Computer Vision Tracking Techniques Applied to Vibration Analysis

Thesis · September 2016

DOI: 10.13140/RG.2.2.11772.08325

1 author:

Luis Gomez Camara
National Institute for Research in Computer Science and Control
15 PUBLICATIONS **184** CITATIONS

SEE PROFILE

Máster Oficial en Visión Artificial

PROYECTO FIN DE MÁSTER

Computer Vision Tracking Techniques Applied to Vibration Analysis

Autor: Luis Gómez Cámara

Tutores (Airbus): Antonio Pérez, Gianluca Marsiglia, Daniele Nadalutti

Tutora (Universidad Rey Juan Carlos): Dra. Cristina Conde Vilda

Tesis realizada en las instalaciones de Airbus Group en Manching, Alemania

· Curso académico 2016/2017 ·

Abstract

El análisis de vibraciones, también conocido como análisis modal, es la disciplina que se ocupa de medir y analizar la respuesta dinámica de estructuras y fluidos expuestos a una excitación. Las técnicas tradicionales de análisis modal suelen ser caras, requieren una etapa de preparación compleja o ambas. Un enfoque alternativo que ha demostrado ser válido es el uso de cámaras de alta velocidad en combinación con técnicas de visión artificial. Su coste es más bajo que el de otras técnicas y además no requiere ningún tipo de contacto con la estructura analizada. En esta tesis se han comparado diferentes algoritmos de seguimiento, tanto clásicos como de última generación, evaluando su capacidad para describir el movimiento oscilatorio que caracteriza a las vibraciones. Los tests se han realizado en vídeos de alta velocidad correspondientes a diferentes situaciones experimentales. Se muestra que cuando la amplitud de las vibraciones comprenden varios píxeles dentro del vídeo, la mayoría de los *trackers* son capaces de seguirlas. Sin embargo, para amplitudes cercanas a un píxel, problemas debidos a la falta de resolución espacial de los *trackers* comienzan a emerger. En estos casos, se observa que la adición de una cantidad considerable de ruido blanco puede ayudar a mitigar estos problemas. Cuando las amplitudes se llevan a nivel de subpíxel, la mayoría de los *trackers* fracasan en el seguimiento, con la excepción del algoritmo Median Flow, basado en cálculos de flujo óptico y el cual demuestra de forma consistente su robustez y alto rendimiento en cuanto a resolución espacial se refiere. Teniendo en cuenta los resultados obtenidos, se ha desarrollado un software que implementa Median Flow para el análisis modal de vídeos, donde se puede realizar el seguimiento de regiones seleccionadas por el usuario. El resultado es mostrado en ambos dominios del tiempo y de la frecuencia. También permite crear mapas de color cuyo objetivo es doble: revelar qué partes del vídeo son más apropiadas para realizar el análisis modal y cuáles vibran de forma más energética. Finalmente, se muestra tanto el funcionamiento del software como su rendimiento, utilizando para ello varios ejemplos reales.

Abstract

Vibration analysis, also known as modal analysis, is the field of measuring and analysing the dynamic response of structures and or fluids during vibration excitation. Traditional modal analysis techniques are either expensive, require a complex set up or both. Lower cost, non-contact measuring devices like high-speed cameras in combination with computer vision techniques have been shown to be a valid alternative approach for measuring vibrations in structures. In this work, a number of both classic and state-of-the-art computer vision tracking algorithms are compared and their ability to track the oscillatory movement that characterises vibrations is tested on high-speed videos under different experimental conditions. It is shown that vibrations covering a few pixels in a video can be easily resolved by most trackers, but quantisation problems due to limited spatial resolution start to appear when dealing with vibration amplitudes close to the pixel. In such cases, it seems that adding a fairly large amount of dither to the frames can help to mitigate those problems. In the sub-pixel level most of the algorithms considered start to fail, with the exception of the Median Flow, an optical flow-based tracker that consistently shows its robustness and high spatial resolution performance. An interactive tool implementing this tracker has been created for the purposes of modal analysis. The tool can load a video and perform tracking on specific user-selected regions, showing the outcome in both time and frequency domains. It also permits the creation of frequency color maps whose goal is twofold: to reveal what parts of a video are more suitable for modal analysis and also which ones vibrate more energetically. A few real life examples are put in place in order to show the capabilities of the tool and demonstrate its performance.

Acknowledgments

My acknowledgements go to Airbus Group for funding this project and to my thesis supervisors: Antonio Pérez, Gianluca Marsiglia and Daniele Nadalutti. I would also like to thank Dra. Cristina Conde Vilda from University Rey Juan Carlos for all the administrative work.

I confirm that the work presented within this document is entirely my own. Any inclusion of, and references to works authored by other persons are clearly marked out as such.

This work has been funded by Airbus Defence and Space.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The physical phenomenon of vibration is extraordinary common in Nature, to the point that entire unifying theories of physics like the String theory [1] are based on the assumption that particles and fundamental forces in Nature can be modelled as vibrations of tiny strings.

Far from the sub-microscopic world, on a scale in which we humans can better relate, vibration can be defined as the "periodic back-and-forth motion of the particles of an elastic body or medium, commonly resulting when almost any physical system is displaced from its equilibrium condition and allowed to respond to the forces that tend to restore equilibrium" [2].

Vibrations are present everywhere in our daily life, from an electric toothbrush or a hair dryer to the platform of a train station at peak time. They can get us sick if we happen to be on top of a high building during a storm, or induce the breakage of products that have been subject to them for a long time. Vibrations can also cause noise, which may affect us in different ways, from discomfort to hearing impairment [3].

In engineering, as one would expect, vibrations play a major role so procedures such as condition monitoring and vibration analysis have become standard [4]. The most common areas where these are applied to are related to the identification, reduction or elimination of unwanted modes in order to not only improve a product's quality, but also to avoid premature failure as well as reduce maintenance costs.

In civil engineering, for instance, one can think of a bridge oscillating under the action of the wind, passing-by traffic or even an Earth's tremmor. The characterization of the natural modes of vibration and their sensitivity to effects such as mechanical resonance [5] or aeroelastic flutter [6] can not be omitted at design stage, as it has been

the cause for some of these structures to collapse [7] (see Figure 1.1).

Figure 1.1: Tacoma Narrows Bridge collapsed in November 1940 due to aeroelastic flutter caused by wind conditions.

In the automotive industry, NVH stands for Noise, Vibration and Harshness and is an important field of research [8, 9]. Figure 1.2 shows a typical setup where NVH tests are performed. The main sources of vibrations and noise in cars originate in the actual

Figure 1.2: Example of an acoustically treated NVH testing chamber.

road and its interaction with the tyres and the suspension, but also in the engine and power train vibrations. Others factors like the wind, antennas etc. also affects in one way or another. NVH is taken very seriously by car manufacturers, as it has a great impact on user comfort and therefore on the overall quality image of a vehicle [9, 10].

Being able to analyse the vibration characteristics of the different parts of a vehicle is paramount at both design and testing stages.

Another area of engineering where vibration analysis is crucial is in the aerospace industry. Aircrafts are subject to all sort of vibrations due to aerodynamics, acoustics and the propulsion systems themselves. In the latter, as described in [11], engines have a characteristic frequency spectrum or "vibration signature" that must be measured before delivery. Departure from that signature provides valuable diagnostic information that can be used to detect poor functioning or potential future failures.

Other than caused by the engines, there are basically three types of vibrations occurring in an aircfraft [12]:

- *Buffet*: It consists of a disturbance of the air flow, which is random in nature and can be caused by some of the aircraft components like the aerodynamic breaks or the flaps but also by external factors like the air turbulences that most of us have experienced when flying.

- *Noise vibrations*: They are caused by parts of a plane moving back and forth very rapidly under the action of the wind. Noise is also originated by the vibration of the engine/s.

- *Aeroelastic flutter*: It is not so common but can be very dangerous. It occurs when the energy from the air flow cannot be absorbed by structures such as the wings, so their natural frequencies are amplified to a magnitude that can cause failure.

All these types of vibrations must be taken into account (specially the aeroelastic flutter) during the developing process, so again engineers need to do all sort of tests for the identification and characterisation of the vibration frequencies.

Vibration analysis can be applied to many other disciplines such as structural dynamics, environmental engineering, fatigue analysis, vibration monitoring, acoustics, power plants, medical science, industrial production and transportation to cite some [13].

1.1 Methodologies used for vibration analysis

The field of vibration analysis, also known as modal analysis, is formally defined as "the field of measuring and analysing the dynamic response of structures and or fluids during vibrational excitation" [14]. Typically, both analytical and experimental approaches are used. Even though this thesis is primarily concerned with vibration analysis using computer vision techniques, in order to provide some background a brief overview is presented on the methods that are most commonly used in industry.

1.1.1 Analytical or model-based analysis

Analytical approaches to modal analysis are mainly based on the *finite element method* (FEM) [15, 16] and can be implemented using computer-aided drafting (CAD) tools [17]. The basic concept of the finite element approach is to subdivide a large complex structure into a finite number of simple elements, such as structure elements, quadrilaterals, or triangles. For a given property of interest, the otherwise complex differential equations are solved for the simple elements and assembled into a global matrix that models the entire problem and that can be solved numerically [18]. The properties being modelled depend on the application and can be those such as component displacements, modal analysis, velocity fields, stress-strain, electric potential fields, heat transfer, etc.

The main goal of FEM is, apart from greatly reduce costs, to understand, predict, optimize, and control the design or operation of a device or process, which can be clearly oriented towards vibrational analysis. Figure 1.3 shows an example of a radial flow impeller, more commonly known as turbomachines. They are widely used in industry to increase or decrease the pressure of a fluid. During the development stage, vibration resonances that can lead to failure must be avoided and FEM provides a powerful tool for this purpose.

FEM is largely used in industry and provides good representations of the structures under study. However, it is just an approximation tool that can lead to errors when the structures are very complicated. As explained in [19], the sources of errors can be as follows:

- Inaccuracy in estimation of the physical properties of the structure
- Poor quality of mesh generation and selection of individual shape functions

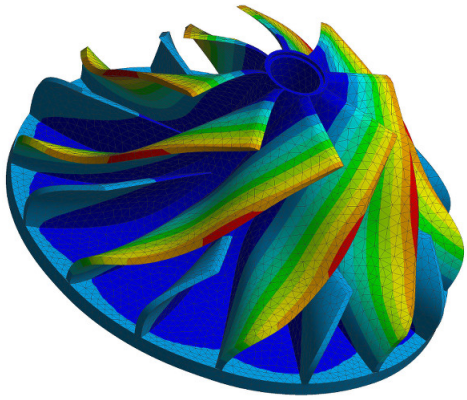


Figure 1.3: Vibration modal analysis of a radial flow impeller (image courtesy of MRIGlobal).

- Poor approximation of boundary conditions
- Omission or poor modelling of damping properties of the system
- Computational errors which are mainly due to rounding off

The interpretation of an FEM analysis must ultimately rely on the specialist carrying out the analysis and not solely on the software.

1.1.2 Experimental analysis

The ultimate goal of vibration analysis is to find the transfer function or Frequency Response Function (FRF) of the structure or machinery of interest, which describes the input-output relationship between two points on the structure as a function of an excitation frequency [20].

Obtaining the different vibration modes (and their shape) and summing them up leads to the FRF, so the aim is to find these modes. This is normally achieved by physically exciting the structure at several frequencies and then measuring how it reacts to them. The most commonly used excitation methods are *impact hammer* and *shakers* [21]. The first consists of striking with a special hammer, containing a force load measuring unit (see Figure 1.4), the structure of interest so as to excite its modes. The second is a more elaborated device that contains, as its name suggest, of a mobile part that can shake the structure of interest at both the desired frequencies (typically following a sine sweep) or in random fashion.

Figure 1.4: Impact hammer instrumented with a load cell.

Impact hammer is more suitable for small and less heavy structures. Its main advantage is the easy of use and its adaptability to complicated testing environments. In the ideal case, where the impact duration approaches to zero, all modes of vibration are excited with the same amplitude. In real situations, however, as the impact time gets longer, the range of covered frequencies shrinks and therefore the applicability is reduced.

In cases where one is dealing with delicate surfaces, needs a wider frequency range or simply the structure is too large or heavy, impact hammer may not be the best excitation method. Artificial excitation in theses cases can be provided by one or more shakers.

Independently of the chosen excitation method, there exist a number of techniques that can be used in the actual measurement of the vibrations. In the following, a description of the most relevant is provided. They have been broadly separated into contact and non-contact methods.

1.1.3 Contact methods

1.1.3.1 Accelerometers

The contact measuring methods can be considered as classic and are dominated by transducers operating as sensors. They are mainly known as accelerometers [13] and measure both static (like gravity) and dynamic (movement, vibration, etc.) accelerations. In figure 1.5, an experimental setup is shown where a car's door is being tested. The setup contains an excitation source that in this case is a shaker and a number of accelerometers carefully glued at positions of interest. Each accelerometer sends vibration information of that specific location, which later will be used in the frequency

analysis to obtain the vibration modes.

Figure 1.5: Modal testing being performed on a car's door by means of a shaker and several accelerometers. Picture adapted from Wikipedia.

Some of the disadvantages of this classical approach are that is time-consuming and costs tend to be high, as it requires expensive specialized equipment, a more or less complex setup, and highly qualified personnel. In addition, the accelerometers themselves (and the associated wires) are physical entities, so they can interfere with the response of the measured structure by dumping the vibrations to certain degree.

1.1.4 Non-contact methods

From all the non-contact measuring methods, optical ones are the most widely used so descriptions of the most relevant are given below. Other less important methods include acoustic, proximity detection and stress measurement. The interested reader is referred to [22] for further information on these particular techniques.

Compared to accelerometers, non-contact or non-intrusive optical methods have the advantage of not interfering with the resonance frequencies of the measured structure, which is specially useful when measuring lightweight structures. In addition, they can be directed to locations of difficult access (like cavities) or that simply are too small (or too hot) to place an accelerometer.

1.1.4.1 Laser Triangulation

Laser triangulation is a relatively cost-effective technique consisting of a solid-state laser that projects a beam onto the vibrating target (see Figure 1.6). The reflection from the target is focused and projected back on a standard optical sensor like a CMOS. Depending on the distance of the target, the reflection beam will hit different areas of the sensor and this fact is used to calculate the distance of the target at each time.

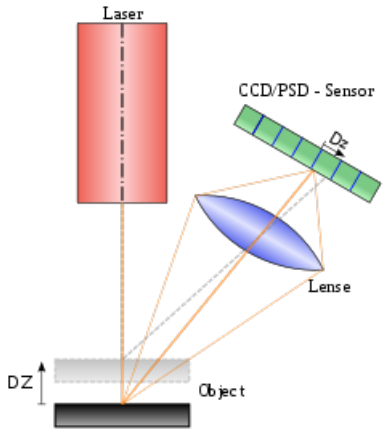


Figure 1.6: Laser triangulation principle shown for two target positions (figure adapted from wikipedia).

The resolution of laser triangulation devices depends on the spatial resolution of the optical sensor and typically can be in the order of 10μm and cover a frequency range up to 10kHz. For higher frequencies, laser doppler vibrometry (LDV) is however a more appropriate technique [22].

1.1.4.2 Laser Doppler Vibrometry

LDV represents the most commonly used non-contact vibration measuring technique and often a suitable alternative to accelerometers [22]. It is based on the Doppler effect of coherent laser light, which conceptually is similar to the more familiar Doppler sound effect. The laser light source, of known wavelength, is directed towards the target surface and the wavelength of the reflected beam is measured by means of interferometry [23] with a reference beam. Depending on whether the sample is moving towards or away from the source, its wavelength will increase or decrease, respectively.

The resolution of LDV is of around 1 μm and the measurable frequency range is 1Hz-1MHz [24, section 3.6.2]. The major inconvenients with this type of device are

Figure 1.7: Example of a LDV device (OFV – Modular Laser Vibrometer from Polytec).

the high price [25] and a decreased performance in situations where the beam is not perpendicular to the analysed surface. In addition, the equipment tends to be bulky.

1.1.4.3 Holographic Vibrometry

This technique is very similar to LDV in that it uses a target and a reference beam and the principles of interferometry. The target beam is used to illuminate the vibrating structure and the light scattered from it is mixed up with the reference beam on a holographic plate, producing an interference pattern that is stored at each time. From them, the 3D displacement of the target can be resolved afterwords by reconstructing the light wave [24].

As opposed to LDV, the holographic approach has the advantage that the vibration of the entire structure, and not just a point, can be resolved at a time. Also, it has an extremely high-resolution (less than one nanometer) and its frequency range is 20Hz-50KHz. It is however a very expensive technology, which certainly limits its use.

1.1.4.4 Computer Vision

A less standard approach that nonetheless has been shown to be a valid one (see Section 2.2) is the computer vision approach to modal testing. This is in fact the main subject of this thesis and therefore next chapter is entirely dedicated to its introduction.

1.2 Scope and Structure of this Thesis

1.2.1 Scope

The main goals of this work are the following:

1. To compare an array of both classic and state-of-the-art computer vision tracking algorithms and to establish their suitability for vibration analysis applications.

2. To select the most appropriate algorithm based on the comparison results and create a basic interactive, proof-of-concept tool for the analysis of vibrations in high-speed videos of structures.

1.2.2 Structure

In the current chapter, a general introduction to the field of vibration analysis in engineering has been provided, commenting on some of the most relevant problems different industries must deal with. The most commonly used methodologies in the field of vibration analysis, both theoretical and experimental, have also been introduced.

Chapter 2 presents the computer vision (CV) approach as a tool to tackle the problem of vibration analysis and previous attempts to use CV in this area are reviewed and discussed. Also, a number of tracking algorithms that can potentially be used in this context, and that have been tested throughout this work, are explained in some detail.

Chapter 3 is the core of this work, where a methodology is devised that will serve to test and characterise the chosen tracking algorithms and to determine which of them perform the best in the situations of interest.

Chapter 4 presents a proof-of-concept GUI that can be utilised for vibration analysis in high-speed videos. Using the GUI, a number of real-life examples are explored.

Chapter 5 is devoted to conclusions and future work.

Chapter 2

Computer Vision

In the previous chapter, computer vision was included as one of the non-contact, optical alternatives in modal analysis. Where is, however, the link between these two fields? Once we have a recorded video of a vibrating structure, how do we extract the required information from it? The answers to these questions rely on the ability of an algorithm to follow the oscillatory movement of vibrating structures through a sequence of images. This brings us to one of the most relevant areas of computer vision: motion analysis.

As a discipline, motion analysis started in the early 1970's and still remains today a major open research topic [26]. The most typical task of motion analysis is probably the problem of video tracking. The literature on this subject is so extensive that even a short review would be outside the scope of this work. Several surveys on video tracking can be found in [27, 28, 29, 30, 31].

The main goal of this thesis is to investigate and evaluate a number of standard, well-known computer vision video trackers (described in Section 2.2) in the context of vibration analysis. In other words, we want to explore how standard trackers perform when following the oscillatory movement of vibrating objects and whether this can be used for modal analysis.

In the following, a review of the relevant work made to date on this specific subject is provided. The remaining of the chapter is then devoted to describe the tracking algorithms under investigation.

2.1 CV applied to modal analysis: Literature review

Early attempts to use computer vision techniques to obtain the modes of vibration of a structure were concentrated in the area of civil engineering. The work of Olaszek [32] was a successful attempt to measure the displacement of a chosen point in a bridge structure. This type of displacement measurements are standard in bridge testing and normally carried out by means of accelerometers. The locations, however, are not always easy to access so his method represented a clear advantage by being contactless. The author recorded bridge points where he placed patterns consisting of black crosses on white background, and obtained their edges from the images by simple pixel gray-level differentiation. He then introduced a form of interpolation to get sub-pixel accuracy in the cross edge detection. Damped sinusoids caused by load of the bridge with vehicles were measured and the vibration modes extracted from them, showing very good agreement with experimental results obtained from accelerometers.

Patsias et al. [33] used a high-speed camera at 600 frames per second (fps) and the wavelet transform (for edge detection) to characterise the spatial displacement and the mode shapes of a cantilever beam, using that information to create a damage detection system. The major advantage of their method was the large number of discrete points used to describe the modes, unlike in more classical approaches where a limited number of sensors are used.

Lee et al. proposed in 2006 another vision-based system also to remotely measure dynamic displacement of bridges. It represented an improvement with respect to previously proposed systems in that the measurements were characterised by a high spatial resolution and the system was highly cost-effective and easy to implement.

Caetano et al. [34] devised a system to monitor the cable vibrations of the Guadiana Bridge between Spain and Portugal. The camera was located at 850 m from the bridge and the frame rate was 25 fps. In order to follow the displacement of the cables, the authors employed block-matching correlation and optical flow methods. In their paper, they explain how the latter is more suitable for situations where the amplitude of the motion is very small compared with the size of the images, in the order of only a few pixels (see Appendix A for an explanation on resolution of optical flow methods). Results showed peak frequencies in agreement with a previous cable measurement campaign using accelerometers.

More recently, increasingly sophisticated computer vision techniques have been used

to identify vibration mode shapes of structures. In [35], the authors used high-speed cameras and *phase-based motion magnification* [36] to calculate the displacement signal from a vibrating cantilever beam. From the peaks in the Fourier transform, they isolated the frequencies of the modes and performed, for each one of them, a narrow temporal bandpass filtering that allowed the estimation of the modes shape. They compared their results with data from accelerometers and laser vibrometry. Only the first four modes (from a total of eight) were detected by the camera but comparison of frequencies and mode shapes against experimental data showed very good agreement.

In [37], Davis et al. showed that it is possible to obtain valuable information from videos on the properties of different objects subject to very small vibrations, often even invisible to the naked eye. They performed experiments on clamped rods and hanging fabrics which they excited using sound emitted by a loudspeaker. For the clamped rods they reasonably recovered the theoretical vibration modes whereas for the hanging fabrics, their results evidenced a clear correlation between the vibration spectrum, characterised by wide resonance bands, and fabric stiffness and area weight.

The research group supervising this Master thesis at Airbus has also been working in the analysis of vibrations of structures using computer vision techniques.

All the works discussed above show that cameras combined with computer vision can be a real alternative to the more traditional (and costly) measuring techniques used in industry.

2.2 Trackers

The core of the trackers used throughout this work are those included in version `3.1.0` of OpenCV [38]. They consist of five state-of-the-art tracking algorithms: MIL, BOOSTING, MEDIANFLOW, TLD and KCF, all of which are easily implemented through a common interface consisting of three main functions that create, initialise and update the trackers after each successive frame.

In addition to these, four extra algorithms have also been implemented for completeness: CMT, Template Matching, MeanShift and CamShift. CMT is a fairly new algorithm and therefore it is not yet included in OpenCV. All the rest have their own functions in OpenCV and can be considered as classic trackers. A description of the nine algorithms is provided below.

2.2.1 MIL

MIL stands for Multiple Instance Learning. The MIL tracking algorithm implemented in OpenCV is based on [39], whereas the MIL approach is presented in [40]. This algorithm trains a classifier in an on-line fashion in order to separate the target to be tracked from the background. The state of the tracker in the current frame is used to extract positive and negative samples for the learner. What makes it different from traditional supervised learning trackers is the way it uses the samples for learning.

In traditional supervised learners, small inaccuracies in the tracker may lead to incorrect labels for the training samples, which in turn can make the tracker to perform poorly or to drift [41]. In MIL, these problems are overcome by updating the appearance model of the tracked object with a set of image patches, rather than individuals, even though it is not known which one of them locates the object more precisely. This set of patches represent a bag, which is the one carrying the positive or negative label for the learner. A bag is labelled positive if at least one the containing patches is positive, and negative if none of them are. By using several patches in a labelled bag, the learner can decide which patch is the most accurate, resulting in a more robust tracking.

Apart from the appearance model described above, the MIL algorithm also contains an image representation and a motion model. The former is based on Haar-like features [42, 43] extracted from each patch. The latter consists on setting a radius s from the tracker at time $t-1$ and making the position of the tracker at time t equally likely within this radio. The exact position is then obtained from the patch with maximum probability.

2.2.2 BOOSTING

The implementation of this algorithm in OpenCV is based on [44] and uses an on-line version of the AdaBoost (Adaptive Boosting) algorithm [45]. AdaBoost basically fits a classifier with the initial data and then adjusts the weights of incorrectly classified instances so that new versions of the classifier focus more on difficult cases.

The tracking step itself is based on the classical approach of template matching [46], which takes into account problematic aspects such as change in pose relative to the viewing camera, changes in illumination relative to light sources, and partial or full occlusions.

The BOOSTING algorithm is initialised by assuming the initial target of the first

frame as positive and choosing background regions around the target, and of the same size, as negative. With these samples, AdaBoost is run and a stable initialisation model created. In the next frame, based on the model, a binary classifier is used to create a map of confidences for a region of interest. The maximum of that map is employed to locate the tracker in the current frame. A motion model can be used to reduce the search region.

The last step consists on updating the classifier to make it robust to potential changes in target appearance or differences in background.

2.2.3 Median Flow

As it will be shown in the next chapter, this is the tracker that provided better results as a whole and therefore it will be described here in some more detail.

Let us start the discussion of this algorithm with a brief introduction to the Lukas-Kanade differential method for optical flow estimation [47], as this is the method used by the Median Flow tracker and also one of the most widely used methods in computer vision [27, 48]. The reader is referred to the work of Barron et al. [49], where the authors quantitatively compared different optical flow methods, emphasising measuring accuracy. Their results showed that the Lukas-Kanade [47] differential method was one of the most accurate.

2.2.3.1 Lucas-Kanade optical flow estimation

In video sequences, optical flow is the velocity field associated to the change in the intensity levels of the pixels in consecutive frames. It is directly related to the apparent pattern of movement in the scene. Hence, optical flow can be used to detect the movement of objects and is useful in motion-based segmentation and tracking applications.

It is computed using an equation that sets the brightness constraint, which assumes that the brightness of pixels corresponding to a given object does not change in consecutive frames:

$$I_x u + I_y v + I_t = 0, \tag{2.1}$$

where u and v are, respectively, the horizontal and vertical components of the velocity of each pixel (the unknowns), I_x and I_y are the derivatives of the intensity in those directions and I_t is the intensity temporal derivative. In a nutshell, what Eq. 2.1 expresses is that if there is a change in intensity for a given pixel from one frame to the

next, it is due to a movement of the object to which that pixel belongs.

There is, however, a problem with Eq. 2.1: it is a single equation with two unknowns. This is known as the aperture problem [50] and the Lukas-Kanade method deals with it through the introduction of a local constraint. The constraint consists of assuming that the velocity in small regions of the image stays constant, i.e., it tries to minimise Eq. 2.1 by least squares. This involves choosing a local region surrounding each pixel and minimising on each of them, then performing the same operations for the entire image or for the pixels of interest. In this fashion, a system of two equations with two unknowns is obtained for each pixel, which can be solved using the desired technique.

The general expression for the calculation of velocities is the following:

$$\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} \sum I_x^2 & \sum I_x I_y \\ \sum I_x I_y & \sum I_y^2 \end{bmatrix}^{-1} \begin{bmatrix} -\sum I_x I_t \\ \sum I_y I_t \end{bmatrix}, \tag{2.2}$$

where the sums run over the pixels surrounding each central pixel. Once the velocities are known, one can follow the trajectories of the pixels and therefore the algorithm can be thought as a tracker.

2.2.3.2 Median Flow tracker

The Median Flow tracking algorithm is described in the work of Kalal et al. [51]. It is based on two ideas: (i) the Lukas-Kanade differential method explained above and (ii) the forward-backward (FB) consistency assumption. The latter states that correct tracking should be independent of the direction of time flow.

Median Flow consists of applying the pyramidal Lucas-Kanade optical flow algorithm [52] to a number of points. The points are initialised on a rectangular grid (square in this work) within an initial bounding box β_t, which is selected from the first frame of a sequence. The tracker then calculates the flow between images at t and $t+1$ and predicts the locations of the points at $t+1$. In OpenCV, the default number of points per grid is set to 100 (10x10).

The quality of the predictions on each grid point is then evaluated using an error measure that is directly related to the FB assumption. This consists on calculating the trajectory of each grid point backwards in time and then comparing it with the forward trajectory. A distance measure such as the Euclidean distance or the Normalised Cross-Correlation (NCC) is computed between the two trajectories, from where the error measure is obtained. Arranging the points by their error in increasing order and

selecting only the first 50% leads to a subset of points (and gives the name to the tracker), from where the position of the bounding box β_{t+1} is estimated.

The scale of the new updated bounding box β_{t+1} is estimated using a measure of the change in distance between pair of points in the previous and current frame, then using only the 50% with less change to make the estimation. It is important to note that the OpenCV's Median Flow code was slightly altered in this work in order to disable this last scaling feature, as it showed to be unstable in many tests and a fixed bounding box proved to work better.

2.2.4 TLD

Also developed by Kalal et al. [51], the novelty of this tracking algorithm is that it separates the tracking task into the tasks of Tracking, Learning and Detection.

In the detection task, each frame is treated as independent and a full scanning of the image is performed to find appearances of the objects that have been learnt in previous frames. In order to do this, a series of cascade stages is performed on a large number of bounding boxes (50k for an image of size 240x320). These include a patch variance filter, an ensemble classifier and a nearest neighbour classifier whose output is typically of only a few hundred bounding boxes.

The tracker task consists of a Median Flow tracker with added functionality that can detect if the object gets out of the image or is fully occluded, in which case the tracker does not return any bounding box. The bounding box output from the tracker and the detector is combined to give a single result for the TLD algorithm. This is accomplished by means of a so-called integrator.

The learning task initialises the detector in the first frame and also updates it in subsequent frames, using for that a combination of a P-expert and an N-expert. The former improves the generalisation power of the detector. The latter is used to generate negative samples once the location of the object is known, with the aim to improve the discrimination power.

The combination of the three tasks makes the TLD tracker specially suitable for challenging trackings, such as those performed on long sequences, or where the objects change appearance frequently, are partially or totally occluded or move in and out of the camera view. The complexity inherent to the algorithm makes it however rather slow as compared for instance with a much simpler Median Flow. As it will be seen in next chapter, the TLD implementation included in OpenCV does not seem to work

properly and exhibits instabilities in most of the tracking tests performed.

2.2.5 KCF

KCF stands for Kernelised Correlation Filters. This tracker is based on the work of Henriques et al. [53] and fits, alongside MIL, BOOSTING and TLD, the tracking model known as tracking-by-detection (see for instance [54, 55] for two nice examples using this approach).

The main idea in tracking-by-detection is to learn and adapt frame by frame the changes of the initial target in an online fashion, using for that purpose standard machine learning classifiers. The samples used to train the classifiers are obtained from a neighbourhood around the target and typically consist of windows of the same size as the latter.

In the case of KCF, a dense set of samples is used. This differs from other methods where, due to computational cost, only a small number of samples is used. KCF exploits a property known as *circulant structure* of matrices, which is induced by the sole fact of taking subwindows of an image. In their paper, Henriques et al. show that the circulant structure permits the used of the Fast Fourier Transform (FFT) and several types of kernels in order to incorporate the information from all the samples without having to iterate through all of them, hence making the learning process orders of magnitude faster (see for instance Table 3.1 in next chapter).

2.2.6 CMT

Code for the CMT implementation has been extracted from `http://www.gnebehay.com/cmt` and adapted as needed. It is based on the work described in [56, 57].

CMT stands for Consensus-based Matching and Tracking. It uses a keypoint-based representation of the objects to track. Firstly, a matching of keypoints based on a Hamming distance is performed between two consecutive frames. Secondly, a pyramidal Lucas Kanade optical flow [52] is estimated on the keypoints to obtain their displacements in next frame. Then both matched and tracked keypoints are fused, discarding the tracked ones that have a correspondence with a matched one.

The keypoints resulting from this fusion act as voters in the localisation of the object. The validity of the voting is assessed through a consensus that consist on a hierarchical agglomerative clustering [58], where data is structured in a hierarchical

fashion according to a dissimilarity measure and a threshold that is applied in order to filter out the votes with less consensus.

The authors claim that CMT performs accurately under changes in translation, scale and orientation and it is robust to partial and full occlusions and well as changes in appearance.

2.2.7 Template Matching

Template matching (see for instance the on-line OpenCV documentation) consists of creating a template image o patch from an object of interest and then compare it against different regions of a source image where the patch needs to be found. This is normally achieved by sliding the patch over the source. A metric measure, for instance a cross correlation function, is calculated for each new sliding position. The location with higher matching probability is then identified as the location of the patch in the source.

The function `matchTemplate` in OpenCV compares a template against overlapping image regions in a source and offers a number of metrics for the comparison. In the same manner that one can compare a patch in a source image, this can be done in a sequence of frames, and therefore the patch can be followed. This is the idea behind template matching trackers.

2.2.8 MeanShift

MeanShift as such is a pattern recognition procedure developed in 1975 by Fukunaga et al. [59] that is used to find the modes of a distribution without the need to compute over all the data. Its application to the field of computer vision was made by Comaniciu et al. [60]. It is an iterative process consisting of three main steps:

1. A window size and location is selected.

2. Based on the selected window and the density distribution of a set of pre-segmented points (using for instance techniques such as color filter, background subtraction, etc.), the mean shift vector is calculated, which points from the center of the window towards the direction of maximum increase in the density. Within the window, a weighting function may be applied to give more weight to points closer to the center of the window.

3. The location of the window is updated according to the calculated mean shift vector and the process iterated until convergence (vector magnitude smaller than a threshold).

In the context of tracking, an image is typically converted to HSV format and a model histogram created from the hue of an initial bounding box. With this model, a back projection image is created based on the next frame, which basically gives the probability of the new frame describing the model. The MeanShift algorithm is then applied to the back projection image to find the location of the target (local maximum) in the new frame.

2.2.9 CamShift

This tracker is an extension of MeanShift (Continuously Adaptive MeanShift) in that it adapts the size and orientation of the bounding box according to changes in the target. It was presented by Bradsky in [61] and consists on applying a standard MeanShift first, followed by a second MeanShift where the new scaled/rotated search window is used, repeating the process until convergence.

2.3 Hardware and software

In this section, the hardware and software resources available throughout this project are are given.

2.3.1 Hardware

2.3.1.1 Computer

The specifications of the machine utilised in this work for both software development and computer vision experiments are the following:

Operating system:	Ubuntu 14.04 LTS
Memory:	15.6 GiB
Processor:	AMD FX(tm)-8150 Eight-Core Processor x 8
Graphics:	GeForce GTX 660 Ti
OS type:	64-bit

2.3.1.2 Camera and illumination

Most of the videos used in this work were recorded using a state-of-the-art high-speed camera and high quality lenses and illumination sources. The data sheet for the camera is shown in Figure 2.1 and the specifications for the lens in Figure 2.3. The light source shown in Figure 2.2 was also used for illumination purposes.

2.3.2 Software

All the software developed during the realisation of this thesis has been made in the Python 2.7 programming language and using the PyCharm integrated development environment, version COMMUNITY 2016.2. Key Python libraries were NumPy, SciPy and Matplotlib. The thesis itself has been written in LaTeX using Texmaker version 4.5. Additional software used for image edition was GIMP. Audacity 2.0.6 was used for audio recording and editing.

INNOVATION IN MOTION

X-STREAM1440P SPECIFICATION SHEET

The IDT X-Stream Cameras offer continuous frame streaming via the PCI Express® 2.0 x4 interface with a sustained transfer speed of 1.75 GB/sec. This flexible design has been implemented around two CMOS sensors with Global Shutter: the 720p delivering over 1,700 fps at full resolution (1280×720 pixels) and 1440p delivering over 400 fps at full resolution (2560×1440 pixels). Advanced features include Frame to frame Auto-exposure and Motion Trigger and Double-exposure for PIV users. The especially tuned Motion Monitor application operates the cameras in the Windows, Linux or Mac OSX environments, with features that include always-on live, record while saving and on-demand playback from disk. The cameras are especially suited for a variety of uses ranging from industrial and packaging inspection, microscopy, media/cine including special effects, traffic control and surveillance.

KEY FEATURES

Maximum Resolution	2560 x 1440
Maximum FPS @ Maximum Resolution	425 fps
Maximum FPS	81,050 @ 2560 x 8
Minimum Exposure Time	1µs
Sensitivity ASA/ISO	15000 ASA mono 5000 ASA Color
Power Requirements	7-12 VDC
Operating Temperature	-40+40 °C / -40+104 °F

SENSOR

Sensor Type	CMOS – Proprietary
Sensor Size	36 x 36 mm
Sensor Format	1.3 inch
Pixel Size (micron)	7 x 7 um
Pixel Depth	24-bit mono 8-bit color

INPUTS

Trigger	TTL & Switch/Circular buffer with on-camera or software trigger
Sync	Phase-lock TTL

OUTPUTS

Sync	Frame sync / Strobe

FEATURES

Approx. Size	65 x 116 x 18 mm (W x H x L)
Approx. Weight	0.24 kg or 0.53 lbs
Shock/Vibration Rating	Shock: 100G /Vibration: 40G – All axes
Mount	C-Mount standard, F & PL Adaptor optional

SOFTWARE

Motion Studio	Windows 32/65
Motion Inspector	Windows 32/64 – MAC OS X – Apple iOS
Plug-ins/SDK	SDK, LabVIEW™ or MatLab®
File Formats	Proprietary RAW
On-the-fly Conversion	TIF, BMP, JPG, PNG, AVI, MPG, TP2, MOV, MRF, MCF

COMMUNICATION

Ethernet	100/1000BaseT

Specifications are subject to change without notification. | Data accurate as of 26 Apr. 2016 | Please reference our website for updates: http://www.idtvision.com
- 1 / 1 -

Figure 2.1: Specifications for the camera used throughout this work.

veritas

SPECIFICATION SHEET

Constellation

Model 120E

Constellation 120E offers intelligent digital illumination in one economical package. Support for continuous and strobe modes as well as availability in multiple color temperatures and beam angles makes this the perfect multipurpose light. The C120E's sync-in port accepts incoming pulses from a camera or signal generator and sync-out allows for daisy-chaining of multiple luminaires from the Veritas Constellation Series.

KEY FEATURES

White Light Color Temp. (Standard)	6200K
Estimated LED Lifetime	40,000 Hours
Operating Temperature	From -40°C up to 70°C
Maximum Pulse	100 kHz
Minimum Pulse	2 µsec Rise/Fall Time 500 nsec

PHOTOMETRIC

Luminous Flux

Color Temp.	*Continuous*	*Strobe*
6200K	*12,700lm*	*22,000lm*

White Light Color Temp. (Options)	2700K, 3000K, 3500K, 4000K, 4500K, 5,000K, 5700K, 6200K

CONTROL

Continuous and Strobe via External TTL Pulse	
Dimming	DMX Ready (requires DMX Interface)

POWER

Power Input to Luminaire	48VDC
External Power Supply	100-240 VAC Input - 220W
Power Consumption	120W Continuous - 200W Strobed
Power and Synchronization	Power: 4-pin DIN Connector Sync: BNC Connector

OPTICAL

Optical System	Lens Array
Beam Angles	15° or 28° @ Full Width Half Max

MECHANICAL

Dimensions	92 x 93 x 105 mm (W x H x L)
Weight	1.61 lb or 0.73kg
Mounting	¼-20" on bottom and sides

veritaslight.com sales@veritaslight.com

Specifications are subject to change without notification.
Data accurate as of 8 June 2015

Figure 2.2: Illumination equipment used in this work.

APO-XENOPLAN 2.0/24 CMPCT RUG

***24mm, F2.0, 1.3" sensor, lockable, rugged
 Part #: 27-1992024***

Large View

Home | **Product Details**

ATTRIBUTE NAME	VALUE
Manufacturer	Schneider Optics
Component	Lens
Lens Type	Ruggedized
Lens Format Size	1.3"
Lens Mount	C Mount
Lens Focal Length	24.5 mm
Lens F Stop	2.0

Figure 2.3: Lens used in this work.

Chapter 3

Tracker evaluation

This chapter represents the core of the thesis in that it sets up and presents results for the experiments that will help us to understand the behaviour and performance of the trackers described in Section 2.2 of the previous chapter. The experiments consist of the tracking of a certain area within a video and can be divided into three types: (i) tracking of an object/region in a synthetically created video, (ii) tracking of a real object/region in recorded videos under well-defined movement conditions and (iii) tracking of a real object/area under undefined movement conditions. They are all aimed at evaluating the performance of the different trackers considered.

3.1 Tracking of a synthetic moving object

The performance of a video tracker can be determined by comparing the trajectory of the tracked object with the ground truth, if the latter is known. This suggests the idea of creating videos including a synthetic object that moves following a well characterised equation of motion. In order to realise this, the method explained below has been implemented.

3.1.1 Methodology

A synthetic, circular-shaped, Gaussian moving cloud has been produced and added to a sequence of frames consisting of a black background. The cloud is represented by different levels of grey, with the brightest (white) at the center of the cloud. An example can be seen in Figure 3.1.

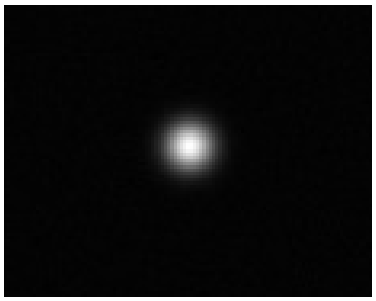


Figure 3.1: Example of a synthetically created video frame containing a Gaussian cloud over a black blackground

The reason for choosing a Gaussian cloud (instead of, for instance, a well defined, sharp object such as a circle) is to try to get closer to real situations, where the edges of objects of interest may not be so well defined. For simplicity, the movement of the cloud has been narrowed down to the vertical direction and modelled as the sum of three zero-phase sinusoids. This allows for an easier characterisation of the motion in terms of properties that are meaningful to vibration analysis, like the amplitude and frequency of oscillations. The frequencies of the sinusoids have been set to $f_1 = 90$, $f_2 = 430$ and $f_3 = 780$, all in Hz, and have been selected with no specific criteria other than to cover a wide range of the spectrum and to avoid them being multiples of each other. The sampling frequency of the videos was set to $f_s = 2000$ frames per second, which according to the sampling theorem [62] allows for all three frequencies to be detected. The equation of motion of the cloud is described by the following expression:

$$y(t) = y(0) + A_{f_1} \sin(2\pi f_1 t) + A_{f_2} \sin(2\pi f_2 t) + A_{f_3} \sin(2\pi f_3 t), \tag{3.1}$$

where $y(t)$ is the vertical position at time t. Similar equation can be set up for the horizontal direction, hence defining, if needed, the 2D movement of the cloud.

The maximum amplitude of the sinusoids have been chosen so as to resemble, to certain degree, real situations. For instance, an *impact hammer model* [21] is widely used in vibration analysis and consists on hitting a structure of interest with a hammer so as to excite its natural vibration modes, as explained in Section 1.1.2. In this model, the energy of the frequency spectrum typically drops off at 6dB/octave [63, p.26] or as $1/f^2$. For practical visualisation reasons, the ratio between the amplitudes has been finally set to $A_{f_2} = 0.5 A_{f_1}$ and $A_{f_3} = 0.25 A_{f_1}$, which still represents a decreasing envelope but not to the point of making peaks at higher frequencies too small compared to those

at lower frequencies. Once the frequency ratio has been fixed, A_{f_1} is adjusted in each test video so as to cover both large and small displacements.

In order to synthesise the tiny, sub-pixel cloud displacements associated to the rather high frame rate of $f_s = 2000$, the initial strategy was to scale up the frames according to the resolution needed (e.g. 100 times if the displacement is of the order of 0.01 pixels), realise the cloud displacement described by Eq. (3.1) at that scale and then downscale back using the OpenCV function `resize()` and the cubic interpolation option. In practice, this approach turned out to be computationally too expensive so eventually it was applied only to the cloud and its close surroundings, then the result appropriately pasted into the unscaled black background. A scale factor of 300 was used for the cloud previous to the position update.

Python code for the creation on the Gaussian cloud is shown in Listing 3.1. The function `gaussianKernel()` provides a mask that, once position-updated and downscaled, it is placed on the frame centered at the location given by Eq. (3.1).

```
import numpy as np
def gaussianKernel(size_kernel, sigma_gaussian):
    # minimum full kernel's tap length is 3
    if (size_kernel < 3):
        return -1

    if (size_kernel % 2 == 0):
        kernel_half_size = size_kernel / 2
        kernel_full_size = 2 * kernel_half_size + 1
    else:
        kernel_half_size = (size_kernel - 1) / 2
        kernel_full_size = 2 * kernel_half_size + 1

    kern = cv2.getGaussianKernel(kernel_full_size, sigma_gaussian, cv2.
    CV_32F)
    kern2D = (kern * np.transpose(kern))
    kern2D = kern2D / np.sqrt(np.sum(np.sum(kern2D ** kern2D)))

    return kern2D
```

Listing 3.1: Function for the creation of a 2D Gaussian kernel.

Based on the explained above, several videos have been created where the maximum amplitude of the lowest frequency, A_{f_1}, has been set to values that range from 5 pixels to 1/100 of a pixel. A bounding box of size 20x20 pixels is initially located at the center of

the first frame, surrounding the Gaussian cloud. The frames are of size 160x120 pixels (width x height) so the exact initial location of the bounding box top left corner is (70, 50).

The tracking algorithms discussed in Section 2.2 have been attempted on these videos and the results compared with the ground truth. For the particular case of vibrational motion, the comparison focuses on (a) identifying the modelled frequencies by means of spectral analysis and (b) checking that the maximum amplitude displayed by the trackers' trajectory is compatible with the maximum amplitude attainable by the combination of the three sinusoids. For instance, if $A_{f_1} = 5$ pixels, then $A_{f_2} = 2.5$ and $A_{f_3} = 1.25$ and the maximum amplitude attainable would be 8.75 pixels. That means that the vertical displacement of the tracker should not oscillate more than 17.5 pixels.

A small amount of dither [64] has been added to all the videos to help improve detection of frequencies with very small amplitudes. It consists of Gausssian noise with $\mu = 0$ and $\sigma = 1$. The function to add noise to a frame is shown in Listing 3.2. It will be shown later in Section 3.1.2.2 that some trackers can even benefit from considerably larger dither levels.

3.1.2 Results and discussion

Figures 3.2 to 3.4 present tracking results for different sets of maximum amplitudes in the modelled sinusoids. Each row corresponds to a tracker, whose vertical, oscillatory displacements are shown on the left column (only a few frames shown for clarity) and their respective frequency spectra for a total of 2000 frames on the right column. Magnitudes in the spectra are expressed in decibel according to the following expression [65, p. 15]:

$$\alpha(dB) = 20 \log_{10} \left(\frac{\alpha}{\alpha_{ref}} \right), \tag{3.2}$$

where α is the absolute value of the Fourier transform of the displacement and α_{ref} is a reference value that has been assumed to be unity.

Case 1: $A_{f_1} = 5$ pixels

Starting with Figure 3.2, the maximum amplitude of f_1 is $A_{f_1} = 5$ pixels and therefore $A_{f_2} = 2.5$ and $A_{f_3} = 1.25$. The vertical displacement profiles are very similar to each other in 8 of the 9 trackers, with MeanShift being the exception. Looking in more detail

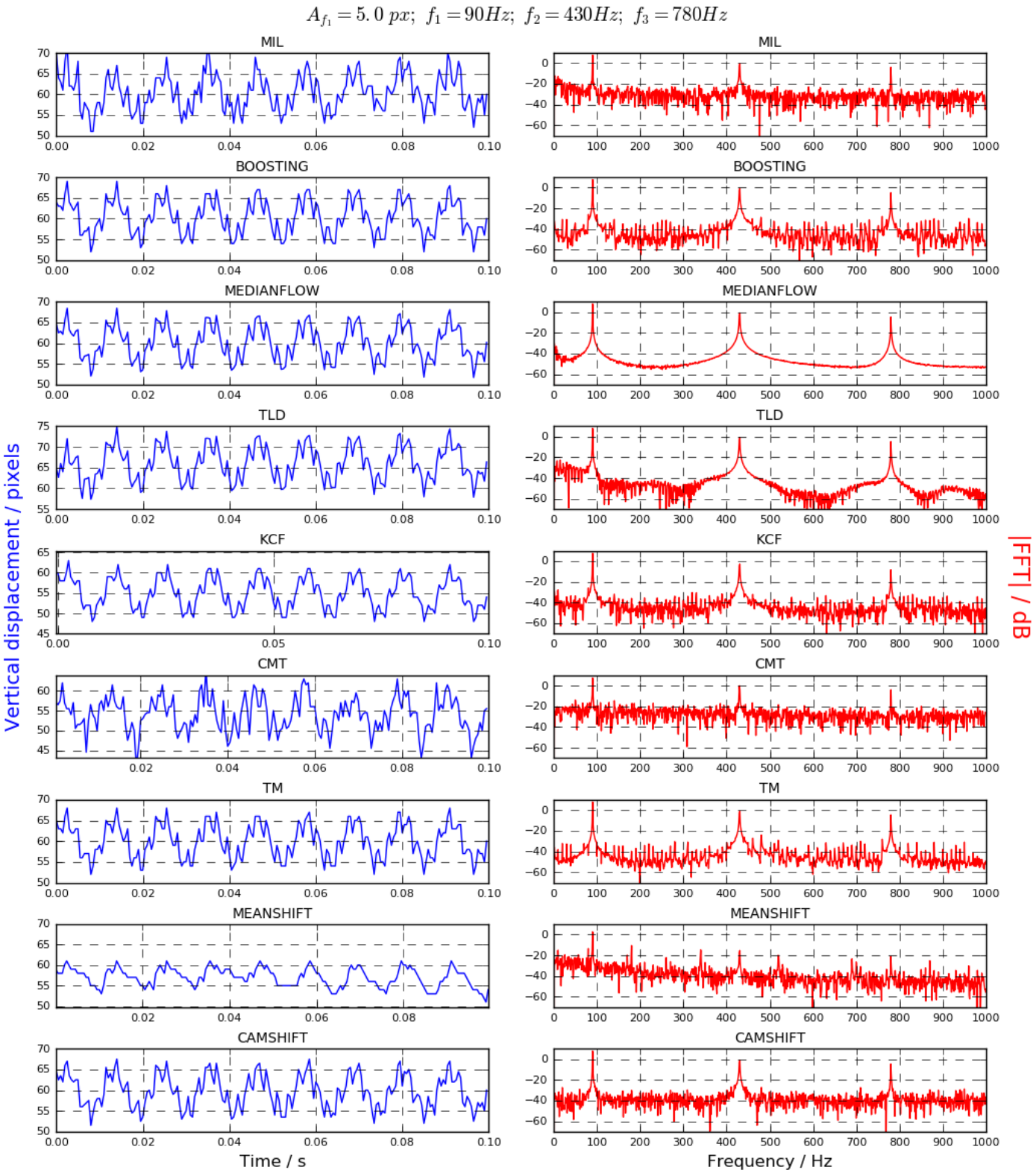


Figure 3.2: Tracker vertical displacement (left) and frequency spectrum (right) for a synthetic moving Gaussian cloud with $A_{f_1} = 5$ pixels.

at the profile for this specific tracker, it shows discrepancies both in amplitude, which is greatly reduced, and resolution. The tracker seems to have problems to appropriately resolve rapid movements of the target, this fact translating into either (a) a flat response in parts where the rest of the trackers show clear maxima and minima (see for instance at time 0.05 s) or (b) simply an absence of them. The sudden local flatness in the signal produces non linearities (distortions) that ultimately show up in the spectrum on the right as harmonics [66]. They appear for example as peaks at $2f_1 = 180$ Hz or $2f_2 = 860$ Hz. The distortion also seems to introduce other modes that are not directly related to the fundamental frequencies, like the ones at around 340Hz or 510Hz. In addition, the inability of the tracker to follow rapid movements results in reduced peaks for f_2 and f_3 as compared with the other trackers.

The spectra for the rest of the trackers accurately show the three fundamental frequencies being modelled, so for displacements of this order of magnitude they are all valid trackers in terms of frequency detection. It is worth noting that the Median Flow spectrum is specially clean compared to the others.

In terms of maximum observed amplitude, none of the algorithms produced peak-to-valley displacements larger than 20 pixels, and most of them were in fact between 10 and 15 pixels. This is compatible with the theoretical maximum displacement of 17.5 pixels for this case.

Another important consideration is the execution times of the different algorithms. They are given in Table 3.1 as seconds per 1000 frames. MeanShift is the fastest although as seen above certainly not the most accurate tracker. On the other hand, TLD and MIL execution times are several orders of magnitude larger than the rest of the trackers so they would not be the first choice for this particular tracking problem.

Table 3.1: Trackers execution times (in seconds) per 1000 frames.

MIL	BOOSTING	Median Flow	TLD	KCF	CMT	TM	MeanShift	CamShift
212.0	20.0	2.6	339.9	1.8	7.2	1.2	0.06	0.11

Case 2: $A_{f_1} = 0.5$ pixels

Moving on to a different set of amplitudes, results for $A_{f_1} = 0.5$ pixels are displayed in Figure 3.3. Half a pixel lies at or below the spatial resolution of many of the trackers considered and therefore quantisation effects start to show up in the vertical displacements. This is specially true for displacements associated with the other two fundamental frequencies, whose maximum amplitudes are $A_{f_2} = 0.25$ and $A_{f_3} = 0.125$ pixels. In order

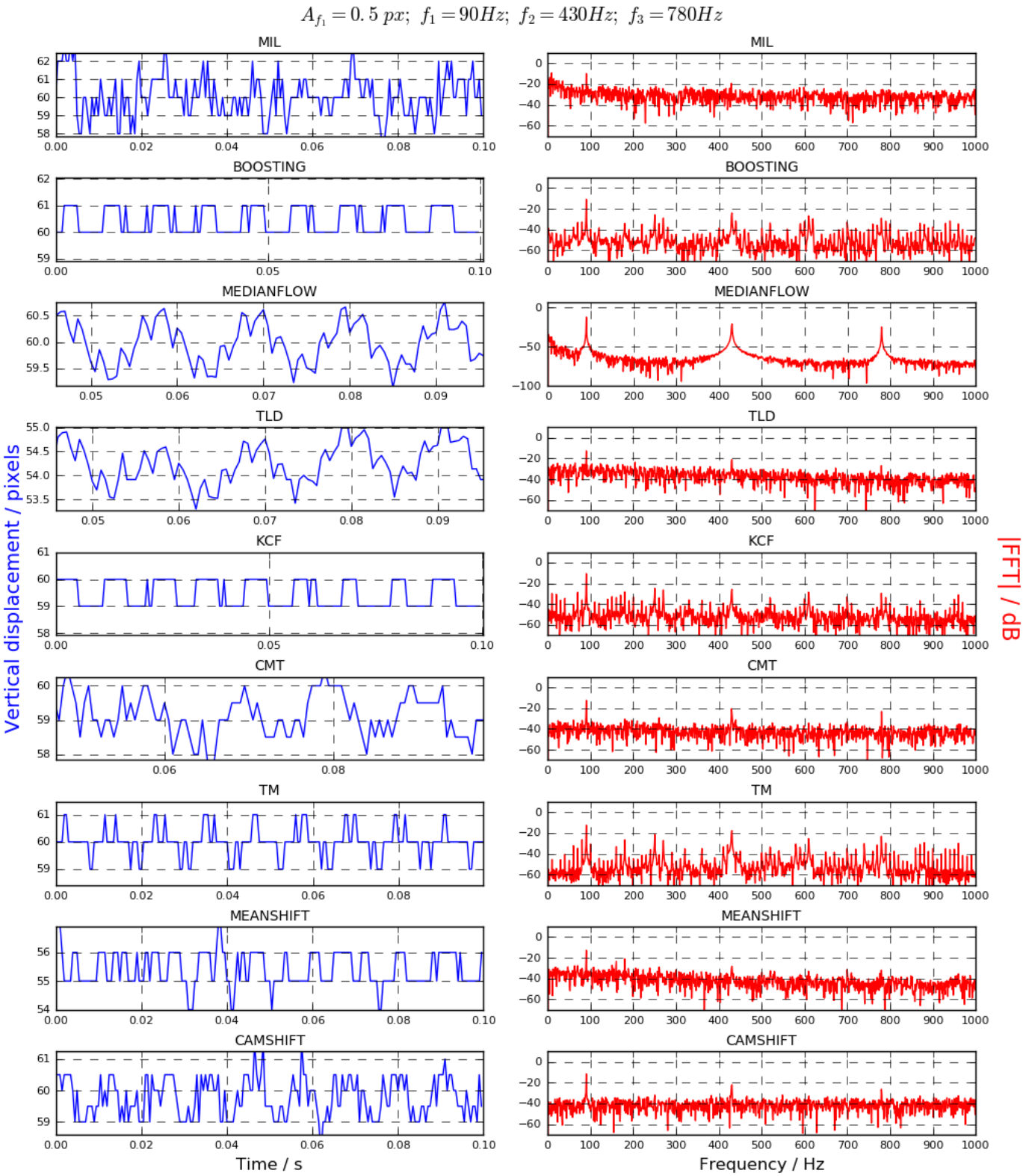


Figure 3.3: Tracker vertical displacement (left) and frequency spectrum (right) for a synthetic moving Gaussian cloud with $A_{f_1} = 0.5$ pixels.

to see the spatial resolution of the trackers in the figure, the displacement profiles have been appropriately scaled as needed. The discussion of Figure 3.3 can be separated into three tracker groups according to: (i) those with resolution of one pixel, (ii) those with resolution of 0.5 pixels and (iii) those with resolution less than 0.5 pixels.

The first group includes MIL, BOOSTING, KCF, Template Matching and MeanShift. The most significant characteristic observed in MIL is the rather noisy displacement profile. Its spectrum can resolve f_1 and hardly f_2, but not f_3. The noise is probably responsible for the differences between this tracker and some of the other trackers belonging to this group. For instance, BOOSTING, KCF and Template Matching profiles are characterised by a highly periodic and stable behaviour. This would lead to well-defined frequency peaks but the fact that the signal is heavily clipped causes spectra that are very rich in harmonics. One can indeed see the peaks of the three fundamental frequencies in their spectra but there are so many other peaks that without previous knowledge it makes the task of picking the right one more difficult. In the case of MeanShift, the tracker seems to be more noisy and the resulting signal, even though it is also heavily clipped, appears to be not so periodic. Similarly to what happened with MIL, this greatly reduces the harmonic content but also makes very hard to discern f_2 and f_3.

In conclusion, for vibrations whose magnitude is of the order of the resolution limit, noisy trackers help reducing quantisation effects such as harmonics but also reduce the sensibility to resolve vibration with smaller amplitudes. On the other hand, less noisy trackers can resolve even smaller vibrations but suffer from corruption due to harmonics that find their origin in the signal clipping.

The second group of trackers includes CMT and CamShift and are the ones with a spatial resolution of 0.5 pixels. This is well below the maximum displacement of the moving cloud for the set of amplitudes considered, that is, $2(f_1 + 0.5f_2 + 0.25f_3) = 1.75$ pixels. Therefore, quantisation effects are not expected to be so significant as in the previous group.

With a resolution of 0.5 pixels, the trackers do not get stuck in fixed positions so easily and as a result, as can be seen in the figure, the displacement profiles are smoother and adapt better to the actual movement of the cloud. The maximum displacements in both trackers are around 2 pixels, a value that is compatible with the 1.75 mentioned earlier. Looking at the spectra, the three fundamental frequencies can be easily resolved so these two trackers would in principle be valid for this tracking problem.

The last group of trackers have a spatial resolution better than 0.5 pixels and consists

of the Median Flow and TLD trackers. They are based on optical flow calculations, which ultimately use pixel intensity levels and can potentially achieve better resolution (see Appendix A for a discussion). As seen in Figure 3.3, both trackers have very similar displacement profiles which are also compatible with the amplitude of the fundamental frequencies. However, TLD suffers from instabilities during the tracking (not shown) that consist of the bounding box occasionally jumping up or down several pixels at a time from one frame to the next one. Once the jump occurs, the bounding box correctly continues the tracking for a number of frames but then it jumps down again and so on. This behaviour must explain the differences in the spectrum as compared with Median Flow, since otherwise the profiles look very similar. The differences are basically a noisier spectrum with less prominent peaks at the modelled frequencies, although the three of them can actually be resolved. In the case of Median Flow, the spectrum shows very clear peaks at the fundamental frequencies and no visible harmonic corruption, making it the best performing tracker investigated so far in view of the results.

Case 3: $A_{f_1} = 0.01$ pixels

The last set of investigated amplitudes corresponds to a maximum amplitude for f_1 of $A_{f_1} = 0.01$ pixels and is mostly aimed at determining the displacement detection limit of the trackers. As one can expect from the results of the the previous case, if there is a tracker able to detect this rather tiny vibrations it will be the Median Flow and possibly the TLD.

The results for this case are displayed in Figure 3.4. The first thing to notice is that no apparent oscillatory behaviour can be observed in the displacement plots. Instead, they are either completely flat (BOOSTINF, KCF and Template Matching), revealing absolutely no movement of the tracker, or look rather random. In the frequency domain one can see this randomness in the form of noise and lack of peaks. The exception is the Median Flow tracker, whose displacement profile is quite stable (note the scale of the displacement) and for which the spectrum is, contrary to the rest of trackers, capable of showing the three investigated fundamental frequencies. Note that the earlier-discussed unstable behaviour of TLD (with its "jumps") can be now clearly seen in the figure.

In conclusion, for this very small vibration amplitudes, only the Median Flow is capable of detecting the fundamental frequencies. The TLD should also, since it is based on the Median Flow itself, but the instabilities introduce artefacts that end up masking the peaks.

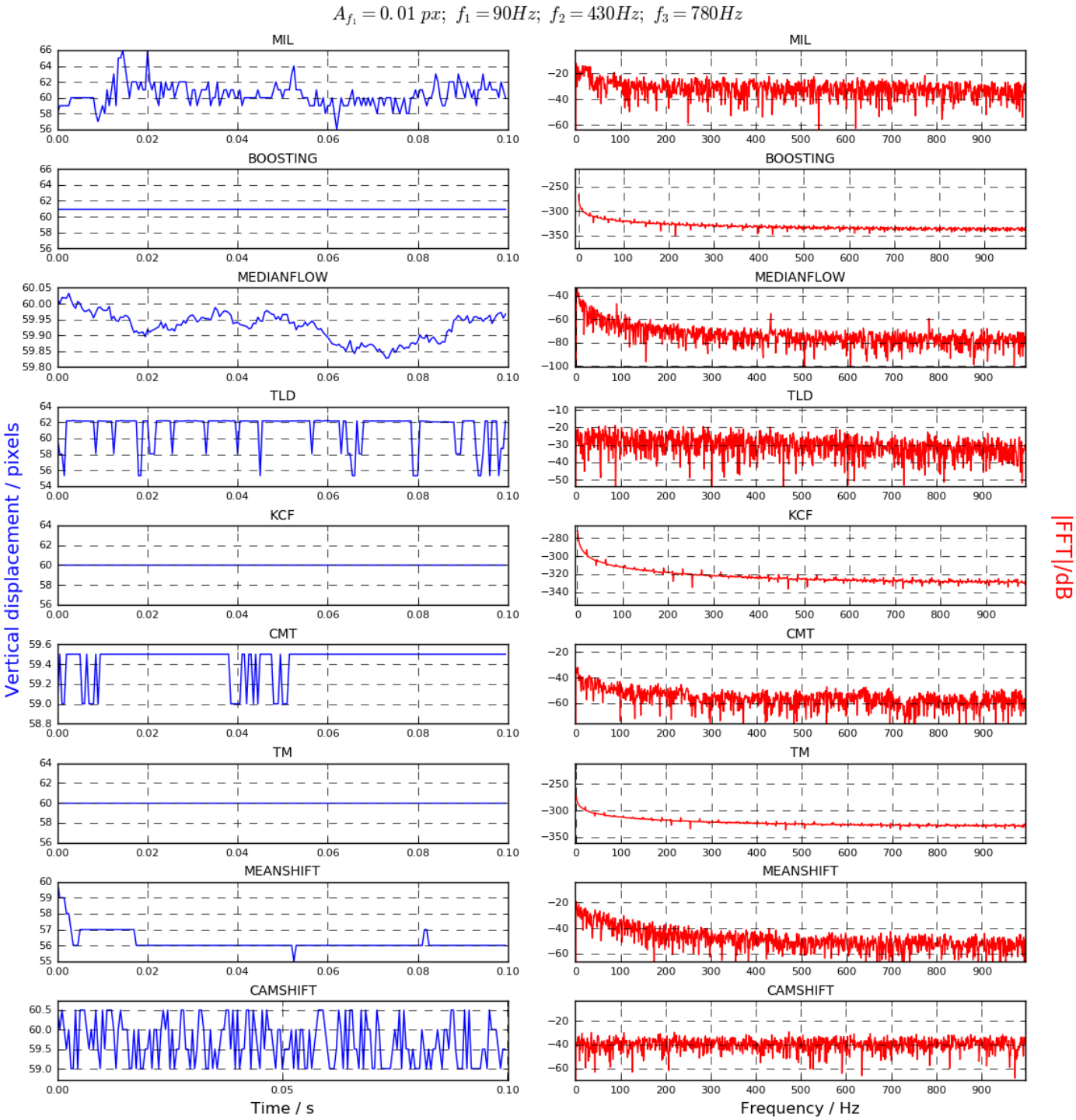


Figure 3.4: Tracker vertical displacement (left) and frequency spectrum (right) for a synthetic moving Gaussian cloud with $A_{f_1} = 0.01$ pixels.

3.1.2.1 Vertical displacement analysis

So far the focus has been on the detection of the three modelled frequencies and the qualitative analysis of the displacement profiles. In order to get a quantitative measure of the latter, two statistics have been considered to compare the trackers with the ground truth. The first of them is the standard deviation σ [67, Sec. 8.3] of the vertical displacement, given by

$$\sigma = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i - \mu)^2}, \tag{3.3}$$

where N is the number of samples (number of frames), y_i is the vertical displacement at frame i and μ the mean of all measured displacements.

The second measure is the Root Mean Square Error (RMSE) between the trackers' displacement and the ground truth. This is a frequently used statistic to measure the differences predicted by a model and the observed data [68]. It is computed using the following expression:

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_{m,i} - y_{t,i})^2}, \tag{3.4}$$

where $y_{m,i}$ and $y_{t,i}$ are, respectively, the vertical displacements of the model and the tracker at frame i. Notice how the RMSE expression is very similar to the standard deviation one. The difference is that it compares two different series of data whereas the standard deviation compares a single series with its own mean. The standard deviation gives an idea on the dispersion of the data around the mean so comparing this statistic in two series sheds some light on how similar the distribution of data are to each other. On the other hand, the RMSE is more of an error measure between the model and the data, i.e., how close are one to each other on average. This introduces a problem whenever the tracker's location exhibits a bias with respect to the center of the moving object. Sometimes this can happen and often it does not mean that the tracking is not valid, but only that the bounding box is slightly displaced. In these cases the RMSE values will be larger than expected so in order to deal with this problem, the RMSE has been calculated only after the vertical location of the trackers has been corrected by a constant value that makes its mean the same as the ground truth.

Results for the standard deviation and corrected RMSE are presented in Table 3.2. Clearly, the Median Flow tracker has the most similar standard deviation and the smaller error when compare with the ground truth, a fact that is true for the two larger

Table 3.2: Standard deviation and corrected Root Mean Square Error in pixels for the trackers and cases discussed in this chapter. The RMSE is computed by comparing each tracker with the ground truth.

Tracker	$\sigma_{A_{f_1}=5}$	$\sigma_{A_{f_1}=0.5}$	$\sigma_{A_{f_1}=0.01}$	RMSE		
				A_{f_1}=5	A_{f_1}=0.5	A_{f_1}=0.01
Ground truth	4.05	0.40	0.007	0	0	0
MIL	4.41	1.89	1.97	1.79	2.0	–
BOOSTING	4.06	0.50	0.0	0.32	0.27	–
Median Flow	**4.05**	**0.41**	**0.074**	**0.10**	**0.06**	**0.07**
TLD	4.06	2.11	2.87	0.55	0.27	–
KCF	3.73	0.50	0.0	0.69	0.30	–
CMT	4.50	0.57	0.135	2.30	0.42	–
TM	4.06	0.53	0.0	0.29	0.31	–
MeanShift	2.25	0.63	0.52	3.06	0.55	–
CamShift	4.03	0.60	0.57	0.55	0.44	–

sets of amplitudes investigated. In the case of the smaller set of amplitudes ($A_{f_1=0.01}$) it does not even make sense to calculate the error for the rest of the trackers, since they can not detect the movement of the cloud. Their standard deviation are given in the table only to give an idea of the noise that they present.

3.1.2.2 Robustness against noise

The behaviour of the trackers under noisy conditions has been investigated by adding different quantities of Gaussian noise to the videos. An example of resulting frames can be seen in Figure 3.5 for the four different levels of noise investigated, which have been added using the function shown in Listing 3.2. By modifying the variance of the Gaussian (the `var` variable), one can add the desired amount of noise. The shown frames in Figure 3.5 correspond to `var` = 10, 40, 160 and 640 in the order of increasing amount of noise, respectively.

From the three different sets of amplitudes discussed, only the one with $A_{f_1} = 0.5$ pixels has been chosen for the noise analysis. As it was shown earlier, this is a borderline case that intuitively can be expected to provide more information on the effects of corrupting the frames with noise.

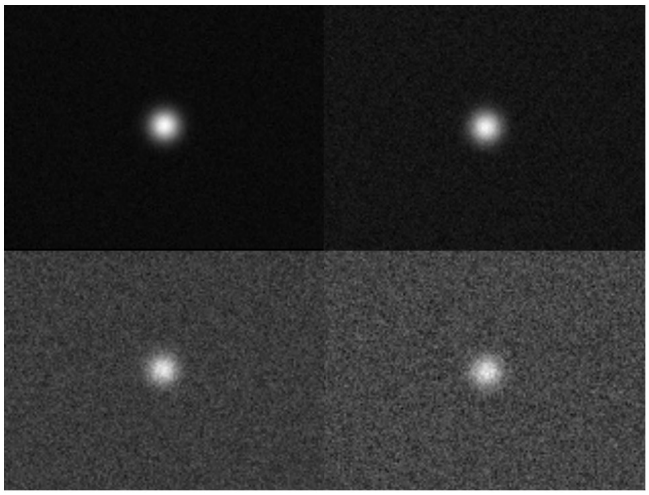


Figure 3.5: Example of frames with four different degrees of added Gaussian noise.

```
def noisy(image):
    row,col,ch= image.shape
    mean = 0
    var = 1
    sigma = var**0.5
    gauss = np.random.normal(mean,sigma,(row,col,ch))
    gauss = gauss.reshape(row,col,ch)
    noisy = image + gauss
    return noisy
```

Listing 3.2: Function used for the addition of Gaussian noise to a frame.

Results for the statistics discussed in the previous section are presented in Table 3.3 for the different levels of noise. As a reference, the values given in Table 3.2 for $A_{f_1}=0.5$ pixels are also provided, where `var`=1 accounts for the added dither, as explained in Section 3.1.1. In addition, in Figure 3.6, the frequency spectrum for each of the trackers at each level of noise is depicted. These can be compared to Figure 3.3 (`var`=1).

First conclusion drawn from the data in Table 3.3 is that some trackers are more robust to noise than others. CamShift quickly becomes unstable (`var`>10), with the bounding box expanding to the size of the entire frame and therefore invalidating the tracking. Even for `var`=10, the spectrum in Figure 3.6 shows a very week peak at f_1, with f_2 or f_3 not being resolved at all.

Median Flow, although so far the best tracker investigated, also becomes unstable for the largest level of noise tested. At lower levels, dispersion and error both increase

Table 3.3: Standard deviation and corrected RMSE for $A_{f_1} = 0.5$ pixels and under different noise conditions. Data for column `var`=1 are taken from Table 3.2 and given as a reference.

Tracker	σ | RMSE				
	`var`=1	`var`=10	`var`=40	`var`=160	`var`=640
Ground Truth	0.40 | 0	0.40 | 0	0.40 | 0	0.40 | 0	0.40 | 0
MIL	1.89 | 2.0	1.61 | 1.57	2.07 | 2.03	2.55 | 2.56	1.55 | 1.50
BOOSTING	0.50 | 0.27	0.50 | 0.29	0.50 | 0.28	0.54 | 0.47	0.56 | 0.38
Median Flow	0.41 | 0.06	0.63 | 0.48	3.05 | 3.02	3.11 | 3.10	–
TLD	2.11 | 0.27	0.53 | 0.41	1.00 | 0.91	0.39 | 0.40	0.93 | 0.80
KCF	0.50 | 0.30	0.50 | 0.29	0.50 | 0.29	0.50 | 0.29	0.52 | 0.32
CMT	0.57 | 0.42	0.46 | 0.22	0.46 | 0.23	0.67 | 0.54	2.55 | 2.52
TM	0.53 | 0.31	0.53 | 0.31	0.53 | 0.31	0.52 | 0.31	0.55 | 0.38
MeanShift	0.63 | 0.55	0.56 | 0.52	0.68 | 0.72	0.87 | 0.88	1.18 | 1.23
CamShift	0.60 | 0.44	0.74 | 0.76	–	–	–

with the level of noise. Looking at the spectra, this is revealed as a progressive decrease in the height of the peaks at the fundamental frequencies. The Median Flow tracker is characterised, as previously shown, by a very high spatial resolution. The reason for this is that optical flow methods use even the tiniest image intensity differences to compute the flow. From this perspective, it is not difficult to see how variations in intensity introduced by noise can rapidly degrade the performance of the tracker.

The rest of the trackers can handle all noise levels and the general behaviour, specially towards high noise content, is also an increase in both the standard deviation and the RMSE. In other words, the noise tends to make the trackers' movement more dispersed and less similar to the ground truth.

Some of the algorithms are however significantly more robust than others. For instance, BOOSTING, KCF and Template Matching are rather unaffected by the noise (perhaps only in the case of `var`=640 the values depart a bit more from the ground truth). This result is expectable, as the three trackers are based on correlation filters, which have the property of maximising the signal to noise ratio (SNR) in the presence of additive stochastic white noise [69]. TLD's implementation is unstable as discussed earlier and repeating the tracking gives very different results every time, so it is hard to draw conclusions.

MIL, CMT and MeanShift measurements worsen with the noise more regularly, although for `var`=640, MIL experiences an unexpected improvement. An interesting

feature of these trackers is that they actually improve when going from the reference `var`=1 to `var`=10 and CMT even maintains the same values for `var`=40. This supports the idea that a fairly large amount of noise, in other words, heavy dithering, can actually improve the performance of certain trackers. Investigating this effect in detail is outside the scope of this thesis but it would be an interesting topic for future research. For now, looking at the spectra in Figure 3.6 one can see how the detection of the fundamental frequencies really improves in some of the trackers as the noise level is increased. This is specially true in those trackers that suffered most from quantisation effects due to poor spatial resolution. For instance, BOOSTING, KCF or even Template Matching presented a high amount of harmonics due to these quantisation effects, which in turn made the task of resolving the fundamental frequencies more difficult. What Figure 3.6 clearly shows is that adding large amounts of noise can actually help to reduce the appearance of the harmonics, resulting in much cleaner spectra.

3.2 Tracking of real objects: Shaker tests

The next step forward in the characterisation of the investigated trackers would be the analysis of real videos. We know from Section 1.1.3.1 that the use of shakers is standard in modal analysis. As a reminder, a shaker or vibration tester [70] is a device used to supply energy to a system under well-known frequency conditions. What it is proposed in this section is the use of high-speed videos recorded during shaker tests in order to perform tracking of the containing vibrating parts. The final aim is to compare the result from the tracker with the ground truth provided by the shaker, so that the performance of the former can be estimated.

The approach discussed in this section represents a halfway stage between the analysis of a synthetic object discussed in previous sections and the vibration of objects in real situations, which will be considered in Section 3.3 and in Chapter 4.

3.2.1 Methodology

The tests considered consist of a shaker that vibrates in the vertical direction at the desired frequency. The metallic cylinder seen in Figure 3.7 acts as a platform to which the objects being tested are attached. In these particular tests a rectangular object has been attached with the sole purpose of providing some sharp edges that can improve the tracking. Consequently, the initial bounding box for all the trackers has been placed

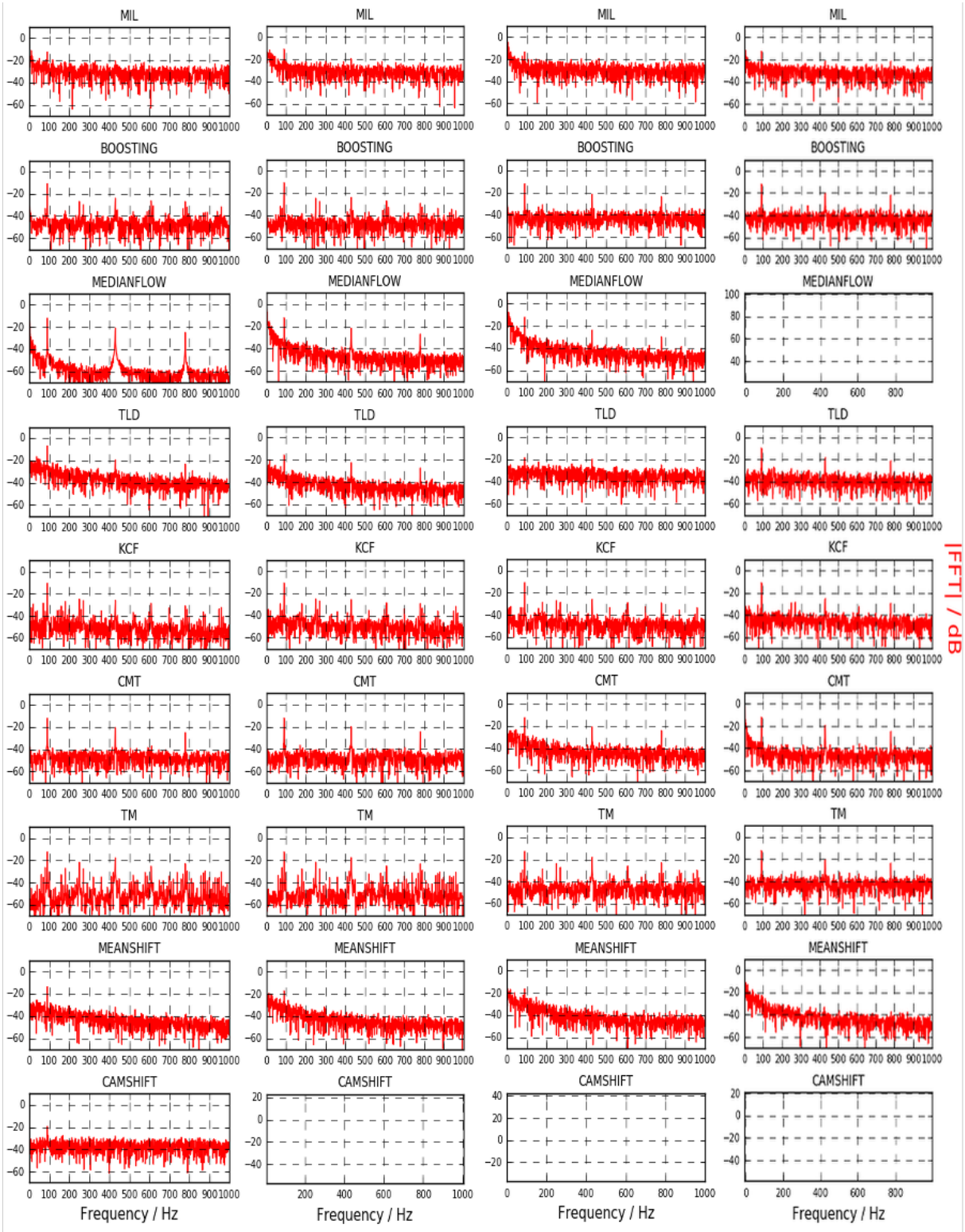


Figure 3.6: Frequency spectra for all trackers and noise levels considered. Columns form left to right correspond to `var`= 10, 40, 160 and 640, in that order.

at the location shown by a green square in the first frame of the figure (the location in the rest of frames corresponds to Median Flow tracking results). The platform itself is firmly attached to a moving drum that transmits the required oscillatory motion.

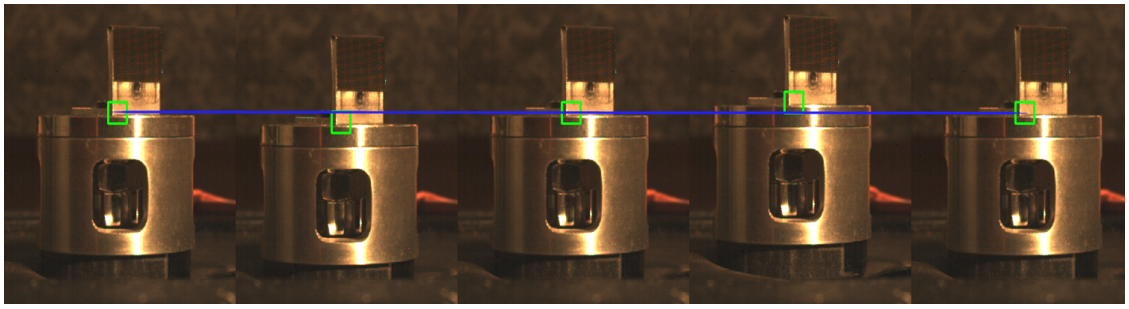


Figure 3.7: Characteristic frames from a video recorded during a shaker test, spanning an entire oscillation cycle. Green box represents a typical tracked region. The blue straight line is given as a reference to help visualising the oscillatory movement.

All the shaker test videos used in this work have been recorded at a high-speed frame rate of 2000 fps using the equipment described in Section 2.3.1. The different images in Figure 3.7 are showing a few representative frames of a complete oscillation cycle for one of the tests. The straight line is drawn as a reference to help visualising the oscillatory movement. For this particular case, the maximum amplitude from the equilibrium position (represented by the first, third and fifth frames) is around 3 mm.

3.2.1.1 Ground truth estimation

Typical known parameters in a shaker experiment are the frequency of vibration f (in Hertz) and a_{max}, the maximum acceleration applied (in G's). From these, it is also possible to calculate the maximum displacement of the shaker, i.e., the amplitude of the oscillatory movement. This is made as follows: Assuming the shaker follows a zero-phase simple harmonic motion (SHM) [71], its displacement from the equilibrium position is given by

$$y(t) = A\sin(\omega t), \tag{3.5}$$

where A is the maximum amplitude, ω is the frequency in radians and t is the time. By taking first and second derivatives with respect to time in Eq. (3.5), the velocity and acceleration are obtained, respectively:

$$v(t) = \omega A\cos(\omega t), \tag{3.6}$$

$$a(t) = -\omega^2 A \sin(\omega t), \tag{3.7}$$

The maximum acceleration in a shaker is applied when the amplitude is also maximum, which for Eq. (3.5) occurs at exactly $n\pi/2$, n being a integer number. Since $\sin(n\pi/2) = 1\ \forall n$, we have that the magnitude of the maximum amplitude is given by

$$A_{max} = |-a_{max}/\omega^2| = |a_{max}|/(2\pi f)^2 \tag{3.8}$$

As an example, if a shaker is working at a vibration frequency of $f = 30$ Hz with a maximum acceleration of 0.1 G, that is, 10 % of gravity (9.8 m/s^2), its maximum amplitude will be $A_{max} = 0.98/(2\pi 30)^2 = 27.6\,\mu m$.

From a total of 19 shaker tests performed and filmed at the laboratory, only three of them will be discuss here. Their set frequencies and accelerations are such that the resulting maximum amplitude spans four orders of magnitude, from just a few micrometers to a few millimetres. The experimental conditions for these tests are specified in table 3.4, along with the maximum amplitude A_{max} obtained from Eq. (3.8). Frequencies and amplitudes in the table provide the ground truth against which the performance of the trackers will be evaluated. This is discussed in the next section.

Table 3.4: Specification of parameters and calculated amplitudes for the shaker tests investigated in this work.

Test no.	f(Hz)	Acc. (G's)	No. frames	fps	$A_{max}(\mu$m)
03	30.0	10.0	8000	2000	2758
10	120.0	10.0	8000	2000	172
05	55.0	0.1	8000	2000	8

3.2.2 Results and discussion

For each of the tests in table 3.4, all the usual tracking algorithms have been tried out. It was found out, after some experimentation where different bounding box locations and sizes were attempted, that MeanShift and CamShift were not stable for these particular tests. By "not stable" it is meant that the bounding box was either moving to unacceptable locations from frame to frame or not following the movement appropriately (MeanShift) or increasing the size of the bounding box to unacceptable values (CamShift). It was previously seen in Section 3.1.2 how MeanShift was arguably the

worst performing tracker and in Figure 3.6 how CamShift was very sensitive to noise, presenting the mentioned bounding box size problem. These issues seem to reappear here and therefore the trackers have not been considered any further.

Comparing the size of the shaker within the videos with its real physical dimensions is required in order to obtain a conversion ratio that relates pixels to units of distance. Its value has been calculated by using the real width of the outer diameter of the cylinder (54 mm) and counting the number of pixels it spans (207). This leads to a conversion factor of 3.83 pixels/mm.

Results for the selected trackers are shown in Figure 3.8 for Test 03. This is the

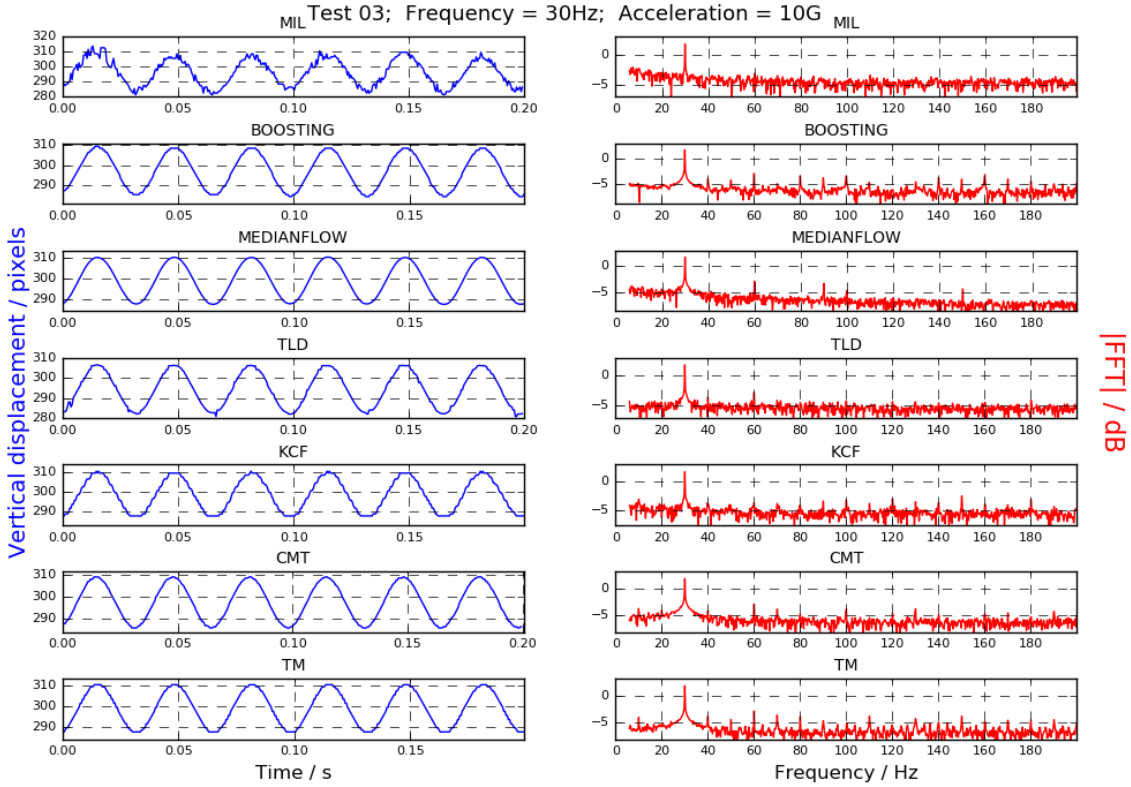


Figure 3.8: Bounding box centroid vertical position (left) and frequency spectrum (right) for shaker Tests 03.

test for which the shaker experiences the largest displacements. The vertical position of the bounding box's centroid and its FFT frequency spectrum are displayed in the figure. The time and frequency axis have been cut down, respectively, for clarity. The amplitude has been estimated as half the observed peak-to-valley change. From the figure, one can roughly observe in most algorithms a vertical change that ranges between 20 and 25 pixels. Assuming a mean value of 22.5 pixels and applying the conversion

ratio to half that value results in a visual estimate for the maximum amplitude of approximately 2.9 mm. Considering this is only a rough estimation, it agrees rather well with the value of A_{max}= 2.8 mm in table 3.4. The displacement spectra show a very distinctive peak at 30 Hz for all trackers, so clearly the vibration frequency can be determined in this manner. Some recurring characteristics that have already been observed in Section 3.1.2 when studying the synthetic object are for instance the rather noisy displacement profile of MIL or the large amount of harmonics introduced by BOOSTING, KCF and Template Matching.

Figure 3.9 shows results for Test 10, where all trackers were able to detect the shaker's vertical displacement and their respective frequency spectra show a peak at the right frequency of 120 Hz. Due to the limited spatial resolution of some of the

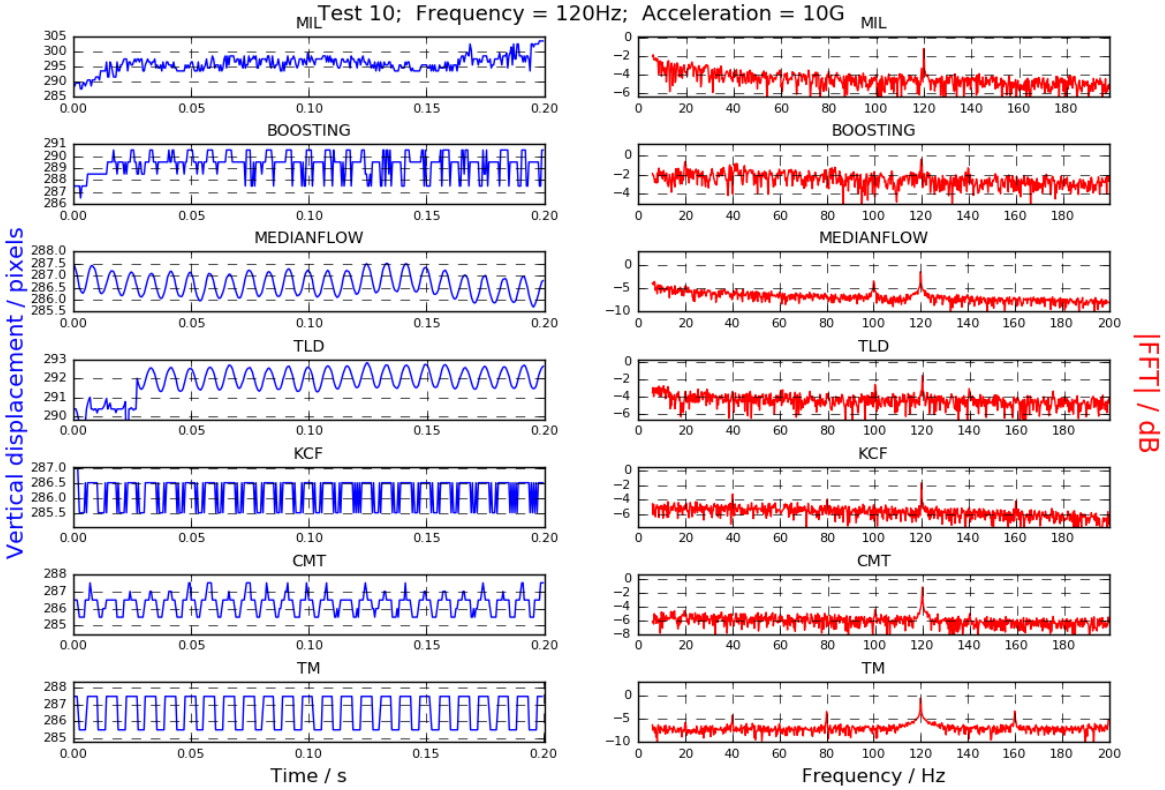


Figure 3.9: Bounding box centroid vertical position (left) and frequency spectrum (right) for shaker Tests 10.

trackers and the smaller amplitude of vibration in this test, it is likely that the error in estimating visually the maximum displacement will be larger than in the previous case. For this reason, the opposite conversion has been performed, i.e., the maximum

amplitude in Table 3.4 has been converted to pixels, which results in a peak-to-valley displacement of 1.32 pixels. We can see in the figure that this value is highly compatible with the displacement profiles. Specially easy to see this is in the profile of the Median Flow tracker, whose resolution is higher.

Finally, results for Test 05 in Figure 3.10 corresponds with the tiniest shaker displacement considered. In Table 3.4 this is estimated to be only a few micrometers, hence several orders of magnitude smaller than the previous tests. Performing the inverse con-

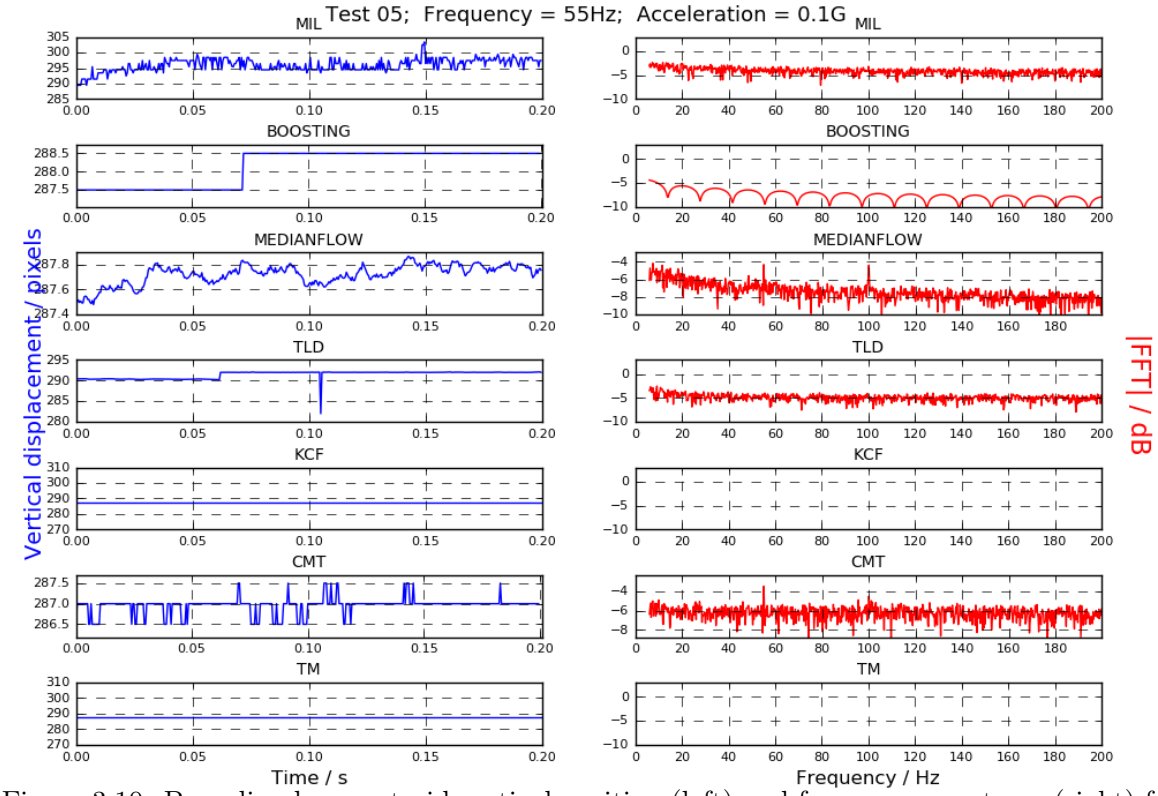


Figure 3.10: Bounding box centroid vertical position (left) and frequency spectrum (right) for shaker Tests 05.

version from distance units to pixels gives a peak-to-valley displacement of 0.06 pixels. This is well within the detection threshold for the Median Flow (see Sec. 3.1.2), a fact that also seems to be true for the CMT tracker, as these two were the only capable of detecting the frequency of the shaker at 55 Hz. There is also a very distinctive peak in the spectra at 100Hz (only Median Flow) that most likely corresponds with the flicker frequency of the light source [72].

So far, different trackers have been put to the test in two different experiments, one consisting of a synthetic moving object and the other consisting of a real object, a

shaker, operating under strictly controlled motion conditions. In the following section, a final comparison on the performance of the different trackers has been carried out on a truly real industrial situation such as the testing of an aircraft's landing gear, which hopefully will give us even more evidence on the suitability and performance of the Median Flow tracker.

3.3 Tracking of an aircraft landing gear

Figure 3.11 shows a few frames of a video recorded at 500 fps on laboratory landing gear (LG) tests. The vertical position of the supporting structure (thick metal panel on top of the LG) has been measured in the tests by using accelerometers. This moving vertical position is taken as the ground truth for any tracking algorithm that attempts to follow the LG as it falls and bounces on the ground. In the figure, two bounding boxes are being displayed. The larger, rectangular one on the top, is used for tracking the mentioned vertical movement. The other bounding box (square-like) is used to track the horizontal displacement of the wheel axis during landing. The initial location and the size of the boxes have been chosen experimentally by trial and error so that vertical and horizontal movements are reasonably followed by the trackers. As it will be seen in Sections 3.4 and 3.5, a more robust and systematic method can be developed in order to select the most appropriate bounding boxes in the context of vibration analysis.

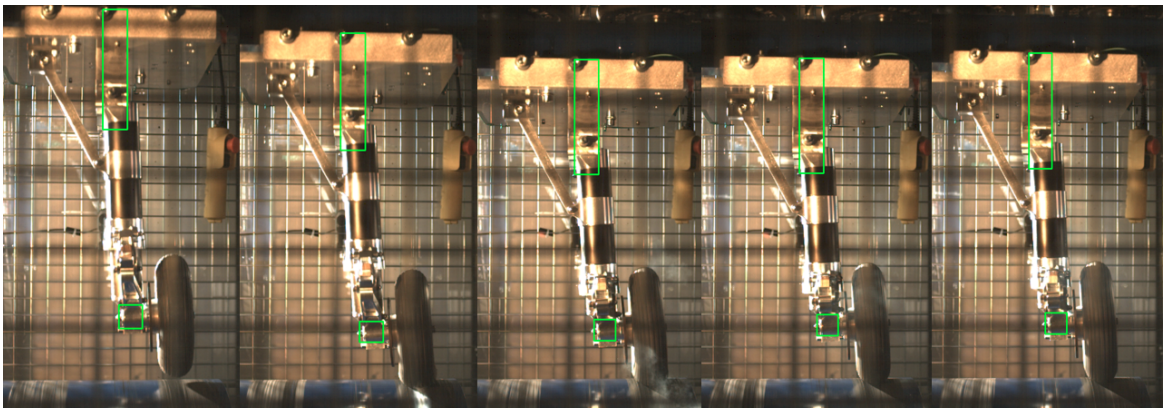


Figure 3.11: Example frames of a video recorded on landing gear laboratory tests. The green bounding boxes are used to track the vertical (top box) and horizontal (bottom box) position of different parts of the landing gear. Used tracker is Median Flow.

The ground truth for the LG is shown in Figure 3.12 as a function of time. The

vertical displacement units have been intentionally omitted for confidentiality reasons. The region of major interest lies inside the framed part of the graph (zoomed in for

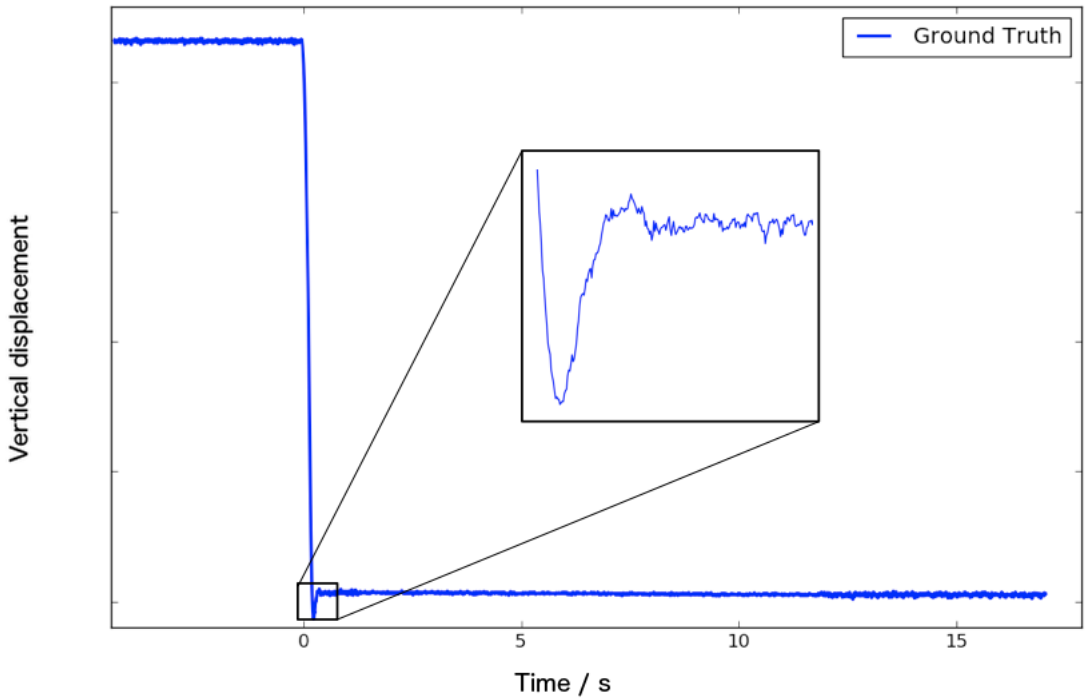


Figure 3.12: Vertical displacement of landing gear as a function of time as measured by accelerometers located on the top part of the landing arm (ground truth). Displacement units have been omitted for confidentiality reasons.

clarity), since it is in this region where the LG hits the ground and bounces back. The LG follows a free fall-like behaviour until the moment the wheel lands on the moving ground (blue horizontal cylinder at the bottom of the frames). At that time, and under the action of the LG suspension, the vertical movement progressively decelerates until complete stop, then it reverses upwards and after a short time again downwards and so on until the landing gear completely stabilises. Notice how the signal provided by the accelerometer is rather noisy.

The aim of this tracking experiment is to replicate the ground truth data as closely as possible by analysing the LG video. Focusing on the large bounding box, the midpoint between the y coordinate of its top and bottom left corner is taken as a measure of the vertical position of the landing gear. The initial bounding box is the same for all the algorithms tested. Within the frames, the vertical coordinate increases as the gear descends. In the ground truth, however, the vertical direction is measured as height in

meters, and therefore decreases as the landing gear falls. In addition, it is necessary to convert pixel units to distance units and also synchronise the times of both ground truth and tracking data. These differences require the following steps in order to make them comparable:

1. Reverse the sign of the vertical displacement tracking data in order to make it decrease as the landing gear descends.

2. Scale the vertical displacement tracking data to the scale of the ground truth. This is accomplished by taking a vertical reference from the ground truth, which is the difference between the minimum height and the (averaged) height once the landing gear has stabilised. By measuring that same difference in the tracking data, one obtains a conversion factor that allows to take the tracking data to the ground truth's scale.

3. Translate the time dimension in the tracking data so that the time of the minimum height is the same as in the ground truth.

The function that implements such transformations, `match_curves()`, is shown in Listing 3.3 at the end of this chapter. It also implements the error minimisation explained in the next paragraph.

Taking the minimum in the vertical displacement as a reference to synchronise the times of both sets of data is a simple method, but if the data is noisy it may lead to time synchronisation errors. A more robust approach would be to minimise the square difference between both sets of data around the time where the minimum is located. In order to do this, time-shifted versions of the ground truth have been compared with the tracking data. The shift ranges from -20 to +20 time steps with respect to the time location of the minimum displacement, `xmingt` in function `match_curves()`. For each of the time-shifted versions, the quadratic error between the two sets of data within a range around `xmingt` is calculated and the version with minimum error is chosen as the best match between the two sets of data.

Figure 3.13 shows, separated into two graphs for clarity, the tracking results for the algorithms considered after all the explained pre-processing. TLD, MeanShift and CamShift have not been included for the reasons explained in previous sections. Median Flow is as usual the tracker with the best spatial resolution. Notice how the ground truth is considerably noisy as compared with this tracker. The other trackers, as we already know, exhibit poorer resolution that makes their respective curves rather

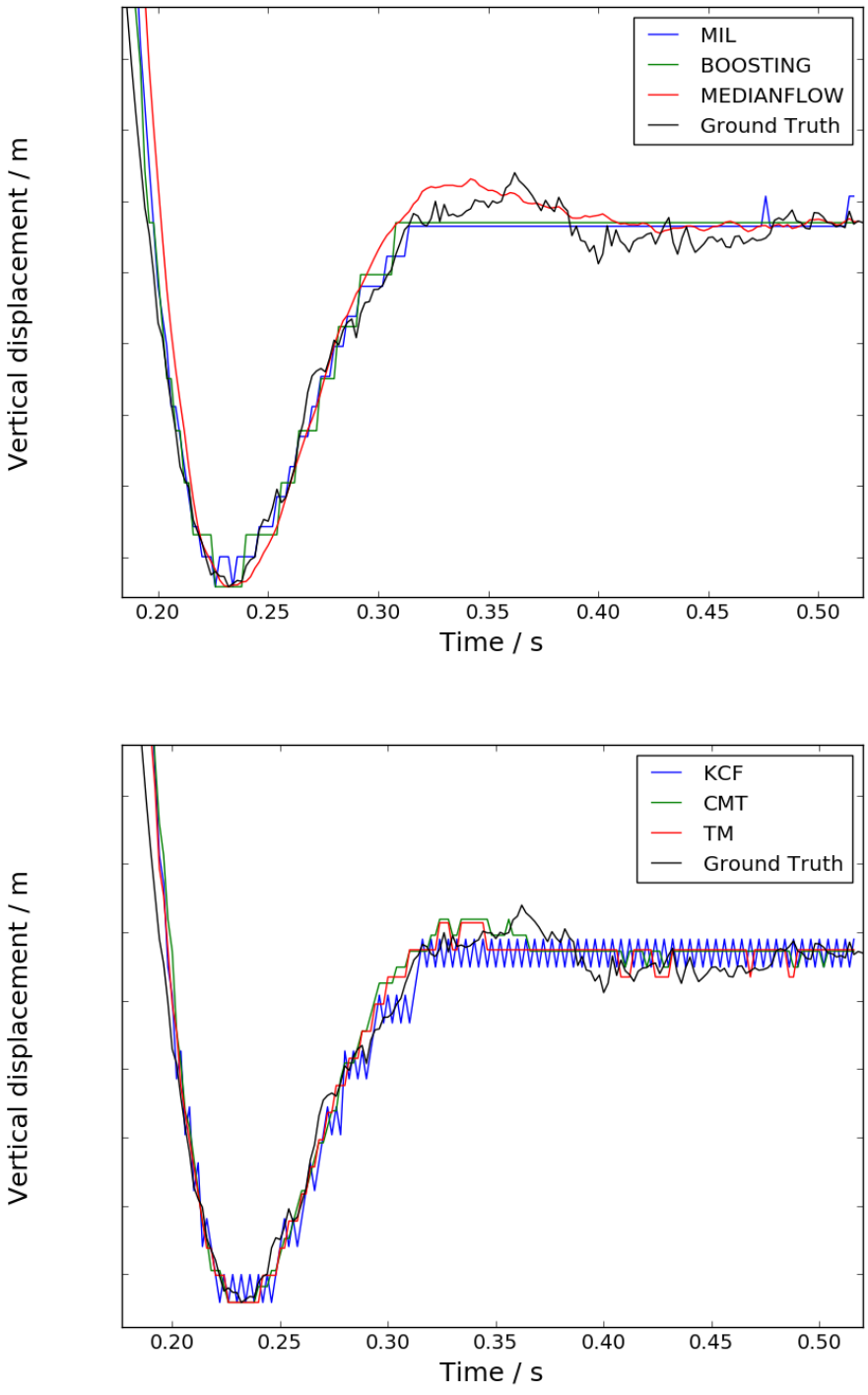


Figure 3.13: Vertical displacement of landing gear as a function of time for the different algorithms considered and adjusted to the ground truth's units and scale. Displacement units have been omitted for confidentiality reasons.

snappy, since the region of interest in the figure only spans a few pixels in the vertical direction (≈ 15 pixels). In the case of MIL and BOOSTING, for instance, the shoulder appearing in the ground truth at time around 0.35 s can not even be resolved and the trackers basically remain static after that time. The same behaviour is observed in KCF, the only difference being that this tracker presents an added constant oscillation of one pixel from frame to frame. This does not however seem to affect the general behaviour of the tracker and it is not known whether this is caused by a bug in the algorithm or simply it is mean to be that way. The remaining two trackers, CMT and Template Matching are not as smooth as Median Flow but manage to partly detect the mentioned shoulder.

The mean square error (MSE) for each curve against the ground truth are given in Table 3.5. This error measure is equivalent to the square of Eq. (3.4) and has been calculated considering all curve values greater than `xmingt`-5. The obtained errors are of the same order of magnitude and quite similar to each other, with the exception of KCF that is almost twice as large as the rest. The fact that the ground truth is noisy makes difficult the task of comparing the trackers' performance. For instance, MIL has a slightly lower error than Median Flow, yet the former cannot resolve some of the features of the ground truth, such as the shoulder.

Overall, from a qualitative point of view and mainly with the help of the vertical displacement profiles, it seems to be reasonable to say that the Median Flow performs the best thanks to its high spatial resolution, which translates into much smoother curves.

Table 3.5: Mean square error around the ground truth's minimum location for the tracking algorithms considered.

Tracker	MSE/10^{-7} (m^2)
MIL	2.7
BOOSTING	3.2
Median Flow	3.0
KCF	4.8
CMT	2.8
Temp. Matching	3.1

As mentioned earlier, a second bounding box has also been considered in order to study the horizontal displacement of the landing gear's wheel axis. Results are shown in Figure 3.14. On this occasion, ground truth data was not available, so the displacements

are given in pixel units. They have been converted from absolute pixel position in the image to relative pixel displacement respect to their initial absolute position.

The displacement profiles for this direction have a more oscillatory character than for the vertical direction. The wheel axis bounces right and left a few times before stabilising. The spectrum has been calculated for the Median Flow tracker and although not shown, it presents significant frequency peaks at 5, 14 and 45 Hz. The rest of the trackers exhibit similar quantisation effects as for the vertical direction.

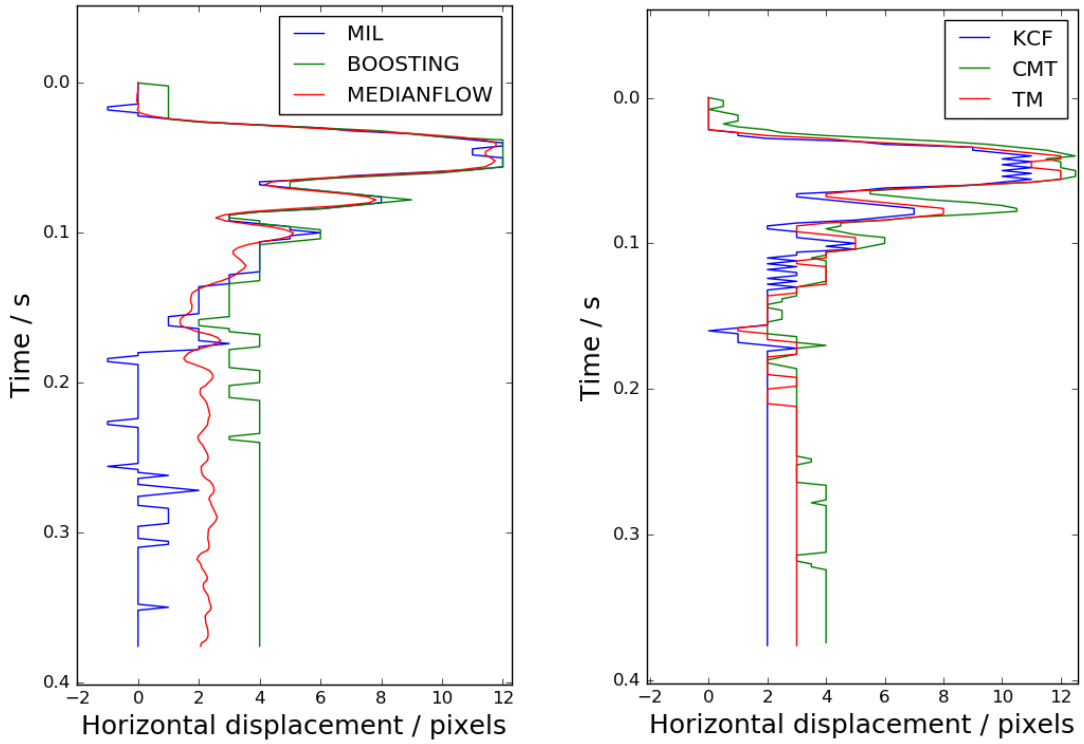


Figure 3.14: Horizontal displacement of landing gear as a function of time for the different algorithms considered.

The results obtained in this section and in Sections 3.1.2 and 3.2.2 provide enough evidence that the Median Flow tracker may be one of best trackers, as a whole, for the purposes of this work. It is true that this tracker is rather sensitive to noise and may not support occlusions or situations where objects are not visible throughout the whole sequence [38], but most of the tests of interest to this work will be performed under controlled conditions where these complications should be minimised. On the positive side, Median Flow is one of the fastest and its spatial resolution largely outperforms the

rest of the trackers considered. This is specially important, as many of the tests will be performed, unlike the landing gear test, on structures where vibration amplitudes are very small, often imperceptible to the naked eye.

3.4 The role of the initial bounding box

The effect of the bounding box initial location has been investigated on the shaker tests. As it will be shown below, this location can have a great impact on the outcome of the tracking and therefore in the frequency content of the analysed area.

Assuming all parts of the shaker's platform vibrate at the same frequency and with equal intensity (as it should be for a testing device), the differences observed in the spectrum for different bounding boxes will come solely from differences in their image attributes. Edges perpendicular to the direction of movement are specially relevant to optical flow-based trackers such as the Median Flow, since they are based on intensity gradients. It can be relevant to this work to determine the correlation (or lack of it) existing between the edge content within an initial bounding box and the performance of the Median Flow tracker for that box. By performance, in the context of vibration analysis, it is meant the ability of the tracker to provide a displacement profile whose spectrum can resolve the frequencies of interest.

In order to measure the correlation, a tracking test has been set up where the strength (the height) of the peak at the shaker's frequency of interest is obtained. A wide region of the shaker is first selected in the initial frame, within which a number of bounding boxes are defined. The shape of the bounding boxes is the same as the selected region in terms of width/heigh ratio, but their actual sizes depend on how many of them are required inside the region (typically 16 or 25, see Figure 3.15 (a)). For each of them, the whole video sequence is run and the tracker's frequency spectrum calculated.

Once the spectrum is available for each box, a peak detection algorithm developed by the author and based on signal-to-noise ratio considerations [73], is run on a small frequency range around the frequency of interest so as to avoid other peaks interfering in the detection. It works as follows: for each current frequency bin in the spectrum, the algorithm looks at a group of bins on its left, separated from the former by a few bins, i.e. not immediately contiguous. The mean and standard deviation of the group are computed as well as the difference between the value of the current bin and the

just calculated group mean. Exactly the same procedure is followed on the right side of the current bin. If both calculated differences are larger than a fixed number of left and right group standard deviations, respectively, then the current bin is considered a peak. The ratio between each difference and the corresponding standard deviation, averaged between both sides, is defined as the peak strength relative to its surroundings. The reason why means and standard deviations are calculated on bins which are not contiguous to the current one is to avoid bins from wider peaks being included in the calculation. Parameters such as the number of bins used as surroundings and its separation from the current bin must be adjusted by hand as well as the minimum number of standard deviations required to detect a peak.

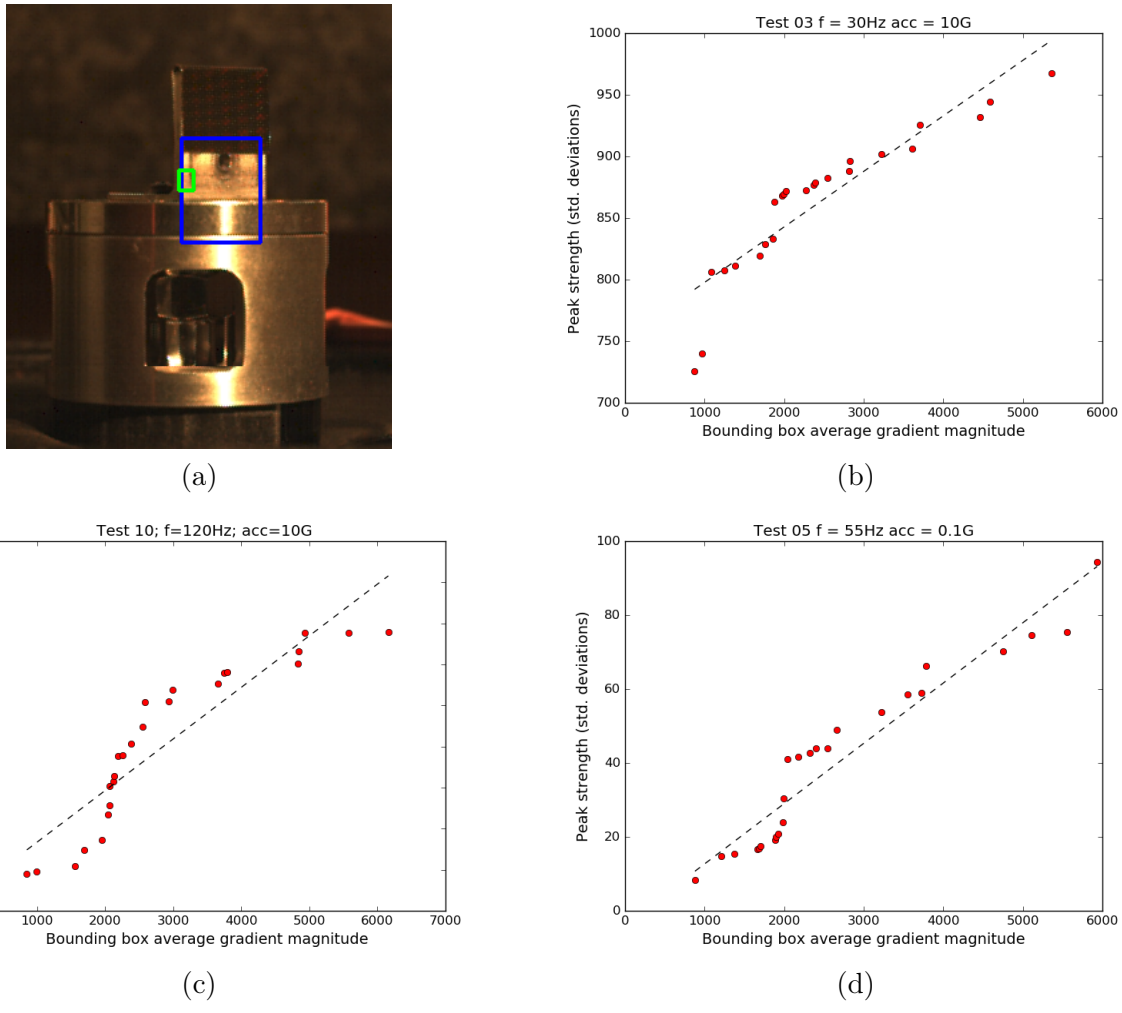


Figure 3.15: (a) Region considered for the selection of initial bounding boxes. (b), (c) and (d) Correlation between the peak strength of the frequency of interest and a measure of the gradient (as average gradient magnitude) in the vertical direction.

Correlation between the average gradient of the different bounding boxes (25 in total) and the strength of the peaks at the frequencies of interest, expressed in standard deviation units as explained above, are depicted in Figure 3.15 for the three shaker tests considered. In (a) the wide area selected appears in blue and one of the initial bounding boxes in green. As can be seen in (b), (c) and (d), the general behaviour is that of the peak strength increasing in a rather linear fashion when plotted against the average gradient, with a correlation coefficient $R > 0.90$ in all cases.

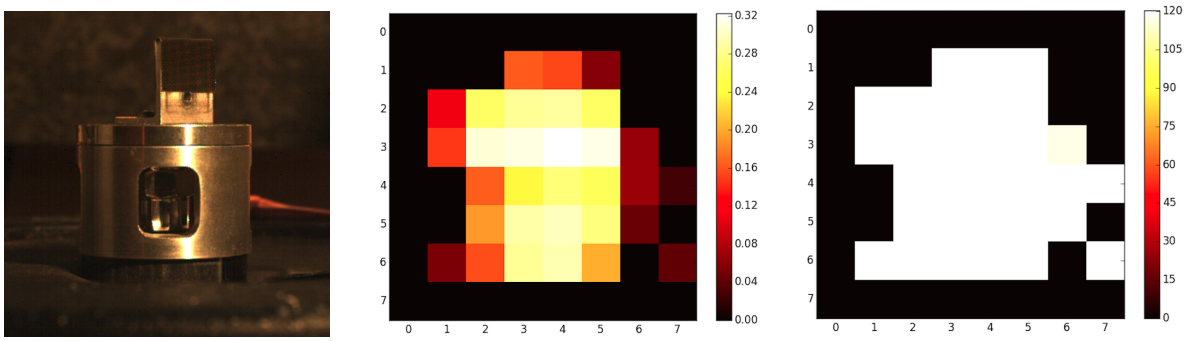
It can be concluded from the Figure 3.15 that the Median Flow tracker implementation of OpenCV works best in areas where there is great concentration of edges in the direction perpendicular to the movement. In situations where the movement is more complex or simply not perpendicular to any of the main directions, it would be necessary to use both components of the gradient to account for any possible direction.

3.5 Frequency color maps

The tests performed in the previous section show that the initial location of the bounding box in a video is important in order to optimise the detection of interesting frequencies. This is because the image properties of certain bounding boxes can help the tracker to perform better.

Extending this idea even further, and without assuming any specific property, one can just divide the initial frame or an interesting subregion of it into equal-size bounding boxes, then track them separately for the whole duration of the video. The frequency spectrum for each tracker can then be computed and the peak strength at the frequency of interest used to populate a matrix that can be visualised as an image. This image provides spatial information about how strongly the frequency of interest can be detected within the video. It is best to illustrate this with an example.

Figure 3.16 (a) shows a square region extracted from the video associated to Test 10 of Table 3.4. The region contains the entire shaker cylindrical platform and some of its surroundings, such as part of the shaker's main body at the bottom, a wire and some background. Image (b) shows a color map where each pixel corresponds to an initial bounding box located at that specific location in image (a). The strength of the largest frequency peak within a region of 117-123 Hz is displayed as a color map. One can see how the regions not belonging to the moving parts, like the background and the main body of the shaker at the bottom, present zero strength, whereas the region occupied



(a) (b) (c)

Figure 3.16: (a) Example of a square region extracted from shaker Test 10. (b) Color map for frequency strengths at 120 Hz. (c) Color map for to frequency at 120 Hz.

by the cylindrical platform shows the maximum peak strength. Regions in between, like the edges of the platform, are associated with intermediate strength values. In image (c) the actual frequency of the peak is displayed, which as expected for this test, is 120 Hz for the regions where the peak has been detected and is taken to zero (by convention) where the peak is not.

Figure 3.16 represents an ideal situation where there is either an area vibrating at a single frequency or an area with no vibration at all. In real life situations one could expect different scenarios. One could have for instance a structure vibrating at different frequencies depending on which part it is being looked at. As long as those frequencies are within the frequency range chosen for the peak detection, image (c) would provide a color map of them. Otherwise, new analysis should be done in order to obtain maps for each frequency of interest. Another common situation would be a vibrating structure where some areas vibrate more energetically than others, unlike in the shaker tests.

Colour maps are a very intuitive way to see the spatial distribution of a given frequency in terms of the strength of its peak in the spectrum. However, one must take into account the fact that this measure depends in principle on the interaction of to properties of very different nature: the actual vibration taking place and the image properties of the region being tracked. It has been shown that for the Median Flow tracker, the peak strength is highly correlated with the gradient. Hence, the color maps must not be taken as a quantitative measure of the vibration but more like a tool to understand which parts of a video are more easily tracked for a given frequency. In some situations, for instance when all parts of the investigated structure have similar texture, it would be fair to associate the bright colors in the map with the areas of

larger vibration energy at the frequency investigated, but in general this does not need to be the case.

Figures 3.16 (b) and (c) have a rather low resolution of 8x8 pixels. Increasing this resolution is accomplished by reducing the size of the initial bounding boxes, which obviously means more individual trackings to be performed. As this can easily lead to very heavy computations, a parallel implementation has been put in place and the video has also been loaded in RAM memory for faster access. The parallel implementation is based on the Python `threading` module and makes use of all the cores available on a given machine.

Figures 3.17 and 3.18 show colour maps of peak strength (first row) and exact frequency (second row) at different levels of resolution for tests 03 and 10. As the

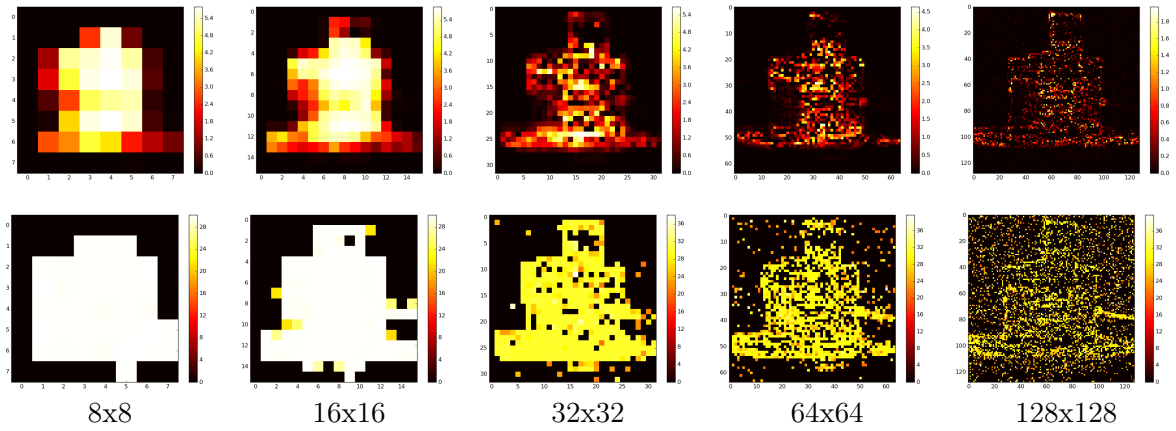


Figure 3.17: Colour maps for peak strength (top row) and exact frequency (bottom row) at different levels of resolution for shaker test 03.

size of the initial bounding boxes is reduced (more resolution), one can see how not all of them are capable of tracking the oscillations, as evidenced by dark pixels in both strength and frequency maps. It seems that 32x32 or even 64x64 would be a good compromise between resolution and robustness. It is at these resolutions that horizontal edges can be seen as brighter, suggesting that they are a good choice if the tracking is to be optimised.

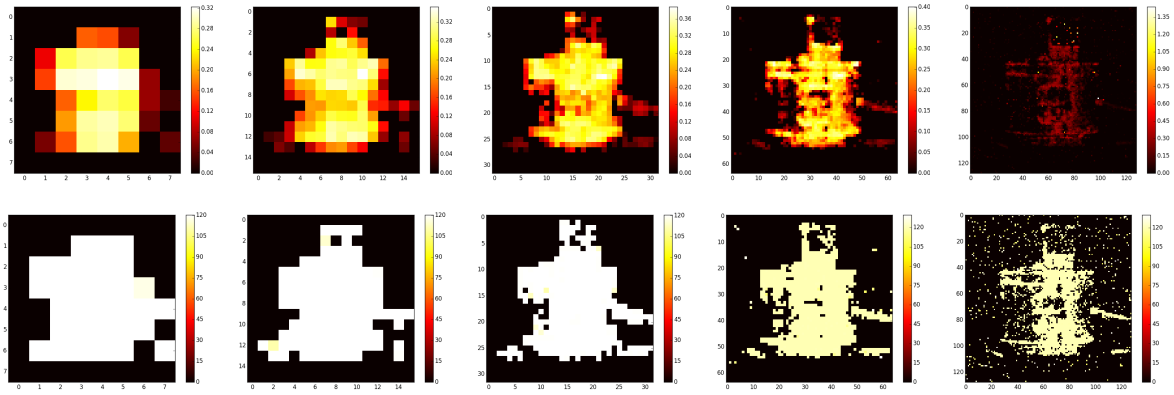


Figure 3.18: Colour maps for peak strength (top row) and exact frequency (bottom row) at different levels of resolution for shaker test 10.

3.6 Conclusions

The results obtained throughout this chapter have shown that, given the set of classic and state-of-the-art tracking algorithms applied to the specific test scenarios of interest, the Median Flow outperforms the rest of trackers in terms of spatial resolution and also proves to be one of the fastest. The remaining of this work will be therefore focusing on this specific tracker.

3.7 Listings

```
def match_curves(dsplv):
    '''Convert the tracking data to units comparable to ground truth
       and match the two curves using least squares on the area near to
    the curves' minima'''

    dsplv = np.asarray(dsplv)
    np.savetxt('dsplv.txt', dsplv, delimiter=",")
    x, y = np.loadtxt('dsplv.txt', delimiter=',', unpack=True)

    # Invert the data to make it compatible with ground truth
    y = -y

    # Determine value and time location of the minimum in the data
    ymin = np.min(y)
    xmin = x[np.argmin(y)]

    # Get number of elements from the minimum to the end of data
```

```
    min2end = len(x) - np.argmin(y)

    # Difference between the minimum and some average of the final position
    diff = np.mean(y[-25:]) - ymin

    # Load ground truth
    xgt, ygt = np.loadtxt('/Videos/ground_truth.txt', delimiter=',', unpack=True)

    # Value and time location of the minimum in the ground truth
    ymingt = np.min(ygt)
    xmingt = xgt[np.argmin(ygt)]

    # Difference between the minimum and the final position (taken in similar area as in the tracker data)
    diffgt = np.mean(ygt[np.argmin(ygt)+min2end-25: np.argmin(ygt)+min2end]) - ymingt

    # Rescale to ground truth data
    ratio = diffgt / diff
    y = y * ratio

    # Recalculate the minimum of rescaled data
    ymin = np.min(y)

    # Bring the y values to the ground truth's minimum
    y = y - (ymin - ymingt)

    # Adjust the x
    x = x + (xmingt - xmin)  # simple approach using the two minima

    # More elaborated approach estimating the bias in the x direction by least square (own code)
    minarr = np.argmin(y)  # array element number of the minimum
    minarrgt = np.argmin(ygt)  # array element number of the minumum
    min_err = 1e6  # set a very high starting error
    bias = 0

    # Estimate the displacement that minimises the error between the GT and tracking curves
    for i in range(-20, 20):  #  -20:20 around the minimum position

        # Calculate the mean square error
        err = np.mean(np.square(ygt[minarrgt + i - 5:minarrgt + i+min2end] - y[minarr - 5:minarr+min2end]))

        # Store the minimum error so far along with the displacement
        if err < min_err:
            min_err = err
            bias = i
    print min_err

    x = x + bias * (x[1] - x[0])

```

```
65     return x, y, xgt, ygt, ratio
```

Listing 3.3: Function for comparing tracking results with the ground truth in aircraft landing gear tests.

Chapter 4

Vibration analysis: An interactive tool with examples

In the previous chapter, a series of experiments were set up in order to investigate a number of trackers and their suitability for vibration analysis applications. The general conclusion was that the Median Flow tracker performed the best among all of them. In this chapter, a basic, proof-of-concept, interactive tool that implements that tracker for the analysis and detection of vibrations in videos is presented. In addition, a series of real live examples are examined and the applicability of the tool in the context of vibration analysis demonstrated.

4.1 The interactive tool

The tool's user interface has been created using the PyQt library. This is a Python binding for the Qt cross-platform GUI/XML/SQL C++ framework [74], which is widely used for developing application software with graphical user interfaces (GUIs).

A screenshot of the tool developed in this work is shown in Figure 4.1. It offers two main functionalities, executed through the `Analyze` and `Color Map` buttons, which are explained below. Before pressing any of these buttons, the user must select and load a video file using the `Pick a file` button. Once loaded, the first frame of that video will show up on the right side of the window, where the user can select a bounding box by pressing, dragging and releasing the left mouse button. The bounding box is automatically resized to a square according to the width selected with the mouse.

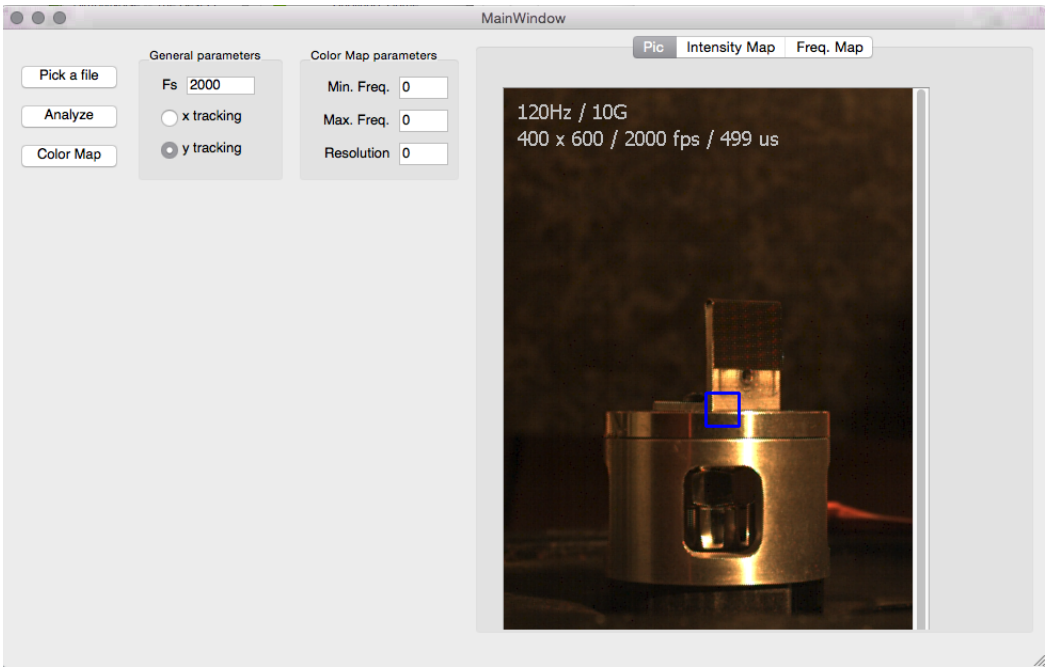


Figure 4.1: User interface of the developed vibration analysis tool.

4.1.1 Analyze

By pressing the `Analyze` button, Median Flow tracking is performed on the mouse-selected bounding box. Internally, the tool crops a region from the video that contains the bounding box plus some margin, typically 20 pixels thick, around it. The tracking is then performed on the cropped version, where the margin is expected to be large enough to accommodate the amplitude of the vibration. The goal of the cropping is to reduce the tracking processing time.

Previous to pressing `Analyze`, the parameter `Fs` must be set to the video frame rate. Also the user must select which direction should be analysed, the horizontal (`x tracking`) or the vertical (`y tracking`). After pressing `Analyze` the results are presented on the left part of the interface, as seen in Figure 4.2. They consist of three plots that have many of the usual functionalities included in the Python `matplotlib` package, such as zoom, move, reset, save to file, etc.

The top plot shows the trajectory, in pixels, followed by the tracker. This helps in deciding whether the tracker behaved as expected, failed, etc. The middle plot is arguably the most important to modal analysis, since it presents the frequency spectrum of the data represented in the top plot and provides valuable information such as the

vibration modes, their frequency, shape, etc. Notice in the figure the clear peak at exactly 120 Hz, as corresponds for the particular loaded test. The bottom plot is known

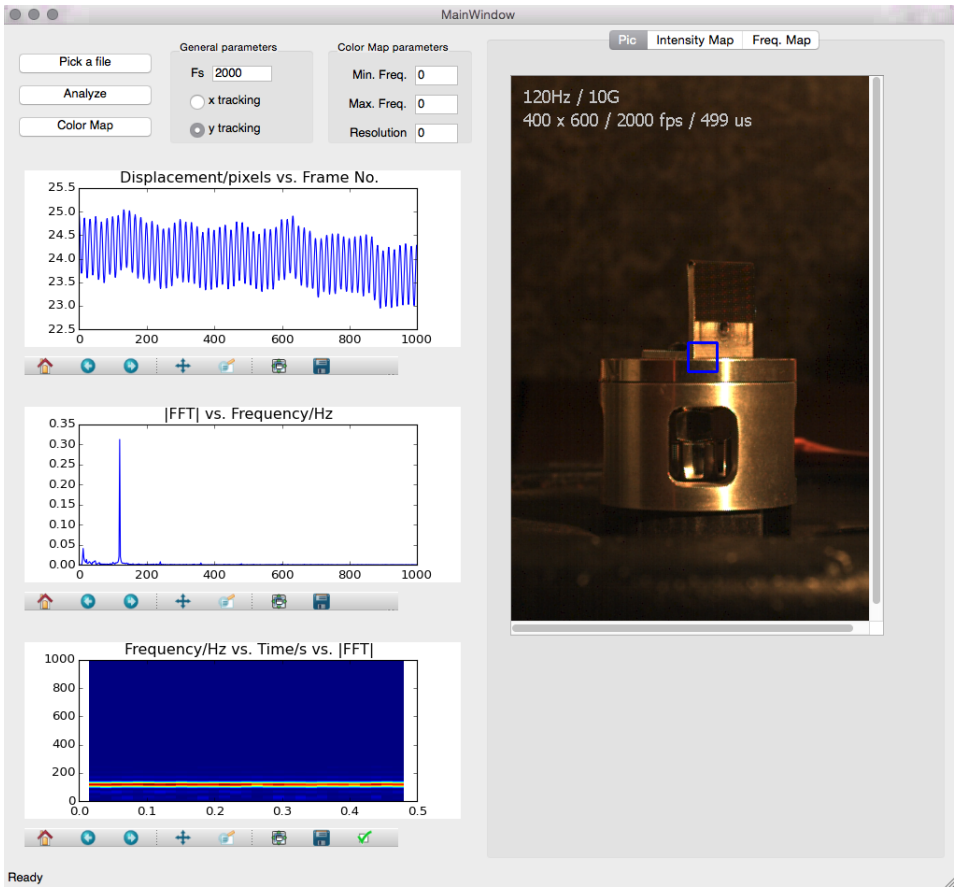


Figure 4.2: Typical result of the `Analyze` functionality.

as a spectrogram. It gives a temporal evolution of the frequency spectrum. To create a spectrogram, the signal on the top plot is chopped into overlapping windows and the frequency profile is calculated for each of them, then plotted vertically as a color map (red is high, blue is low) and one after the other in time sequence. In the spectrogram one can see a red color line at exactly 120 Hz that lasts for the whole duration of the video, indicating the existence of a long-standing mode at this frequency. Spectrograms are a valuable tool to determine for instance the times at which certain resonances occur during a test.

4.1.2 Color Map

The second functionality of the tool is related to the color maps discussed in Section 3.5. The idea is to divide a pre-selected area (using the mouse) into a number of bounding boxes, then internally perform the tracking and spectrum analysis for each one of them. Each bounding box is then assigned the height and frequency of the maximum peak detected within an also pre-selected range of frequencies and all of them are plotted together as two color maps, once for the frequency peak strength, and one for the frequency itself.

Before pressing `Color Map`, there are three parameters that must be set up in the user interface, apart from the other two previously discussed. These are `Min. Freq`, `Max. Freq.` and `Resolution`. The first two set the range of frequencies within which the peak detection is run on the spectrum for each bounding box. The third parameter is the square root of the number of bounding boxes in which the selected area is divided. If a color map of resolution 64x64 is required, then the resolution parameter must be set to 64.

While for the `Analyze` functionality one would typically choose a small bounding box on a specific location, it is more common in the `Color Map` case to choose a wider area. A typical operation would consist on (1) analyse a few bounding boxes on target locations using `Analyze` to get an idea of the frequencies of interest , (2) use that result to set `Min. Freq` and `Max. Freq.` and (3) execute `Color Map` to obtain the color maps for the region of interest.

After pressing `Color Map`, an image covering the selected area (plus some margin) pops up in a different window. This image shows the progress of the tracking for the various bounding boxes. An example can be seen in Figure 4.3, where the user has selected a wide area that includes the shaker's cylindrical platform and its surroundings. The green squares represent the bounding boxes where the tracking has already been completed. Depending on the number of processors on the host, several bounding boxes can be tracked at a time (8 in the figure).

After processing all bounding boxes, the resulting color maps can be visualised by clicking on the two tabs located on the top right part of the user interface, next to the picture tab. For the example considered here, the results are shown in Figures 4.4 and 4.5.

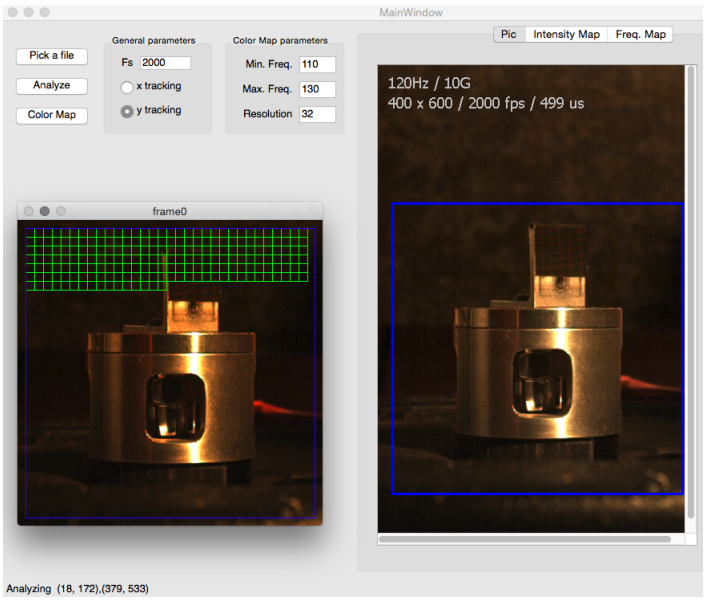


Figure 4.3: Screenshot of the interactive tool showing a set up for the `Color Map` functionality. The popup window on the left shows in green the bounding boxes where tracking has already been performed.

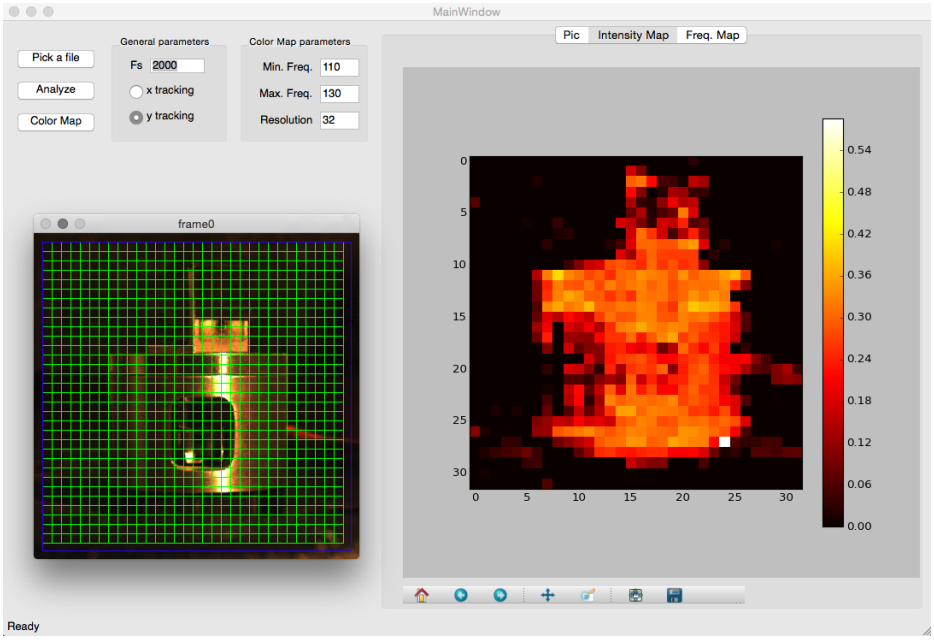


Figure 4.4: Color map for the peak strength at the frequency of interest.

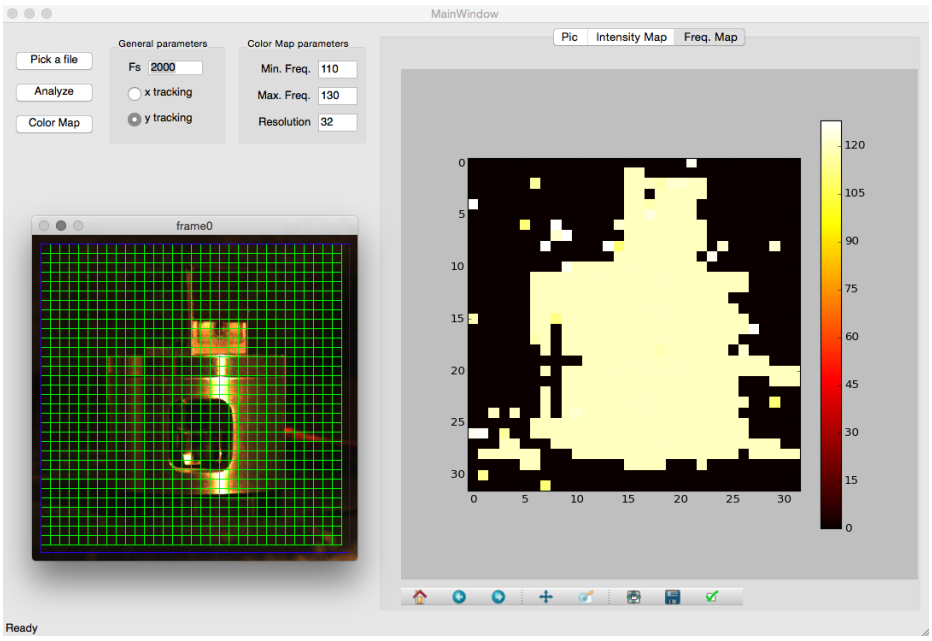


Figure 4.5: Color map example for the frequency at the peak of interest.

4.2 Examples

Apart from the shaker test just seen above, a number of examples are presented in this section where the tool is tested in real life scenarios. All except Example 2 correspond to tests performed by the author or co-workers and by using the equipment specified in Section 2.3.1.2.

4.2.1 Example 1: Car engine at idle

The top of a car engine running at idle has been recorded at a frame rate of 500 fps. While recording, the tachometer displayed 950 RPM (revolutions per minute), which converts to approximately 16 Hz. Using the tool to track the small selected metal area shown in Figure 4.6 provides a main frequency at 13.5 Hz. Taking into account some errors introduced by the tachometer (and visual error of reading it), this value can reasonably be considered as a match. Notice how several harmonics at multiples of this main frequency also appear on the frequency plot. The main frequency is also shown in the spectrogram to be approximately constant throughout the video.

In Figure 4.7 (b) and (c), the frequency maps at the main frequency and corresponding to the area shown in Figure 4.7 (a) are depicted. As one would expect, areas

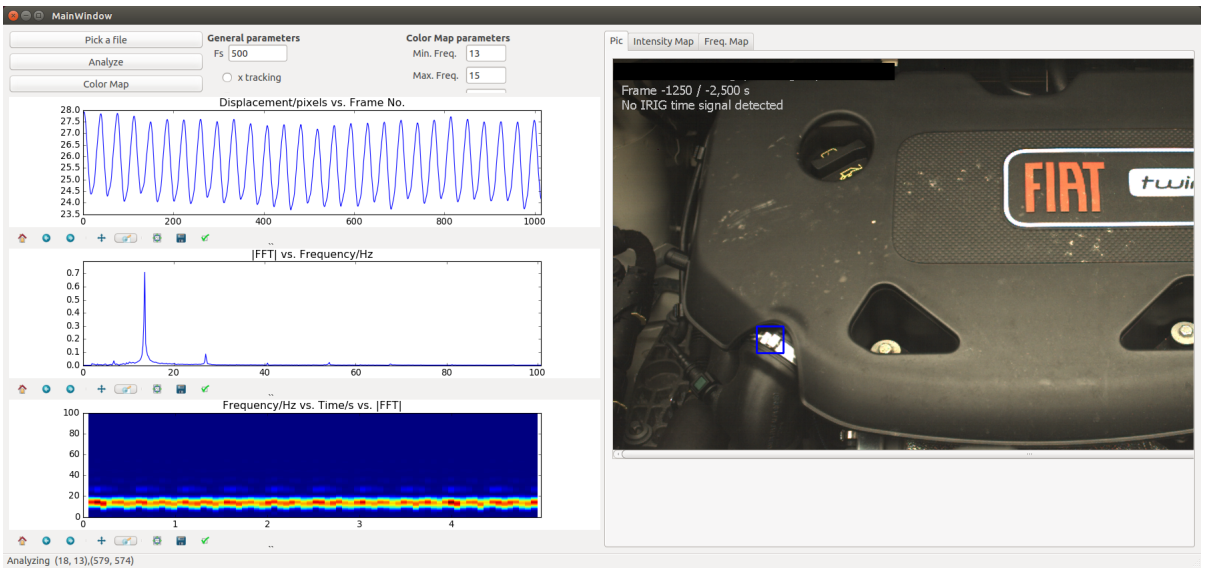
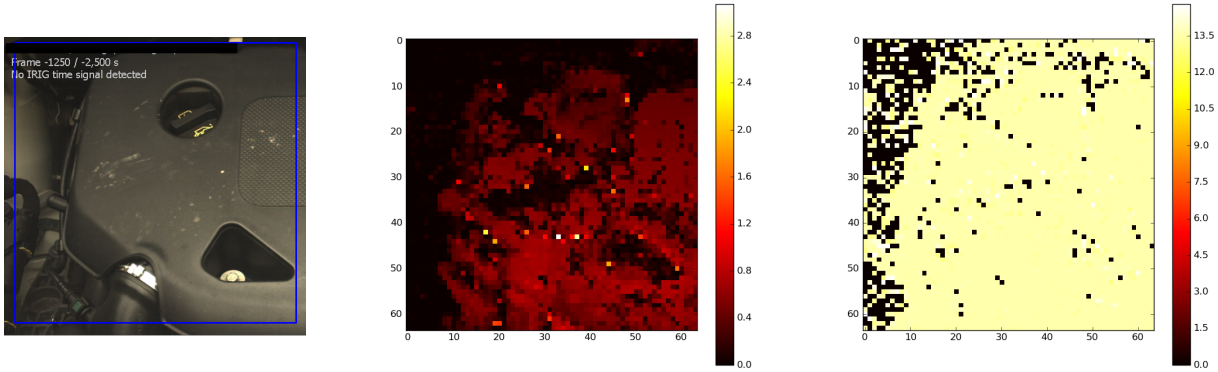


Figure 4.6: Vibration analysis on a car engine at idle.



(a) (b) (c)

Figure 4.7: (a) Region considered for the calculation of frequency maps. (b) Frequency strength. (c) Peak frequency.

not belonging to the engine are associated with very low peak strengths so one can clearly differentiate the edge of the engine on the left hand side of the strength map. Within the engine area, however, there exist regions with higher peak strengths than others. Looking at the original image it is easy to see that white marks over the black plastic cover help by adding the texture necessary for the tracker to perform better (some pixels in the color map are even white or yellow), whereas clean areas with no marks appear much darker.

Overall, the initial chosen location over the metal area seems to be adequate for the tracking, as it does not belong to any of the darker regions. Also, it is interesting to see in the color map for the actual frequency (image (c)) that even in areas outside of the engine the main frequency is still detected, even though the strength is very small. This may well be explained by the vibration of the engine transmitting to certain extend to the body of the car.

4.2.2 Example 2: Rotary equipment testing

This is a good example to see how different parts of a structure can resonate at much larger amplitudes than others, although it is also the only example where the ground truth is not available (the video was provided by a third-party). It consists of an industrial rotary component mounted on the platform of a shaker that ramps down in frequency. A couple of accelerometers are installed on the sides of one of the structure's edge, so it seems natural to try to detect vibration using the tool near those locations. The results are shown in Figure 4.8.

The vertical displacement profile (top plot) for this test shows a drift of around 4 pixels occurring approximately between 1700 to 3800 frames (1.05 s interval at the frame rate of 2000 fps). The vibrations within this time interval are so intense that the tracker has difficulties to precisely follow the movement and the bounding box slightly drifts downwards in the frame. Both examination of the video for this tests and the spectrogram confirms the heavy vibration in this time interval. The spectrogram shows the slow linear descend of the shaker's frequency and how at time around 1.6 s in the sequence the structure starts to resonate, reaching a peak at around 1.9 s and abruptly decreasing after that time. The frequency content displayed in the central plot shows a strong peak at 626 Hz, which by comparing with the spectrogram can be easily identified as the mentioned resonance.

Using some minimum and maximum values around the resonance frequency, color

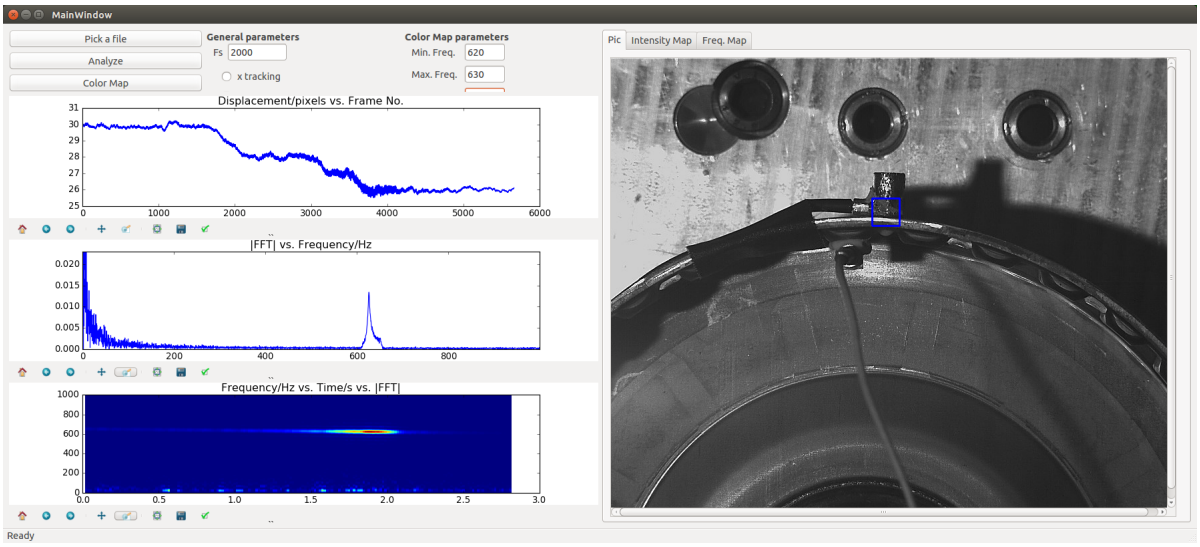
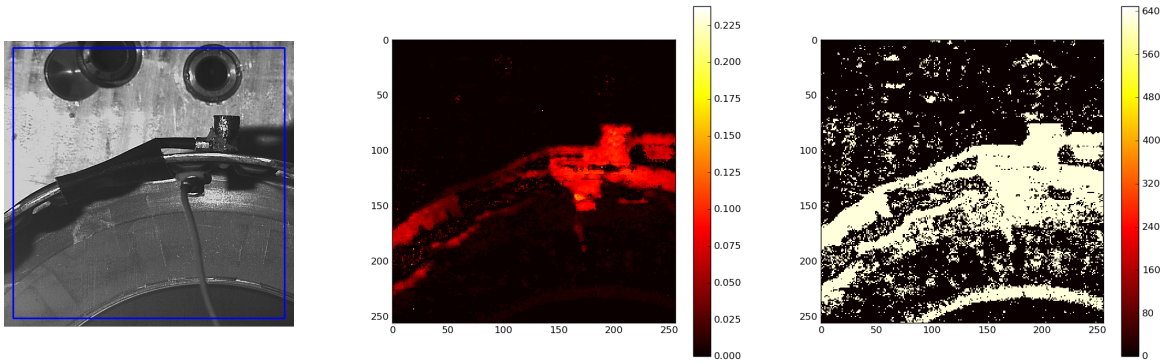


Figure 4.8: Vibration analysis on rotary industrial equipment.

maps have been created with a very high resolution of 256x256 pixels. They are shown in Figure 4.9 and illustrate how the regions around the accelerometers present the higher peaks in the associated spectra. In map (c) we see that many regions within the



(a) (b) (c)

Figure 4.9: (a) Region considered for the calculation of frequency maps for example 2. (b) Frequency strength. (c) Peak frequency.

structure present a frequency peak at 626 Hz. In particular, there is a rounded band near the bottom of the image that corresponds with a change of texture in the structure. This gradient-intense band does not show up in the strength map, which suggests that the bright red colors in map (b) may be due mainly to the vibration itself, and not so

much to the gradient of the image.

There are also two interesting features that can be seen in the maps. One is the shadow of heavily moving parts, which also generate bright colors, as can be seen in the lower left part. This must be taken into account when interpreting the maps. A second feature, not proved but arguably likely, is the visual detection of a vibration node [75]. Both the accelerometer region and the left part of the strength map exhibit large values of peak strength. Between these two regions, however, the strengths smoothly decrease as one moves to the midpoint, being minimum at it. This is compatible with a vibration node of minimum amplitude, with the mentioned two regions being the anti-nodes.

The current example reveals the importance of separating the effects of the gradient from those due to vibration if one is to draw conclusions about the vibration energy from the color maps. One partial solution to this problem would be to have a gradient image as a reference to interpret the true origin of the color in these maps but in any case, the maps are a valuable tool in order to decide in which part of an image the tracking should be performed.

4.2.3 Example 3: Vibrating mobile phone

Mobile phone vibra-motors have a vibration frequency which generally lies between 190 to 250Hz [76]. In this example, a vibrating mobile phone placed on top of a table has been recorded at a frame rate of 1000 fps. As a ground truth, sound produced by the vibrations of that same phone have been recorded using the audio tool Audacity [77]. The audio spectrogram is shown in Figure 4.10 for the duration of a single vibration sequence (the vibration between to silences), where as usual the vertical axis is the frequency, the horizontal is the time and the color encodes the frequency magnitude. The spectrogram evidences the existence of rich frequency content between approximately 200 and 300 Hz during the vibration sequence.

The result of tracking a small area of the phone corresponding to one of the corners of the device is shown in Figure 4.11. The amplitude of the vibrations in this example are very small and rather close to the noise level, as can be seen in the central plot. Still, two well differentiated frequency peaks appear at 250 and 322 Hz, which is highly consistent with the results of the spectrogram in Figure 4.10. Notice how the light conditions for this particular example are rather poor, suggesting that better illumination may be used to improve the results.

In Figure 4.12, the color maps show that the edges of the phone closer to the camera

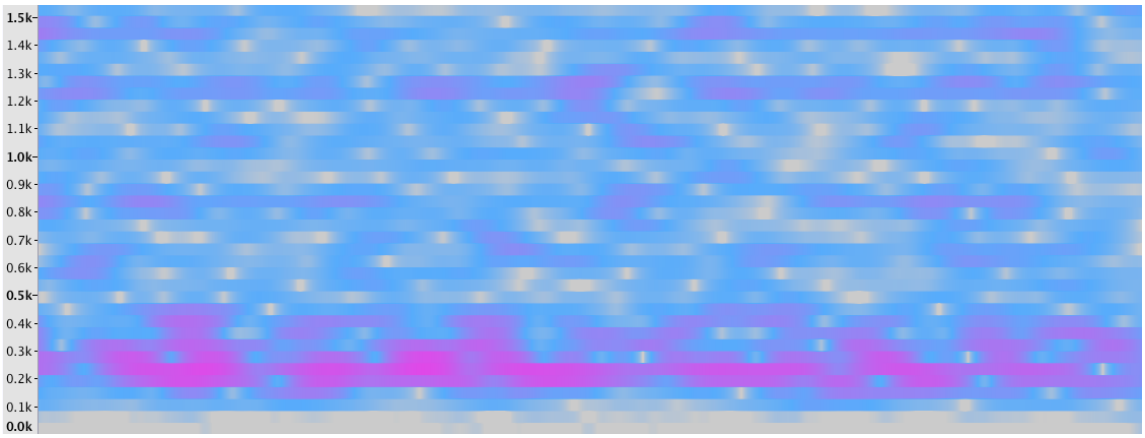


Figure 4.10: Spectrogram of the sound produced by a vibrating mobile phone.

are the best areas to choose for the tracking, with the corner being the best choice of all. Notice also that vibrations at the front part of the table next to the phone seem to be detected as well in the analysis, a situation similar to that of the car's body where vibrations were transmitted from the engine in Example 1.

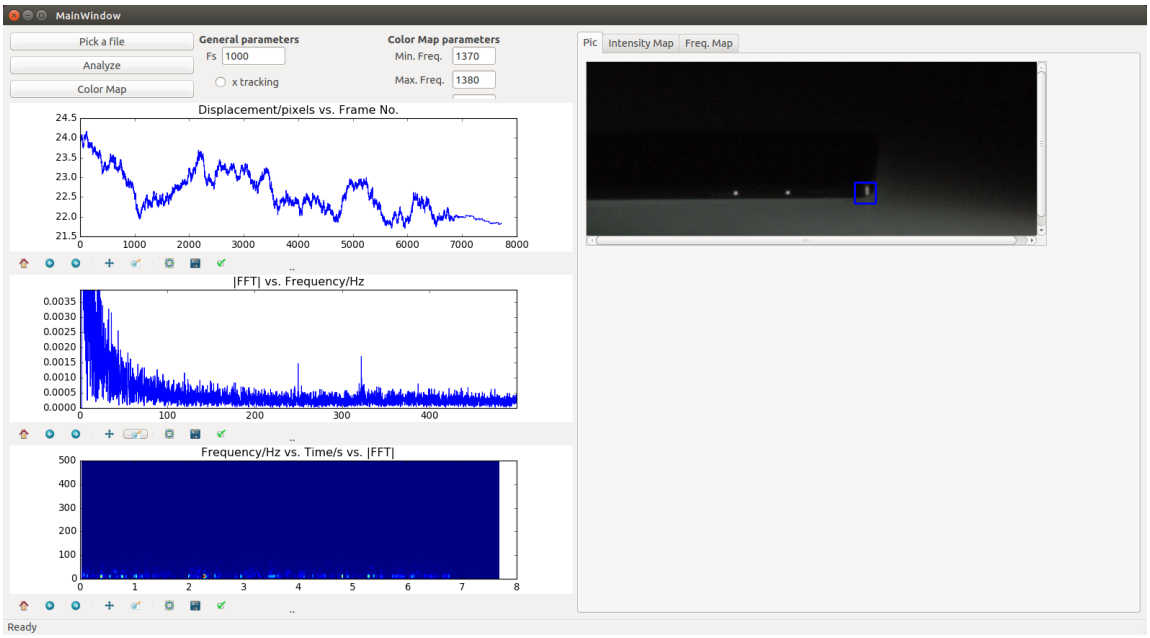
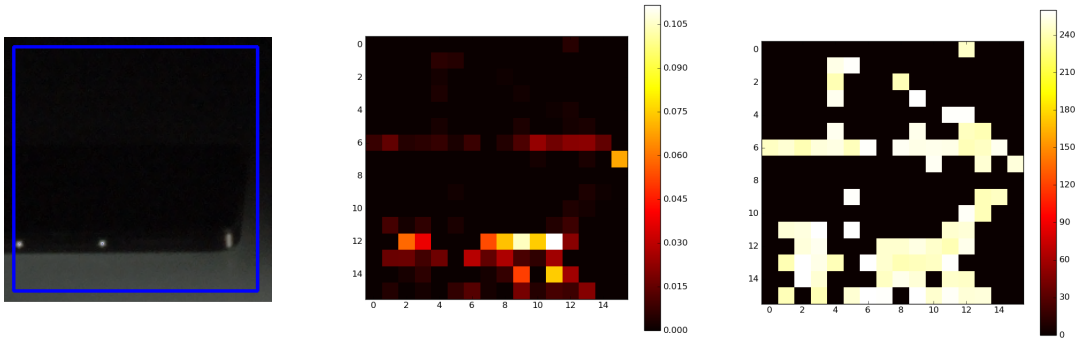


Figure 4.11: Vibration analysis in a vibrating mobile phone.



(a) (b) (c)

Figure 4.12: (a) Region considered for the calculation of frequency maps in Example 3. (b) Frequency strength. (c) Peak frequency.

4.2.4 Example 4: Cooling fan

A standard cooling fan has been recorded at 2000 fps. At that rate, by looking at the video it is easy to follow the movement of the three individual blades, from where the ground truth rotation frequency can be easily obtained. The aim of this test is to resolve the ground truth by looking at the vibration of an area comprising the protective grill of the fan.

A total of 84 frames were counted for a particular blade to complete a rotation cycle. At 2000 fps, this corresponds to a frequency $f = 1/(84/2000) = 23.8$ Hz. Figure 4.13 shows how the tool can perfectly detect the exact rotation speed of the fan by tracking the tiny vibrations of a small area of the protective gill.

4.2.5 Example 5: Glass hit with a spoon

In this last example, the sound produced by hitting a wine glass with a teaspoon has been recorded using Audacity. The audio spectrogram in shown on the left image of Figure 4.14. Within the frequency range displayed, main resonating frequencies appear at around 1500 and 3500 Hz, although weaker ones are also are present at higher frequencies. The video corresponding to this example was recorded at 4000 fps, so the higher frequency that can be detected with the tool is 2000 Hz. The right image in Figure 4.14 shows the FFT in decibel for approximately this range of frequencies. The fundamental frequency appears at exactly 1429 Hz (see cursor located at the highest peak).

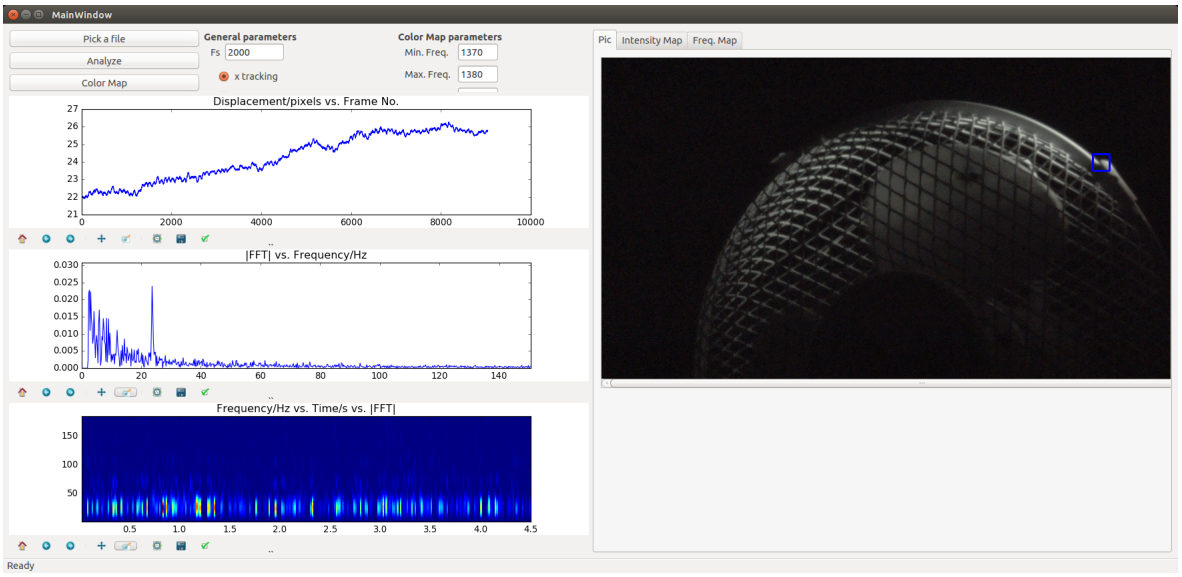


Figure 4.13: Vibration analysis on a cooling fan.

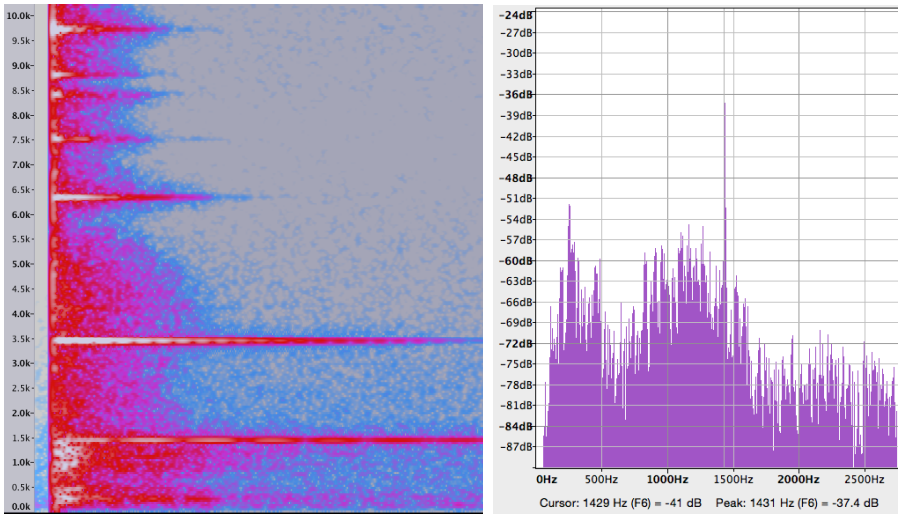


Figure 4.14: Left: Audio spectrogram of the sound produced by hitting a wine glass with a teaspoon. Right: FFT in decibel vs. Frequency. Images extracted from Audacity audio editing software.

In this example, the aim is to detect that lower fundamental frequency using the computer vision tool, whose results are depicted in Figure 4.15. This appears to be the most difficult test of all considered here. In fact, it was very difficult to choose a location for the bounding box that did not drift during the duration of the video, as

can be seen in the top plot of the figure. The analysed direction in this example was the horizontal one. The spectrum shows what it seems to be a peak at the frequency of the ground truth, but the signal is so weak that could be easily deemed as noise if the ground truth was not known in advance. Still, this example gives an idea of the real detection limit of the developed tool, which proves to be significantly sensitive to small vibrations. Again, further improvement on illumination conditions seem to be possible in order to get better results.

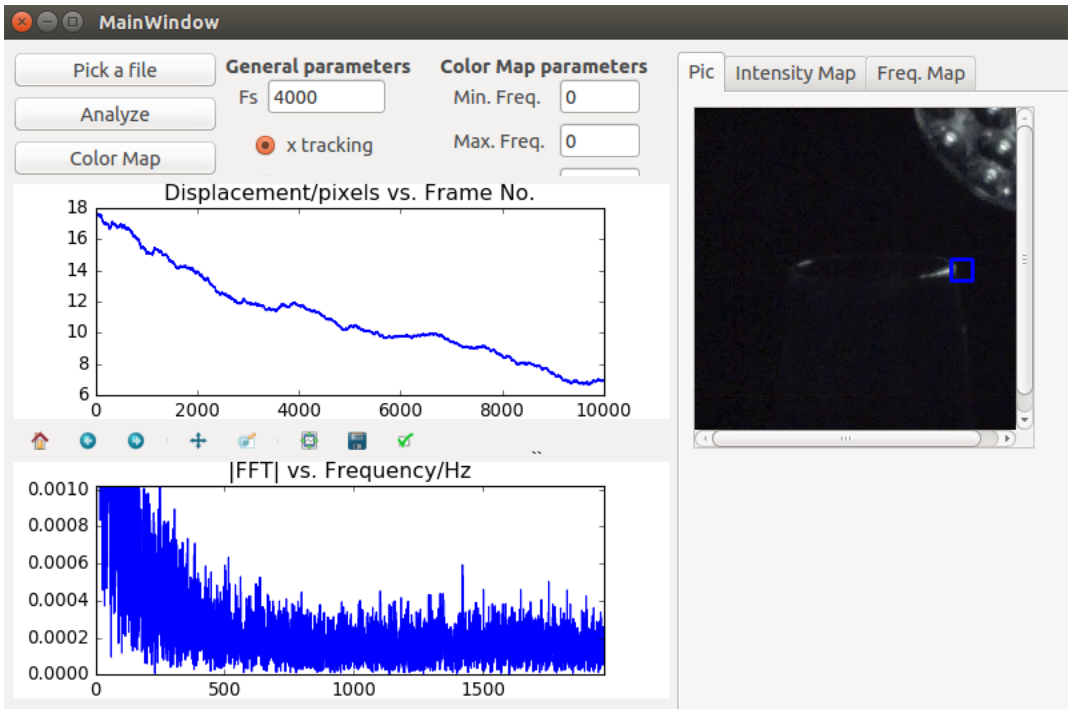


Figure 4.15: .

4.2.6 Conclusions

The tool explained and tested in this chapter demonstrates that it is possible to perform quite accurate and sensitive vibration analysis on high-speed videos recorded on a variety of real life situations.

As it will be mentioned in next chapter, the tool is only at a proof-of-concept stage and many improvements and extensions are soon too follow.

Chapter 5

Conclusions and future work

5.1 Conclusions

The aim of this thesis was to compare a number of well known video tracking algorithms, most of them included in the OpenCV library, and to establish their suitability in vibration analysis applications. Based on the outcome of this comparison, an interactive tool for modal analysis in videos of vibrating structures was developed and its use illustrated with some real live examples.

On a first set of experiments, synthetic videos containing an oscillatory moving cloud were employed for the tests. By simulating oscillation amplitudes of different orders of magnitude, it was shown how all considered trackers were capable of resolving modes associated with amplitudes of a few pixels. At lower amplitudes, however, the spatial resolution limitations of many of the trackers were evidenced, most of them showing quantisation effects for amplitudes at around one pixel. This generally translated into harmonic-rich spectra that interfered with the mode detection process. It was also shown that adding a substantial amount of Gaussian white noise to the videos helped to removed the excess of harmonics in those trackers more heavily affected by quantisation effects, although otherwise it affected negatively. Tests were also performed at deep sub-pixel level, where the Median Flow appeared to be the only tracker able to detect the simulated frequencies. Processing times varied greatly, the slower trackers taking as long as three orders of magnitude longer to complete than the fastest.

On a second set of experiments, real recordings obtained from standard shaker tests working at different frequencies were investigated. The tests covered a range of amplitudes that again spanned several orders of magnitude. Two of the trackers,

MeanShift and CamShift, were excluded from the tests due to poor performance or instabilities. The shaker's ground truth was estimated using its own experimental parameters and agreed well with the results obtained from the tracking. However, similar quantisation effects to those seen for the synthetic cloud appeared for tests where the oscillation amplitude of the shaker was similar to the spatial resolution limit of the trackers. For oscillation amplitudes in the sub-pixel region, only Median Flow and CMT trackers were able to resolve the tested frequency, with the former exhibiting a much greater spatial resolution.

The third performed experiment consisted of an aircraft landing gear laboratory test. The ground truth moving trajectory was provided by accelerometer measurements and mean least square errors comparisons were made against the results of the trackers, which in addition to MeanShift and CamShift, also excluded TLD due to instabilities. A measure of the error turned out to be not conclusive due to the high noise content of the ground truth, but quantisation effects due to poor resolution were clearly displayed by most of the trackers, with the exception of the Median Flow.

The major outcome obtained from the experiments was the increased performance of the Median Flow as compared with the other trackers considered, specially regarding its spatial resolution. Based on this, the influence of the initial bounding box location on the outcome of the tracking was investigated for the Median Flow tracker on the shaker tests. Results showed that those bounding boxes with larger content of edges perpendicular to the direction of movement provided a cleaner, better tracking, which translated into higher peaks in the frequency domain. The peak heights were plotted against the average vertical gradient content of the bounding boxes and revealed a fairly good linear correlation for all the tests.

In order to get a visual representation of the best locations for tracking within a video, a methodology consisting on frequency color maps was created. For each frequency of interest, pixels in the color maps were related to initial bounding box locations and properties such as the frequency-of-interest peak height or the frequency of the peak itself were used to codify the pixel color. The resulting maps were also seen as a tool to identify those regions within a vibrating structure that resonate more energetically.

The knowledge acquired during this work has made possible the creation of a software interactive tool whose functionality comprises most of what it has been investigated. A description of the tool has been provided and real-life examples showing its capabilities demonstrated.

The main general conclusion of this thesis is perhaps that it is possible to use computer vision techniques for modal analysis on structures that present even the tiniest vibrations, often invisible to the naked eye. Computer vision offers a cost-effective, contactless alternative to more standard modal analysis technologies and so it is expected that we will see more applications using techniques from this area in the near future.

5.2 Future work

Continuing this work will certainly focus on adding functionality to the modal analysis tool. At the moment, it only supports the Median Flow tracker. Depending on the application, however, there may be situations where one would choose a different tracker. For instance, optical flow trackers are good at detecting tiny vibrations but for larger ones, a slightly faster tracker such as Template Matching or a more robust one like KCF could in principle be good alternatives. Also, as discussed in Section 3.1.2.2, a more thorough investigation on the effect of added noise to the sensitivity of the different trackers would be of great interest and, depending on results, the implementation of a tool functionality that allows the addition of various types and levels of noise will be considered.

Even for each individual tracker, extended functionality is likely to be added. For example, a possible solution to avoid the drifting problems seen in some of the examples for the Median Flow could be an increase in the number of pyramid levels utilised in the optical flow calculation. This can make the tracker more robust to larger displacements and, in this sense, the tool would benefit from an option that permits selecting the number of levels to be used. Along these lines, the possibility to customise the OpenCV Median Flow implementation to specific needs is also likely to be examined.

Related to the color maps discussed in the previous two chapters, it would be interesting to find ways to decouple the influence of the image gradient from that of the actual vibrations, since that would allow a clear picture of the parts of a structure that vibrate more energetically.

Finally, stereo vision is one of the main fields of interest to this project in the future, since it can be used to analyse vibration in the direction perpendicular to the video plane. We are interested for instance in investigating how tracking techniques can be used in stereo video sequences in order improve or accelerate the detection of disparities among left and right views.

Appendices

Appendix A

Subpixel resolution in optical flow trackers

While many of the trackers discussed in this work achieve a spatial resolution in the order of 0.5-1 pixels, Median Flow or TLD work on a much higher, sub-pixel resolution. This is because the are based on calculations of the optical flow, which make use of spatial and temporal pixel intensity gradients.

For an 8-bit image the the number of intensity levels available is 256. Looking at the optical flow constrain equation and assuming movement in the vertical direction only:

$$I_y v + I_t = 0 \ , \tag{A.1}$$

Since

$$v = -I_t / I_y \ , \tag{A.2}$$

the minimum vertical displacement achievable will be that corresponding to a minimum change in intensity over time and a maximum change in intensity in the vertical direction, i.e., $v_{min} = 1/256$. Hence, we see that under the appropriate time and spatial gradient conditions, a theoretical resolution of 4×10^{-3} pixels is possible, making optical flow tracking methods specially suitable for small movements.

Bibliography

[1] Barton Zwiebach. *A first course in string theory*. Cambridge: Cambridge Univ. Press, 2004. URL: https://cds.cern.ch/record/789942.

[2] URL: https://www.britannica.com/science/vibration.

[3] H 1 Ising, B Kruppa, et al. "Health effects caused by noise: evidence in the literature from the past 25 years". In: *Noise and Health* 6.22 (2004), p. 5.

[4] E Peter Carden and Paul Fanning. "Vibration based condition monitoring: a review". In: *Structural health monitoring* 3.4 (2004), pp. 355–377.

[5] K Yusuf Billah and Robert H Scanlan. "Resonance, Tacoma Narrows bridge failure, and undergraduate physics textbooks". In: *American Journal of Physics* 59.2 (1991), pp. 118–124.

[6] URL: https://en.wikipedia.org/wiki/Aeroelasticity#Flutter.

[7] Jamilur Reza Choudhury and Ariful Hasnat. "Bridge collapses around the world: Causes and mechanisms". In: *IABSE-JSCE Join Conference on Advances in Bridge Engineering-III* (2015).

[8] Juha Plunt. "Finding and fixing vehicle NVH problems with transfer path analysis". In: *Sound and vibration* 39.11 (2005), pp. 12–17.

[9] Maria Antonietta Panza. "A Review of Experimental Techniques for NVH Analysis on a Commercial Vehicle". In: *Energy Procedia* 82 (2015), pp. 1017–1023.

[10] Klaus Genuit. "The sound quality of vehicle interior noise: a challenge for the NVH-engineers". In: *International journal of vehicle noise and vibration* 1.1-2 (2004), pp. 158–168.

[11] Paul Hayton et al. "Support vector novelty detection applied to jet engine vibration spectra". In: *NIPS*. 2000, pp. 946–952.

[12] URL: http://www.boeing.com/commercial/aeromagazine/aero_16/vibration_story.html.

[13] Anders Brandt. *Noise and vibration analysis: signal analysis and experimental procedures*. John Wiley & Sons, 2011.

[14] URL: https://en.wikipedia.org/wiki/Modal_analysis.

[15] Tirupathi R Chandrupatla et al. *Introduction to finite elements in engineering*. Vol. 2. Prentice Hall Upper Saddle River, NJ, 2002.

[16] Thomas JR Hughes. *The finite element method: linear static and dynamic finite element analysis*. Courier Corporation, 2012.

[17] C. Fosalau and O. Postolache. *Computer Aided Design of the Measurement Systems*. Academic Press, 1999.

[18] Demeter G Fertis. *Mechanical and structural vibrations*. John Wiley & Sons, 1995.

[19] Mohammad Reza Ashory. "High quality modal testing methods". PhD thesis. University of London, 1999.

[20] Brian J Schwarz and Mark H Richardson. "Experimental modal analysis". In: *CSI Reliability week* 35.1 (1999), pp. 1–12.

[21] URL: https://en.wikipedia.org/wiki/Modal_testing.

[22] Philip Ind. "The non-intrusive modal testing of delicate and critical structures". PhD thesis. University of London, 2004.

[23] Parameswaran Hariharan. *Basics of interferometry*. Academic Press, 2010.

[24] C. Sujatha. *Vibration And Acoustics*. McGraw-Hill Education (India) Pvt Limited, 2010. ISBN: 9780070148789. URL: https://books.google.de/books?id=wnPNkbJUBxkC.

[25] Justin G Chen et al. "Developments with Motion Magnification for Structural Modal Identification Through Camera Video". In: *Dynamics of Civil Structures, Volume 2*. Springer, 2015, pp. 49–57.

[26] JK Aggarwal. "Motion analysis: Past, present and future". In: *Distributed Video Sensor Networks*. Springer, 2011, pp. 27–39.

[27] Alper Yilmaz, Omar Javed, and Mubarak Shah. "Object tracking: A survey". In: *Acm computing surveys (CSUR)* 38.4 (2006), p. 13.

[28] Brendan Tran Morris and Mohan Manubhai Trivedi. "A survey of vision-based trajectory learning and analysis for surveillance". In: *IEEE transactions on circuits and systems for video technology* 18.8 (2008), pp. 1114–1127.

[29] Sayanan Sivaraman and Mohan Manubhai Trivedi. "Looking at vehicles on the road: A survey of vision-based vehicle detection, tracking, and behavior analysis". In: *IEEE Transactions on Intelligent Transportation Systems* 14.4 (2013), pp. 1773–1795.

[30] Arnold WM Smeulders et al. "Visual tracking: An experimental survey". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 36.7 (2014), pp. 1442–1468.

[31] Hanxuan Yang et al. "Recent advances and trends in visual tracking: A review". In: *Neurocomputing* 74.18 (2011), pp. 3823–3831.

[32] Piotr Olaszek. "Investigation of the dynamic characteristic of bridge structures using a computer vision method". In: *Measurement* 25.3 (1999), pp. 227–236.

[33] S Patsias and WJ Staszewskiy. "Damage detection using optical measurements and wavelets". In: *Structural Health Monitoring* 1.1 (2002), pp. 5–22.

[34] E Caetano, S Silva, and J Bateira. "A vision system for vibration monitoring of civil engineering structures". In: *Experimental Techniques* 35.4 (2011), pp. 74–82.

[35] Justin G Chen et al. "Structural modal identification through high speed camera video: Motion magnification". In: *Topics in Modal Analysis I, Volume 7*. Springer, 2014, pp. 191–197.

[36] Ce Liu et al. "Motion magnification". In: *ACM transactions on graphics (TOG)* 24.3 (2005), pp. 519–526.

[37] Abe Davis et al. "Visual vibrometry: Estimating material properties from small motions in video". In: *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE. 2015, pp. 5335–5343.

[38] URL: `http://opencv.org/opencv-3-1.html`.

[39] Boris Babenko, Ming-Hsuan Yang, and Serge Belongie. "Visual tracking with online multiple instance learning". In: *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*. IEEE. 2009, pp. 983–990.

[40] Thomas G Dietterich, Richard H Lathrop, and Tomás Lozano-Pérez. "Solving the multiple instance problem with axis-parallel rectangles". In: *Artificial intelligence* 89.1 (1997), pp. 31–71.

[41] Helmut Grabner, Christian Leistner, and Horst Bischof. "Semi-supervised on-line boosting for robust tracking". In: *European conference on computer vision*. Springer. 2008, pp. 234–247.

[42] Constantine P Papageorgiou, Michael Oren, and Tomaso Poggio. "A general framework for object detection". In: *Computer vision, 1998. sixth international conference on*. IEEE. 1998, pp. 555–562.

[43] Paul Viola and Michael Jones. "Rapid object detection using a boosted cascade of simple features". In: *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*. Vol. 1. IEEE. 2001, pp. I–511.

[44] Helmut Grabner, Michael Grabner, and Horst Bischof. "Real-time tracking via on-line boosting." In: *BMVC*. Vol. 1. 5. 2006, p. 6.

[45] Yoav Freund and Robert E Schapire. "A desicion-theoretic generalization of on-line learning and an application to boosting". In: *European conference on computational learning theory*. Springer. 1995, pp. 23–37.

[46] Gregory D Hager and Peter N Belhumeur. "Efficient region tracking with parametric models of geometry and illumination". In: *IEEE transactions on pattern analysis and machine intelligence* 20.10 (1998), pp. 1025–1039.

[47] Bruce D Lucas, Takeo Kanade, et al. "An iterative image registration technique with an application to stereo vision." In: *IJCAI*. Vol. 81. 1. 1981, pp. 674–679.

[48] Rafael C. Gonzalez and Richard E. Woods. *Digital Image Processing (3rd Edition)*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 2006. ISBN: 013168728X.

[49] J. L. Barron, D. J. Fleet, and S. S. Beauchemin. "Performance of Optical Flow Techniques". In: *Int. J. Comput. Vision* 12.1 (Feb. 1994), pp. 43–77. ISSN: 0920-5691. DOI: 10.1007/BF01420984. URL: http://dx.doi.org/10.1007/BF01420984.

[50] URL: https://en.wikipedia.org/wiki/Motion_perception#The_aperture_problem.

[51] Zdenek Kalal, Krystian Mikolajczyk, and Jiri Matas. "Forward-backward error: Automatic detection of tracking failures". In: *Pattern recognition (ICPR), 2010 20th international conference on*. IEEE. 2010, pp. 2756–2759.

[52] Jean-Yves Bouguet. "Pyramidal implementation of the affine lucas kanade feature tracker description of the algorithm". In: *Intel Corporation* 5.1-10 (2001), p. 4.

[53] João F Henriques et al. "Exploiting the circulant structure of tracking-by-detection with kernels". In: *European conference on computer vision*. Springer. 2012, pp. 702–715.

[54] Chang Huang, Bo Wu, and Ramakant Nevatia. "Robust object tracking by hierarchical association of detection responses". In: *European Conference on Computer Vision*. Springer. 2008, pp. 788–801.

[55] Michael D Breitenstein et al. "Online multiperson tracking-by-detection from a single, uncalibrated camera". In: *IEEE transactions on pattern analysis and machine intelligence* 33.9 (2011), pp. 1820–1833.

[56] Georg Nebehay and Roman Pflugfelder. "Consensus-based matching and tracking of keypoints for object tracking". In: *IEEE Winter Conference on Applications of Computer Vision*. IEEE. 2014, pp. 862–869.

[57] Georg Nebehay and Roman Pflugfelder. "Clustering of Static-Adaptive Correspondences for Deformable Object Tracking". In: *Computer Vision and Pattern Recognition*. IEEE, June 2015.

[58] K Chidananda Gowda and G Krishna. "Agglomerative clustering using the concept of mutual nearest neighbourhood". In: *Pattern recognition* 10.2 (1978), pp. 105–112.

[59] Keinosuke Fukunaga and Larry Hostetler. "The estimation of the gradient of a density function, with applications in pattern recognition". In: *IEEE Transactions on information theory* 21.1 (1975), pp. 32–40.

[60] Dorin Comaniciu and Peter Meer. "Mean shift: A robust approach toward feature space analysis". In: *IEEE Transactions on pattern analysis and machine intelligence* 24.5 (2002), pp. 603–619.

[61] Gary R Bradski. "Computer vision face tracking for use in a perceptual user interface". In: *Citeseer* (1998).

[62] Sanjit Kumar Mitra and Yonghong Kuo. *Digital signal processing: a computer-based approach*. Vol. 2. McGraw-Hill New York, 2006.

[63] Richard H Lyon. *Machinery noise and diagnostics*. Butterworth-Heinemann, 2013.

[64] John C Russ. *The image processing handbook*. CRC press, 2016.

[65] Júlio M Montalvão e Silva and Nuno MM Maia. *Modal analysis and testing*. Vol. 363. Springer Science & Business Media, 2012.

[66] Daniel Arfib. "Digital synthesis of complex spectra by means of multiplication of non linear distorted sine waves". In: *Audio Engineering Society Convention 59*. Audio Engineering Society. 1978.

[67] Mark J Schervish and MH DeGroot. *Probability and Statistics*. Pearson Education, 2014.

[68] URL: `https://en.wikipedia.org/wiki/Root-mean-square_deviation`.

[69] URL: `https://en.wikipedia.org/wiki/Matched_filter`.

[70] B.H. Tongue. *Principles of Vibration*. Oxford University Press, 2002. ISBN: 9780195142464. URL: `https://books.google.co.uk/books?id=wAGqXVImUjYC`.

[71] John Robert Taylor. *Classical mechanics*. University Science Books, 2005.

[72] URL: `https://en.wikipedia.org/wiki/Fluorescent_lamp`.

[73] URL: `https://en.wikipedia.org/wiki/Signal-to-noise_ratio`.

[74] URL: `https://www.qt.io/`.

[75] URL: `https://en.wikipedia.org/wiki/Node_(physics)`.

[76] Nirupam Roy, Mahanth Gowda, and Romit Roy Choudhury. "Ripple: Communicating through physical vibration". In: *12th USENIX Symposium on Networked Systems Design and Implementation (NSDI 15)*. 2015, pp. 265–278.

[77] URL: `http://www.audacityteam.org/`.