Index

Index 1
1. Abstract 2
2. Introduction 3
3. Robotic Vision and Navigation 5
4. Image Acquisition and Comparison 7
   4.1. Image Comparison 7
5. Omni Directional Vision 9
   5.1. The Omni Directional Camera 10
      5.1.1. The Retina-like Sensor 11
6. Log-Polar Imaging 12
   6.1. The Log-Polar Transform (LPT) 13
   6.2. The Log-Polar Transform: How It Works 14
   6.3. The Discrete Log-Polar Transform 15
   6.4. Discrete Mapping of the Foveal Region 16
   6.5. Discrete Mapping of the Peripheral Vision 16
7. The Fourier Transform 17
   7.1. Two Dimensional Fourier Transform 19
8. The Discrete Fourier Transform (DFT) 20
9. The Fast Fourier Transform (FFT) 22
10. Software 23
11. Experiments and Results 25
12. Conclusion 31
13. References 32

1. Abstract

This thesis proposes to study and demonstrate the properties of the Fourier Transform on log-polar images obtained by an omni-directional camera.

This study will be directed to log-polar images used in a previously developed method of robotic visualization and orientation, which bases its work on acquiring an image of the environment and comparing it to a similar image saved on the robot's computer database.

Its been observed that robotic orientation in a closed environment doesn't need most of the information contained in an image taken by the sensor in use. This gives the possibility of shortening the amount of information, retaining only that which matters.

Thus, one of the developed techniques is to apply the Fast Fourier Transform on the image taken at the robot's current location, obtaining a signature of this image. Such signature is compared to the signatures of **reference location images** using a similarity computation.

2. Introduction

One of the most elaborated problems concerning today's robotics is to find a correct method of navigation through an either simple or elaborated environment. One of the solutions to this problem passes through the development of an efficient method of identification of the environment where the robot is placed. Such method must be efficient concerning either computational speed, either the amount of information processed. It is taken as a 'good' method that which is able of increasing the former, with the least decrease of the latter.

So far, the environment mostly used in the development of these solutions has been an interior one. This is a natural choice, as these environments are less susceptible to unpredictable changes, and thus being easier to control and predict.

There have been several developed methods to allow the knowledge of the surrounding environment to the robot. One is the acquiring of information through sensors mounted on the robot, such as laser or sonic sensors. But these sensors might be noisy, and the information received not always trust-worthy. Another method is to give the robot a topical map of the environment. But also this method is not full proof, as the environment, although controlled, might change.

A method that allows a positive balance between these previous factors is the visual acquisition of information. For being a type of information that is processed at the instant of the robot's operations, it's trustful to provide the robot with the knowledge of a change in the environment. Also, certain visual methods are accurate to a certain point to give out enough correct information. One of those methods is the omni-directional vision.

The type of omni-directional vision in which this paper is based uses an omni-directional camera pointed at a hyperbolically curved mirror placed on top of it. This provides an instantaneous 360º image, thus obtaining a complete vision of the surrounding environment, without the need of camera movement for the withdrawal of other vision angles like it would succeed in a traditional camera.

The omni-directional camera sensor is called **retina-like** sensor, as it imitates the human retina when it comes to a high resolution in the center of the image, called the fovea, resolution that decreases with the increase of the distance to the center.

The images obtained with this camera have a circular field. In order to facilitate the processing of this information, it can be applied the Log-Polar Transform, which transforms these cartesian images into log-polar ones.

The processing of the information of an image can be done on one or several aspects of the picture. Color, shapes or sizes, hue, light polarization or gray-scale analysis. The subject under study in this paper uses the gray-scale information of the recorded images. To study the information it's used the **Fourier Transform**.

The Fourier Transform is a mathematical tool that decomposes an image into its sine and cosine components. The objective of this paper is to study the properties of this tool in log-polar images. For such, a software will be developed using the **Fast Fourier Transform**, which is a useful algorithm to quickly compute the **Discrete Fourier Transform (DFT)**. The DFT is a simplification of the Fourier Transform as it only works in a discrete set of points, withdrawing the Fourier Transform characteristic of working in an infinite set of points.

A second objective of this work is to ascertain the differences of the FFT computation between log-polar images and cartesian images, concluding on which one may provide a faster computational speed.

The paper is composed as the following: in Chapter 3 there are some types of robotic vision described; Chapter 4 refers to the characteristics of image acquisition; Chapter 5 talks about omni-directional vision, and more specifically, of the omni-directional instruments used for this work; Chapter 6 explains the Log-Polar Transform; in Chapter 7 there is a small resume of the Fourier Transform; the following Chapters 8 and 9 talk about the Discrete Fourier Transform and the Fast Fourier Transform, respectively; following this is Chapter 10 with a description of the software used and the one developed and Chapter 11 describes the procedure and results of the experiments with the software used.

3. Robotic Vision and Navigation

Following a wall, avoiding an obstacle. The movements of robots are limited to simple actions. Yet many applications would benefit from genuinely autonomous systems: robots able to move across partially unknown and changing terrain.

Several approaches to this matter have been under development, each one with a different guideline. The **Sahabot 2** robot (Image 1), for example, developed at the Zurich University laboratories, works by memorizing fixed reference points, and using sensors to analyze the polarization of the light.

Image 1 – The Sahabot 2 records visual reference points and navigates with the help of sensors which analyse the polarization of the light.

The main interest as of now concerning robotic visualization and motion is for interior spaces, in which the reference points are composed of one or several environment objects or characteristics. Recognizing these points facilitates localization, which is a prerequisite for movement. One advantage is that this recognition works even when the reference points are observed from different positions.

Researchers at the Royal Institute of Technologies in Stockholm have designed a robot fitted with one central camera and two side cameras. Drawing on the information supplied by this tri-ocular vision, it is able to move around, successfully avoiding obstacles.

This initial research on visual landmarks made it possible to select and recognize different objects or configurations for both metric and topological localisation.

O.A. Aider, T. Chateau and J. T. Lapresté [1] developed a method for autonomous environment mapping, localisation and navigation for an indoor mobile robot using monocular vision and multiple 2D pattern tracking. The environment map is a mosaic of 2D patterns detected on the ceiling plane and used as natural landmarks. The algorithm developed enables the mobile robot to reproduce learned trajectories defined by key images representing the visual memory. The pattern tracker is a contour model-based one and takes into account the image gray level scale level variations. It uses the condensation algorithm[1] to track efficiently 2D patterns on cluttered background. An original observation model is used to update the particle filter[2] state.

In another navigation method proposed [2], a robot is equipped with an omni-directional camera and takes a set of omni-directional images (log-polar images) at reference locations, and then compares the current omni-directional image with the reference images stored in its database. To store and compare the images, it uses a **Fourier Signature**, which is, as known, the subject under study in this paper. The referred database is created in a first phase with the robot exploring the environment and taking pictures at **reference locations** that will be recorded as **reference images**. Afterwards, when the robot is in its normal function, it uses a **similarity computation**, thus acknowledging the robot of its approximate whereabouts in the environment.

But the scope of a robot can be a little more complex than to move around in a closed environment. D. Nikovski studied a method for a robot to not only move around in an environment, but also to do a more precise task: in this case, exiting a room through a door [3]. The problem ascertained is that the robot cannot be sure that it has successfully exited the room, because any gaps between two pieces of furniture can be mistaken for a door. So, in order to successfully exit and enter rooms, the robot must first recognize where the door is. A convenient representation of its position are the polar coordinates of the midpoint of the line that connects the two door posts. The robot needs to find the direction to the door, as well as the distance from the center of the robot to that midpoint. These two coordinates can be found independently by means of memory-based learning (MBL). The advantage of this scheme, when in comparison to an NN-based system[3], is that it requires no training prior to its use. The first step is to acquire a wide panoramic image of the surrounding scene. The second step is to find the door in the panorama, and the third is to estimate the distance to the door.

[1] The **condensation algorithm** (**Con**ditional **Dens**ity Propag**ation**) is a probabilistic algorithm for tracking objects in a cluttered background.

[2] **Particle filter methods**, also known as *Sequential Monte Carlo* (SMC), are sophisticated model estimation techniques based on simulation.

[3] Neural Network Based System

4. Image Acquisition and Comparison

The proposed technique for robotic navigation requires a fast and minimally accurate form of image acquisition.

There are two types of image acquisition and annotation:

- One, based on full content retrieval, such as marked objects, foreground and background, made by human effort. This approach results in an accurate retrieval of information, but it's time consuming.
- The other is based on the automatic extraction of physical properties such as color, light, texture, dimensions, etc. Although smaller in content of information, this approach allows fast computation of it, and can be done unassisted.

Due to its characteristics, the latter form of image acquisition is the one used in robotic vision and navigation, as it may, in some cases, present a good enough performance for the purpose at hand. Also, because of its unassisted nature, it is worth to be considered in large databases and for applications where the cost of image processing must be kept low in terms of human resources.

In the case under study, robot navigation is obtained by retrieving images by visual similarity.

In order to accomplish this task, a signature is necessary. This signature, obtained by the FFT of the image taken (discussed in a following chapter), holds values computed in such a way that simple matching procedure can be used for retrieving images with similar features.

A metric, that allows comparison of different signatures, is also needed. This comparison may be made possibly according to some filtering procedure that accounts for partial query specification or weight of query terms.

Finally, a verification procedure is in want [4].

4.1. Image Comparison

Retrieval of images by similarity can be grounded on the distance between the image signatures.

Since the values in the signature represent an integration over the image, the choice of the whole signature seems appropriate when the image exhibits only one main direction. More frequently, images contain components that visibly suggest a main orientation, together with other components with different layouts. Therefore, using only a subrange of the signature allows more precise retrieval of images that, while presenting a similar overall layout, may differ in details or small components [4].

5. Omni-directional Vision

Recently, an increased interest in omni-directional vision for applications in robotics could be noted. Technically, omni-directional vision, sometimes also called *panoramic* vision, can be achieved in various ways. Examples are cameras with extreme wide angle lenses ("fish-eye"), cameras with hyperbolically curved mirrors mounted in front of a standard lens, sets of cameras mounted in a ring or sphere-like fashion, or an ordinary camera that rotates around an axis and takes a sequence of images that cover a field of view of 360 degrees.

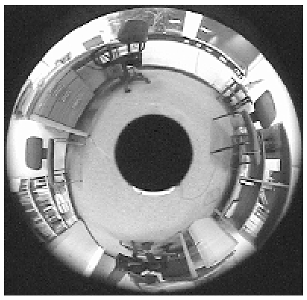


Image 2 - Catadioptric image of an office

Omni-directional vision provides a very large field of view, which has some useful properties. For instance, it can facilitate the tracking of moving objects in the scene.

On the other hand, vision algorithms have to account for the specific properties of the particular omni-directional imaging sensor setup at hand. This may comprise theoretical and methodological challenges, as is the case for catadioptric[4] vision. Here, the extreme geometrical image distortions caused by the hyperbolic mirror (Image 2) require a suitable adaptation of image interpretation methods.

As you may understand, the size and shape of an object viewed in a curved mirror depends on the shape of the mirror and the position of the object. If the object moves, then it usually gets distorted into different shapes as it moves. In particular, for most catadioptric sensors, the area of the object can change drastically, even if the distance from the object to the sensor remains constant [5].

[4] (Also spelled catadioptic): Reflecting telescope, so-called because the beam of light is 'folded', i.e. reflected, back through a hole in the main mirror, before reaching the eyepiece. The effect is to increase the telescope's focal length, thus producing a more portable but also costlier instrument. Catadioptrics use a lens-like correcting plate in the front for spherical aberration.

5.1. The Omni Directional Camera

Image 3 – Omni-directional Camera

This is the camera used in the work developed that lead to the making of the current thesis.

Normally, the visual sensor used in traditional cameras is composed of photoreceptors that are organized according to a square matrix, and have all the same size.

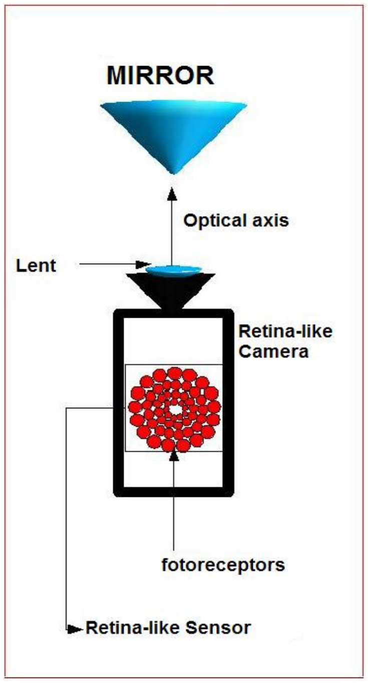


Image 4

This camera uses a new type of sensor, the **retina-like sensor** (Image 5), structurally different from the traditional one. The retina-like sensor is a new generation sensor whose structure is inspired in the functionality of the human retina. In fact, the human eye possesses, at the center of the retina, a big visual accuracy, in a zone called the 'fovea', and while increasing the distance from it, the visual accuracy decreases. Imitating this biological aspect, the sensor presents a high density of CMOS photoreceptors in the center allowing a good resolution at the center of the picture, and a decreasing photoreceptor density from the center to the periphery of the sensor, where the resolution of the picture becomes minimum.

The most important benefit of the introduction of this omni-directional camera use is, as said, the increase of the visual field, which associated to a notable reduction of the memory size of the pictures, may create a considerable increase in computational efficiency applied to numerous applications. The inherent characteristics and the invariables in the log-polar images may also be used by to develop algorithms of several types to be used in the ambit of robotics, but also in other several applications, such as surveillance, teleconference or the exchange of data through the Internet.

5.1.1. The Retina-like Sensor

The retina-like sensor is characterized by a space-variant resolution that, as said, imitates the distribution of the photoreceptors present in the human retina. The density of the CMOS photoreceptors is less in the periphery than in the center, increasing as it comes closer to the latter. The photoreceptors are disposed in concentric rings and the distance between them is not constant, as it also decreases while getting closer to the fovea's delimitating ring. Simultaneously to this phenomenon, it's also verifiable that the photoreceptors have their biggest dimensions on the periphery of the sensor, dimension which, obviously, decreases towards the center. This phenomenon stops when it's reached the delimitating ring of the fovea. Inside this ring the photoreceptors have all the same dimension, which is the minimum possible. Therefore, in the fovea, it is verifiable the space-variance characteristic of the sensor.

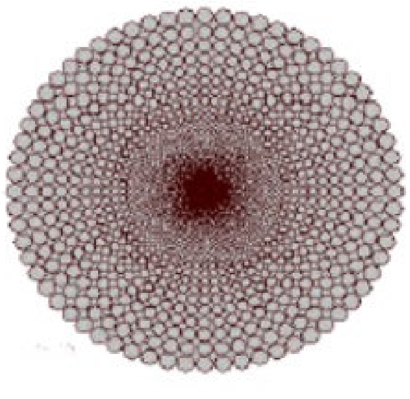


Image 5 – Retina-like Sensor Layout

6. Log-Polar Imaging

The Log-Polar Imaging is a method with space-variant resolution, that as been introduced, as previously referred, in order to mimic the imaging geometry of the human retina.

When compared to the usual Cartesian images, the log-polar images allow faster sampling rates on artificial vision systems without reducing the size of the field of vision and the resolution on the central part of the fovea.

The log-polar geometry has been shown to have advantages for several robotic applications such as linear flow estimation, looming detection and vergence control [6].

The log-polar transformation is a conformal mapping from *t* points on the Cartesian plane (x, y) to points in the Log-Polar plane (ξ, η):

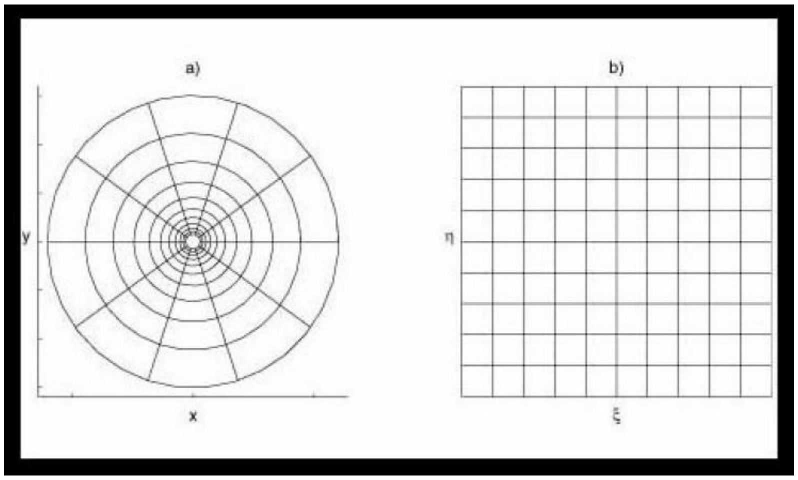


Image 6 – (a) Cartesian plane; (b) Log-Polar plane

The mapping is described by:

$$\xi = \log\cdot\sqrt{x^2 + y^2}$$

$$\eta = \operatorname{atan}\left(\frac{y}{x}\right)$$

One advantage of this kind of sampling is data reduction. This is obtained by reducing the resolution at the image periphery, thus high resolution in the fovea improves the performance of the tracking process.

For tracking, the main advantage of the log-polar geometry is that objects occupying the central high resolution part of the visual field become dominant over coarsely sampled background elements in the image periphery. This embeds an implicit focus of attention in the center of the visual field where the target is expected to be most of the time. The use of log-polar images increases the size range of objects that can be tracked using a simple translation model.

6.1. The Log-polar Transform

The log-polar transform (LPT) is a space-variant image-encoding scheme. The transform domain is the set of two-dimensional images, I(x,y), indexed by a Cartesian coordinate system. The range of the transform is the surface, $I^*(\xi, \eta)$, of a cylinder of unit radius. The ξ-coordinate axis is a designated line on the surface parallel to the cylinder's axis. The η-coordinate axis is the unit circle at the origin end of the cylinder.

The LPT is centered in I at specific point, (x_0, y_0), called the transform origin. The ξ-coordinate of a point in I* is proportional to the logarithm of the radial distance from the origin (plus an offset) of the corresponding point in I. The η-coordinate is equal to the angular distance of the corresponding point from the x-axis in I.

For a digital image, the mapping from I to I* is discrete and not one-to-one.

The area of the regions increases with radial distance from the origin of I. In regions of I sufficiently far from the origin, the mapping from I to I* is many pixels to one. Thus, resolution decreases with axial distance, ξ.

As said before regarding Log-Polar images, the utility of the LPT stems from some of its invariance properties and from data reduction which results from the low resolution of much of I*. In its periphery, I is downsampled increasingly with distance from the origin, since all the pixels in one region of the cobweb map to a single pixel in I*. The data reduction is significant and can greatly speed image processing.

The LPT has some useful properties beyond the data reduction afforded by space variance. It is conformal – continuity is preserved, angles can be measured, and local geometric properties are measurable. It is invariant with respect to scale and rotation – linear scaling and rotations are transformed into linear shifts along the ξ and η axes.

One disadvantage of the LPT is the low resolution of the image away from the origin, as it complicates feature detection and identification. To view an area in the periphery in high resolution requires the shift of the LPT origin to a point in that area. In a computer vision system that uses the LPT, the transform origin usually lies at the fixed point in the image plane of the camera.

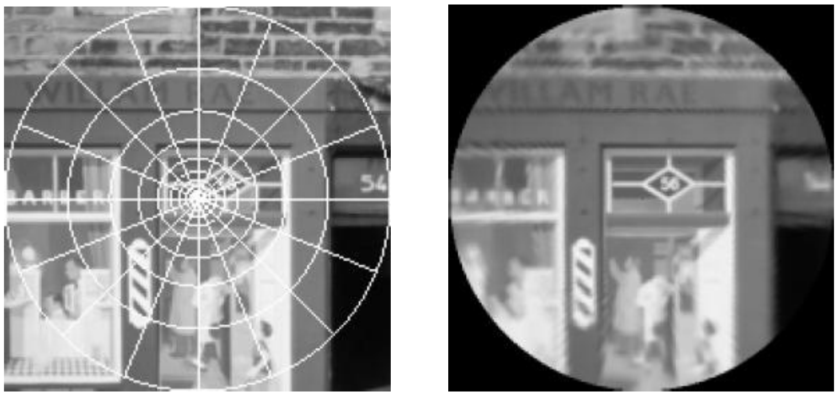


Image 7 – (a) Before LPT (b) After LPT – Take notice of the number 54 on the periphery

After the LPT is applied, only large features are visible in the extreme periphery of the image (Image 7b). To image a feature in the periphery at high resolution, the camera must be moved so the optical axis intersects the scene close to the feature. Thus, high resolution imaging of a scene requires motion of the camera. Active vision increases the mechanical complexity of the system but provides many advantages. In some cases, these advantages, which include computational savings gleaned from lower data input bandwidth, more than offset the complexity introduced by the camera motion.

6.2. The Log-Polar Transform: How It Works

If I(x, y) is an image with support on a rectangular set in the Euclidean Plane, then the log-polar transform with origin (x_0, y_0) is

$$I^*(\xi,\eta) = L\{I(x, y); (x_0, y_0)\}$$

where

$$\xi = M\cdot\log(r + \alpha),$$

$$r = \sqrt{(x - x_0)^2 + (y - y_0)^2}, \text{ and}$$

$$\eta = \tan^{-1}\cdot\left(\frac{y - y_0}{x - x_0}\right)$$

The transform maps a two-dimensional image onto the surface of a cylinder. The cylinder is indexed by ξ and η. The ξ-axis is parallel to the axis of the cylinder. The η-axis forms a circle around the cylinder. When the LPT is used in conjunction with a moving camera system the origin, (x_0, y_0), of the transform of I, is generally fixed on the camera's image plane in a pixel near the optical axis. The origin of I* is fixed at one end of the cylinder.

6.3. The Discrete Log-Polar Transform

The discrete sampling of a digital image, I, prevents direct implementation of the transform, which would require infinite resolution near the origin. Moreover, the cylindrical image, I*, is discrete in any real application. Let **p** be the discrete pixel location in I defined by

$$\mathbf{p}(r, \theta) = (x(r, \theta), y(r, \theta)),$$

where (x, y) is the pair of integers

$$x = |r \cos(\theta)|, \text{ and } y = |r \sin(\theta)|.$$

which minimizes

$$\sqrt{(x - x_0)^2 + (y - y_0)^2} \geq r$$

Then **p**(r, θ) is the pixel location in I closest to a Euclidean radial distance r from the origin at an angle θ from the x-axis.

Given an integer, m, there is a radius, $r = r_0$, such that maps one pixel, $I(x,y)=I(p(r_0,k2\pi/m))$, to one pixel, $I^*(\xi, \eta)$ uniquely for each integer k from 0 through m-1. This circle of m pixels in I is called the one-to-one circle. L maps the one-to-one circle to a ring of pixels around I* at a distance $\xi = \xi_0 = M \log (r_0 + \alpha)$ from the origin end of the cylinder. Thus, the number of pixels around the η-axis of I* is m. Equivalently, if I* is specified as m x n, where 2π/m is the sampling distance along the η-axis, then r_0 must be the smallest integer for which

$$\| \{(x, y) = p(r0, k2\pi/m) | \; k = 0, ..., m\text{-}1, \qquad \text{and}$$
$$p(i, k2\pi/m) \neq p(j, k2\pi/m) \text{ for } i \neq j\}\| = m$$

where ||A|| is the number of elements in the set A.

Inside the one-to-one circle ($r < r_0$) is the fovea where one pixel, I(x, y), is mapped to more than one pixel, I*(ξ, η). Outside the circle ($r > r_0$) is the periphery where it is possible that more than one pixel from I is mapped to a single pixel in I*.

6.4. Discrete Mapping of the Fovea Region

There are many ways to code the foveal region of I. The most simple, and probably most useful, is to leave it unchanged. This is equivalent to discarding the foveal portion of the cylinder and starting the cylinder at the one-to-one ring. Then the first ring of pixels on the tubular surface of I* contains the pixels

$$\{I(p(r_0, (2k\pi)/m)) | \; k = 0, ..., m-1\}$$

The rest of the periphery is coded as usual.

This arrangement complicates the data structure for representing the cylinder since the fovea, in effect, must be coded as a separate image. But, this has the advantage of permitting standard cartesian image processing routines to be applied to the fovea, which is, of course, very small.

Another way to code the fovea is to map the circle of pixels at a radius r in I to the ring $\xi = r$ in I*. This is done for each radius $r = 0$ to $r = r_0 - 1$. Since the number of pixels in each circle is less than the number of pixels in the ring from I*, either the pixels must be duplicated with their corresponding rings or some pixels in the foveal rings must be left blank. Normally it's done the former.

6.5. Discrete Mapping of the Peripheral Vision

Computation of the periphery is more complicated. It requires the averaging of sets of pixels from I and it depends on the values of transform parameters M and α. Within the periphery of I the mapping is many-to-one. I is partitioned into a cobweb-like set of regions, each of which maps through L to one pixel in I*. The area of the regions in I increases with radial position in I*. Along the ρ-axis a unit distance maps:

$$\begin{aligned} \xi - (\xi - 1) \Leftrightarrow\ & r(\xi) - r(\xi - 1) \\ &= e^{\xi/M} - e^{(\xi-1)/M} \\ &= e^{\xi/M} (1 - e^{-1/M}), \end{aligned}$$

which, clearly, grows exponentially with ξ. Along η-axis the *angular* distance in I is constant but the *spatial* distance increases as $2\pi r/m$ [7].

7. The Fourier Transform

The Fourier Transform is an important image-processing tool, which is used to decompose an image into its sine and cosine components. The output of the transformation represents the image in the *Fourier* or frequency domain, while the input image is the spatial domain equivalent. In the Fourier domain image, each point represents a particular frequency contained in the spatial domain image.

The Fourier Transform is simply a method of expressing a function in terms of the sum of its projections onto a set of basic functions. As an image is defined on a closed domain, it can be assumed that the image is *zero* outside the window, that is, its integrable over the real line.

A step function can be represented as a sum of sine waves of frequency ω, 3ω, 5ω, ... where ω is the frequency of the square wave.

In images we are concerned with spatial frequency, that is, the rate at which brightness in the image varies across the image, or varies with viewing angle.

From the decomposition of the signal into varying sinusoidal components we can construct a diagram displaying the amplitudes of all the sinusoids for all frequencies.

Note that negative frequencies (whatever that might actually mean) must be considered so the sinusoidal component of frequency f and amplitude A has to be split into two components of amplitude A/2 at the frequencies +f and –f.

When the Fourier transform of an image is calculated, we treat the intensity signal across the image as a function, not just an array of values. The Fourier Transform describes a way of decomposing a function into a sum of orthogonal basis functions.

To calculate the Fourier Transform of a function, one must consider that the function is just, conceptually, a point in some vector space. Given our orthogonal basis functions, we calculate the component of our given function in each of the basis functions by calculating the inner product between the two. The standard basis functions used for Fourier transform are $\{\sin(2\pi\omega x), \cos(2\pi\omega x), \omega \in R\}$ or, equivalently $\{e^{-i2\pi\omega x}, \omega \in R\}$. It is the frequency ω that varies over the set of all real numbers to give us the infinite collection of basis functions.

Since

$$e^{i2\pi\omega x} = \cos(2\pi\omega x) + i.\sin(2\pi\omega x),$$

we see that the Fourier transform has real and imaginary components. Moreover, the exponential form of basis function allows us to represent both real and complex valued functions by their Fourier transform.

We can show that any two basis functions of different frequencies are orthogonal by calculating their inner product and showing that it is 0.

Thus, we project our given function *f* onto our basis functions $e^{-i2\pi\omega x}$ to get the Fourier amplitudes F(ω) for each frequency ω:

$$F(f(x)) = F(\omega) = \int f(x)\, e^{-i2\pi\omega x}\, dx$$

In general, F(ω) will be complex, for example of the form a(ω) + i b(ω).

F is often expressed in polar form though:

$$F(\omega) = |F(\omega)|\, e^{\,i\Phi(\omega)},$$

Where

$$|F(\omega)| = \sqrt{(a^2 + b^2)}$$

and

$$\Phi(\omega) = \tan^{-1}(a/b)$$

The norm of the amplitude, |F(ω)| is called the *Fourier spectrum* of *f*, and the exponent Φ(ω) is called the *phase angle*. The square of |F(ω)|, $|F(\omega)|^2 = P(\omega) = a^2(\omega) + b^2(\omega)$ and is called the *power spectrum* of *f.*

In many applications only the amplitude information is needed and the phase information is discarded. However, phase information should not be discarded, as it carries considerable information. If synthetic images made from the amplitude information of one image and the phase information of another are constructed, it is the image corresponding to the phase data that we perceive, if somewhat degraded. Also, if we want to re-transform the Fourier image into the correct spatial domain after some processing in the frequency domain, we must make sure to preserve both magnitude and phase of the Fourier image.

When it is necessary to reconstruct the original function from its Fourier components, it is simply done by summing up all the Fourier components multiplied by their corresponding basis function:

$$f(x) = \int F(\omega)\cdot e^{2\pi\cdot i\cdot\omega\cdot x}\, d\omega$$

Note that the inverse Fourier Transform uses the basis function $e^{i2\pi\omega x}$, whilst the Fourier transform uses the basis functions $e^{-i2\pi\omega x}$. This prevents a sign change occurring in the reconstruction process.

7.1. Two-dimensional Fourier Transform

Now an image is thought of as two-dimensional function and so the Fourier transform of an image is a two dimensional object. Thus, if *f* is an image, then

$$F(\omega, v) = \int\int f(x,y)\cdot e^{-2\pi\, i(\omega x+\omega y)}\, dx\, dy$$

Fortunately, as we will see in a following chapter, it is possible to calculate this integral in two stages, since the 2D Fourier transform is separable. Thus, we first form the Fourier transform with respect to x:

$$F(\omega, y) = \int f(x,y)\cdot e^{-2\pi i\cdot\omega\cdot x}\, dx$$

and then we calculate the Fourier transform of this function of y,

$$F(\omega, v) = \int f(\omega,y)\cdot e^{-2\pi i\cdot v\cdot y}\, dy$$

8. The Discrete Fourier Transform

In digital images one can only process a function defined on a discrete set *f* points. This leads to the *Discrete Fourier Transform* (DFT). The DFT provides information over a discrete number of frequencies, so it's necessary to determine precisely which frequencies these are. The sample of frequencies, however, is large enough to fully describe the spatial domain image. To do this, we sample a continuous function *f* at intervals of Δ, generating sequence of sampled values:

$$f_k = f(k\cdot\Delta)$$

where k = ..., -2, -1, 0, 1, 2,... The reciprocal of Δ is the sampling rate, or frequency. For any sampling interval Δ, there is a special frequency ω_c called the Nyquist critical frequency:

$$\omega_c = \frac{1}{2\cdot\Delta}$$

This frequency represents the highest frequency that can be represented by something sampled at intervals of Δ, or, in other words, having the wavelength of 2Δ. If we are searching to describe a function by a set of discrete values, we must sample the function at 2 times the highest frequency in the function.

We generate the Fourier transform at discrete frequencies,

$$\omega_n = \frac{n}{N\cdot\Delta}$$

where,

$$n = \frac{-N}{2}, \frac{-N}{2} + 1, ..., \frac{N}{2}$$

These limits correspond to the upper and lower Nyquist frequencies.

For a square image of NxN, the two dimensional DFT is given by:

$$F(k,l) = \frac{1}{N^2} \cdot \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} f(i,j) \cdot e^{-i \cdot 2 \cdot \pi \left(\frac{k \cdot i}{N} + \frac{l \cdot i}{N} \right)}$$

where f(i,j) is the image in the spatial domain and the exponential term is the basis function corresponding to each point F(k,l) in the Fourier space. The equation can be interpreted as: the value of each point is obtained by multiplying the spatial image with the corresponding base function and summing the result.

The basis functions are sine and cosine waves with increasing frequencies, i.e. F (0,0) represents the DC-component of the image which corresponds to the average brightness and F (N-1, N-1) represents the highest frequency.

Because the Fourier Transform is separable, as said before, it can be written as:

$$F(k,l) = \frac{1}{N} \cdot \sum_{j=0}^{N-1} P(k,l) \cdot e^{-i2\pi \frac{i \cdot j}{N}}$$

$$P(k,j) = \frac{1}{N} \cdot \sum_{i=0}^{N-1} f(i,j) \cdot e^{-i \cdot 2 \cdot \pi \cdot \frac{k \cdot i}{N}}$$

The DFT needs N complex multiplications of f_k by $e^{-2\pi ikn/N}$ and N-1 additions of resulting values to transform N values; its complexity is thus proportional to N^2.

As it will be seen in the next chapter, the Fast Fourier Transform (FFT) is an ingenious algorithm, which exploits various properties of the Fourier Transform to enable the transformation to be done in N $\log_2$ N operations. However, the FFT requires the size of the input data to be a power of 2; if this is not the case, the data are either truncated or padded out with zeros.

9. The Fast Fourier Transform

As said before, Fast Fourier Transform (FFT) is an efficient algorithm to compute the DFT and its inverse.

By far the most common FFT is the Cooley-Tukey algorithm. This is a **divide and conquer**[5] algorithm that recursively breaks down a DFT of any composite size $n = n_1 n_2$ into many smaller DFTs of sizes n_1 and n_2, along with $O(n)$ multiplications by complex roots of unity traditionally called **twiddle factors**.

This method (and the general idea of an FFT) was popularized by a publication of J. W. Cooley and J. W. Tukey in 1965, but it was later discovered that those two authors had independently re-invented an algorithm known to Carl Friedrich Gauss around 1805 (and subsequently rediscovered several times in limited forms) [8].

The most well-known use of the Cooley-Tukey algorithm is to divide the transform into two pieces of size $n / 2$ at each step, and is therefore limited to power-of-two sizes, but any factorization can be used in general (as was known to both Gauss and Cooley/Tukey). These are called the **radix-2** and **mixed-radix** cases, respectively (and other variants have their own names as well). Although the basic idea is recursive, most traditional implementations rearrange the algorithm to avoid explicit recursion. Also, because the Cooley-Tukey algorithm breaks the DFT into smaller DFTs, it can be combined arbitrarily with any other algorithm for the DFT.

The FFT does not directly give you the spectrum of a signal. It can vary dramatically depending on the number of points (N) of the FFT, and the number of periods of the signal that are represented. There is another problem as well.

The FFT contains information between 0 and Δ; however, as said before, the sampling frequency must be at least twice the highest frequency component. Therefore, the signal's spectrum should be entirely below $\Delta/2$, the Nyquist frequency.

Recall also that a real signal should have a transform magnitude that is symmetrical for positive and negative frequencies. So instead of having a spectrum that goes from 0 to fs, it would be more appropriate to show the spectrum from $-\Delta/2$ to $\Delta/2$.

[5] In computer science, **divide and conquer** (**D&C**) is an important algorithm design paradigm. It works by recursively breaking down a problem into two or more sub-problems of the same (or related) type, until these become simple enough to be solved directly. The solutions to the sub-problems are then combined to give a solution to the original problem [8].

10. Software

In order to work with the log-polar images recorded with the omni-directional camera, it was necessary to use the Log-Polar Transform to create cartesian images out of the log-polar ones. For that, it was used the software (Image 11) developed by C. Ardito [9], which works on the MATLAB™ environment. It allows to load and graphically visualize the log-polar images and their correspondent cartesians.

The software also allows saving the log-polar and cartesian images in the disk, under the bitmap (*.bmp) format., being this the functionality used for this paper. The software has also other functionalities, but not of interest to this paper.

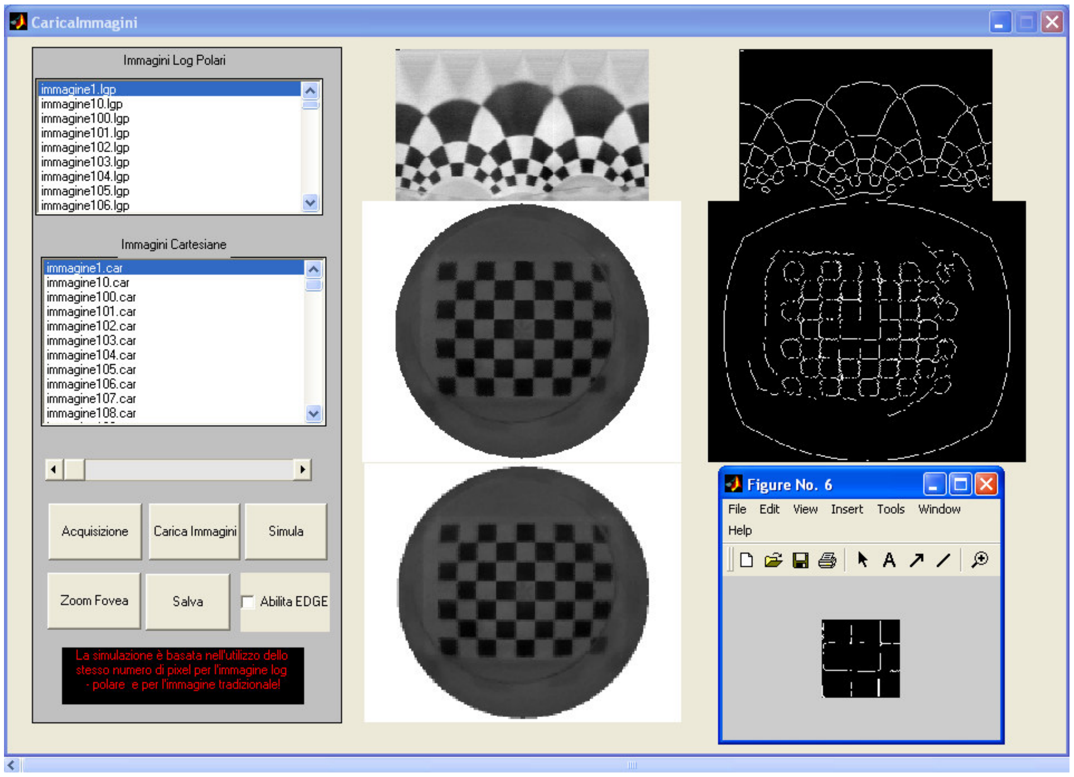


immagine1.lgp
immagine10.lgp
immagine100.lgp
immagine101.lgp
immagine102.lgp
immagine103.lgp
immagine104.lgp
immagine105.lgp
immagine106.lgp

immagine1.car
immagine10.car
immagine100.car
immagine101.car
immagine102.car
immagine103.car
immagine104.car
immagine105.car
immagine106.car
immagine107.car
immagine108.car

La simulazione è basata nell'utilizzo dello stesso numero di pixel per l'immagine log - polare e per l'immagine tradizionale!

Image 11

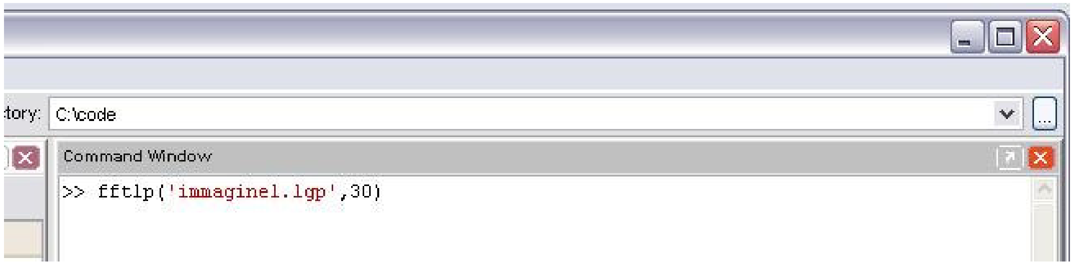
The software designed for this paper to implement and analyse the Fast Fourier Transform properties in Log-Polar images was also developed in the MATLAB™ environment.

For such, two MATLAB™ functions were developed to implement this algorithm on the images. One for the log-polar and another for the cartesian images. This difference is due to the different between the number of rows of the log-polar images (152) and the number of rows in the cartesian ones (288).

The input of the functions is two variables: the name of the image that the user wishes to analyze, and the number of components of the FFT the user wishes to implement on the analysis.

Its output is a plot of two graphics, the left one representing the **Magnitude-to-Frequency** variation, and the right one representing the **Magnitude-to-Number of Rows** variation.

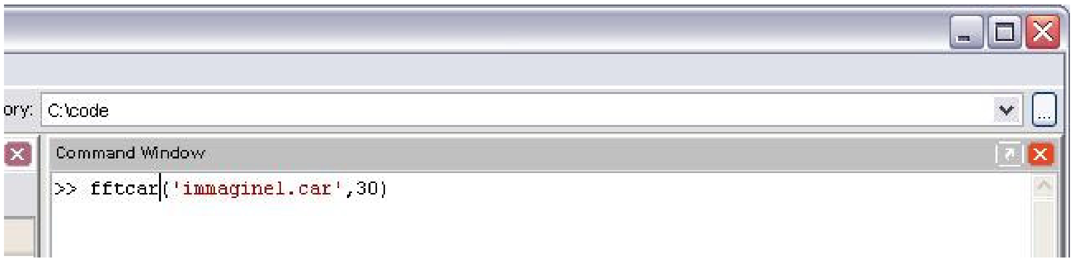
If the user wishes to analyze **log-polar** images, must only write `fftlp(NAMEOFFILE,N)` on the MATLAB™ command window, being `NAMEOFFILE` the name of the image and `N` the number of components to be processed in the FFT. For example:



```
>> fftlp('immaginel.lgp',30)
```

Of course, the image files must be in the same directory as the function.

If the user wishes to analyze the **cartesian** images, it must write `fftcar(NAMEOFFILE,N)` on the MATLAB™ command window. For example:



```
>> fftcar('immaginel.car',30)
```

All else that has been said concerning the previous function also applies to this one.

11. Experiments and Results

With the previously described software it was possible to obtain an output of the FFT done on the log-polar and cartesian images. The output was, as said, two graphics that demonstrated the variation of the FFT magnitude with the frequency and with the rows of the picture.

One thing that the output of the software provided was the possibility to test what the value of N should be in order to give enough information for the Fourier signature, but without being too high, so as not to increase the time of the FFT computation. This test was done using a log-polar image (Image 12) on the appropriate software referred in the previous chapter.

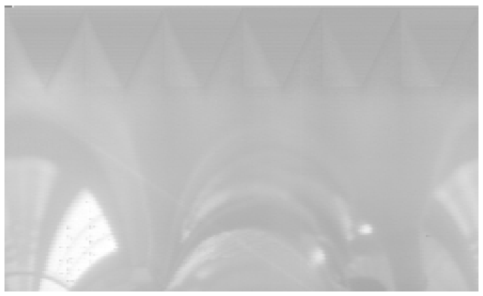
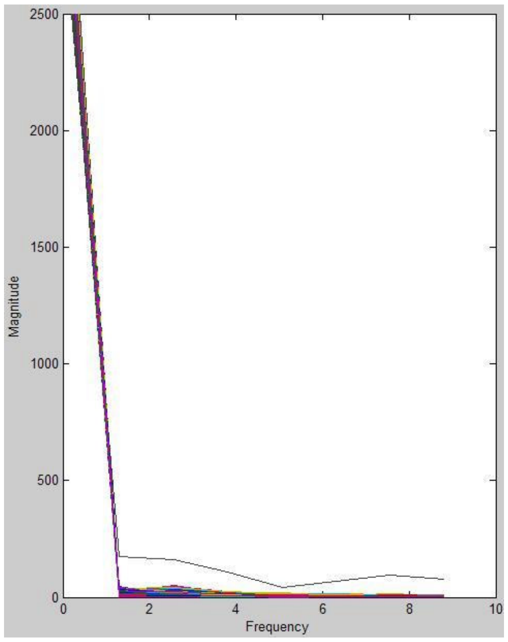
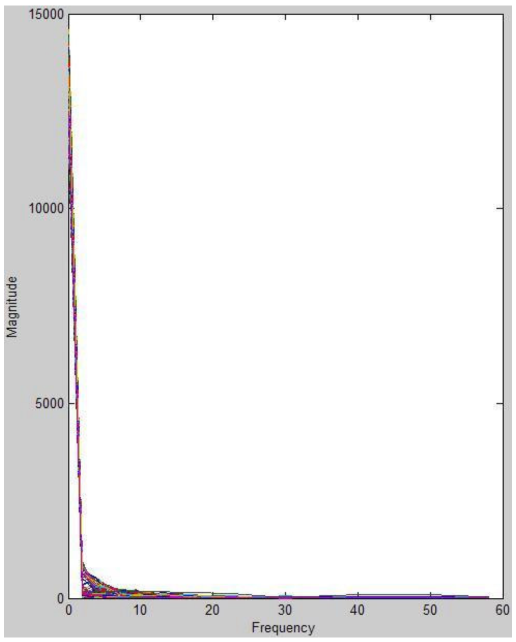


Image 12 – Log-polar image

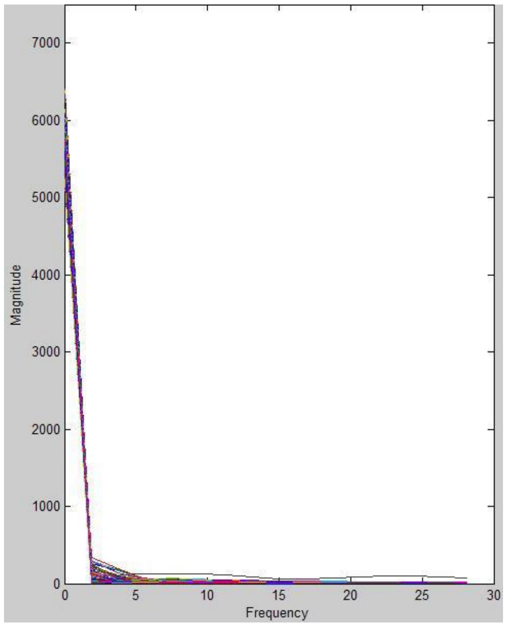
The outcome of this test was:



With N = 10



With N = 60



With N = 30

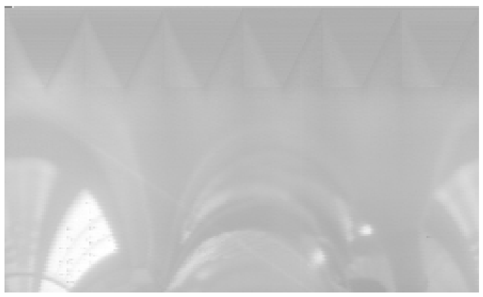
With these results, one can verify that the signal's magnitude decreases with the frequency. This means that the biggest amount of information is retrieved at the lowest frequencies.

It is also verifiable that with N = 10 the amount of information is quite small, probably not enough for the purpose of creating a valid Fourier signature, and with N = 60 there is only a significant amount of information up until around the 30th component, after which it is very small, and can be ignored, thus N = 60 being considered too much. With N = 30, it is verifiable that it presents a good amount of information up until the 20th component, meaning that probably N = 30is a good value, and that the components up to the 20th are worth being saved.

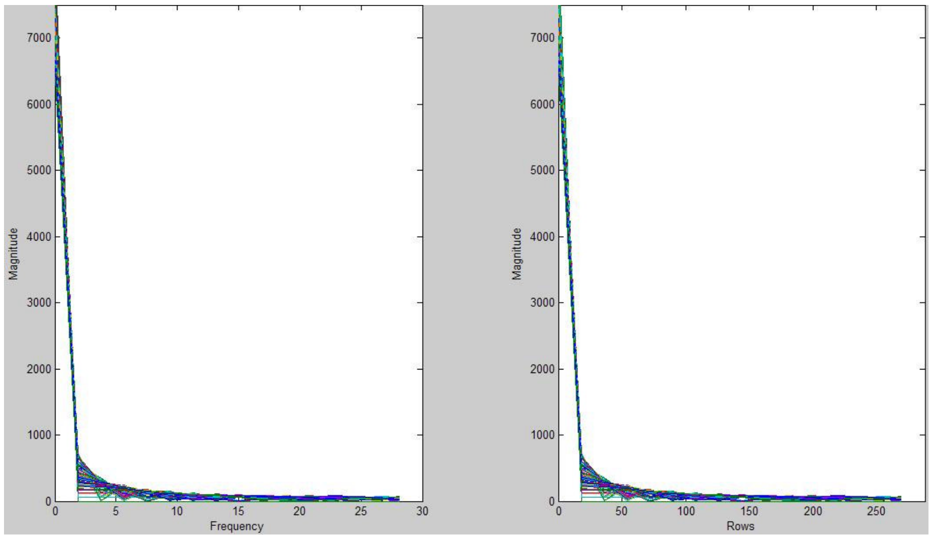
Next, it was studied the variation of the magnitude of the signal with the frequency and the row of the image. For this, the outcome of the tests on 3 cartesian images and on their log-polar equivalent will be presented. All tests were done with N = 30.

The outcome of these tests are as such:

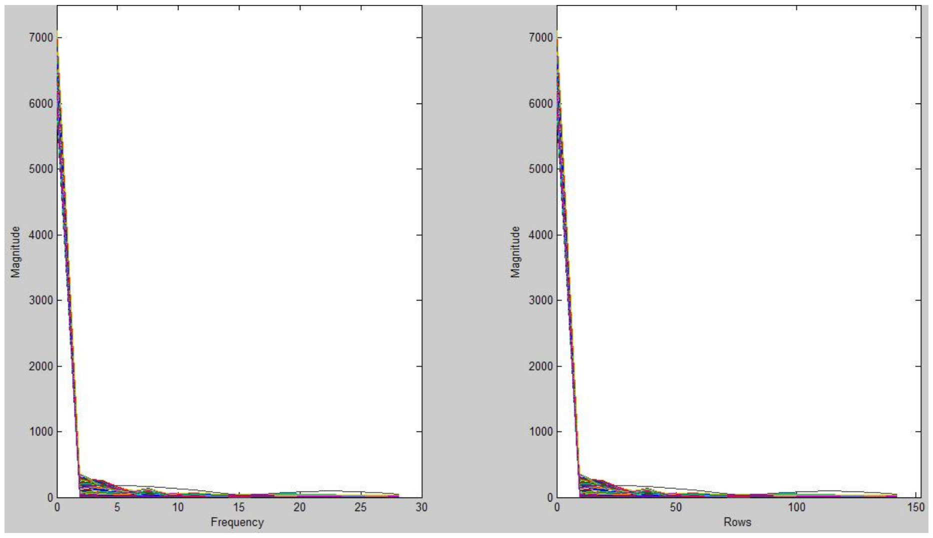
Cartesian Image 1



Log-Polar Image 1

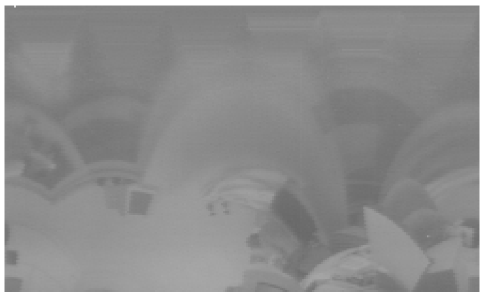


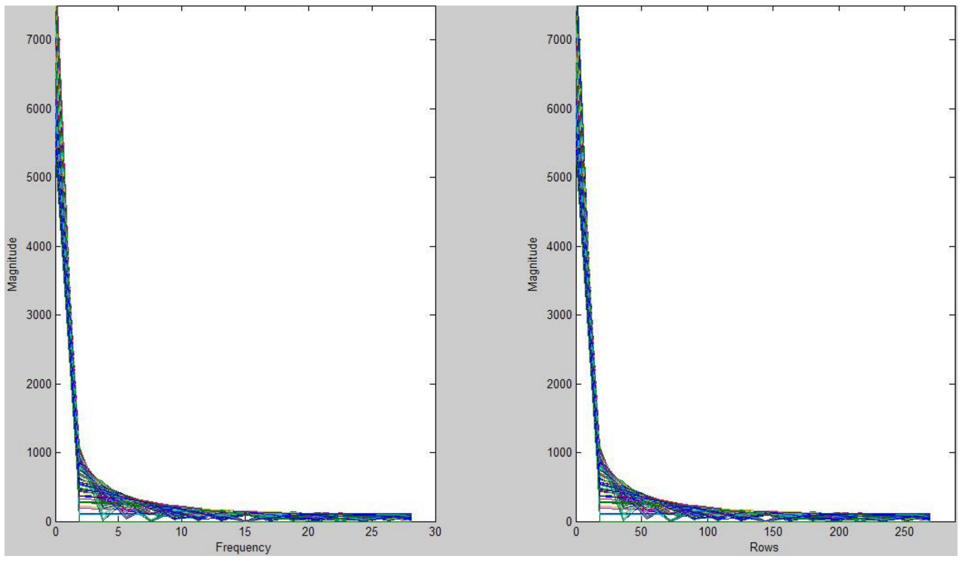
FFT results of Cartesian Image 1



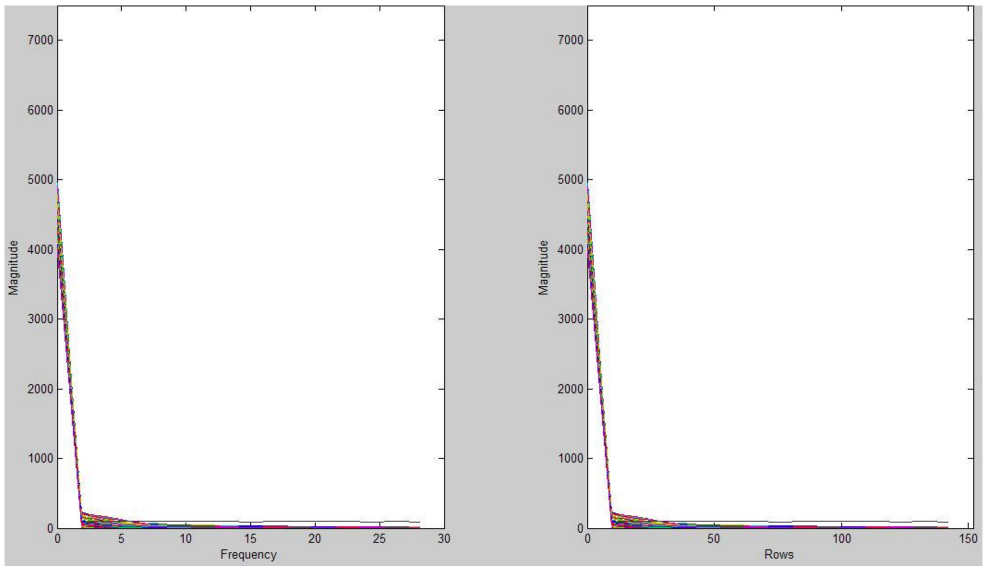
FFT results of Log-Polar Image 1

Cartesian Image 2


Log-Polar Image 2

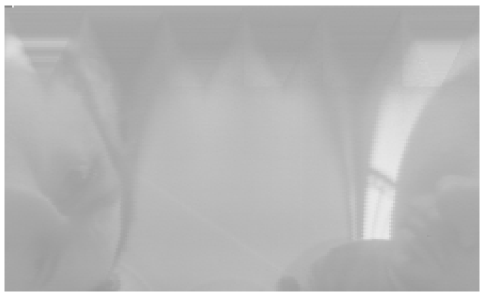


FFT results of Cartesian Image 2

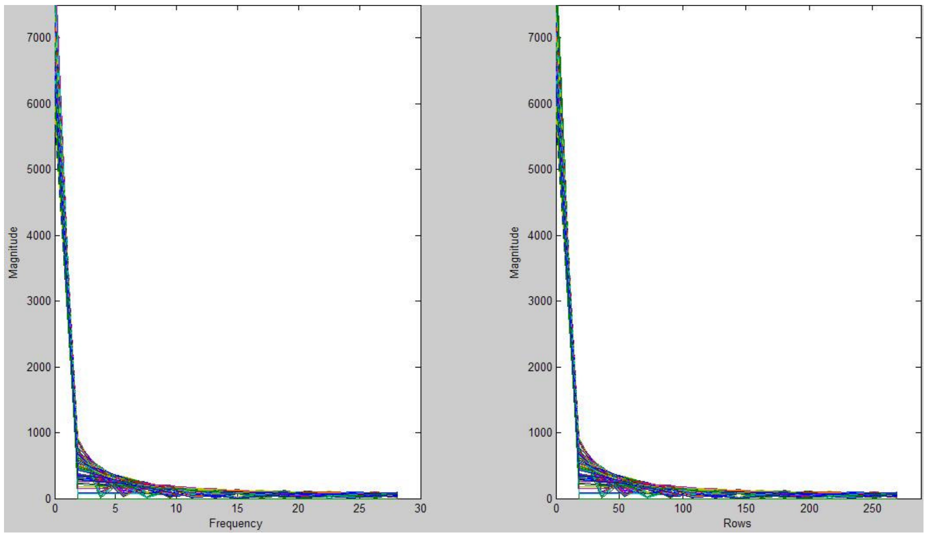


FFT results of Log-Polar Image 2

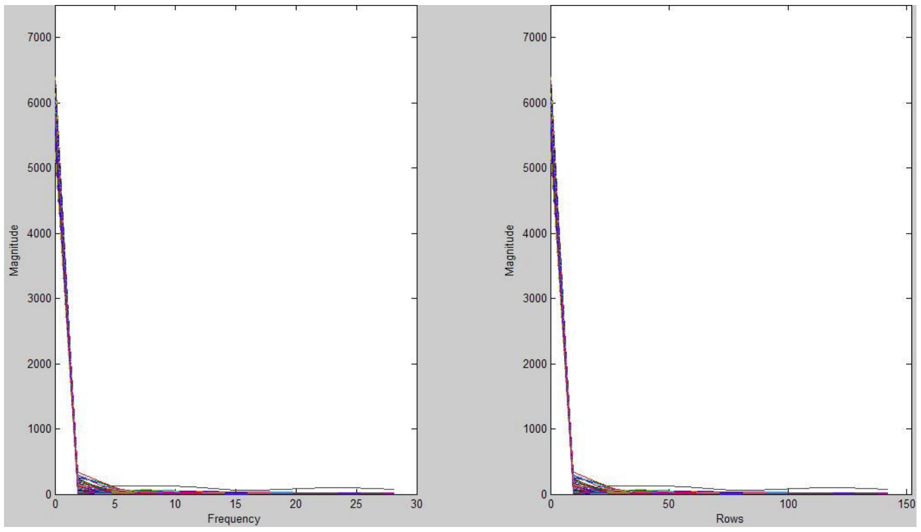
Cartesian Image 3



Log-Polar Image 3



FFT results of Cartesian Image 3



FFT results of Log-Polar Image 3

Tests on other images were also done, and the outcome was considered relatively similar to those seen here, thus not being presented.

Upon observing the graphics corresponding to the Magnitude to Frequency variation obtained with the software, it is verifiable that the magnitude of the signal in the cartesian images is much greater than the one of the corresponding log-polar images.

On the graphics concerning the Magnitude to Row variation, the fact that the magnitude decreases with the increase of frequency is verifiable. This means that the most important information of the images, either log-polar, either cartesians, is obtained in the first rows.

Both of the previously observed facts have to do with the characteristics imposed by the Log-Polar Transform of the cartesian images, which is the reduction of the information in the periphery of the picture.

It was also concluded that the extreme values of the magnitude in the first rows is due to the fact explained in the Log-Polar Transform Chapter: in the foveal region, the Log-Polar Transform is one pixel to many. This meaning that the information of a pixel will be duplicated in other pixels, thus incrementing the magnitude of the signal in these first rows.

12. Conclusion

This work proposed to study and observe the properties of the Fourier Transform applied on log-polar images taken with an omni-directional camera, and the differences between this type of image and the cartesian ones.

The omni-directional vision provides, as seen, a 360º vision, thus simplifying the attainment of the information about the location of the robot in the environment. In order to process this information in a fast and reliable way, it can be used the Fourier Transform properties on log-polar images, as opposed to the processing of full cartesian pictures.

This is demonstrated with the retrieved results from the software applied either to the cartesian, either to the log-polar images. As known, the Log-Polar Transform of a cartesian picture provides an equivalent log-polar picture, but with less resolution in its periphery, thus decreasing the amount of information to be processed by the Fourier Transform. Fact observed on the Magnitude to Frequency graphic, where the difference between the magnitude in the cartesian FFT results and the ones of the log-polar FFT is quite visible, being the magnitude of the latter smaller.

Also, from the graphics concerning the Magnitude to Row variation, it is attainable that the biggest, and therefore, most important amount of information is mainly in the first rows of the log-polar images, which correspond to the center of the cartesian images. Of this, it is deductible that it is possible to apply the LPT without loss of important information.

Concerning the Fourier Transform of the pictures, it was observed that it represented a fast and memory-saving method to compute information, as $N = 30$ is small but enough to obtain the amount of information needed, and that only the first 20 components of the signal were worthy to save in memory, this way reducing the memory size of the information.

It has been thus illustrated that the solution of using the Fourier Transform on log-polar images is a fast and trust-worthy solution for the problem of providing visual information for today's robotic navigational systems using omni-directional vision.

13. References

[1] O. A. Aider, T. Chateau, J. T. Lapresté, "*Indoor Autonomous Navigation Using Visual Memory an Pattern Tracking*"

[2] E. Menegatti, T. Maeda, H. Ishiguro, *"Image-based memory for robot navigation using properties of omnidirectional images"*

[3] D. Nikovski, "*Visual Memory-Based Learning for Mobile Robot Navigation*"

[4] A. Celentano, V. Di Lecce, *"A FFT based technique for image signature generation"*

[5] Taken from http://www.cs.drexel.edu/~ahicks/design/equiresolution.html

[6] K. Shindler, H. Bischof, *"The Epipolar Geometry of the Log-Polar Image Plane"*

[7] R. A. Peters II, M. Bishay, T. Rogers, "*On the Computation of the Log-polar Transform"*

[8] Taken from http://en.wikipedia.org/wiki/Fast_Fourier_transform

[9] Ardito, "*Telecamera Omnidirezionale con sensore retinico Cmos*"