported in the table 1.1.

Gas	**K** (cm^3/g)	**Wavelength** (μm)	**Temperature** (K)
He	0.196	0.633	295
Ne	0.075	0.633	295
Ar	0.157	0.633	295
Kr	0.115	0.633	295
Xe	0.119	0.633	295
H_2	1.55	0.633	273
O_2	0.190	0.589	273
N_2	0.238	0.589	273
CO_2	0.229	0.589	273
H_2O	0.310	0.633	273
CH_4	0.617	0.633	295
CF_4	0.122	0.633	295
SF_6	0.113	0.633	295

Table 1.1: Gladstone-Dale constant for different gases.

Once the relationship between the refractive index and the fluid density of a gas is established, the focus can be moved to the refraction of a light ray in a varying refractive

field.

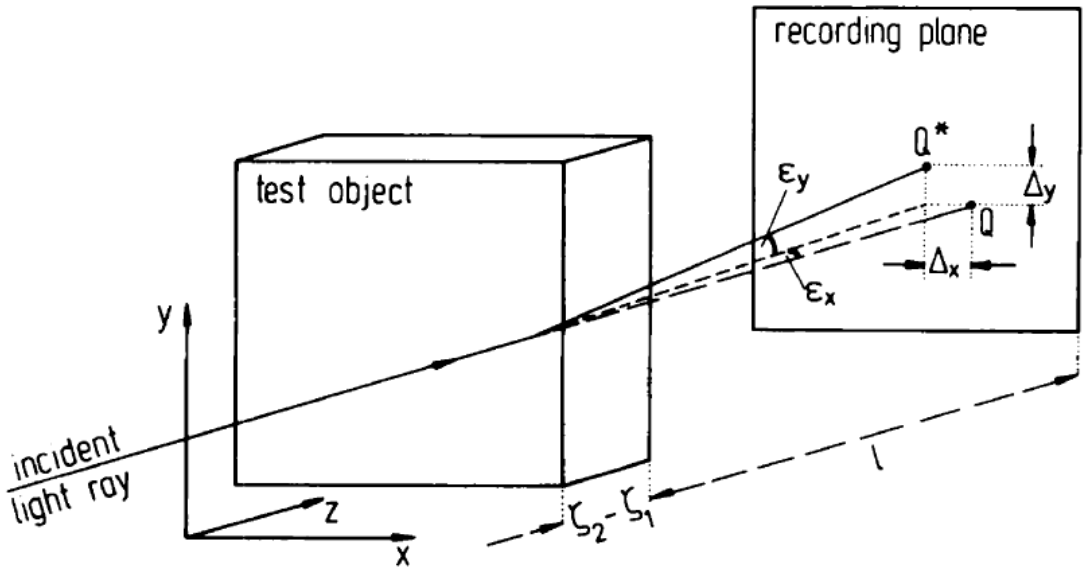


Figure 1.1: Deflection of a light ray in an inhomogeneous test object, [17].

As shown in figure 1.1, if a light passes through a test object with a different refractive index from the surrounding ambient, it will undergo a deflection and a phase change, as explained in [17]. Assuming optical geometry and assuming small deviations of the light ray inside the test object, it is possible to obtain equations for both the angles of deflection ϵ_x and ϵ_y (and the eventual displacements on a recording plane beside the test object ΔX and ΔY) and for the phase change $\Delta\phi$ [24].

$$tan(\epsilon_x) = \int_{\zeta_1}^{\zeta_2} \frac{1}{n}\frac{\delta n}{\delta x}\, dz; \tag{1.3a}$$

$$tan(\epsilon_y) = \int_{\zeta_1}^{\zeta_2} \frac{1}{n}\frac{\delta n}{\delta y}\, dz; \tag{1.3b}$$

$$\Delta X = l \int_{\zeta_1}^{\zeta_2} \frac{1}{n}\frac{\delta n}{\delta x}\, dz, \tag{1.3c}$$

$$\Delta Y = l \int_{\zeta_1}^{\zeta_2} \frac{1}{n}\frac{\delta n}{\delta y}\, dz; \tag{1.3d}$$

$$\frac{\Delta\phi}{2\pi} = \frac{1}{\lambda}\int_{\zeta_1}^{\zeta_2} (n(x,y,z) - n_\infty)\, dz. \tag{1.3e}$$

In the equations, λ is the wavelength of the deflected light ray, ζ_1 - ζ_2 is the depth of the schlieren object and n_∞ is the unaltered refraction index of the fluid.

Through history, many density gradient methods have been developed; some of them, like the interferometry, exploit the phase change of the light ray inside the test object, others, like the shadowgraph of the schlieren techniques, exploit the deflection of the light rays, and their displacement on a recording plane.

1.3. Schlieren technique

As mentioned before, the schlieren technique is a density gradient technique that is capable of visualising the deflection of light rays that pass through a media with variable density.
It has been theorized by Foucault (1859) and Toepler (1864), and it is widely used also today in many applications. It can be found in the aerospace field, mainly in wind tunnels, to map the high-speed air flows around an aerodynamic part or inside a conduct; it can be also used in optics, *as a quality control measure to check whether a glass pane is free from defects and other issues that could affect its performance* [5]; it is used also in combustion engines, to map the spray diffusion and evaporation.

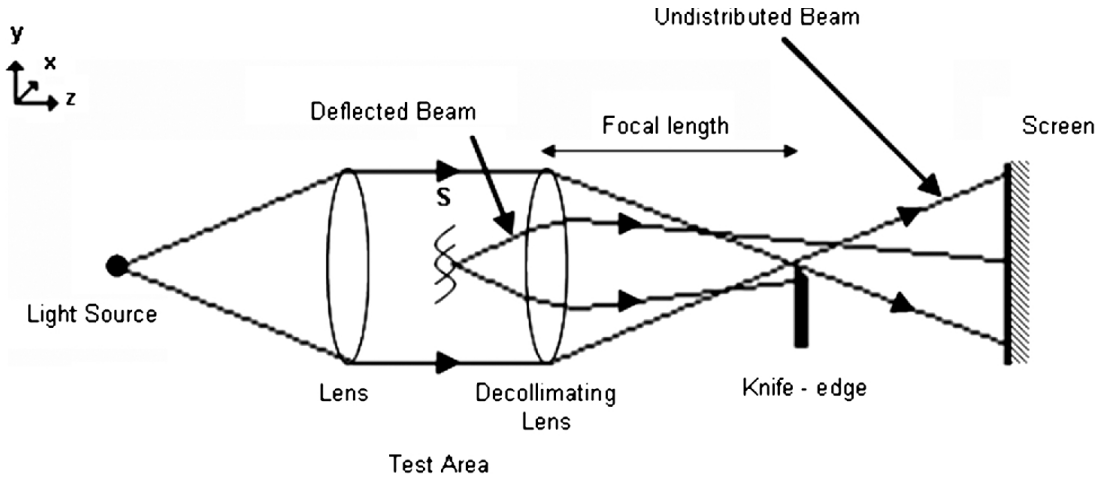


Figure 1.2: Diagram of a lens based schlieren system, [27].

The name of the method comes from the German word "schliere" that means a line (more or less irregular) with a different color from the background.
As shown in figure 1.2 and in [24], the classical setup for a schlieren system is composed by a point light source, two lenses, a knife edge and a screen. The light coming from the source is collimated by the first lens and passes through the test area; then the second lens refocuses the light rays to a point; from there, the rays diverge to the screen where the image is projected.

Without the schlieren object (this is how is called the test area where the studied phenomenon is happening), the screen would result in a uniform white image, but by adding a razor blade to the focal point, it is possible to reduce the overall luminosity and obtain a darker picture. If the schlieren object is introduced, the deflection of the light rays will cause some rays to avoid the obscuration introduced by the knife edge, resulting in a brighter area on the screen; some other rays, instead, will be completely blocked by the razor blade, resulting in a darker area on the screen.
The final picture will be a representation of the schlieren object made by darker or brighter areas where the density gradient is different from zero.
A various number of setups have been used along the years for the schlieren technique, those are obtained by varying the number of lenses, their position and their shape, by adding parabolic or plain mirrors and other devices, but the main concept remains the one explained above.

1.4. BOS technique

The Background Oriented Schlieren (BOS) *is an optical density visualization technique, belonging to the same family as schlieren photography, shadowgraphy or interferometry. In contrast to these older techniques, BOS uses correlation techniques on a background dot pattern to quantitatively characterize compressible and thermal flows with good spatial and temporal resolution* [21].
When compared to the traditional schlieren, the BOS requires a simpler setup, that usually consists in a camera and a background, with the schlieren object in between.
The technique is based on the comparison of two pictures, the first is made with the schlieren object located between the background and the camera, and the second is a simple picture of the background (the results obtained by varying the order of the two pictures are the same, except for the verse of the displacement); in the first picture, the ray of light coming from the background to the camera lens is disturbed and deflected by the presence of the schlieren object, whereas in the second picture the ray is undisturbed, thus creating a difference in the two pictures that will show the same point of the background in two different positions.

The pictures are then compared with the help of a software, usually a cross-correlation based algorithm, and a map of displacement is then reconstructed. By using the inverse formulas of the light ray defection 1.3, it is possible to obtain the refractive index of the schlieren object from the displacement map, and by using the Gladston-Dale rule 1.2 the density of the fluid is obtained, as explained below.

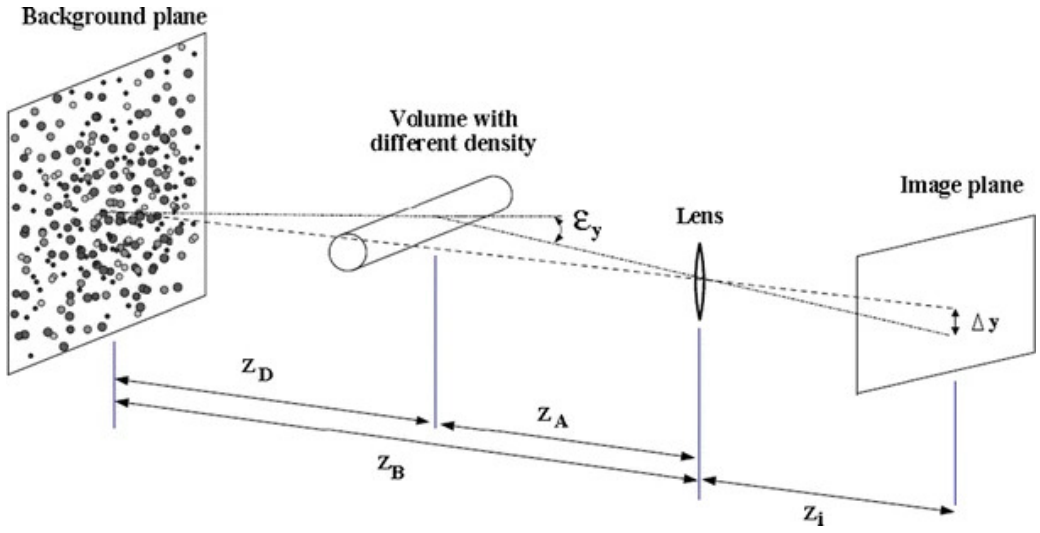


Figure 1.3: Typical BOS setup, [21].

Unlike schlieren and shadowgraph, and similar to the interferometry, the BOS is capable of a quantitative measure of the density gradient field, giving the technique the possibility to be applied in more contexts and to give more information about the flow field.
In case of 2D schlieren object, i.e. those showing density variation only along 2 different directions, the deflection of the light rays is concentrated in one plane perpendicular to the z axis, it is possible to express the displacement on the image plane as a function of the deflection angle and some geometrical parameters.
Referring to the configuration shown in picture 1.3, it results:

$$\Delta y = \frac{z_D \, z_i}{z_B} \epsilon_y \tag{1.4}$$

Where Δy is the light ray displacement on the sensor, z_D and z_B the distance from the background to, respectively, the schlieren object and the lens, and z_i the focal length.
By combining equations 1.2, 1.3 and 1.4, by assuming the refraction index constant through the Schlieren object, and by assuming small deflection angles, the displacement can be expressed as:

$$\Delta y = \frac{b \, K \, z_D \, z_i}{z_B \, n_0} \frac{\delta \rho}{\delta x} \tag{1.5}$$

Introducing the magnification factor M and reversing the equation, it is possible to obtain the density gradient as a function of the displacement map and the setup parameters.

$$\frac{\delta\rho}{\delta x} = \frac{n_0}{bKMz_0}\Delta x \tag{1.6}$$

$$\frac{\delta\rho}{\delta y} = \frac{n_0}{bKMz_0}\Delta y \tag{1.7}$$

1.4.1. Background

The background plays a key role in the BOS: placed behind the schlieren object, it makes it possible to compare the two pictures and generate the displacement field. It is usually made by number of small elements on a background of a different color.
As shown by Vinnichenko [26], the choice of the background has a significant impact on the accuracy of the measure, and it has to respect certain conditions:

- it has to be bigger than the observed phenomenon;
- it has to have the highest spacial frequency, in order to grant the highest resolution as possible. However, a minimum dimension is suggested for the elements, in order to avoid computational problems such as the peak-locking (more information about this phenomenon on [18]);
- it has to present a high contrast, to make it easier for the cross-correlation algorithm to evaluate the displacement field. Usually, greyscale images are used, but the use of colored dots has also been tested [19];
- it has to present a random pattern, in order to avoid the recognition of the regular pattern as apparent particle's moving by the software.

An example of a typical background is shown in figure 1.4, but many different solutions are possible. In some cases, it has been used a natural background (a bush, a tree, or something that presents the characteristics listed above) instead of an artificial one [9].

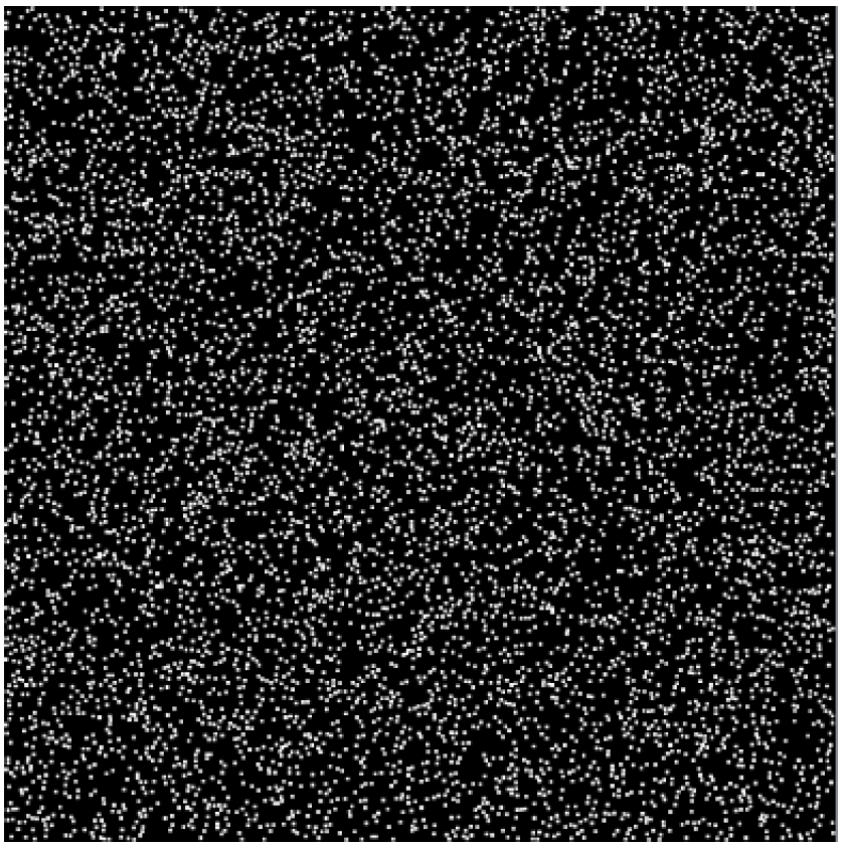


Figure 1.4: Example of a BOS random background.

Traditionally, the background image was generated by spraying a black speckle pattern on a white background [28], and the background was illuminated by a light source.
However, in the last years it has been preferred the use of a monitor that can generate instantaneous images and is self illuminated [28], [22], [15], [3], [25] reducing the complexity of the setup and allowing a faster generation of the images.

1.4.2. Image analysis

After the acquisition of the two images, the unaltered background and the background modified by the schlieren object, a software is needed to evaluate the displacement field. Since this comparison between the two images with dots is very similar to what happens in the Particle Image Velocimetry (PIV), and being the PIV a more common and studied technique, most of the times the algorithms from the PIV are used, after some adaptations, also for the BOS. The PIV software used for the BOS are usually based on cross-correlation algorithms.
It is possible to use different techniques, like the optical flow; however, they have not been treated in this thesis work.

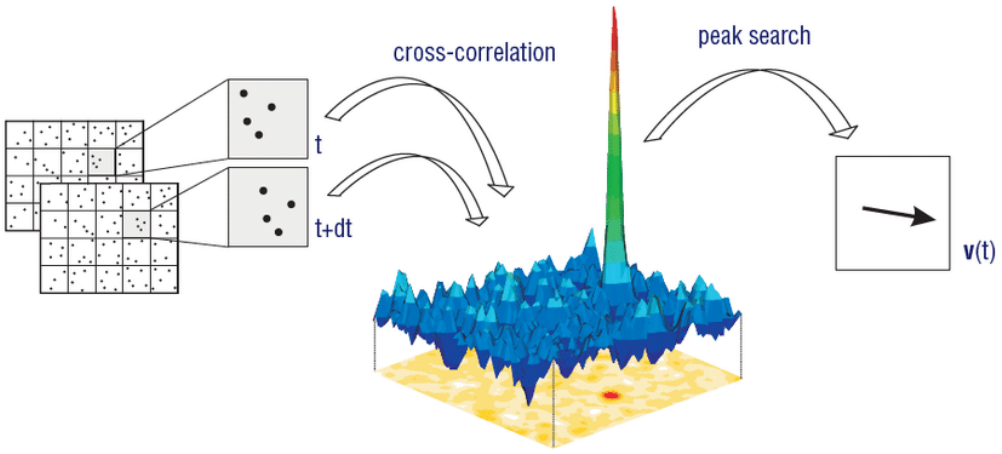


Figure 1.5: Cross-correlation algorithm.

As shown in figure 1.5, the two pictures are divided in interrogation areas (IA), then the cross correlation is evaluated between the IAs of the two pictures, resulting in a displacement, which is attributed to the whole IA. In order to increase the spacial resolution, it is possible to use the information obtained by the first cross correlation to move the IA of the second picture of the displacement found, and subdivide the IAs in smaller IAs. After that, a second cross correlation is carried out, resulting in a more detailed displacement map.
This procedure is called "adaptive cross correlation" and it can be done several times in order to increase the resolution of the displacement field.

The cross-correlation algorithms usually offer wide possibilities of configuration. The user has the possibility to choose the dimension of the first and of the last IA (in sensor pixels), the number of iteration for the adaptivity and often also the dimensions of the IAs between the first and the last.
Another important parameter is the *overlapping*, which is the fraction of two subsequent IAs that are overlapped one to the other. The use of this function allows to have a displacement field more dense without a decrease in the last IA dimension. Most of the times it is used for the first iteration of the adaptivity, in order to have a comparable number of vectors as the analysis goes on. Sometimes the overlapping command is substituted by the *grid step size*, which returns the distance between the two subsequent IAs, but their purpose is the same and the functioning is very similar.

A different part of the setup of the PIV solver is the validation, in which the vectors found by the cross-correlation are checked and they are determined valid or not.

There are different strategies to determine whether a vector is valid or not, but they are divided in two categories:

1. the first group checks the peak found in the cross-correlation, its height and ratio to the second peak or to the noise in the same IA;
2. the second group check the vector resulted from the cross-correlation: its absolute value, its direction and value with respect to the whole field or to the neighborhood.

The vectors that are considered invalid are marked and substituted by an interpolating function.
This function is usually polynomial and uses the values found in the neighborhood of the vector to be replaced to define the interpolating function.

1.5. EBOS technique

Being the BOS not only a qualitative but also a quantitative technique, a number of studies has been carried in order to improve its resolution and accuracy.
A first method to improve the resolution is to reduce the IA of the last cross correlation, obtaining a higher number of data. The smaller interrogation areas, however, present a weaker cross correlation signal, thus decreasing the peak to noise ratio and causing a bigger error, so a trade off needs to be done between the resolution and the accuracy of the measure.
In the work of Vinnichenko [26] it is shown how the optimization of the background plays an important role in the minimization of the error, so one aspect of this thesis work will be the optimization of the background.
A different way to improve the technique is the application of filters, that can reduce the noise and thus increase he signal to noise ratio. However this alters the original pictures and can cause a loss in resolution.

The Enhanced Background Oriented Schlieren (EBOS) technique is an improved version of the BOS, theorized by *Cozzi* and *Göttlich* in 2019. It *makes use of n different undistorted images of a single grey-scale background pattern* [4], and compares this n images to a single Schlieren acquisition; the n pairs of images are then cross-correlated and averaged, to obtain a single map of displacements.

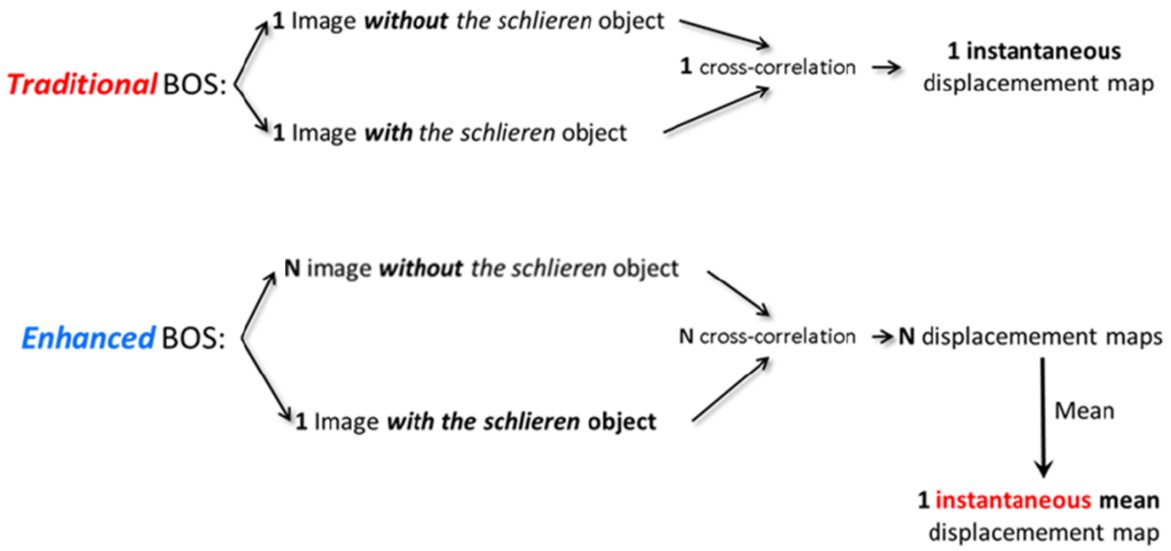


Figure 1.6: Comparison between BOS and EBOS, [4].

The main purpose of the EBOS is to improve the accuracy of the displacement estimation of the BOS, without altering the acquired images with filters and without any increment to the technique intrusivity.
The n background images are acquired *without* the Schlieren object, and only one picture is needed whit it; for this reason the technique can be also used to measure unsteady and turbulent flows, whereas acquiring multiple pictures of the Schlieren object, and averaging them would be unsuitable for a non-stationary flow field.
In order to increase the accuracy of the solution, the n background need to present some differences among them, such as a random motion of the particles or a rigid translations of the whole background, but this topic will be deeper discussed in chapter 6.

2 | Experimental setup

In this chapter, the components used for the experimental setup and their disposition will be analyzed.
The main elements of the setups, such as the camera, the lens, the monitor and the test bench, used in this thesis will be shown; then, the two configurations used for the experiments will be presented.

2.1. Components

The principal components for a BOS analysis are the camera and the background.
As explained in chapter 1.5, the background used for this analysis is displayed by a monitor, which is connected to a computer where the images to show are prepared. The camera is connected to a different computer where the pictures acquired are saved. A lens is attached to the camera.
The test bench connects rigidly these two elements, in order to avoid relative motion of one with respect to the other.

2.1.1. Monitor

The monitor used for all the experiments is an "Adafruit Qualia 9,7" Displayport", which is a LCD monitor with a resolution of 2048 x 1536 pixels at 60 Hz, 16.7 million colors and a contrast ratio of 1000:1 [13].
The monitor's pixel pitch is 0.096 mm both for the x and for the y axes, as proved by Casero in his thesis work [3], where he used the same monitor.

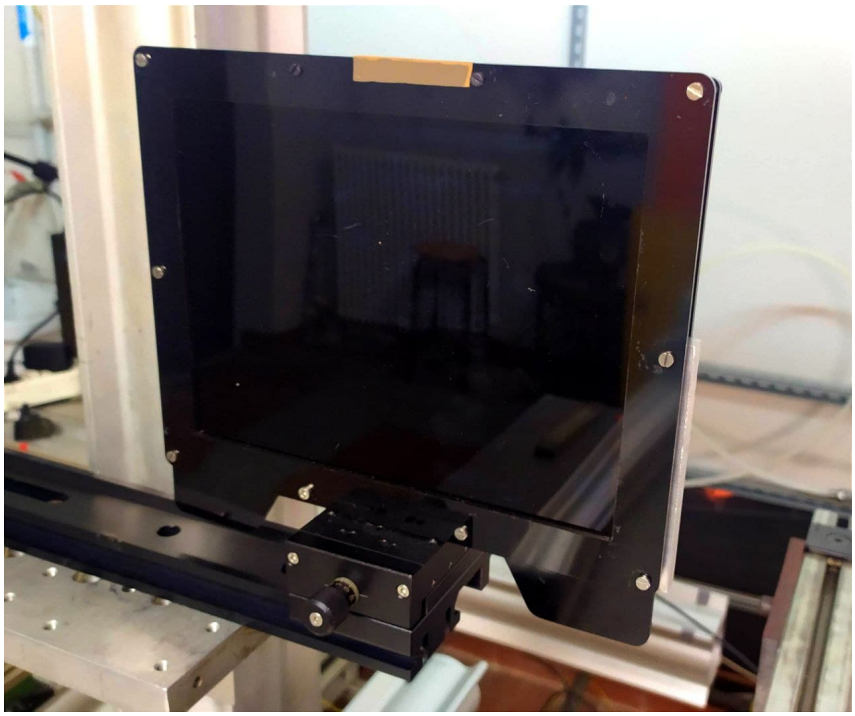


Figure 2.1: The monitor used for all the analysis.

2.1.2. Camera

The camera used for all the experiments is a Hamamatsu C8484-05.
The Hamamatsu C8484-05 is a CCD camera, without mechanical shatter and with a resolution of 1.37 million pixels [20].

Frame rate	Sensor resolution [pix]	Pixel pitch	Exposure time
12.2 to 68.4 fps	1344 (H) x 1024 (V)	6,45μm	4.5 ms to 1 s

Table 2.1: Technical characteristic of the camera.

The camera is connected to a Manfrotto 400 geared head, which is a high precision device that allows the rotation of the camera about the 3 axes.

Figure 2.2: Picture of the CCD camera used for BOS image acquisitions.

2.1.3. Lenses

The lens used for the majority of the experiments is a AF Nikkor 50mm f/1.8D from Nikon. It has a focal length of 50mm, an aperture that goes from 1.8mm to 22mm, subdivided in 8 stops, and a minimum focus distance of 45cm.

Figure 2.3: Picture of the 50mm lens, used for part of the analysis.

A different lens that has been used for some experiments is the 35-135mm f/3.5-4.5 Zoom-Nikkor from Nikon. It has a focal length that can vary from 35mm to 135mm, an aperture that goes from 3.5mm to 22mm, subdivided in 7 stops, and a minimum focus distance of 40cm.

Figure 2.4: Picture of the 35-135mm lens, used for part of the analysis.

2.2. Different setups

As mentioned before, two different setups have been used.
The first one is the one used for most of the analysis, which are presented from chapter 3 to chapter 6, and it is called "Synthetic displacement setup" (more information on synthetic displacement on chapter 3).
The second one is the one used in the last analysis, which is treated in chapter 7, and it is called "Hydrogen flame setup". This setup is more complex, since the presence of the hydrogen flame between the camera and the sensor imposes the need for a better maneuverability of the components.

2.3. Synthetic displacement setup

In the Synthetic displacement setup, the camera and the monitor are mounted on the same optical rail, that maintains their relative position constant.
They are both set on an optical rail carriage that allows movement along the rail on the z axis (referring to figure 1.1). This is necessary to regulate the distance between the camera and the monitor, and thus the magnification factor and the distances discussed

in 1.4.

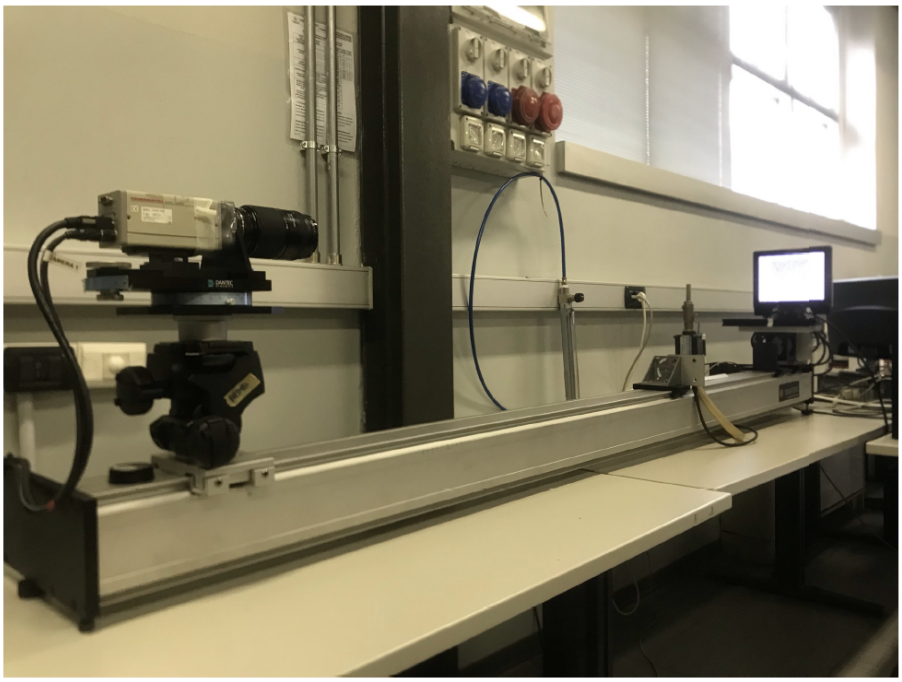


Figure 2.5: The synthetic displacement setup.

The monitor is mounted on an optical rails and carriers system that allows the lateral and vertical displacement, in order for it to be placed exactly on the same line of sight as the camera.

2.4. Hydrogen flame setup

Similarly to the synthetic displacement setup, in the hydrogen flame setup the camera and the monitor are mounted on the same optical rail. But since the position of the flame is fixed, in this case the setup has to be adapted and to be built around it.
The flame needs to be exactly between the camera and the monitor, perpendicular to the line connecting them. the monitor is placed on a longer rail then the one used before, allowing a greater vertical shift. The camera is set on an additional optical rail carriage that allows the horizontal displacement, since the flame position is aside to the main rail, as shown in figure 2.6.
Finally, the whole set is mounted on a table with adjustable height, to match the exact vertical position of the flame.
The flame comes out of a blowpipe, which is connected to the hydrogen distribution system, and is ignited by an electrical igniter placed above the blowpipe.

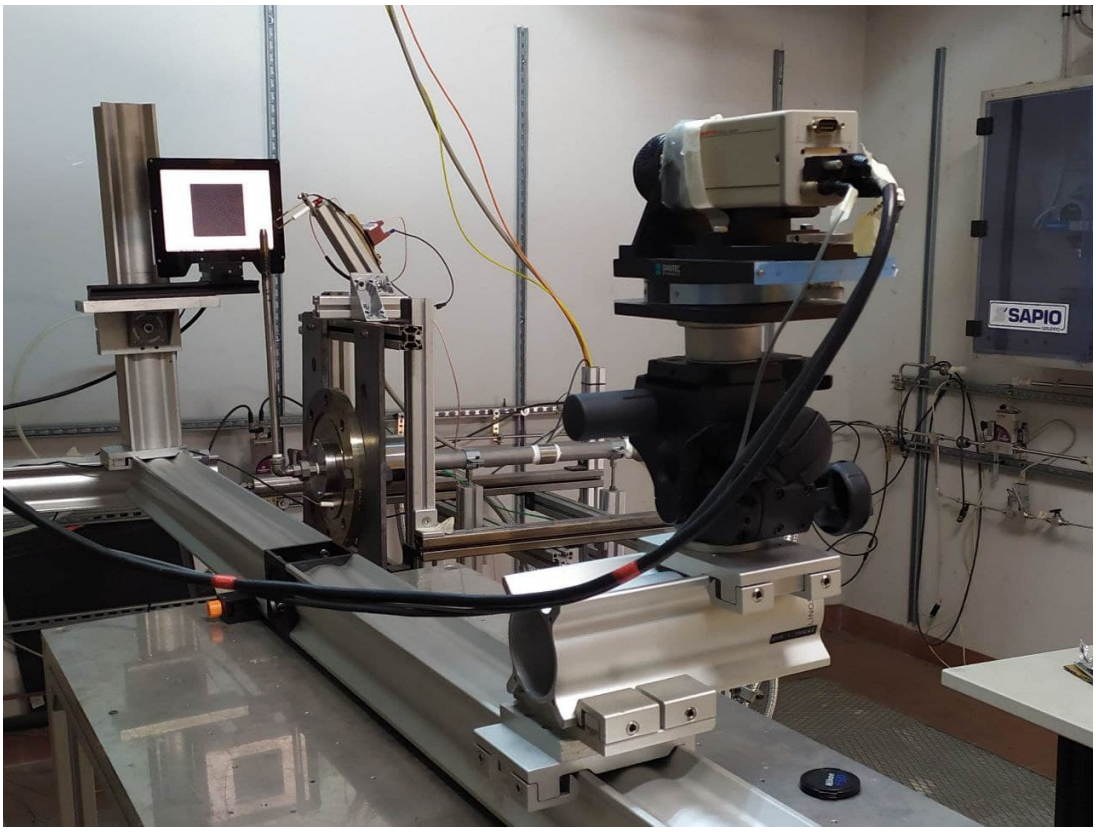


Figure 2.6: The hydrogen flame setup.

2.4.1. Hydrogen distribution system

The hydrogen distribution system at the combustion and optical diagnostic laboratory has been realised by Carrisi and Iapaolo during their thesis work [2]. It is composed by a set of pipes, valves and sensors that allow to control the air and hydrogen mixture.
The sensors and the valves are connected to the NI acquisition system, that is connected to a computer. On the computer, a Labview program allows to control the system parameters and to regulate the air and hydrogen flow rates. It is possible to directly impose the volumetric flow rate of the mixture components, or to impose the thermal power of the plant and the equivalence ratio of the mixture.
The last component of the hydrogen distribution system, before the actual burner, is a plenum: a hollow cylinder with a porous septum, to stop an eventual flashback of the flame. The plenum has four threaded holes where the burner can be mounted.

2.4.2. Flange and burner

The burner that have been used is a stainless steel pipe with an internal diameter of 8mm; it has been assumed to be a Bunsen burner.
In order to connect the burner to the distribution system, an elbow fitting and a series of joints was used, including a flange. The flange would connect a three-piece joint to the plenum of the distribution system, but since the plenum had very particular locations for the threaded holes, a new flange has been created for this specific purpose.
Starting from the CAD drawings of the plenum and from the dimensions of the three-piece joint, the flange was 3D modelled on Solidworks and a drawing was printed.
As shown in figure 2.8, the flange had a central threaded hole for the three-piece joint, four through holes for the connection to the filter and a guide for an o-ring.
An electrical igniter has been mounted on the structure, to ignite the flame. The igniter has been placed right over the burner, and its ground wire has been connected to the burner itself. This way, the spark would generate between the igniter and the burner border. A picture of the igniter and burner is shown in figure 2.7

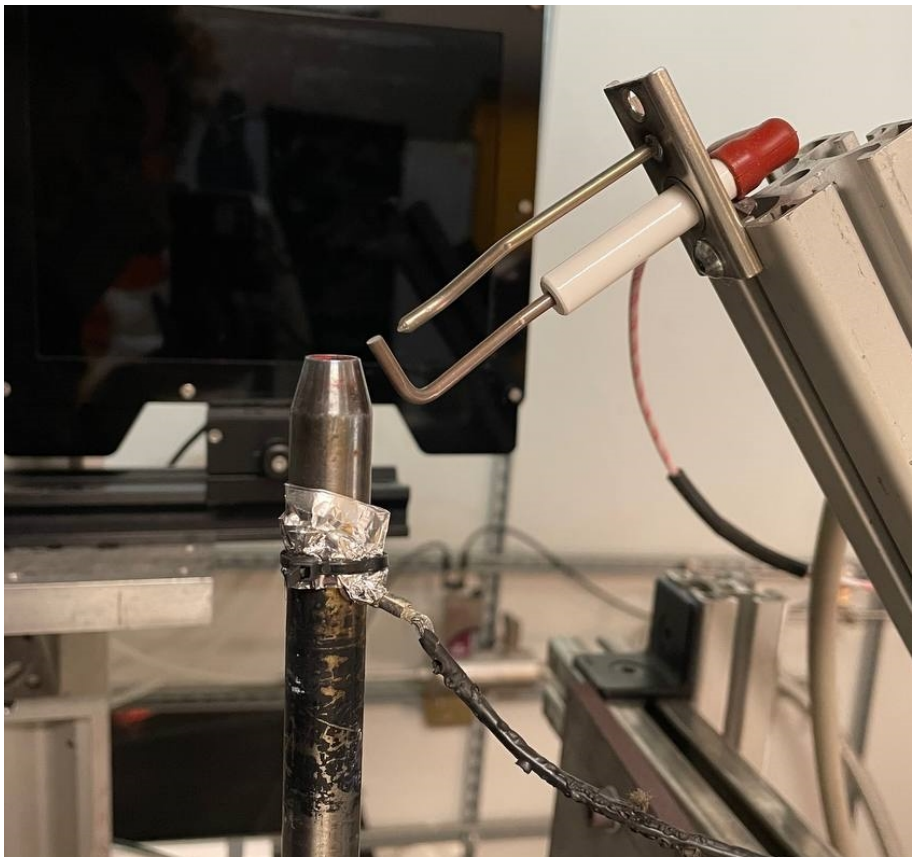


Figure 2.7: The igniter and the burner for the hydrogen flame setup.

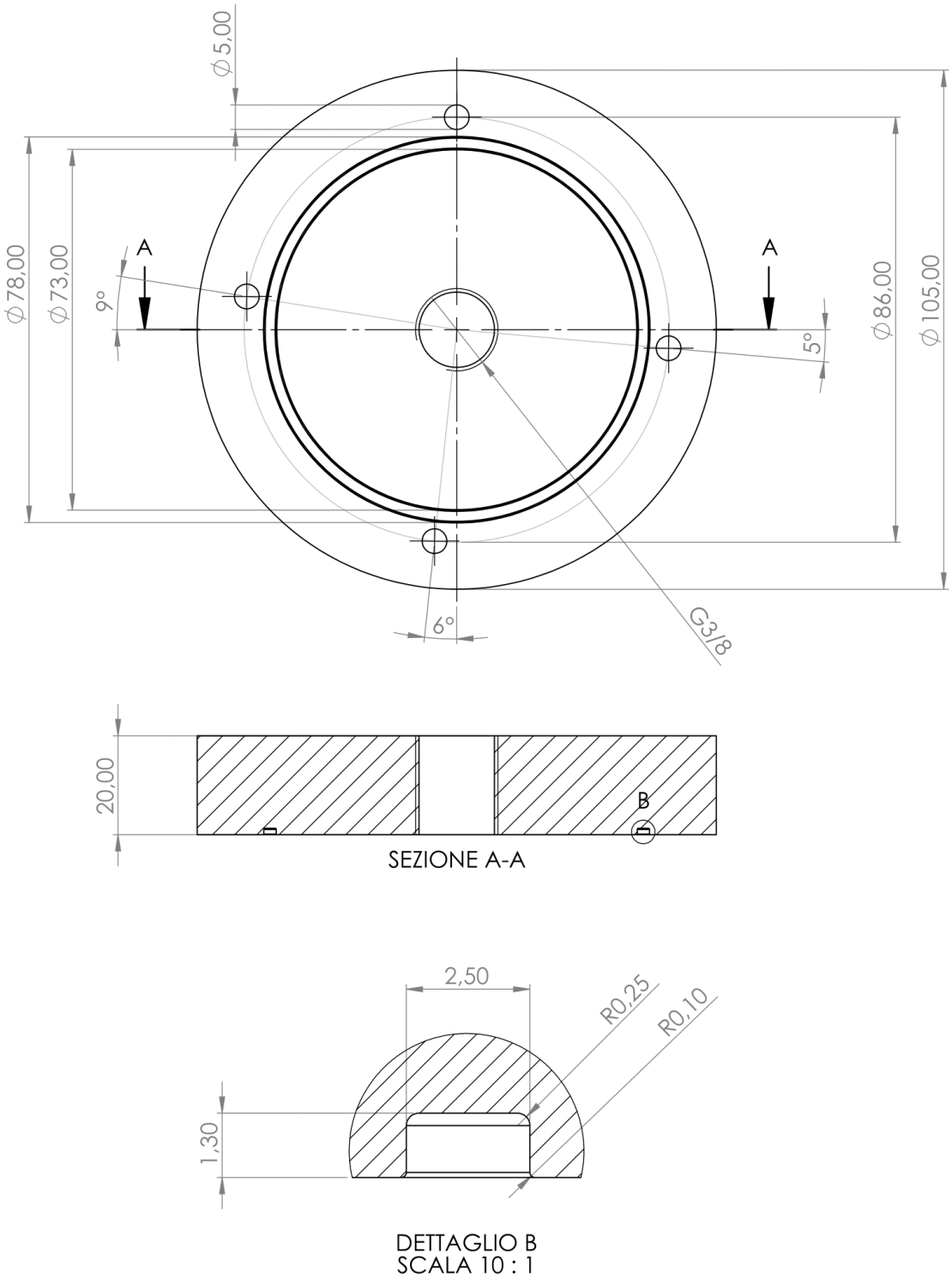


Figure 2.8: The flange 2D drawing.

3 | Background optimization

The main topics of this chapter are the background generation, or the explanation of the code used to create the background, and the background optimization, that is the set of analyses made to study the background sensitivity to the variation of its main parameters. The use of a monitor for the creation of the background is a recent improvement for the technique, but it has been treated in articles [22] and in some master thesis works [25], [3]. However, since the methods used to generate the background are various, it is necessary to explain how the background generation worked in this particular thesis.
The use of a monitor allows the best customization of the background generation parameters, and also the fast creation of the different backgrounds needed to evaluate the analysis behaviour at the variation of those parameters. For these reasons, the use of a monitor allows an in-depth study of the BOS background optimization.

3.1. Background generation

Like it happens for the cross correlation algorithm, also for the background generation it is easier to use a modified PIV software than to create a completely new program to generate the BOS backgrounds.
The program used in this case is a set of Matlab functions called "PIV-image-generator" [16], which was already adapted to the BOS by Sosio in 2021 [25]. The program has been further modified and adapted to the necessities in this thesis work.
Given the input parameters, the software makes the following steps:

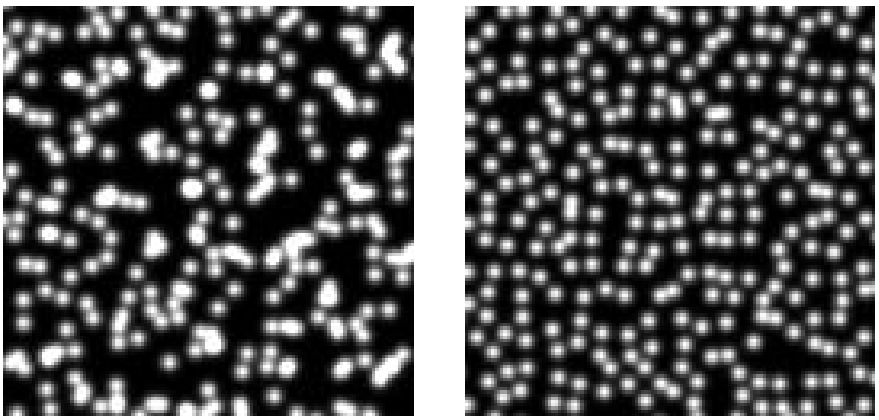
1. a black image of the desired dimensions is created;
2. the particles locations are assigned on the background;
3. to each pixel in the neighborhood of a particle is assigned a luminosity value depending of its distance from the particle center and the particle shape;
4. the previous step is repeated for each particle;
5. the final image is reconstructed from the recomposition of every neighborhood;

6. the light intensity of every pixel is regulated to be 8-bit.

The following sections will analyse the most relevant steps in the background creation, focusing on the ones of main interest in the next analyses.

3.1.1. Background creation

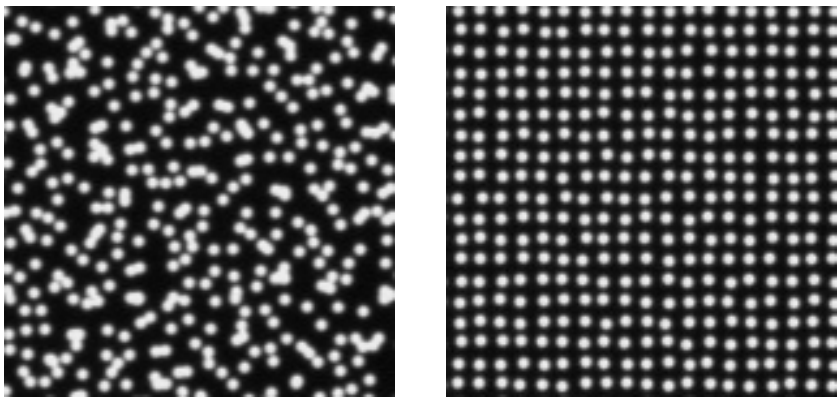
The first parameter of interest is the dimension of the background. After setting a dimension for the x and y directions, a matrix of the desired dimensions is created.
The background is then divided into background interrogation areas (bIA, to distinguish from the cross-correlation interrogation areas, or cIA), and a certain number of particles Ni per bIA is assigned. This allows to have a defined number of particle in each portion of the background, which results in a constant particle density on the screen. For each bIA, Ni random positions are generated, and a particle center is assigned to each location.
Given the particle radius, an anti-overlapping function allows to impose a minimum distance between the center of a particle and the center of the others, in order to avoid the overlapping of different particles.
The algorithm for the anti overlapping function can be found at A.2



(a) Anti overlapping function is off. (b) Anti overlapping function is on.

Figure 3.1: Comparison between anti overlapping function on and off ($Ni = 4$).

If there is only one particle per IA, a different function allows to impose that the position of the particle fits inside some desired margins. Increasing the dimension of these margins, it is possible to create a more and more regular background, as shown in figure 3.2.
A new variable is thus introduced, "mrg", which is defined as the unavailable area for the particle center with respect to the total area of the bIA. For an analysis without margins, $mrg = 0\%$. More information about the margin function can be found in A.1



(a) $mrg = 0\%$. (b) $mrg = 84\%$.

Figure 3.2: Comparison between bIA with and without margins.

3.1.2. Particle creation

Once determined the position of every particle in the background, the actual particles are generated.
An area of 40 x 40 pixels, centered in the particle center, is taken into account. Three functions have been defined to assign a light value to each pixel, depending on the distance between the pixel center and the particle center.

1. The "Black and White" function assigns a maximum value (255 = white) if the pixel center is inside the particle radius, and a null value (0 = black) if the pixel center is outside the radius.

2. The "Gaussian" function creates a two-dimensional Gaussian function with the peak on the particle center, the value assigned to a pixel is the value of the Gaussian function at its center. The particle radius determines the width of the Gaussian function, whereas the height is 255.

3. The "Truncated" function is similar to the Gaussian function, but the peak is set to 1500 instead of 255, and all the values greater than 255 are imposed as 255. This creates particles shaped as truncated cones, that give the name to the function.

Figures 3.3, 3.4 and 3.5 show an example for the pixel intensity distribution of a particle with a radius of 2 pixels.
For simplicity, it is represented only the pixel intensity along the x axis, and the vertical distance from the particle center is zero. In this case the particle center does not coincide with a pixel center, this the light intensity distribution is not symmetrical.
Figure 3.6 shows the rendering of few particles of different shapes with the same nominal

diameter.

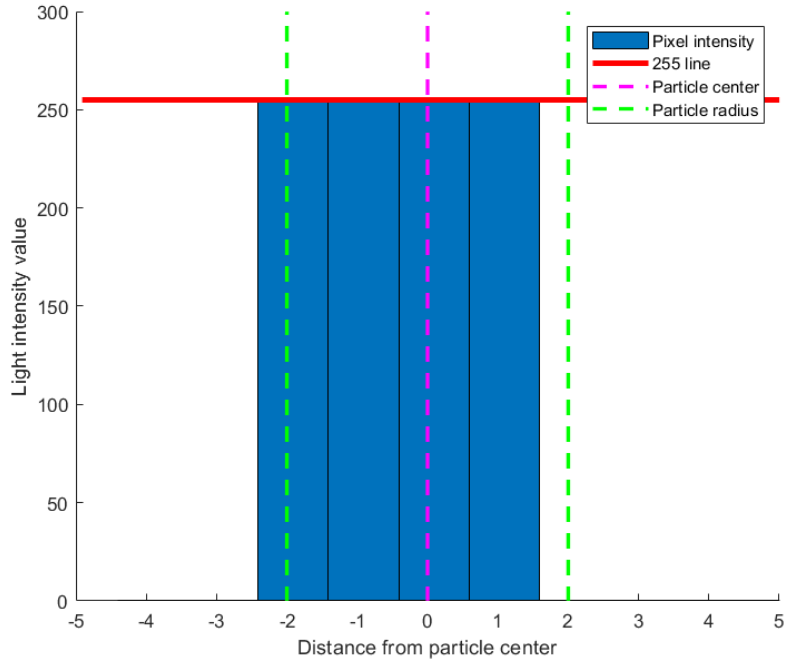


Figure 3.3: Black and white histogram.

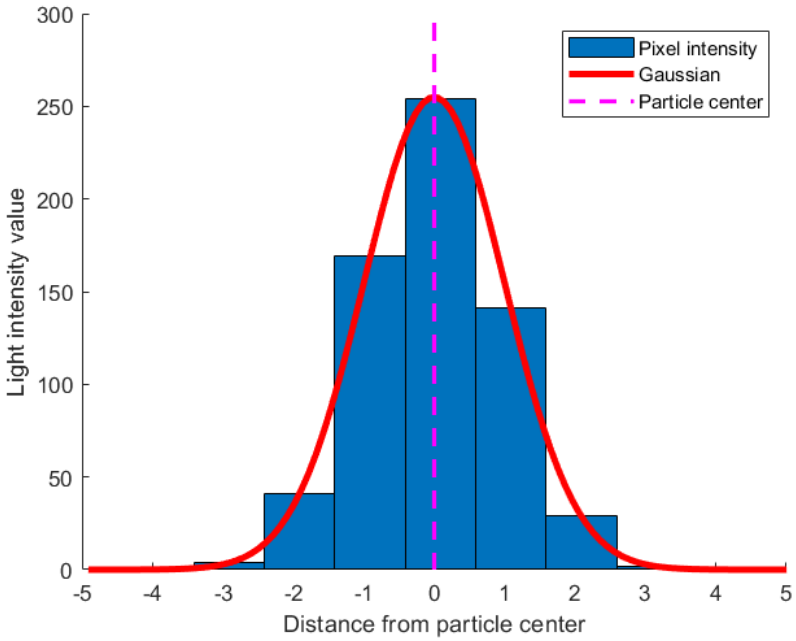


Figure 3.4: Gaussian histogram.

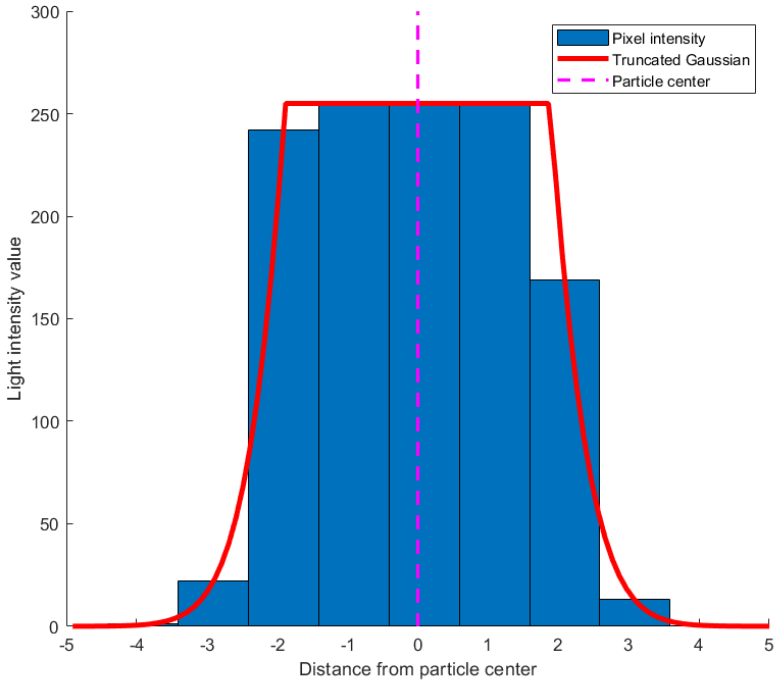
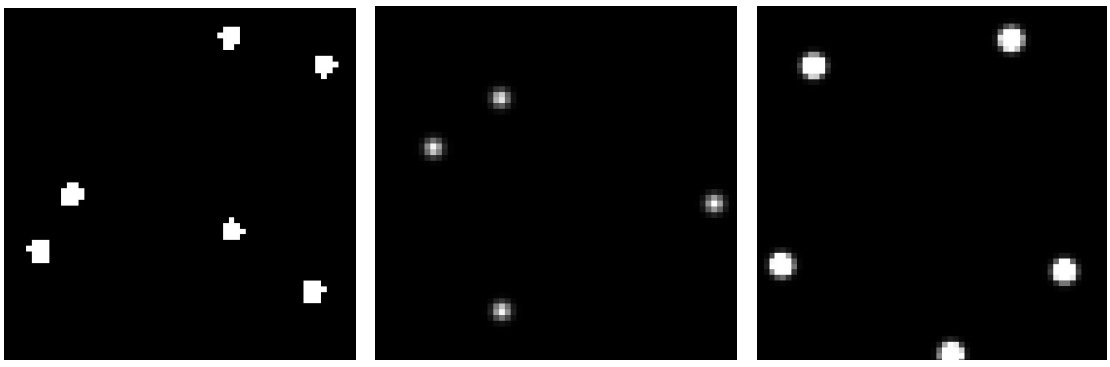


Figure 3.5: Truncated histogram.



(a) Black and white particles. (b) Gaussian particles. (c) Truncated particles.

Figure 3.6: Different particles with a 2 pixel radius.

As easily visible from picture 3.6, for the same nominal diameter, particles with different shapes result in having different dimensions.
More information about the particle shapes can be found in appendix A, in the algorithm A.3

3.1.3. Synthetic displacements creation

In order to mathematically evaluate the accuracy of a BOS analysis, a synthetic background has been used.
In a BOS analysis, one picture is acquired with the undistorted background, and one picture is acquired with the background distorted by the Schlieren object. A synthetic displacement is a modification of the background, that simulates the presence of a Schlieren object.
Since the imposed displacement field is known, it is possible to compare the displacement field measured by the cross correlation algorithm to the one imposed. The difference between the two fields is a measure of the error introduced by a BOS analysis, as better explained in section 3.2.2. This error evaluation has been used in studies that used a monitor as background, as Sosio [25] or Reichenzer [22].
Different synthetic displacement fields evaluate the accuracy of a BOS analysis from different views; for this reason various synthetic displacement fields have been used in the analysis:

- the "Diagonal" field has a constant region with null displacement, a region with linear displacement along both X and Y axes and a zero order discontinuity between them. The discontinuous region is critical for the cross-correlation algorithm, as it will see two peaks, which correspond to the particles at the two sides of the discontinuity. This is the most complete field since it presents a constant region, a non constant region and a zero order discontinuity.
  The displacement along x axis is maximum (3 pixels on the sensor) at the left side border of the image and is null at the right side border; the displacement along y is maximum (-3 pixels) at the bottom border and is null at the top border.

- The "Cross" field has four regions with different constant displacement, and 4 different zero order discontinuities. This field is mainly used to evaluate the BOS accuracy in measuring displacement zero order discontinuities.
  The displacement value is 3 pixels along x direction in the left quarter of the displacement field, and half that quantity in the right quarter, 3 pixels along y direction on the top quarter and half that quantity on the bottom quarter.

- The "Sinusoidal" field has a sinusoidal-type displacement along both X and Y axes. It is used to evaluate the BOS accuracy when non-polynomial displacements are present.
  The field is created by two sinusoidal functions (one function of the x and one function of the y position) multiplied together. The frequency is two times the

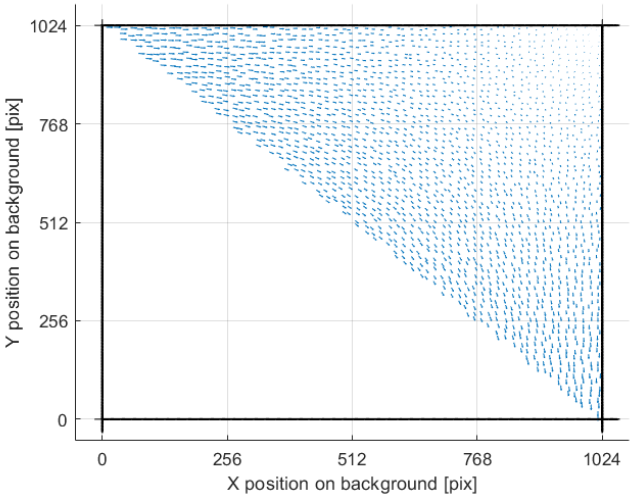
image size for both the functions and the overall amplitude is 3 pixels on the x direction and 2 pixels on the y direction.

- The "Parabolic" field has 4 regions of parabolic-type displacement along X and Y axes. It is used to evaluate the BOS accuracy in measuring polynomial displacements and second order discontinuities.
  The coefficients of the four parables are chosen such that the displacement is zero at the borders and along the straight lines perpendicular to the axes which pass through the center of the image. The maximum displacement at the parables' vertex varies depending on the analysis.

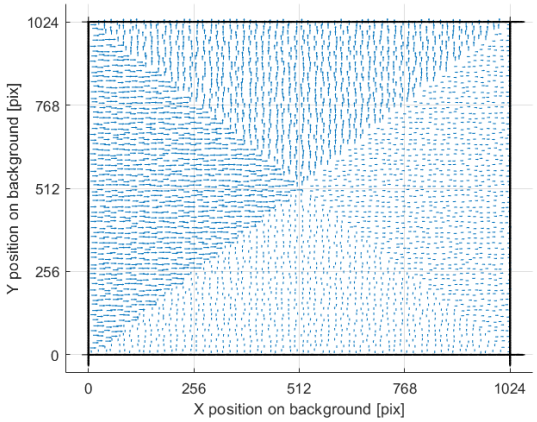
- The "Linear" field has to regions with increasing and decreasing displacements along the X axis. It is used to evaluate the BOS accuracy for large displacements and first order discontinuities.
  The displacement is equal to zero at the lateral sides of the image, and maximum at the center. The value for the maximum displacement varies depending on the analysis.

The algorithms used for the synthetic displacement fields can be found in Appendix A at A.4

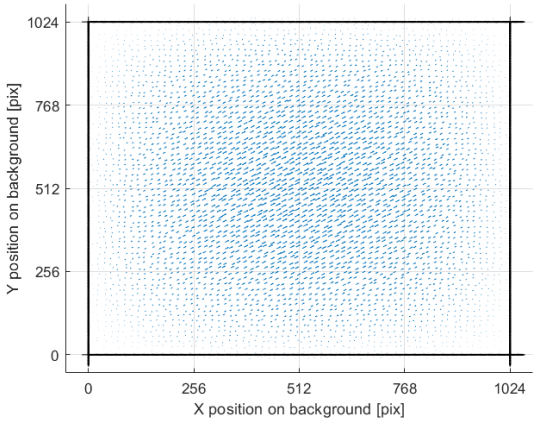
The distorted background images are created the following way: it is created a function that, depending on the location on the background, gives the new distorted position. The positions of all the particles centers are multiplied by this function, returning the new particle centers locations. Then, as it happened for the undistorted background, the new image is created from the particle positions, their shapes and their dimension.
The particles can be moved by fraction of pixels, but their shape is always circular, so the distortion is applied to their location but not to their shape.



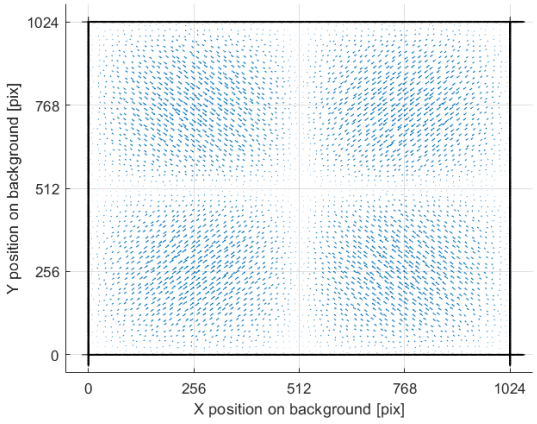
(a) Diagonal.



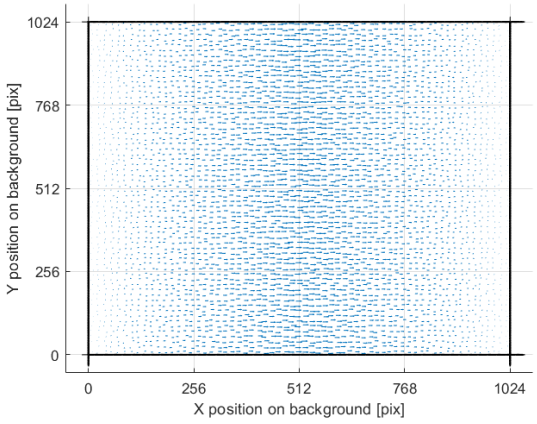
(b) Cross.



(c) Sinusoidal.



(d) Parabolic.



(e) Linear.

Figure 3.7: Visualization of the synthetic displacement fields used.

3.2. Analysis parameters

As mentioned before, the purpose of this analysis is to study the BOS sensitivity to the variation of some parameters related to the background creation.
The background used is a 1024x1024 pixels image, with Ni pixels every 16x16 bIA, and the non-overlapping function has been turned on.
The other parameters, of which the variation is studied, are:

1. The particle radius.
2. The particle shape.
3. The particle density Ni.
4. The margins for the particle centers.

For each background, a distorted image is generated, which has been created using the "Diagonal" synthetic displacement field. The undistorted and distorted backgrounds are acquired by the camera, transferred on Python where they are analyzed.
The distance between the camera sensor and the monitor is 1.49 meters, resulting in a magnification factor of $M = 0.0336$, which means that 2 pixels on the monitor correspond to 1 pixel on the sensor. Therefore, the image acquired by the camera is a 512x512 pixels image.
The camera and lens parameters that have been used are reported in table 3.1.

Parameter	Measure
Lens focal length	50mm
$F\#$	11mm
Camera shutter speed	30000μs
Background distance	1.49m

Table 3.1: Camera and lens parameters used for chapter 4 analyses.

3.2.1. PIV settings

The program used in this analysis is called "OpenPIV" (more information on [14]), which is an open source software for the PIV analyses that mainly operates on Python.
OpenPIV is a very customisable software that allows the user to choose every parameter for the adaptive cross-correlation algorithm. In particular the main parameters of interest concern the adaptivity of the algorithm, the validation of the results obtained and the

substitution of the outliers.
The parameters used are reported in table 3.2.

Adaptivity	
N° iterations	3
cIA dimension [pixels]	32 - 16 - 8
Overlapping	50% - 50% - 0%
Validation	
Signal to noise threshold	1.5
Global STD threshold	9
Local median threshold	5
Substitution	
Substitute	True
Interpolating neighborhood	±3 pixels

Table 3.2: PIV parameters used for chapter 4 analysis.

As shown in table 3.2, the cross-correlation algorithm is composed by 3 iterations with decreasing window sizes; for each iteration a different overlapping is used.
The PIV parameters will be deeper discussed in chapter 6.

3.2.2. Error evaluation

A Python script is necessary to compare the displacement field returned by the cross-correlation algorithm to the theoretical one, and to measure the difference between the two.
OpenPIV returns a matrix containing the positions of the last cIA centers, the displacements along X and Y axes for those positions, and a mark if the values have been substituted. The matrix is elaborated and two new matrices are created, containing the X and Y displacements. Being the image 512x512 and the last cIA size 8x8, these matrices have dimension 64x64.
Then, two more matrices are created, which contain the theoretical displacement that was imposed to the background at the same locations of the measured displacement field. For each element the difference in vector is calculated by using the following formula.

$$\begin{cases} e_u^{i,j} = \left(u_{disp}^{i,j} - u_{eff}^{i,j}\right)^2 \\ e_v^{i,j} = \left(v_{disp}^{i,j} - v_{eff}^{i,j}\right)^2 \\ e^{i,j} = \sqrt{e_u^{i,j} + e_v^{i,j}} \end{cases} \tag{3.1}$$

Following the error evaluation introduced by Atcheson & al [1] which was used specifically for the BOS and EBOS by Reichenzer [22], the total error is calculated as the mean of the errors of each matrix element.

$$e_{tot} = \frac{1}{N_x N_y} \sum_{i=1}^{N_x} \sum_{j=1}^{N_y} e^{i,j} \tag{3.2}$$

The error evaluated this way represents the mean pixel displacement (mpd) between the measured and the theoretical displacement fields.
Since the particles near the borders may exit the background limits and cannot be found by the cross correlation algorithm, the outer border of the matrix can be excluded from the calculation of the mean error. However, all the synthetic displacement fields that have been used present a null tangential displacement at the background border, so in this case the border correction has not been applied.
An error map can be created, with the local error value for each cell in the matrix. The error map is useful to easily understand where the error is mainly located, and to give a qualitative feedback of the analysis.
The algorithm used for the error evaluation is reported in appendix B at B.1.
An error map example is shown in figure 2.2.

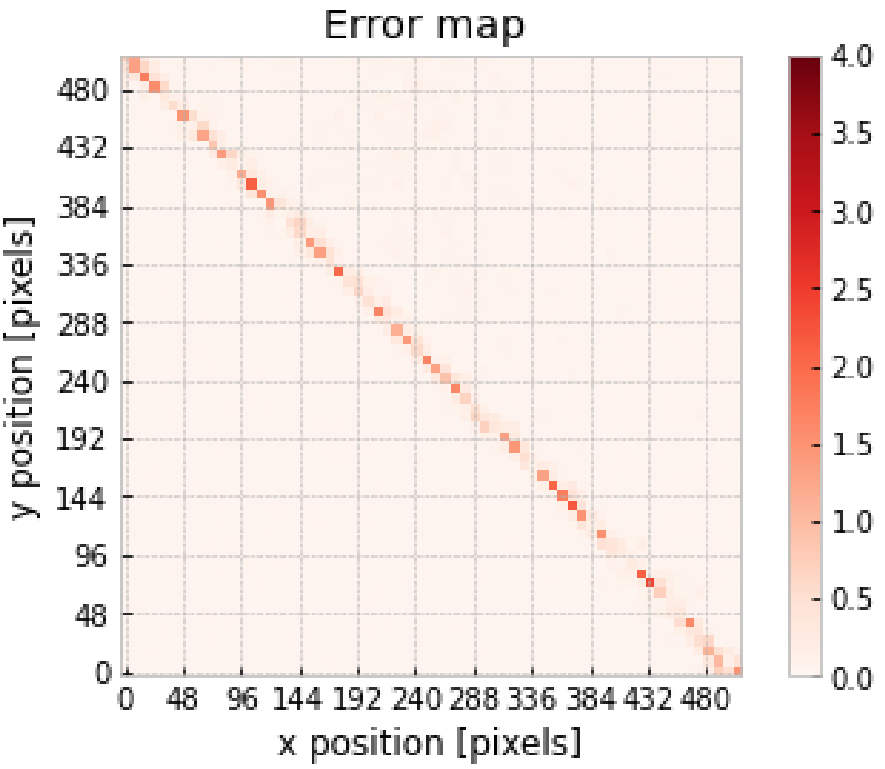


Figure 3.8: Mean error e_{tot} [mpd] map for Diagonal displacement field.

In this example, as it often happens for the Diagonal displacement field, the error is mainly located on the diagonal discontinuity, with a smaller error right aside of the diagonal

3.3. Results

In this section, the results obtained by varying the parameters mentioned in section 3.2 are presented. For each analysis, the mean error e_{tot} is used as an index of the quality of the background.

As happened in figure 3.8, the error is mainly concentrated near the zero order discontinuity, but the number of cIA along it is rather small when compared to the total of the elements in the error matrix. For this reason, the comparison between the analyses will take into account the whole displacement field, and the behaviour near the diagonal will only be decisive for the analyses close to the optimal.

All the variables taken into account are strictly connected: the shape of a particle also influences its size, and the increase in density is possible only if the particles are small enough. For these reasons the analyses that have been made take into account multiple parameters at once.

Three analyses have been made for this part of the study:

1. The first one takes into account the three different shapes and their behaviour as a function of their shape. The density is $Ni = 1$
2. The second one compares two shapes at different densities and diameters.
3. The third one studies the behaviour of a shape at its optimal diameter as the margins increase.

3.3.1. Influence of particle shape and dimension

The results obtained by comparing the three shapes at diameters that vary from 2 to 8 pixels are reported in table 3.3.
The diameter has to be intended as nominal, and it is measured as pixels on the camera sensor. The density is $Ni = 1$, the displacement field is "Diagonal", $mrg = 0$, and the maximum imposed displacement on the sensor is 3 pixels.

Particle diameter	d=2	d=3	d=4	d=5	d=6	d=7	d=8
Black and white	0.0825	0.0733	0.0640	0.0594	0.0614	0.0753	0.1098
Gaussian	0.0843	0.0631	0.0544	0.0517	0.0521	0.0571	0.0631
Truncated	0.0591	0.0542	0.0524	0.0537	0.0659	0.0891	Nan

Table 3.3: Mean error e_{tot} [mpd] for different particle shapes and sizes, Ni = 1.

As shown in the table, the accuracy of a measure decreases with diameters too small or too big. For a small diameter, the cross correlation algorithm has a little signal to cross correlate, so the signal to noise ratio decreases. For a particle too big, the image is mainly white and the noise increases, again lowering the signal to noise ratio. This result was also shown by Vinnichenko [26], who suggested that the particle radius should go from 2 to 3 pixels.
However, the optimum radius varies depending on the particle shape. The results are also showed in figure 3.9.

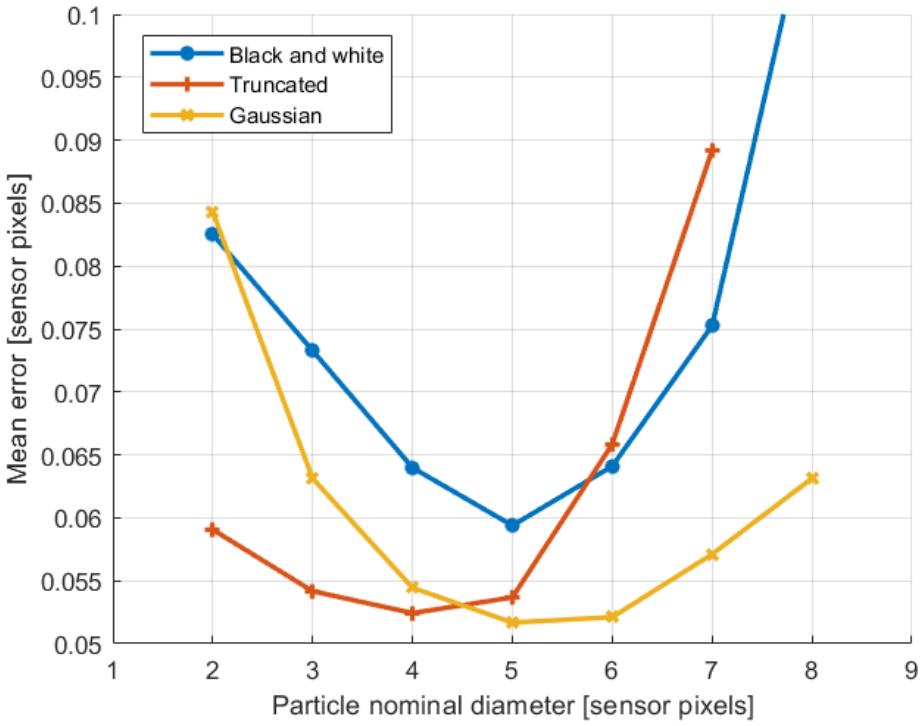


Figure 3.9: Mean error e_{tot} for different particle shapes and radii.

The "Black and White" shape results to be the worst performing one, and it will not be used in the future analyses.
The "Gaussian" shape is smaller than the "Truncated" one, so it makes sense that the best diameter for the Gaussian is larger than the best one for the Truncated. Apart from that, the two shapes show similar behaviours that need to be further investigated.

3.3.2. Influence of particle density

The second analysis compares the two shapes at different pixel densities. Since the density is strictly connected to the particle dimension, different diameters are used in this analysis too. The density used is 2 and 3 particles per cIA of 8x8 pixels, $mrg = 0$, and the diameter varies from 1 to 7 pixels.

Ni = 2	d=1	d=2	d=3	d=4	d=5	d=6	d=7
Gaussian			0.0539	0.0521	0.0528	0.0559	0.0658
Truncated	0.0774	0.0509	0.0478	0.0550	0.0110		
Ni = 3	d=1	d=2	d=3	d=4	d=5		
Gaussian			0.0542	0.0513	0.0549		
Truncated	0.0739	0.0469	0.0554				

Table 3.4: Mean error e_{tot} [mpd] for different particle shapes and sizes, $Ni = 2$ and $Ni = 3$.

As the particle density and therefore the total amount of particles increases, the number of white pixels on the background increases too. For this reason, at higher densities the optimum diameter decreases in order to keep constant the ratio of white pixels per image size.

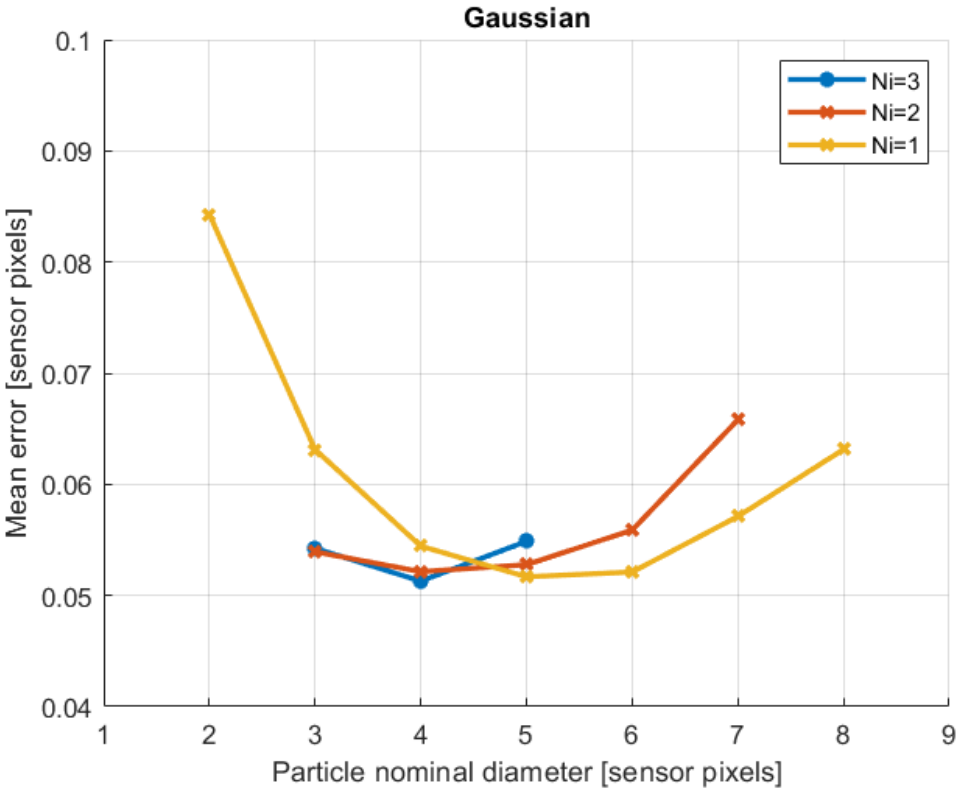


Figure 3.10: Mean error e_{tot} for Gaussian particles as a function of particle diameter d for three different particle densities, Ni from 1 to 3.

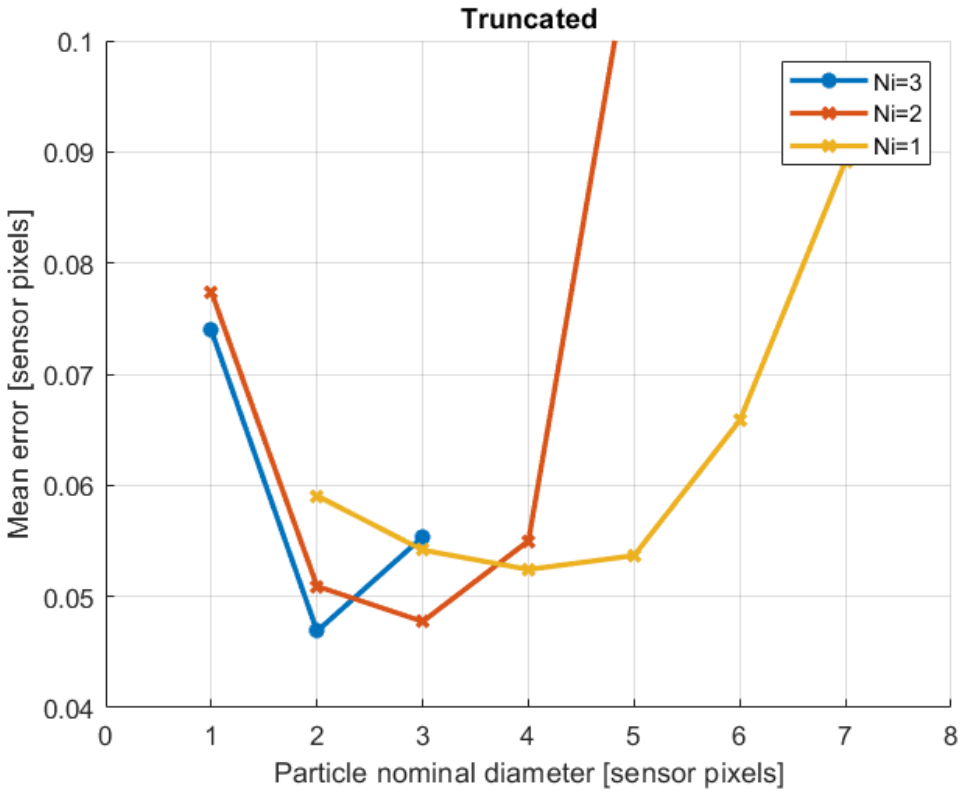


Figure 3.11: Mean error e_{tot} for Truncated particles as a function of particle diameter d for three different particle densities, Ni from 1 to 3.

As shown in figures 3.10 and 3.11, and as previously supposed, the optimal diameter decreases as the particle density density increases. However, the response to the density increase for the Truncated particle compared to the Gaussian particle is quite different. For every particle shape and density, the mean error relative to the optimum diameter is compared to the e_{tot} at $Ni = 1$. The results presented in table 3.5 show that there is a clear improvement in the quality of the measure along with the density only for the Truncated particle.

Pixel density	$Ni = 1$	$Ni = 2$	$Ni = 3$
Gaussian	0%	+2.11%	−0.75%
Truncated	0%	−8.93%	−10.59%

Table 3.5: Difference between e_{tot} for a specific density and e_{tot} at $Ni = 1$, for Gaussian and Truncated shapes. Optimum diameter considered.

The Truncated particle shows better performances if the density increases. This allows to have better results also for very dense backgrounds, which are necessary to operate with higher resolutions at smaller cIA.
For this reason, The truncated particle has been preferred to the Gaussian, and in the future analyses all the particles will present a Truncated shape.
From the same data, it can be determined the relationship between density and optimal radius. Comparing the total particle area, obtained by multiplying the particle area per the number of particles, it results that the nominal particle area should be from 15% to 22% of the total background area. If decimal diameters had been used for this analysis, it is probable that this value would be more precise.
This result allows to determine the optimal radius for any particle density that will be used in future analyses.

3.3.3. Influence of IA margins

The last analysis is about the use of the margin function discussed in chapter 3.1.1: the Truncated shape is tested with increasing margins to see if the implementation of a margin increases the accuracy of the measure.
For the analysis, a density of $Ni = 1$ and a diameter of 4 pixels have been chosen.

Margin	$mrg = 0\%$	$mrg = 36\%$	$mrg = 84\%$	$mrg = 95\%$
Truncated	0.0528	0.0496	0.0468	0.0458

Table 3.6: Mean error e_{tot} [mpd] for different margins, Truncated particles.

The table 3.6 shows an improvement in the analysis as the margins increase. However, margins too high can create a background too regular, with the consequence that the cross-correlation algorithm may confuse a particle with the displacement of another particle, thus calculating the displacement field incorrectly.
For this reason, only a margin up to $mrg = 50\%$ is suggested, and in the future analyses it has been used only the margin necessary to avoid particles overlapping, which varies from $mrg = 10\%$ to $mrg = 46\%$, depending on the particle dimension.

4 | Monitor quality analysis

The aim of this chapter is to present the work that has been done in order to analyse the quality of the monitor and to correct the possible non uniformity.
Even if theoretically the monitor is uniform, with each pixel having the same light intensity when given the same input, actual monitors show lighter and darker areas, due to hardware imperfections.
In this analysis the uniformity of the monitor is studied, and suggestions to compensate the non regularities are proposed. Finally, a short comparison between the BOS made with corrected and uncorrected monitor is carried out, and the results obtained are presented. For this work, all the calculations and the figures generation have been made on Python.

4.1. Preparation

The purpose of this work is to create a correction matrix for the monitor to use in the future analyses. For this reason, the correction matrix needs to have the same dimensions of the actual images acquired in the future works, so the study has been conduced using the same setup and background dimensions as the ones used in chapter 3: the setup is the synthetic displacement setup and the background dimensions are 1024 x 1024 pixels. The monitor has been studied in the configuration that maximizes the irregularities, that is the maximum intensity for each pixel, or a completely white image (light intensity 255/255).
To avoid external influences on the analysis, all the lights in the test room were turned off and the blinds were closed.
In order to also avoid camera and lens to have an impact on the analysis, two tests were made, the pictures were acquired at two different f-stops, with a shutter speed such that the sensor did not saturate (saturation homogenizes the image). The camera parameters that have been used are shown in table 4.1.

	Test 1	Test 2
f-number	11	5.6
Shutter speed	38000μs	9500μs

Table 4.1: Camera parameters for the monitor quality analysis.

For each test, a picture with the lens cap on was taken before acquiring the actual background images. The purpose of this procedure is to have a reference for what is a homogeneous dark picture for the camera at set parameters.
For each test, four different images were acquired, in order to limit casual monitor fluctuations.
A region of interest, relative to the 1024 x 1024 white square, has been cropped for each image, the cap picture relative to a test was subtracted to the four images to remove the possible camera influence, and the eight images obtained were eventually averaged to a single image.
Since in the synthetic displacement setup the magnification factor is equal to 0.0336, and given the pixel pitches of camera sensor and monitor, the actual image acquired by the camera has dimensions of 512x512 pixels.

4.2. Analysis

The eight images coming from the two tests were then averaged into a single image, that is a good approximation of what the part of the monitor used in the analyses looks like when set at maximum intensity.
As shown in figure 4.1, the monitor presents small imperfections and wide areas of higher and lower luminosity.
The imperfections are caused by scratches on the screen, visible also to the naked eye, whereas the areas of different luminosity are not observable without a contrast increase.
However, the light variation is up to 10% of the average, which is non negligible, and it is necessary to further investigate whether or not this has a negative impact on the BOS analyses.

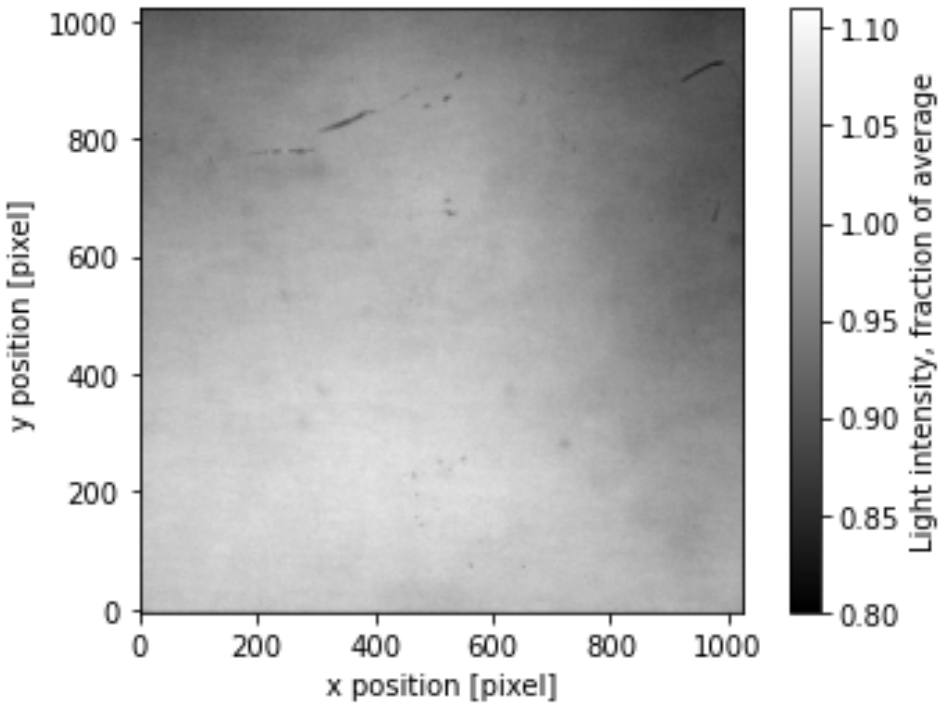


Figure 4.1: Average image of the monitor.

The figure 4.2 shows the statistical distribution of the luminosity in the averaged image, and the table 4.2 enlists the principal statistical parameters: as it can be seen the light is not normally distributed on the monitor, as it presents a negative skewness.

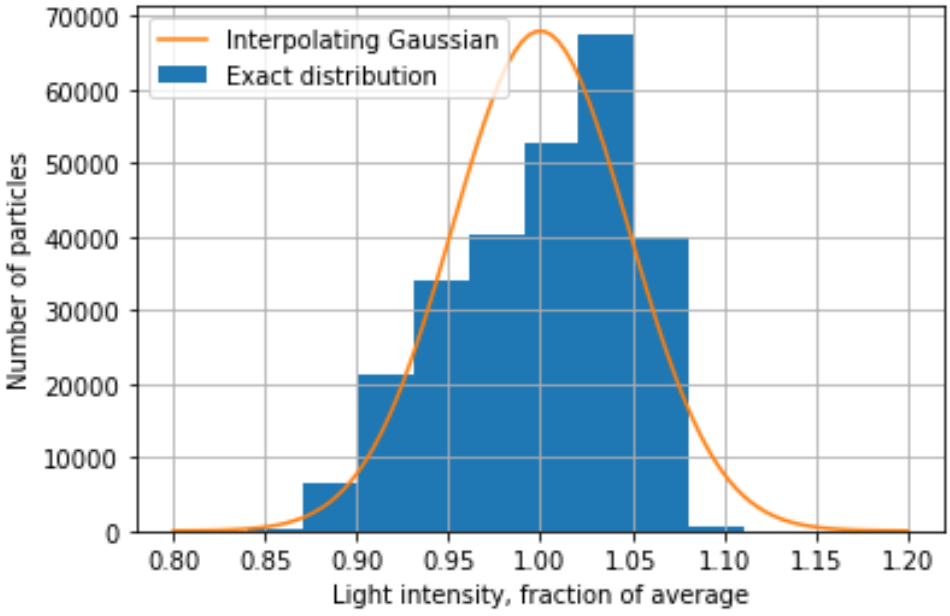


Figure 4.2: Average light intensity distribution of the monitor.

Mean value	1
Variance	0.048
Skewness	-0.499

Table 4.2: Statistical parameters of monitor's average light distribution.

4.2.1. Correction matrix

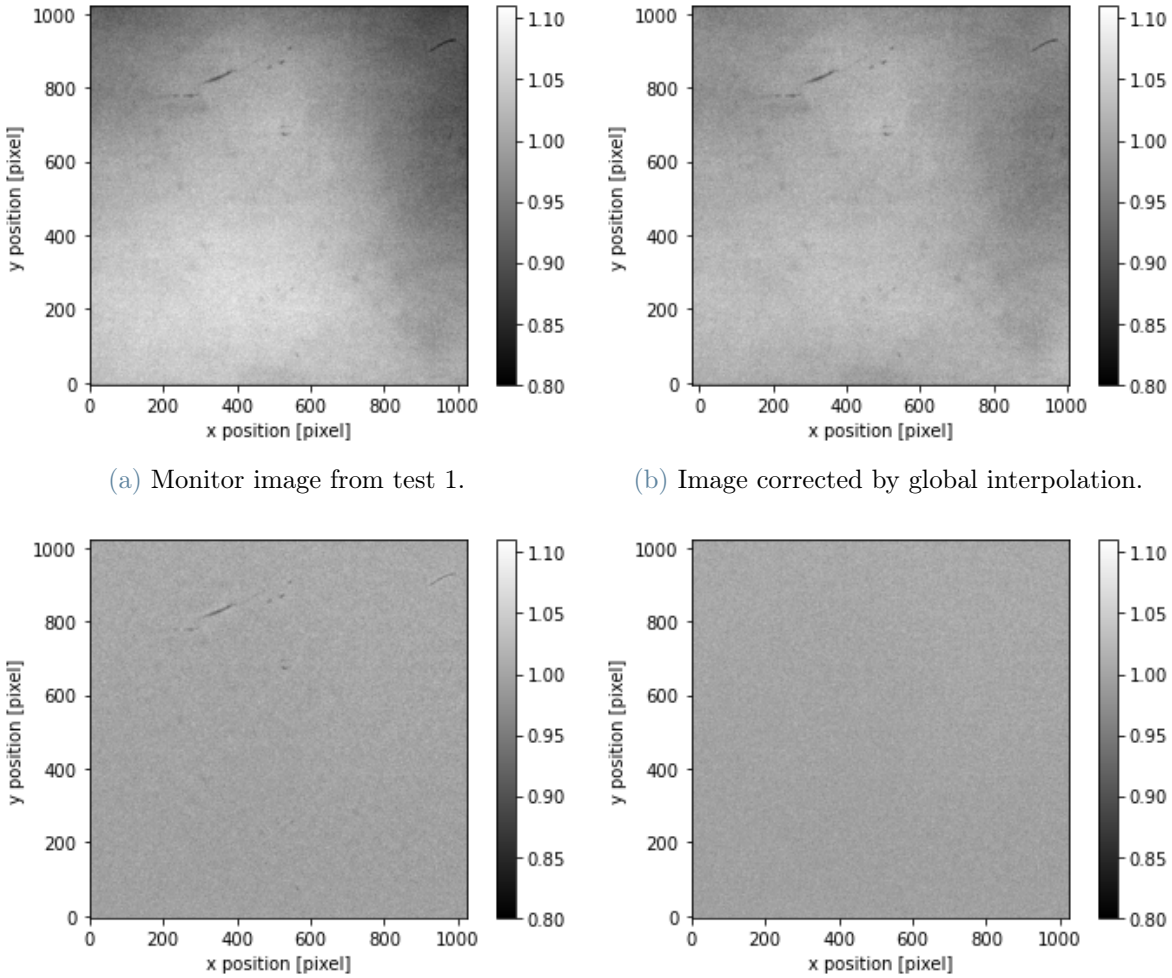
In order to build the correction matrix for the acquired images, different attempts has been tried, and the results are presented in figure 4.3, where one actual monitor image (from the test 1) and the three correction methods have been tried.

- Firstly, it has been used a global interpolation function. The average picture has been interpolated by a polynomial surface, a matrix has been rebuilt from the polynomial and the acquired images has been divided by this matrix.
  This correction method manages to solve the bigger non-uniformities, but has more difficulties to correct the small ones and the screen scratches.

- Secondly, many local interpolation functions has been used. A different polynomial function has been used to interpolate each row and each column of the average image. From the 1024 interpolating functions, a matrix has been rebuilt and the acquired images have been divided by it.
  This correction method manages to solve both big and small non-uniformities, but it still cannot correct the scratches on the wall.

- Finally, the average picture itself has been used. The acquired images have been simply divided by the average image of the monitor.
  This correction method, in addition to be the least time consuming for the program, manages to solve big and small non-uniformities but also the scratches on the screen.

The algorithms used for the correction matrices can be found at B.2.



(a) Monitor image from test 1.

(b) Image corrected by global interpolation.

(c) Image corrected by local interpolation.

(d) Image corrected by the average picture.

Figure 4.3: Different strategies for monitor non-uniformity correction.

As clearly shown by figure 4.3, the correction that can better compensate the monitor irregularities is the one obtained by the average of the monitor images.

4.2.2. BOS test case

The correction matrix has then been used on four different BOS analyses, in order to see if the modification introduced to the acquired image would influence the results of a BOS analysis, and how.

The parameter chosen for the evaluation is the mean pixel error, as explained in chapter 3: by comparing the error in the BOS with the unmodified background to the error obtained in the BOS with corrected background, it is possible to understand whether or not the background correction has a positive impact on the BOS.

The comparison has been made with four different backgrounds, to grant statistical reliability: the first background has Gaussian shaped particles with a diameter of four pixels, the second has Truncated particles of the same diameter, the third has Gaussian particles with a diameter of six pixels, and the last one has Truncated particles with the same diameter.
The results are the ones shown in table 4.3.

	Test 1	**Test 2**	**Test 3**	**Test 4**
Particle diameter [pixels]	4	4	6	6
Particle shape type	Gaussian	Truncated	Gaussian	Truncated
Error before correction [mpd]	0.0546	0.0524	0.0485	0.0577
Error after correction [mpd]	0.0546	0.0525	0.0486	0.0578
Error variation	+0.04%	+0.11%	+0.23%	+0.07%

Table 4.3: Comparison between background corrected and uncorrected BOS.

4.3. Conclusions

As it can be seen in figure 4.1, the monitor presents evident scratches and wide non-uniform areas.
The non-uniformity of the monitor has been studied and three correction matrices has been proposed. From the three correction methods, the one obtained by using the average image of the background has been identified as the better performing as it best succeeds in uniforming the images.
In the end, a comparison has been made between background-corrected and background-uncorrected BOS, and the results are shown in table 4.3.
The variation in the BOS analysis introduced by correcting the monitor is minimal for every analysis taken into account, and it seems to have a negative impact, by slightly increasing the mean pixel error.
The influence of the background correction on the BOS has been subdivided into two cases:

- If the background correction is spread along a large are of pixels, as in the cases of lighter and darker areas, it should not influence the analysis, as it affects in the same way both the particle and the near background, keeping the signal to noise ratio the same as before.

- For harsher background corrections, concentrated in a small amount of pixels as the cases of the screen scratches, if the scratch covers (partially or totally) a particle, even if the scratch is mathematically corrected, the particle is not reconstructed by the correction; if the scratch covers only an area of the black background, it is seen by the PIV algorithm as a signal to cross-correlate (just like a particle), and if it is removed this results only in a loss of information for the cross-correlation algorithm.

For these reasons, the differences between corrected and uncorrected background BOS are very small, and negative since the mathematical removal of scratches from the screen causes a loss of information in the cross-correlation.
In view of the above, it has been chosen not to correct the monitor non-uniformity in the following BOS and EBOS analyses.

5 | Comparison between Dantec DynamicStudio and Openpiv

In this chapter two different PIV software will be analysed, and their performances will be compared.
The main parameter that has been taken into account is the accuracy of the software, that has been evaluated the same way as in chapter 3: a synthetic displacement is imposed to the background; the images are displayed by the monitor and acquired by the camera; the same images are then analysed by the two software and the results are compared to the theoretical displacement field.
Other parameters that have been considered in the comparison are the speed of the analysis, the adaptivity to different situations and the ease of use.
The first software is OpenPIV, the one that has been used in chapeter 3, the second one is Dantec DynamicStudio.
Since Dantec is not a Python-based software, the results obtained with Dantec have been exported as text files, loaded on Python and then compared to the theoretical displacement field, as happened for the results obtained with OpenPIV.

5.1. OpenPIV

OpenPIV is a effort of scientists to deliver a tool for the analysis of PIV images using state-of-the-art algorithms [14].
OpenPIV is available on Matlab, Python and $C++$, even though the main developing efforts are actually aimed towards the Python version. As previously mentioned, as an open source code it is possible to get easy access to the code, study the functioning of the program and, if necessary, modify the desired functions. The numerous customization possibilities allow OpenPIV to adapt particularly well to the BOS needs and to speed up many processes if small modifications are introduced.
However, the absence of a graphic interface makes it more difficult to pre-process the images, and the post processing is virtually absent; for these reasons, the OpenPIV user

needs some programming skills to fully exploit the software, and the not experienced used may find some difficulties the first times.
The software is composed by a setting script, a main function, and many auxiliary functions. The first section of the settings is dedicated to the cross-correlation Interrogation Areas (cIA).

```
# select the number of PIV passes
settings.num_iterations = 4
# add the interroagtion window size for each pass
# if longer than n iteration the rest is ignored
settings.windowsizes = (64, 32, 16, 8, 4)
# The overlap of the interroagtion window for each pass.
settings.overlap = (32, 16, 8, 0, 0)
```

Figure 5.1: OpenPIV cIA configuration.

The first parameter to be chosen is the number of iterations, that is the times the cross-correlation algorithm is evaluated. Each time, the dimension of the cIA is different, and can be arbitrary chosen by the command *settings.windowsizes*, the shape is always a square. Usually the cIA decrease with a factor of two, but it is possible to choose any dimensions. There is not a minimum number for the cIA size, but smaller areas will give weaker cross-correlation signal and less accurate results.
The following command, *settings.overlap*, determines the number of pixel of neighbours cIA that overlap at each iteration. In the example shown in figure 5.1 the overlapping is set to 50% for the first 3 iterations and to 0 for the last one. The presence of some overlapping simplifies the adaptivity process, at the cost of increasing the computational time.

The second part of the settings is dedicated to the validation of the vectors.

```
'Validation Parameters'
# The validation is done at each iteration based on three filters.
# The first filter is based on the min/max ranges.
settings.MinMax_U_disp = (-30, 30)
settings.MinMax_V_disp = (-30, 30)
# The second filter is based on the global STD threshold
settings.std_threshold = 9  # threshold of the std validation
# The third filter is the median test (not normalized at the moment)
settings.median_threshold = 5  # threshold of the median validation
# On the last iteration, an additional validation can be done based on the S/N.
settings.median_size=3 #defines the size of the local median
'Validation based on the signal to noise ratio'
settings.sig2noise_method = 'peak2mean'
settings.sig2noise_threshold = 1.5
```

Figure 5.2: OpenPIV validation configuration.

The validation is carried out by 4 filters, of which the first three are applied at each iteration, and the fourth is applied only to the last iteration.

1. In the first filter, the displacement values that exceed an imposed threshold, defined as *settings.MinMax_ U/V_ disp*, are rejected.

2. The second filter rejects the vectors that differ too much from the mean of the analysis. The mean value and the standard deviation of all the vectors are calculated, and vectors bigger that the *std_ threshold* are rejected.

3. The third filter is for local outliers: the median value for a neighborhood around each vector is evaluated, and vectors too different from the median are rejected.

4. The last filter can be set as *peak2mean* or as *peak2peak*. In the first case, the peak found in the cross correlation needs to be bigger that the mean by a factor of *sig2noise_ threshold*, in the second case the reference number is not the mean but the second biggest peak of the cross-correlation.

These filters allow to identify all the values that are considered non correct, and mark them as non valid. The invalid vectors are substituted in all the iterations except for the last one, when the choice is left to the user.
The outliers are substituted by a polynomial interpolation function, with parameters chosen by the user.

5.2. Dantec

DynamicStudio is the main software package for image acquisition and analysis in for example the PIV, LIF, LII, and Particle Sizing areas. It contains tools for configuration,

acquisition, analysis, post-processing of acquired data. [11]
Dantec is a state of the art software when it comes to image acquisition and analysis. It has many different functions suited for various analyses, and presents a easy to learn tool for the acquisition system. The graphic interface and the user manual make it simple for the user to easily learn the key concepts and to use the program.
However, the inaccessibility of the code and the design of the software make some easy tasks more difficult that they should be. As an example, the images to elaborate must have a specific format and name, and some parameters cannot be chosen.
The first difference from OpenPIV is the acquisition tool, that allows the configuration of the camera and the connection to the NI acquisition system. This tool has been used for every analysis, even the ones not carried out by Dantec.
The second useful tool is the Region Of Interest (ROI) cutting. This function allows the selection of a region of the image to be processed. Even if OpenPIV has a similar tool, the ROI parameters are here chosen graphically and not mathematically, speeding up the ROI cutting by far.

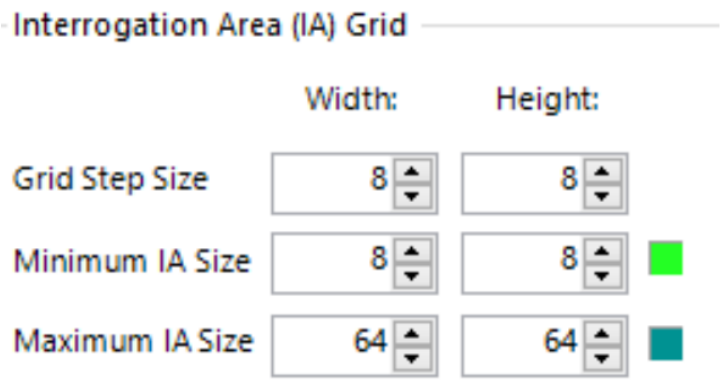


Figure 5.3: Dantec cIA configuration.

The first parameters to be chosen are relative to the cIA: the *Minimum IA Size* is the last cIA dimension, where the *maximum IA Size* represents the initial cIA dimension. Dantec does not allow to choose how many iterations are made, but if possible the cIA areas are divided by 4 at each iteration; for example, the parameters used in figure 5.3 mean that the iterations will be four, with sizes of 64, 32, 16 and 8 pixels. The minimum cIA dimension is 8x8 pixels.
The *Grid Step Size* imposes the distance between two adjacent cIA centers; this number is not adaptive and it is the same for each iteration. This means that the overlapping progressively decreases as the analysis goes on. In the example shown in figure 5.3, the overlapping of two near cIA will be the 87.5% for the first iteration, the 75% for the

second, the 50% for the third and there will be no overlapping at the last iteration.

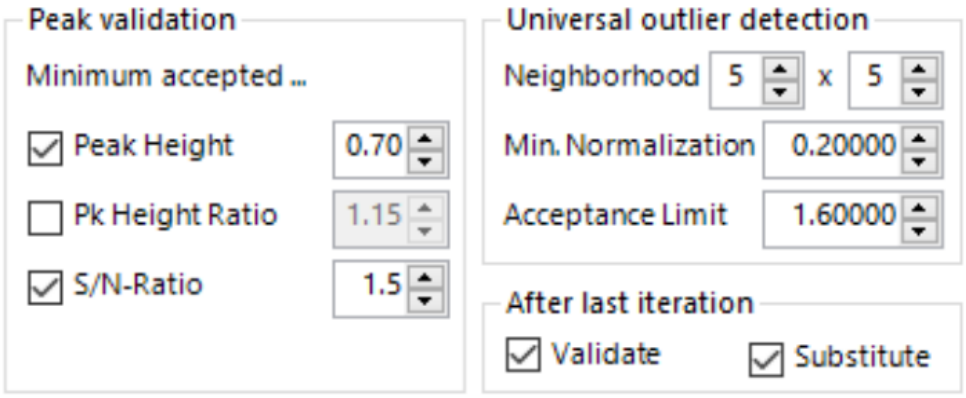


Figure 5.4: Dantec validation configuration.

The first validation method *Peak Height*, checks the absolute value of the cross correlation peak, rejecting signals too weak. The *Pk Height ratio* compares the height of a peak to the height of the second tallest peak in the same cIA, the ratio should be higher than the imposed value. The *S/N-Ratio* is the signal to noise ratio filter: it rejects peaks that are not above the noise signal with a ratio smaller than the imposed one. These filters can be turned on and off at will, and the values modified.
The *Universal outlier detection* is instead a filters based on the vectors that passed the peak validation: it rejects vectors too different from the neighborhood, according to the imposed parameters. The outliers can then be substituted or not by a interpolation function.

5.3. Analysis parameters

For this analysis, different synthetic displacement fields are taken into account, in order to test more exhaustively the accuracy of the software.
The fields used are all the ones mentioned in chapter 3.1.3: The Diagonal field has been used as a standard field, the Cross field has been used to measure the accuracy near the zero order discontinuities in the displacement field, the sinusoidal field has been used to measure the accuracy for non-polynomial displacement functions, and the Parabolic and Linear fields have been used for the discontinuities in, respectively, the second order and first order derivatives of the displacement field.
The particle chosen is the Truncated one, with a diameter of 4 pixels (on the sensor), the density is 1 pixel per bIA of size 8x8 and the margins are set at $mrg = 36\%$. The camera and lens parameters are the same that has been used for the particle optimization

analysis, or:

Parameter	Measure
Lens focal length	$50mm$
$F\#$	$11mm$
Camera shutter speed	$30000\mu s$
Background distance	$1.49m$

Table 5.1: Camera and lens parameters for chapter 5 analyses.

For each synthetic displacement field, four different analysis have been made in order to avoid random fluctuations. For the Diagonal, cross and Sinusoidal fields the maximum displacement is set to 3 pixels on the sensor for all the analyses; for the Parabolic and Linear fields the maximum displacement increases in the four analyses from one pixel to ten, in order to evaluate the software behaviour in as many situations as possible.

5.3.1. PIV settings

In order to compare the two programs at the same conditions, the PIV parameters should be the same. Unfortunately, the different algorithms for the filters and the different functions for the substitution of the outliers make it impossible to have the same exact conditions, and compromises had to be done.
Being OpenPIV the most customizable, most of the times the parameters are firstly set on Dantec and then imposed on OpenPIV.
The table 5.2 shows the parameters that have been used for this analysis.
The cIA sizes are the same in both the analyses: they are squares with the side decreasing with a factor of 2. The overlapping used is the only one available on Dantec, with a large overlapping for the first iterations and an absent overlapping for the last one.
The signal to noise filter is the same in the algorithms, but it is the only filter that can be set equal in the two software: the others use different algorithms. Hence, and they are removed, or kept as similar as possible by modifying the available parameters.
The substitution has been turned on for both programs, with same neighborhoods but again different functions.

Adaptivity	OpenPIV	Dantec
N° iterations	4	4
cIA dimension [pixels]	64 - 32 - 16 - 8	64 - 32 - 16 - 8
Overlapping	87.5% - 75% - 50% - 0%	87.5% - 75% - 50% - 0%
Validation	**OpenPIV**	**Dantec**
Peak height	none	present
Peak to second peak	none	none
Peak to noise	1.5	1.5
Global outliers detection	present	none
Local outliers detection	present	present
Substitution	**OpenPIV**	**Dantec**
Substitute	True	True
Interpolating neighborhood	±5 pixels	±5 pixels

Table 5.2: PIV parameters used for chapter 5 analyses.

5.4. Results

In this section the results obtained for the analysis will be shown and commented.
The accuracy of the software varies depending on the synthetic displacement field used, so a particular care will be dedicated to the behaviour of the programs when facing a specific displacement field.
Eventually, the pros and cons of each software will be drawn up and some conclusions about the program choice will be made.
The table 5.3 shows the mean pixel error made by the two software (as explained in chapter 3.2.2) for the different displacement fields. For the Diagonal, Cross and Sinusoidal displacement fields, the result is the mean value of four different analyses with identical parameters; for Parabolic and Linear displacement fields, the results shown refer to a single analysis.
To better understand the behaviour of the two software, table 5.4 shows the difference in the error evaluation for the Dantec analysis with respect to the OpenPIV one: a positive number means a greater error in Dantec, and a negative number means a greater error in OpenPIV.

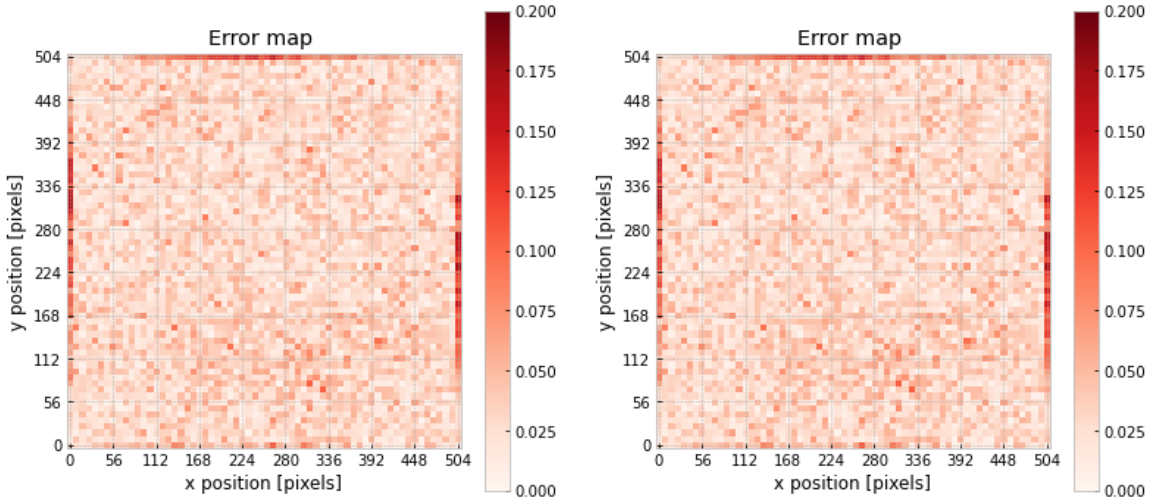
OpenPIV	1 pixel	3 pixels	5 pixels	10 pixels
Diagonal		0.0529		
Cross		0.1140		
Sinusoidal		0.0325		
Parabolic	0.0327	0.0435	0.0435	0.0902
Linear	0.0298	0.0318	0.0384	0.0934
Dantec	1 pixel	3 pixels	5 pixels	10 pixels
Diagonal		0.0495		
Cross		0.0902		
Sinusoidal		0.0390		
Parabolic	0.0399	0.0610	0.0706	0.1221
Linear	0.0295	0.0362	0.0496	0.1322

Table 5.3: Mean error e_{tot} [mpd] for different software.

Error variation with Dantec	1 pixel	3 pixels	5 pixels	10 pixels
Diagonal		−6.44%		
Cross		−20.84%		
Sinusoidal		+20.15%		
Parabolic	+22.01%	+40.38%	+62.07%	+35.39%
Linear	−1.01%	+13.90%	+29.07%	+41.54%

Table 5.4: Percentage difference between Dantec and OpenPIV mean error, (open PIV took as reference).

The error map is different for each displacement field (5.6, 5.7, 5.8, 5.10, 5.9), but there is a region that in many analyses shows a high error: the border. Indeed, in many analyses the cIA that are the nearest to the image border present large errors, as clearly visible from image 5.5. This effect is more present in the displacement fields analysed by Dantec, but it also happens for OpenPIV.
This error cannot be due to the fact that particles exit or enter the borders, fact that would cause the cross-correlation algorithm not to find a particle in one of the two images, because all the displacement fields present null displacement perpendicular to the borders near them.



(a) OpenPIV. (b) Dantec.

Figure 5.5: Error maps [mpd] for the Sinusoidal displacement field.

Since the mean pixel error appears to be heavily influenced by the behaviour of the cross correlation algorithms near the borders, a second error evaluation has been made, neglecting the value for the error at the border of the map.
These results are shown in tables 5.5 and 5.6.

OpenPIV	1 pixel	3 pixels	5 pixels	10 pixels
Diagonal		0.0518		
Cross		0.1010		
Sinusoidal		0.0299		
Parabolic	0.0307	0.0406	0.0605	0.0841
Linear	0.0301	0.0314	0.0375	0.0923
Dantec	1 pixel	3 pixels	5 pixels	10 pixels
Diagonal		0.0450		
Cross		0.0832		
Sinusoidal		0.0295		
Parabolic	0.0321	0.0422	0.0707	0.0872
Linear	0.0267	0.0284	0.0368	0.1007

Table 5.5: Mean error e_{tot} [mpd] (without outer border) for different software.

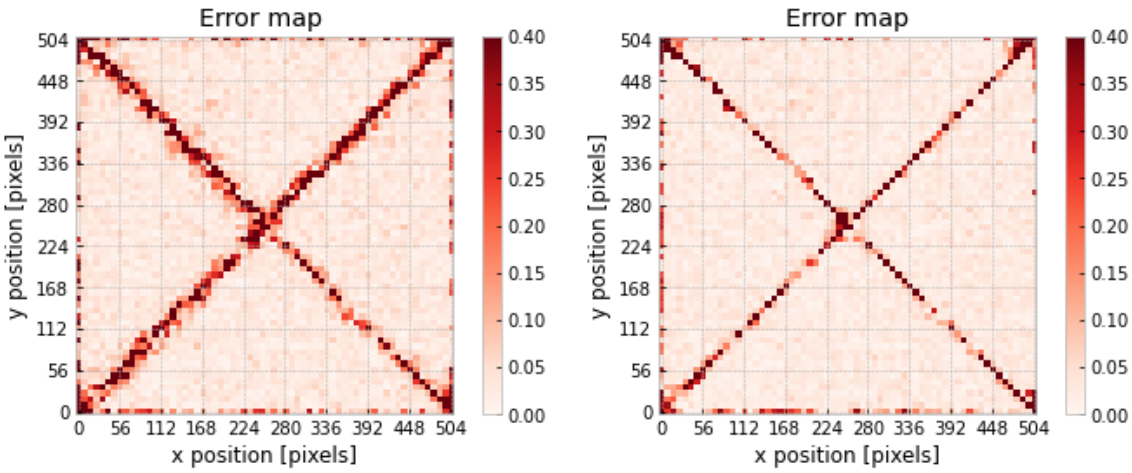
The mean error calculated without considering the border decreases for every analysis, for both the software and all the five synthetic displacement fields. The main effect, however, is more visible for Dantec, and in particular for the displacement fields with a lower mean error.

Error variation with Dantec	1 pixel	3 pixels	5 pixels	10 pixels
Diagonal		−13.23%		
Cross		−21.33%		
Sinusoidal		−1.46%		
Parabolic	+4.21%	+3.86%	+14.31%	+16.52%
Linear	−15.14%	−10.67%	−1.93%	+8.28%

Table 5.6: Percentage difference between Dantec and OpenPIV mean error without outer borders, (open PIV took as reference)

5.4.1. Influence of synthetic displacement field

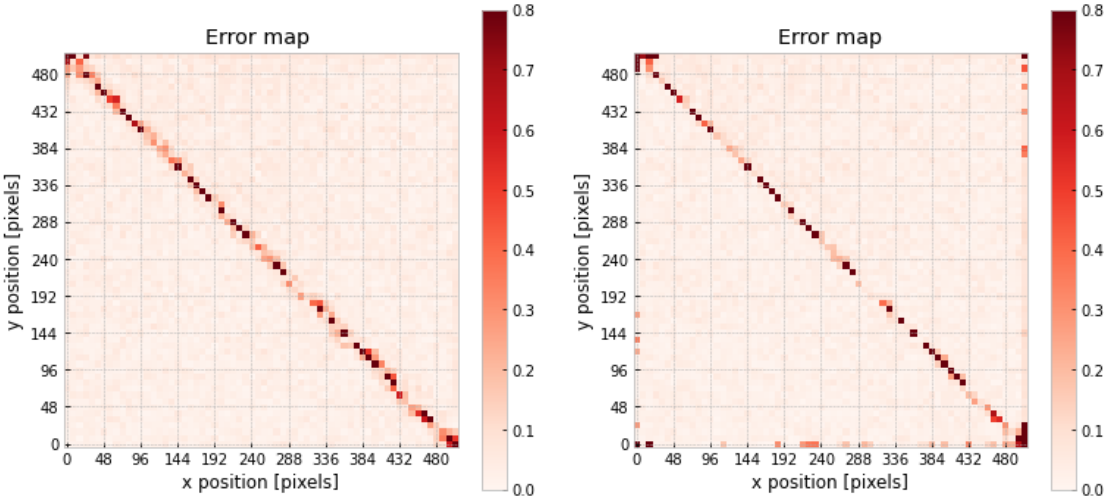
Tables 5.4 and 5.6 show that the behaviour of the two software is significantly different depending on the synthetic displacement field that is considered.
For Diagonal and Cross fields, Dantec seems to be more accurate, and for Parabolic and Linear fields OpenPIV seems to have a smaller error, mainly for large maximum displacements. For the Sinusoidal field, in the end, the response is not clear, as the results change if the border is considered or not.
The greater error of OpenPIV for the Cross and the Diagonal displacement fields can be explained by the zero order discontinuity areas.



(a) OpenPIV. (b) Dantec.

Figure 5.6: Error maps [mpd] for the Cross displacement field.

As it can be seen in figure 5.6, the error for this analysis is mainly located along the zero order discontinuities in the displacement field, but the OpenPIV error seems to be spread to the neighbour areas much more that the Dantec one.
The same effect happens also for the Diagonal field, even if it is less evident.

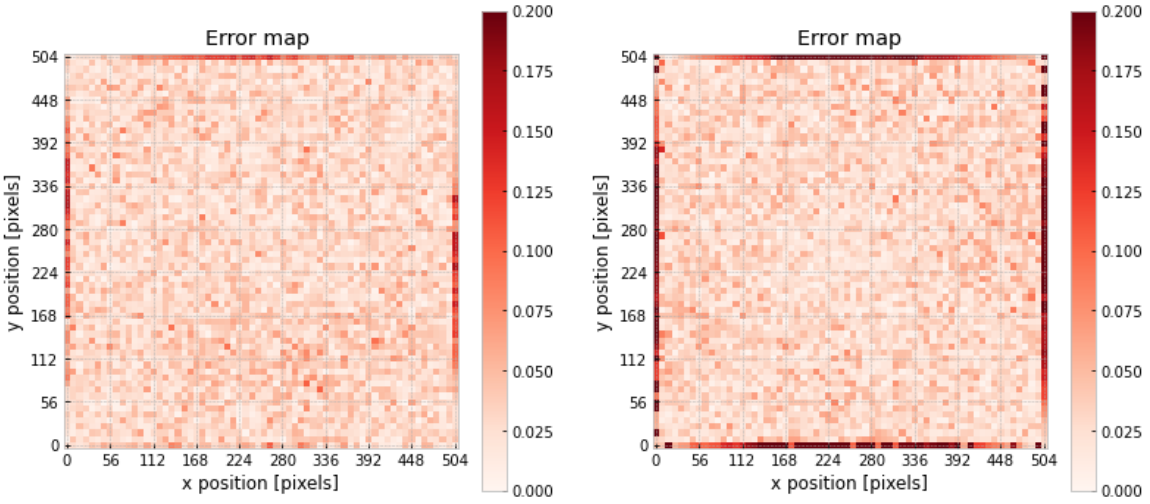


(a) OpenPIV. (b) Dantec.

Figure 5.7: Error maps [mpd] for the Diagonal displacement field.

For the Sinusoidal field, the error maps seems to be very similar if the border is excluded. The maps are homogeneous and it doesn't seem to be a region where the error is con-

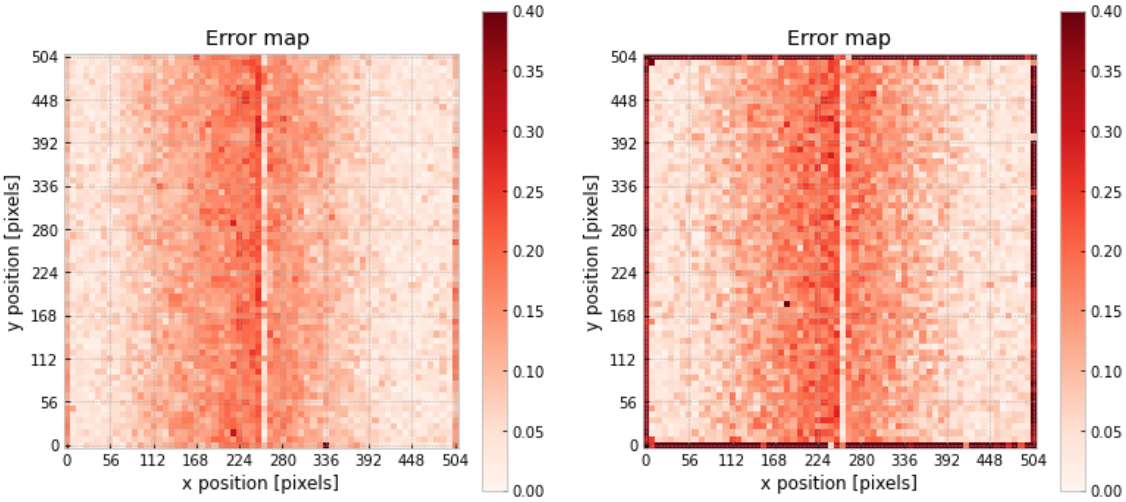
centrated. This is probably due to the fact that the Sinusoidal map doesn't present any discontinuity of any order.



(a) OpenPIV. (b) Dantec.

Figure 5.8: Error maps [mpd] for the Sinusoidal displacement field.

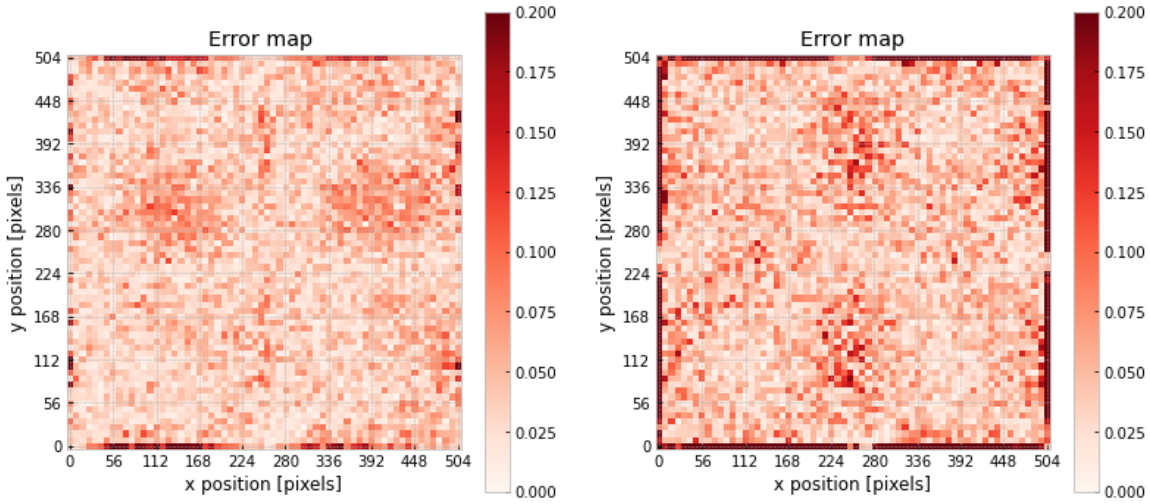
For the Linear synthetic displacement field, for both the software the error is mainly located in the center of the map, where the displacements are higher (figure 5.9).
One possibility is that both the software have more difficulties in evaluating higher displacements, but this is refuted by the fact that in the sinusoidal field there is not a higher error region near the center, and in the Diagonal field there is not a higher error region to the right of the diagonal, where displacements are maximums.
A second possibility is that the programs have more difficulties near first order discontinuities, like it happens along the vertical line at the center of the map. It is possible that those difficulties have repercussions also at some distance from the actual discontinuity.
The reason for the white vertical line at the center of the error map, is that the symmetry in the derivative of the displacement field probably helps the cross correlation algorithm to find the exact solution.



(a) OpenPIV. (b) Dantec.

Figure 5.9: Error maps [mpd] for the Linear displacement field.

For the Parabolic displacement field (figure 5.10), the results present some similar aspects to the ones obtained with the Linear field.
The error is concentrated in some areas that are close to the second order irregularities, which are located along the horizontal and vertical axes that pass through the center of the image, and near the borders. The results are however not clear and no conclusions can be drown from this displacement field, apart from the fact that OpenPIV is more accurate for this kind on analysis.



(a) OpenPIV. (b) Dantec.

Figure 5.10: Error maps [mpd] for the Parabolic displacement field.

5.4.2. Pros and cons of each software

From the analyses that have been made, some differences can be found between the two software.

Dantec seems to be more accurate when zero order discontinuities are present in the displacement field, as it happened for the Diagonal and the Cross synthetic displacement fields. In particular, both programs have difficulties in finding the correct displacement for the cIA that are located over the discontinuity, but for OpenPIV the error is also spread to the nearby cIA.
For each analysis, Dantec presents a higher error than OpenPIV at the boundaries; the reasons for this behaviour are however not clear, and they could be investigated in the future.
For displacement fields with discontinuities of first and second order, Dantec seems to return a higher error than OpenPIV, specially when the maximum displacements are higher.
For displacement fields without any discontinuities, the two software seem to present similar performances.

In Dantec, it is possible to configure the acquisition system, by setting the camera parameters and acquire images; on OpenPIV this is not possible. Also, the pre-processing tools (the region of interest cutting in particular) on Dantec are more intuitive and easy to use, even though they are also present on OpenPIV.

The post-processing is easier on OpenPIV, since the results are already on Python and they are more ready to use. On Dantec, the post-processing is designed for the PIV, and the mathematical results need to be exported and elaborated on a different software.

Dantec is more fast that OpenPIV in analysing the images, even though on OpenPIV it is more simple to load the images to acquire and to save the results.

The configuration parameters are similar in terms of validation and substitution, but very different in terms of adaptivity: Dantec allows minimum cIA of 8x8 pixels, when OpenPIV has no theoretical limits. This allows OpenPIV to use smaller and denser particles, cross-correlate the images with very refined grids and have a higher resolution. This as-

pect is very crucial for the study of phenomenons that take place in small spaces, or that have rapid variations of density. OpenPIV also allows to choose the dimension of the cIA and its overlapping for each iteration.

The following list summarizes the main differences between the software.

- **Resolution**: Dantec is more accurate near zero order discontinuities, OpenPIV is more accurate near higher order discontinuities and image borders. Overall accuracy seems comparable.
- **Completeness of use**: Dantec is necessary for the acquisition configuration, a tool absent in OpenPIV. The results of both programs need to be post-processed elsewhere.
- **Ease of use**: Dantec is more user friendly, even though after some experience with OpenPIV it ends up to be more straightforward to use.
- **Customization**: OpenPIV allows to decrease indefinitely the cIA, allowing better resolutions. OpenPIV is generally more customisable.

Since the accuracy of the two software is similar, and as a result of the better customization in OpenPIV, for the future analyses Dantec has been used for the acquisition and pre-processing of the images, and OpenPIV for their analysis and post-processing.

6 | EBOS optimization

The aim of this chapter is to study the Enhanced Background Oriented Schlieren (EBOS), to propose solutions for the background alteration, and to analyse which of the proposed solutions gives the best results when compared to the BOS.
Firstly, a comparison of the studies about the EBOS that are present in literature will be carried out, and the pros and the cons of the proposed solutions found in literature will be analysed. Then, two solutions for the background alterations will be presented, discussed and tested over an experimental case.
For each solution, a BOS and an EBOS analysis will be applied to the same test case, to see whether the implementation of the EBOS results in a better accuracy for the technique. The two background alteration techniques will be eventually compared and the results will be discussed.

6.1. EBOS techniques

The Enhanced Background Oriented Schlieren was conceived in 2019 by *Cozzi* and *Göttlich* [4].
The main principle of this technique is to acquire n different variations of a background, to compare them to the same Schlieren acquisition and to average the n results found. This should allow to obtain more accurate results without an increase in the invasiveness of the technique.
The main aspect to be studied in the EBOS technique is the background variation, that is how to create the n different versions of the background.

The first solution, proposed by *Cozzi* and *Göttlich* [4], is to rigidly translate the background by a small quantity at each iteration. The vector sum of all the displacements should be null, in order to not have a bias error when the mean of all the n displacement fields is computed.
However, the use of a printed background and not a monitor-displayed one, means that the background variations are obtained by the actual displacement of the background.

This introduction of moving parts in the setup significantly increases the complexity of the technique, which finds in the simplicity of implementation one of its main strengths.

In Casero's thesis work [3], the use of a monitor-displayed background allows to rigidly translate the background without increasing the complexity of the setup, by shifting the displayed image on the monitor. In this work, though, Casero did not use a synthetic displacement field, and instead of the accuracy of the technique, the standard deviation is used as main parameter of the quality of a measure. The mean displacement of the analysis is compared to the single displacements, without taking into account what is the theoretical displacement of the background.
This leads to results that show a decrease in the STD for the EBOS with respect to the BOS if a small number of background images is considered, and a increase for the STD when a greater number of backgrounds is considered, but give no information about the accuracy of the technique.

In Sosio's work, [25], different strategies for the background variation are proposed: in the *Brownian motion* each particle is randomly displaced at each EBOS image by a probability density function; in the *Rigid displacements*, the whole background is rigidly translated by a random value, defined by the same probability density function.
The BOS and the different EBOS are applied to the same synthetic displacement field and then compared. The results are once again uncertain, as the EBOS does not seem to introduce any improvement, except for when the particle density is very low.
This is probably due to the fact that the random displacements used for the variation of the background have a mean different from zero, and thus the EBOS introduces a bias error in the analysis. The more the measure is accurate and the error is small, the more this bias error stands out; far analyses with a higher mean pixel error, the bias error is less evident and the benefits of the EBOS can be better appreciated. Furthermore, for a density greater than $Ni = 1$, the presence of particles with different displacements in the same cIA causes a deformation in the cross-correlation peak.

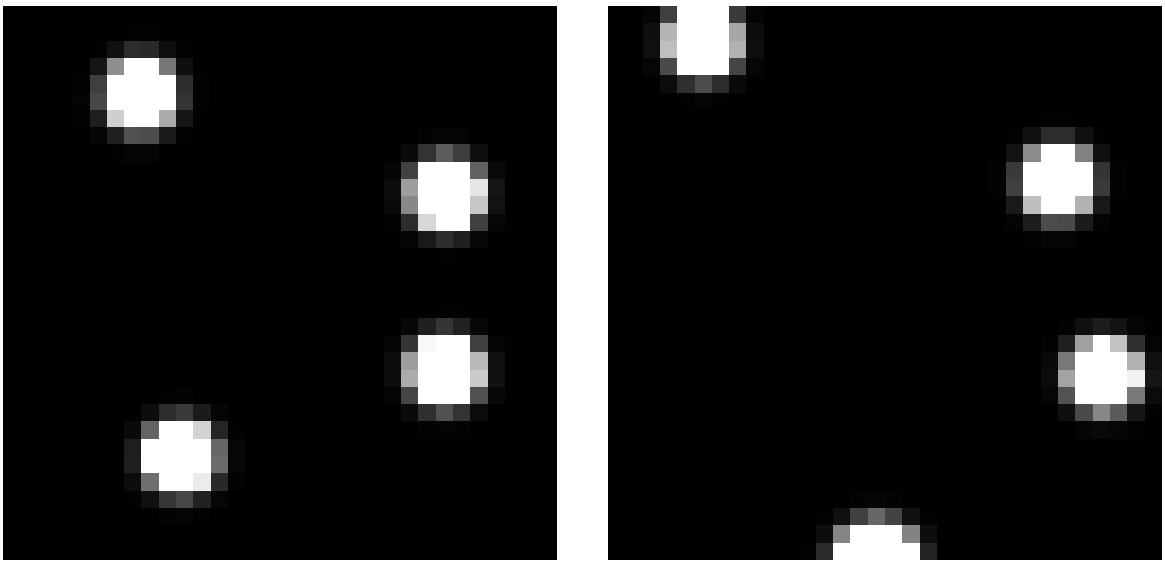
In this thesis work, the Brownian motion used by Sosio is used as a reference for the EBOS, and an improved version for the rigid displacements is proposed. The two techniques are applied to a synthetic displacement field and then compared.

6.1.1. Brownian motion

The Brownian motion is the random thermal motion of a particle suspended in a fluid. In Sosio's work this term has been used by extension to indicate a certain type of background variation. If every particle is randomly displaced in a neighborhood of the initial position, at every EBOS image, the effect of all the EBOS images is similar to a thermal vibration of the particles.

Mathematically, a probability density function (PDF) is defined accordingly to triangular distribution imposed by the user: the greater is the with of the triangular function, the larger the maximum motion of the particles. Then, for each particle, a value for the vertical translation and a value for the horizontal translation are randomly selected from the PDF.

Similarly to how it happened for the synthetic displacements in chapter 3.1.3, the centers of the particles are translated by the values found, and the whole image is reconstructed from the particle centers. If the random displacement is not an integer of a monitor pixel, the particle will be deformed by the translation, since the particle creation function, like "Truncated", defines each pixel intensity by the distance from the particle center with a continuous function.
However, this effect is minimal, and as it will be demonstrated at the end of this chapter, it has no influence in the measure.



(a) Reference image. (b) Variated background.

Figure 6.1: Detail of EBOS image with Brownian motion.

Figure 6.1 is a detail of the same section of two different pictures: the first one is the standard background, the second one is on the the n variations of the background which compose the EBOS set of images. The Brownian motion is a random displacement of the particle centers, thus in the second picture the particles are shifted in different directions and with different module.

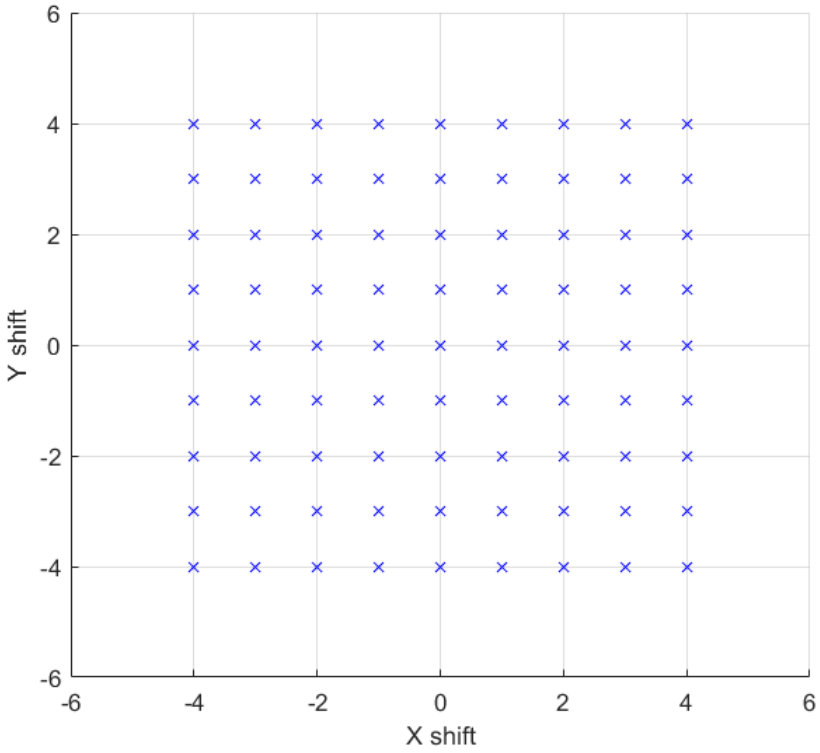
6.1.2. Rigid translations

The main advantage of the rigid translations of the background is that the mean of all the displacements is equal to zero. In order to do so, specific patterns for the translations are used, which usually involve symmetries along both x and y axes, and a number of figures which is a multiple of four.
The first approach that have been used in this work is similar to the one used by Casero [3]: in order not to distort the particle shapes, the translations are multiples of a monitor pixel, and all the possible translations have been used, as shown in table 6.1.

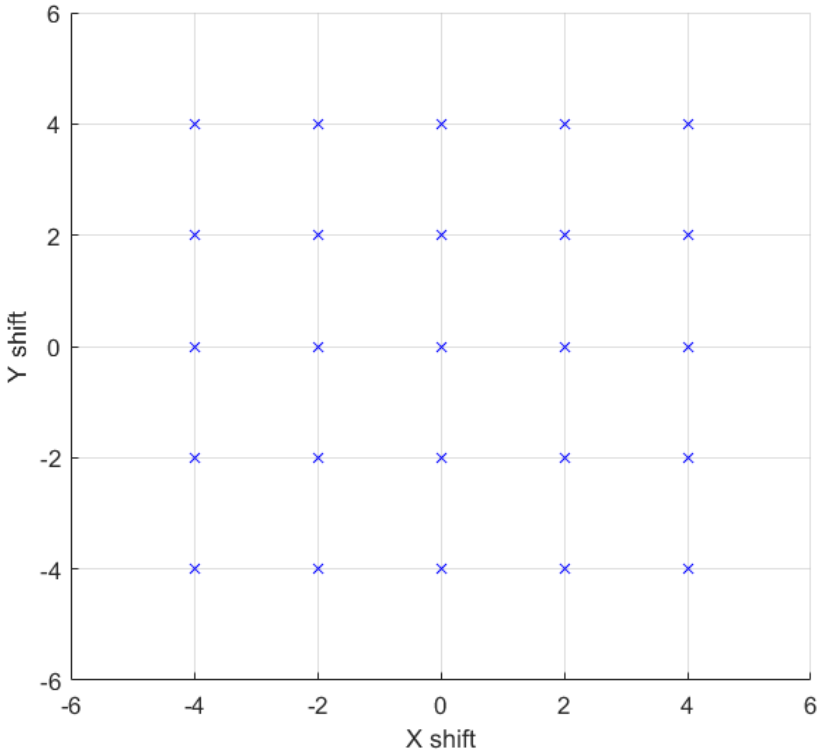
Shift	$\Delta x = -1$	$\Delta x = 0$	$\Delta x = +1$
$\Delta y = +1$	EBOS #6	EBOS #7	EBOS #8
$\Delta y = 0$	EBOS #5	Ref. Image	EBOS #1
$\Delta x = -1$	EBOS #4	EBOS #3	EBOS #2

Table 6.1: Scheme of the first 8 translated images of the "Rigid Translation" EBOS.

The same scheme can be applied for images from the 9^{th} to the 24^{th}, with a maximum displacement of ± 2 pixels, and so on.
For a maximum displacement of ± 4 pixels, the 81 images that compose the EBOS backgrounds (1 reference image and 80 shifted images), compose the first pattern shown in figure 6.2: the "x" stands for a translated background, the "0" for the reference image. This type of EBOS will be called "complete square displacements", since the 81 images cover the total of the possible displacements making a square image.
A different approach is to consider only some of the possible displacements; the displacements obtained this way will be called "partial square displacements", since only a part of the displacements are used. The scheme for a maximum displacement of four pixels is the second one shown in figure 6.2



(a) Complete square displacements.



(b) Partial square displacements.

Figure 6.2: Schemes for squared Rigid Translation EBOS.

The last approach used is to shift the backgrounds with a constant absolute value and a different angle. The first 8 images will be shifted by ±1 pixel with an angle from 0° to 315° every 45°, the images from 9 to 24 will be shifted by ±2 pixels with an angle from 0° to 337.5° every 22.5° and so on, as showed in figure 6.3.

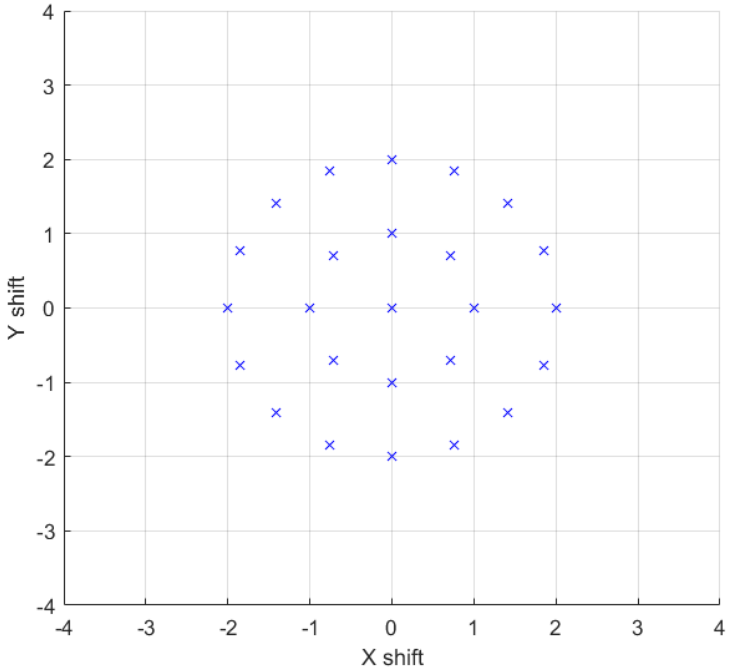


Figure 6.3: Scheme for circular Rigid Translation EBOS.

The result is a circular scheme around the reference image instead of a squared one, and these EBOS background will be called "circular displacements".
This type of backgrounds presents a fractional shifting along the axes, and therefore the particles will be deformed in the EBOS images, as it happened for the Brownian motion.
The algorithm used for the rigid translations can be found at A.5

6.2. Analysis parameters

For this analysis, a single synthetic displacement field has been used, which is the "Diagonal", as it has been considered the most complete.
The background parameters are the ones used also in chapter 5: a "Truncated" particle with a diameter of 3 pixels (on the camera sensor) and a density of one particle every bIA of 8x8 pixels, the margins has been imposed to $mrg = 36\%$.
The camera and lens parameters are equivalent to the ones used in chapters from 3 to 5,

and they are reported in table 5.1. The software that have been used is OpenPIV and the PIV parameters are the ones shown in table 6.2

Adaptivity	Type 1	Type 2
N° iterations	4	5
cIA dimension [pixels]	64 - 32 - 16 - 8	64 - 32 - 16 - 8 - 4
Overlapping	50% - 50% - 50% - 0%	50% - 50% - 50% - 50% - 0%
Validation		
Signal to noise threshold	1.5	1.5
Global STD threshold	9	9
Local median threshold	5	5
Substitution		
Substitute	True	True
Interpolating neighborhood	±3 pixels	±3 pixels

Table 6.2: PIV parameters used for chapter 6 analysis.

Two different types of parameters have been used in OpenPIV for this analysis. In the first case, the cIA dimension is the same of the bIA. This means that for each interrogation area of the cross correlation, a particle is found. In the second type of analysis, the cIA is smaller than the bIA; this means that each cIA has only a fraction of a particle to cross correlate.
Since monitors have practical limits in terms of particle dimensions, due to the dimension of the monitor pixels, the maximum resolution obtainable for the BOS is limited. The aim of this work is also to find out if it is possible to have a resolution finer than the actual dimension of a particle, without a loss in the analysis accuracy, by using the EBOS.

These two different analysis types has been applied to 2 sets of measures:

1. The first analysis is a comparison between BOS, Brownian motion EBOS, and rigid translation EBOS.
   For each EBOS technique 81 backgrounds have been used, and the mean pixel errors of the EBOS to the variation of the number of images has been evaluated.

2. The second analysis is a comparison between the different types of Rigid Translation. Similarly to the first one, the different EBOS techniques have been compared to the BOS in terms of mean pixel error.

6.3. Results

The results obtained are presented in tables 6.3 and 6.4. The column "1" means that only one image is used for the EBOS, that is the same as applying the BOS.
The EBOS results are obtained by applying the mean to all the displacement fields obtained with the n backgrounds, and then by comparing the obtained displacement field to the theoretical one, as it happened for the BOS.
These results will be better discussed and analysed in the following sections.

Number of EBOS images	1	9	25	49	81
Last cIA dimension 8 pixels					
Brownian motion	0.0567	0.1147	0.0853	0.0727	0.0672
Rigid displacements	0.0559	0.0533	0.0514	0.0487	0.0464
Last cIA dimension 4 pixels					
Brownian motion	0.0753	0.0971	0.0801	0.0733	0.0708
Rigid displacements	0.0707	0.0615	0.0575	0.0541	0.0513

Table 6.3: Mean error e_{tot} [pixels] for the different types of EBOS to the number of background images.

Number of EBOS images	1	9	25	49	81
Last cIA dimension 8 pixels					
Complete squared EBOS	0.0559	0.0533	0.0514	0.0487	0.04641
Partial squared EBOS	0.0559	0.0509	0.0458		
Circular EBOS	0.0524	0.0487	0.0475		
Last cIA dimension 4 pixels					
Complete squared EBOS	0.0707	0.0615	0.0575	0.0541	0.0513
Partial squared EBOS	0.0707	0.0579	0.0512		
Circular EBOS	0.0682	0.0585	0.0549		

Table 6.4: Mean error e_{tot} [pixels] for the different types of Rigid Displacement EBOS to the number of background images.

The first data that can be seen in tables 6.3 and 6.4 is that for every analysis, to the increase in the number of EBOS backgrounds corresponds an increase in the technique

accuracy.
This is true for the Brownian motion, where the increase in the total amount of images leads to a smaller bias error (except to the case of a single image, where no bis error is present), but also for the Rigid Displacements, where the mean of the background shifts is equal to zero and there's no bias error.

6.3.1. Comparison between Brownian motion and Rigid Displacements

Since the reference backgrounds are different for each analysis, the BOS error is not constant, and a small difference can bee seen from the Brownian motion and the Rigid displacements at 1 image; for this reason the mean pixel error for the EBOS analyses will be expressed as compared to the BOS error, that is the error of an EBOS with just one background.
As clearly shown in figures 6.4 and 6.5, the Brownian motion EBOS has results significantly worse than the Rigid Displacement EBOS, and for a standard analysis has results also worse than the BOS. This is probably due to the bias error intrinsic to the technique, as when the mean error of the analysis increase (i.e. when the cIA decreases), the bias error is less predominant and the technique has errors comparable to the BOS.
The Rigid Displacement EBOS, on the contrary, has better results than the BOS for every analysis type and for every number of images considered, suggesting a reliable improvement of the BOS technique.

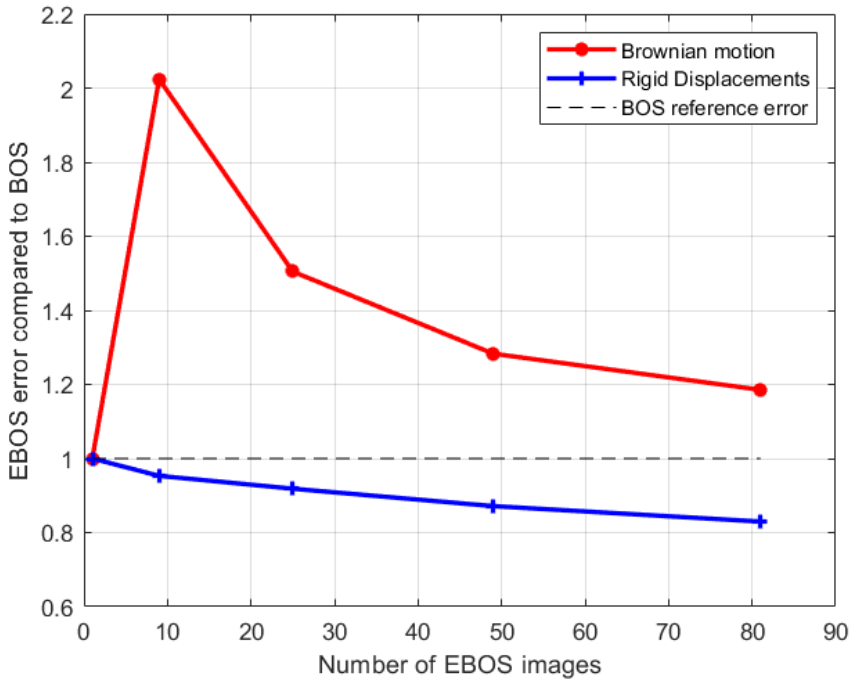


Figure 6.4: e_{tot} [pixels] of different EBOS methods to the variation of the number of background images, cIA of 8 pixels.

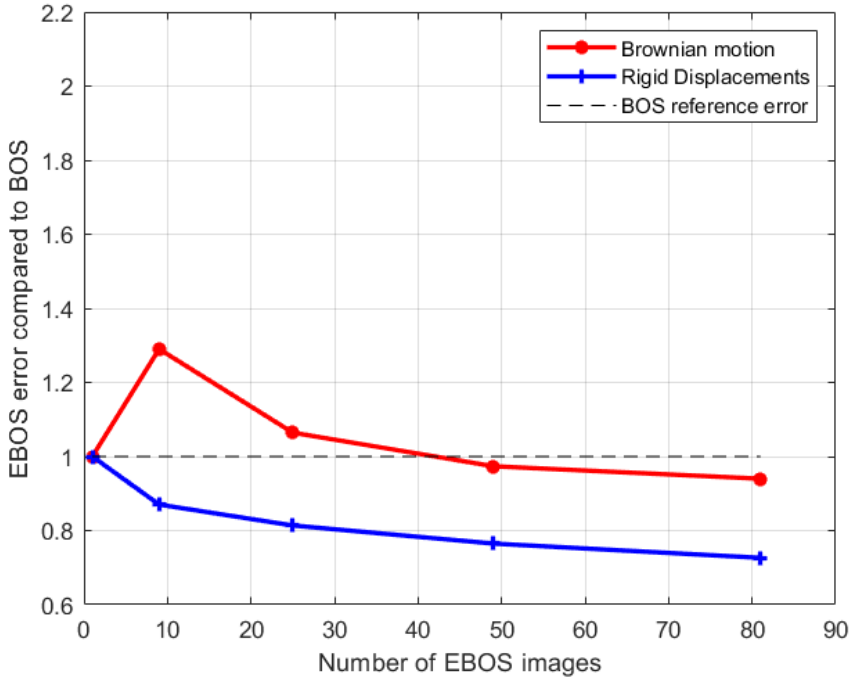


Figure 6.5: e_{tot} [pixels] of different EBOS methods to the variation of the number of background images, cIA of 4 pixels.

6.3.2. Comparison between EBOS and BOS

The EBOS seems to work particularly better than the BOS when small cIA are used, this is probably due to the fact that some cIA have no particles to cross-correlate in the BOS, but those particles may be present in the same cIA for a variated image in the EBOS.
The Rigid Displacement EBOS capability to find accurate results even for particles too sparse for the cross-correlation algorithm allows this technique to have comparable results, or even better results than the BOS analysed with bigger interrogation areas.
As an example, referring to table 6.3, the BOS analysed with a particle per cIA of 8 pixels has a mean pixel error around 0.056 pixels, but the Rigid Displacement EBOS made with 81 images and analysed with a particle every 4 cIA of 4 pixels has an mean pixel error of 0.05134 pixels.
This means that the EBOS is capable to use finer cross-correlation grids and have a higher resolution than the BOS, without a decrease in the measure accuracy.
If the resolution is the same, instead, the EBOS allows to have a decrease in the measure error in the order of $15\% - 25\%$ with respect to the BOS.

6.3.3. Comparison between different Rigid Displacement schemes

In this section the squared displacements are compared to the circular displacements, in order to establish whether the particle deformation in the EBOS causes a loss in the analysis accuracy or not.
As reported in table 6.4 and as more clearly shown in figure 6.6, the circular displacement scheme doesn't seem to introduce a greater error than the squared displacement scheme. The trend of the mean pixel error seems to be very similar, and it can be concluded that the particle deformation does not imply a loss in the analysis accuracy.
This conclusion also excludes the possibility that the Brownian motion's greater error could be caused by the fact that the pixel shifting is not integer and the particles are deformed.
Lastly, this result allows to impose fractional displacements for a squared scheme.

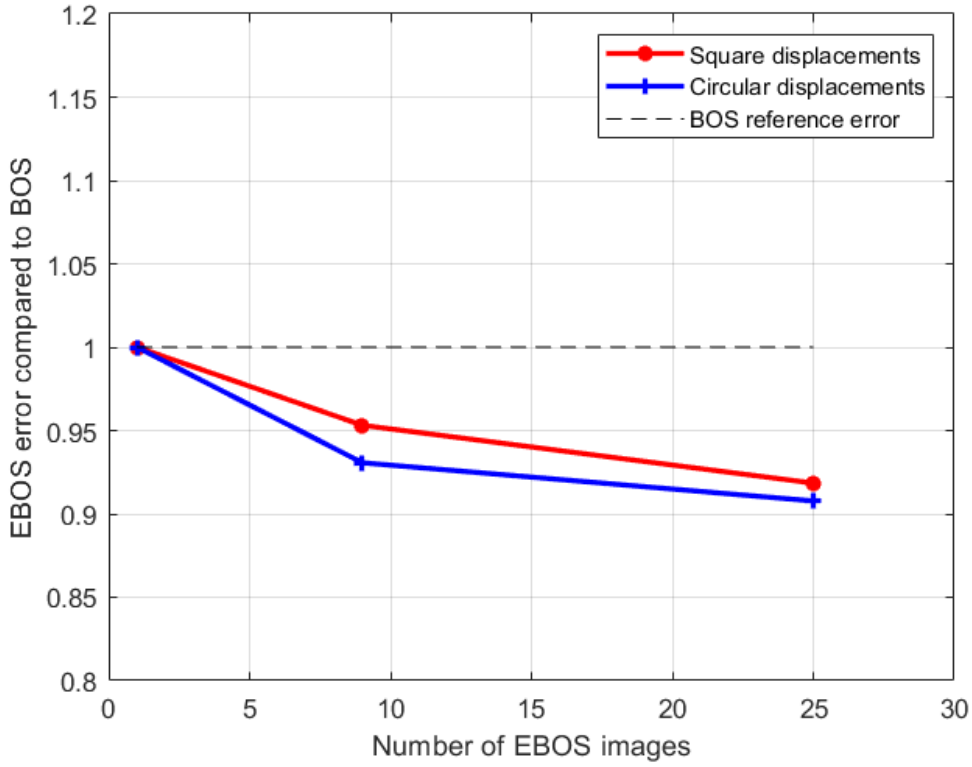


Figure 6.6: e_{tot} [pixels] of different Rigid Displacement patterns (square and circular) to the variation of the number of background images.

Then, the use of all the possible displacements for the EBOS backgrounds is compared to the use of only some displacements.
The advantage of using not a complete but a partial squared EBOS is that less images need to be cross correlated, with a consequent reduction of computational time and disk use. Some EBOS analyses with high resolution may take hours, speeding up the process without losing accuracy could at times be necessary.
Since the first 9 images of the partial squared EBOS have a shift of two pixels, as shown in figure 6.2, the mean pixel error of the displacement schemes will be plotted as a function of the maximum displacement of the analysis, and not as a function of the number of images.

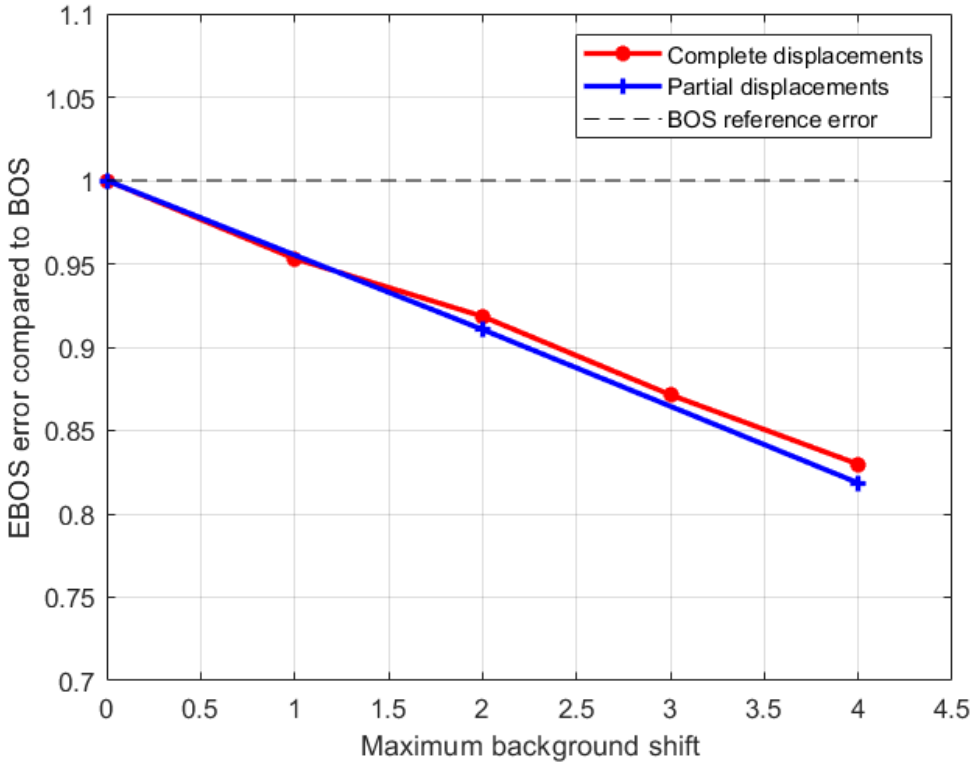


Figure 6.7: e_{tot} [pixels] of different Rigid Displacement patterns (complete and partial) to the variation of the number of background images.

The results presented in figure 6.6 show that the accuracy of the measure does not depend on the number of the EBOS images considered, but on the maximum shift of the background. Different sets of the background images have been used (with mean always equal to zero), and it resulted that, for this specific case, if at least 4 images are used for the ebos, the results are comparable to the complete displacements case. A possibility for this behaviour is the fact that the presence of higher shifts make the smaller shifted background redundant, as they don't introduce any different perspective in the cross-correlation.
A more in-depth study about which particular displacement to use could optimise the computational cost for the EBOS.

6.4. Conclusions

Referring to tables 6.3 and 6.4, and to the consideration that have been made in this chapter, the following conclusions can be drown about the EBOS technique:

- the Brownian motion is less accurate than the Rigid Displacements;
- the use of the Rigid Displacements EBOS instead of the BOS can cause the error in the measurement to decrease from 15% to 25%;
- the Rigid Displacements EBOS allows to use smaller cIA than the ones used for the BOS, without an increase in the mean pixel error. This means that a EBOS analysis can have a higher spatial resolution then a BOS analysis, without a loss in the measure accuracy;
- the deformation of the particles introduced by a non integer background shift does not affect the accuracy of an EBOS measure;
- the accuracy of an EBOS measure is a function of the maximum shift of the background and not of the number of images. Thus it is possible to use only some background displacements and obtain the same results with a smaller computational cost.

7 | BOS and EBOS experiments on a hydrogen flame

In this chapter, the techniques previously studied and optimized are applied over an experimental case.

The BOS technique has been used in the past to study a flame temperature distribution [10] and shape [7], and combustion is one of the many fields where the BOS can be applied. In this work, a premixed hydrogen flame will be used to test both the BOS, as optimized in chapters 3 and 5, and the EBOS, as optimized in chapter 6; then, a comparison will be made between the BOS and its enhanced variation to see whether the improvement noticed in chapter 6 applies also on an experimental case.
The rapid expansion of the hydrogen as a fuel for both the energy production and for the transportation, has led to a rapid increase in the studies around this element. In particular, the transparency of the hydrogen flame in the visible spectrum means that the study of such a flame is particularly difficult, and it requires the development of techniques capable of visualising the flame with sufficient resolution.
For this reason, an hydrogen flame seemed the perfect test case for the application of the BOS and EBOS techniques.

The first part of the work, after the implementation of the experimental setup discussed in chapter 2, has been dedicated to the flame stabilization: studies has been made in order to avoid the occurring of flashbacks and blow-offs, while possibly maintaining the flow laminar.
Then, both BOS and EBOS has been applied to the same schlieren picture of the flame, and a comparison has been made between them.
Eventually, the results are presented, and conclusions are drown about the use of the Enhanced BOS on a premixed hydrogen flame.

7.1. Flame stabilization

For a premixed flame, the fuel and oxidizer flows has to be regulated in order to avoid the instability of the flame, which can be caused by many phenomena. The most common are the flashback and the blow-off. Then a study has been made to try and keep the flame laminar.

7.1.1. Flashback

The first effect that can occur is the flashback, that can happen when the gas flow is reduced to the point where *the gas velocity becomes smaller than the burning velocity and the combustion wave propagates against the gas stream into the tube* [6]. The flashback phenomenon is particularly dangerous for the integrity of the plant, as the wave of high temperature and pressure that propagates backwards inside the pipes may damage some elements of the plant.
The flashback can be avoided by increasing the velocity of the mixture or by decreasing the flame burning velocity. Since the control parameters of the hydrogen distribution system are the plant thermal power and the equivalence ration, this can be obtained by increasing the plant power at the same equivalence ratio, and thus increasing the total flow velocity, or by reducing the equivalence ratio at the same plant power, increasing the air flow rate and reducing the flame speed at the same time.

7.1.2. Blow-off

The second effect that should be avoided is the blow-off. The blow-off is defined by Von Elbe [6] as the condition when *the boundary velocity gradient is so large that the combustion wave is driven beyond this* stable *position, the gas velocity exceeds the burning velocity in every stream line and the combustion wave blows off.*
The blow-off is not directly dangerous, but since the hydrogen is no longer burnt, it is released in the surrounding area, and at particular concentrations can self ignite. The laboratory is provided with an exhaust hood that promptly vents the very diluted air hydrogen mixture outside the lab, thus preventing any hydrogen buildup inside the test cell.
The blow-off can be avoided the opposite way of the flashback: the mixture flow rate can be slowed down by decreasing the plant power or by increasing the equivalence ratio, and the flame velocity can be accelerated by the increase in the equivalence ratio.

7.1.3. Turbulence

The last flame condition to be avoided is the turbulent flame, which is caused by a turbulent flow in the pipes. This condition is not dangerous and can be avoided by reducing the Reynolds number of the mixture, or by decreasing the plant thermal power or by increasing the equivalence ratio at the same power, thus reducing the air flow rate. The main reason for avoiding the turbulence of the flame is to have a laminar flame which is stationary. Hydrogen laminar flames have been widely studied and the models could be compared to the results obtained by BOS and EBOS. The second reason is stability-wise: the majority of the studies about the flame stability are made with a laminar flame, and the stability region such defined does not stand for a turbulent flame. In particular, the flashback phenomenon is more likely to occur for a turbulent flame [12].

7.1.4. Stability plot

A Matlab code has been developed in order to express the hydrogen and air flow rates as functions of the plant power and of the mixture equivalence ratio. From the flow rates, the mixture velocity and the Reynolds number has been calculated, and the turbulence limit have been plotted as function of the plant parameters. The algorithm A.6 shows the mathematical passages used.

The calculation of the flashback and blow-off limits has been made by following the concept of the critical boundary velocity gradient [8]. The data for the critical gradient for both the flashback (CG_f) and the blow-off (CG_b) as a function of the gas concentration can be found in tables 7.1 and 7.2 respectively.

Gas concentration	0.375	0.415	0.461	0.427	0.452	0.517	2.25	2.19
CG_f $[s^{-1}]$	256	463	813	335	622	1244	1235	1610
Gas concentration	0.556	0.650	2.00	0.762	1.84	0.978	1.53	0.578
CG_f $[s^{-1}]$	1775	3160	2840	5090	4860	8860	9080	2120
Gas concentration	2.12	0.683	1.93	0.892	1.68	1.44	1.22	0.636
CG_f $[s^{-1}]$	2060	3740	3420	7000	6980	8860	10030	2900
Gas concentration	2.07	0.724	1.83	0.853	1.01	1.57	1.16	1.41
CG_f $[s^{-1}]$	2735	4400	4450	6490	8540	8380	10040	9640

Table 7.1: Flame stability for flashback, hydrogen-air mixture [8].

Gas concentration	0.363	0.399	0.390	0.422	0.436	0.424	0.411	0.430
CG_b $[s^{-1}]$	265	460	785	337	618	1200	1690	2920
Gas concentration	0.432	0.418	0.459	0.443	0.475	0.453	0.429	0.441
CG_b $[s^{-1}]$	4530	1275	3450	7060	7520	12740	4060	5600
Gas concentration	0.451	0.468	0.580	0.603	0.634	0.496	0.530	0.533
CG_b $[s^{-1}]$	8600	12080	99400	127800	182700	60000	159000	291000
Gas concentration	0.584	0.631	0.702	0.766	0.817			
CG_b $[s^{-1}]$	17100	28700	45500	472000	633000			

Table 7.2: Flame stability for blow-off, hydrogen-air mixture [8].

Since the independent variable is expressed as *gas concentration, fraction of stoichiometric* instead of the equivalence ratio, the following equation has been used to obtain Φ.

$$\Phi = gas_c \frac{q_{tot}}{q_{air}} - \frac{q_{h2}}{q_{air}} \tag{7.1}$$

Where q_{tot}, q_{air} and q_{h2} are, respectively, the mixture, the air and the hydrogen mass flow rates.

To interpolate the data, polynomials of the sixth order have been used; in order to have more accurate results for low values of the critical gradient, it has been used the natural logarithm of the latter for the interpolation, and exponential of the fitted results have been used.
The critical gradient thus obtained was used to determine the critical velocity, as shown in equation 7.2.

$$U_{crit} = \frac{d}{8\,CR} \tag{7.2}$$

The complete algorithms used for the stability diagrams can be found in A.7 and A.8.

As it can be seen in figure 7.1, the region where the flame is stable is very limited, since the turbulent condition is already reached at low power. The parameters suggested for the flame are therefore a power of $Pw = 0.2kW$ and an equivalence ratio of $\Phi = 0.5$.
Unfortunately, these values are close to minimum values imposable to the flowmeter of the plant, which operates at non optimal conditions causing small accuracy in the imposed flow rate and oscillations. Since the stability region is already limited, variations in the

flow rate can easily cause the flame to flashback, damaging the hydrogen distribution system.
For this reason, it has been chosen not to maintain the flame laminar and operate in the turbulent region.

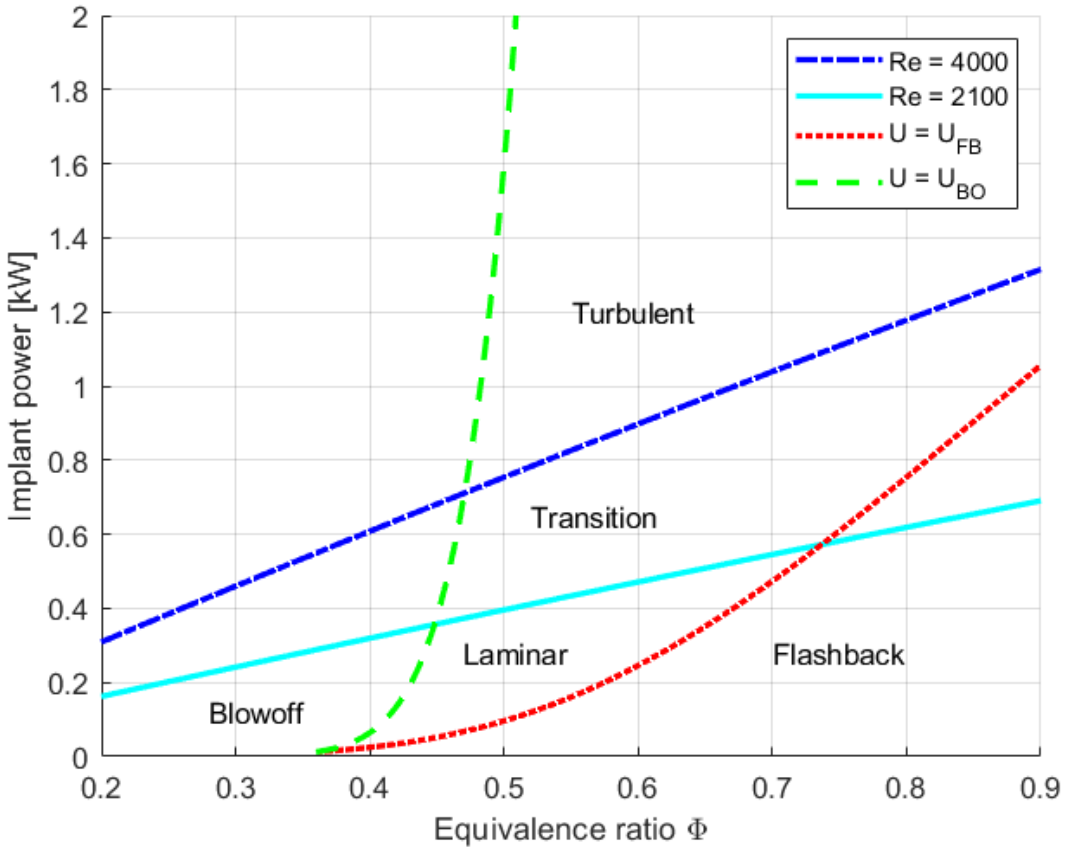


Figure 7.1: Flashback, blow-off and turbulence limits as function of the plant input parameters, the thermal power and the equivalence ratio Φ

If the flame stability region has been well recorded, the limits to the flashback and blow-off for an hydrogen-air mixture for a turbulent flow are less documented.
For the flashback an article of Kalantari and McDonnell [12] proposes experimental formulas to determine the flashback limits. With respect to the laminar flame, however, the results are less reliable, and some data are necessary, like the temperature of the burner and the temperature of the unburnt mixture, for which no measure instrument is present in the plant.
Given the inaccuracy of the results that could have been calculated, the stability regions for the turbulent hydrogen flame have not been determined theoretically, and the values for the equivalence ratio and for the plant power have been chosen empirically.

7.2. Analysis parameters

7.2.1. Setup parameters

For this analysis, the "hydrogen flame setup" discussed in chapter 2 has been used, with the same monitor and camera used before, but with the 35-135mm zoom lens instead of the 50mm.
The burner has been placed at a distance of 20cm from the monitor, a distance sufficient for the monitor not to overheat, and not to excessive, in order not to magnify the background displacements. The camera has been placed at 1.4 meters from the monitor and the lens has been regulated such that a pixel on the monitor would correspond a pixel on the camera sensor. It is impossible to establish with precision the focal length set for the zoom lens without a precise calibration, in any case it was estimated to be about 95mm. The values of the parameters used for the subsequent analyses are reported in table 7.3.

Parameter	Measure
Lens focal length	$\sim 95mm$
$F\#$	$8mm$
Camera shutter speed	$20000\mu s$
Background distance	$1.4m$

Table 7.3: Camera and lens parameters used for chapter 7 analyses.

7.2.2. Background parameters

In order to achieve the maximum resolution possible, a minimum cIA of 4x4 pixels is used in the images analysis; for this reason a density of one particle per bIA of 4x4 pixels has been used for the background composition. For high particle density backgrounds, it has already been shown how the Truncated particles give the best results in terms of accuracy, and how the particle area should cover from 15% to 22% of the total background area (see chapter 3).
For these reasons, Truncated particles with a diameter of 2 pixels have been used.

$$\begin{aligned} \frac{\pi}{4}\frac{d^2}{(l_{bIA})^2} &= 15\% - 22\% \\ d &= 1.75 - 2.12 \ \ [monitor\ pixels] \end{aligned} \tag{7.3}$$

Where d is the particle diameter in pixels and l_{bIA} is the side size in pixels of the bIA.

As shown in chapter 6, the most accurate EBOS method between the ones analysed is the Rigid Displacements. For this reason, 80 EBOS images with a square pattern have been created, and then added to the reference background to compose the 81 EBOS images for the Rigid Displacements.
As shown in chapter 6, a smaller amount of backgrounds could have been used with similar results in terms of accuracy; however, the computational time was not a limit for this analysis, and for reasons that will be better explained in the results section, a high number of images was necessary for the comparison with the BOS.

7.2.3. Cross-correlation parameters

As the temperature gradients are very sharp near the flame front, the maximum resolution available was necessary to study the hydrogen flame. Since a cIA with a side of 2 pixels would have been too small for any kind of particle to fit in, cIA of 4x4 have been used. Five iterations have been used, with cIA from 64 to 4 pixels and a overlapping of 50% for each iteration except for the last one. For small interrogation areas, the cross-correlation signal is weaker then the one obtained with cIA with side of 8 pixels, thus the signal to noise threshold need to be reduced. The non valid vectors have been substituted, but they are still marked as non valid for later purposes.

Adaptivity	
N° iterations	5
cIA dimension [pixels]	64 - 32 - 16 - 8 - 4
Overlapping	50% - 50% - 50% - 50% - 0%
Signal to noise threshold	1.2
Global STD threshold	9
Local median threshold	5
Substitute	True
Interpolating neighborhood	±3 pixels

Table 7.4: PIV parameters used for chapter 7 analysis.

7.2.4. Mixture parameters

As the laminar stability plot could not be used for the choice of the mixture composition, experimental parameters for the distribution system that would maintain the turbulent flame stable have been found.
Three different analyses has been made, with the parameters shown in table 7.5

Parameter	plant thermal power	Equivalence ratio Φ
Analysis 1	$3\,kW$	0.65
Analysis 2	$2\,kW$	0.5
Analysis 3	$1\,kW$	0.45

Table 7.5: Mixture parameters used for the experiments.

The flow's Reynolds number has been calculated at these three conditions, by following the passages that are reported in algorithm A.6. The results show, as supposed, that all the three flames are turbulent.

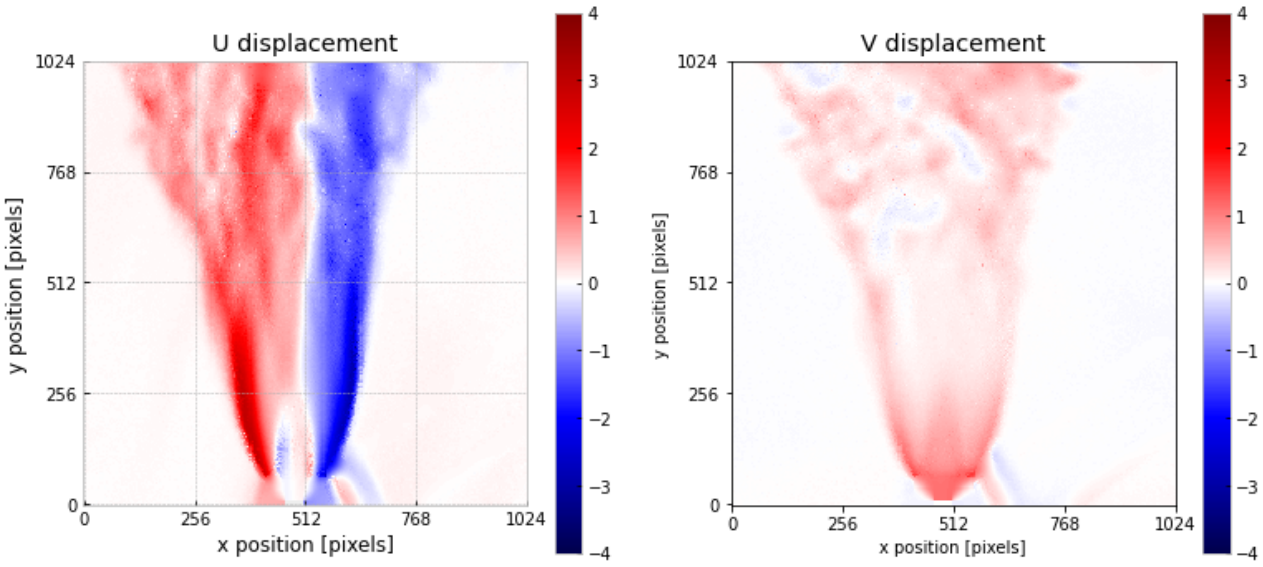
Flame	$Pw = 3kW$	$Pw = 2kW$	$Pw = 1kW$
Reynolds number	5870	10609	12390

Table 7.6: Reynolds number of the three flames.

7.3. Results

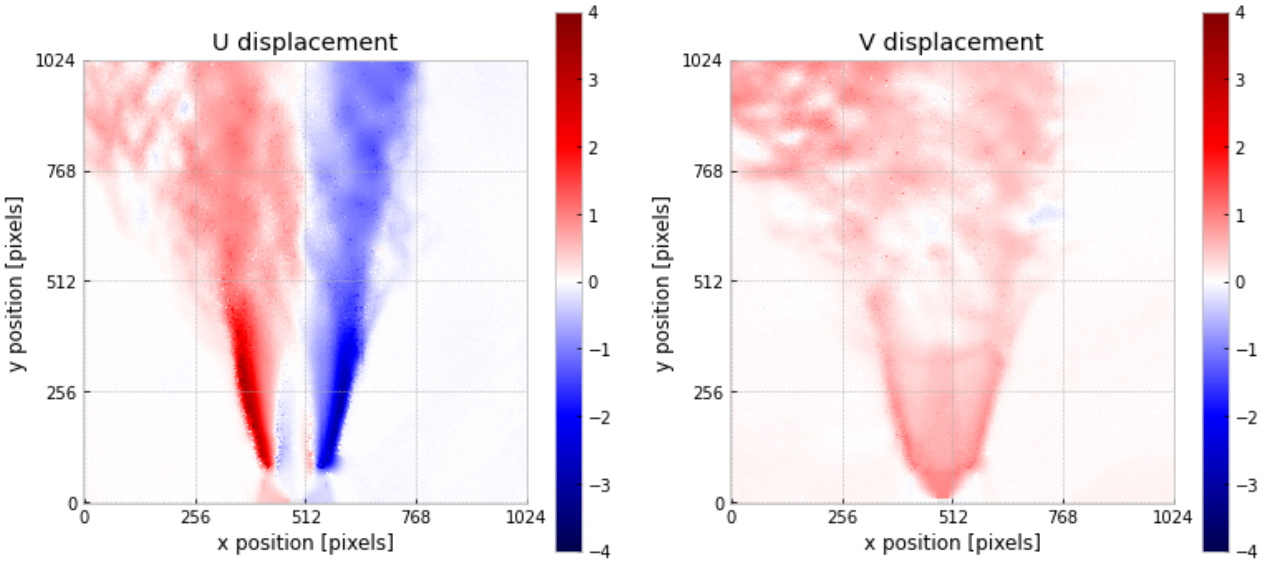
7.3.1. Flames comparison

The first study that can be done is qualitative: a visual comparison between the displacement fields obtained by the BOS analyses for the three flames at different conditions has been made, to find possible differences and similarities between the three flame shapes.



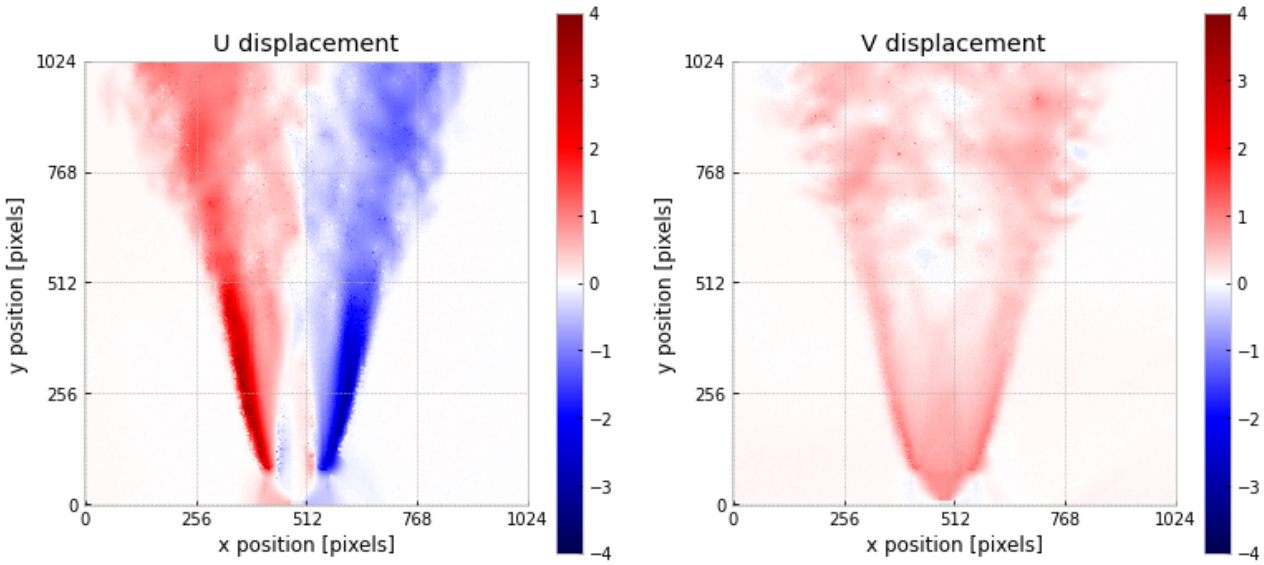
(a) Displacement along x. (b) Displacement along y.

Figure 7.2: Displacements [pixels] for the flame at 1kW.



(a) Displacement along x. (b) Displacement along y.

Figure 7.3: Displacements [pixels] for the flame at 2kW.



(a) Displacement along x.

(b) Displacement along y.

Figure 7.4: Displacements [pixels] for the flame at 3kW.

In all the three pictures, the red area indicates a positive horizontal displacement, and the blue areas a negative displacement along x direction; with respect to how the images where acquired this means that the red area indicates a positive temperature gradient and the blue area a negative one. The white areas between the red (to the left) and blue (to the right) areas are the zones of maximum temperature, whereas the areas with blue to the left and red to the right are cold spots.
The vertical displacement images are similar to the horizontal displacement ones: they present the same shapes but the temperature gradients are always negative, as the temperature decreases along with the distance from the flame. For this reason the vertical displacement images are less useful to determine the flame shape.
Since the flow is turbulent and thus non axial-symmetric, it is impossible to reconstruct the flame 3D geometry and then the flame temperature distribution with a single camera, as a tomography technique would be necessary [7]; the study of the images will be therefore limited to the flame 2D geometry, and the temperature of the flame will not be evaluated.

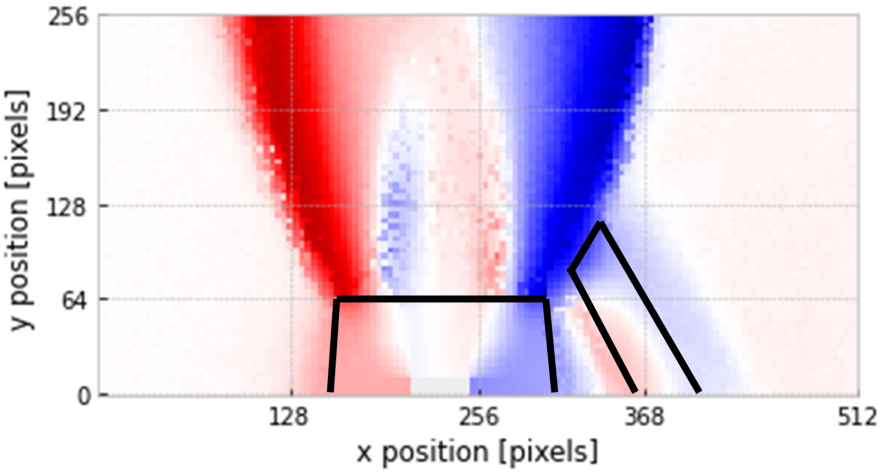
In all the three BOS images it it possible to identify the flame as the area in the bottom of the image at the center: the two white lines with red to the left and blue to the right that converge in a point are the flame front, with the colder urburnt mixture inside. At the sides of the flame front, as the flow is very fast and the convection rate is high, the

temperature gradients are the highest. Above the flame, the plume is vividly turbulent and the only clear region is a vertical white line, which represents the mid point of the plume, where the derivative of the temperature gradient is zero.
If the plume is clearly turbulent, the flame is more laminar, and only the tip evidences major turbulence, as the white lines representing the flame are mainly not blurred. The flame turbulence (along with its height) increases with the plant power and it is maximum at 3 kW. This can be noticed by the shape flame front, which is less symmetric, and by its width, which is more pronunced. For this reason, it has been chosen as test case for the BOS and EBOS the first analysis, where the plant power is set to 1kW and the equivalence ratio is $\Phi = 0.45$.

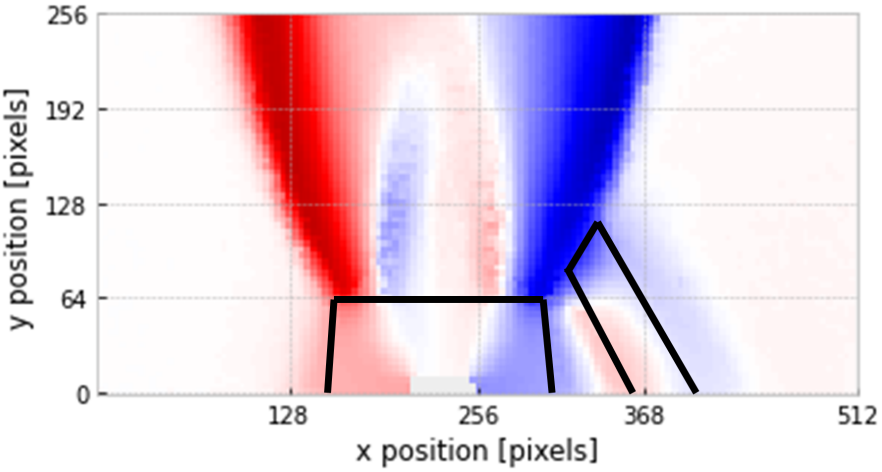
7.3.2. Comparison between BOS and EBOS

Since for this analysis there is not a synthetic displacement field and the schlieren image is an actual displacement field, it cannot be defined a theoretical displacement field to compare the BOS and the EBOS with.
A first comparison can be qualitative, with a particular focus to the flame front area, shown in detail in figure 7.5.

In figure 7.5 the position of the burner and of the igniter is highlighted in black. The background is completely obscured by the presence of these object and the cross-correlation algorithm cannot find any signal. As the resulting vector is non valid, the cIA behind the burner replaces the not found values with an interpolation of the closest valid vectors, crating the halo visible behind the burner and the igniter.
The metal objects close to the flames are heated by it, and the air around them is consequently heated; the corresponding temperature gradient is captured by the BOS and shown by the red and blue areas aside of the burner and of the igniter.
As is can be seen in figure 7.5, the two images are similar, but the EBOS image is less grainy. This happens because in the BOS image there are many vectors which are substituted, and in the EBOS image these vectors are averaged over a larger number of valid vectors.



(a) BOS.

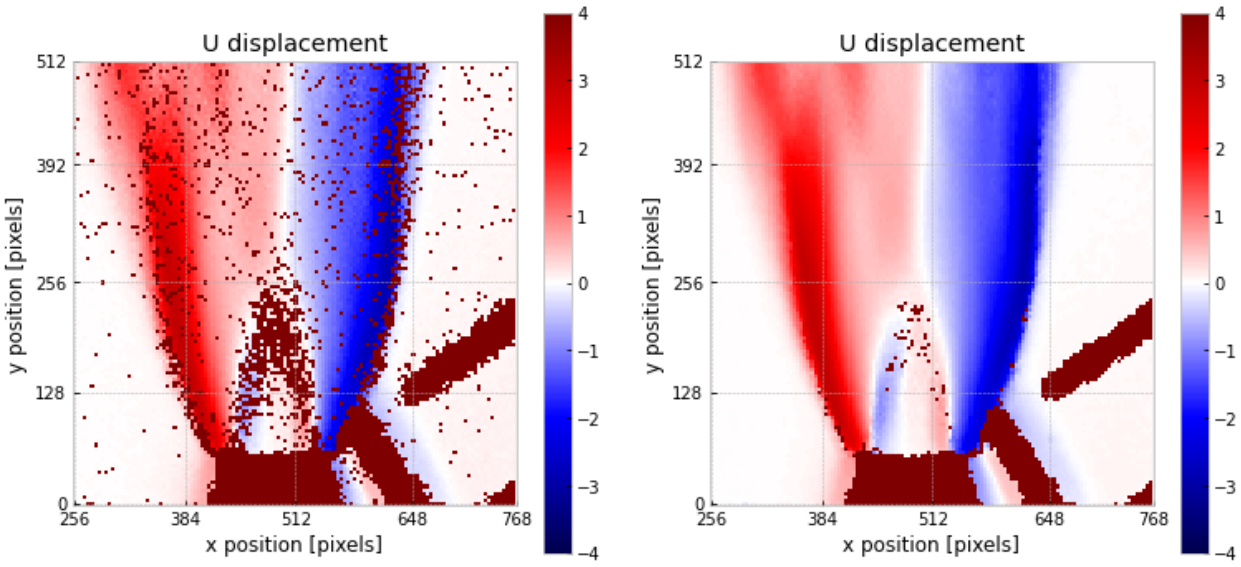


(b) EBOS.

Figure 7.5: BOS and EBOS horizontal displacements [pixels] for a detail of the flame at 1kW. Burner and igniter perimeters highlighted in black.

For BOS and EBOS images rendering, two ways are viable: in the first case the invalid vectors are substituted as it happened in figure 7.5, in the second case they are not substituted but highlighted as non valid. This leads to the possibility of a different type of comparison between BOS and EBOS, where the quality of a measure can be determined by the number of non valid vectors.
For the BOS, valid vectors have been normally represented, and invalid vector has been highlighted in brown (figure 7.6. For the EBOS the discussion is more complex as some vectors can be valid for a certain number of images and non valid for the others. For every cIA, it has been chosen to average only the valid vectors, when at least five of them are present. If the number of valid vectors is below five, the average has not been computed

and the cIA (referring to figure 7.6) is highlighted in brown.
The algorithm wrote for the average of the sole valid vectors is reported in B.3.



(a) BOS. (b) EBOS.

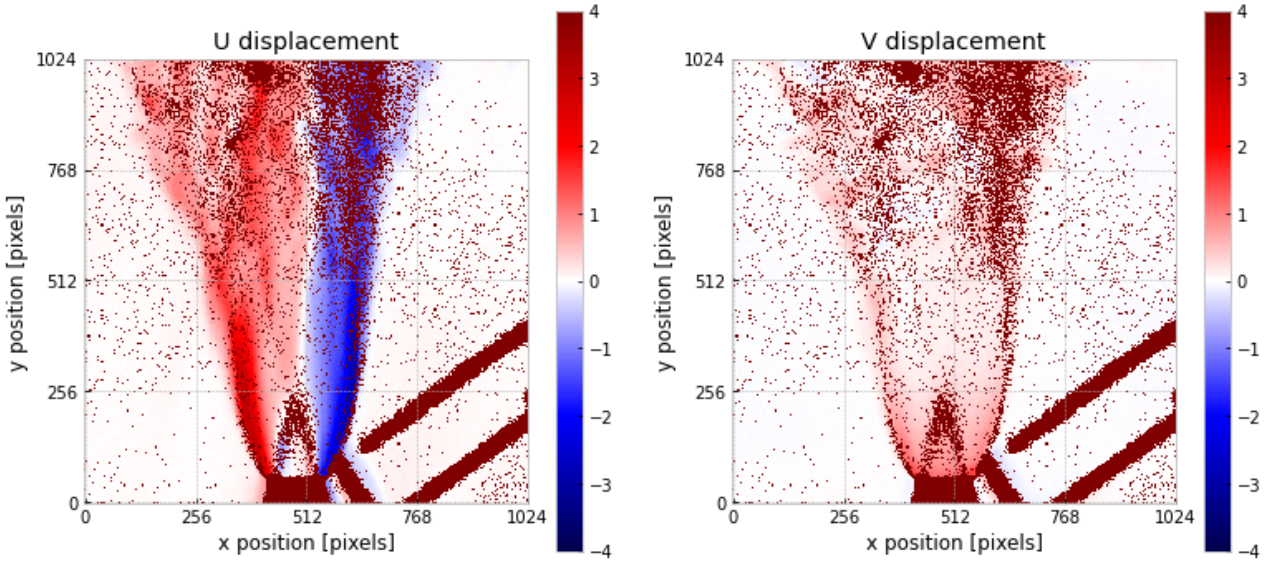
Figure 7.6: BOS and EBOS horizontal displacements [pixels] for a detail of the flame at 1kW. Non valid vectors are highlighted in brown.

The difference BOS and EBOS in figure 7.6 is evident, as the number of valid vectors for the EBOS is clearly higher. The flame profile in the EBOS image is more detailed and the overall picture is less grainy.
To compare the number of invalid vectors, a BOS analysis has been made between the reference background and itself. The displacement field resulted obviously null except for the cIA behind the burner and the igniter, that returned invalid vectors. The number of invalid vector has been counted and subtracted to the BOS and EBOS analyses, in order to count only the vectors non valid because of the cross correlation algorithm. The results are shown in table 7.7.

Technique	BOS	EBOS
Total vectors	65536	65536
Invalid vectors (total)	11868	3290
Invalid vectors (image covered)	2910	2910
Invalid vectors (algorithm)	8958	380

Table 7.7: Vector differentiation for BOS and EBOS applied to the 1kW flame.

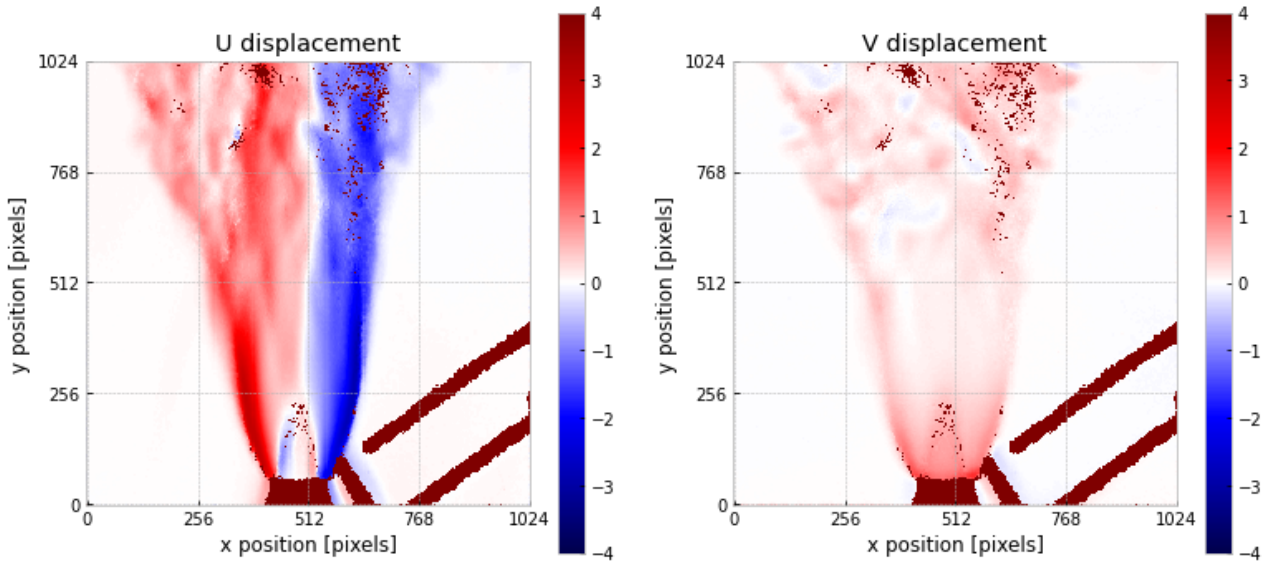
If cIA covered by the burner and by the igniter are not considered, the number of non valid vectors for the EBOS is 23 times smaller than it is for the BOS, and the percentage of valid vectors in the image goes from being the 85.3% for the BOS to the 99.4% for the EBOS.
The full images for the BOS and EBOS results are presented in figures 7.7 and 7.8, respectively, whereas a bigger image for the EBOS horizontal displacement is presented in figure 7.13 at the end of the chapter.



(a) Displacement along x.

(b) Displacement along y.

Figure 7.7: BOS displacements [pixels] for the flame at 1kW. Non valid vectors are highlighted in brown.



(a) Displacement along x.

(b) Displacement along y.

Figure 7.8: EBOS displacements [pixels] for the flame at 1kW. Non valid vectors are highlighted in brown.

A more specific comparison can be made for the horizontal displacements near the flame front. Figure 7.9 shows the values for the horizontal displacements for both BOS and EBOS analyses. The plot refers to a horizontal line at 176 pixels of height and it goes from 280 to 720 pixels on the x axis.
The two biggest peaks (one positive and one negative) represent the sudden temperature drop outside of the flame, which is also highlighted by the darker red and blue areas at the sides of the flame that can be seen in figure 7.6. The two smaller peaks (one positive and one negative) at the center of the plot represent the flame front.
A first thing that can be noticed is the leak of information for the BOS about the flame, which is the whole area between the flame fronts: the temperature decreases on the inside, as the light rays pass through the fresh mixture, whereas on the borders of the cone flame no fresh mixture is present and the integral of the temperature is maximum. For the BOS, almost none of this behaviour is noticed, as only a small quantity of displacement values are present. For the EBOS, on the contrary, a complete description of the flame region is possible, as only a couple of values are missing. Even if the EBOS flame is not perfectly symmetric, with the right side more irregular than the left side, this is probably due to the turbulent nature of the flame in that region.
The temperature gradient between the flame and the higher peaks seems to be well represented by both BOS and EBOS, as the actual peaks. The second difference, though,

is in the region where the displacement gradient is the highest: also in this region the BOS has many missing information, when the EBOS misses only a couple of vectors. However, this problem is less severe as the region presents displacements almost linear and an interpolation could reconstruct quite precisely the displacement field.

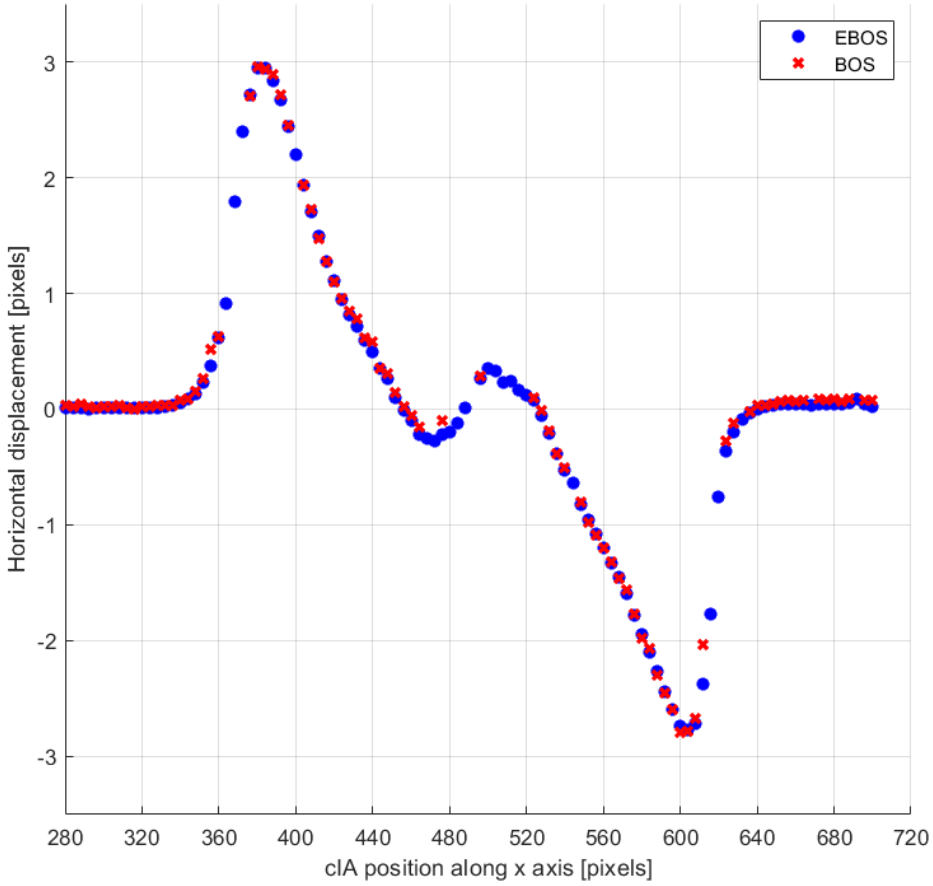


Figure 7.9: BOS and EBOS horizontal displacements at y = 176 pixels, x position from 280 to 720 pixels.

Far from the flame, where the background should not be affected by the schlieren object, all the horizontal displacement should be equal to zero. Figure 7.10 shows the horizontal displacement of the two optical techniques far from the flame. As it can be seen in the figure, the displacements for the EBOS are lower in mean value and more regular than the ones found by the BOS. This small horizontal displacement cannot be due to a small temperature gradient as it is positive on both sides of the flame, and it probably represents a small error in the measure.

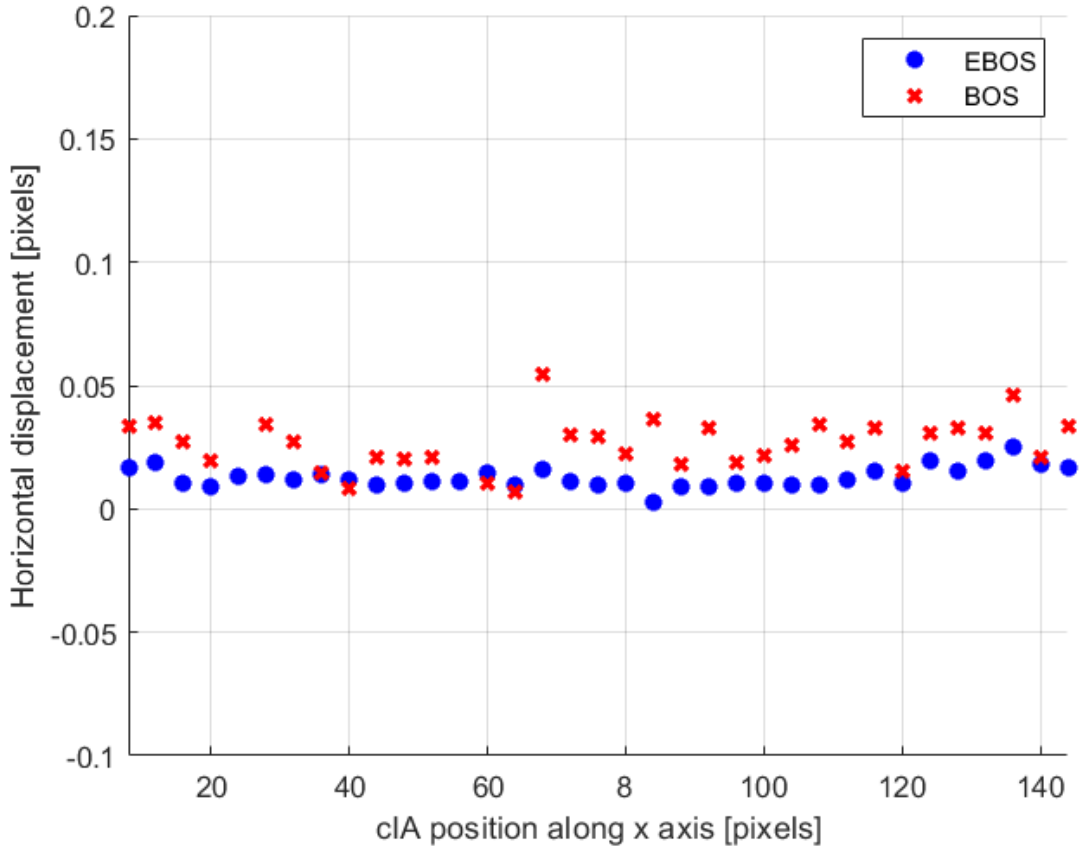


Figure 7.10: BOS and EBOS horizontal displacements at y = 176 pixels, x from 17 to 141 pixels.

In order to try to understand why the cross-correlation algorithm cannot find valid vectors for some cIA, a comparison between the BOS displacement map and the actual image acquired by the camera has been made. Since the invalid vector are such for both the horizontal and vertical displacement maps, only the horizontal displacements have been compared to the acquired image.
In figure 7.12 the areas with high numbers of invalid vectors are highlighted by some arrows. By comparing this image to figure 7.11 is it possible to notice why the algorithm could not find a solution for some cIA.
In figure 7.12, the blue arrows highlight the area where the temperature gradient is maximum, thus the displacements are highest and the particles are deformed. The red arrows highlight the flame where the light rays undergo a first deflection caused by the flame, a second (opposite in direction) gradient caused by the colder unburnt mixture, and a third deflection caused by the other side of the flame cone. Even if the combination of these deflections is small or even null, the background results blurred. A similar effect

happens for the white arrows: the high turbulence in this area causes the alternation of gradients in different directions, and as for the flame, the light rays are deflected many times before reaching the camera sensor, fact that causes the image to be blurred. The green arrows highlight the presence of the burner and of the igniter, which completely cover the background.
In the last case the algorithm cannot find a peak since there is no information to be cross-correlated. Where the image is blurred, instead, the light intensity of the particle spreads in the area around, causing the particle itself to weaken and black background to lighten. For this reason, the cross-correlation signal is weaker, and at the same time the noise is higher, causing the signal to noise ratio to decrease significantly.
The last effect that can be observed in figure 7.11 is a number of invalid vectors randomly displayed over the BOS image (the brown dots), this is probably due to the small dimension of the cIA, which in some cases cannot find enough information to cross-correlate.

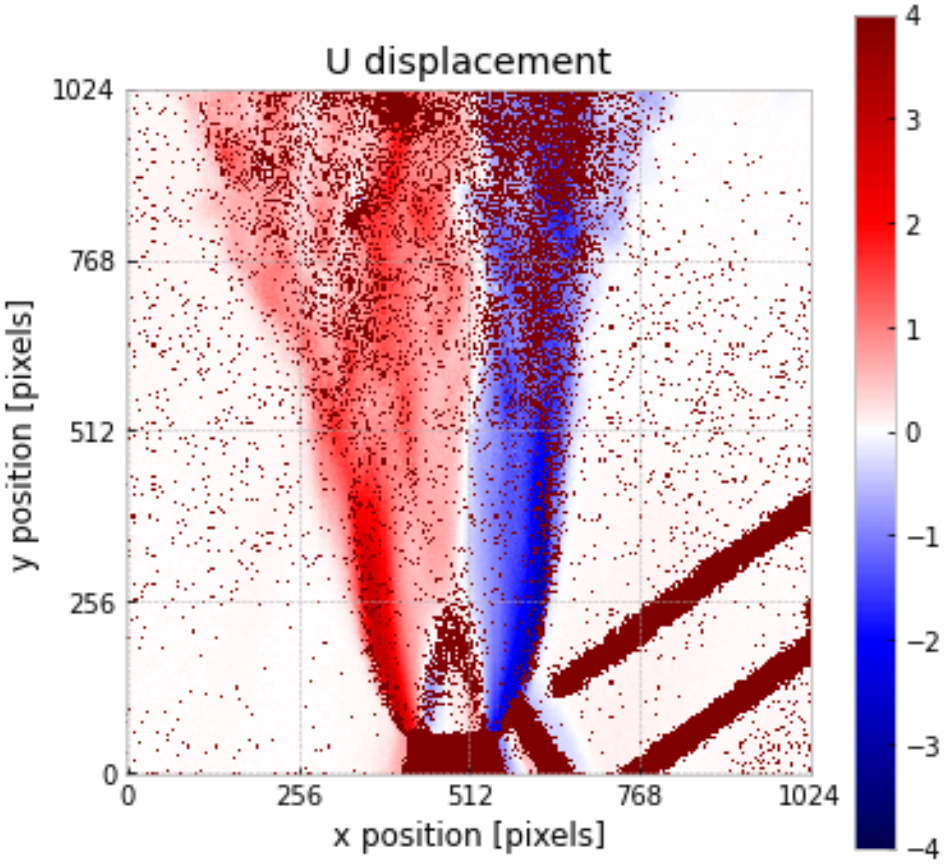


Figure 7.11: BOS horizontal displacements [pixels]. Non valid vectors highlighted in brown.

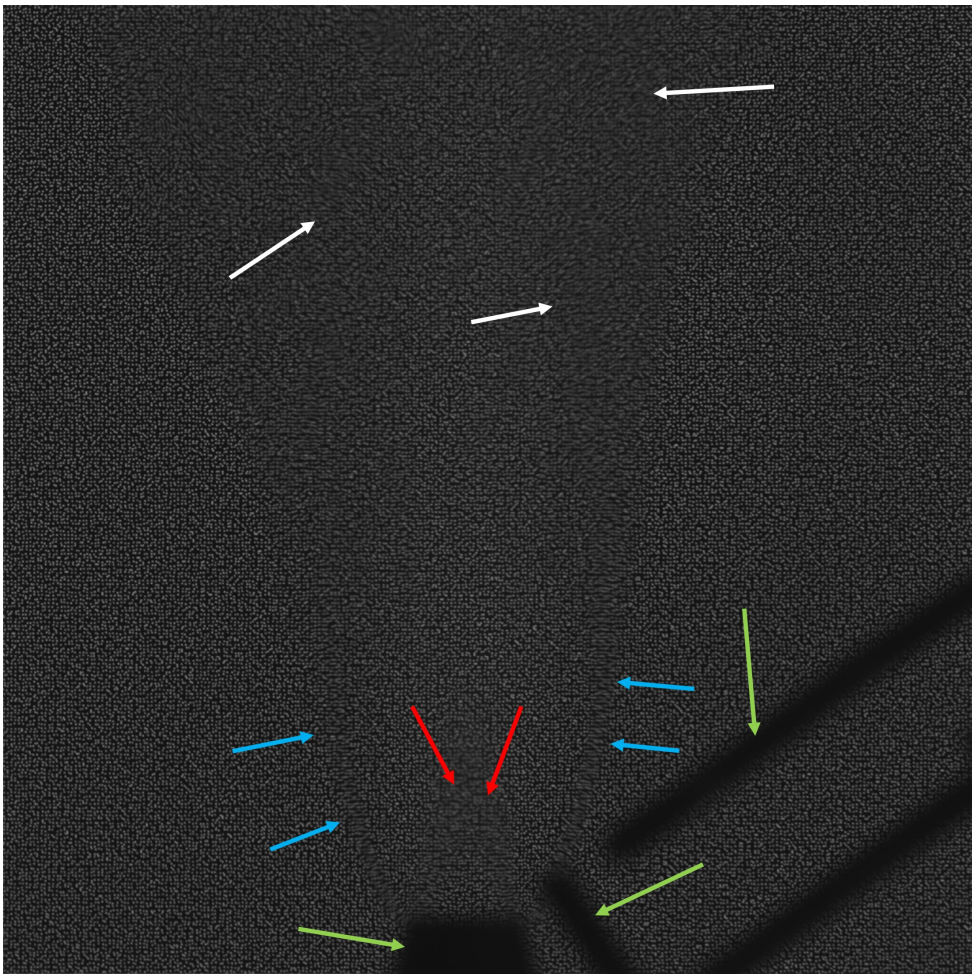


Figure 7.12: Background image, distorted by the flame and acquired by the camera.

For what concerns the random errors, if the single image has no particles, one of the background shift will likely have enough information to cross-correlate. For the blurred image, the use of different background images in the analysis helps the EBOS to find, at least in some of them, a signal strong enough to return a valid vector.

7.4. Conclusions

The comparison between BOS and EBOS over the experimental case could not have been made as in the previous chapters, as no exact displacement field is available. The comparison has been qualitative, made by comparing the displacement field, and quantitative, by comparing the number of non valid vectors.
The results obtained are the following:

- the three different flames are different in size and in shape, and the flame obtained

with minimum power seems to be the least turbulent. For this reason, only the flame at 1kW has been further analysed;

- both the BOS and the EBOS images are capable of visualizing all the density gradients of the mixture, which are the flame, the plume and the air heated by the metal objects near the flame;
- the images obtained with the EBOS are less grainy than the ones obtained by the BOS;
- the number of valid vectors for the EBOS resulted to be significantly higher than the ones obtained with the BOS;

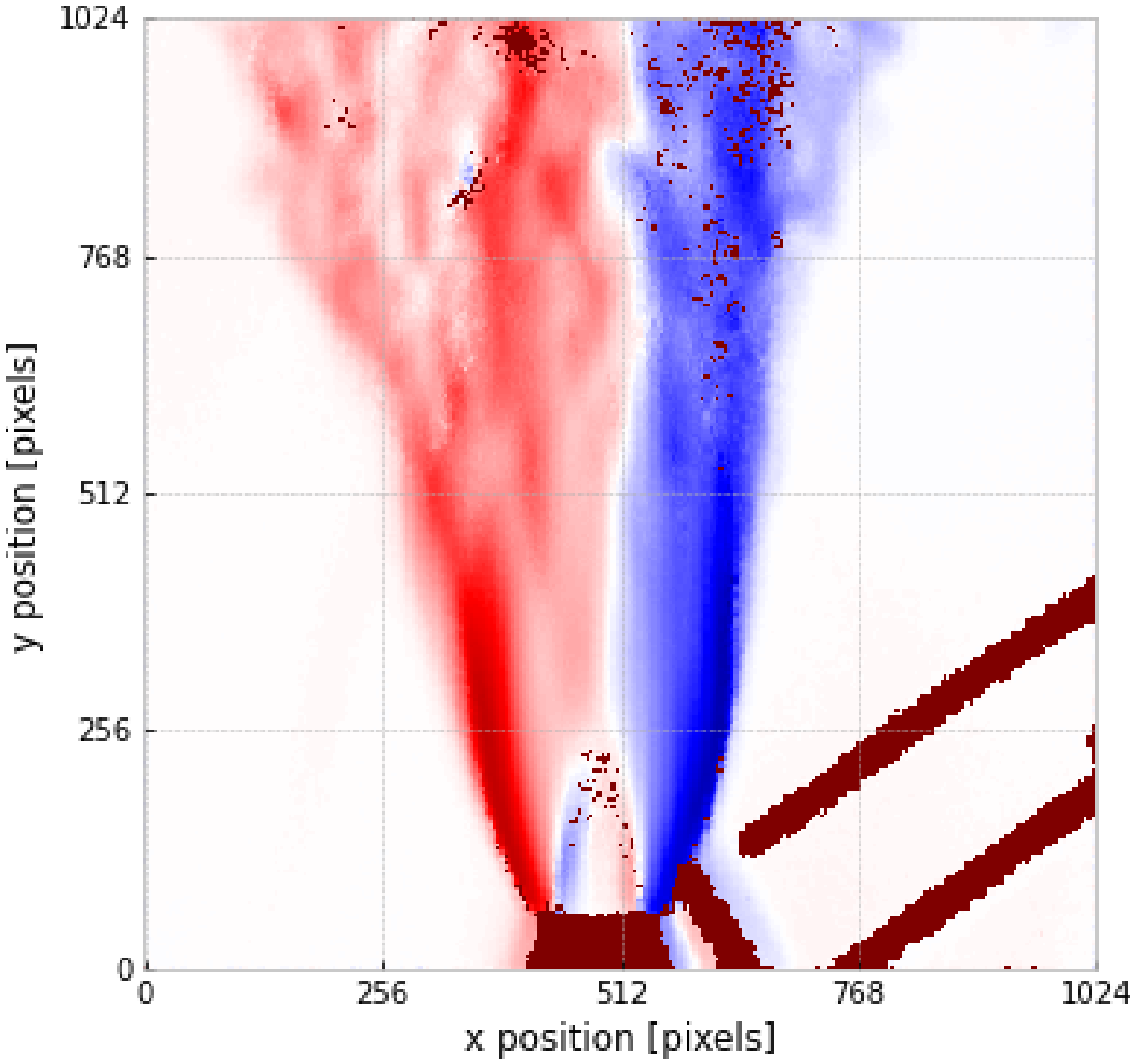


Figure 7.13: EBOS horizontal displacement [pixels] for the flame at 1kW.

8 | Conclusions

The aim of this work was to study the use of the EBOS technique with a monitor-displayed background.
By using a monitor in the BOS analysis, it is possible to create the background from scratch in a very customisable and fast way, and to apply a synthetic displacement field to it. The synthetic displacement field allows to impose a controlled displacement to the background, and after the acquisition and analysis of the images is carried out, to compare the measured results to the theoretical ones that have been imposed.
This process has been used to compare the results of different BOS analyses, and the parameters used for the background definition were varied to study their effect on the accuracy of the technique.
The study, accordingly to the results in literature [26] showed that there is a correlation between the particle dimension, the particle density, and the accuracy of the measure. In particular, a higher particle density gives the best results but only if the dimension of the particles is decreased accordingly. Three different shapes for the particles were tested and the Truncated particles resulted to best behave at higher densities, the ones that are necessary to increase the resolution of the technique.

Then, an analysis of the uniformity of the monitor used for this work was made. The monitor turned out to show different non-uniformities, that would vary from small scratches on the screen to wide areas of different luminosity. Three different ways to correct the non-uniformity of the monitor were proposed, and it was carried out an analysis to compare the BOS performances with and without the correction of the monitor.
The results showed that the accuracy of the measure was not particularly influenced by the monitor correction and thus the mathematical correction of the non-uniformity of the monitor is not necessary for a BOS analysis.

Next, OpenPIV and Dantec Dynamic Studio, two different PIV software, were used for the same BOS analyses, in order to study their customization and performance differences, and to establish which one would have been better for the BOS. For this comparison, five

different displacement fields were used, in order to highlight the software behaviour in different situations.

- The two programs showed similar overall accuracy: Dantec was more accurate near zero order discontinuities in the displacement field and OpenPIV was more accurate near higher order discontinuities.
- Being an open source Python package, rather then a proper software, OpenPIV resulted to be more customizable then Dantec, in particular for what concerns the interrogation areas. The limits in the cIA dimensions present in Dantec deny the possibility to increase the BOS resolution, whereas in OpenPIV the adaptivity parameters can be set more freely, and optimised with respect to every analysis.

Once the main background parameters were defined and the software for the cross-correlation was chosen, the EBOS technique was presented and then optimized.
With reference to the cases in literature where the EBOS was used, two different background variation techniques were proposed and compared thanks to the synthetic displacement fields. The Brownian motion resulted to be less accurate than the Rigid Displacements, as the imposed random displacements don't sum up to zero. The Rigid Displacements showed a higher accuracy then the BOS for every analysis made, in particular when the cIA are smaller. This means that the EBOS can always produce better results than the BOS without an increase in the setup complexity, and this is true in particular for analyses with high resolution.
For the Rigid Displacement EBOS, the study showed that the accuracy is a function of the maximum background shift, and not of the number of EBOS images, and that the particle deformation due to the non integer shifting of the background does not affect the accuracy of an measure.

Eventually, both the BOS and the EBOS, made with the parameters which resulted to be optimal in the previous chapters, were applied to an experimental case: a hydrogen flame.
The stability limits for the flame were studied, in order to draw a stability plot for the flame. However, the characteristics of some elements in the plant made it impossible to have a laminar flame, thus mass flow rates for hydrogen and air have been experimentally chosen at the conditions such that neither the flashback nor the blow-off would occur.
After having defined the flow rates of air and hydrogen, a qualitative comparison between the BOS and the EBOS was made, which showed that both the BOS and the EBOS were capable to capture all the density gradients in the field of view. The EBOS was less grainy

than the BOS, particularly near the flame front.
Since no theoretical displacement field was available, the quantitative comparison between BOS and EBOS was made on the number of valid vectors: for the BOS the cross-correlation algorithm could not find a solution for many interrogation areas, and the displacement field was partially reconstructed from the neighbours of the invalid vectors. For the EBOS, since there is a large number of displacement fields, it is possible to obtain a map composed only by the valid results, and to obtain a more accurate displacement field than the BOS.
The reasons for the invalid vectors have been investigated, and it resulted that the blurring of the pixels, which can be caused by high density gradients or by the turbulence, is responsible for the majority of the non valid vectors. For the rest of the cases, the responsible for the non validity of the vectors is the small dimension of the cIA. In the EBOS, the higher number of slightly different cross-correlations allows to obtain valid vectors at least for some of the n cases, from which the displacement field is obtained.

Both in a simulated case and in an experimental one, the Rigid Displacement EBOS as defined in chapter 6 resulted to be a more accurate optical technique than the BOS, returning a smaller error when applied to a synthetic displacement, and a clearer image of the density gradients when applied a hydrogen flame.

8.1. Possible future development

In this work, only two different EBOS techniques have been studied, the Brownian motion and the Rigid Translations; however, the use of a monitor to display the backgrounds allows to create many different techniques for the background variation, which would be interesting to compare. As an example, a different type of Brownian motion with a null average of all the displacements would be interesting to study.
For the Rigid Displacements EBOS, a deeper analysis could be made around the number of background to use, and what should be the minimum number of backgrounds to use.

Lastly, a different setup could be used, which could produce a laminar hydrogen flame; this way the displacement field obtained by the BOS and by the EBOS could be compared to the theoretical field obtained by the literature, and the comparison between the techniques could be more precise.

Bibliography

[1] B. Atcheson, W. Heidrich, and I. Ihrke. An evaluation of optical flow algorithms for background oriented schlieren imaging. *Experiments in fluids*, 46, 2009.

[2] M. Carrisi and F. Iapaolo. Experimental study of laminar premixed hydrogen burners. Master's thesis, Politecnico di Milano, 2021.

[3] L. Casero. Analysis of the ebos technique for application in the trova supersonic wind tunnel. Master's thesis, Politecnico di Milano, 2019.

[4] F. Cozzi and E. Göttlich. Enhanced background oriented schlieren (ebos). *Journal of physics: conference series*, 1249, 2019.

[5] L. Critchley. Applications of schlieren optics. *AZO Optics*, 2018.

[6] L. V. Elbe. *Combustion, flames and explosions of gases*. Academic Press Inc, 2 edition, 1968.

[7] S. Grauer, A. Unterberger, A. Rittler, K. Daun, A. Kempf, and K. Mohri. Instantaneous 3d flame imaging by background-oriented schlieren tomography. *Combustion and flame*, 196, 2018.

[8] J. Grumer, M. Harris, and V. Rowe. *Fundamental flashback, blowoff, and yellow-tips limits of fuel gas-air mixtures*. United States Department of Interior, 1956.

[9] M. J. Hargather and G. S. Settles. Natural-background-oriented schlieren imaging. *Experiments in fluids*, 48, 2009.

[10] E. Iffa, A. Aziz, and A. Malik. Gas flame temperature measurement using background oriented schlieren. *Journal of applied sciences*, 11, 2011.

[11] N. Instruments. Dynamicstudio manual, 2021.

[12] A. Kalantari and V. McDonell. Boundary layer flashback of non-swirling premixed flames: Mechanisms, fundamental research, and recent advances. *Progress in Energy and Combustion Science*, 61, 2017.

[13] LG. Lg display-specification for approval-lp097qx1 lcd datasheet, 2012.

[14] A. Liberzon, T. Kaufer, A. Bauer, P. Vennemann, and E. Zimmer. Openpiv-python v.0.23.6, 2021.

[15] E. Marino. Analysis and validation of quantitative bos technique. Master's thesis, Politecnico di Milano, 2019.

[16] L. Mendes, A. Bernardino, and R. Ferreira. piv-image-generator: An image generating software package for planar piv and optical flow benchmarking. *ScienceDirect*, 12, 2020.

[17] W. Merzkirch. *Flow visualization*. Academic press, INC., 2 edition, 1987. ISBN 0-12-491351-2.

[18] D. Michaelis, D. R. Neal, and B. Wieneke. Peak-locking reduction for particle image velocimetry. *Measurement science and technology*, 27, 2016.

[19] M. Ota, F. Leopold, F. Jagusinski, and K. Maeno. Comparison between cbos and cgbos for supersoni flow. *15th International Symposium on Flow Visualization*, 2012.

[20] H. photonics K.K. Digital ccd camera c8484-05c datasheet, 2015.

[21] M. Raffel. Background-oriented schlieren (bos) techniques. *Experiments in fluids*, 56, 2015.

[22] F. Reichenzer, M. Schneider, and A. Herkommer. Improvement in systematic error in background-oriented schlieren results by using dynamic backgrounds. *Experiments in Fluids*, 62, 2021.

[23] G. P. Russo. *Aerodynamic Measurements*. Woodhead Publisher, 2011. ISBN 978-1-84569-992-5.

[24] G. S. Settles. *Schlieren and Shadowgraph Techniques*. Springer, 2001.

[25] L. Sosio. Analysis of the ebos technique implemented with random particles displacements. Master's thesis, Politecnico di Milano, 9 2021.

[26] N. Vinnichenko, A. Uvarov, and Y. Plaksina. Accuracy of background oriented schlieren for different background patterns and means of refraction index reconstruction. *15th International Symposium on Flow Visualization*, 2012.

[27] F. J. Weinberg. *Optics of Flames*. Butterworths, 1963.

[28] M. P. Wernet. Real-time background oriented schlieren with self-illuminated speckle background. *Measurement Science and Technology*, 31, 2019.

A | Appendix A: Matlab algorithms

Since the total number of functions used is very high, only the scripts that have been developed for this thesis work are here reported.

A.1. Margins function

The "margins" algorithm is a small modification of a longer and more complex function, the "percx" and "percy" variables are defined and then used to limit the position of the particle center.

```
1 mrg = 56;
2 percx = 100-sqrt(mrg*100);
3 percy = 100-sqrt(mrg*100);
4
5 ...
6
7 x0s = random('unif', winLeft+pivParameters.lastWindow(2)*(percx/100), ...
      winLeft + pivParameters.lastWindow(2)*(1 - percx/100), [1, 1]);
8 y0s = random('unif', winTop+pivParameters.lastWindow(1)*(percy/100), ...
      winTop + pivParameters.lastWindow(1)*(1 - percy/100), [1, 1]);
```

A.2. Anti overlapping function

The algorithm presented below is part of a longer script, which creates the particle centers given the background parameters.

```
while dist < diametro
    number = number+1;

    if number > 10000
        error('densita di particelle troppo alta, non converge')
    end

    x0s = random('unif', ...
        winLeft+pivParameters.lastWindow(2)*(percx/100), winLeft + ...
        pivParameters.lastWindow(2)*(1 - percx/100), [1, 1]);
    y0s = random('unif', ...
        winTop+pivParameters.lastWindow(1)*(percy/100), winTop + ...
        pivParameters.lastWindow(1)*(1 - percy/100), [1, 1]);
    x(n) = x0s;
    y(n) = y0s;
    dist_v = zeros(n-1);
    for kk = 1:n-1
        dist_v(kk) = sqrt((x(n)-x(kk))^2+(y(n)-y(kk))^2);
    end
    dist = min(dist_v);
end
```

A.3. Particle shapes

Some of the syntax used refers to a different part of the script, which is absent for reasons of space. However, the definition of the shape is completely reported in the algorithm below.

```
% Black and white particles
elseif metodo == 2

    for i = minY:maxY
        for j = minX:maxX
            d = sqrt((i+0.5-centerY)^2+(j+0.5-centerX)^2);
            if d≤r
```

```
                Im(i,j) = Im(i,j) + 255;
            end
        end
    end

% Truncated particles
elseif metodo == 3

    [xs, ys] = meshgrid(minX:maxX, minY:maxY);
    intensity = intensity/255*1500;
    if maxY-minY > 0 && maxX-minX > 0
        Im(minY:maxY,minX:maxX) = Im(minY:maxY,minX:maxX) + ...
            intensity .* exp(-((double(xs)+0.5-centerX).^2 + ...
                (double(ys)+0.5-centerY).^2)./(0.5*r.^2));
    end

% Gaussian Particles
elseif metodo == 4

    [xs, ys] = meshgrid(minX:maxX, minY:maxY);
    if maxY-minY > 0 && maxX-minX > 0
        Im(minY:maxY,minX:maxX) = Im(minY:maxY,minX:maxX) + ...
            intensity .* exp(-((double(xs)+0.5-centerX).^2 + ...
                (double(ys)+0.5-centerY).^2)./(0.5*r.^2));
    end
end
```

A.4. Synthetic displacement fields

The displacements fields functions here reported are condensed in a single script for reasons or space, but actually are each part of a longer and more complex function. However, the main aspects of each synthetic field are presented.

```
dx_max = 6;
dy_max = 6;

% Diagonal displacement field
if y1(i) > x1(i) % fuori dalla diagonale
    % nulla
else
    x2(i) = x0(i)+dx_max*(1-x0(i)/(larg*2+obj.marginsX/2));     % ...
        tutto il resto si sposta di x
```

```
9       y2(i) = y0(i)+dy_max/(2*larg)*(y0(i)-(obj.marginsY/2));     % ...
            tutto il resto si sposta di y
10  end
11
12  % Cross displacement field
13  if x0(i) > y0(i) && y0(i)<larg*2+marg-x0(i)     % primo quadrante
14      y2(i) = y0(i) - dy_max;
15
16  elseif x0(i) > y0(i) && y0(i)>larg*2+marg-x0(i) % secondo quadrante
17      x2(i) = x0(i) + dx_max/2;
18
19  elseif x0(i) < y0(i) && y0(i)<larg*2+marg-x0(i) % terzo quadrante
20      x2(i) = x0(i) + dx_max;
21
22  elseif x0(i) < y0(i) && y0(i)>larg*2+marg-x0(i) % quarto quadrante
23      y2(i) = y0(i) - dy_max/2;
24  end
25
26  % Sinusoidal displacement field
27  x2(i) = x0(i) + ...
        dx_max*sin((x0(i)-marg/2)/larg/2*pi)*sin((y0(i)-marg/2)/larg/2*pi); ...
          % seno con periodo doppio della larghezza (spostamento orizz.)
28  y2(i) = y0(i) - ...
        dy_max*sin((y0(i)-marg/2)/larg/2*pi)*sin((x0(i)-marg/2)/larg/2*pi); ...
          % seno con periodo doppio della larghezza (spostamento vert.)
29
30  % Parabolic displacement field
31  a1 = -1/(256^2);
32  b1 = -512*a1;
33  c1 = 0;
34  a2 = -1/(256^2);
35  b2 = -1536*a2;
36  c2 = 1+768^2*a2;
37
38  if x1(i) > 0 && x1(i) < larg && y1(i) > 0 && y1(i) < larg   % prima ...
        parabola
39      x2(i) = x0(i) + x_max*(a1*x1(i)^2 + b1*x1(i) + c1)*(a1*y1(i)^2 + ...
            b1*y1(i) + c1);
40      y2(i) = y0(i) + y_max*(a1*x1(i)^2 + b1*x1(i) + c1)*(a1*y1(i)^2 + ...
            b1*y1(i) + c1);
41
42  elseif x1(i) > 0 && x1(i) < larg && y1(i) > larg && y1(i) < larg*2  % ...
        seconda parabola
43      x2(i) = x0(i) + x_max*(a1*x1(i)^2 + b1*x1(i) + c1)*(a2*y1(i)^2 + ...
```

```
        b2*y1(i) + c2);
44      y2(i) = y0(i) - y_max*(a1*x1(i)^2 + b1*x1(i) + c1)*(a2*y1(i)^2 + ...
        b2*y1(i) + c2);
45
46  elseif x1(i) > larg && x1(i) < larg*2 && y1(i) > 0 && y1(i) < larg  % ...
    terza parabola
47      x2(i) = x0(i) - x_max*(a2*x1(i)^2 + b2*x1(i) + c2)*(a1*y1(i)^2 + ...
        b1*y1(i) + c1);
48      y2(i) = y0(i) + y_max*(a2*x1(i)^2 + b2*x1(i) + c2)*(a1*y1(i)^2 + ...
        b1*y1(i) + c1);
49
50  elseif x1(i) > larg && x1(i) < larg*2 && y1(i) > larg && y1(i) < ...
    larg*2     % quarta parabola
51      x2(i) = x0(i) - x_max*(a2*x1(i)^2 + b2*x1(i) + c2)*(a2*y1(i)^2 + ...
        b2*y1(i) + c2);
52      y2(i) = y0(i) - y_max*(a2*x1(i)^2 + b2*x1(i) + c2)*(a2*y1(i)^2 + ...
        b2*y1(i) + c2);
53
54  end
55
56  % Linear displacement field
57  if x0(i) > marg/2 && x0(i) < larg+marg/2 && y0(i) > marg/2 && y0(i) < ...
    larg*2+marg/2 % prima meta
58      x2(i) = x0(i) + x_max*x1(i)/larg;
59
60  elseif x0(i) > larg+marg/2 && x0(i) < larg*2+marg/2 && y0(i) > marg/2 ...
    && y0(i) < larg*2+marg/2  % seconda meta
61      x2(i) = x0(i) + x_max*(2-x1(i)/larg);
```

A.5. Rigid Translations

The algorithm used for 81 complete square displacements is proposed. In order to have circle displacements it is sufficient to remove the "round" functions from "dispX" and "dispY". To obtain the partial displacements, multiply all the displacements obtained per 2 and use only 25 EBOS images.

```
1  for n=1:length([particleMap.allParticles.x])
2
3      x = particleMap.allParticles(n).x; %x coordinate of the dots in Im0
4      y = particleMap.allParticles(n).y; %y coordinate of the dots in Im0
5
6      if n_ebos < 9 % primo giro
```

```
7           alpha = n_ebos*pi/4;
8           dispX = round(cos(alpha));
9           dispY = round(sin(alpha));
10
11      elseif n_ebos < 25 % secondo giro
12          alpha = (n_ebos-8)*pi/8;
13          dispX = round(cos(alpha)*2.2);
14          dispY = round(sin(alpha)*2.2);
15
16      elseif n_ebos < 49 % terzo giro
17          alpha = (n_ebos-24)*pi/12;
18          dispX = round(cos(alpha)*3.6);
19          dispY = round(sin(alpha)*3.6);
20          if abs(dispX) > 3
21              dispX = dispX/4*3;
22          end
23          if abs(dispY) > 3
24              dispY = dispY/4*3;
25          end
26
27      else % quarto giro
28          alpha = (n_ebos-48)*pi/16;
29          dispX = round(cos(alpha)*5);
30          dispY = round(sin(alpha)*5);
31          if abs(dispX) > 4
32              dispX = dispX/5*4;
33          end
34          if abs(dispY) > 4
35              dispY = dispY/5*4;
36          end
37      end
38
39      intensityB = 255;
40      x2 = x + dispX;
41      y2 = y + dispY;
42      Im2 = renderParticle(pivParameters, imageProperties, x2, y2, ...
            intensityB, Im2);
43  end
```

A.6. Reynolds number

This is the script that has been used for the calculation of all the mixture parameters (such as flow rates, velocity, density...) starting from the plant parameters: the thermal power and the equivalence ratio. With small variations, this script has been used as a function for flashback and blow-off stability diagrams.

```
%% Diagrammi per la stabilita di fiamma %%
% Creato da Mauro Rossi

close all
clear
clc

%% Parametri
Pw = 1000;                  %  potenza impianto                    [W]
phi = 0.5;                   % rapporto di equivalenza             [/]
P_bar = 1.01325;             % pressione nel condotto              [bar]
T_cel = 15;                  % temperatura dei gas                 [C]


%% Dati
r = 0.004;                   % raggio interno del cannello,      [m]
A = pi*r^2;                  % area della sezione                [m2]
pci_2 = 33.33;               % potere calorifico inferiore h2    [KWh/Kg]
stechio = 34.33;             % rapporto stechiometrico aria-idrogeno [/]
R = 8.314;                   % costante gas prfetti              [J/k*mol]
mu_o2 = 2e-5;                % viscosita ossigeno                [Pa*s]
mu_n2 = 1.73e-5;             % viscosita azoto                   [Pa*s]
mu_h2 = 8.69e-6;             % viscosita idrogeno                [Pa*s]

% conversioni in sistema internazionale
pci = pci_2*3600*1000;       % potere calorifico inferiore h2    [Ws/Kg]
P = 100000*P_bar;            % pressione aria                    [Pa]
T = T_cel+273.15;            % temperatura dei gas               [k]
Mm_h2 = 2.016/1000;          % massa molecolare idrogeno         [kg/mol]
Mm_aria = 28.85/1000;        % massa molecolare sria secca 79-21 []kg/mol]

%% calcoli da NIST (1 atm 15C, usati per confronto)

% Densita
rho_h2_nist = 0.0852;                       % [kg/m3]
rho_o2 = 1.354;                             % [kg/m3]
```

```
rho_n2 = 1.185;                                % [kg/m3]
rho_aria_nist = 1.354*0.21+1.185*0.79;         % [kg/m3]

%% Calcoli

% Portate
q_h2 = Pw/pci;                % portata idrogeno                [kg/s]
q_aria = stechio*q_h2/phi;    % portata aria                    [kg/s]
q_tot = q_h2+q_aria;          % portata totale                  [kg/s]

% Densita
rho_h2 = Mm_h2*P/R/T;         % densita idrogeno                [kg/m3]
rho_aria = Mm_aria*P/R/T;     % densita aria                    [kg/m3]

% Frazioni molari e densita totale
qv_h2 = q_h2/rho_h2;          % portata volumetrica idrogeno      [m3/s]
qv_aria = q_aria/rho_aria;    % portata volumetrica aria          [m3/s]
qv_tot = qv_h2+qv_aria;       % portata volumetrica totale        [m3/s]

y_h2 = qv_h2/qv_tot;            % Frazione molare dell'idrogeno   [m3/m3]
y_aria = qv_aria/qv_tot;        % Frazione molare dell'aria       [m3/m3]
rho_t = rho_h2*y_h2+rho_aria*y_aria;  % densita totale miscela    [kg/m3]

% velocita
v = qv_tot/A;                % velocita della miscela all'efflusso [m/s]

% Viscosita                                                       [Pa*s]
mu = (y_h2*mu_h2*sqrt(2)+y_aria*0.21*mu_o2*sqrt(32)+ ...
    +y_aria*0.79*mu_n2*sqrt(28))/(y_h2*sqrt(2)+y_aria*0.21*sqrt(32)+ ...
    +y_aria*0.79*sqrt(28));

% Reynolds
Re = rho_t*v*2*r/mu;          % Reynolds della miscela           [/]
Re_ref = 4000;                % soglia moto turbolento            [/]

if Re > Re_ref
    error(['Moto turbolento: Re=' num2str(Re), ', ovvero ', ...
        num2str(Re/Re_ref),' volte il limite'])
end
```

A.7. Flashback and blow-off interpolation

In this script, a section of the flashback and blow-off functions is reported. The polynomial interpolation has been made on the logarithms of the values found for the velocity gradient, in order to have a higher accuracy for small equivalence ratios.

```
%% Flashback %%

% interpolazione flashback
x = [0.375 0.415 0.461 0.427 0.452 0.517 2.25 2.19 0.556 0.650 2.00 ...
    0.762 1.84 0.978 1.53 0.578 2.12 0.683 1.93 0.892 1.68 1.44 1.22 ...
    0.636 2.07 0.724 1.83 0.853 1.01 1.57 1.16 1.41]';
x_f = x*q_tot/q_aria-q_h2/q_aria;
y_f = log([256 463 813 335 622 1244 1235 1610 1775 3160 2840 5090 ...
    4860 8860 9080 2120 2060 3740 3420 7000 6980 8860 10030 2900 2735 ...
    4400 4450 6490 8540 8380 10040 9640]');

fit_f = fit(x_f,y_f,'poly6');
CR = exp(fit_f(phi));

% controllo velocita
u_f = 0.008/8*CR;
diff = v-u_f;

%% Blow-off %%
% interpolazione blowoff
x = [0.363 0.399 0.390 0.422 0.436 0.424 0.411 0.430 0.432 0.418 ...
    0.459 0.443 0.475 0.453 0.429 0.441 0.451 0.468 0.580 0.603 0.634 ...
    0.496 0.530 0.553 0.584 0.631 0.702 0.766 0.817]';
x_b = x*q_tot/q_aria-q_h2/q_aria;
y_b = log([265 460 785 337 618 1200 1690 2920 4530 2175 3450 7060 ...
    7520 12740 4060 5600 8600 12080 99400 127800 182700 17100 28700 ...
    45500 60000 159000 291000 472000 633000]');

fit_b = fit(x_b,y_b,'poly4');
CR = exp(fit_b(phi));

% controllo velocita
u_b = 0.008/8*CR;
diff = v-u_b;
```

A.8. Flame stability

The flame stability script maps the stability limits for every equivalence ratio, by imposing a Φ, and finding the correct plant power for which the stability limit is reached. For turbulence, flashback and blow-off the scheme is the same: a Newton's tangent method is used to reach a fast convergence of the functions previously defined. Here the flashback part is presented as an example.

```
%% controllo flashback

kf = 0;      % counter delle iterazioni
for i=1:dim

    kf = kf+1;
    phi = phi_f(i);

    % Dati
    Pw1 = Pw_first_guess;    % potenza impianto      [W]
    Pw2 = Pw_second_guess;

    % Soluzione
    err = 10;
    toll = 0.0000001;

    while err > toll
        er1 = flashback2 (Pw1, phi, P_bar, T_cel);
        er2 = flashback2 (Pw2, phi, P_bar, T_cel);
        zero = Pw1-er1*(Pw2-Pw1)/(er2-er1);
        Pw1 = zero;
        Pw2 = Pw1*1.1;
        err = abs(flashback2 (Pw1, phi, P_bar, T_cel));
    end
    Pw_f(i) = Pw1;
end
```

B | Appendix B: Python algorithms

Like it happened for appendix A, the algorithms here reported are only the ones written for this thesis work.

B.1. Error evaluation

The algorithm here presented is a section of the one used to evaluate the mean pixel error for the Diagonal displacement field. The algorithm used for the other displacement fields is identical, except for the definition of u_eff and v_eff

```
# calcolo di spostamenti effettivi

dx_max = 3
dy_max = 3

u_eff = np.zeros((x_dim,y_dim))
v_eff = np.zeros((y_dim,x_dim))

for i in range(0,y_dim):
    for j in range(0,y_dim):

        if y_val[i,j] > x_val[i,j]: # oltre la diagonale

            a=5     # inutile
        else:
            u_eff[i,j] = dx_max*(1-x_val[i,j]/512)    # tutto il resto ...
                si sposta
            v_eff[i,j] = -dy_max/512*y_val[i,j]      # tutto il resto ...
                si sposta

# calcolo errore
```

```
e_tot = 0
e_mat = np.zeros((y_dim,x_dim))                    # inizializzo mappa errore

for i in range(0,y_dim):                           # giro prima sulle colonne
    for j in range(0,x_dim):                       # e poi sulle righe

        eu = (U_mean[i,j]-u_eff[i,j])**2           # errore su U per ogni pixel
        ev = (V_mean[i,j]-v_eff[i,j])**2           #   //      V   //
        e = np.sqrt(eu+ev)                         # errore totale
        e_mat[i,j] = e      # assegno l'errore di ogni pixel alla matrice
        e_tot += e          # errore totale come somma degli errori ...
            di tutti i pixel

e_tot = e_tot/tot_dim       # errore medio del pixel
e_max = np.max(e_mat)

print("L'errore medio di questa simulazione e", e_tot)
print("L'errore medio di questa simulazione senza contare il bordo ...
    esterno e", np.mean(e_mat[1:-1,1:-1]))
```

B.2. Correction matrices

The next algorithm defines the three matrices used for the correction matrices.

```
b_n = (img_1_5_6 + img_2_5_6 + img_3_5_6 + img_4_5_6 + img_1_11 + ...
    img_2_11 + img_3_11 + img_4_11)/8

...

# fitting globale
ay = np.arange(0,dim)/1          # asse generico, va da 0 a 512
corx = np.zeros((dim,dim))       # inizializzo matrici
cory = np.zeros((dim,dim))

x_mean = np.mean(b_n,1)          # media della matrice su x (medio ...
    tutte le colonne)
y_mean = np.mean(b_n,0)          # media della matrice su y (medio ...
    tutte le righe)
px = np.polyfit(ay,x_mean,2)     # fitting grado 2 sulla media colonne
py = np.polyfit(ay,y_mean,2)     # fitting grado 2 su media righe
r1 = np.polyval(px,ay)           # funzione derivante dal fitting
r2 = np.polyval(py,ay)
```

```
16
17 for i in range(0,dim):
18     corx[:,i] = r1              # matrice fitting su x: serie di ...
           colonne tutte uguali
19     cory[i,:] = r2              # matrice fitting su y: serie di ...
           righe tutte uguali
20
21 g_corr = np.sqrt(corx*cory)        # matrice finale: media tra le due
22 new_1 = img_1_5_6/g_cor
23
24 # fitting locale
25 ax = np.arange(0,dim)/1          # nuovo asse generico 0-512
26 corrx = np.zeros((dim,dim))
27 corry = np.zeros((dim,dim))
28
29 for i in range(0,dim):
30
31     x_fun = b_n[i,:]                  # estraggo riga da matrice
32     p_x = np.polyfit(ax,x_fun,100)  # calcolo fitting della riga
33     rx = np.polyval(p_x,ax)           # funzione derivante dal fitting
34     corrx[i,:] = rx                   # aggiungo riga a matrice di ...
           correzione
35
36 for i in range(0,dim):
37
38     y_fun = b_n[:,i]                  # stessa cosa per le colonne
39     p_y = np.polyfit(ax,y_fun,100)
40     ry = np.polyval(p_y,ax)
41     corry[:,i] = ry
42
43 l_corr = np.sqrt(corry*corrx)          # matrice finale = media tra ...
       ottenuta con righe e con colonne
44 new_2  = img_1_5_6/l_corr      # matrice media corretta
45
46 # fitting invertendo la matrice
47 i_corr = b_n                  # matrice media
48 new_3 = img_1_5_6/i_corr      # matrice media corretta
```

B.3. EBOS average

The next algorithm has been used to average only the valid vectors for the EBOS. The imposed background shift is subtracted to the EBOS analysis, this is necessary since not all the EBOS images are used to average, and the use of only some EBOS would lead to a mean different from zero. After that, a control on the validity of the vector is made, and valid vectors are averaged.

```
for k in range(0,n_max):          # ciclo sulle 81 bos diverse
    kk = str(k)
    ite = int(k)

    f = open('prove_15_02/risultati/pw 1 phi ...
        0.45/Open_PIV_results_'+an+'_Test_'+kk+'/field_A000.txt', 'r')
    content = f.read().split()          # apro il file giusto, grazie ...
        a kk

    # prendo intervalli su x e y, nonche il numero di pixel su x e y
    dx = float(content[11])-float(content[6])    # intervallo su x
    x_dim = int((float(content[-5])-float(content[6]))/dx+1)    # ...
        lunghezza asse x in pixel
    dy = float(content[12])-float(content[17+x_dim*5])    # intervallo ...
        su y
    y_dim = int((float(content[7])-float(content[-4]))/dx+1)    # ...
        lunghezza asse y in pixel
    tot_dim = x_dim*y_dim                # N tot di pixel
    deffe = np.zeros((tot_dim,5))    # matrice con tutti gli ...
        spostamenti ordinati: inizializzazione

    if ite == 0:
        dispX = 0
        dispY = 0

    elif ite < 9: # primo giro
        alpha = ite*np.pi/4
        dispX = round(np.cos(alpha))
        dispY = round(np.sin(alpha))

    elif ite < 25: # secondo giro
        alpha = (ite-8)*np.pi/8
        dispX = round(np.cos(alpha)*2.2)
        dispY = round(np.sin(alpha)*2.2)
```

```
29
30
31      elif ite < 49: # terzo giro
32          alpha = (ite-24)*np.pi/12
33          dispX = round(np.cos(alpha)*3.6)
34          dispY = round(np.sin(alpha)*3.6)
35          if abs(dispX) > 3:
36              dispX = dispX/4*3
37          if abs(dispY) > 3:
38              dispY = dispY/4*3
39
40      else: # quarto giro
41          alpha = (ite-48)*np.pi/16
42          dispX = round(np.cos(alpha)*5)
43          dispY = round(np.sin(alpha)*5)
44          if abs(dispX) > 4:
45              dispX = dispX/5*4
46          if abs(dispY) > 4:
47              dispY = dispY/5*4
48
49      # Composizione di matrice con tutti gli spostamenti, matrici di ...
            U,V dal file .txt
50      for i in range(0,tot_dim):  # leggo tutto il file .txt, vado ...
            nella zona di ogni punto
51          for j in range(0,5):    # leggo tutta la zona di ogni punto
52
53              pos = i*5+j+6       # posizione esatta per ogni informazione
54              deffe[i,j] = content[pos]   # carico l'info sulla matrice ...
                    completa
55
56      for i in range(0,y_dim):            # giro sulle righe
57          for j in range(0,x_dim):        # giro sulle colonne
58
59              pos = np.array(i*x_dim+j)   # definisco la posizione ...
                    esatta su deffe
60              if deffe [pos,4] == 0:
61                  u_disp[ite,i,j] = deffe [pos,2] + dispX # piazzo i ...
                        dati in una matrice solo delle u
62                  v_disp[ite,i,j] = deffe [pos,3] - dispY # piazzo i ...
                        dati in una matrice solo delle v
63                  check[ite,i,j] = 0
64
65  u_sum = np.zeros((y_dim,x_dim))
66  v_sum = np.zeros((y_dim,x_dim))
```

```
67
68 counter_ebos = 0
69
70 for i in range(0,y_dim):              # giro sulle righe
71     for j in range(0,x_dim):          # giro sulle colonne
72         cc=0
73         for kappa in range(0,n_max):
74             if u_disp[kappa,i,j] != 0:
75                 u_sum[i,j] = u_sum[i,j]+u_disp[kappa,i,j]
76                 v_sum[i,j] = v_sum[i,j]+v_disp[kappa,i,j]
77                 cc = cc+1
78         if cc < 5:
79             counter_ebos = counter_ebos+1
80             u_sum[i,j] = 100
81             v_sum[i,j] = 100
82         else:
83             u_sum[i,j] = u_sum[i,j]/cc
84             v_sum[i,j] = v_sum[i,j]/cc
```

List of Figures

List of Tables

List of Symbols

Variable	Description	SI unit
ρ	Density	kg/m^3
n	Refraction index	-
L	Loschmidt's number	m^{-3}
e	Euler's number	-
m_e	Mass of electron	kg
Mm	Molar weight	kg/mol
v	Resonance frequency	s^{-1}
f	Oscillator strength	-
K	Gladstone-Dale constant	-
ϵ	Deflection angle	rad
n	refractive index	-
ΔX	Horizontal ray displacement on the screen	m
ΔY	Vertical ray displacement on the screen	m
ζ_1	Text object end position	m
ζ_2	Text object start position	m
ϕ	Wave phase	-
λ	wavelength	m^{-1}
n_∞	Refractive index of test field	-
z_D	Distance between background and Schlieren object	m
z_D	Distance between background and Schlieren object	m
z_A	Distance between Schlieren object and cameralens	m
z_B	Distance between background and camera lens	m
z_i	Distance between camera lens and camera sensor	m

Variable	Description	SI unit
b	width of the Schlieren object	m
M	Camera magnification factor	-
$u_{disp}^{i,j}$	Measured displacement along x direction for cIA "i,j"	*sensor pixels*
$v_{disp}^{i,j}$	Measured displacement along y direction for cIA "i,j"	*sensor pixels*
$u_{eff}^{i,j}$	Theoretical displacement along x direction for cIA "i,j"	*sensor pixels*
$v_{eff}^{i,j}$	Theoretical displacement along y direction for cIA "i,j"	*sensor pixels*
$e_u^{i,j}$	Square error along x direction for cIA "i,j"	*sensor pixels*^{2}
$e_v^{i,j}$	Square error along y direction for cIA "i,j"	*sensor pixels*^{2}
$e^{i,j}$	Mean displacement error for cIA "i,j"	*sensor pixels*
N_x	Number of cIA along x direction	-
N_x	Number of cIA along y direction	-
e_{tot}	Mean displacement error for displacement field	*sensor pixels*
Φ	Mixture equivalence ratio	-
d	Particle diameter	*monitor pixels*

Ringraziamenti

Desidero innanzitutto ringraziare il professor Fabio Cozzi, per avermi dato la possibilità di svolgere questo lavoro e per avermi costantemente seguito nonostante spesso fossi "un po' distratto".
Poi mi tocca ringraziare Francesca, per avermi tenuto compagnia in laboratorio in questi mesi, chiaccherando nei tempi morti, giocando a ping-pong in pausa pranzo e aiutandomi quando facevo errori grossolani. Insieme vorrei ringraziare anche Alejandro, Gabriele, Marco e Carla, per i mesi in laboratorio trascori assieme.
Vorrei poi ringraziare la mia famiglia, in particolare mia madre per aver subìto lezioni non richieste di aerodinamica a tavola, e mio padre perché alla fine è lui quello che ha pagato la retta.
Un grazie va a Beatrice, per aver reso più leggere le mie giornate in questi ultimi due anni e per non avermi insultato quando provavo a spiegarti dei princìpi fisici.
Ringrazio poi i miei amici del Poli, Teo, Christian, Lucio, Ste e Richi Comple, e i miei amici fuori dal poli, Richi, Leo e tutti gli altri che però sono tantissimi e non posso elencare per questioni ecologiche di carta stampata, un grande grazie a tutti voi per avermi supportato e sopportato in questi anni (a quanto pare questa formula è d'obbligo nei ringraziamenti).
Infine un grazie a nonno Enrico da cui ho ereditato la mente ingegneristica, a nonna Mirella per la determinazione, a nonna Lidia per la creatività e a nonno Fabio per il senso pratico, caratteristiche che mi sono state indispensabili per raggiungere a questo traguardo.