UNIVERSIDAD DE INVESTIGACIÓN DE TECNOLOGÍA EXPERIMENTAL YACHAY

Escuela de Ciencias Matemáticas y Computacionales

TÍTULO: Optical flow sensor to measure the displacement of objects in videos

Trabajo de integración curricular presentado como requisito para la obtención del título de Ingeniero en Tecnologías de la Información

Autor:

Morejón Macías Winter Joshue

Tutor:

Ph.D Chang Tortolero Oscar Guillermo

Urcuquí, abril 2020

Urcuquí, 8 de abril de 2020

SECRETARÍA GENERAL
(Vicerrectorado Académico/Cancillería)
ESCUELA DE CIENCIAS MATEMÁTICAS Y COMPUTACIONALES
CARRERA DE TECNOLOGÍAS DE LA INFORMACIÓN
ACTA DE DEFENSA No. UITEY-ITE-2020-00023-AD

A los 8 días del mes de abril de 2020, a las 10:00 horas, de manera virtual mediante videoconferencia, y ante el Tribunal Calificador, integrado por los docentes:

Presidente Tribunal de Defensa	Dr. IZA PAREDES, CRISTHIAN RENE , Ph.D.
Miembro No Tutor	Dr. CUENCA PAUTA, ERICK EDUARDO , Ph.D.
Tutor	Dr. CHANG TORTOLERO, OSCAR GUILLERMO , Ph.D.

El(la) señor(ita) estudiante **MOREJON MACIAS, WINTER JOSHUE**, con cédula de identidad No. **1311712416**, de la **ESCUELA DE CIENCIAS MATEMÁTICAS Y COMPUTACIONALES**, de la Carrera de **TECNOLOGÍAS DE LA INFORMACIÓN**, aprobada por el Consejo de Educación Superior (CES), mediante Resolución **RPC-SO-43-No.496-2014**, realiza a través de videoconferencia, la sustentación de su trabajo de titulación denominado: **OPTICAL FLOW SENSOR TO MEASURE THE DISPLACEMENT OF OBJECTS IN VIDEO.**, previa a la obtención del título de **INGENIERO/A EN TECNOLOGÍAS DE LA INFORMACIÓN**.

El citado trabajo de titulación, fue debidamente aprobado por el(los) docente(s):

Tutor	Dr. CHANG TORTOLERO, OSCAR GUILLERMO , Ph.D.

Y recibió las observaciones de los otros miembros del Tribunal Calificador, las mismas que han sido incorporadas por el(la) estudiante.

Previamente cumplidos los requisitos legales y reglamentarios, el trabajo de titulación fue sustentado por el(la) estudiante y examinado por los miembros del Tribunal Calificador. Escuchada la sustentación del trabajo de titulación a través de videoconferencia, que integró la exposición de el(la) estudiante sobre el contenido de la misma y las preguntas formuladas por los miembros del Tribunal, se califica la sustentación del trabajo de titulación con las siguientes calificaciones:

Tipo	Docente	Calificación
Tutor	Dr. CHANG TORTOLERO, OSCAR GUILLERMO , Ph.D.	8,0
Presidente Tribunal De Defensa	Dr. IZA PAREDES, CRISTHIAN RENE , Ph.D.	9,0
Miembro Tribunal De Defensa	Dr. CUENCA PAUTA, ERICK EDUARDO , Ph.D.	7,0

Lo que da un promedio de: **8 (Ocho punto Cero)**, sobre 10 (diez), equivalente a: **APROBADO**

Para constancia de lo actuado, firman los miembros del Tribunal Calificador, el/la estudiante y el/la secretario ad-hoc.

MOREJON MACIAS, WINTER JOSHUE
Estudiante

CRISTHIAN RENE IZA PAREDES
Digitally signed by CRISTHIAN RENE IZA PAREDES
Date: 2020.05.29 20:48:56 -05'00'

Dr. IZA PAREDES, CRISTHIAN RENE , Ph.D.
Presidente Tribunal de Defensa

Dr. CHANG TORTOLERO, OSCAR GUILLERMO , Ph.D.
Tutor

Firmado electrónicamente por:
OSCAR GUILLERMO CHANG TORTOLERO

ERICK EDUARDO CUENCA PAUTA

Signature numérique de ERICK EDUARDO CUENCA PAUTA
Date : 2020.05.28 21:51:22 -05'00'

Dr. CUENCA PAUTA, ERICK EDUARDO , Ph.D.
Miembro No Tutor

DAYSY MARGARITA MEDINA BRITO

Firmado digitalmente por DAYSY MARGARITA MEDINA BRITO
Fecha: 2020.06.11 21:04:10 -05'00'

MEDINA BRITO, DAYSY MARGARITA
Secretario Ad-hoc

AUTORÍA

Yo, **Winter Joshue Morejón Macías**, con cédula de identidad 1311712416, declaro que las ideas, juicios, valoraciones, interpretaciones, consultas bibliográficas, definiciones y conceptualizaciones expuestas en el presente trabajo; así cómo, los procedimientos y herramientas utilizadas en la investigación, son de absoluta responsabilidad de el/la autora (a) del trabajo de integración curricular. Así mismo, me acojo a los reglamentos internos de la Universidad de Investigación de Tecnología Experimental Yachay.

Urcuquí, abril y 2020

Winter Joshue Morejón Macías
CI: 1311712416

AUTORIZACIÓN DE PUBLICACIÓN

Yo, **Winter Joshue Morejón Macías**, con cédula de identidad 131172416, cedo a la Universidad de Tecnología Experimental Yachay, los derechos de publicación de la presente obra, sin que deba haber un reconocimiento económico por este concepto. Declaro además que el texto del presente trabajo de titulación no podrá ser cedido a ninguna empresa editorial para su publicación u otros fines, sin contar previamente con la autorización escrita de la Universidad.

Así mismo, autorizo a la Universidad que realice la digitalización y publicación de este trabajo de integración curricular en el repositorio virtual, de conformidad a lo dispuesto en el Art. 144 de la Ley Orgánica de Educación Superior

Urcuquí, abril y 2020.

Winter Joshue Morejón Macías
CI: 1311712416

Dedicatoria

El presente proyecto es dedicado a Dios, ya que gracias a él he logrado concluir mi carrera universitaria.

A mi madre con mucho amor y cariño le dedico todo mi esfuerzo y trabajo puesto para llevar a cabo esta tesis.

A mi familia por haber sido mi pilar fundamental y brindarme consejos, confianza y oportunidades para lograr esta meta.

Winter Joshue Morejón Macías

Agradecimiento

En primer lugar, a Dios por haberme guiado por el camino correcto hasta este momento.

A mis padres por haberme forjado como la persona que soy en la actualidad y me motivaron con constancia para alcanzar mis metas.

A todos los que son miembros de mi familia por ser el motor de todas mis metas y por quien cada día quiero ser mejor tanto en lo personal como en lo profesional.

A mi tutor Oscar Chang por ser mi guía y apoyo en todo el proceso de realización de este arduo trabajo.

A los miembros de la Escuela de Ciencias Matemáticas y Computacionales, tanto docentes como administrativos, por su constante apoyo para finalizar este trabajo.

A mis amigos, los cuales me han acompañado a lo largo de esta hermosa etapa universitaria y han hecho de esta etapa una experiencia única y que sin su apoyo no pudiera haber logra esta meta. En especial a Claudia Moncada, Sergio Hidalgo, Osiris Román, Kevin Chamorro, David Garcia, Carlos Bustamante, Emil Vega, William Arrellano, y Paul Silva.

A todos les agradezco y para todos ellos hago esta mención.

Winter Joshue Morejón Macías

Resumen

Determinar el desplazamiento de objetos es un desafío persistente en la inteligencia artificial y visión por computadora. En términos de imágenes digitales, cualquier traslación de objetos produce un flujo de píxeles correspondiente a un archivo digital o video. En este proyecto de grado, proponemos un sensor de flujo óptico para medir el desplazamiento de manera global mediante la correlación de fase. Primero, al usar la transformada de Fourier, registramos la diferencia de desplazamiento entre las imágenes. Luego, en base a esta técnica, se determina la traslación de los objetos. Siguiendo este proceso, se puede determinar si un objeto se mueve en la dirección correcta y a la velocidad correcta en una determinada trayectoria. Mediante el uso de estas técnicas, el desarrollo urbano del tráfico vehicular de una ciudad se podría gestionar de manera eficiente. También podemos aplicar este procedimiento en el análisis del flujo de personas dentro de un edificio, para determinar posibles puntos de vulnerabilidad o mayor atención. Al agregar cualquier sistema entrenable, como redes neuronales artificiales, el flujo de píxeles se puede utilizar en instrumentos dedicados para detectar y reportar desastres naturales como incendios forestales, incendios urbanos, efectos de terremotos y tsunamis. El interés es generar un código práctico que pueda ejecutarse en una plataforma de hardware libre.

Palabras Claves: Arduino, Flujo Optico, Correlacion de Fase, Lucas-Kanade.

Abstract

In this grade project, we propose determining the displacement of objects as a persistent challenge in artificial intelligence and computer vision. In terms of digital images, any object translation produces a flow of pixels in the corresponding digital file or video. In this grade project, we propose an optical flow sensor to measure displacement in a global manner by means of phase correlation. First, by using Fourier transform we record the difference in displacement between the images. Then, based on this technique, the translation of objects is determined. Following this process, we can know determine if an object is moving in the correct direction and at the correct speed in a certain trajectory. By using these techniques the urban development of a city vehicular traffic can be managed in an efficient way. We can also apply this procedure in the analysis of the flow of people inside a building, to determine possible points of vulnerability or greater attention. By adding any trainable system, such as artificial neural networks, pixel flow can be used in dedicated instruments to detect and report natural disasters like forest fires, urban fires, earthquakes effects and tsunamis. The interest is to generate a practical code that can be run on a free hardware platform.

Keyword: Arduino, Optical Flow, Phase correlation, Lucas-Kanade.

Contents

List of Figures

List of Tables

Introduction

Technology has made great advances today. The fields of computer science and electronics are increasingly able to create software and hardware that can simulate at a certain level the physical capabilities of people, and even be able to do something that a human could not. Among the things that computer science is capable of performing is computer vision with the help of different branches of computing, such as artificial intelligence. The computational vision allows understanding mathematically brain-eye functioning. In this way, we can reconstruct the properties of images and what is seen, such as shape, color, shadows, and more [2]. With the constant advance of technology, the interaction of human beings with technological systems has increased. The existing problems related to image processing and determining the displacement of objects has become a constant challenge. Therefore, these two situations represent two key problems within computer vision and have been the subject of extensive study for decades. The importance of optical flow lies primarily in a large number of applications it has, such as video compression, segmentation and object detection.

With this, determining the displacement of objects is a persistent challenge. Since there are cases, in which examining the flow could avoid traffic jams by allowing policies applied in areas of greater concurrency. In this work, an optical flow sensor is proposed to measure the displacement in a global way by means of the phase correlation. First, when using the Fourier transform, we record the displacement difference between the images. Then a better method will be applied, and it is more complex since it is based on the constancy of brightness of the video image, as it can be that of Lucas-Kanade or Horn-Schunck based on the determination of which is better prior detailed investigation forward. Then, based on this technique, the translation, direction and speed of the objects are determined. Following this process, we can know if an object is moving in the right direction, and speed in a certain trajectory. Then by using these techniques, the urban development of a city can be managed efficiently. It also has applications in the analysis of the flow of people inside a building, to determine possible points of vulnerability or greater attention.

Based on this premise, the implementation of optical flow algorithms and feature extraction algorithms to determine the recognition and detection of real-time objects of an image sequence is shown throughout the project. The main idea of this is to develop a sensor optical flow which allows two images to be compared and their displacement measured by the optical flow. The content of this thesis has focused on the study and development of methods or alternatives that allow determining certain variations of space and time to be able to define the calculation of the optical flow in image sequences.

In the previous literature, multiple types of techniques have been proposed to solve these two problems. Computer vision is a broad field of research in which properties and tools can be extracted which have been implemented by a series of sensors in the real world and adapted to the needs of each consumer.

Therefore, a respective comparison will be made with the values obtained in the two differential methods of optical flow such as Lucas-Kanade and Horn-Schunck respectively. The most representative method to perform this procedure is that of Lucas-Kanade due to the practicality of its handling and operation. It is estimated that the results obtained from the following work allow us to estimate the level of confidence, error, and limitations related to the algorithms of this technique.

Chapter 1

Preliminaries

1.1 Problem Statement

In all parts of the planet, traffic, crowds, traffic jams, and even accidents may occur due to the displacement of people or means of transport, in which factors such as speed and direction of the object intervene. This is also true for the real time analysis and detection of major natural disasters like earthquake, big fires and tsunamis [3] [4] [5]. For this, mechanisms are needed to control solve or prevent these problem. Currently, several methods of tracking objects, in real time from a camera or videos already recorded are based on the calculation of optical flow. It is optimized so that the computational cost is low. In the determination of optic flow, to give a solution to a specific detection or control problem, the use of microcontrollers may be a very important step because it will allow the construction of stand alone, robust sensors and detectors that can be readily integrated in a massive web network via the internet of things (IoT). These advances instruments could be placed on roads, buildings forest, sea shores or places where visual anomalies can occur.

The problem of optical flow estimation and image processing has been studied for years due to the wide range of applications to which they can be subjected. The content of this thesis has focused on the study and development of an optical flow sensor with the ability to analyze the displacement of objects in videos. To cite a general example in the application of these mechanisms is the field of meteorology. In general terms, the use of multispectral satellite images is very frequent. The analysis of the phenomena recorded in these sequences is essential to predict the weather. Thus, this is transported to other branches such as traffic, traffic jams, conglomerations of people in public spaces, air traffic, natural disasters evaluation and warning, among others.

The different optical flow techniques that have been developed with the main purpose of determining the position and displacement of objects are generally carried out in uncontrolled environments. This in turn generates external influence related to variable conditions in the environment, which translates into alterations to the main sources of information such as lighting, shadows and reflections. All these aspects are reflected in the final calculation or estimation of the optical flow, so it may not be making a sufficiently accurate estimate. Therefore, we seek to incorporate mechanisms that reduce the margin of error from abroad.

1.2 Justification

So far the research and development of techniques that allow to detect objects through image processing has increased greatly [6] [7]. This is justified thanks to the wide field of application that transcends even the field of routine life as it can be in traffic control [8]. Recent inquiries related to the study of traffic have focused mainly on the automatic analysis of vehicular and pedestrian flow. This has been achieved through the development of computer vision algorithms [9]. All this under the justification of the exponential and accelerated increase in traffic in large cities mainly, but also as it has transcended even small and medium cities. The use of computer vision in traffic monitoring systems can offer many advantages over other technologies.

There is a significant amount of research techniques which do not have a considerable reliability index related to the use of certain variables or uncontrolled conditions. These are those from outside and which alter the final calculation of the optical flow to some extent. This in turn would significantly alter the estimation or calculation of the optical flow. In general, its estimation does not turn out to be simple. In addition, the entire computation process for tests that come from a real environment can be complicated by the various external factors. For this reason the pre-established algorithms solve certain hypotheses such as intensity, shadows, opacity, brightness, contrast and brightness, just to name some of the most used.

In addition, the optical flow is presented as an alternative with countless applications in everyday areas such as security systems, digital processing and even industrial automation. All these represent sources of constant research due to their importance and implementations in the modern world. With all this, it is intended that new devices or applications generated from previous studies can gradually contribute to new technological devices that help to improve the state and well-being of society in general.

1.3 Contribution

The objective of this work is to perform a general analysis which allows establishing important aspects to determine the optical flow of a sequence of images provided. The main contributions made by carrying out this work have been based on three fundamental points, which are described as:

1. Analyze optical flow algorithms for displacement measurement, ensuring that they can be as accurate as necessary.

2. Implement various existing functions, which are easily adaptable, easy to use, and understand for any user who can use the algorithm.

3. Implement sensors capable of determining the displacement of objects and at the same time that they are low cost.

This could also be determined from a socioeconomic point of view. These sensors are positioned in various aspects of daily life, which is why they have represented a high cost for their production, maintenance and use. Each sensor varies depending on the particular need of the consumer, so that in cases of specific situations such as accident flow, pedestrian and vehicular

traffic, operational and logistics costs could be reduced with the implementation of this new mechanism. All this without jeopardizing the quality of the product or service provided as it follows the basic fundamentals of optical flow and does not skimp on its implementation. The direct contribution to the implementation of these techniques would represent a decrease in municipal and value-added taxes for consumers, which are finally used to implement these mechanisms. This would be a social and economic contribution in the implementation of these mechanisms.

In general, it is intended to introduce new mechanisms or techniques that minimize the difference in the results of the optical flow. This will help reduce possible error variations and reach results with greater precision. All this is possible, regardless of the displacement measurement of the study objects captured in sequential images.

1.4 Work Organization

This work is generally divided by 8 chapters which are defined as: Preliminaries, Objectives, Related Works, Theoretical Framework, Methodology, Results, Discussion and Conclusions and Future Work.

Chapter 1 provides details about the importance of real-time analysis and detection of object tracking methods based on the calculation of optical flow. Chapter 2 provides details of the general and specific objectives that are planned to be fulfilled once the investigation is completed. Chapter 3 provides theoretical details on key research concepts such as: optical flow, phase correlation, Fourier transformation and microcontrollers. Chapter 4 is an extension of works related to the central theme of the work. Highlights of related works include: image data processing for storage, development and characterization of a low-cost optical sensor module, optical flow sensor in a miniature vehicle and analysis on motion estimation techniques.
Also, chapter 5 sets out the methodology used, ranging from documentary research to implementation of the Fourier transform in Arduino. The Middlebury library is also established as a database, from which the phase correlation algorithm was used to perform the three tests. Chapter 6 presents the results on the comparison of the efficiency of the Lukas-Kanade algorithm and the implementation of the phase correlation algorithm. Chapter 7 highlights future efforts to expand the FFT library for future applications in Arduino. In the last chapter, we conclude on the efficiency of the use of microcontrollers due to its low cost and applications.

Chapter 2

Objectives

2.1 General Objective

- Implement an optical flow algorithm in an Arduino board that can measure the displacement of objects in videos.

2.2 Specific Objectives

- Recollect information about optical flow algorithms, such as phase correlation and results to determine implementation between Lucas-Kanade and Horn-Schunck.
- Implement the phase correlation algorithm to understand the operation of optical flow algorithms.
- Implement an algorithm to send images to the Arduino device by serial in real time.
- Carry out tests, analyze results, and provide feedback to obtain a quality microsensor.

Chapter 3

Theoretical Framework

3.1 Moving Objects

The technique of moving an object is carried out by means of distance and direction, previously indicated and determined, by means of a starting point which is followed by a second point of arrival. To describe the movement of an object, it first describes the position it occupies in a particular sector of space and at a specific time. A frame of reference is determined to proceed to determine its position. If an object is moved relative to a frame of reference, then the position of the object changes. It is precisely the variation of the position that is known as displacement. In this way, the displacement directly implies that an object changed its position within the frame of reference [10].

Moving object detection is the first step for the process of analyzing a video. This process is carried out in each study box or at the moment that the object of interest appears for the first time in the video. In addition, the elimination of objects located at the bottom of the object of interest in motion is proposed [11].

3.2 Optical Flow

The concept of optical flow is related to that of optical pattern and refers to the structure possessed by light at a particular point under observation in motion. This is defined as a pattern of movement of a particular object located in a specific position in time. This is caused by the apparent movement of an observer, for example as a camera. The main applications of the optical flow are based on the detection of movement, the segmentation of objects and the determination of the time until the collision of the elements involved. In general, it can be said that the optical flow is notorious at the moment when a point of the image has moved or moved from one place to another within it. [12].

In order to perform the mathematical calculations of the optical flow and have a mathematical representation that allows describing the included videos, the following expressions are available:

$$I(x, y, t_1) = I(x + v_x\Delta t, y + v_y\Delta t, t_2) \tag{3.1}$$

where:
$I\ (x, y, t)$ is the time
$t1$ and $t2 = t1 + \delta t$ are consecutive moments
$t1$ and $t2$ are the frames
(x, y) are the pixels
(Vx, Vy) is the velocity of position change

Assuming small displacements is obtained:

$$\frac{\partial I}{\partial x}\Delta x + \frac{\partial I}{\partial y}\Delta y + \frac{\partial I}{\partial t}\Delta t = 0 \tag{3.2}$$

If the intensity changes slightly (applying Taylor series) finally you get:

$$\frac{\partial I}{\partial x}v_x + \frac{\partial I}{\partial y}v_y + \frac{\partial I}{\partial t} = 0 \tag{3.3}$$

In addition to this, Taylor Series is an expansion of a function in a infinite series of one variable x. The function to expand it should have a derivative in the expansion interval [13]. It is an approximation of functions through a series of powers or sum of powers. An image can be defined as a two-dimensional function $f(xy)$ where x and y are the spatial coordinates, and the value of f at any coordinate pair (xy) is the intensity of the image at that point.

3.2.1 Optical Flow Sensor

Optical flow sensors have the particularity of incorporating into a single compartment an image compilation system and a digital processor designed to calculate the optical flow of the image to be studied. In this way, these types of sensors do not require any additional processor system, and can even work without any other external element for their control [14].

Objects that have a displacement in 3D surfaces infers a displacement in a plane of images (2D). This displacement in the plane can be described by changing the brightness patterns in the image in two consecutive frames of the video and thus describing an apparent speed [15]. The determination of the optical flow in the analysis of the video when calculating the velocity vectors gives the information of the change of spatial position of the analyzed object [16]. In this way, we know which object is displaced and the information is obtained of speed and direction.

The main objective for the use of an optical flow sensor is to measure the amount of movement or displacement in a plane. All this is done through the use of measurement techniques

based on image processing. Thus, a common optical flow sensor is mainly made up of a main sensor, which has a digital signal microprocessor and a commonly infrared light source which illuminates the surface. It also contains an integrated light conductor aligned to a pair of plastic lenses, which are intended to focus the captured image at a specific distance and to be able to concentrate the lighting produced by the light source in the area visible by the camera [14].

3.3 Phase Correlation

In general, when a digital processing mechanism is started, it is necessary to quantify the degree of interdependence between two processes or the similarity between two differentiated points respectively. In other words, determine the correlation between two images or signals. Thus, detailing the fields of application for this particular process highlights the detection and identification of signals. In this way, a measure of the correlation between the two images can be developed by means of making the sum of the products of the corresponding pairs of points by means of the expression known as cross correlation. A negative result indicates a negative correlation, that is, an increase in one variable is related to a decrease in the other [17].

The correlation is the guideline generally used for the detection of objects, because under certain restrictions the value of the correlation at the origin of an object with itself is greater than that of the correlation with any other object [17].

In the case of large images, it is advisable to perform this operation in the frequency domain using the correlation theorem as a starting point. Defining this domain as the values it can take and for which the function is defined. This allows to interpret its movement with respect to the frequency. This seeks to verify that the correlation operation is reduced to the multiplication of the Fourier transforms of each of the images to be compared [18].

3.3.1 Phase Correlation Algorithm

The intention with this algorithm [19] is to have a notion of how optical flow algorithms work after their implementation. This in turn works as an introduction to these sensors because it is easy to understand and implement. This algorithm can measure the displacement of objects in videos globally. All this is possible because the information related to the displacement of two images resides in the cross-power phase. In general, the cross-power phase is used as part of a frequency domain analysis of the cross-correlation between two time series and what the phase relationship is between them.

It is also related to other algorithms such as Lucas-Kanade and Horn-Schunck, which are approached in a more complex way, since they take into account other factors of greater complexity of analysis. In relation to this, the factors involved are related to the restriction of the constancy of brightness in the image to be analyzed. Also the difference that the Lucas-Kanade has the gradient restriction while the Horn-Schunck has the softness restriction.

3.3.2 Fourier Transformations

The methods based on the Fourier transform are very efficient since it is used to calculate the cross correlation between the two images. This mathematical method has the peculiarity of being widely used due to the flexibility of the Fourier series and transforms. In addition, these stand out for their resistance to noise and other typical defects that can be found in remote sensing images [20]. In this way the digital correlation calculation between two respective images is limited to calculating the Fourier transform. These are evident in the amazing variety of applications that they have in various branches of mathematics and mathematical physics, from number theory and geometry to quantum mechanics [21].

The mathematical expression of Fourier Transformations is:

$$\hat{f}(y) = \int f(x) exp(-2\pi ixy) dx \tag{3.4}$$

The expression 3.5 is known as the inverse Fourier function with respect to f:

$$\check{f}(x) = \int f(y) exp(2\pi ixy) dy \tag{3.5}$$

where:
$i = \sqrt{-1}$ is the representation for irrational numbers.
$exp(-i\pi yx) = cos(2\pi yx) - i \cdot sen(2\pi yx)$.
y is a variable that represents the frequency.

These two functions $f(x)$ and $f(y)$ are called a pair of Fourier transforms. The basic idea is to create the possibility of forming any function as a sum of a series of sine and cosine terms of increasing frequency.

3.4 Lucas-Kanade

The Lucas-Kanade method is generally used in the differential method to estimate the optical flow. This tool allows to solve the basic equations of optical flow for all pixels in that data set, by the least squares criterion. It is a method generally used because it does not allow to provide flow information inside uniform regions of the image. The main objective of using the Lucas-Kanade algorithm is to minimize the sum of the error squared between two test images [22].

This method has a significant number of advantages depending on its use, such as the ease of comparing with another method, very fast calculation and precise time derivations. Some of its disadvantages are reduced to errors in the boundaries of the moving object [22]. The Lucas-Kanade algorithm is a simple method which serves to provide an estimate of the movement of particular elements, in this case successive images.

This algorithm assigns a motion vector to each pixel in the scene, which was obtained by comparing the two consecutive images. The algorithm makes some assumptions such as that the images are separated by a small time interval, which is exemplified in that the algorithm works best with slow moving objects [23].

The algorithm also does not use color information directly. Nor does it seek to scan the second image looking for a match for a given pixel above. Contrary to the aforementioned, this works by trying to decipher the direction in which a particular object has moved and thus have the ability to explain the specific changes in the intensity of the study image [22].

This method can be represented in the following equation:

$$\mathbf{A^T W^2 A v = A^T W^2 b} \tag{3.6}$$

where:
W is a neighborhood weighting.
v is a vector of velocity.
A and b are matrices of partial derivatives.
T is the transposed of the matrix A.

It assumes that the flow is essentially constant in a local neighbourhood of pixels, and solves the basic optical flow equations for all the pixels in that neighbourhood. It does that by the least squares criterion. By combining information from several nearby pixels, the Lucas Kanade method can resolve the ambiguity of the optical flow equation.

3.5 Horn-Schunck

This method is widely used to determine the optical flow and to analyze the motion vector of the optical flow in each pixel of the selected image. In this way, try to minimize distortions in the flow and give specific solutions that show more smoothness in the default image. The proposal of this method consists in formulating the approach of the problem of the estimation of the optical flow as a variational problem, in which the desired vector field is finally expressed as a minimizer of a certain initial energy [24].

$$E = \int\int [(I_x u + I_y v + I_t)^2 + \alpha^2(||\nabla u||^2 + ||\nabla v||^2)]dxdy \tag{3.7}$$

where:
I_x, I_y, I_t are derived from the intensity values in the dimensions x and y and time.
u, v are the horizontal and vertical components of the flow field.
α is a regularization constant.
∇ is the gradient operator.
λ weights the regularization.

3.6 Motion Estimation

It is defined as the process by which the characteristic motion vectors of a particular image to be encoded with respect to one or more images are obtained. The general procedure that is performed by means of this type of motion estimation techniques is based on dividing the frame of the selected video sequence, which is nothing more than a set of images, into a matrix divided into a particular number of blocks to later look for this same block in one of the consecutive video frames [25].

3.7 Digital Image

Image is known as that as a two-dimensional function determined by a basic expression given by $f(x, y)$. For this expression x and y represent the coordinates of a plane that contains all the points of it. This function is recognized as the amplitude at the given point. All this is represented as the intensity or level of gray that makes up the image. Finally, if these established values are defined as discrete and finite, we are talking about a digital image [26].

A digital image is also formed by a finite number of elements to which a specific value and position in space belongs. These elements are known as elementary points of the image, or simply pixels. This is what is generally used as the minimum unit of measure for a digital image. Therefore, one of the most important parameters in a digital image is the resolution with which it is projected. The resolution is known as the number of pixels that make up a given image. The dimensions of an image with low resolution means being low in pixels, so you don't have many pixels to start extracting information. Nor would it allow comparison between images, limiting the size of images that can be displayed on a screen. Image processing techniques are mostly used to improve the visual appearance of images depending on the needs and expectations of the observer. It also works to prepare the photographic content prior to the perception by machines intended for these purposes [26].

3.7.1 Pixel

This is determined as the unit of measure for each image. This consists of a certain set of pixels. These are represented as the basic unit of analysis in an image. For this reason, some other element of smaller or finer dimension is not determined in an image that is not a pixel. Generally, this unit is related to color or intensity in which it is represented by a grid, each square in the grid contains only one particular pixel [27].

3.7.2 Image Processing

The image processing phase has as a priority to improve the quality of the image and allow the development of the various algorithms of the project and thus facilitate the search for information immersed in them. Generally the images used are generated in many ways as photographic

or electronically [28]. There are several types of image processing techniques like: Image preprocessing, Image enhancement, Image restoration, Image analysis, Image reconstruction or Image data compression.

Starting with image processing, it indicates that a particular type of data can be composed of different signal intensities. The preprocessing functions are based on the elimination of errors and detect irregularities of the sensor and atmospheric noise and then be completely processed. Then, another technique would be image analysis, with which information is extracted from an image by using scene analysis, image description, image compression, etc [29].

Another important technique to highlight within this set is image compression, since it allows reducing the size of digital images to save storage space and transmission time. Lossless encoding can be obtained, which allows to recover exactly the original quality of the image, being used to compress information that cannot be degraded. On the other hand, those methods with loss are especially suitable for images that can work with a minor loss of information and thus achieve a substantial reduction in bit rate [29].

3.8 Microcontroller

A microcontroller is a programmable integrated circuit, which has the ability to execute preestablished commands or actions through internal memory. This has several blocks that fulfill specific tasks. These are designed to reduce the economic cost and energy consumption that the system can execute. Returning to the above concepts, an Arduino Due is a microcontroller board [30]. It is an integrated circuit that contains a central processing unit (CPU), memory units (RAM and ROM), input and output ports. With these the user directs a certain amount of instructions that are recorded and executed at a certain moment. These parts are internally connected within the microcontroller, and together they form what is known as a microcomputer. With all these adjectives it is concluded that a microcontroller is a complete microcomputer, which is contained in an integrated circuit designed to perform specific functions [30].

The fundamental purpose of microcontrollers is to have the ability to read and carry out the programs that the user indicates previously. Thanks to the ease that microcontrollers can be programmable, they simplify the design of electronic circuits that can be used later in various applications. These in turn allow modularity and flexibility, because the same circuit can be used to perform different functions just by changing the initial microcontroller program.

These circuits are designed primarily to reduce the economic cost and energy consumption of a particular system. That is why the size of the processing unit and the amount of memory depend on the application you want to develop. To cite a simple and everyday example of its usefulness, controlling a simple appliance such as a mixer will use a very small processor and depending on another range of appliances will require other specifications, both memory and processing, for proper use and maintenance [31].

3.8.1 Arduino Due

The Arduino Due is an electronic board based on a 32-bit ARM core microcontroller. It also has 54 digital input and output pins, 12 analog inputs, 4 UARTs (serial hardware ports), an 84 MHz clock, a USB OTG capable connection, 2 DAC (digital to analog), 2 TWI, a connector Power supply, an SPI connector, a JTAG connector, a reset button and a complete erase button[32].

Compared to other Arduino boards generally used, the Arduino Due works and operates at 3.3 volts. In this way, the maximum voltage to which the board pins can be subjected is 3.3 volts. The application of voltages higher than 3.3 volts to any pin on the board and under any circumstances could damage the board in certain situations. In addition, the board has all the minimum and indispensable elements to feed the operation of the microcontroller. To achieve its operation, it is sufficient to connect it to a computer by means of a micro-USB cable or power it by means of a primary battery. The Due is compatible with all the different Arduino board models that work exclusively at 3.3V and comply with the 1.0 Arduino pinout standard[32].

3.8.2 Electrical Power Supply

As detailed above, the plate corresponding to the Arduino Due can be powered by means of the USB connector or alternatively connecting it to an external power supply, this being the most used. External power, which is not USB type, can come with an AC to DC adapter or a battery. The adapter can be connected to a plug to the power port of the board. Cables from a battery can be inserted into the GND and Vin connectors of the power connector. Thus, the card has the ability to operate with an external power supply of approximately 6 to 20 volts. If more than 12 V is used, the voltage regulator may overheat and damage the board. The recommended range for optimal operation is 7 to 12 volts [33].

Chapter 4

Related Works

This chapter studies and analyzes various works related to the objectives of this particular research work. Common characteristics such as image processing and the application of optical flow sensors are determined. It is sought that each reference presented includes the central theme, the methodology applied, the results obtained, and if possible the suggested recommendations. With this is possible to create a solid contrast between the approach of ideas that contribute to the fulfillment of the objectives of this work.

4.1 Ravindra. S (2010)

The interest in digital image processing methods as Ravindra. S puts it [34] comes from two main areas of application: image enhancement information for human interpretation; and image data processing for storage, transmission and representation for the perception of autonomous machines. The objectives of this article is to define the meaning and scope of image processing, discuss the various steps and methodologies involved in a typical image processing and applications of image processing tools and processes at the border research areas.

It also focuses on the steps to improve digital images, which is a point of interest for this work. The next step is image improvement, which is one among the simplest and most attractive areas of digital image processing. Basically, the main idea behind the improvement techniques is to highlight details that are obscured, or simply to highlight certain features of interest in an image.

4.2 Chan. R et al (2010)

Chan. R, Mulla. A, and Stol. S [35] carry out a study about the development and characterization of a low-cost optical sensor module which would be primarily intended for the location of specific objects. Sensory data has been characterized based on the distance of the objects and the quality of the image. This paper initially proposes a mathematical model that allows the transformation of sensory data into adequate data for sensor fusion. As a result of executing this, it is shown that the transformation successfully eliminates error bias. This model also allows estimating the variance of residual errors. This in turn allows data from lower image

quality to remain useful for estimation. This causes the useful range of the distances of the optical sensor module to be extended and thus allows a more accurate speed estimation.

They also propose the use of various imaging devices to implement optical flow techniques. They suggest that recently those most predominant optical flow sensors were standard cameras. These cameras usually provide high resolution images, up to 30 frames per second. As an alternative to low cost, CMOS integrated circuits are determined, those generally used in computer mice. The factors that motivated researchers to opt for this option were first its low price and in the background the relatively low energy consumption which helps with the performance of the device.

The current technologies of CMOS sensors for the realization of the calculation of optical flow, have the particularity of performing the two main activities: correspondence of the image in consecutive frames and calculate the displacement by sequence. These factors include sufficient illumination, constant distance to the image surface and slower movement than the processing of these CMOS sensors. Despite this, if a controlled environment of these factors is maintained, the range and accuracy of the sensors can become a feasible and economical method for localization in many motion applications. This translates into its use in unmanned vehicles, robot limbs, flight control, medical images, obstacle avoidance and manufacturing in general, to name a few of its applications.

4.3 Xiaoming. L et al (2016)

In this work the authors [36] make special reference to the importance of Optical Flow Sensors (OFS). They clarify that these have special features such as their small size, light weight, low cost and low energy consumption. Additionally, as an innovation in their work, they establish that they are suitable for antennas in miniature vehicles (MAVs) to be able to detect the environment around them and make conjectures about what they want to know and infer from it. According to recent work, this article proposes the creation of an innovative model that mainly seeks to estimate the flight states of an MAV equipped with optical flow sensors (OFS) and speed gyroscopes. Although the most common navigation sensor used in UAVs is GPS, they make it clear that they do not want to make use of traditional GPS, radars or anemoscopes.

They also place special emphasis on the existence of numerous applications for unmanned aerial vehicle (UAVs) antennas. One of the main applications of these antennas lies in the field of civil, commercial and especially in the military field. On the other hand, they emphasize the miniature aerial vehicles (MAVs), which may vary in size according to their functionality. They play an important role in military applications, the most important fields of application being those related to general recognition , surveillance, battle damage assessment and communications networks.

4.4 Basha. S et al (2018)

The estimation of movement, as stated by Basha. S and Kannan. M [37], is one of the most intense computational operations in terms of video compression techniques. They propose the

use of the Block Matching algorithm which consists in the elimination of temporary redundancies among a significant number of successive photographs of each other. In turn, this technique has become more important in most video compression and coding standards based on motion compensation.

The article emphasizes that compression performance can be affected and drastically increased through the use of efficient motion estimation techniques. All this occurs by reducing energy within the residual frames involved in motion compensation. The comparison is made between the existing block matching algorithms and their limitations in the estimation of movement along with their applications in various fields of interest.

Also, authors refer to the different techniques for motion estimation. They propose one of the most popular techniques such as the block matching algorithm (BMA). In general, they define their operation as the blocking of pixels in the current frame and are compared with the corresponding block of pixels in the reference frame within the search area. Finally, the values of MAD (mean absolute difference) and the best matched block having less average absolute difference are calculated to generate more optimal results.

Comparisons

In relation to the related works, a comparative chart (Table 4.1) is prepared to show the contributions of various authors based on the central theme of the work. All this in order to synthesize the content and make it more accessible to the reader.Important aspects are highlighted as their contribution, year of publication, authors and general topic.

#N.	Autors	Age	Topic	Contribution
1	Ravindra, S.	2010	Digital image processing	Image data processing for storage, transmission and representation for the perception of autonomous machines.
2	Chan, S. et al.	2010	Low cost optical flow sensor	Development and characterization of a low-cost optical sensor module, which will be used for localization.
3	Xiaoming, L. et al.	2016	Optical flow sensor	Scheme of placement of optical flow sensor in a miniature vehicle,
4	Basha, S. et al.	2018	Motion estimation	Analysis on motion estimation techniques

Table 4.1: Comparative chart of related works.

Chapter 5

Methodology

This chapter contains the explanation of the mechanisms used for the analysis of the research problem (Figure 5.1). In general, certain programming languages and digital development tools were used. The purpose of using all these tools was the compatibility they had with Arduino Duo. It also had available work libraries that facilitate working with matrices.

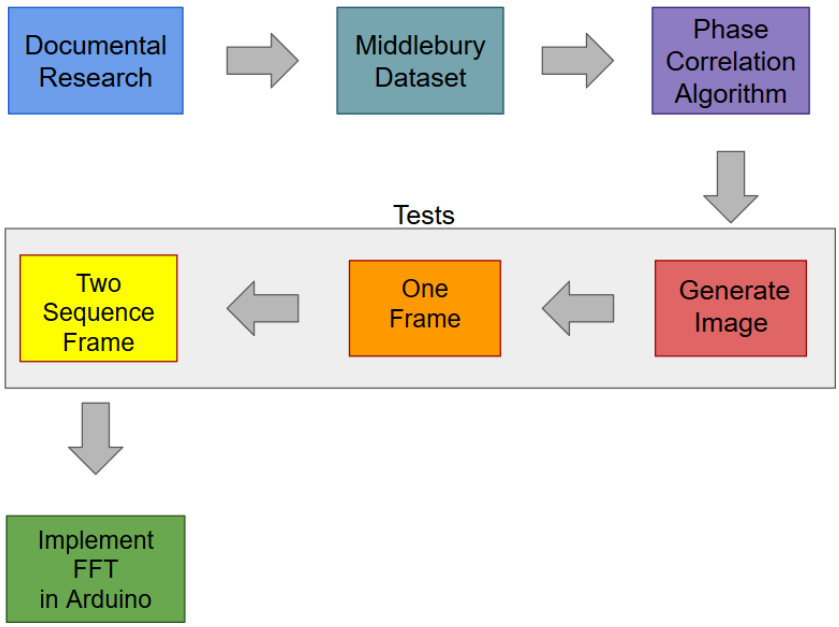

Figure 5.1: Methodology diagram.

In the documentary method, as much information as possible was collected in scientific articles and a book on optical flow algorithm. The results, on which has better performance for what you want to perform optimally and with quality. At the same time, we looked for data (videos) which were used as an example to carry out the project. The results were expected to obtain the best performance for what you want to perform optimally and with quality. At the same time, we searched for data and that were used as an example to carry out the project.

The prototype-incremental method that the sensor is implementing by implementing it little by little and seeing if it works or not. In this way, it allows evaluating in an early way how the development of the microcontroller is going and if it is doing what it should. Step by step its functionality grows. By combining it with the incremental one, it allows to control the complexity of the optical flow methods that were implemented [38].

This section defines the various stages by which the algorithm was executed. This step began by simulating two consecutive frames in order to observe the displacement of objects contained in these frames and which were of our interest. According to the theory, a video is a set of frames sequentially, which is the same as the set of sequential images into which a movie is divided. In turn, the resolution of the image is related to the number of pixels per unit area, which translates into the details of the image [39].

Behind the elaboration of the phase correlation technique, the parameters by which this work has been guided are established, as well as its implementation in the image process. This is closely related and very important in records based on Fourier transformations. To achieve this, the algorithm is based on the idea that two points contain sufficient information about its displacement. Therefore, a general analysis is made of the points or data from each captured image. This seeks to track them and obtain the optical flow. The change between these translated signals could be solved by the following equations:

$$S\left(u,v,k\right) = \frac{F1(u,v,k)}{\left|F1(u,v,k)\right|} \bullet \frac{F2(u,v,k)*}{\left|F2(u,v,k)\right|} \tag{5.1}$$

$$s(x,y,z) = F^{-1}\{S(u,v,k)\} \tag{5.2}$$

$$(Xp,Yp,Zp) = \arg_{(x,y,z)} \max s(x,y,z) \tag{5.3}$$

where the expressions given by $F1(u,v,k)$ and $F2(u,v,k)$ two shifted signals, and $F1(u,v,k)$, while $F2(u,v,k)$ is their spectra of Fourier respectively. We also define (Xp,Yp,Zp) as the displacement between two signals and $F2(u,v,k)*$ as the conjugate.

5.1 Arduino

Arduino is mainly defined as an open source electronics creation platform or environment, therefore it is based on the development of free hardware and software. This platform allows you to create different types of microcomputers on a single board to which each developer can give them different types of use. In this particular case, it was useful to determine the displacement of objects and measure interactions between both images [32].

5.2 Data Set in Images

Since the objective is to see the displacement of objects in videos, it is necessary to have a large set of videos, although the best option is to have consecutive frames since it is known that a video is composed of a set of frames. For this, it is important to have a database containing these frames.For this work it was decided to use a repository called Middlebury [1]. It is support by Middlebury College, Microsoft Research, and the National Science Foundation, and develop by Daniel Scharstein and Richard Szeliski. This has a section for evaluation of optical flow algorithms with many consecutive frames of different color or black and white videos. It is also separated by sets of two consecutive frames of different videos. It was decided to use an example of two consecutive frames since this allows to demonstrate the phase correlation algorithm. These frames will be read through OpenCV [40] and will become an information matrix which will be used in the algorithm.

The dataset extracted from Middlebury [1] is the evaluation set of two consecutive grayscale frames. This section contains twelve different samples from two consecutive frames. Of which three of those samples (Teddy, Urban and Grove) were taken for testing. All these images are in png format and have a variable resolution between 360 * 420 pixels up to 480 * 640 pixels.

Within the set of data used, it should be noted that random images were generated and applied translation movement to compare the algorithm's performance. An image of the consecutive frames was selected and translation movement was applied to check if the movement applied manually was the same that the algorithm detected. With the two consecutive squares it was proved when it was the displacement from one to the other.

5.3 Informatics tools

The different informatics tools used during the project are listed below.

5.3.1 Visual Studio Community 2017

Visual Studio is known as an integrated development environment (IDE) that Microsoft developed primarily for Windows systems in order to provide users with a much more complete application development tool than others. Because it is a multiplatform tool, this environment works to program the C ++ language and in turn is compatible with Arduino boards. All this through Visual Micro. For these reasons it was the environment decided to run the program. This is a complement to the creation of efficient and high-performance applications, thus allowing the creation of web sites and applications, as well as other services in any of the environments that support the platform according to its disposition. It also has the possibility of extending the sharing of tools and thus facilitating the creation of solutions in several programming languages such as C.

5.3.2 C++

This is the language intended for the elaboration of the components. C ++ is a programming language designed primarily to be able to extend to the existing C programming language with mechanisms that allow the manipulation of objects. The language is compatible with Arduino Duo because the board intended for use belongs to this range of boards. In this language there are libraries that facilitate the management of matrices and the use of the Fourier transform.

Two libraries of great importance to execute the codes stand out. One of them is the **FFTW3** [41] library which allows the handling of Fourier transforms. On the other hand, the **Eigen** [42] library allows the handling of matrices in a simpler way.

5.3.3 OpenCV

This is handled as a free C ++ library [40]. A large number of applications are available. Among those that stand out the most are the utility to create security systems with motion detection. It also allows developing control applications where object recognition is required. For this reason they have established methods for image management, which was the purpose of their use in this project. At the same time it is a cross platform so it is compatible with our operating system Windows 10 [43].

5.3.4 Processing IDE

It is a programming language based on Java, so it allows you to develop applications or projects in this language. It has the ability to communicate with the Arduino through connections to the serial port. In turn, this will be used to send relevant information through the serial port to the Arduino, achieving communication with microcontrollers [44].

Chapter 6

Results

This chapter aims to expose and describe the data obtained in the investigation. Subsequently results were interpreted and contrasted with the theory. These were directly related to the general and specific objectives for the investigation. Thus, the first relevant result was the determination of the implementation of the Lucas-Kanade algorithm based on the research carried out, since it allows better performance in terms of time and this is vital to implement in the microcontroller [2].

A compilation of information on the Lucas-Kanade and Horn-Schunck algorithms, respectively, is proposed. According to the theory consulted throughout the project and the various comparisons obtained according to the research, it is considered that the most appropriate algorithm is Lucas-Kanade. The various studies support this idea in the speed at which the program is able to solve the equations included in the process [2] [45] [46].

Another important point where they coincide is the structure of the algorithm. The Lucas-Kanade algorithm has the advantage and characteristic of having the direct resolution of a matrix equation, which is derived from the brightness constancy equation. All these points imply in terms of program performance a relatively lower computational cost if compared to the performance and results obtained from the Horn-Schunck algorithm [2]. With all this we assume that the main advantage of using the Lucas-Kanade method is given by the ease it gives when assuming that the calculation of the optical flow is constant.

The realization of a phase correlation algorithm capable of interpreting the data obtained by the optical flow algorithm is proposed. During this stage, the results of the information search and implementation of the algorithm are presented, as well as desktop tests on the operation of the algorithm about how it works on the computer and the application of this in Arduino board.

The results, once the 2D phase correlation code was implemented:

6.1 Generate Figure

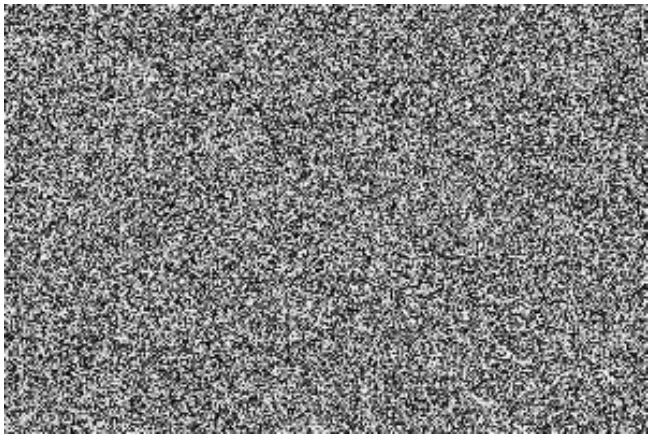

Figure 6.1: A figure generated with a random value of pixel.

The algorithm used for these tests was the phase correlation algorithm. The first time the algorithm was tested in 2D, it was to generate the image (Figure 6.1) with a random value for each pixel. Then, each pixel of the image was displaced an **x** (right) and a **y** (down) number of times, this being the second image used as a scene. By including both images in the program, the algorithm returns the amount of **x** and **y** displaced from the pixels. It should be noted that the code recognizes the offset of the second image, and that the displaced image is taken as if it were the sequential image.

Below you can see in the Table 6.1, the same test performed with three different samples. Where **x** and **y** is the offset value of each pixel pixel and size is the resolution of the image.

#N. of sample	**Size**	**x**	**y**
1	(200,300)	-125	25
2	(360,420)	25	-15
3	(560,560)	-15	-10

Table 6.1: Test one results summary.

6.2 One Frame

Figure 6.2: One frame of video sequence "Teddy" taken from database [1].

Figure 6.3: Frame 6.2 translation.

For a second test, it was proceeded to take one frame (Figure 6.2) from the database, and then this frame proceeded to each pixel was displaced an $\mathbf{x}$ (right) and a $\mathbf{y}$ (down) number of times, this being the second image used as a scene (Figure 6.3). This is done in order to corroborate the first test. The result of the displacement obtained by the algorithm was the same amount of pixels displacement used for this test. This corroborates the operation of the algorithm as in the first test and shows in a visual way how the displacement of the two test presented was generated. For this test we use a d Middlebury dataset [1]. The first sample is "teddy" dataset, the second is "urban" dataset, and the last is "grove" dataset.

The results of this test are shown below in the Table 6.2, with the same parameters as the previous test

#N. of sample	Size	x	y
1	(360,420)	-150	100
2	(480,640)	-57	33
3	(480,640)	-32	-28

Table 6.2: Test two results summary.

6.3 Two sequence Frame

Figure 6.4: A first sequential frame [1].

Figure 6.5: A next sequential frame to figure 6.4.

For a third test, it was taken two consecutive frames (Figure 6.4 and Figure 6.5) in which is difficult to see the displacement and be able to analyze on whether there was any variation in these frames with a simple view. The algorithm show that there are a displacement between the first and the second frame of the image.

This shows that exists a displacement in frames which is the objective of the test. Figures 6.4 and 6.5 correspond to the first sample of the Table 6.3 are taken from the same data set of the previous test. This means that the first sample is the two consecutive frames of "teddy" dataset, the second of "urban" dataset, and the last of "grove" dataset.

The results of this test are shown below in the Table 6.3, with the same parameters as the previous test

#N. of sample	**Size**	**x**	**y**
1	(360,420)	-11	0
2	(480,640)	7	-2
3	(480,640)	-3	-1

Table 6.3: Test tree results summary.

All test results are summarized in Table 6.4, being the parameters **x** and **y** the displacement of scene in terms of pixels, and the parameter **size** is resolution of image.

#N. of test	#N. of sample	Size	x	y
1	1	(200,300)	-125	25
	2	(360,420)	25	-15
	3	(560,560)	-15	-10
2	1	(360,420)	-150	100
	2	(480,640)	-57	33
	3	(480,640)	-32	-28
3	1	(360,420)	-11	0
	2	(480,640)	7	-2
	3	(480,640)	-3	-1

Table 6.4: Test results summary.

6.4 Arduino Due

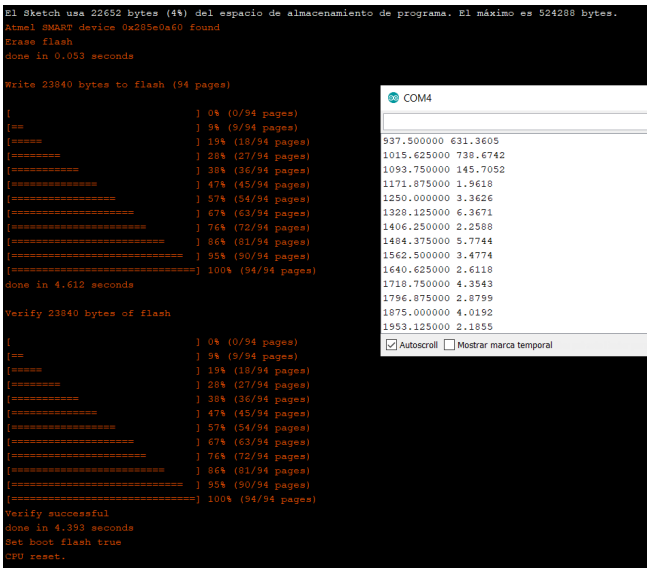

Figure 6.6: Compilation and result in arduino.

In this part, you should use the phase correlation algorithm previously used in the performed desktop tests and proving that it works. At this point it is important to remember that the previous tests were carried out thanks to the help of the C ++ library, FFTW3 [41]. At the time of wanting to use this library on arduino, the problem arose that it is not fully compatible with Arduino. I was looking a library that can perform the FFT on Arduino, which is called Arduno FFT [47]. This did not have all the methods used for the algorithm, so it did not serve as it should. We proceeded to test an example of this library, "FFT_01" with modifications (Appendix A), to also test another library used in the phase correlation algorithm. In the Figure 6.6, you can see that it loads correctly on the Arduino and runs giving results. This shows that Arduino was capable of FFT which is an important step since that implies that Arduino would be able to perform the phase correlation algorithm. But to carry out this algorithm, the Arduino FFT library must be developed further and it must be able to perform the methods of the FFTW3 library.

Chapter 7

Discussion

This chapter will be based on analyzing and comparing these results to obtain a general interpretation of them and gain a better understanding of the objectives.

As the result shows in Table 6.4, the phase correlation can be done both in generated images and in consecutive frames. There were no special restrictions to execute this procedure. For this reason, It is determinate that the phase correlation can effectively evaluate the displacement on a video, and it is the reason why this method is used.

This consideration is taken under the premise that a video has sequential frames. It was established at the beginning that this test uses two sequential frames, which is why this premise is assumed. This means that if an object is displaced from one image to another, the algorithm establishes favorably when the object moved in terms of x and y pixels. In relation to the amount of pixels, in smaller files and with fewer pixels, they are easier and faster to process in the future. In addition, these files can be handled more easily, since lower resolution images do not need to be resized. Therefore, if you need to process them quickly you can do it easily.

Therefore, phase correlation can be made on a video, and it would have the ability to determine whether there is object displacement or not on this video. For the interpretation of the following results, it is established that the 2D phase correlation can be carried out in Arduino Duo. This is because it is capable of performing the FFT (Figure 6.6) and running it through the Arduino FFT library [47]. This technique is commonly used in image registration and has been applied to estimate disparity. In addition, this helped identify the magnitude and direction of some change in the position of an object within an image.

On the other hand, the 2D phase correlation that emphasizes the entire image needs to continue developing and expanding the entire FFT Arduino library [47]. This with the purpose of finishing adapting the FFTW3 library [41] and making it possible for Arduino to perform the phase correlation in two dimensions. This implementation will allow the development of the library, which can be approached as a central theme for future investigations of a more capable library with different scopes. This would result in a technique of low computational cost and with better results.

All this with the purpose of eventually create dedicated instruments that can detect and

report traffic jams, unusual and dangerous human crowd behaviors and the effects of natural disasters like fires, earthquakes and tsunamis.

Chapter 8

Conclusions and Future Work

In this thesis we have studied the measurement of the displacement of objects in a sequence of images (videos). Also, we have implemented the phase correlation algorithm. In addition, we have studied existing optical flow calculation methods such as the Lukas-Kanade algorithm in order to evaluate which algorithm is best for measuring the displacement of objects in a microcontroller. This chapter summarizes the research and findings reported in this work. The main conclusions are highlighted and the chapter ends suggesting possible directions for future work.

8.1 Conclusions

Throughout this thesis, four main conclusions have been made:

1. Based on the bibliographic investigation of the optical flow algorithms, a comparison is made with the Lukas-Kanade algorithm. Accordingly, the Lukas-Kanade algorithm can produce faster results, it can be said that so far it is the most efficient method to run on a microcontroller.

2. It is determined that the microcontrollers are widely used thanks to their functionality and can execute the code related to the optical flow. It also takes into account the use of low cost and easily interconnectable microcontrollers, such as the Arduino Due. All this due to its programming and performance environment, is a viable element to develop more optical flow applications.

3. The method used to develop this work is established as a phase correlation algorithm, which was launched on a computer. The performance of the algorithm was measured using the set of images used by the Middlebury library.

4. We also perform Fourier transform in Arduino board like sample to that Fourier Transform can be done, which is the previous step of Phase Correlation algorithm.

8.2 Future Directions for Research

The research presented in this thesis focuses on the implementation of an object displacement sensor in videos through an optical flow algorithm. However, there is much scope for future

work to be done in this direction. Some of those are:

1. Implementing the entire phase correlation algorithm on an Arduino board remains a challenging projection for future work. All due to the lack of implementation of certain libraries for professional operation, despite this, it is presented as the most efficient option.

2. Combine microcontrollers with tools based on artificial intelligence and expand their use in activities such as fire control, natural disasters, people flow control, traffic control in large spaces, etc. Its use is simple and allows many applications in the design of dynamic Models.

3. Expand the existing Arduino FFT library. This with the main objective of expanding its capabilities and doing the same processes as other existing libraries such as FFTW3. In addition to this, you will have the ability to perform phase correlation on the Arduino board. These libraries will contain the specifications of different functionalities for the application of specific conditions of optical flow.

Bibliography

[1] Middlebury, *Optical Flow.* Middlebury Web site, 2018. [Online]. Available: http://vision.middlebury.edu/flow/data/

[2] R. Pazmiño, *Implementación y comparación de los algoritmos de determinación de flujo óptico de Horn-Schunck y LucasKanade para rastreo de objetos.* Universidad San Francisco de Quito, 2016.

[3] S. Hossain, S. Rafi, M. Rokonuzzaman, and M. Wahed, *An Enhanced Tsunami Detection System.* International Journal of Innovation and Scientific Research, 2016.

[4] A. Alkhatib, *A Review on Forest Fire Detection Techniques.* International Journal of Distributed Sensor Networks, 2014.

[5] L. Martins, P. Silva, and M. Gattass, *An Application of Optical Flow to Time-Lapse Analysis in 2-D Seismic Images.* 11th International Congress of the Brazilian Geophysical Society and EXPOGEF, 2009.

[6] Z. Zou, Z. Shi, Y. Guo, and J. Ye, *Object Detection in 20 Years: A Survey.* arXiv: Computer Vision and Pattern Recognition, 2019.

[7] A. Agarwal, S. Gupta, and D. Singh, *Review of optical flow technique for moving object detection.* 2nd International Conference on Contemporary Computing and Informatics (IC3I), 2016.

[8] H. Gani, O. Khalifa, T. Gunawan, and E. Shamsan, *Traffic Intensity Monitoring using Multiple Object Detection with Traffic Surveillance Cameras.* IEEE 4th International Conference on Smart Instrumentation, Measurement and Application (ICSIMA), 2017.

[9] J. Walsh, N. O'Mahony, S. Campbell, A. Carvalho, L. Krpalkova, I. Velasco, D. Riordan, and S. Harapanahalli, *Deep Learning vs. Traditional Computer Vision.* Computer Vision Conference (CVC), 2019.

[10] M. Prieto, M. Marufo, L. Di Matteo, A. Verrastro, R.and Hernández, J. Gomez, and C. Verrastro, *Algoritmo de seguimiento de objetos en imágenes mediante reconstrucción iterativa de histograma en tiempo real.* Jornadas Argentinas de Robótica, 2014.

[11] S. Balaji, *A survey on moving object tracking using image processing.* 11 th International Conference on Intelligent Systems and Control (ISCO), 2017.

[12] A. Alvarez and R. Caballero, *Detección de Características de Imagen y Flujo Óptico en Espacios Confinados Acuáticos*. 15th LACCEI International Multi-Conference for Engineering, Education, and Technology: "Global Partnerships for Development and Engineering Education", 2017.

[13] J. Fosso-Tande, *Applications of Taylor series*, 2013.

[14] M. Ribes, *Application of optical flow sensors for the development of new low cost measurement systems*. Departamento de Informática e Ingeniería Industrial Universitat de Lleida.

[15] C. Kuglin, *The Phase Correlation Image Alignment Method*. IEEE Int. Conf. on Cybernetics and Society, 1975.

[16] G. Peng, *An improvement of image registration based on phase correlation*. Optik - International Journal for Light and Electron Optics, 2014.

[17] M. Artigao, E. Rubio, S. Díez, and C. V., *Técnicas de correlación cruzada en el registrado de imágenes de diferente resolución espacial*. XI Congreso Nacional de Teledetección, 2005.

[18] J. Goodman, *Introduction to Fourieroptics*, 1968.

[19] B. S. Redd, *An FFT-based technique for translation, rotation, and scale-invariant image registration*. IEEE Trans. Image Process., 1996.

[20] A. Ordonez, F. Arguello, and D. Heras, *Fourier transform applied to the alignment of multidimensional images in GPU*, 2017.

[21] G. Gonzales, *Fourier series, Fourier Transforms and Applications*. Divulgaciones Matemáticas, 1997.

[22] Z. Zhariy, *Introduction to Optical Flow*, 2005.

[23] B. Lucas and T. Kanade, *An Iterative Image Registration Technique with an Application to Stereo Vision*, 1981.

[24] E. Meinhardt-Llopis, J. Sanchez, and D. Kondermann, *Horn–Schunck Optical Flow with a Multi-Scale Strategy*, 2019.

[25] A. Barjatya, *Block Matching Algorithms For Motion Estimation*. IEEE Transactions Evolution Computation, 2004.

[26] S. Satadal, B. Subhadip, and N. Mita, *A Comprehensive Survey on Different Techniques and Applications of Digital Image Processing*. Proc. of Int. Conf. on Computing, Communication and Manufacturing 2014. ECE Dept., MCKV Institute of Engg., Liluah, 2014.

[27] A. Van den Oord, N. Kalchbrenner, and K. Kavukcuoglu, *Pixel Recurrent Neural Networks*. 33rd International Conference on Machine Learning, 2016.

[28] R. Saroha, R. Bala, and S. Siwac, *Review paper on Overview of Image Processing and Image Segmentation.* International Journal of Research in Computer Applications and Robotics, 2013.

[29] P. Ravi and A. Ashokkumar, *Analysis of Various Image Processing Techniques.* International Journal of Advanced Networking & Applications (IJANA), 2017.

[30] Y. Guven, E. Cosgun, S. Kocaoglu, H. Gezici, and E. Yilmazlar, *Understanding the Concept of Microcontroller Based Systems To Choose The Best Hardware For Applications.* International Journal of Engineering And Science, 2017.

[31] L. Ganan and V. Quintana, *Introduccion a los microcontroladores PIC y programacion de una matriz de LED's.* Centro de Diseño e Innovación Tecnológica Industrial Regional Risaralda, 2009.

[32] L. Louis, *Working principle of arduino and using IT as a tool for study and research.* International Journal of Control, Automation, Communication and Systems (IJCACS), 2016. [Online]. Available: https://airccse.com/ijcacs/papers/1216ijcacs03.pdf

[33] Arduino, *Arduino Due.* Arduino Project web site, 2018. [Online]. Available: https://www.arduino.cc/en/Main/ArduinoBoardDue/

[34] S. Ravindra, *Image Processing: Research Opportunities and Challenges.* National Seminar on Research in Computers, 2010.

[35] R. Chan, A. Mulla, and K. Stol, *Characterisation of Low-cost Optical Flow Sensors.* Proceeding of the Australasian Conference on Robotics and Automation, 2010.

[36] L. Xiaoming, C. Zhongyuan, C. Wanchun, and X. Xiaolan, *Motion estimation using optical flow sensors and rate gyros.* IEEE International Conference on Mechatronics and Automation, 2016.

[37] S. Basha and M. Kannan, *Literature survey on motion estimation techniques.* International Journal of Engineering and Technology, 2018.

[38] R. Janeth, *Metodología de Desarrollo de Software: MBM (Metodologia Basada en Modelos).* INGENIARE, Universidad Libre-Barranquilla, 2009.

[39] F. Brandi, R. De Queiroz, and D. Mukherjee, *Super-resolution of video using key frames.* IEEE International Symposium on Circuits and Systems (ISCAS), 2008.

[40] OpenCV, *OpenCV.* OpenCV web site, 2020. [Online]. Available: https://opencv.org/about/

[41] FFTW, *Features.* FFTW web site. [Online]. Available: http://www.fftw.org/

[42] Eigen, *Eigen.* Eigen web site, 2019. [Online]. Available: http://eigen.tuxfamily.org/index.php?title=Main_Page

[43] M. 2020, *Windows 10.* Microsoft web site, 2020. [Online]. Available: https://www.microsoft.com/es-es/windows/features

[44] Processing, *Processing Overview.* Processing Web site, 2001. [Online]. Available: https://processing.org/overview/

[45] A. Yassine, C. Guillaume, M. El Mustapha, and C. Fatima, *Adaptive Lucas-Kanade tracking.* mage and Vision Computing, 2019.

[46] B. Duvenhage, J. Delport, and J. De Villiers, *Implementation of the Lucas-Kanade image registration algorithm on a GPU for 3D computational platform stabilisation.* 7th International Conference on Computer Graphics, Virtual Reality, Visualisation and Interaction, 2010.

[47] E. Condes, *Arduino Library List.* Arduino Library List, 2020. [Online]. Available: https://www.arduinolibraries.info/libraries/arduino-fft

Appendices

Appendix A

Arduino Code

Below is the Arduino code in which the FFT example with modifications is performed.

Code

```
#include <Eigen30.h>
#include <EigenAVR.h>
#include "arduinoFFT.h"

// Calls main Eigen matrix class library
// Calls inverse, determinant, LU decomp., etc.
using namespace Eigen;   // Eigen related statement; simplifies syntax for declaration of
    matrices

arduinoFFT FFT = arduinoFFT(); /* Create FFT object */

//These values can be changed in order to evaluate the functions

const uint16_t samples = 64; //This value MUST ALWAYS be a power of 2
const double signalFrequency = 1000;
const double samplingFrequency = 5000;
const uint8_t amplitude = 100;
//These are the input and output vectors
//Input vectors receive computed results from FFT

double vReal[samples];
double vImag[samples];

#define SCL_INDEX 0x00
#define SCL_TIME 0x01
#define SCL_FREQUENCY 0x02
#define SCL_PLOT 0x03

void print_mtxf(const Eigen::MatrixXf& K);
```

```
void setup() {

    Serial.begin(9600);

    Eigen::Vector2i image_size;
     image_size << 200,300;
     EigenMatrixRowXf object_img = EigenMatrixRowXf::Zero(image_size(0), image_size(1));

     for (int i=0; i<image_size(0); i++)
     {
         for (int j=0; j<image_size(1); j++)
         {
             object_img(i,j) = std::rand()%255;
         }
     }

     // Generate scene_img by
     // shifting object_image to DOWN by translation(0)
     // shifting object_image to RIGHT by translation(1)
     Eigen::Vector2i translation;
     translation << -25, 125;
     EigenMatrixRowXf scene_img = EigenMatrixRowXf::Zero(image_size(0), image_size(1));
     int x_index, y_index;
     for (int i=0; i<image_size(0); i++)
     {
         for (int j=0; j<image_size(1); j++)
         {
             if (i-translation(0)<0)
                 x_index = i-translation(0)+image_size(0);
             else if (i-translation(0)>=image_size(0))
                 x_index = i-translation(0)-image_size(0);
             else
                 x_index = i-translation(0);

             if (j-translation(1)<0)
                 y_index = j-translation(1)+image_size(1);
             else if (j-translation(1)>=image_size(1))
                 y_index = j-translation(1)-image_size(1);
             else
                 y_index = j-translation(1);

             scene_img(i,j) = object_img(x_index, y_index);
         }
     }
```

```
    // DECLARE MATRICES
    //--------------------
    MatrixXf Pp(6,6);   // Produces 6x6 float matrix class


    // INPUT MATRICES (so-called comma-initialize syntax)
    //-------------------------------------------------------------------
    Pp << 0.3252, 0.3192,   1.0933, -0.0068, -1.0891, -1.4916,
          -0.7549,  0.3129,   1.1093,   1.5326,   0.0326, -0.7423,
           1.3703, -0.8649, -0.8637, -0.7697,  0.5525, -1.0616,
          -1.7115, -0.0301,  0.0774,   0.3714,   1.1006,   2.3505,
          -0.1022, -0.1649, -1.2141, -0.2256,  1.5442, -0.6156,
          -0.2414,  0.6277, -1.1135,  1.1174,   0.0859,   0.7481 ;



    // Print Result
    //-------------------------
     print_mtxf(Pp);        // Print Matrix Result (passed by reference)

}
```

```
void loop()
{
  /* Build raw data */
  double cycles = (((samples-1) * signalFrequency) / samplingFrequency); //Number of
      signal cycles that the sampling will read
  for (uint16_t i = 0; i < samples; i++)
  {
    vReal[i] = int8_t((amplitude * (sin((i * (twoPi * cycles)) / samples))) / 2.0);/*
        Build data with positive and negative values*/
    //vReal[i] = uint8_t((amplitude * (sin((i * (twoPi * cycles)) / samples) + 1.0)) /
        2.0);/* Build data displaced on the Y axis to include only positive values*/
    vImag[i] = 0.0; //Imaginary part must be zeroed in case of looping to avoid wrong
        calculations and overflows
  }
  /* Print the results of the simulated sampling according to time */
  Serial.println("Data:");
  PrintVector(vReal, samples, SCL_TIME);
  FFT.Windowing(vReal, samples, FFT_WIN_TYP_HAMMING, FFT_FORWARD); /*
      Weigh data */
  Serial.println("Weighed data:");
  PrintVector(vReal, samples, SCL_TIME);
  FFT.Compute(vReal, vImag, samples, FFT_FORWARD); /* Compute FFT */
  Serial.println("Computed Real values:");
  PrintVector(vReal, samples, SCL_INDEX);
  Serial.println("Computed Imaginary values:");
  PrintVector(vImag, samples, SCL_INDEX);
  FFT.ComplexToMagnitude(vReal, vImag, samples); /* Compute magnitudes */
  Serial.println("Computed magnitudes:");
  PrintVector(vReal, (samples >> 1), SCL_FREQUENCY);
  double x = FFT.MajorPeak(vReal, samples, samplingFrequency);
  Serial.println(x, 6);
  while(1); /* Run Once */
  // delay(2000); /* Repeat after delay */
}
```

```
// PRINT MATRIX (float type)
void print_mtxf(const Eigen::MatrixXf& X){
    int i, j, nrow, ncol;
    nrow = X.rows();
    ncol = X.cols();

    Serial.print("nrow: "); Serial.println(nrow);
    Serial.print("ncol: "); Serial.println(ncol);
    Serial.println();

    for (i=0; i<nrow; i++){
        for (j=0; j<ncol; j++){
            Serial.print(X(i,j), 6);    // print 6 decimal places
            Serial.print(", ");
        }
        Serial.println();
    }
    Serial.println();
}

void PrintVector(double *vData, uint16_t bufferSize, uint8_t scaleType)
{
  for (uint16_t i = 0; i < bufferSize; i++){
    double abscissa;
    /* Print abscissa value */
    switch (scaleType){
      case SCL_INDEX:
        abscissa = (i * 1.0);
    break;
      case SCL_TIME:
        abscissa = ((i * 1.0) / samplingFrequency);
    break;
      case SCL_FREQUENCY:
        abscissa = ((i * 1.0 * samplingFrequency) / samples);
    break;
    }
    Serial.print(abscissa, 6);
    if(scaleType==SCL_FREQUENCY)
      Serial.print("Hz");
    Serial.print(" ");
    Serial.println(vData[i], 4);
  }
  Serial.println();
}
```