UAV vision system: Application in electric line following and 3D reconstruction of associated terrain

Alexander Cerón Correa

Universidad Nacional de Colombia
Facultad de Ingeniería
Departamento de Ingeniería de Sistemas e Industrial
Bogotá, D.C.
2017

UAV vision system: Application in electric line following and 3D reconstruction of associated terrain

Alexander Cerón Correa

Thesis submitted in partial fulfillment to obtain the degree of
Doctor in Engineering - Systems Engineering and Computing

Advisor
Flavio Prieto, Ph.D.

Coadvisor
Iván Mondragón
Ph.D.

Universidad Nacional de Colombia
Facultad de Ingeniería
Departamento de Ingeniería de Sistemas e Industrial
Bogotá, D.C.
2017

Title in English

UAV vision system: Application in electric line following and 3D reconstruction of associated terrain

Título en español

Desarrollo de un sistema de visión monocular para UAVs: aplicación al seguimiento de líneas eléctricas y reconstrucción del terreno asociado

Abstract: In this work, a set of vision techniques applied to a UAV (Unmanned Aerial Vehicle) images is presented. The techniques are used to detect electrical lines and towers which are used in vision based navigation and for 3D associated terrain reconstruction. The developed work aims to be a previous stage for autonomous electrical infrastructure inspection. This work is divided in four stages: power line detection, transmission tower detection, UAV navigation and 3D reconstruction of associated terrain. In the first stage, a study of algorithms for line detection was performed. After that, an algorithm for line detection called CBS (Circle Based Search) which presented good results with azimuthal images was developed. This method offers a shorter response time in comparison with the Hough transform and the LSD (Line Segment Detector) algorithm, and a similar response to EDLines which is one of the fastest and most trustful algorithms for line detection. Given that most of the works related with line detection are focused in straight lines, an algorithm for catenary detection based on a concatenation process was developed. This algorithm was validated using real power line inspection images with catenaries. Additionally, in this work a tower detection method based on a feature descriptor with the capacity of detecting towers in times close to 100 ms was developed. Navigation over power lines by using UAVs requires a lot of tests because of the risk of failures and accidents. For this reason, a virtual environment for real time UAV simulation of visual navigation was developed by using ROS (Robot Operative System), which is open source. An onboard visual navigation system for UAV was also developed. This system allows the UAV to navigate following a power line in real sceneries by using the developed techniques. In the last part a 3D tower reconstruction that uses images obtained with UAVs is presented.

Resumen: Este trabajo presenta un conjunto de técnicas de visión aplicadas a imágenes adquiridas mediante UAVs (vehículos aéreos no tripulados). Las técnicas se usan para la detección de líneas y torres eléctricas las cuales son usadas en un proceso de navegación basada en vision y para la reconstrucción de terreno asociado en 3D. El proyecto está planteado como una fase previa a un proceso de inspección de infraestructura electrica. El trabajo se encuentra dividido en cuatro partes: la detección de líneas de transmisión eléctrica, la detección de torres de transmisión, la navegación de UAVs y la reconstrucción tridimensional de objetos tales como torres de transmisión. En primer lugar se realizó un estudio de los algoritmos para la detección de líneas en imágenes. Posteriormente se desarrolló un algoritmo para la detección de líneas llamado CBS (Búsqueda basada en círculos), el cual tiene buenos resultados en imágenes azimutales de líneas eléctricas. Este método ofrece un tiempo de respuesta más corto que la transformada de Houg o el algoritmo LSD (line segment detector), y un tiempo similar a EDLines el cual es uno de los algoritmos más rápidos y confiables para detectar líneas. Debido a que la mayoría de trabajos relacionados con detección de líneas se enfocan en líneas rectas, se desarrolló un algoritmo para detectar catenarias que cuenta con un proceso de concatenación de segmentos, esta

técnica fue validada con imágenes de catenarias obtenidas en inspecciones reales de infraestructura eléctrica. Adicionalmente se desarrolló un algoritmo basado en descriptores de características para la detección de torres de transmisión con la intención de facilitar los procesos de navegación e inspección. El proceso desarrollado ha permitido detectar torres en videos en tiempos cercanos a 100 ms. La navegación sobre líneas eléctricas mediante UAVs requiere una gran cantidad de pruebas debido al riesgo de fallos y accidentes, por esto se realizó un ambiente virtual para la simulación en tiempo real de técnicas de navegación basadas en características visuales haciendo uso del entorno de ROS (Robot Operative System), el cual es de código abierto. Se desarrollo un sistema de navegación a bordo de un UAV el cual permitio obtener resultados de navegación autónoma en el seguimiento de líneas en escenarios reales usando las técnicas desarrolladas. En la parte final del trabajo se realizó una reconstrucción 3D de torres electricas haciendo uso de imagenes adquiridas con UAVs.

Keywords: line detection, inspection, navigation, tower detection, onboard vision system, UAV.

Palabras clave: detección de lineas, detección de torres, inspeccion, navegación, systema de visión a bordo, UAV.

Acceptation Note

Thesis Work

Jury
Fabio Gonzales, Ph.D.

Jury
Julian Colorado, Ph.D.

Jury
Luis Mejias, Ph.D.

Advisor
Flavio Augusto Prieto, Ph.D.

Coadvisor
Iván Mondragon, Ph.D.

Bogotá, D.C., 2017

Dedicatory

To my beautiful sons Thomas and David who are the best of my life.

Acknowledgments

First, I wish to thank Professor Flavio Prieto for his mentoring through all these years and his endless patience. Also, I thank Iván for encouraging me through the thesis development. Of course I am grateful to the Computer Vision Group laboratory of Professor Pascual Campoy at the Universidad Politecnica de Madrid where I found valuable recommendations and interesting work during the internship I carried out with them. I thank the company Union Fenosa for providing some of the images of real inspection tasks.

I appreciate the collaboration of Professor Luis Mejias of the Queensland University for sharing a dataset of power line images.

I want to thank the people of the Mechatronics Laboratory at Universidad Nacional de Colombia in Bogotá, where I received interesting knowledge and many good friends.

I appreciate the help of Johan Agudelo for his collaboration and commitment to the last part of this work and of Carlos Terraza who helped me in the outdoors tests.

I want to thank to my mother and my sister Marlen for supporting me and feeding me. Also to Nestor Cerón for being my big brother, his recommendations on my manuscripts were very useful. Alejandra you give me support and care very well of our sons suffering my absences during these years of work.

Finally, I thank Universidad Militar Nueva Granada (UMNG) for partially supporting this work through the Grant for Advanced Studies.

Contents

List of Tables

List of Figures

*List of Algorithms

CHAPTER 1

Introduction

There exist different kind of unmanned vehicles: unmanned ground vehicles (UGV), autonomous underwater unmanned vehicles (AUV), autonomous surface vehicles (ASV) and unmanned aerial vehicles (UAV) among others. A special type of UAV is the micro aerial vehicle also known as MAV, which has the ability of a vertical take off and landing (VTOL) and also hovering [1].

UAVs have received increasing attention and have proliferated due to their low cost, advances in control theory, mechanical design and suitable power supply. This has propitiated the development of more stable systems compared with helicopters that consist on more than two propellers. Some of them are called quadcopters (composed of four motors and four propellers), hexacopters (provided with six motors and six propellers), and octocopters (eight motors).

Although the UAVs have been focused in military applications, the demand has risen in civilian applications that include search and rescue, emergency or inspection of power lines.

1.1 UAVs for power line inspection

Those benefits plus the reduction of risks and logistic problems have made UAVs a good alternative for several applications like power inspection of electric infrastructure; specially power line inspection, as they can replace the more expensive and risky inspection which is carried out manually or by manned helicopters [2]. In Figure 1.1 different kind of power line inspections are shown.

According to [3], this application depends on the perspectives of the electricity companies, the regulatory laws and the possibilities of trading for its implementation. The methods that use a manned helicopter present the following disadvantages:

- The helicopter has to fly as near as possible of the line (10 m), which is dangerous for the crew.
- It is necessary to avoid the impact in areas where low altitude flight may cause perturbations to people as well as farm or wild animals.

(a) Manual http://www.arganogroup.com

(b) Manned Flight http://www.airnor.com/electricas.html

(c) UAV http://www.aibotix.es

Figure 1.1: Different kinds of power line inspection process.

- The data obtained about the power line is sparse.
- The process is expensive.

These difficulties make evident the need of new devices like UAVs for line inspection. In [2], three aspects have been presented. 1) Strategies for risk administration in high power line corridors. 2) Selection of suitable platforms for sensor locations and 3) Data automation techniques for the vegetation. On the other hand it needs to mentioned that an important feature of some UAVs is vertical take off and landing VTOL [4].

There have been made a lot of developments in the area one of them is the design of a navigation system for an unmanned helicopter [5]. In the report of [6], a framework for power line inspection based on vision by using an UAV with a pan tilt camera is presented. In [7], an algorithm for power lines tracking by using GPS is proposed.

One of the main problems to resolve in the development of vision technologies is the image degradation as a consequence of the camera movement. Different approaches have been developed, for example the Jacobian, which is considered because it uses the kinematic of the generated path while the line is followed; the approach can be used to predict the generated degradation during movement. For example, in [8] a camera compensation with servo-visual control was used to avoid problems generated by the helicopter movement.

1.2 UAV vision systems

Vision systems have been used for UAV navigation in exterior environments to replace inertial sensors. This can be achieved with the use of one or two cameras or by sensorial fusion [9]. A problem that comes up with the development is the zero position due to

the impossibility of turning the actuators off when trying to keep the device in a fixed position.

The visual odometry uses features extracted from associated camera images in order to determine the position and orientation of a robot. This is very important to recover a path pose to pose and can be optimized over last n poses. An approach consists on a local sliding optimization with the aim to obtain a more accurate trajectory is presented in [10, 11].

A related approach is the simultaneous localization and mapping (SLAM), which intends to obtain a consistent global estimation of the robot trajectory using different types of sensors. When the sensor is a camera this process is called visual SLAM (VSLAM). SLAM implies to have an environment map updated. Because it is necessary to detect when the robot returns to a previously visited area, this is called loop closure. When a loop closure is detected, this information helps to reduce the sliding in the map and the camera path. Examples of different applications include the use of SURF (a feature descriptor that will be presented in the next chapter) in the SLAM for 3D mapping and positioning in outdoors environments for UAV as shown in [12].

In [13], the skyline detection with a catadioptric lens is used to stabilize the UAV flight. Additionally, a framework of see and avoid techniques and the mosaic generation is shown.

In the work of [14], the proposal of the referenced terrain vision-based navigation is presented. In this case a model for UAV position estimation based on Bayes criteria and homography are used.

An interesting work related to vision systems for UAV is presented in [9]. In this work, a sensorial fusion for micro helicopters is presented and the three following aspects are highlighted: 1) the weight of the payload devices for the UAV (particularly a MAV) is an important restriction; 2) the computational cost of the control algorithms must be minimal because the energetic availability is limited; 3) only one sensor does not provide enough information. Also, the importance of having a suitable combination of sensorial information to get a reliable state estimation is remarked.

1.3 Problem statement

Power lines are fundamental for maintaining the supply and distribution of electricity. Most of the work related to the inspection of electrical infrastructure based on image processing techniques is dedicated to power line detection. These techniques are based on edge and ridge detectors, magnitude and gradient orientation, voting schemes, line support regions, growing regions, chain codes, circle based search and graph cut models [15, 16, 17, 18, 19, 20]. Different sensors such as infrared [18] and LiDAR (Laser Imaging Detection and Ranging) have been used. Multi-source feature fusion information, such as LiDAR, color and texture, have been used for power line corridor inspection processes that include power lines and vegetation [21]. Although the use of a combination of different sensors, such as LiDAR or RGB-D, is very useful for object detection and recognition, the payload of an UAV is limited. For this reason, we were interested in exploring the capacity of 2D image techniques for object detection with normal cameras.

In power line environments, distinctive objects such as towers have to be detected properly in order to help the autonomous visual based navigation process. I concluded that it would be valuable to take advantage of 2D feature descriptors as an object description method since they can be obtained in the relevant parts of the image by using feature detectors and can be extracted in a short period of time. Several studies that evaluated the advantages of different 2D features using defined metrics [22] and matching applications [23] but I considered that it is also valuable to evaluate and use them in an object detection scheme.

Another aspect that demands attention in this kind of applications is image registration, since it is possible to compute the correspondences in a sequence of images captured with an UAV to have coverage of the interest area. In order to track the object of interest, a homography based on characteristic keypoints can be used as shown in [24].

Pylon and tower detection are important tasks in a power line inspection since it helps to determine the areas and perform an inspection of other elements such as isolators. This problem has been tackled with different methods, for example: using image segmentation together with priori knowledge and the Radon Transform for pylon detection as shown in [25] and HOG descriptors in combination with machine learning techniques [26] for tower detection. The development of new strategies that include other kind of descriptors for tower detection, tracking and 3D reconstruction are interesting topics in the areas of computer vision that are developed in this dissertation.

Autonomous visual based navigation for UAVs is a complex process that requires short computing times and accurate measurements in order to provide suitable and safe control commands to the device. It is clear that UAV navigation requires real time measurements to produce a response within a specified time (at least 100 ms), otherwise severe consequences, including failure, may affect the device. The evaluation of the performance of a visual based navigation system for power line following is presented in simulation in some works as shown in [27]. The evaluation with a real multirotor UAV as worked in this dissertation represents a major concern in the area of robotics.

1.4 Thesis proposal and objectives

The main objective of this thesis is the development of a vision system for an UAV that consists on a set of methods for line and tower detection, visual-based navigation, and the reconstruction of the associated environment. The developed methods are image and video processing techniques for detecting interest objects such as transmission lines and towers in order to facilitate the UAV navigation. In this process, 2D keypoint detectors and descriptors are used to find correspondences between frames and, finally, to obtain a 3D reconstruction. This thesis aims to improve the state of the art in this area by making contributions considering the following objectives:

- **To develop an algorithm for power line detection and a method for catenary detection.**
- **To develop an algorithm for tower detection by using line detection methods and 2D feature descriptors.**
- **To develop an UAV visual-based navigation system for power lines following.**

- **To develop a procedure for the 3D reconstruction of electrical infrastructure environments.**

1.5 Contributions

This thesis makes several contributions to the fields of computer vision and robotics.

- An algorithm for line detection using a circle-based search (CBS) from aerial images. This algorithm takes into account the geometric symmetries of the circle for searching valid points. Also, it uses features of the lines such as: symmetry, length, proximity, and concurrence for linking segments detected in order to obtain longer lines. The algorithm presents very good results with images of power lines. The computing time competes with state of the art algorithms (see video in `https://youtu.be/RsKAst6o-0o`).
- An algorithm for catenary detection in images. This algorithm uses the line properties in order to convert a set of unconnected segments into a polyline, rejecting the unconnected and isolated elements. The process was evaluated with images from an inspection of electrical infrastructure.
- A new method for object detection that is able to detect electrical towers. The method includes a line detection stage. The algorithm is based on the concept of a grid of 2D descriptors. The used descriptors are useful not only for detection but also for tracking the interest region. The results shown very good performance with different videos of electrical towers (see video in `https://youtu.be/D3elLJ8656E` and video in `https://youtu.be/0RvdW0Bd7Ag`).
- Strategies for UAV navigation in power line environments. We developed two strategies for UAV navigation: The first is based on power line detection. (see video in `https://youtu.be/eIR3h8L0yAs`). The second is based on tower detection. In both cases a 3D simulation was done by using virtual environments (see video in `https://youtu.be/Xe9VZo41YAY`).
- An onboard system for visual-based navigation. The system uses the CBS algorithm for power line detection. It also uses a Histogram of Oriented Segments that helps to obtain the orientation of power lines. The results shown how the onboard visual-based navigation system enables an UAV to perform an autonomous flight for power line following (see video in `https://youtu.be/c60UfQ7FKyI`).

  The datalogger of an autonomous mission is shown in the following video (see `https://youtu.be/pa4fqoGJmDU`).

During this thesis several papers have been submitted and published in peer reviewed conferences and journals:

- Journal Papers:
  - Alexander Ceron, Flavio Prieto, and Ivan Mondragon. “Real time transmission tower detection from video based on a feature descriptor”. Accepted in the Journal IET Computer Vision (July 25, 2016) (Available online: 27 July 2016 `http://ietdl.org/t/v5Ngtb`).

- Johan Agudelo, Alexander Cerón and Flavio Prieto. "Evaluation of 2D descriptors for 3D reconstruction of electrical infrastructure". Submitted to the International Journal of Computational Vision and Robotics (IJCVR). (September 30, 2016).
- Conference papers:
  * Alexander Ceron and Flavio Prieto. "Evaluating and Comparing of 3D Shape Descriptors for Object Recognition". 9th International Symposium on Visual Computing, ISVC 2013, Rethymnon, Crete, Greece, July 29-31, 2013, Lecture Notes in Computer Science (LNCS) series, Springer Verlang, Vol 8034, pp 484-492. 2013 [28].
  * Alexander Ceron, Flavio Prieto, and Ivan Mondragon. "Power line detection using a circle based search with UAV images". In 2014 International Conference on Unmanned Aircraft Systems (ICUAS), 2014 [19].
  * Alexander Ceron, Flavio Prieto, and Ivan Mondragon. "Towards Visual-Based Navigation with Power Line Detection". 10th International Symposium on Visual Computing, ISVC 2014, Las Vegas, NV, USA, December 8-10, 2014, Lecture Notes in Computer Science (LNCS) series, Springer Verlang, Vol 8887, pp 827-836. 2014 [29].
  * Alexander Ceron, Flavio Prieto, and Ivan Mondragon. "Visual-based navigation for power line inspection by using virtual environments". Proc. SPIE 9406, Intelligent Robots and Computer Vision XXXII: Algorithms and Techniques, 94060J (8 February 2015) [30].

1.6 Dissertation outline

Chapter 2, presents state of the art on feature extraction methods. Some of the methods are used into the developed techniques in the context of power line and tower detection as shown in the following chapters.

Chapter 3, presents a new method developed as part of this research for line detection that is applied to power lines. The method consists on a circle-based search using the geometrical symmetry for detecting lines.

Chapter 4, presents a new method developed as part of this research for tower detection based on a grid of descriptors that uses line detection methods and 2D feature descriptors.

Chapter 5, presents strategies for UAV navigation in power lines by using virtual environments developed as part of this research.

Chapter 6, presents an onboard vision-based navigation system that use the proposed line detection method to navigate in real power line environments.

Chapter 7, presents the 3D reconstruction of electrical infrastructure and associated terrain using Structure From Motion (SFM). In this chapter different 2D descriptors were evaluated towards to reduce the computing time in 3D reconstruction.

Chapter 8, presents the conclusions and new perspectives open with this work.

CHAPTER 2

Feature extraction in 2D images

2D descriptors encode important information in regions or distinctive points of the images which have the properties of being invariant regarding to change of scale, orientation and contrast. These descriptors have become essential to object detection, tracking and 3D reconstruction processes.

In turn, such descriptors are complemented by other methods that are able to detect higher level information such as linear segments in images, which is very useful in the image understanding process. In the context of electrical infrastructure, the images are composed by plenty of lines and regions with a lot of distinctive points where the 2D descriptors are very important.

The first step to obtain 2D descriptors is a preprocessing where the image is segmented in order to obtain the image edges with techniques such as Laplacian, Prewit and Canny among others.

Other techniques focused in finding the more distinctive points in the images which are know as detectors, such a Harris (that is used for corner detection), FAST and AGAST are used to obtain the locations where the 2D feature descriptors have to be computed. In some cases, the keypoint detection is part of the 2D feature descriptor algorithm.

2.1 2D feature descriptors

2D descriptors encode important information of distinctive parts of an image. For this reason, in this project, they are considered as a primordial stage in different activities that include the object detection process, tracking and recognition. In chapter 4, the use of 2D descriptors in an object detection scheme is presented. In this chapter, 2D descriptors that will be further employed, in chapter 7, in the process of 3D reconstruction are presented.

An important task that is present inside UAV vision systems related with processes like: the navigation, object detection and reconstruction is feature extraction. 2D descriptors have several uses: 1) find distinctive features in one or more images to allow a UAV to localize in an environment, 2) detect important objects in a scene, 3) detect and evaluate feature points in order to find correspondences inside a 3D reconstruction process.

A good descriptor must be able to represent information efficiently in keypoints (feature points) or important areas of images. This information has to be relevant and invariant since it is used to establish correspondences among feature points in different images with the purpose of inferring the depth of the scene which is necessary in the object detection and 3D reconstruction process. This type of task make necessary the use of computer vision techniques.

Although there are different kind of 2D descriptors that define points of interest present at the scene and offer good performance in several tasks, they are not commonly suitable for closing the loop in real time systems since the time of response is not enough to create a realistic mesh in a 3D reconstruction process.

One of the more widely used descriptors in different contexts due to its representation capacity and invariance to different types of transformations is SIFT (Scale-Invariant Feature Transform) [31]. Other important descriptors are SURF (Speeded Up Robust Features) [32], BRIEF (Binary Robust Independent Elementary Features) [33], ORB (Oriented Fast and Rotated BRIEF) [34], BRISK (Binary Robust Invariant Scalable Keypoints) [35] and FREAK (Fast Retina Keypoint) [36] which have shown good performance.

2.1.1 SIFT

This is a method for distinctive image invariant feature extraction that can be used for matching between different views of a scene [31]. These features are invariant to scale and rotation and have shown robustness in images with distorted affine point of view changes, noise addition and change of illumination.

The SIFT algorithm is composed of the following stages:

1. Detection of extremal changes in scale-space. Here, it is necessary to search over different scales and localizations of the image. These points have to be invariant to scale and orientation. This process is easily implemented using the difference (D) between images (I) with Gaussian filter (G), for key point identification. This stage generates a space of Gaussian differences.

$$G(x, y, \sigma) = \frac{1}{2\pi\sigma^2} e^{\frac{-(x^2+y^2)}{2\sigma^2}}. \tag{2.1}$$

$$L(x, y, \sigma) = G(x, y, \sigma) * I(x, y). \tag{2.2}$$

$$D(x, y, \sigma) = L(x, y, k\sigma) - L(x, y, \sigma). \tag{2.3}$$

After this, the process consist on comparing each pixel with eight neighbours in the same scale, nine in the upper scale and nine in the lower scale, depending on whether the value of the pixel is lower or greater than the others. The selected pixel becomes an interest point.

2. Stable points location. In this step two operations have to be performed. The first is to compare the contrast of the selected pixels with a threshold of defined contrast. If the contrast is lower than the threshold, this point is discarded. The second operation consists on eliminating the edge responses, since they present a big curvature along the edges, and a small curvature in the perpendicular direction. This compute is performed using a 2×2 Hessian matrix:

$$H = \begin{pmatrix} D_{xx} & D_{xy} \\ D_{xy} & D_{yy} \end{pmatrix}.$$

After that, the trace $Tr(H)$ and the determinant $Det(H)$ of this matrix are computed to compare this ratio with the principal curvature ratio considering a threshold r, such as:

$$\frac{Tr(H)^2}{Det(H)} < \frac{(r+1)^2}{r}. \tag{2.4}$$

If the Equation 2.4 is fulfilled, the point is kept, otherwise is discarded.

3. Orientation assignation. In this step, one or more orientations are assigned to each localization of a keypoint based on the local gradient directions of the image.

All the future operations are performed over a database that has been relatively transformed into an assigned orientation, scale and localization for each feature in order to provide invariance to this transformation. To determine this orientation, Gaussian pyramidal images are used for obtaining a histogram (with 36 values) around a region of an interest point.

The magnitude $m(x, y)$ and the orientation $\theta(x, y)$ have to be obtained as:

$$m(x, y) = \sqrt{D_x^2 + D_y^2}. \tag{2.5}$$

$$\theta(x, y) = tan^{-1}\left(\frac{D_y}{D_x}\right). \tag{2.6}$$

4. Descriptor extraction. In this last stage the algorithm constructs a representation of interest points in a region of pixels in its local neighbourhood. The first step is to perform a sampling of magnitude and gradient orientations in a specific area around an interest point. After that, the descriptor is rotated in order to avoid variations in the coordinate orientation. A set of smoothed oriented histograms are created to capture the important aspects of the area around the interest point. These histograms, each one with 8 directions, are stored in a 4×4 array. Finally, this vector of 128 elements ($4 \times 4 \times 8$), is normalized to a range of 0 to 1 values and is thresholded to discard elements with very low values.

2.1.2 SURF

This is a detector and descriptor invariant to rotation and scale. According to [32], this is similar to related descriptors but it has the advantage of improving its velocity. The

reference mentions some cases where the image rotation is not present such as a modified version called upright SURF (U-SURF) that is only invariant to scale which is used in cases such as mobile robot navigation where the camera remains horizontal. It is important to note that the detector does not use color. This detector is based on the Hessian matrix and presents a good performance in computing time and accuracy; the location and scale are established using the determinant. The SURF descriptor is based on similar properties as SIFT but with a reduced complexity. Unlike SIFT [31], that uses Gaussian filters for second order derivative approximations, in this case box filters with different resolutions $(9\times9, 15\times15, 21\times21, 27\times27, etc)$ are used. They can be evaluated very fast independently of its size by using integral images. An integral image $I_\Sigma(\mathbf{x})$ in a position $\mathbf{x} = (x, y)$ represents the sum of all the pixels in the input image I of a rectangular region defined by the point $\mathbf{x}$ and the origin, $I_\Sigma(\mathbf{x}) = \Sigma_{i=0}^{i<x}\Sigma_{j=0}^{j<y} I(i, j)$. With the compute of I_Σ done, four additions are enough to calculate the sum of intensities over any rectangular top-right region.

The SURF algorithm is composed of the following elements:

- Orientation assignation: With the purpose of being rotation-invariant, reproducible orientations are identified for the interest points.

  For this purpose, responses wavelet of Haar in x and y directions were computed in a circular neighbourhood of radius $6s$ around the interest point; being s the scale in which an interest point was detected. Also, the sampling step is dependent of the scale and selected to be s. According to this, with big scales the wavelet size is big. Therefore, integral images and fast filtering is used. Only six operations are required to compute the response in the direction x and y for any scale. The length of the wavelet is $4s$. Once the wavelet responses are computed and weighted with a centered Gaussian filter in the interest point, they are presented as space vectors with the horizontal response intensity over the abscissa and the intensity of the vertical response over the ordinate. The dominant orientation is estimated by computing the sum of all the responses with a sliding orientation window that cover an angle of $\frac{\pi}{3}$. The horizontal and vertical responses inside a window are summed. This produces a new vector. The longer vector gives its orientation to the interest point. The size of a sliding window is a parameter that can be obtained experimentally. Small sizes point to dominant wavelet responses, big sizes produces maxims in the length of the vector.

- Descriptor components. For the descriptor extraction, the first step consists on constructing a squared region centered around the interest point and oriented along the selected orientation. This region is regularly divided in subregions of 4×4. This keeps important spacial information. For each subregion a few features in points regularly spaced of 5×5 are computed. Being d_x and d_y defined as the Haar wavelet responses in the horizontal and vertical directions respectively. For information extraction over the polarity of the intensity changes the sum of the absolute values of the responses $|d_x|$ and $|d_y|$ is obtained. Each subregion has a vector descriptor of four dimensions in its structure of intensity $v = (\sum d_x, \sum d_y, \sum |d_x|, \sum |d_y|)$.This results in a vector descriptor for all the subregions of length 64, which is a standard size of SURF. A final variation called SURF 128 can be obtained with this descriptor when summatories of dx and $|d_x|$ are divided in two, according with the sign of d_y. This same division is computed for dy and $|d_y|$, depending on the sign of dx.

2.1.3 BRIEF

Binary chain is proposed as an efficient descriptor of feature points. This is based on the fact that a patch of images can be classified over a relatively small number of comparisons of intensity. For this task, a vector of bits to store the result of a classifier test is created. This is computed to smooth the patch of the image. More specifically, the test τ is defined over a patch $\mathbf{p}$ of size $S \times S$ as:

$$\tau(\mathbf{p};\mathbf{x},\mathbf{y}) = \begin{cases} 1 & \text{if } \mathbf{p}(\mathbf{x}) < \mathbf{p}(\mathbf{y}); \\ 0 & \text{in other case.} \end{cases} \tag{2.7}$$

The pixels position are preselected randomly according with a Gaussian distribution around the image path center. Where $\mathbf{p(x)}$ is the intensity of the pixels in a smoothed version of $\mathbf{p}$ in $\mathbf{x} = (u, v)^T$. Choosing a set of $n_d(\mathbf{x},\mathbf{y})$ pairs of locations, a set of binary test are defined. The BRIEF descriptor is taken as the bits n_d-dimensional chain.

$$f_{n_d}(\mathbf{p}) := \sum_{1 \leq i \leq n_d} 2^{i-1} \tau(\mathbf{p}; \mathbf{x}_i, \mathbf{y}_i). \tag{2.8}$$

In the work of [33] $n_d = 128, 256$ and 512 are considered, showing the performance in terms of velocity, efficiency and recognition rate. It is good to mention that this descriptor is not robust to orientation changes and this reduces its capacity in cases of recognition.

2.1.4 ORB

This descriptor is based on BRIEF and is defined as ORB (Oriented FAST and Rotated BRIEF). It uses elements of the key point detector FAST (Features from Accelerated Segment Test) [37], a keypoint detector based in machine learning techniques.

The main contributions of ORB are:

- The addition of an orientation component as FAST. This is denominated oFAST (FAST Keypoint Orientation), which gives an estimation of point orientation.
- The efficient compute of BRIEF as an oriented feature.
- A learning method for the de-correlation of BRIEF features under rotation invariance has good performance in closest neighbour applications.

The orientation measure evaluates the intensity centroid. It is assumed that the intensity of a corner is a vector located in the center that can be used to determine the orientation. ORB uses the moments defined by Rosin [38].

$$m_{pq} = \sum_{x,y} x^p y^p I(x, y), \tag{2.9}$$

With this moment, it is possible to obtain a centroid:

$$C = \left(\frac{m_{10}}{m_{00}}, \frac{m_{01}}{m_{00}} \right) \tag{2.10}$$

The corner orientation can be computed with respect to this center as following:

$$\theta = atan2\left(m_{01}, m_{10}\right) \tag{2.11}$$

2.1.5 BRISK

This method is for detection, description and correspondence of feature points: It takes less computational cost than SURF [35]. Although this method can be used with FAST detector, it can be used with AGAST (Adaptive and Generic Accelerated Segment Test) [39], an improved version of FAST. This calculates interest points through different sub-sampled versions of the original image. AGAST has not to be trained for an specific scene, since it is dynamically adapted to the image. According to its author this method is modular and allows BRISK to be combined with any other descriptor and conversely. This method is composed of the following stages:

1. Detection of features in the scale space. The BRISK detector estimate the true scale for each one of the keypoints in the space scale. Unlike other approaches that discretize the scale axis in thickness as is the case of SURF [32], by computing a fast Hessian, the BRISK detector estimates the correct scale of each feature point in the continuous scale space.

   For this approach, the spacial layers of the pyramidal scale are composed of n octaves c_i and n intra-octaves d_i for $i = \{0, 1, ..., n-1\}$ and, typically, $n = 4$. The octaves are created by sub-sampling of the original image c_0. Each intra-octaves d_i is located between layers c_i y c_{i+1}. The first intra-octave d_0 is obtained by subsampling of the original image c_0, by a 1.5 factor, the rest of intra-octave-layers are derivated by means of successive subsampling. An entropy-based interest measure selects points on surfaces that exhibit strong local variation in surface orientation. Therefore, if t denote the scale then $t(c_i) = 2^i$ and $t(d_i) = 2^i \cdot 1.5$.

   In this descriptor, a mask of 9-16 pixels, which requires at least 9 consecutive pixels in a circle of 16 pixels is used, in order to satisfy the criteria of FAST detector.

   Initially, the FAST 9-16 detector is applied in each octave and intra-octave separately using the same threshold T in order to identify a region of potential interest. After that, the points that belong to those regions pass through a maximal non suppression in the scale space. The point have to accomplish the maximal condition in respect to the FAST scores s of its eight neighbours in the same layer.

2. Feature point description. Due to a set of feature points (which consists of refine locations of the image of sub-pixels and scale values in float point), the BRISK descriptor is a binary chain created by simple results of concatenation of brightness comparison. In BRISK, the feature direction for each feature point is identified in order to obtain normalized orientation descriptors and achieve a rotation invariance.

3. Correspondence of the descriptor. The correspondence between two BRISK descriptors is performed by using the Hamming distance as with BRIEF [33]. This process can be summarized as an XOR operation followed by a bits count. The BRISK descriptor has an efficient performance in actual architectures.

2.1.6 FREAK

This descriptor is inspired in the retina of the human visual system [36]. A cascade of binary chains is computed by efficient intensity image comparisons over a retinal pattern. Its structure is similar to BRISK since it consists on a concentric pattern, but it differs in the level of concentration in the central part with some overlaps.

This descriptor F is built by threshold difference between pairs of receptive layers with its Gaussian corresponding kernels. F is a bits chain composed by a sequence of difference of Gaussian (DoG) of a bit:

$$F = \sum_{0 \leq a < N} 2^a T(P_a), \tag{2.12}$$

where P_a is a pair of receptive fields, N is the desired size of the descriptor and

$$T(P_a) = \begin{cases} 1 & \text{if } (I(P_a^{r_1}) - I(P_a^{r_2}) > 0) \text{ ;} \\ 0 & \text{in other case.} \end{cases} \tag{2.13}$$

($I(P_a^{r_1})$ is the smoothed intensity of the first receptive field of the pair P_a.

2.2 Line detection methods

Different techniques can be used for detecting lines, circles and other geometric primitives, such as the Hough transform. Since the electrical infrastructure consists of many linear segments, this section is focused in different methods for line detection.

2.2.1 Line Segment Detector (LSD)

This algorithm was proposed by Grompone in [40]. It is composed of two parts: an edge detection by gradients and a growing regions procedure. Although it offers a superior detection performance than Hough Transform its computing time is bigger when the image have a lot of noise or different textures. An advantage of this method in a false alarms detection stage.

The LSD method is based on the work of [41], which consists of the following steps:

1. Subdivide the image in line support regions and gather connected pixels that share the same gradient angle with a given tolerance.
2. Find the line segment that presents the best approximation to each region of the support line.
3. Validate each line segmented based in the region information.

In LSD, step 1 was replaced with the algorithm for growing regions as shown in 2.1

The algorithm in LSD improves the work of [41] by introducing the use of growing regions, a rectangular pattern and a validation on the number of false alarms.

Each region begins with a pixel and a region angle. Neighbouring pixels in the region are evaluated and if they have a superior value than a threshold they are added to the region. In each iteration, the region angle is updated to an approximate orientation level which is defined as following:

$$arctan\left(\frac{\sum_i sen(ang_i)}{\sum_i cos(ang_i)}\right). \tag{2.14}$$

Algorithm 2.1: Region_grow

Input : An image I; a starting pixel (x, y); an angle of tolerance τ; an image *Status* where pixels used by other regions are marked.

Output: A list *Region* of pixels.

```
Region ← (x, y)
θ_region ← LevelLineAngle(x,y)
S_x ← cos(θ_region)
S_y ← sin(θ_region)
foreach pixel P ∈ Region do
    foreach pixel P̄ a neigbor of P and Status(P̄) ≠ a Used do
        if Diff(LevelLineAngle(P̄), θ_region < τ then
            Add P̄ to Region
            Status(P̄) ← Used
            S_x ← S_x + cos(LevelLineAngle(P̄))
            S_y ← S_y + sin(LevelLineAngle(P̄))
            θ_region ← arctan(S_y/S_x)
        end
    end
end
```

Algorithm 2.2 presents the complete process.

2.2.2 Edge Drawing Lines (EDLines)

This is one of the best performance methods and it is based in the use of filter combinations and a painting algorithm to join line segments [42]. An advantage of this technique is that it is faster than LSD.

This method is composed of edge extraction (by using the Edge Drawing algorithm), segment detection and line validation.

2.2.2.1 Edge Drawing

1. The image has to pass through a filter in order to do noise suppression and image smoothing. A Gaussian kernel of 5×5 with $\sigma = 1$ is used.

2. The magnitude of the gradient and the direction of each pixel in the smooth image have to be obtained. Prewith, Sobel or Scharr can be used in this stage.

3. A set of pixels that has high probability of belonging to an edge are called anchorage. They have to be computed first. The anchorage points correspond to a pixels where

Algorithm 2.2: LSD: Line Segment Detector

Input : An image I, parameters ρ, τ and ϵ; an angle of tolerance τ; an image *Status* where pixels used by other regions are marked.

Output: A list *out* of rectangles.

```
(LLAngles, GradMod, OrderedListPixels) ← Grad(I, ρ)
Status(allpixels) ← NotUsed
foreach pixel P ∈ OrderedListPixels do
    if Status(P) = NotUsed then
        region ← Region_Grow(ρ, τ, Status)
        rect ← RectAprox(P, τ, Status)
        nfa ← NFA(rect)
        nfa ← ImproveRect(rect)
        if nfa < ε then
            Add recto to out
            Status(region) ← Used
        else
            Status(region) ← NotIni
        end
    end
end
```

the gradient operator produces maximums values. They are the peaks of the gradient map.

4. Finally, the anchorages are connected, drawing the edges between them.

2.2.2.2 Line segment detection

Given a continue sequence of chains of pixel borders, the goal is to divide this chain into one or more rectilinear lines. The basic idea is to walk over pixels in this sequence and adjust the lines using least squares within a specified error value.

2.2.2.3 Line validation

A binomial distribution is used to model the number of false alarms which are not necessary part of the line.

2.3 Conclusions

A review of line detection methods has been presented. These methods simplify the image in its relevant parts. This can increase the efficiency of an object detection system focused in linear elements. For these reasons, chapter 3 presents the use of these methods for power line and catenary detection. Also, in chapter 4, line detection methods are used in tower detection.

In this chapter, several 2D descriptor are presented. Their usage in the area of object detection is discuses in the chapter 4. An important application of 2D descriptor is in 3D reconstruction; this is discussed in chapter 7.

CHAPTER 3

Power line and catenary detection

A new method for power line detection based on computer graphics algorithms is presented. The algorithm uses geometric relationships that are inherent to the circle symmetry. The method detects line segments that are linked in a posterior stage. For the segmentation stage, we use Canny and Steerable Filters. We developed two tests for validating the proposed approach. The first one uses synthetic images and the second one real power line images taken from UAVs. The results show not only the high efficiently of this method for line detection, but also the short time it takes compared with state of the art algorithms. The proposed algorithm has the benefit of not using GPU.

Additionally, a method for catenary detection is developed from a study of the capacity of different line detection methods. This is based on a segment concatenation.

3.1 Related Work

There exist different methods for line detection that have been used in power line detection from UAV images. They are based on edge and ridge detectors, magnitude and gradient orientation [43, 42], voting schemes [15, 44], line support regions [41], growing regions and chain codes [18].

The typical procedure for line detection is to do a segmentation prior to the application of line detection. A good method for edge detection is the Canny Filter [45] which is commonly used together with a different line detection method. The classical method for line detection is the Hough transform [15, 44] which can detect lines in well contrasted and segmented images. This method was used in combination with a PCNN (Pulse-Coupled Neural Network) for removing background and clustering for power line detection [46]. In the work of Zhang [47], a process for power line detection based on Hough transform with Kalman filter is presented; they used the Otsu threshold method obtaining better results than with PCNN filters.

There are other approaches for line detection based on gradient extraction stage used to find the more relevant changes present in images. Usually, these kind of approaches use a filter like Prewitt, Sobel or Scharr.

More accurate and faster methods than the Hough transform have been developed. In the work of Burns [41], a straight line detection method based in gradient information and line support regions is presented.

Another relevant work is the Line Segment Detector (LSD) that uses growing regions and a method for the consideration of false alarms [40] that can detect lines with a high level of accuracy but it takes more computing time than Hough.

One of the best line detection methods, is the Edge Drawing Lines (EDLines) [42]. This method is based on Edge Drawing (ED) [48], which is used for fast edge detection and least squares line fitting. The results show that EDLines is faster than LSD with similar results in terms of lines detected.

Methods for line detection based on other kind of filters have also been developed. For example, the steerable filters have been used in order to obtain ridge detection. They are very useful for detecting power lines from UAV images [16]. According to these authors, the process take less time that LSD and EDLines but it requires GPU processing [17].

It is good to mention that some works of line detection have a post processing stage (see Figure 3.1) for connecting line segments or cluster the interest lines [47, 16]. Recently, a method for line detection based in region growing, ridge filters and chain codes was developed in [18] but the results in relation with processing time were not presented. Although many algorithms for drawing of lines and circles have been developed such as Digital Differential Analyzer (DDA), Bresenham Line and mid-point [49], these are not related with the process of line detection.

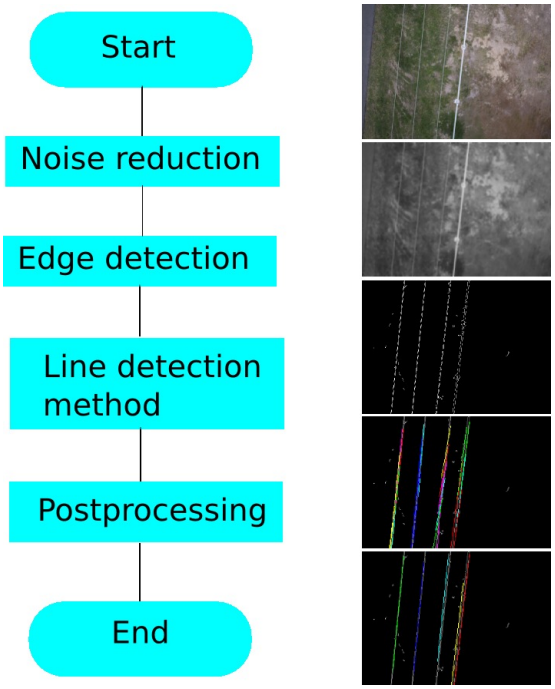


Figure 3.1: Power line detection toolchain.

In addition, there exist some works for Simultaneous Location and Mapping (SLAM) based on line feature detection [50]. In this case they use a constrained Hough Transform for line detection and an Extended Kalman Filter (EKF) to estimate the line position

accurately. Other interesting work is a SLAM based on a Rao- Blackwellized particle filter (RBPF) and an iterative end point fitting (IEPF) for line extraction. [51].

Most of the mentioned works related to power line detection in images, detect power lines as a linear segment but it is clear that all power lines present a curved trend due to gravity. For this reason it is important to detect catenaries in order to recognize more general power line environments, as shown in [20].

This section is divided in two parts: Firstly, a method for power line detection based on the search of lines between two opposite points is proposed. This is done by using computer graphic primitives which are efficient for drawing circles. This method is different from previous ones since it is based on geometric relationships and does not use gradient information. We call this method Circle Based Search (CBS). In addition, we add a procedure for connecting contiguous segments in order to detect longer lines. Secondly, a process for detecting catenaries in images of power line environments is presented. The process is able to store each line or catenary detected as a polyline in a data structure. The polylines obtained are processed in order to select the most suitable catenaries. The process is validated using real images with catenaries taken from different perspectives.

3.2 Preprocesing

3.2.1 Canny Filter

The Canny filter has been used for edge detection in different applications of machine vision [45]. It has been used in the power line detection process in [46, 47]. The main advantage is that it takes a short computing time during segmenting. A disadvantage of this method is that when it is used for detecting thin elements like a power line, it may detect two edges in both sides of the power line.

3.2.2 Steerable Filter

Steerable filters were developed by Freeman [52]. These filters allow the use of rotated versions of filters based on Gaussian functions. The purpose of these filters is to obtain more energy points by applying them with different angles [16].

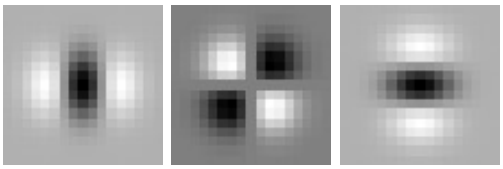
The second derivative of a Gaussian kernel is used.

$$G(x, y) = e^{\frac{x^2+y^2}{2\sigma^2}}. \tag{3.1}$$

This is a two-dimensional circular symmetric Gaussian function in Cartesian coordinates x and y, where σ is the standard deviation which depends on the line's width [53].

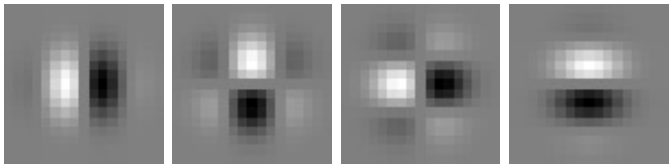
The second derivatives (of $G(x, y)$), $G_{2i}(\theta)$, are computed for three rotations θ to obtain a basis ($1 \leq i \leq 3$) in order to compute the filter rotation in every angle by using an interpolation [52] (see Figure 3.2). For the quadrature filter, the Hilbert transform ($H(x, y)$) of the Gaussian derivatives is used; see Figure 3.3.

By using steerable filters, it is possible to obtain the oriented energy and ridge energy in each pixel of the image. This has been used for segmented power lines in noisy images



(a) Ga. (b) Gb. (c) Gc.

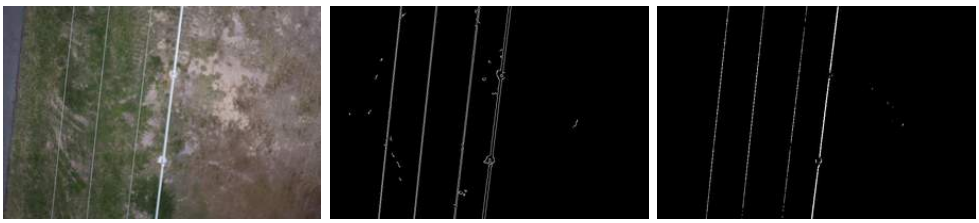
Figure 3.2: Base filters G_2.



(a) Ha. (b) Hb. (c) Hc. (d) Hd.

Figure 3.3: Base filters H_2.

[16];In Figure 3.4 the result of line segmentation with steerable filters is better than using Canny.



(a) Original. (b) Segmented with Canny. (c) Segmented with Steerable Filters.

Figure 3.4: Filters for power line segmentation.

3.3 Circle based search (CBS)

Most of the image processing approaches for line detection are based on convolution kernels (filtering) and neighbour operations (connected components). Nevertheless, in order to develop a line detection approach based on a local search, it may be useful to perform an extended search rather than an eight pixel neighbourhood search. This search may begin from a pixel that belongs to a line. Computer graphic algorithms that are commonly used for primitive efficient drawing are modified to perform extended comparisons in bigger neighbourhoods, and to validate the error in detected lines. A raster circle algorithm is used to search for valid points that belong to a line in segmented images based on the circle symmetry. The Bresenham algorithm [49] is employed to create the positions of the circle contour in pixels, in Figure 3.5 different examples of a circle contours are shown. This method is based on integer math.

Efficient raster circle drawing algorithms are based on their inherent octant symmetry (see Figure 3.6). These allow to paint a whole circle by rounding an octant. For example,

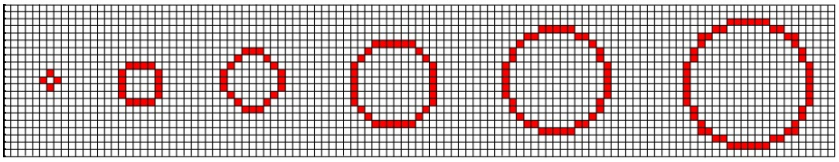


Figure 3.5: Raster circles of different sizes.

from $(0, r)$, to $(\frac{\sqrt{2}}{2}r, \frac{\sqrt{2}}{2}r)$ each point (a, b) is converted to eight points by reflecting in the principal axes and in the diagonals ($y = x$ and $y = -x$).

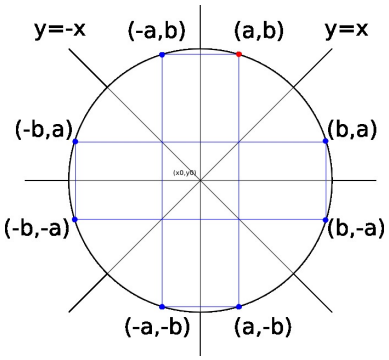


Figure 3.6: Octant circle symmetry.

Additionally, this symmetry can be used to detect if two opposite points belong to a line from a defined center (xc, yc). The algorithm in first instance test for all possible orientations if all the possible symmetry pairs $(xc + x, yc + y)$ and $(xc - x, yc - y)$ are different than background in a segmented image. In each test a line is searched and the orientation with more points is selected.

In Figure 3.7, different synthetic lines are shown, and in Figure 3.8 the detection of the valid point is shown. In order to have greater confidence in the detection, it is necessary to search for how many points that belong to a straight line are present between these two opposite points.

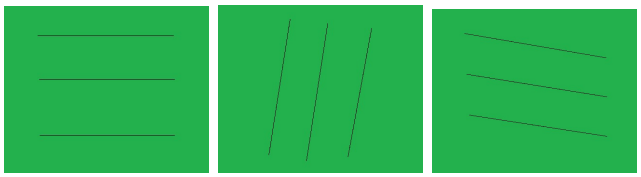


Figure 3.7: Synthetic images for first test.

3.3.1 CBS Line detection process

This algorithm allows us to detect valid points of lines. A valid point is defined as the center of a circle of searching in which it is possible to find two equidistant points that belong to a line (see Figure 3.9). By using a circle drawing algorithm, it is possible to detect valid points in lines. In Figure 3.10, some cases are shown, but the accuracy depends

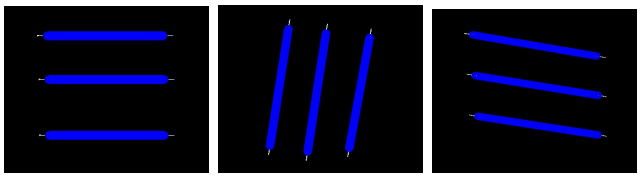


Figure 3.8: Result for the obtaining of the valid point in synthetic images.

on the radius selected. For this reason, the process requires a radius with enough length to draw a circle in a grid. For example, the circles drawn in Figure 3.5 have different pixels radius.

The process comprises the following stages:

- Segment the image with an edge detector, Canny or a Steerable filter may be used.
- For all pixels in the image that are different from the background:
  1. Search for valid points by using a circle drawing algorithm.
  2. If it is a valid point, obtain the value of Dx and Dy (see Figure 3.9).
  3. Move along the direction of symmetry by using the values of (Dx, Dy) (see Figure 3.11).
  4. While (A valid point is found):
     - Move the position (x, y) to the values of $(x + skip \cdot Dx, y + skip \cdot Dy)$.
     - Search for valid points by using a circle drawing algorithm.
  5. Save the first and the final point.
  6. Trace a line between the first and the final point in order to erase the pixels associated to the line in the segmented image.

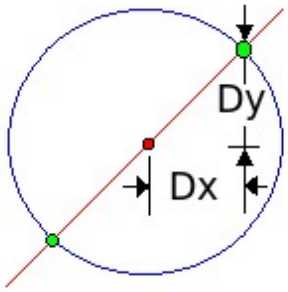


Figure 3.9: Valid point: the center of the circle with two equidistant points.

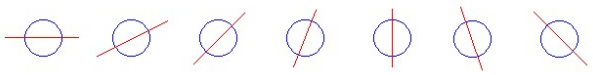


Figure 3.10: Different angles of lines.

Algorithm 3.1: CBS Algorithm

```
Data: x,y,r,rend,skip,pos,Image
Result: Line list[]
p=pos;
while r < rend do
    read current;
    if ValidPoint(M, x, y, r, Dx, Dy,skip) then
        if abs(Dx)>abs(Dy) then
            steps= abs(Dx);
        else
            steps= abs(Dy);
        end
        incX=Dx/steps;
        incY=Dy/steps;
        xi=x-Dx;
        yi=y-Dy;
        while NextCircle(M,x,y,Dx,Dy)>N do
            x+=Dx*skip;
            y+=Dy*skip;
            k++;
        end
        xf=x;
        yf=y;
        for i ← 1 to k do
            // Erase line
            Image[x][y]=0;
        end
    end
    list[p].x0=yf;
    list[p].y0=xf;
    list[p].xf=yi;
    list[p].yf=xi;
    p++;
end
```

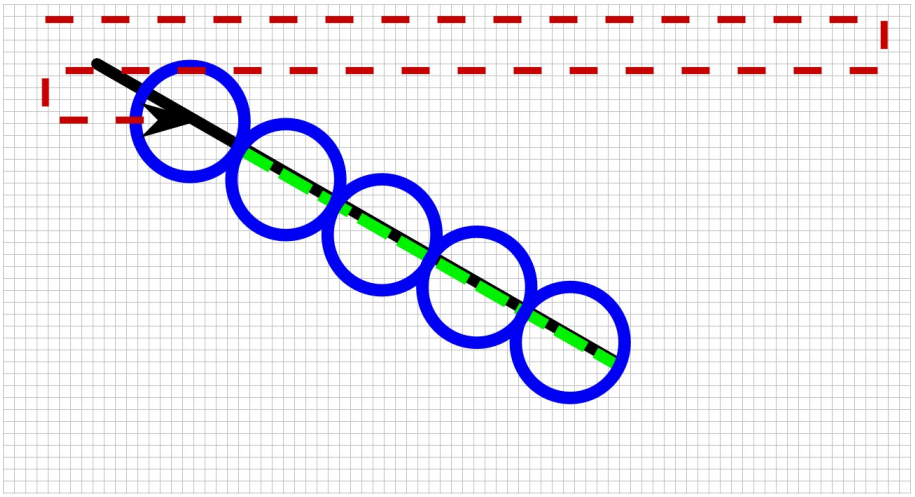


Figure 3.11: CBS process.

In the Algorithm 3.1, a more detailed version of the process is shown:

The main parameters of this method are the following:

- Search radius: This is the radius of the circle which depends on the length of the line and the size of the image. For instance, in Figure 3.5, we can see discrete circles of different radius. When the radius of the circle is increased, it allows to have more possibilities for evaluating angles of the line defined with more precision by its symmetrical points.
- Percentage of points detected: When a valid point is selected, it is necessary to take into account how many points are found in a straight line between its two end points. The points are obtained by using computer graphic primitives for line tracing such as DDA or Bresehnham [49].
- Skip value: This is an increment from a valid point to another. It has a way to control the velocity of the process and the accuracy. This value can be near to the circle diameter.

3.3.2 Linking of rectilinear segments

According with [17], there exists a set of features based on the Gestalt principals [54] that can determine if a segment is contiguous to another and can be linked in order to form longer lines.

In this case, the following features are used:

- Similarity: The angle of the lines is used.
- Length: The process gives priority to longer lines, since power lines are commonly the longest lines in the scene.
- Proximity: If two lines are close enough and are similar, they can be replaced by only one in order to reduce redundant information. The euclidean distance between its points has to be computed.

After the linking process the algorithm produce a set of lines detected. In this set, the lines are represented with the initial (x_i, y_i) and the final points (x_f, y_f).

3.4 Experimental setup for line detection

Two tests were developed. The first one is done with synthetic images. The second one is done with real images. In these tests, we use a Canny filter. For this process, we compute the detection with different radius. We use a computer Asus G53X with a processor Core i7, 8GB of RAM and Ubuntu 12.04.

3.4.1 Synthetic images test

A set of synthetic images with lines in different inclinations (see Figure 3.7) is used. The purpose is to evaluate the level of detection with different radius of circle in the line detection method. Also, a regular radial pattern was created in order to evaluate and compare the performance of the proposed method with lines of different angles. In this case, we evaluate the range of $[0, 2\pi]$ with increments of $\pi/16$ (see Figure 3.15(a)). Additionally, a three dimensional model of the power lines was built in order to generate synthetic images of the power lines with different points of view (see Figure 3.12).

(a) (b) (c) (d)

Figure 3.12: Simulated images of power lines in a 3D enviroment.

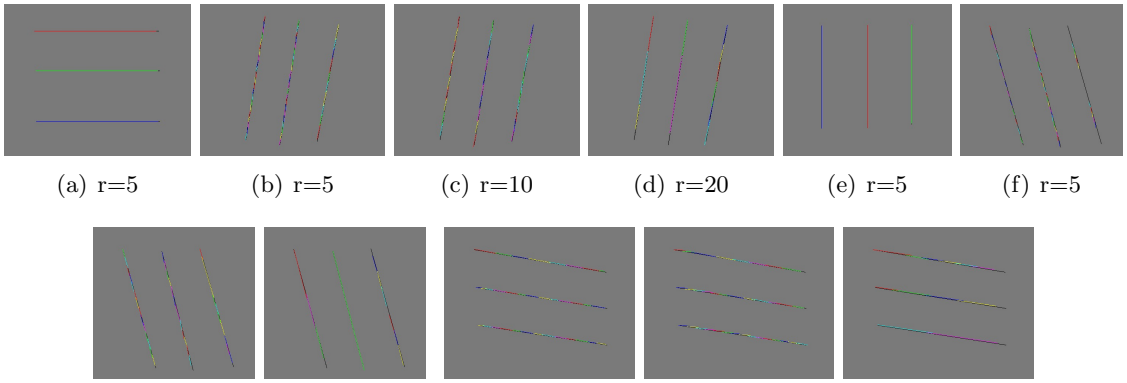
3.4.2 Real images test

For the second test, we use a set of images of power lines in structured environments, in order to compare the performance and computing time of the different line detection methods. In first instance, a Canny detector was used; after that, a steerable filter was used.

3.5 CBS Results

For the synthetic images test, it was observed that the performance depends of the radius which needs to be long enough and also depends on the line length. It is observed that is possible to detect contiguous lines with a radius of 5 pixels, but the performance is improved with a radius increment to 10 and 20 pixels; this is shown with segments of different color that represent the detected lines. It is good to mention that the color of the lines is random and its purpose is to differentiate the segment detected. The best results

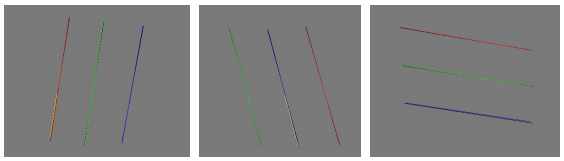
in terms of continuity are obtained with a radius of 20 pixels for the evaluated images since the amount of segments is reduced. For this images of 390 × 317 pixels are used. The results of synthetic images are shown in Figure 3.13.



(a) r=5 (b) r=5 (c) r=10 (d) r=20 (e) r=5 (f) r=5

(g) r=10 (h) r=20 (i) r=5 (j) r=10 (k) r=20

Figure 3.13: Result of detecting lines in synthetic images.

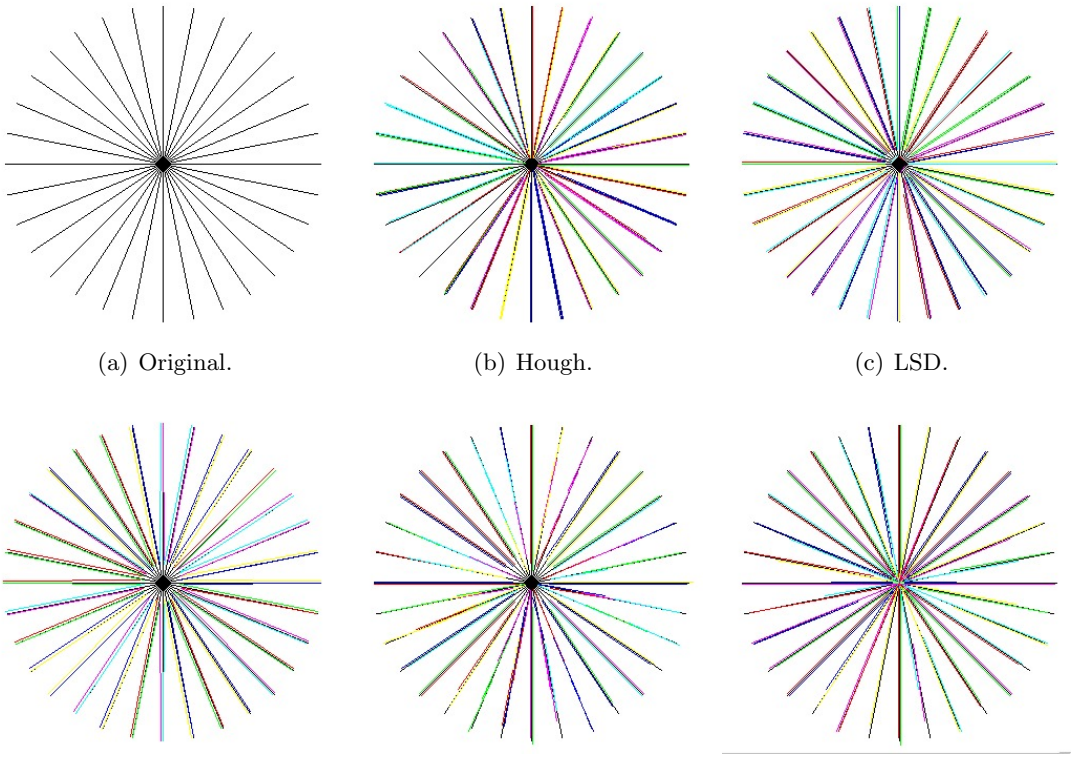


(a) Linking of figure 3.13(a). (b) Linking of figure 3.13(f). (c) Linking of figure 3.13(i).

Figure 3.14: Result of linking segments of some images of figure 3.13.

Although the detection by using a small radius creates many contiguous segments, the implemented process of linking allows to overcome this situation. This improves the performance of the proposed method (see Figure 3.14 and 3.15(f)). It is important to note that for vertical or horizontal lines (Figures 3.13(a) and 3.13(e)) the radius size has no important effect in generating line segments and a linking stage it is not required.

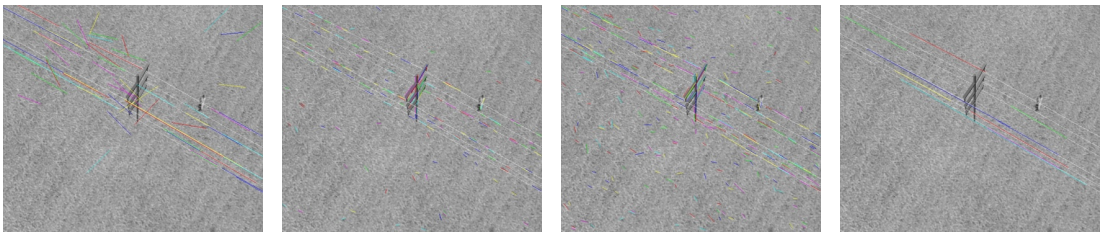
In the test of the radial pattern, we used two consecutive searches. The first search with a radius of r pixels and the second with a radius of $r-4$ pixels, in order to detect all possible lines. We obtained good results with $r = 11$ and $r = 15$ pixels for this pattern. The results of this test are shown in Figure 3.15. In Figure 3.15(f), the linking stage is shown, and, as it can be seen, the result is improved. We can observe that two lines that were not detected by Hough at 135 and 225 degrees were detected by CBS. Also, the lines detected by CBS were very close to the original pattern.



(a) Original. (b) Hough. (c) LSD.

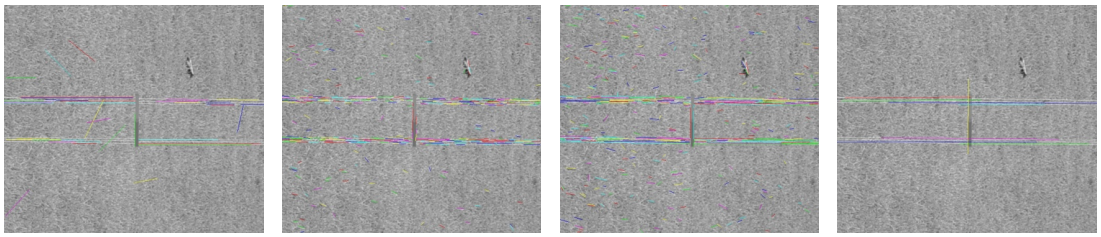
(d) EDLines. (e) CBS. (f) Linking.

Figure 3.15: Comparison of performance with radial lines with an increment of $\pi/16$.



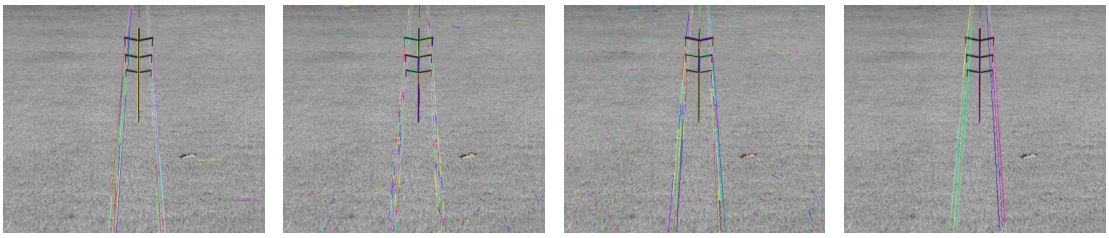
(a) Hough. (b) LSD. (c) EDLines. (d) CBS.

Figure 3.16: Results of detection with different methods for the image in the Figure 3.12(a).



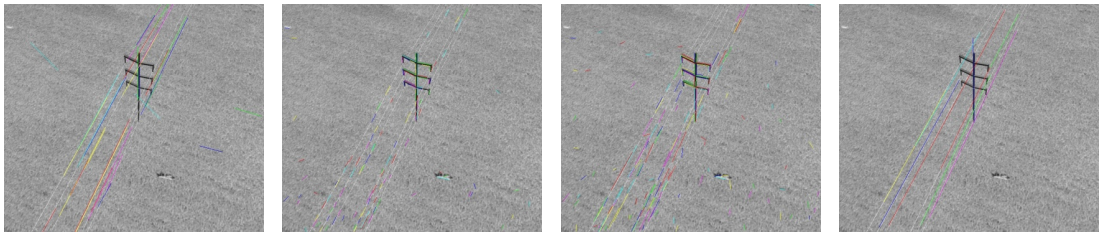
(a) Hough. (b) LSD. (c) EDLines. (d) CBS.

Figure 3.19: Results of detection with different methods for the image in the Figure 3.12(d).



(a) Hough. (b) LSD. (c) EDLines. (d) CBS.

Figure 3.17: Results of detection with different methods for the image in the Figure 3.12(b).



(a) Hough. (b) LSD. (c) EDLines. (d) CBS.

Figure 3.18: Results of detection with different methods for the image in the Figure 3.12(c).

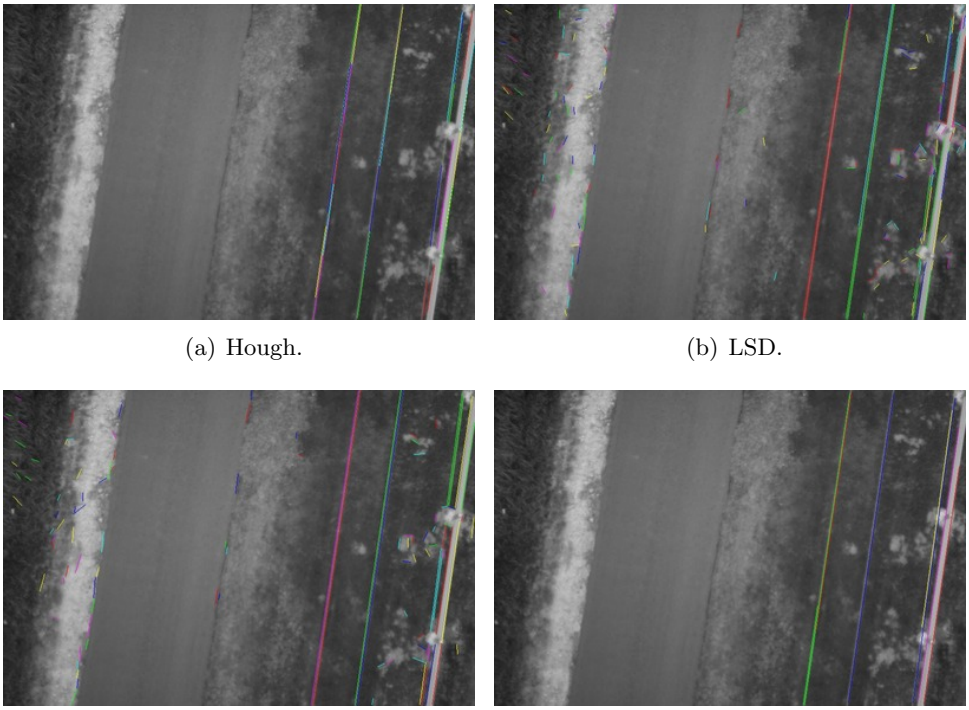
(a) Image 1. (b) Image 2.

(c) Image 3. (d) Image 4.

Figure 3.20: Real images used in the second test with size of 800 × 533.

In Figures 3.16, 3.17, 3.18 and 3.19, the results for a set of images of a synthetic environment are shown. It is important to mention that by using virtual environments,

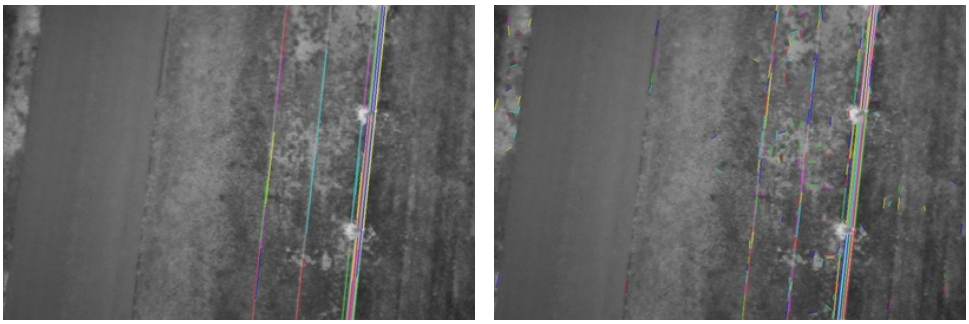
it is possible to create different configurations of the scene for validating computer vision techniques. As we can see, the performance of the CBS is good due to the fact that it detects longer lines. We achieve similar results to EDLines in terms of time and lines detected, (See Table 4.1).



(a) Hough. (b) LSD.

(c) EDLines. (d) CBS.

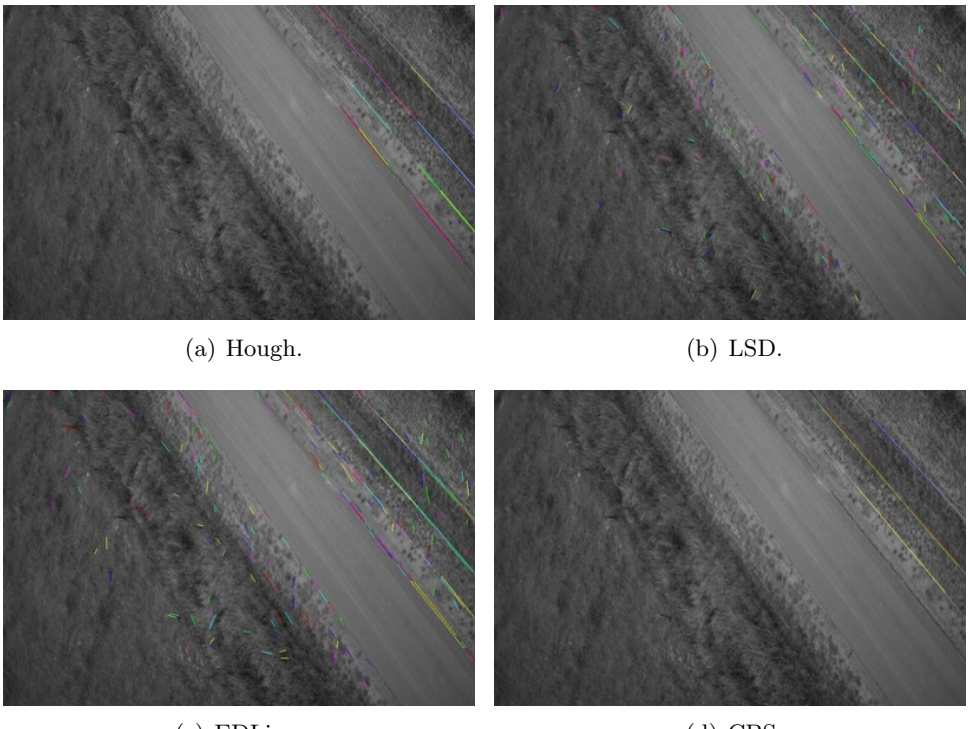
Figure 3.21: Results of different line detection methods for the image in the Figure 6.2(a).



(a) Hough. (b) LSD.

(c) EDLines. (d) CBS.

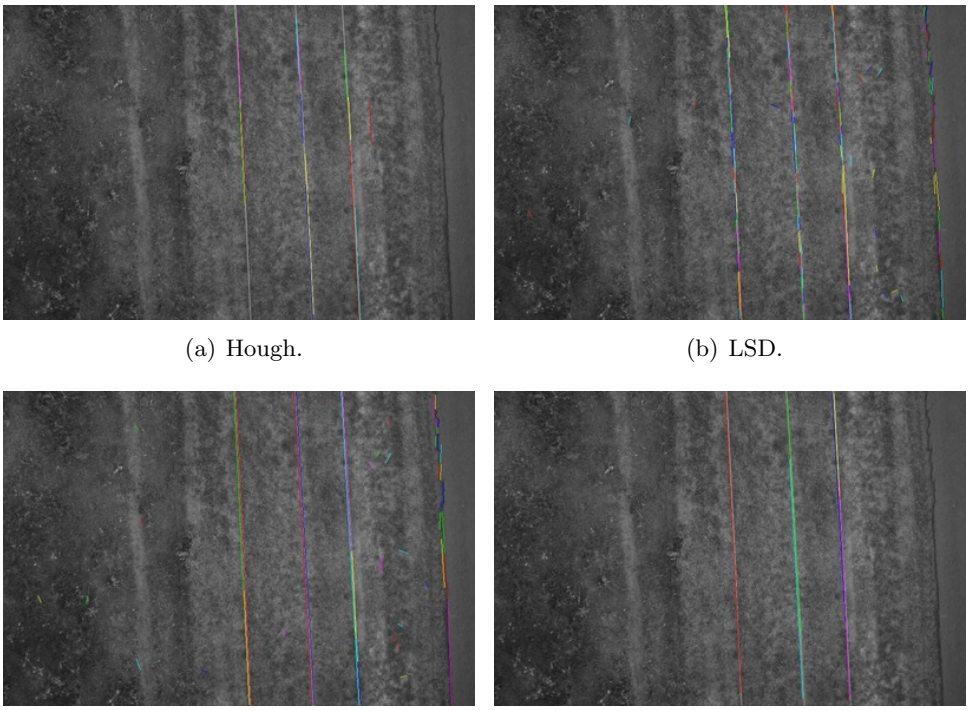
Figure 3.22: Results of different methods for the image in the Figure 6.2(b).



(a) Hough. (b) LSD.

(c) EDLines. (d) CBS.

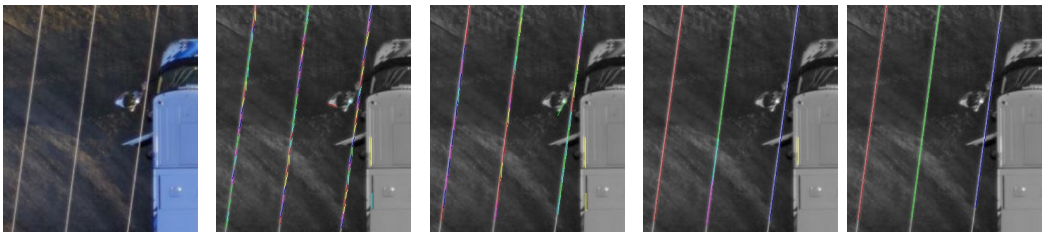
Figure 3.23: Results of different methods for the image in the Figure 6.2(c).



(a) Hough. (b) LSD.

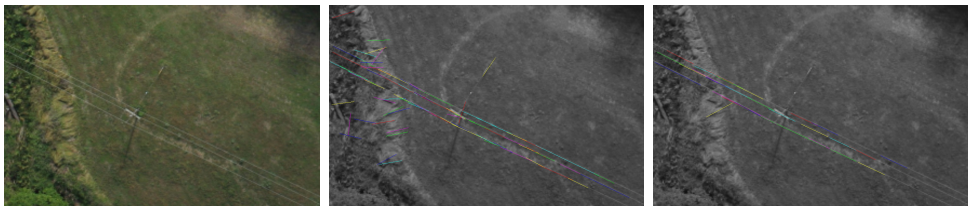
(c) EDLines. (d) CBS.

Figure 3.24: Results of different methods for the image in the Figure 6.2(d).

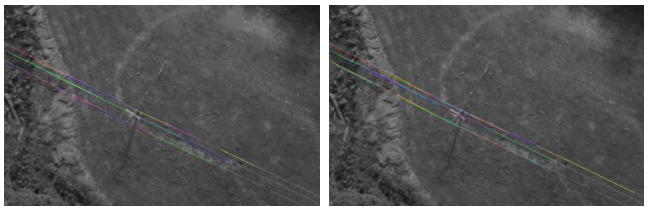


(a) original. (b) r=5. (c) r=10. (d) r=15. (e) r=20.

Figure 3.25: Result of detecting lines in real images using CBS with different radius and steerable filters in a small size image 308 $\times$ 360 pixels.



(a) original. (b) r=30. (c) r=40.



(d) r=50. (e) r=60.

Figure 3.26: Result of detecting lines in real images using CBS with different radius and steerable filters in a big size image 1188 × 756 pixels.

For real images, we used 20 images (see Figure 3.20) and we obtained good results with our line detection method in different situations. In Figures 3.21, 3.22, 3.23 and 3.24, the results of using different line detection methods for four images are shown. Although synthetic images were used, the CBS showed good performance for line detection. In this case, we used a Canny filter for segmentation.

On the other hand, we used a combination of steerable filters and the CBS method for evaluating different radius. The results are shown in the Figure 3.25 and 3.26. Results show that the amount of lines detected by CBS are superior than when using the Hough transform and similar to the ones when LSD and EDLines are used.

We can see that there exists a relation between the radius selected, the width of the line and the size of the image. For instance, in images of a small size (308 × 360) with line size near to 300 pixels of length, we obtained a good detection level with a radius of 15 pixels. Although for small size images with detailed objects it is necessary to use a radius from 3 to 5 pixels for detecting these details, for power lines it is not necessary since longer lines have to be detected. When a bigger image is processed (1188 × 756), a bigger radius can be used to get better detection or longer segments detected.

As shown in Table 3.1, the CBS method has a better performance time than the Hough transform, and in some cases it is better or close to EDLines. It is important to mention that when the radius is increased a longer arc is produced in a search based circle. For this reason, our method takes more time for bigger images.

Finally, in the table 3.2 a comparison between the different methods of line detection for 10 images is presented. In the used images three power lines are presented. In this table, it is remarkable that the number of FP is greater in the LSD method, since it detect a lot of small segments. The CBS present a reduced number of FP compared with the other methods.

Table 3.1: Computation times

Figure	Size	Hough	LSD	EDLines	CBS
3.15	333 × 333	21.1 ms	18.9 ms	5.22 ms	5.88 ms
3.13	390 × 317	8.7 ms	14.5 ms	3.5 ms	3.1 ms
3.21	800 × 533	8.69 ms	62.44 ms	6.12 ms	7.61 ms
3.22	800 × 533	11.41 ms	69.78 ms	6.80 ms	7.97 ms
3.23	800 × 533	11.83 ms	76.87 ms	6.12 ms	7.61 ms
3.24	800 × 533	11.41 ms	69.78 ms	8.50 ms	10.40 ms

3.5.0.1 Precision Recall

Using the total of TP, FP and FN for each line detection method. The precision recall values were obtained in testing two videos using the follow expressions:

$$Precision = \frac{tp}{tp + fp}, \tag{3.2}$$

$$Recall = \frac{tp}{tp + fn}. \tag{3.3}$$

As is shown in Table 3.3, although the recall present a good result the precision is a very low value. The improvement of this performance requires the use of more advances methods related with computational intelligence that are proposed as a future work.

Table 3.2: Quantitative results of line segment detected for 10 images.

	Hough			LSD			EDLines			CBS		
Image	TP	FP	FN	TP	FP	FN	TP	FP	FN	TP	FP	FN
1	3	1	0	3	102	0	3	14	0	3	3	0
2	3	8	0	3	75	0	3	13	0	3	7	0
3	3	5	0	3	81	0	3	21	1	3	0	1
4	3	1	0	3	37	0	3	17	0	2	0	1
5	3	5	0	3	18	0	3	27	0	3	5	0
6	3	19	0	3	110	0	3	33	0	3	14	0
7	3	7	0	3	89	0	2	20	1	2	4	1
8	3	6	0	3	139	0	3	15	0	3	8	0
9	2	1	0	2	84	0	3	16	0	2	3	0
10	3	1	0	3	28	0	3	24	0	3	1	0

Table 3.3: Precision and Recall for line detection methods

	Hough	LSD	EDLines	CBS
Precision	0.34	0.03	0.12	0.37
Recall	1	1	0.93	0.9

3.6 Catenary detection

As already stated, most of the works on power line detection are focused in straight line detection. Nevertheless, the electrical infrastructure is composed of catenaries which are generated when a flexible cable is suspended between two poles or towers. This type of object appears in images of electrical infrastructure taken from non-azimuthal views which could be obtained using manned vehicles or UAVs. For the catenary detection, the accuracy of line detection methods is used in order to get several segments that are part of a catenary. By using line properties, it is possible not only to convert a set of unconnected segments in a polyline but also to reject the unconnected and isolated elements. The final polyline can be stored using data structures composed of linked lists. The process is composed of the following steps:

1. Extract line segments using LSD or EDLines methods and put them in a list.
2. Compute segment properties: length, proximity and colinearity.
3. Sort the list of segments from the longest to the shortest.
4. Search the sorted list, and concatenate contiguous segments depending on segment properties and only if these were not concatenated before. Mark the segment as used.
5. Continue with the next segment not used.

3.6.1 Line properties

There are some properties which are based on the Gestalt principles used to define if two line segments are similar, proximal or continuous. These properties have been used to concatenate line segments in a straight line [17]. In this work, this type of properties have been adapted for curves using proximity and collinear measurement. One difference with the work presented on [55] is that the proximity between two segments is computed as the closest distance between the line extremes e by using the expression presented in Equation 3.4.

$$e = \min(dist(a,c), dist(a,d), dist(b,c), dist(b,d)). \tag{3.4}$$

Being a, b the extremes of $line_1$, c, d the extremes of $line_2$ and m_1, m_2 the midpoints of each line. The collinear measurement is obtained as $\theta = \theta_1 + \theta_2$ (see Figure 3.27). The collinear segments have to be selected in first instance and then the proximal ones.

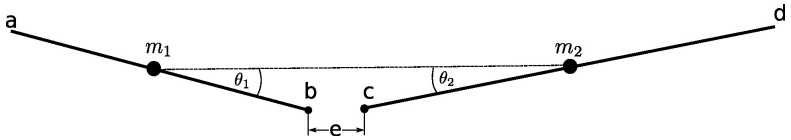


Figure 3.27: Geometric relations used in the concatenation.

3.6.2 Concatenation process

The process begins with a segmentation using Canny or Edge Drawing for edge detection. After that, a line detection method is applied. In the process of line segment detection EDLines is used. For each segment detected, the mentioned line properties are extracted. Depending of the line properties, the segments are concatenated and stored in a list. Finally, the catenaries are selected based on its properties such as length and orientation since short lines (lines with a number of pixels less than a threshold in this case 15) are not taken into account neither the vertical ones since catenaries are not vertical. The concatenation process stages are presented in Figure 3.28. In order to show the performance of the technique with detected segments on an uniform background, the results of a concatenation process with synthetic images (in Figure 3.29(a)) is shown in Figure 3.29(b). In this case, a set of synthetic curved lines with a drawing program is created.

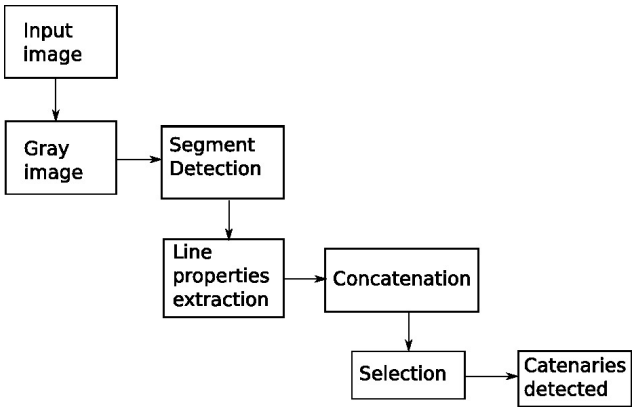


Figure 3.28: Catenary detection process.

(a) Synthetic catenary. (b) Detection results.

Figure 3.29: Example of detection with a synthetic image.

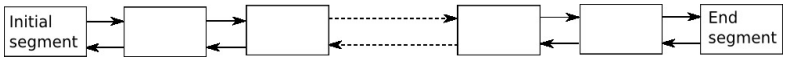


Figure 3.30: Generation of a list of contiguous elements.

3.6.3 Selection

A double linked list of segments with the most suitable closest segments is created in order to construct a polyline for each catenary (see Figure 3.30). The segments are attached if they have the minimal collinear and proximity measure. Depending on the location

of a valid segment, it will be attached to the beginning or the end of the list. For this reason, the two extremes of the list have to be evaluated. If the concatenation of a new segment results in a shorter path through the initial and final extremes of the polyline, the segment must be rejected in order to avoid revisiting the followed path. This means that the nearest elements that do not make a contribution to the catenary length are not being stored. A catenary must not have cyclic connections or loops. An approximation to the length of a catenary composed of N segments is obtained by summing the length of each segment (see Equation 3.5).

$$catenary\_length = \sum_{i=0}^{N-1} length(line_i). \tag{3.5}$$

3.6.4 Catenary Results

For development and testing, an HP Envy laptop with a Core i7 processor, 8GB of RAM and running linux Ubuntu 14.04 was used. OpenCV for implementing the computer vision algorithms is used. The dataset is extracted from two sources high altitude and low altitude. Firstly, a video of a power line inspection where an aerial vehicle was used. A subset of 20 images with the presence of catenaries was evaluated. In Figure 3.31, a selection of those original images of catenaries is shown. The image resolution is 611 $\times$ 471 pixels.

In Figure 3.32, the segment detection results in a partition set of catenary images by using EDLines is shown. In Figure 3.33 the catenary detection results with the proposed process are shown. In these images the catenary detection allow to detect most of the longest catenaries in the scene. Different colors are assigned to each segment or catenary to differentiate them from each other.

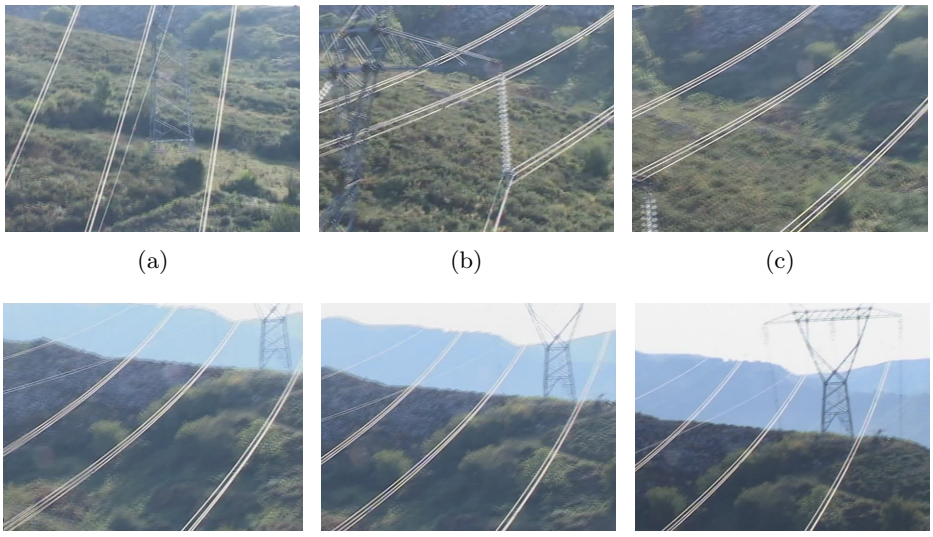
The number of catenaries considered as TP (true positives), FP (false positives) 3.4 and FN (or not detected) are shown in Table 3.4 for the 20 images. Using this information the rate of TP and FP are obtained as shown in Table 3.5.

The lines not detected are mainly short, farther and without a relation with the power lines. For the performance of the catenary detection process the true positive (TP) rate and the false positive (FP) rate are obtained for the set of images. This is shown in the second row of Table 3.5.

The performance shows that the method generate polylines for most of the catenaries in the images. Nevertheless, there exist some cases where a power line can be mistaken by a linear element in the scene such as a pylon or a part of the tower. The computing time was 220 milliseconds per image.

Secondly, the catenary detection is evaluated in 10 images taken from low altitude of places with electrical infrastructure with high resolution: 1280 $\times$ 960 pixels (see Figure 3.34). The performance of the catenary detection for high resolution images process is shown in second row of Table 3.5. In this case, the size of the image is bigger and more complex than the previous ones, more comparisons are required. For this reason, the computing time is increased to 580 milliseconds per image. The selection of the most relevant catenaries is done by selecting the longest polylines. This is done by adding the distance of each one of connected nodes of the catenary.

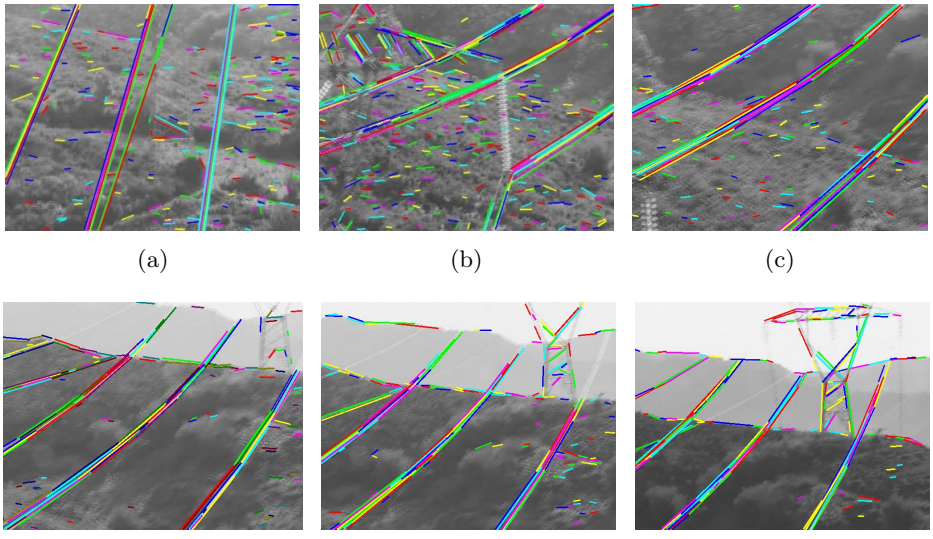
The developed process helps to eliminate segments that in most of the cases do not belong to the power lines as shown in Figure 3.35. The catenary detection is shown in Figure 3.36.



(a) (b) (c)

(d) (e) (f)

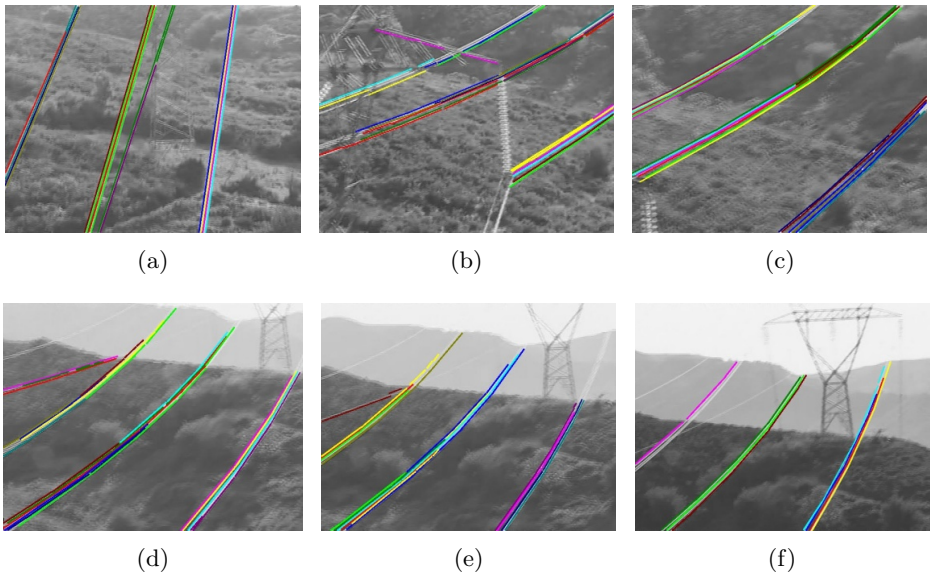
Figure 3.31: Images with real catenaries with a resolution of 611 × 471 pixels.



(a) (b) (c)

(d) (e) (f)

Figure 3.32: Detection of line segments of Figure 3.31.



(a) (b) (c)

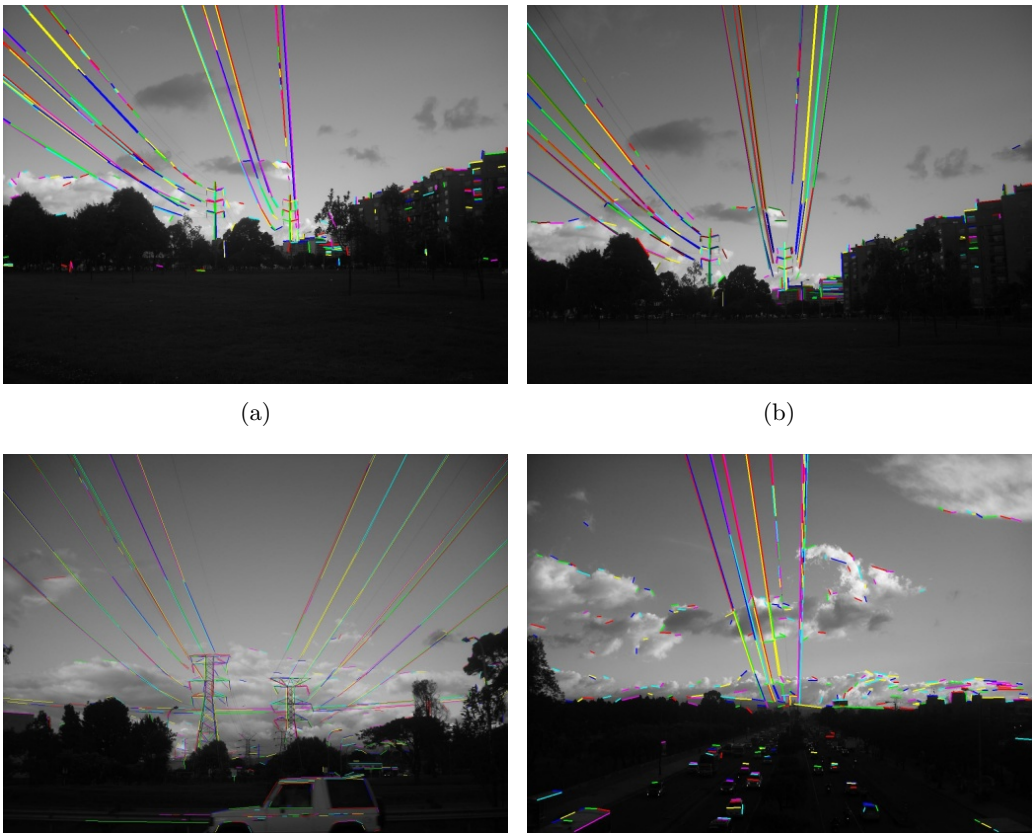
(d) (e) (f)

Figure 3.33: Detection of catenaries of Figure 3.31.

(a) (b)

(c) (d)

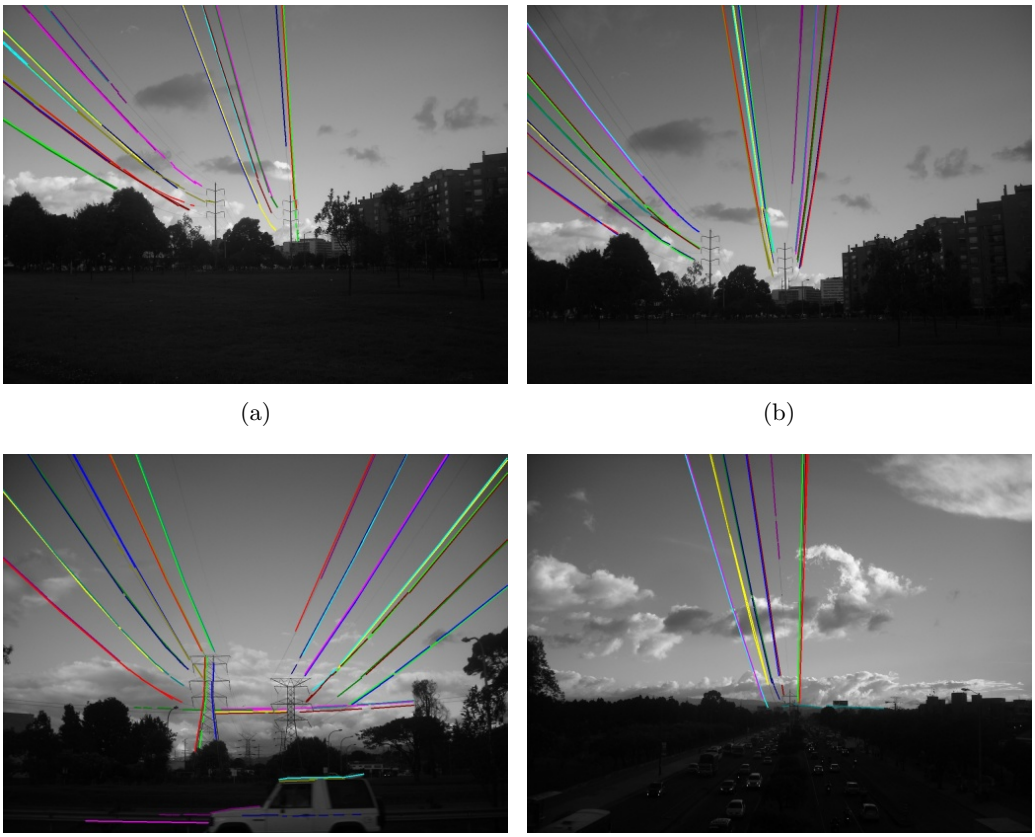
Figure 3.34: Images with real catenaries of 1280 × 960 pixels.



(a) (b)

(c) (d)

Figure 3.35: Detection of line segments of Figure 3.34.



(a) (b)

(c) (d)

Figure 3.36: Concatenation of line segments of Figure 3.35.

Table 3.4: Evaluation per image.

Image	TP	FP	FN
1	9	0	0
2	9	0	0
3	9	0	0
4	9	0	0
5	9	0	0
6	9	0	0
7	12	0	0
8	11	0	0
9	13	0	0
10	13	0	0
11	13	1	2
12	13	1	3
13	13	0	0
14	13	0	2
15	13	0	2
16	13	0	2
17	13	0	2
18	12	1	3
19	12	2	2
20	11	0	2

3.6.4.1 Precision Recall

The precision recall values were obtained in testing two videos using the follow expressions:

$$Precision = \frac{tp}{tp + fp} \tag{3.6}$$

$$Recall = \frac{tp}{tp + fn} \tag{3.7}$$

Table 3.5: Detection results of catenaries on images of different sizes.

Size (pixels)	TP rate (%)	FP rate (%)	FN (rate)	Precision	Recall	Time
611 × 471	91 %	2 %	8 %	0.97	0.91	220 ms
1280 × 960	84 %	5 %	16 %	0.94	0.86	580 ms

3.7 Conclusions

The CBS is a new method for line detection that takes less time than Hough and LSD. The parameters of the method permit detecting lines in images with noise. In addition to this, it has a stage for connecting contiguous segments.

This method was validated using several test on real and synthetic images, obtaining good results in both cases.

With the use of steerable filters, it is possible to obtain better results in segmentation of power lines since only one edge is obtained but CPU processing takes more time than Canny.

The proposed method detected power lines with good performance in comparison to state of the art approaches. Even though, this method is not so accurate as EDLines or LSD for detecting short lines in complex scenes, it has good performance in scenes with longer lines. In addition, it is more suitable for power line detection in images of small size.

Another advantage is that CBS is easy to reproduce since it is based in circle drawing algorithms, such as Bresenham or Mid-point. In consequence, the method uses integer math and can be suitable to be implemented in embedded systems without float point operations.

The proposed catenary detection process allows to reduce drastically the number of line segments that are not belonging to a power line. The approach takes advantage of detecting the most relevant segments in the scene by using a fast method for line segment detection after the segments are concatenated.

It is remarkable the good rate of TP, the reduced rate of FP in images with different resolutions for detecting catenaries in real images from inspection tasks and the short computing time. The use of this information could be valuable for the UAV navigation because the technique can give an idea of the presence of catenaries which is necesary in order to inspect and avoid collisions with them. Catenary detection could give an estimation of the position of the UAV with respect to power lines and gives an initial estimation of the tower location. The slope of the lines could be evaluated in order to detect vertical lines that belong to poles or towers but not to a catenary.

All power lines are composed of catenaries. For this reason, it is valuable to develop techniques that permit its detection in short time. The use of properties like proximity and collinearity allows to concatenate a sort of contiguous segments that are part of power lines in straight and curve lines such as catenaries.

CHAPTER 4

Tower Detection

In this chapter a method for transmission tower detection that involves the use of visual features and linear content of the scene is presented. For this process, a descriptor based on a grid of 2D feature descriptors that is useful not only for object detection but also for tracking the interest region is developed. For the detection and classification a SVM is used. The experiments conducted on a dataset of real world images from transmission tower videos were used to validate the strategy by comparison with the ground truth. The results show that the obtained method is appropriate and fast for tower detection in video sequences of environments that include urban areas. The detection is obtained in less than 50 ms and is faster than other methods.

This work was inspired by two main cases: tower detection for power infrastructure inspection and improvement of the security of UAV navigation systems. In the first case, the detection of towers is considered an input for a visual control system to navigate towards an electrical tower in order to perform inspections. The problem with navigation systems based on power line detection from an azimuthal view, is the difficulty of detecting the power lines in complex backgrounds in real time. The differentiation of power lines from other linear elements, for example, over a road or a fence, is complicated, even more so if there is a heterogeneous background. For this reason, is presumed that the most distinctive objects that can be a reference in a visual based navigation system for power line inspection schemes are the transmission towers. Additionally, tower detection is an important stage to detect other important parts of electrical infrastructure, such as power lines and isolators. In the second case, the detection of electrical towers is important in order to reduce the risk of collisions between UAV and towers or the attached power lines because there are thin elements that are difficult to detect in images obtained while in motion.

4.1 Related work

There is a great number of approaches for object detection: template matching, cascade of classifiers, feature detection and mixtures of multiscale deformable part models. The classical method for object detection is found in the study by Viola-Jones [56]. This approach uses a process called “cascade” to discard the unimportant parts of an image using several stages of classifiers. In addition, an integral image is used for the fast

calculation of rectangular features in order to perform a fast face detection. Although this method is fast for object detection, it takes a lot of training. On the other hand, the methods based on SVM (Support Vector Machine) have several advantages in the discrimination capacity and training time. The study by Dalal and Triggs [57] details the use of HOG (Histograms of Oriented Gradient) descriptors for human detection; in this case, they used a linear SVM as a classifier. More recently, methods based on mixtures of multiscale deformable part models have shown good results for object detection; the study by [58] used 2D HOG descriptors. Another interesting study [59], which used a rigid template of HOG descriptors and exemplar SVM had a good performance, as compared to the previous approach.

There are several methods for pylon or tower detection. In the study by Golightly [60], corner detection and matching were applied to power pole detection. Cheng [25] used segmentation based on a graph cut algorithm for power pole detection. Cetin *et al.* provided an algorithm for the recognition of similar electrical poles from an aerial image by detecting the pole shadow and using the Radon Transformation [61]. Tilawat [62] developed a method based on an optimal IIR filter and Hough transform for line detection. More recently, in a study by [26], a strategy that included Hough transform, HOG descriptors [57], bounding boxes and neural networks was developed for electric tower detection and tracking by using computer vision and machine learning techniques. Some of the previous studies were based on the use of the Hough transform, which is a classical method, but not the best one for line detection in power line environments. For this reason, in this work, a comparison of line detection methods was made in order to obtain the most suitable one in electrical tower scenes. Additionally, the environments seen in these studies did not consist of buildings or other kinds of constructions. The results include the use of images of rural and urban areas, where the towers can be confused with other objects.

HOG descriptors have been used widely for object detection. Nevertheless, other kinds of descriptors that focus on keypoints are used in the context of object recognition by using a matching approach. For example, in [63], the SIFT (Scale-invariant feature transform) descriptor and mean shift clustering were used in a scheme of picking place application. In the study by [64], a combination of SURF (Speeded Up Robust Features) descriptors and cascade filters was used for object detection.

As a precursor of the proposed method, some descriptors based on histograms, such as HOG, where the accumulation in the cells can be mentioned. Also, 3D descriptors were explored in our previous study [28]. Nevertheless, any other study that had developed the idea of a GRID descriptor as an accumulative array of 2D descriptors was not found.

Due to the fact that our strategy is based on fast detectors and efficient feature descriptors, as well as in line detection methods, a set of 2D detectors and line detection methods were considered in order to find the more suitable ones for transmission tower detection. Some of the more representative 2D descriptors were: SIFT [31], SURF [32], BRIEF [33], ORB [34], BRISK [35] and FREAK [36].

Since electrical infrastructure consist on linear infrastructure different line detection methods were Hough Transform, LSD [43], EDLines [42] and CBS [19] are used for help in the detection.

Is considered that 2D descriptors are powerful tools for object featuring that can be exploited for obtaining a grid of accumulated descriptor values that can be used with line

detection methods in a machine learning scheme for object detection, as shown in this chapter.

The process used for the transmission tower detection takes approximately 20-45 ms for the detection and less than 1 hour for training with 319203 samples.

4.2 Motivation for the tower detection method

As is mentioned in the introduction the towers are an important element of the electrical infrastructure that have to be detected in order to perform a visual-based navigation process. Since the 2D features have been proven good results in context of recognition and matching [22, 23].

For these reasons, one of the contribution of this thesis is the development of a process for real time transmission tower detection. This process was done with a novel combination of 2D feature descriptors and machine learning techniques in an object detection process that uses line detection methods.

Additionally, a methodology to manage the training process in an object detection scheme can help to automatize the process is presented.

4.3 Proposed Method

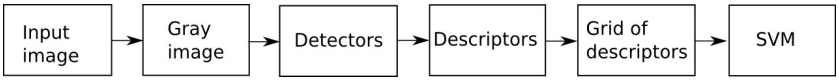
The detection system is based on line detection since there are a lot of linear segments that are present in electric infrastructure areas, as compared with other elements that are not linear, such as trees, mountains and clouds. Line detection is the first stage in a detection system since it quickly provides areas of interest that could be part of an electric tower. Since it is not possible to detect a tower just using line detection methods, 2D feature descriptors to refine the detection of tower elements in areas of interest are used. These descriptors are very useful in different kinds of processes that are related to matching and object detection because they are very representative of the scene content and are fast enough in our application for providing tower detection in less than 100 milliseconds for our dataset.

A composed descriptor based on the geometric location of keypoints in a grid that can be obtained in a short period of time. In addition, the descriptor information to track the area of interest is used, since it improves the results when the classifier fails, for this pre-computed values are being used to reduce the processing time.

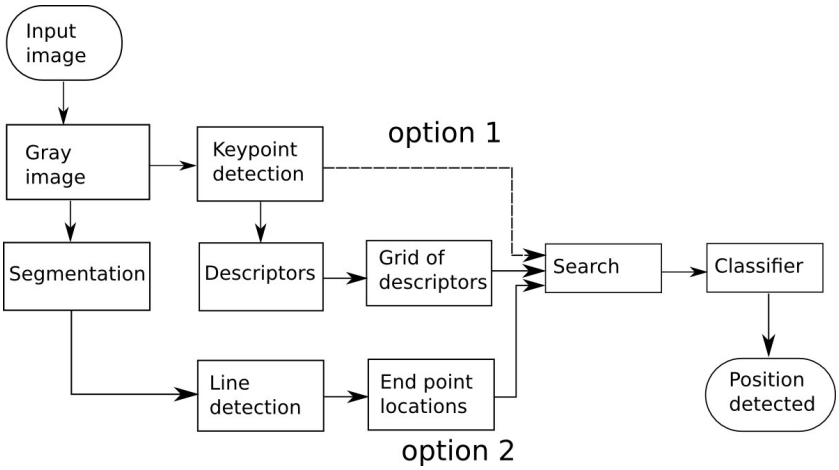
The proposed method was implemented in a system that operates in two modes, training and detection (see Figure 4.1). For the training mode, it was necessary to have a dataset that was composed of tagged ROIs (Region of Interest). The process of tagging was manual. The keypoints that belonged to each ROI were obtained and the descriptors were computed in each keypoint. A grid of descriptors was made with the descriptor values. The grid of descriptors was the input for a SVM which was trained to determine whether the ROI corresponded to a tower or not, see Figure 4.1(a).

In the detection mode, an object search was carried out in order to find the possible location of objects by using the trained SVM. The first stage was the descriptor extraction: here, a keypoint detector was computed in the overall image and the descriptors were

computed in the keypoints. With this information the grid of feature descriptors values was generated. An overview of the detection mode is shown in Figure 4.1(b). Two options can be used: using the list of detected keypoints or using the extremes of the lines detected in the overall image. In the next section, this process is explained in more detail.



(a) Training mode.



(b) Detection mode.

Figure 4.1: Stages of the main components of the proposed method for training and detection.

4.3.1 2D keypoint detectors and descriptors extraction

The descriptor extraction stage is composed of two parts; first the detectors that lead to the selection of relevant points in the image and second the descriptors that provide relevant information in a keypoint. There are different types of keypoint detectors, such as Harris corner detector, SIFT (Scale-invariant feature transform), SURF (Speeded-Up Robust Features) and FAST (Features from accelerated test). In studies such as [22], some keypoint detectors were compared, with FAST [37] being a good alternative in a matching context. Fast is able to find relevant points in images in a short computing time.

On the other hand, descriptors that can be also detectors, such as SIFT [31], SURF [32], BRIEF (Binary Robust Independent Elementary Features) [33], ORB (Oriented FAST and Rotated BRIEF) [34] and BRISK (Binary Robust Invariant Scalable Keypoints) [35], were evaluated in the above studies with different metrics, specially for matching. According to [22], BRIEF has proven slightly faster than ORB and much faster than others, showing a very good performance when used with images that have similar scales and orientations. Although SIFT is regarded as the best descriptor for matching, the computing time is a disadvantage. Another good descriptor is FREAK (Fast Retina Keypoint) [36], but it required more computing time than BRIEF and ORB in our test images, making its use difficult in real time applications. ORB is better with rotated images than BRIEF, and is two orders of magnitude faster than SIFT, which makes this descriptor suitable for real time object detection purposes.

4.3.2 Line detection

Electrical infrastructure has many linear components such as power lines, pylons and towers. For this reason, line detection methods are used to identify each of these components. Although a line detection method cannot identify whether a line is part of a tower or not, this is the first stage in the detection process. The Hough Transform [15], LSD [43], EDLines [42] and CBS [19] are evaluated. The results of the line detection methods applied to tower images are shown in Figure 4.2. As can be seen, it is very difficult to detect the linear structure of a tower by using Hough transform. The time evaluation results of the images of Figure 4.2 are in Table 4.1. The number of detected lines in each image is presented in Table 4.2. These results show that LSD and EDLines are good segment detectors since they accurately detect more segments in images of towers than Hough and CBS. However, CBS is a fast method that detects several relevant lines that are present in the towers. For this reason both EDLines and CBS are evaluated in the proposed approach for tower detection. Since, line detection is a stage in detecting a tower, their ability to detect the lines in the overall scene (whether or not the line belongs to a tower) and the time used for it were evaluated. The color of each segment in Figure 4.2, differentiates the detected lines from each other. It is remarkable that the detection process requires the extraction of more information of the image because there are a lot of detected lines that do not belong to the tower; in the next section, the grid of descriptors is presented since it is used to improve the detection.

(a) Input image 1. (b) Input image 2. (c) Input image 3.

(d) Hough. (e) Hough. (f) Hough.

(g) LSD. (h) LSD. (i) LSD.

(j) EDLines. (k) EDLines. (l) EDLines.

(m) CBS. (n) CBS. (o) CBS.

Figure 4.2: Results of different line detection in images with towers.

Table 4.1: Computation times for line detectors

Image	Size (pixels)	Hough	LSD	EDLines	CBS
Input image 1	640 × 480	12.62 ms	40.04 ms	7.68 ms	3.20 ms
Input image 2	640 × 480	9.29 ms	45.42 ms	11.29 ms	3.07 ms
Input image 3	853 × 480	22.3 ms	120.22 ms	10.93 ms	7.92 ms

Table 4.2: Number of detected lines

Image	Size (pixels)	Hough	LSD	EDLines	CBS
Input image 1	640 × 480	39	382	363	164
Input image 2	640 × 480	22	408	391	117
Input image 3	853 × 480	68	776	697	466

4.3.3 Grid of 2D feature descriptors

In order to achieve a better representation of the object a grid structure for collecting the 2D feature descriptors located in a selected area is proposed. The grid is composed of a set of cells that depends on the image resolution and size. The keypoint and descriptors are obtained in a grayscale image. The descriptors are computed in the keypoints that belong to each cell. For each cell an accumulator is created (in Figure 4.3(a) 12 cells are shown).

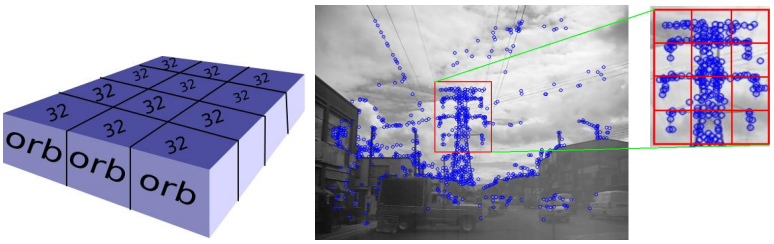
The objective of proposing an accumulator for each cell is to obtain a representative and comparative descriptor for objects.

The ROI acts as a bounding box to select the keypoints that lie within it. The grid allows one to subdivide a ROI of an image in cells, in order to accumulate the values of the descriptors associated with the keypoints that are inside each cell, as shown in Figure 4.3(b). The number of keypoints depends on the non uniform content. For example, there are not many keypoins in a clear sky area, but there are many keypoints in the tower region where the ROI is selected as shown in Figure 4.3(b). When an object is tagged, its region of interest is subdivided in a number of rows and columns and the grid descriptor is computed. This means that the object has a descriptor of $rows \times columns \times size\_of\_descriptor$. For example, a grid can be defined with a 3D array of $4 \times 3 \times 32$ elements in order to obtain a descriptor that considered the aspect of vertical objects, 32 being the number of values of ORB descriptor vector. In each cell, the 32 values of the descriptors computed at each keypoint belonging to the cell are accumulated in a vector of 32 elements, as shown in Equation 4.1.

$$acumulator[n] = acumulator[n] + descriptor[n], \tag{4.1}$$

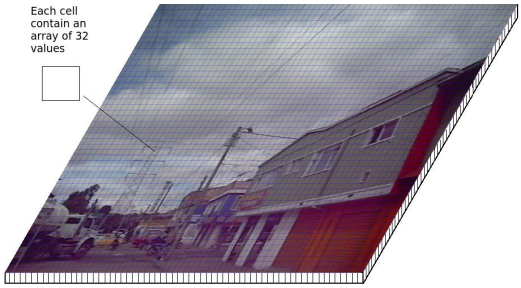
where, n the index for the descriptor values (in this case $0 \leq n < 32$ for ORB descriptor).

In this case the grid of the descriptor with ORB for 12 cells has 384 values (12×32) in the ROI.



(a) Grid structure with an accumulator fore each cell.

(b) ROI subdivision.



(c) Grid in the image.

Figure 4.3: Grid of descriptors.

In order to improve the performance of the process of computing the descriptor and reducing its time in the detection stage, the position of all the keypoints detected in the image has to be scaled to a grid with the same proportions of the original image and lower resolution. This in order to accumulate the value of each position of the descriptors for each cell of the grid (see Figure 4.3(c)), as shown in Equation 4.2; this process will be extended in sections 4.3.4 and 4.3.5.

$$grid\left[\frac{y_i}{bin_y}\right]\left[\frac{x_i}{bin_x}\right][n] = \sum_{(x_i,y_i)\in cell} descriptor_i\,[n]\,, \tag{4.2}$$

where (x_i, y_i) is the position of each keypoint in the image where the $descriptor_i$ is obtained, bin_x, bin_y are the level of selected subdivision (corresponding to 10 pixels in this case) in each (x, y) coordinate, and n the index for the descriptor values (in this case $0 \leq n < 32$ for ORB descriptor), i is the index for the keypoints (in this case $0 \leq i < M$) where M is the number of keypoints belonging to the cell.

Normalization was not used because the magnitude of the descriptor array was significant given that it was an accumulation of values related to the number of keypoints in this cell. Since our feature descriptor was based on the descriptor values in the detected keypoints, the number of detected keypoints in each cell was important because it revealed details in that image area. This means that cells with different number of descriptors should not only have different values but also intensities. If the cell data is normalized, the information can be lost. In order to validate, both cases were tested (with and without normalization) and found better results when normalization was not used, because the number of false positives was reduced. In addition, the number of operations without normalization was reduced. In this case the process of feature extraction and detection was faster than using normalization.

4.3.4 Methodology for tagging, extraction and training

This methodology is composed of four steps: tagging of interest objects, generation of training data, extraction of descriptors and training as is shown in Figure 4.4. This consist on the following stages:

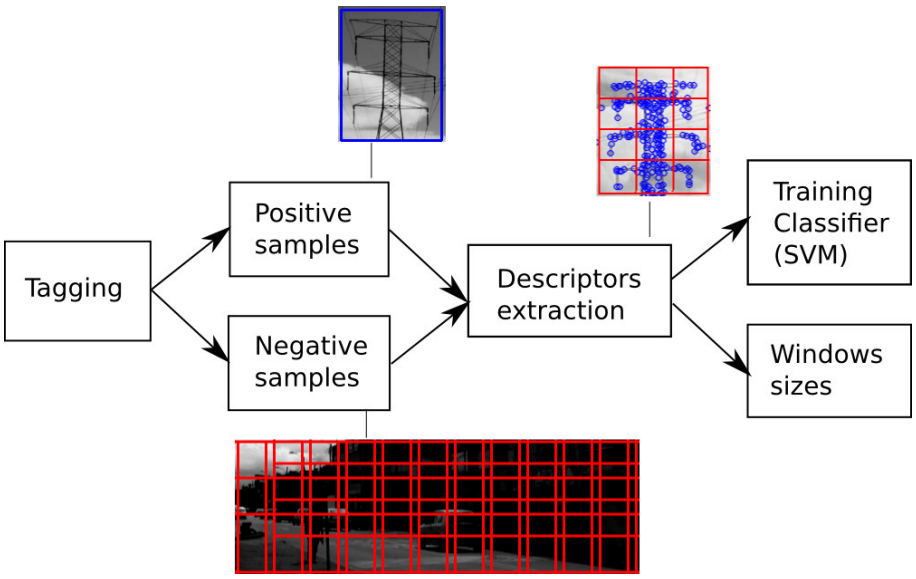


Figure 4.4: Methodology for feature extraction for classifier training.

1. Tagging of object of interest:

   In this stage, the ROI (region of interest) where the tower is present is tagged in order to generate positive and hard examples (negative examples) that are generated using the rest of the image. Several numbers of images from different perspectives have been obtained from videos. The process is defined as follows:

   (a) Tagging a dataset of transmission tower images, by using an application that allows the selection of a rectangle that corresponds to the region of interest (ROI), see Figure 4.5(a). This rectangle was defined manually, by two corners that are represented by two points (min_x, min_y) and (max_x, max_y).

   (b) Gathering the ROIs sizes in a list.

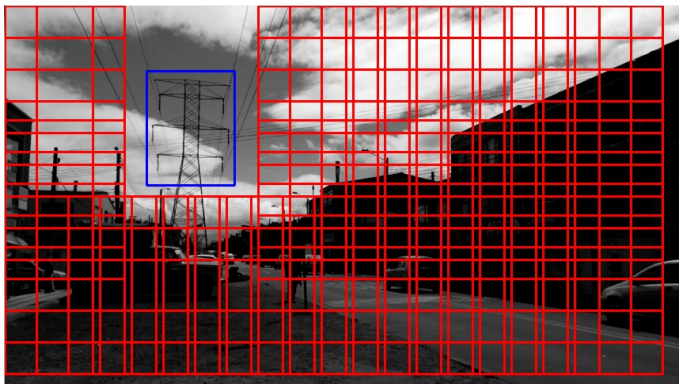
   (c) Given that a search with different sizes is computationally expensive, there are two posibilities: first, to compute a clustering with the list of ROI sizes in order to acquire the more representative search ROI or window sizes for detection. Second, to use a fixed window size and use a bounding box process. In this case, the selection of the window sizes was related to three factors: the size of the objects in the datasets, the time of detection and the required accuracy. The minimal size of the towers in the dataset as a reference for defining the search area is used because using a bigger window is not appropriate for searching small towers. The time required for searching windows is longer when the size is smaller since there are more windows. Also, the descriptor is less accurate when the window is bigger, because there is no opportunity of detecting many details, producing a coarse result.

2. Preparing training data:

   The process of object detection involves the use of images where the tower is presented in a ROI. These regions are called positive samples. In addition, it is necessary to obtain a bigger set of images where the towers are not present (hard examples) using the same dataset in the following way:

   - Positive training data is obtained with the selected ROIs. They have to be tagged first in order to specify the tower location in the image.
   - Negative training data, or hard examples, are obtained by generating several overlapped windows uniformly spaced in the background that do not contain the positive ROIs. This is done by avoiding the previously tagged data. In Figure 4.5(b), the overlapped red regions represent the process of negative samples generation.

(a) Tagging of the area of interest that corresponds to a positive sample.



(b) Method for hard examples generation by using images from the background.

Figure 4.5: Selection of data for training.

3. Extraction:

   In this step, a grid is obtained to generate an image subdivision for 2D feature extraction.

   (a) Scale the image to a fixed size. In this case, two sizes 640×480 and 853×480 pixels were used in order to manage a similar scale for different kinds of aspect

ratio in videos taken from cameras. The image size also favored the computing time reducing it to less than 100 ms.

(b) Compute the keypoints and feature descriptors in the overall image.

(c) Subdivide the image by creating a grid in which each cell corresponds to a square of the selected bin size. For an image of 853 $\times$ 480 pixels and a subdivision of 10 pixels, the created grid has 85 $\times$ 48 cells. In each cell the accumulator is obtained.

(d) Scale the position of the keypoints to the grid size in order to accumulate the value of each position of descriptor, for each cell of the grid, as is shown in Equation 4.2.

(e) Select the grid cells that correspond to the tagged ROI in the training dataset to create an array composed of the concatenation of the values of the accumulators of each cell inside it. This corresponds to the descriptor of the object or training vector.

4. Training:

   SVM, with different kind of kernels is used, since it presented good results as a classifier [57, 65].

4.3.5 Detection

In order to speed up the object detection process, a three dimensional matrix for each frame was obtained in order to precompute the grid of descriptors in the overall image. For images of 640 $\times$ 480 pixels and a selected subdivision of 10 pixels, the 3D matrix had 64 rows and 48 columns and 32 values (according to the descriptors size in this case ORB). Figure 4.6 shows this grid. This means that the descriptors corresponding to keypoints inside a square of 10 pixels are accumulated in a 32 element array. Of course, in the areas where there are no keypoints, this accumulation is a zero vector or 32 elements. For searching the object the different window sizes obtained from tagging are used in the grid of descriptors of the image. This search can be done in two ways: by the location of the keypoints in the image or by using the end points of the detected lines. The detection is composed of the following steps:

1. Compute the keypoints in the overall image.
2. Compute the lines in the overal image (optional).
3. Compute the descriptors in the keypoints.
4. Compute the grid of descriptors by using the accumulator.
5. For each point in the keypoint list (or end point in the line list, this is optional):
   - For each ROI size obtained:
     - Compute the feature vector.
     - Use this vector as a SVM's input.

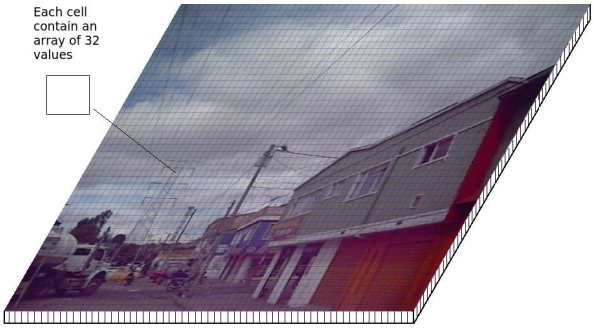


Figure 4.6: Grid generation in an image for detection.

4.3.6 Tracking

In complex environments where different static (trees, fences, walls, houses and buildings) and movable objects (people, cars and trucks) are present in the scene, it is difficult to maintain the detection in all frames of the videos, even more so in a real time approach. For this reason, a tracking stage can be very useful to maintain a detection when the classifier is unable to do so.

Taking advantage of the fact that our method is based on 2D feature descriptors, a matching between the last properly detected tower and the current video frame that can be obtained in a short time was used. This change actually improved the overall process. The results of a matching in order to perform a homography between the last ROI and the present image is used. This was done using the descriptors computed in the keypoints that were contained in the previous ROI, and the overall descriptors were also computed in the actual frame.

4.4 Experiments

4.4.1 Dataset and details

Three different datasets composed of several videos with different kind of towers, perspectives, resolutions, backgrounds, sources and illumination were employed.

In the first dataset, five videos with a resolution of 640 $\times$ 480 pixels and three videos with a resolution of 1280 $\times$ 720 pixels that were obtained using two different video cameras are presented. They were acquired at different locations and illumination conditions.

In the second dataset two videos with HD resolution were recorded in continuous navigation and obtained with different backgrounds, by using an action camera installed on a UAV hexacopter platform.

Finally, in the third dataset, three videos of electrical tower inspection taken from the internet were obtained. These were used for validation.

From these datasets, several sequences of images were obtained for the tagging task. The HD images were resized to 853 $\times$ 480 pixels in order to have a similar image size without losing their proportions and to reduce the computing time.

For the purpose of object background classification, several options in terms of the speed and performance of SVM kernels such as linear, different order polynomial, RFB and sigmoidal were evaluated.

In the evaluation process two experiments were developed: first an evaluation of descriptor performance with selected window sizes for object detection and tracking; second a comparison of performance with other reported approaches.

4.4.2 Evaluation measures

To evaluate the performance of the tower detection and tracking, a ground truth with the tower position in each one of 400 frames of a video was manually tagged. This sequence of frames was not used in the training stage.

The results of the tower position detection were compared with the ground truth and the RMS error was computed, see Equation 4.3, where GT_i is the central position in x or y coordinate of the object in the ground truth, and DT_i the position in the x or y coordinate of the detected object.

$$RMS = \sqrt{\sum_{i=1}^{Number\_of\_frames} \frac{(GT_i - DT_i)^2}{Number\_of\_frames}} \tag{4.3}$$

4.4.3 Setting up

For the development and testing, an HP Envy laptop with a Core i7 processor, 8GB of RAM and Linux Ubuntu 14.04 was used. OpenCV for implementing the computer vision algorithms was employed.

4.4.4 First Test

The first experiment aimed to compare the performance of the detection and tracking of two feature descriptors, the BRIEF and ORB, in the proposed GRID scheme. In this experiment, a set of five window sizes that was obtained from a clustering of window sizes in the training dataset was used. In this case, 3000 tagged images from the first dataset (videos 1,2,3) were employed, and 316721 background images (hard examples) were automatically generated with the established methodology for training. The validation was performed with videos of the other datasets.

4.4.5 Second Test

In the second test, the proposed descriptor called GRID ORB is evaluated with other state of the art approaches. In this case, the GRID ORB results are compared with the HOG results, according to the the approach presented in [26], which is based on the HOG descriptor and Hough transform for line detection.

For the training, the videos of the second dataset were used. The evaluation was done with the third dataset, in which the backgrounds of these videos were different. In order to evaluate a fixed window size, the use of EDLines and CBS for the line detection is used.

The detected segments define a search area for the tower detection. Since the EDLines method has no open source code and it is not posible to change its parameters. CBS is used in order to reduce the number of lines detected by changing its parameters, which also reduced the computing time.

A fixed window size was used to obtain parts of the tagged ROIs used in the training stage. A window size of 64 x 64 pixels was employed because it is the minimal size of the tower presented in the dataset. Also, this size allows to detect a fast response of relevant parts of the tower.

In the detection stage, a bounding box that encloses the single detection is generated for gathering the detected parts on a single rectangle.

A set of 400 tagged images and 147432 generated hard examples were used. For the validation, a internet video that was not used in the training stage was employed.

4.5 Results and discussion

The results were divided into two groups: the first one aimed to evaluate an approach based on a list of windows with representative sizes of the towers in the present dataset and a tracking process. In the second group, an approach based on a fixed window size with a bounding box was evaluated.

4.5.1 First test results

The results of this test were divided into two parts: first, an evaluation of two different kinds of descriptors, BRIEF and ORB in a GRID scheme; second, an evaluation of the descriptors in a tracking scheme.

- Descriptors evaluation:

  The ground truth was obtained by manually tagging the tower in a set of images from a video sequence. The descriptors were evaluated in terms of computing time and the capacity to perform the detection of a tower, as compared with the ground truth. The detection system was evaluated in a previously tagged dataset that was not in the training stage. The central position (X,Y) of the ROI that encloses the tower (ground truth) was compared with the central position (X,Y) of the detected window, as shown in Figure 4.8, where the vertical axis in its subfigures corresponds to the values of the central position coordinates X and Y of the ROI as unidimensional signals and the number of frames is the independent variable. The horizontal axis is the frames. In order to evaluate the behaviour of the detected window size, the changes in each case as another two unidimensional signals were shown. In this case, the window border offset from the central positions in the detection method was compared with the border of the ground truth.

  Is observed that the performance of the object detection with a grid of ORB was better than with a grid of BRIEF, since the trajectories of the detection obtained with ORB were closer to the ground truth than with BRIEF, as shown Figures 4.8(a),(b) and 4.8(c),(d).

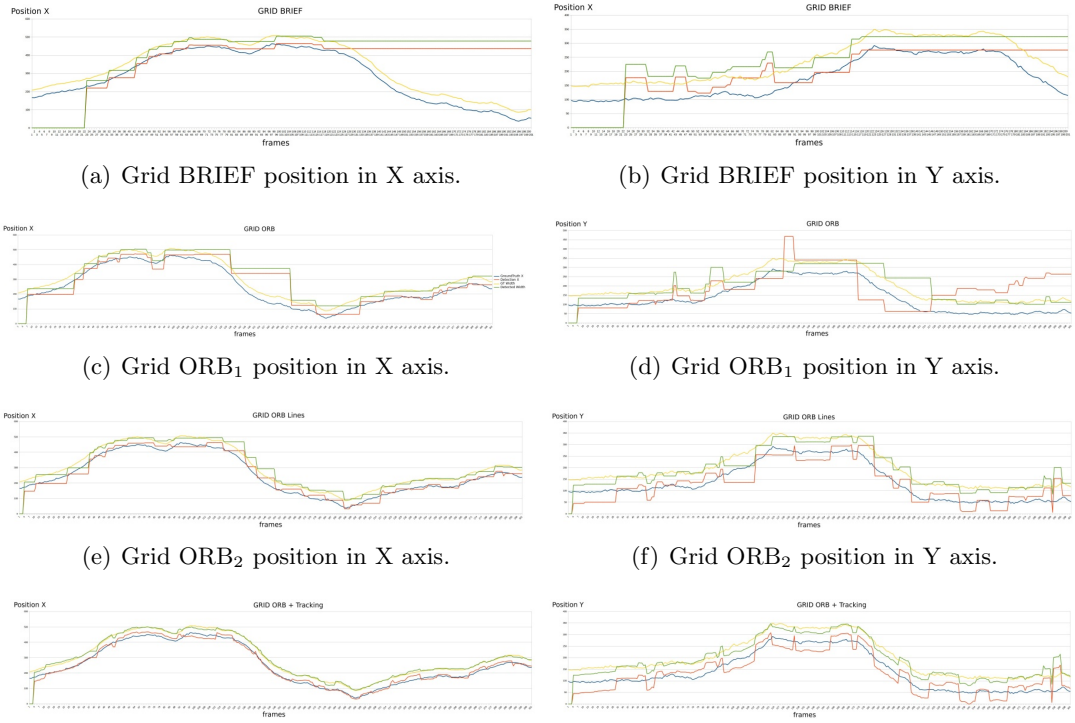
It is necessary to clarify that object detection can be achieved in two ways: first by using keypoints as a place for detection (see in Figure 4.8(c),(d)); and second using the locations of end points of lines as a place for detection (as shown in the third row of Figures 4.8(e),(f)). The detection in each coordinate of the position is compared with the ground truth, as shown in the first two rows of Table 4.3. These results show that using the end points of lines result in a reduced RMS error. The RMS error in the window sizes was obtained and shown in the third and fourth rows of the same table. The results showed that the ORB descriptor presented a good level of performance for object detection and lower RMS error than using BRIEF. This resulted from the rotation capacity of the descriptor, which is very important when images from different perspectives are processed.

The result of the detection using a GRID of ORB in set of frames from videos is shown in Figure 4.7.

Figure 4.7: Result of the tower detection in a set of images using Grid of ORB.

- Tracking evaluation:

  Finally, the overall method was evaluated in terms of the selected descriptor. In order to evaluate the detection scheme, the results of the detection were evaluated by comparison with the ground truth, as previously mentioned. In Figure 4.8(g),(h), it can be observed that the obtained trajectory was closer to the ground truth, as compared to the results obtained without matching of Figure 4.8(e),(f). Additionally, the performance of the window size detected is shown. The RMS error was reduced in this case, as shown in the fifth column of Table 4.3. This highlights the advantage of using descriptor matching as a tracking method.



(a) Grid BRIEF position in X axis.

(b) Grid BRIEF position in Y axis.

(c) Grid ORB_1 position in X axis.

(d) Grid ORB_1 position in Y axis.

(e) Grid ORB_2 position in X axis.

(f) Grid ORB_2 position in Y axis.

(g) Grid ORB_2+Matching position in X axis

(h) Grid ORB_2+Matching position in Y axis

Figure 4.8: Result of the detection using a grid with different descriptors, in each graph the colors means: blue (ground truth), red (our method position in x or y), yellow (size of window in ground truth) and green (size of detected window).

Table 4.3: RMS error related with position and windows size data of Figure 4.8

Coordinate	GRID BRIEF	GRID ORB_1	GRID ORB_2	GRID ORB+Tracking
Error in X	189.91	71.49	29.14	22.10
Error in Y	59.29	96.89	37.86	36.22
Error in Window Width	192.45	68.39	30.72	22.60
Error in Window Heigth	65.17	43.42	28.36	25.41

In this test, the most remarkable result was that the use of the keypoint matching as a tracking method was capable of improving the tower detection performance by using the same descriptors that were used in the detection stage.

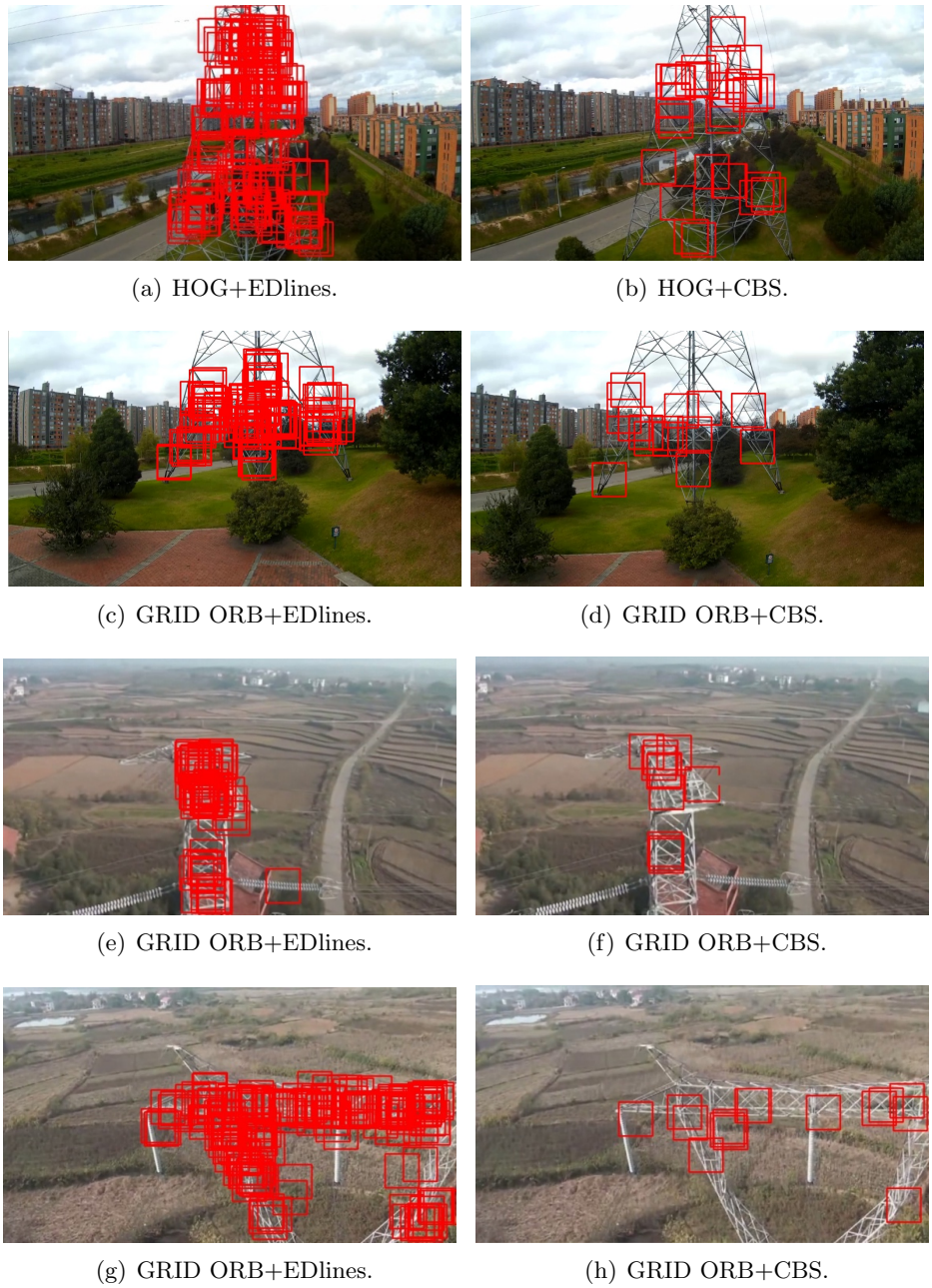
When compared with previous studies, the proposed method was validated in crowded sceneries, where a lot of features can make the detection task difficult. Additionally, the line detection method that was used, detected more lines in shorter times than previous approaches. Finally, it is worth mentioning that the descriptors were not only used for the object detection process, but also for tracking, which reduced the complexity of the process. The computing time for each image of 640 $\times$ 480 pixels was 25 ms for detection and 2.5 ms for tracking. In images of 853 $\times$ 480 pixels, the response time for detection was 40 ms and was 3 ms for tracking, as seen in Table 4.4.

Table 4.4: Computation times for tower detection and tracking

Size	Detection	Tracking
640 × 480	25 ms	2.5 ms
853 × 480	40 ms	3 ms

4.5.2 Second test results

In the second test are presented the results of tower detection by using HOG and GRID ORB. For these descriptors, the use of EDLines and CBS indicated that EDLines detect more relevant lines in the scene than CBS, as shown in Figure 4.9; but the overall process of detection requires more time, see Table 4.5. For this reason, CBS is considered a good alternative within the proposed method.



(a) HOG+EDlines. (b) HOG+CBS.

(c) GRID ORB+EDlines. (d) GRID ORB+CBS.

(e) GRID ORB+EDlines. (f) GRID ORB+CBS.

(g) GRID ORB+EDlines. (h) GRID ORB+CBS.

Figure 4.9: Result of the detection. a) HOG+EDlines; b)HOG+CBS; c),e),f) GRID ORB+EDlines; d),f),g)GRID ORB+CBS.

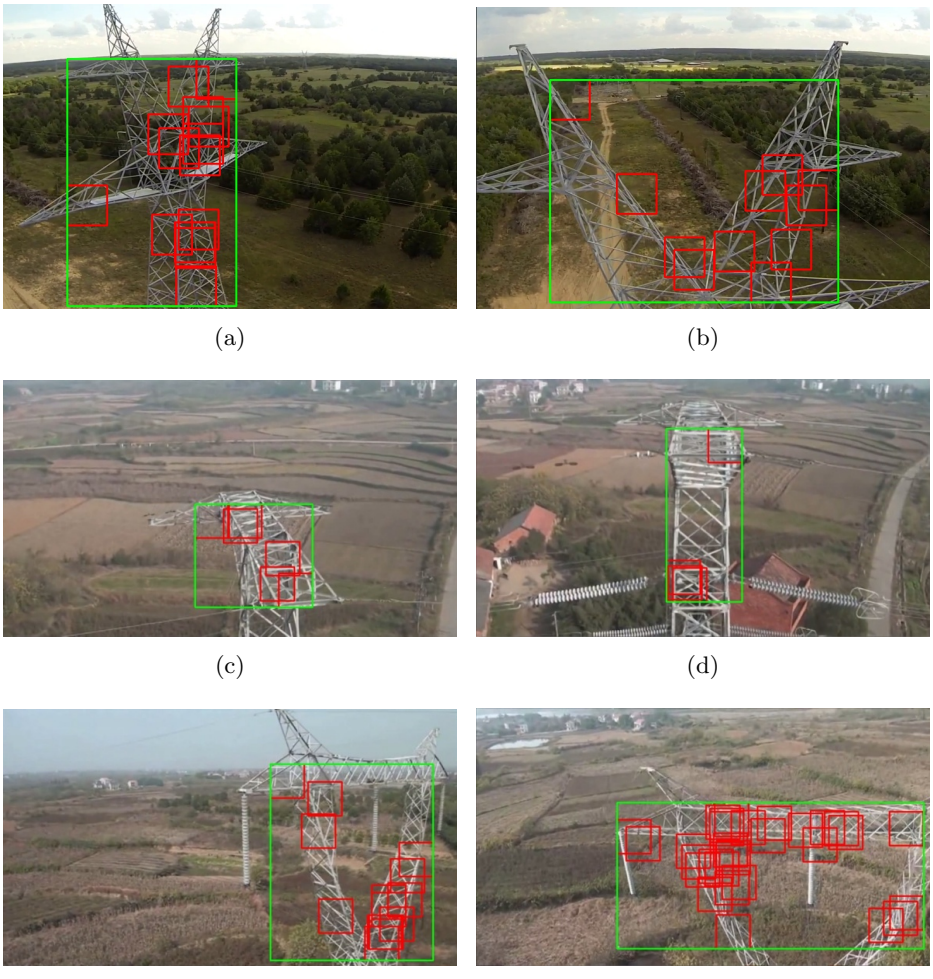
The results of the performance of GRID ORB indicate that it has a good level of differentiation because it had a smaller number of false alarms than using HOG with the described selected data set. Additionally, it was faster than using HOG. The performance, in terms of number of frames with false positives and number of frames where the tower is present but not detected for the first video of the third dataset, is shown in Table 4.6. This portion of the video has a duration of 102 seconds, with 60 frames per second. The tower detector was validated in videos with different backgrounds and presented better results when the background was an open area, as shown in Figure 4.10.

Table 4.5: Computation times using HOG and GRID ORB

Method	Size	EDLines	CBS
HOG	640 × 480	160 ms	51 ms
HOG	853 × 480	250 ms	65 ms
GRID ORB	640 × 480	95 ms	37 ms
GRID ORB	853 × 480	188 ms	45 ms

Table 4.6: GRID ORB and HOG for tower detection performance

Method	GRID ORB	HOG
FP	15	165
FN	22	3



(a) (b) (c) (d) (e) (f)

Figure 4.10: Result of the detection using a GRID ORB + CBS + bounding boxes in different backgrounds.

4.5.2.1 Precision Recall

The precision recall values were obtained in testing two videos using the follow expressions:

$$Precision = \frac{tp}{tp + fp}, \tag{4.4}$$

$$Recall = \frac{tp}{tp + fn}. \tag{4.5}$$

In the first video a sequence of 1800 frames is used. In the second a sequence of 200 frames is selected. The results are presented in the table 4.7.

Table 4.7: Precision recall

Video	TP	FP	FN	Precision	Recall
video 1 (UAV)	1710	90	90	0.95	0.95
video 2 (internet)	192	8	8	0.96	0.96

4.5.2.2 Window precision

Th intersection of AABB (axis aligned bounding boxes) of the tagged image and the detected ROI of the tower according with the Figure 4.11 was implemented as following:

$$Iarea = max(0, max(maxAx, maxBx) - min(minAx, minBx)) * max(0, max(maxAy, maxBy) - min(minAy, minBy)) \tag{4.6}$$

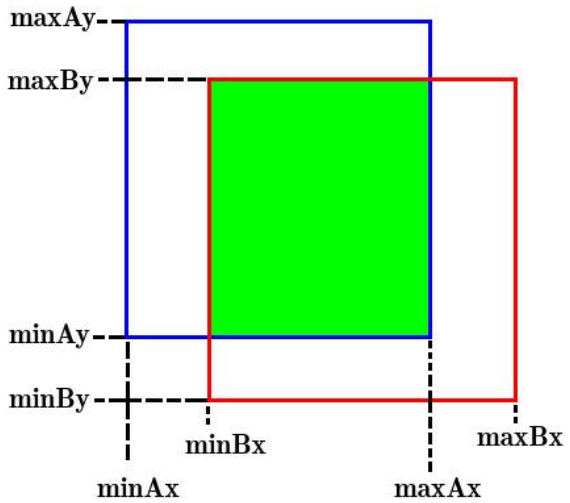
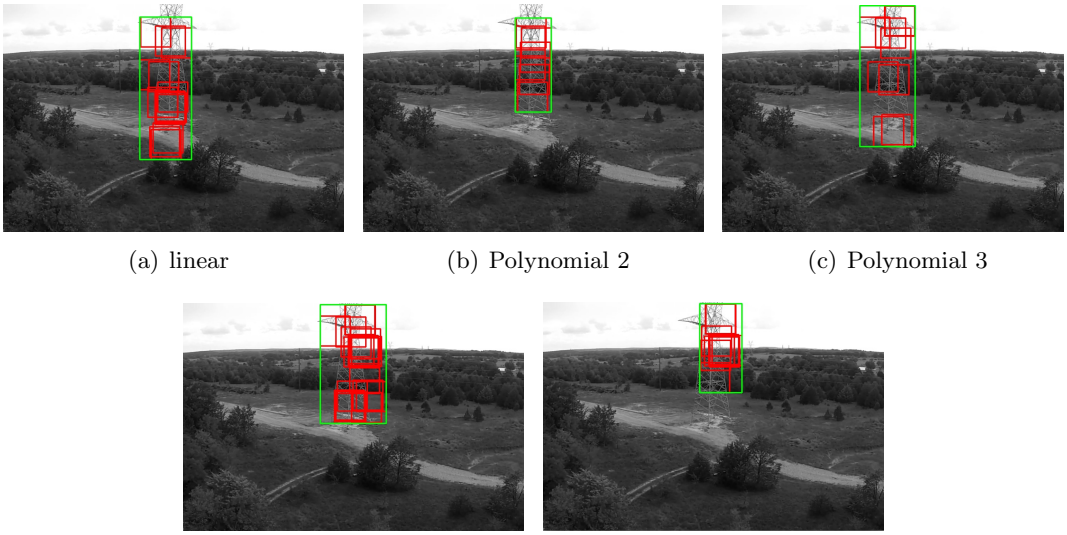


Figure 4.11: Intersected area of AABBs.

In this analysis the overlapping area is close to 83 %.

4.5.2.3 SVM analysis

In the Figure 4.12 the results of detection with different kind of kernels (linear and polynomial of 2, 3, 4 and 5 degree) are shown. In this case the selected parameters are $\gamma = 20$, $\epsilon = 1$, $coef_0 = 1$ and $C = 7$. In this case the linear kernel is faster than others and the best detection is obtained with a polynomial kernel of four degree.



(a) linear (b) Polynomial 2 (c) Polynomial 3

(d) Polynomial 4 (e) Polynomial 5

Figure 4.12: Result of the tower detection with different kernels.

Table 4.8: Kernels evaluation

Parameters	Linear	Poly 2	Poly 3	Poly 4	Poly 5
Squares detected	13	7	9	18	9
Time (ms)	38	151	135	120	104

4.6 Conclusions

In this chapter, an efficient method for tower detection based on line detection and 2D feature descriptors for a real time application was presented. A methodology for generating the training data in an object detection scheme is designed.

The GRID of feature descriptors is a good method for describing electrical towers in an object recognition system. This method permits the detection of electrical towers in real time and is robust when detecting within blurred images. This method could be adapted for a visual control system to navigate towards an electrical tower in order to perform an inspection. On the other hand, the method can be used to avoid areas where electrical towers are located in order to reduce the risk of collision.

The proposed method takes a short time for training and detection. This GRID of feature descriptors allows for the detection of electrical towers in a time close to 45 milliseconds for 853 $\times$ 480 pixel images, but it could be less depending on the complexity of the scene.

The use of the keypoints and descriptors obtained in the detection stage is very useful for performing a matching with the last detected frame. This helps to improve the detection and to maintain the tower in the ROI when abrupt changes of movement occur in UAV navigation.

The computing time was shorter when a SVM with linear kernel was used for the classification, but the detection rate was better when a polynomial kernel was employed for classifying the transmission towers in crowded environments. These results were validated with several videos of transmission towers taken from different perspectives. Finally, the performance of the ORB descriptor for tracking was evaluated.

A future work on the implementation of object detection and a tracking system in a UAV platform is expected.

CHAPTER 5

Simulation of UAV navigation in virtual environments

This chapter has the purpose of showing a step previous a real implementation of the UAV navigation system. Since we are using ROS (Robot Operating System), the experience in this environment is an important task previous to a real implementation. In this chapter, two kinds of simulations are presented. The first one is related to the navigation of UAV by using power lines as a reference. The second one is focused in using the electrical towers as a reference for a frontal camera. The navigation is performed by using the information extracted from virtual cameras in a visual control scheme.

The cost of obtaining images of power lines from different perspectives is high, since it depends on performing manned flights. Logistic problems that imply carrying of equipment to distant areas that could be of difficult access and the lack of regulations on UAVs, makes it necessary to use as much as possible alternatives of flight simulations in virtual environments.

5.1 Related Work

The development of a UAV for power line inspection system requires to perform many UAV navigation tests. It is necessary to find alternatives to avoid damages to the UAV platform by doing several tests in environments that have similarities with the real ones. This can be done by developing virtual environments for simulating under different conditions. This kind of simulations comprise dynamical models of UAVs, as shown in [66]. This offers possibilities to start operations avoiding damages to the UAV, as well as the incorporation of different flight platforms and scenarios. There are some platforms for UAV simulation, such as USARsim [67] [1]. This has been used for SLAM (Simultaneous localization and mapping), as shown in [68, 69].

Another option is to use ROS (Robot Operating System)[2] based simulator, as shown in [66], which is widely used since it is an open source and it has a big amount of libraries, including computer vision.

[1] `usarsim.sourceforge.net`
[2] `http://www.ros.org`

For the development of this part of the project, we decided to use a ROS gazebo simulator called Tumsimulator, developed at the Technical University of Munich[3], which permits to simulate different kinds of robots and its environments. This simulator can represent important aspects of the UAVs such as the sensor information and specially the camera, since it allows to develop and validate computer vision techniques for visual servoing schemes.

5.2 Visual based navigation with power line detection

Due to the high costs of obtaining images of power lines from different perspectives in electrical infrastructure and the logistic problems of manned flights, it is useful to develop methods for visual based navigation by using UAVs. For this reason, visual based navigation strategies for UAV power line inspection are presented; a virtual environment for real time simulation was developed and line detection methods were integrated and validated within the virtual environment.

From the point of view of visual control, it is very important to have the possibility of obtaining camera views from different poses in a short period of time, for a close loop control and to show results in real time.

5.2.1 Virtual environments construction

The simulation environment is composed of terrain, towers, poles and power lines. For visual modelling Blender 3D was used in order to create the shapes of the towers and mountains with textures. In Figure 5.1, the 3D objects developed are shown.

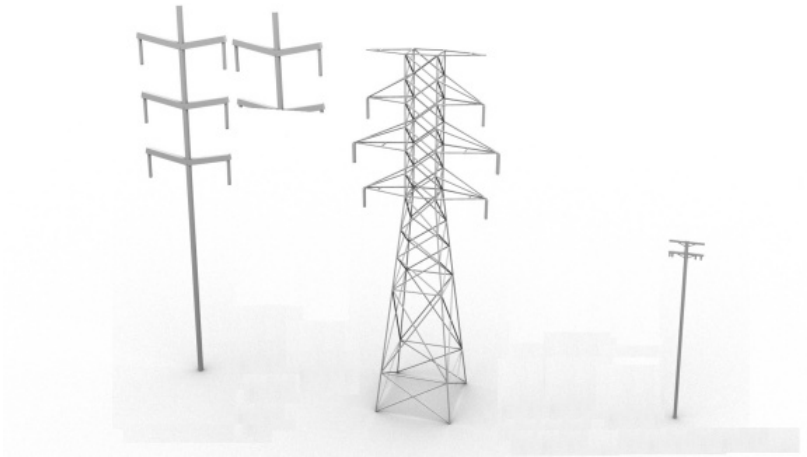
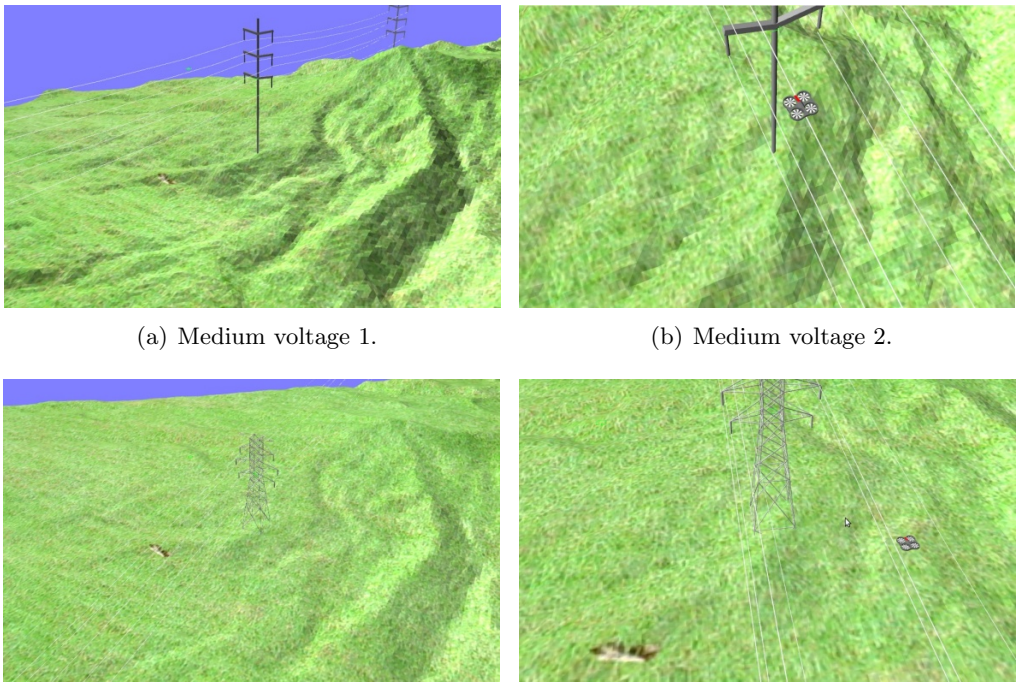


Figure 5.1: 3D Models of different kind of towers and poles.

One advantage of working with a virtual environment is the possibility of creating different configurations for UAV navigation. In this work we developed two principal environments a medium voltage and a high voltage power line one. In figure 5.2, the developed environments are shown.

Power line inspection, by using UAVs, requires the development of different stages; we propose two algorithms for this process. The first one focuses in the initial pose of

[3] `http://wiki.ros.org/tum_simulator`



(a) Medium voltage 1. (b) Medium voltage 2.

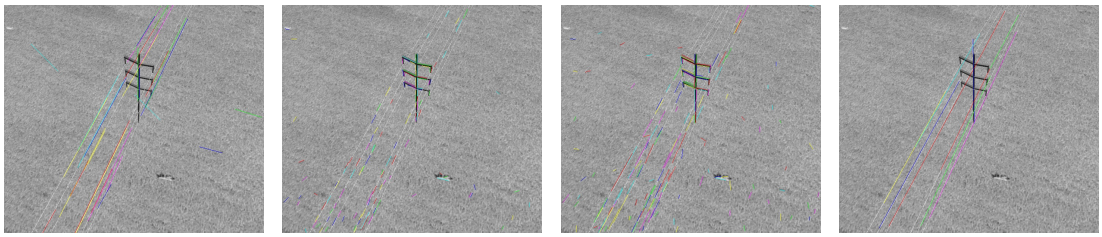
(c) High voltage 1. (d) High voltage 2.

Figure 5.2: Developed environments.

an UAV with respect to the power lines, and the second one focuses in following the power lines. It is fundamental for these algorithms that the power lines have been correctly detected. In this case, we use an image based method for line detection.

5.2.2 Line detection

Line detection is not only useful for power line detection, but also to have an initial idea of where vertical lines that represent poles or towers can be found [25]. This can be used in the process of driving an UAV upwards through the electrical infrastructure in order to pose it in a suitable position for inspection activities. Some examples of line detection are shown in Figure 5.3



(a) Hough. (b) LSD. (c) EDLines. (d) CBS.

Figure 5.3: Comparison between different methods for detection in virtual images.

With the information of extracted lines, it is possible to have elements for differentiating power lines from other lines detected. The main elements are the length of

lines, the direction with more long lines and the structure of the lines; these must be a group of parallel lines.

5.3 Architecture of the simulated system

The UAV for the system is a multi-rotor quadcopter provided with a control system for outdoor stabilization. The advantage of using a multi-rotor is that it is possible to fly near to the inspected area and to get close details. Another advantage is the possibility of a vertical take off. The proposed system is equipped with two cameras, the frontal and the bottom one. The purpose of the bottom camera is to detect the power lines. The frontal camera is required for driving the UAV towards the pole or the electric tower. For developing and testing, an Asus G53S laptop with a processor Core i7, 8GB of RAM and provided with Ubuntu 12.04 was used. For this process, an architecture of processing that involves sensors, actuators and image processing is proposed. The process used the information of UAV sensors and the information extracted from the frontal and bottom cameras. The information of vertical lines of poles at low altitude and the information of horizontal lines at high altitude is extracted from the frontal camera. The information about power lines is extracted from the bottom camera. It is necessary to define an architecture that integrate the cameras, flight controller and computer vision techniques in a task planner. With the information obtained from the cameras the power lines are detected. The information have to be sent to the flight controller (auto pilot). In Figure 6.4, the principal components of the system architecture are presented.

5.4 Strategy for UAV power line navigation

In Figure 5.5, the complete process for autonomous navigation is shown. The strategy is composed of the following stages:

- Initial pose of UAV with respect to the power lines.
- Power line following.
- Return.

5.4.1 Initial pose of an UAV with respect to power lines

In this stage, the UAV has to take off from a suitable position; this means that the UAV has to be located with its frontal camera directed to a pole or tower. The UAV should not be located under the power lines, since it can crash when going upwards. The UAV has to be located offside of the power line area, as it is shown in figures 5.6(a) and 5.6(b). The purpose of this process is to locate the UAV over the power lines in order to begin the inspection. This can also help to correct the UAV position in the case of loosing the trajectory of the power lines by using the position of a pole or tower in the frontal camera as a reference. As a precondition, the UAV must be placed in a side of the power lines in the direction towards a pole or a tower. The steps of this stage are the following:

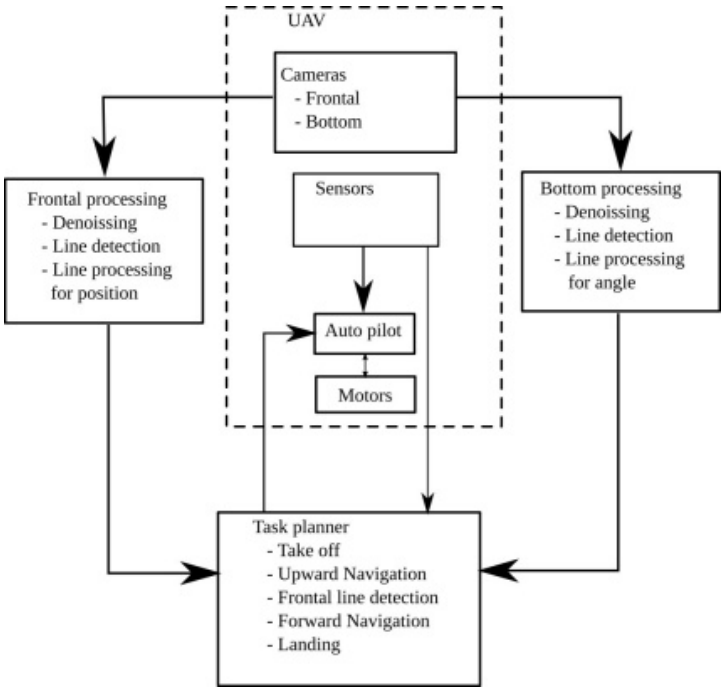


Figure 5.4: System architecture.

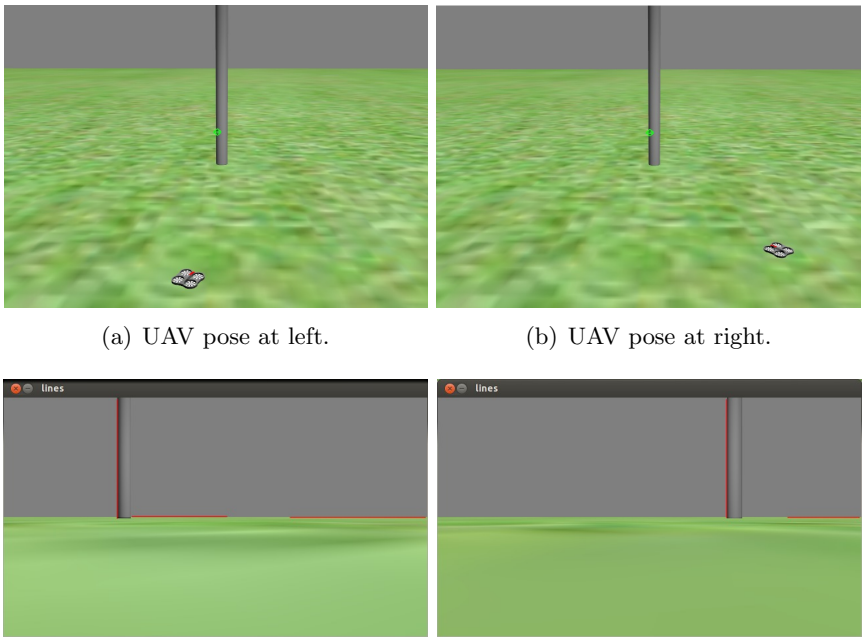
Figure 5.5: Complete process.

- UAV take off.
- Apply a noise reduction filter.
- Segment the image with an edge detector.
- Detect vertical lines from camera.
- Detect the tower or pole with the frontal camera. In this part the distance from the pole to the middle b (offset) of the image is obtained by vertical line detection (see Figure 5.7). The angle α has to be zero when the UAV is aligned to the pole.
- Drive the UAV upwards by using the previous detection as a central reference.
- When the UAV achieves an altitude value higher than the height of the towers, make a rotation of the UAV until the lines are detected in the frontal images.
- Move the UAV towards the horizontal lines detected.

5.4.2 Go over the power line direction

The goal of this stage is to navigate over the power lines, taking them as a reference for the visual control system.

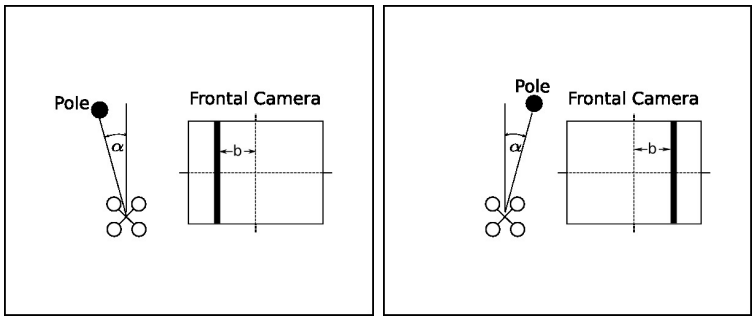
1. Take an image from the bottom camera.



(a) UAV pose at left. (b) UAV pose at right.

(c) Frontal image from left. (d) Frontal image from right.

Figure 5.6: UAV pose and rendered images from the frontal view.

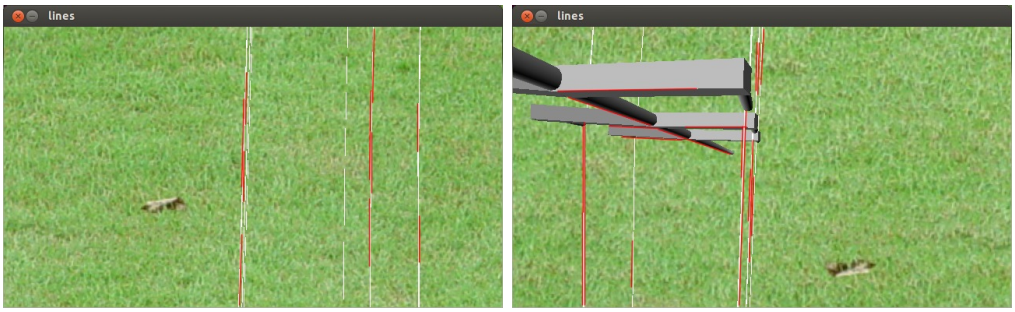


(a) Offset left. (b) Offset right.

Figure 5.7: Control variable in a frontal view.

- Apply a noise reduction filter.
- Segment the image with an edge detector.
- Detect power lines (see Figure 5.8).
- Compute the angle of the lines with respect to the vertical.

2. Use this angle as an input to the yaw control (See Figure 5.9). In this part a proportional control is used in order that the UAV follow the power line. The feedback variable is the difference of the UAV yaw angle with the angle in the image.
3. Compute the distance between lines.
4. Command the UAV forward (See Figure 5.10).
5. Go to step 1.



(a) Lines detected. (b) End lines.

Figure 5.8: Power line detection in video.

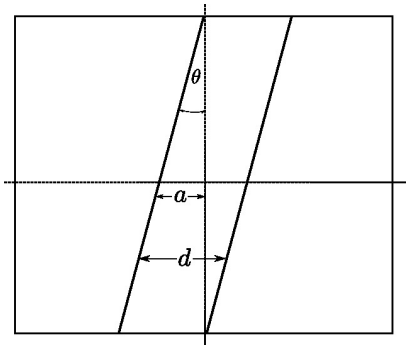
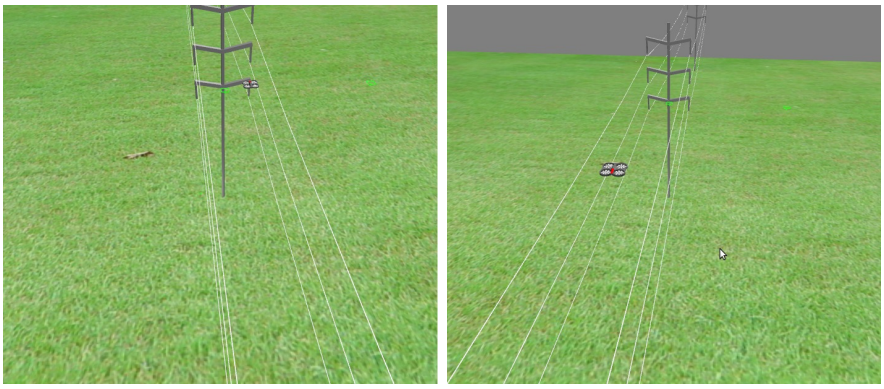


Figure 5.9: Control variables from the azimuthal view image.



(a) Navigation forward. (b) Navigation backwards.

Figure 5.10: Navigation over power lines.

5.5 Visual based navigation with tower detection

The main problem of navigation systems based in power line detection from a camera pointing downwards is the difficulty to detect power lines in real time in complex backgrounds, for example over a road or a fence. It becomes complicated to differentiate from other linear elements even more if a heterogeneous background is presented. For this reason, we consider that a visual based navigation system for power line inspection can be helped by using the tower as a reference.

There are several methods for pylon or tower detection. In the work of Golightly [60], corner detection and matching was applied for power pole detection. In the work of Cheng [25] a segmentation based in a graph cut algorithm for pylon detection was presented. Tilawat developed a method based in an the optimal IIR filter and Hough transform for line detection. More recently, [26] developed an strategy for electric tower detection and tracking by using computer vision and machine learning techniques that includes Hough transform, HOG descriptors, bounding boxes and neural networks.

There is recent interest in the advantages offered by 2D HOG descriptors because their capacity of representation. I consider it is useful to take advantage of 2D features as a method of object detection in virtual environments because they can be obtained in relevant parts of the image by using feature detectors and can be extracted in short time which can be useful for controlling a robot.

The main goal of this section is to present a process for UAV navigation based on electrical tower detection as a preparation for a real implementation. For this reason a set of virtual environments was built. In addition, a method for tower detection in synthetic images with low computational complexity was developed.

5.5.1 Tower detection in virtual environment

In order to perform a navigation based on visual cues extracted from distinctive parts of the electrical infrastructure, a method for tower detection based on visual cues extracted from distinctive parts of the electrical for infrastructure in synthetic images was developed. This method is based on 2D descriptors. A 2D keypoint detector has to be used in order to obtain relevant points in the image.

As a preliminary step on this process a set of images that contain towers have to be labelled. We selected 10 images with different poses of the UAV respect to the tower. This is done by selecting a region of interest (ROI) which is composed of the coordinates of a rectangle *(minx, miny maxx, maxy)*. The keypoints and their descriptors inside this ROI are stored. This using a vector of lists of keypoints and a vector of list of descriptors.

In the work of [22], some keypoint detectors were compared being FAST [37] a good alternative in a matching context. This is a very good method in order to find relevant points in images in short computing time. For this reason in the set of ROIs, a set of keypoints is obtained by using FAST.

There exist different descriptors such as SIFT [31], SURF [32], BRIEF [33], ORB [34] and BRISK [35], which have been evaluated with different metrics, specially for matching. According to the work of [22], BRIEF is a little bit faster than ORB and much faster than others, showing a very good performance when used with images that have similar scales and orientations. For this reason, ORB is used to obtain as a descriptor in the selected keypoints in a second step.

In the last step, the keypoints and descriptors have to be obtained in the current frame of a video that has to be used as an input for the tower detection method. The descriptors of this frame are matched with each of one of elements of the vector of keypoints and the vector of descriptors. After that, the best matches for each one of elements of the vector lists are selected depending of the number of matches

and the value of standard deviation on the position. The position of the tower is extracted as Equation 5.1.

$$xc = \frac{1}{N}\sum_{i=1}^{N} mkp_x(i),$$
$$yc = \frac{1}{N}\sum_{i=1}^{N} mkp_y(i). \tag{5.1}$$

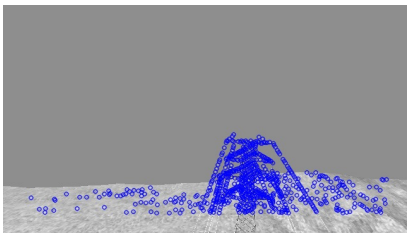
Being mkp_x, mkp_y the coordinates of each one of the list of selected N keypoints.

5.5.2 Visual navigation process using towers

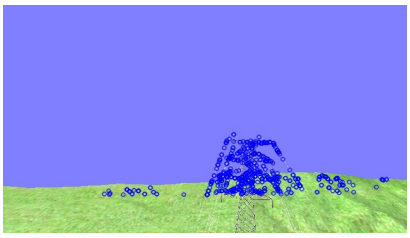
The purposes of this process are to locate the UAV over the power lines and to orient it toward the electrical towers changing its yaw angle. The position of a pole or tower is detected with the information of the frontal camera and it is a reference for driving the UAV towards the electrical infrastructure. The images obtained have a resolution of 640×320 pixels. The process of tower detection begin with a keypoint extraction; after that, matching and finally the selection of the ones with less standard deviation is done. This is shown in Figure 5.11.

The navigation process begins by driving the UAV towards a suitable pose to start the navigation based on tower detection. After tower detection, the UAV pose correction is done in a close loop. The main steps for the navigation based on tower detection are the following:

1. UAV take off.
2. Drive the UAV upwards.
3. Stop when the UAV achieves an altitude value higher than the height of the towers.
4. Tower detection as mentioned in Section 5.5.1.
   (a) Capture a frame in grayscale.
   (b) Apply a noise reduction filter.
   (c) Compute FAST detector.
   (d) Extract ORB descriptor
   (e) Make matching with database.
   (f) If there are enough matches and standard deviation is less than a threshold.
      – Compute (xc, yc) .
   (g) Else Rotate UAV in yaw angle and return to (a).
5. Make corrections in yaw angle of UAV based in the position of tower (xc, yc).
6. Move the UAV towards the detected tower and go to step 4.



(a) Keypoint extraction.



(b) Matching.

(c) Selection.

Figure 5.11: Object detection process.

5.6 Results

A virtual environment of power line infrastructure composed of realistic elements in terms of shape, color and size was developed.

An integration of the virtual environment of simulation with the computer vision techniques was developed. The processes of segmenting and line detection were obtained in real time. The information obtained through video processing allowed the UAV to navigate in the virtual world by using a closed loop control.

For the first strategy, the performance depended on the initial pose of the UAV. In this case, better results were obtained when the UAV was located in an isolated area. For the second strategy, the UAV could navigate following the power lines without loosing its trajectory since lines were detected.

The results shown that the UAV was able to follow the lines when navigating over them using line detection. Also, the towers was detected with the frontal camera.

5.7 Conclusions and future work

A different kind of visual feature based navigation processes for a multi-rotor UAV were proposed, implemented and tested by using simulation in virtual environments.

Some strategies for multi-rotor navigation were proposed and evaluated in virtual environments. These can be extended for other inspection areas and improved by using object detection.

A three dimensional model of environments with the power lines was built in order to generate synthetic images with power lines and towers with different points of view. Different configurations of the scene were created for validating computer vision techniques successfully.

The use of simulation in virtual worlds is very useful since it allows to anticipate problems that could happen in the real world. This can prevent the damage of UAV platforms. In the same way, it allows to save resources required for manned flights.

An integration of virtual environments of simulation with the computer vision techniques was performed. The processes of tower detection is obtained in real time and allowed to correct the UAV orientation. The information obtained through video processing allowed the UAV to navigate in the virtual world by using a closed loop control.

The detection process based on the ORB descriptor takes short time of computing and is able to detect the tower in different perspectives. This process can be improved by using machine learning methods for real images or more complex environments.

The navigation process can be improved by combining the information of the tower detected through a frontal camera with the power lines detected using a bottom camera and a task manager which is in charge of combining different types of flight and sensor information.

CHAPTER 6

Onboard visual navigation system for power line following

In this chapter, the details of an onboard visual navigation system that allows an UAV to perform a power line following are presented. In this process a method for obtaining the main angle orientation of power lines called the Histogram of Oriented Segments (HOS) was developed. The HOS is used not only for obtaining the angle, but also for segmenting lines that are not power lines in images. The different components of the flight system are presented. This includes the platform configuration and the hardware used. After that a set of tests were designed to validate our methods and the overall system. We tested first over a sequence of images taken from different points of view in real inspection tasks. Additionally, we tested with images taken using an UAV platform with a camera pointing vertically downwards. Then, the developed techniques for UAV navigation were employed over power lines by using a multirotor. The results of these experiments are reported in the present chapter.

6.1 Histogram of Oriented Segments

In a power line detection process from an azimuthal view, the following aspects can be highlighted aiming at improving the detection.

- The length of the line. It is expected that the longest lines detected after the linking stage presented at the scene are the power lines.
- The direction with the higher number of long lines.
- The structure of the lines. They must be located in a group of parallel lines that has a symmetrical arrangement. This means that the distance between lines remains proportional from an azimuthal view.

Following the mentioned statements, in this work, a measure based on the linear content of the image is proposed. The measure is obtained from a Histogram of Oriented Segments (HOS) that has three purposes: Firstly, obtaining of the orientation angle of the power lines for UAV navigation. Secondly, selecting the power lines in

the scene. Thirdly, defining a way to differentiate between a scene with and without power lines.

As a prerequisite the image has to be taken, when possible, from an azimuthal view, and the radial distortions must be corrected. In this case, it is assumed that the main linear components in the scene are power lines. This histogram is defined in the interval from 0 to 180 degrees. The main idea is to obtain a histogram of line segment angles that are pondered with the length of each line.

Let (x_i, y_i) be the coordinates of the initial point and (x_e, y_e) the coordinates of the end point of a k segment.

The following are the steps to obtain the HOS:

1. Obtain lines in the overall image. The segments detected are collected in a list of M elements.
2. Create an array H of 180 elements which corresponds to angles in the interval $[0, 180)$.
3. Initialize the array with zeros.
4. For each line segment $line_k$
   - $dx = x_e - x_i$.
   - $dy = y_e - y_i$.
   - $length = norm(line_k)$.
   - $\theta = ComputeAngle(dy, dx)$.
   - Correct θ to interval $[0, 180)$.
   - Use this angle as index in the histogram and accumulate the length of line. $H(\theta) += length$.

The $ComputeAngle(dy, dx)$ function is obtained by using the $atan2(dx, dy)$ function, and is corrected in order to have angles between the interval $[0, 180)$.

$$angle = \begin{cases} ang - 180 & \text{if } ang > 180 \\ ang + 180 & \text{if } ang < 180. \end{cases} \tag{6.1}$$

6.1.1 Main angle

Due to the fact that the histogram of oriented segments contains the most representative angles that are present in the scene, it is possible to obtain the maximum values from it. These values are an estimation of the line orientation, as shown in Figure 6.1. After the histogram is obtained, the main angle is extracted as shown in Equation 6.2. In order to illustrate the process, a set of 8 images (see Figure 6.2) was tested. The main angles of the set of images of Figure 6.2 are shown in the Table 6.1.

$$\underset{\theta}{\operatorname{argmax}}\, H[\theta] = \{\theta | H[\theta] = \max_{\theta'} H[\theta']\}, \tag{6.2}$$

where θ is the angle between the power lines and the vertical axis of the image.

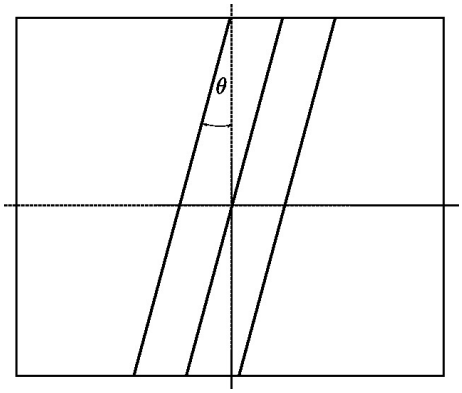
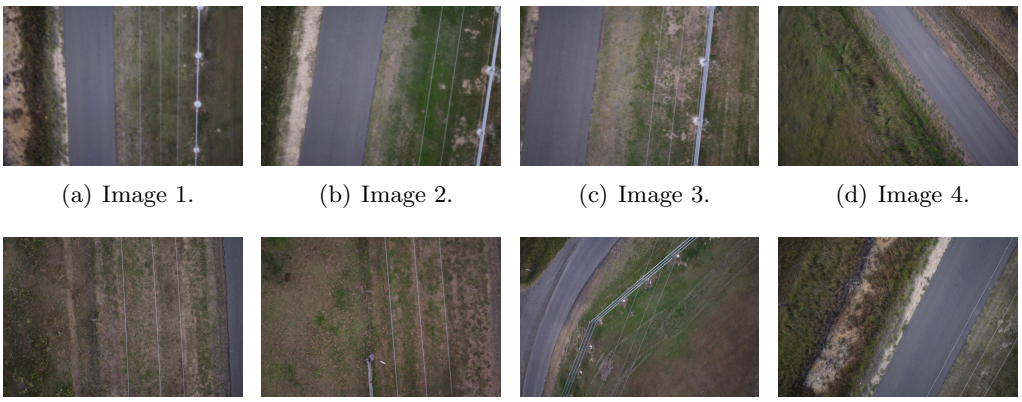


Figure 6.1: Main angle of power lines from azimuthal view.



(a) Image 1. (b) Image 2. (c) Image 3. (d) Image 4.

(e) Image 5. (f) Image 6. (g) Image 7. (h) Image 8.

Figure 6.2: Images of power lines taken from an azimuthal view with different orientations.

Table 6.1: Main angle detection in images of Figure 6.2.

Image	1	2	3	4	5	6	7	8
Angle	-1°	8°	6°	-41°	-3°	-5°	22°	28°

6.1.2 Line segmentation

Once obtained the main angle θ, the image can be segmented by selecting the lines with similar angles. This way only lines with angles close to de main angle are selected.

6.1.3 HOS as a global descriptor

The information obtained with the HOS is useful not only for obtaining the main angle of the power lines but also for discriminating whether power lines exist in the scene or not. It is possible to use the HOS in a scene in order to estimate if there are power lines present in the scene. This is done by comparing the maximum value of

the histogram with a threshold that is obtained when power lines are detected. The angle estimations are selected only when the maximum of the histogram is greater than a threshold which is obtained experimentally from our dataset.

6.2 UAV System description

In this work we used a low cost hexacopter platform type DJI 550 that was adapted to carry the equipment (see Figure 6.3). In order to enable it to perform the navigation tasks that included vision processing and the position control commands, an onboard companion computer was employed. In this case the computer was a ODROD U3 with an ARM processor. We used a pixhawk flight controller of 32 bits, that was controlled through the companion computer. The system run under linux Ubuntu 14.04. We used OpenCV for implementing computer vision algorithms and ROS Indigo to integrate with the robotic platform.

For the vision system, we tested with two USB cameras. An action camera Mobius (equipped wide-angle lens) and an ELP industrial camera USBFHD01M (see Figure 6.3(b)). Although it is possible to obtain the calibration parameters of the action cameras, the radial distortion is difficult to reduce. The detected lines appear as curves, specially at the extremes of the image. With the obtained main angle from our line detection method it is possible to get an estimation of the line orientation. In order to get better results in terms of the rectification of images we used the ELP camera. This device provided good results in the line detection process plus a good frame rate. The UAV system consisted on a set of related components that are shown in Figure 6.4.

(a) Hexacopter.

(b) Camera pointing vertically downwards

(c) Test flights.

(d) Test flights 2.

Figure 6.3: UAV platform and flight tests.

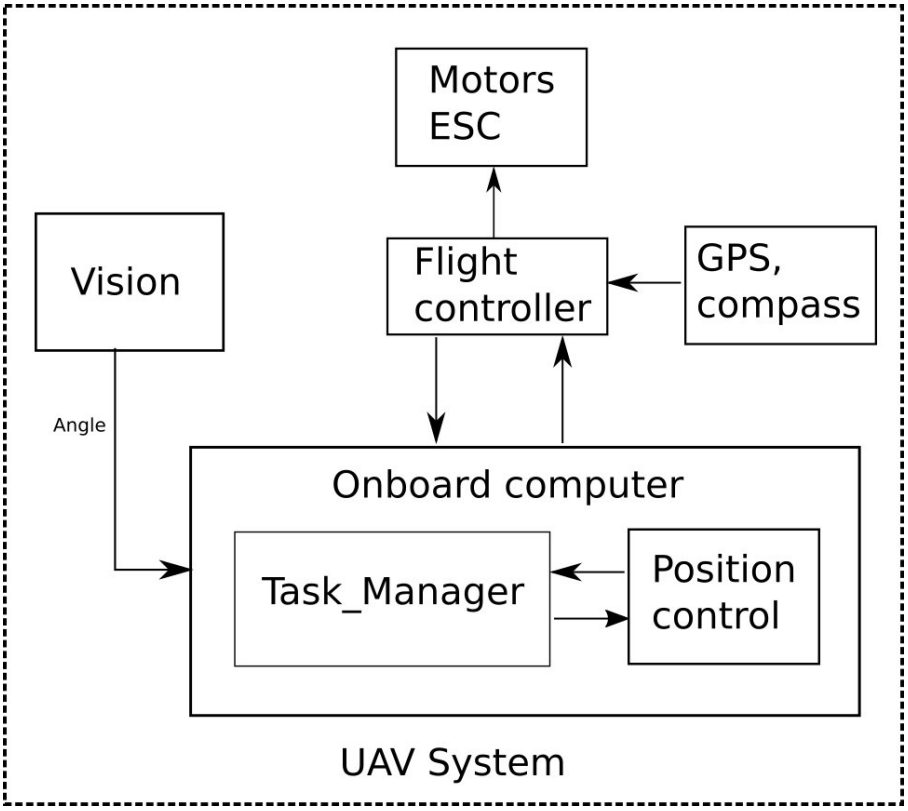


Figure 6.4: UAV System.

6.3 Autonomous navigation system

In this work, the tasks involved in a mission of power line navigation are coordinated by a task manager (see Figure 6.4). The task manager receives data through sensors that included accelerometer and GPS. Also it uses high level information that is obtained from the image provided by the camera. This information is extracted by the developed visual modules with the proposed algorithms providing the inputs for the control commands sent to the flight controller. In this case, we used a position control interface for the pixhawk controller that receives the desired commands from the task manager in the UAV mission execution.

For the implementation, a set of ROS (Robot Operating System) based modules have been developed. The most representative procedures are extended in algorithms 6.1, 6.2 and 6.3. The main modules are the following:

- Vision module. This module performed the video capture, image smoothing, camera calibration, edge segmentation, line detection, histogram of oriented lines and the computing of the main angle.
- Navigation task manager. This module determined the set of autonomous actions to be performed by the UAV during a flight mission. The main tasks were: Take off, Ascend (vertical flight), Rotate, Move_forward, Move_until (see Algorithm 6.1), Align (see Algorithm 6.2), Follow lines (see Algorithm 6.3), Return, Descend, Land. In these algorithms, the initial *yaw* orientation of the UAV respect to the north is θ_0. The *angle* variable is the orientation in UAV

coordinates. In the align algorithm the control loop aims to reduce the angle error. This loop continues until the absolute value of error is less than a threshold ϵ.

In the designed high level architecture, a proportional control is implemented based on the main angle of the detected lines in the image in order to align and following lines. In this case an incremental error is considered. The scheme of the visual servoing system is presented in Figure 6.5. In this case a image based visual servoing was implemented. This control uses the angle obtained in the vision module. In a low level part a position control receives the reference for power line following in order to do a safe flight.

This kind of control was used because the addressed problem consider only the power line detection from a top down view in order to simplify the flight strategy and avoid collisions.

In this Figure the computer vision techniques are located in the block that receives the image from the camera.

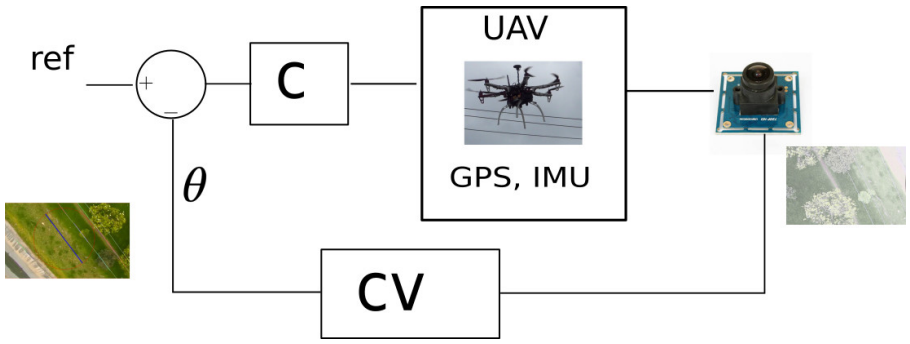


Figure 6.5: Control System.

Algorithm 6.1: Move_until

```
angle ← 0
θ ← angle + θ0
iter ← 0
while not_linedetection and iter < maxiter do
    dx ← r · cos(θ)
    dy ← r · sin(θ)
    x ← x + dx
    y ← y + dy
    iter ← iter + 1
    Set(x,y,z,angle)
end
```

6.4 Experimental Setup

Different experiments have been conducted in order to validate the proposed approach for power line following that includes the CBS line detection method and the HOS algorithm. The experiments were divided in two types. The first included

Algorithm 6.2: Align

error $\leftarrow -angle\_image$
while $|error| > \epsilon$ **do**
  $angle \leftarrow angle + error \cdot P$
  error $\leftarrow -angle\_image$
  Set(x,y,z,$angle$)
end

Algorithm 6.3: Follow_lines

iter $\leftarrow 0$
while $iter < distance$ **do**
  $\theta \leftarrow angle + \theta_0$
  $dx \leftarrow r \cdot cos(\theta)$
  $dy \leftarrow r \cdot sin(\theta)$
  $x \leftarrow x + dx$
  $y \leftarrow y + dy$
  error $\leftarrow -angle\_image$
  $angle \leftarrow angle + error \cdot P$
  iter $\leftarrow iter + 1$
  Set(x,y,z,$angle$)
end

a dataset of a real power line inspection process. The second type gathered the experiments that used an onboard visual navigation system for power line following, with the presented algorithms.

6.4.1 Dataset

This work used two datasets of power lines captured using aerial vehicles. Firstly, a subset of 800 images from a video sequence of a real inspection task taken from a helicopter. This information was used to compute the main angle with the HOS.

Secondly, a set of 1097 images of a video sequence taken with a UAV from a camera pointing vertically downwards in continuous flight during the line following process.

6.4.2 First test: Data collection of power lines inspection

In this case we used images taken from different perspectives from a power line inspection process in continuous flight as shown in Figure 6.6.

6.4.3 Second test: Lines following onboard vision system

The proposed approach was tested with the companion computer equipped with the autonomous navigation system. The system included the CBS algorithm and the HOS process execution. In this case the UAV performed a navigation over power lines. Firstly, the UAV is located close to the power lines before taking off. Also it has to be orthogonal to the lines. Secondly, the UAV flight vertically until a

Figure 6.6: Some images from a power inspection sequence.

predefined altitude. After that, the UAV make a horizontal flight to detect the lines with the camera pointing vertically downwards. The camera shots frames with resolution of 640×480 pixels. An experiment of a power line following mission was prepared. The mission consisted of the following tasks: take off, ascend, move_until, align, follow_lines, rotate(180°), follow_lines, rotate (-90°), move_forward (the same distance required by UAV to reach the power lines from the origin), descend and land as shown in Figure 6.7.

A data logger was used to record the information that includes the position and the captured images of each flight. Four flight missions were performed in different days. A detail that must be taken into account is the number of lines detected. This can be used to avoid false positives of isolated detections. In this case the UAV flight forward continues until the power lines are detected n times. The number n is used to score the number of detections. When n is higher than a threshold (for example four detections), the alignment process is initiated. The alignment process continues until the alignment error in the image (see Figure 6.1) is less than 5°. After that the following lines process is started.

6.5 Results

In this section we present the performance of the developed techniques and processes. Firstly, as an offline process for the first test. Secondly, in a real time system for the navigation strategy.

6.5.1 First test results

The results of the the main angle obtained in a sequence of images are shown in Figure 6.8. In this Figure, the vertical axis is the detected angle and the horizontal is the number of the frame. Also, four images related at the approximate angle degrees in selected frames are shown.

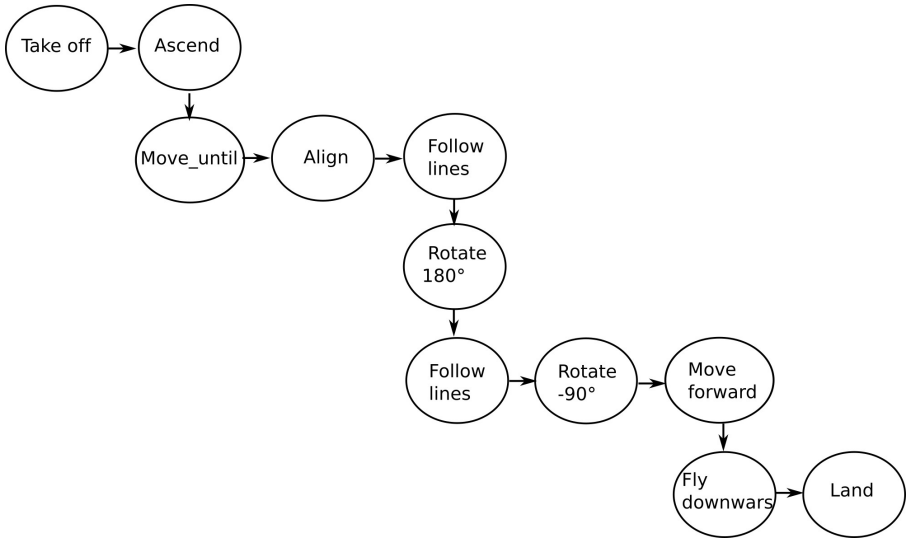


Figure 6.7: Stages of the autonomous mission process.

It is relevant to note that the presented process shown a good estimation of the power line orientation since the images in this sequence were taken from different perspectives and backgrounds. The variation in the results among consecutive frames oscillates around 5°.

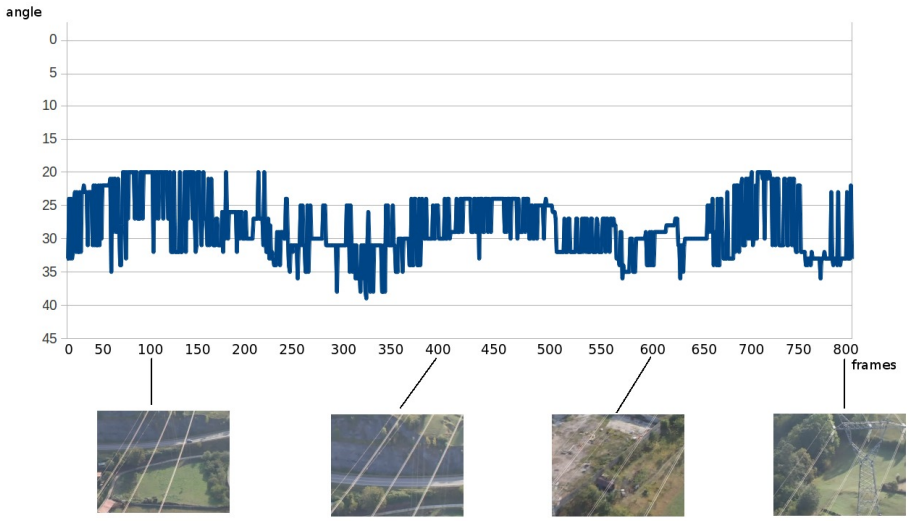


Figure 6.8: Angle obtained in a sequence of images from a perspective view.

6.5.2 Second test results

In the second test, the results of two different autonomous navigation missions are shown. In the Figure 6.9, some images obtained and recorded with the data logger during the process of a autonomous navigation mission are shown. In this figure, the blue lines represent the detected orientation.

(a) Take off. (b) Ascend. (c) Moveuntil.

(d) Detect. (e) Align. (f) Align.

(g) Align. (h) Align. (i) Align.

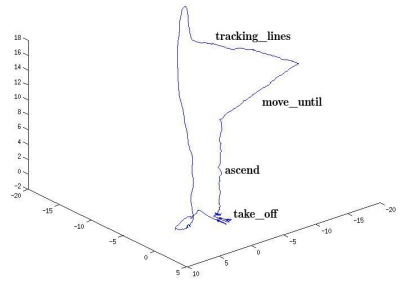
(j) Follow. (k) Follow. (l) Follow.

(m) Return. (n) Descend. (o) Land

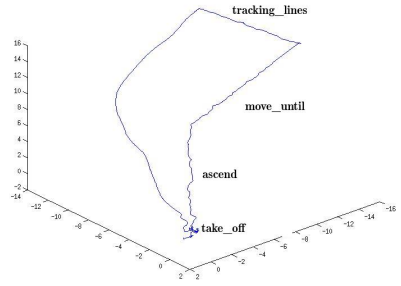
Figure 6.9: Power line following mission.

The flight information recorded with the data logger is presented in Figure 6.10. Here, four trajectories of power line following are shown. In first place, we tested two times a mission composed of the following stages: take off, ascend, move_until, align, follow_lines and return (the return stage was performed manually) as is shown in the Figures 6.10(a) and 6.10(b) from a perspective view.

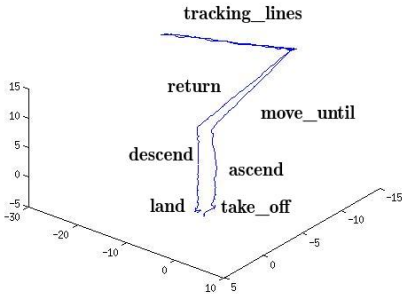
In second place, we tested two times a completely autonomous mission of power line following consisting on the stages presented in Figure 6.7. One of the main purposes of the stages prepared is that the UAV returns to its initial take off location. The results of these two autonomous missions are shown in Figures 6.10(c) and 6.10(d) from a perspective view and in Figures 6.10(e) and 6.10(f) from an upside view. As it can be seen the UAV satisfactorily performed the complete process using the onboard companion computer with no human intervention during flight. Such an achievement was done using our line detection and main angle obtaining methods.



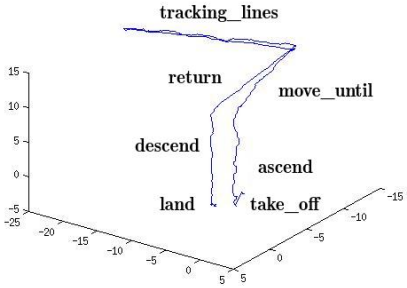
(a) Test 1.



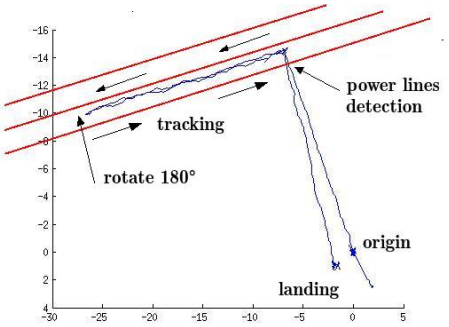
(b) Test 2.



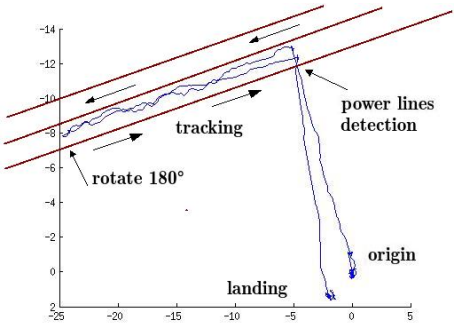
(c) Test 3.



(d) Test 4.



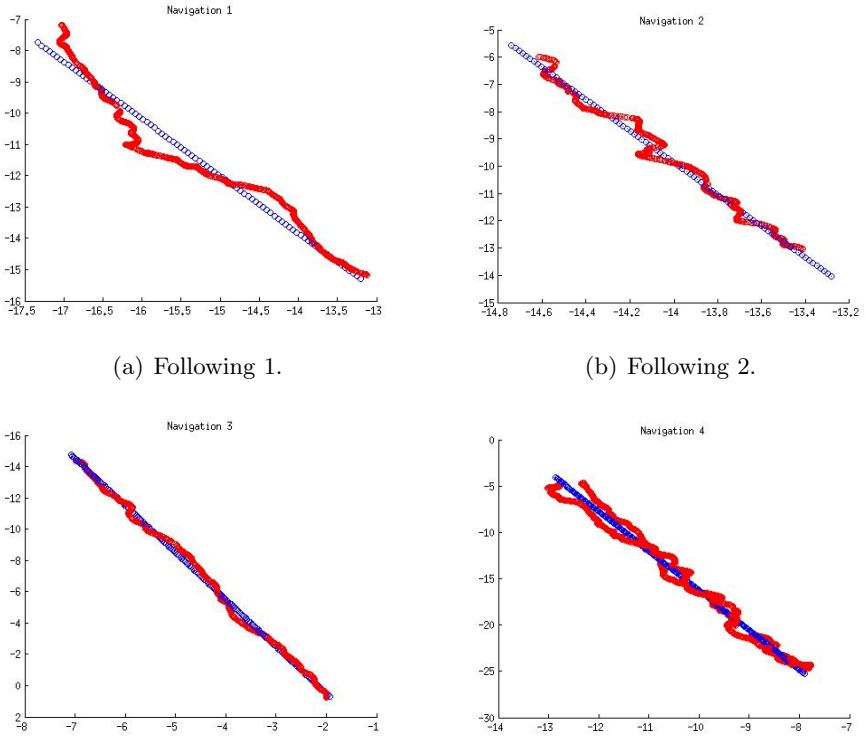
(e) Upside view test 3.



(f) Upside view test 4.

Figure 6.10: Trajectories obtained in two different missions of four flights. The positions in the figures have been measured in meters.

Finally, a linear regression using SVD (single valued decomposition) was computed with the 3D positions of the autonomous trajectory of power line following of the visual-based control. The results of regression are shown in Figure 6.11.



(a) Following 1.

(b) Following 2.

(c) Following 3.

(d) Following 4.

Figure 6.11: Linear regression of the power line following path.

As a result a 3D line is obtained in the parametric form (see Equation 6.3).

$$\vec{p} = \vec{p_0} + \vec{v}t, \tag{6.3}$$

where $\vec{p} = [x, y, z]$, $\vec{v} = [v_x, v_y, v_z]$, and $t \in [0, 1]$. The distance from each point of the obtained trajectory to the parametric line is calculated as it is shown in Equation 6.4, in this v is a normalized vector.

$$dist_i = |\vec{p_0} - \vec{p_i} - ((\vec{p_0} - \vec{p_i}) \cdot \vec{v})\vec{v}|, \tag{6.4}$$

where i is an index for the points in the trajectory.

The results of average error (see Equation 6.5) expressed in meters, for each trajectory of N points (3D positions) of the four missions, are shown in Table 6.2. The obtained error depends on different aspects such as: wind presence, camera resolution and the delay in response of the control system. This error can be reduced by using a camera with better resolution and an onboard computer with better configuration for performance.

$$E_{avg} = \sum_{1}^{N} dist_i. \tag{6.5}$$

Table 6.2: Average error of each mission in meters.

Mission	1	2	3	4
Average error	0.1937	0.0437	0.0880	0.2191

The final result can be observed in the video (see `https://youtu.be/oCEHYRlF5EQ`).

6.6 Conclusions

In the electrical infrastructure, monitoring process power lines inspection calls for the development of automated non-intensive methods of inspection that allow for the actual state of the lines without expensive or dangerous interventions over the installation surroundings.

The Histogram of Oriented Segments is a good tool for computing the direction of the linear content in images. The main peak of this histogram is a good estimation for the orientation of power lines in rural environments as many of these lines are often located in the context with trees and other kinds of vegetation. Histograms can be used not only to obtain an orientation, but also for segmenting the lines that are related with the main orientation angle and to give an estimation of the presence of power lines.

The power lines are thin elements located in aerial images which are very difficult to detect with the same accuracy in all frames of a video. This is due to different reasons, such as the abrupt movement of the camera, the UAV movements, the difference of backgrounds found in a traverse and illumination.

For this reason, a technique that estimates the orientation angle of the power line in a short time is useful. The results shown in Figure 6.8, indicate that there is a good approximation for the line orientation, which can help the UAV to remain in a trajectory related to the power lines. An additional advantage of the detection method is that the computing time for the proposed algorithms is significantly short in images taken from a real aerial inspection and UAV images (see CBS method in chapter 2). The use of this information is valuable for the UAV navigation system.

Also, the proposed navigation system enables the UAV to perform an autonomous flight without any manual intervention of an operator (or pilot).

As a future work, a technique of power line detection based on a machine learning approach could be useful to deal with complex scenes where edge detectors are not effective.

CHAPTER 7

3D reconstruction of electrical infrastructure and associated terrain

This chapter focuses on the process of tower and associated terrain reconstruction. For this process, we employed several images taken with an UAV that was in constant motion. The process is known as Structure From Motion SFM; its stages and parameters were adapted for the problem domain. The 2D feature descriptors are very important because they constitute the first stage in the toolchain for reconstruction. After that step, the bundle adjustment is performed and a dense model is obtained.

Effective and efficient generation of three-dimensional (3D) models from a set of 2D images is a well-studied problem in the literature and the principle of numerous computer vision applications. The keypoint detection and the 2D descriptor extraction is the first step in the reconstruction process followed by the matching. There are different 2D descriptors such as SIFT, ORB, BRISK and FREAK that are evaluated in this chapter in the context of 3D reconstruction.

The results show 3D reconstructions of different objects that include basic regular ones (paralelepieds and spheres) and linear ones (such as electrical towers). For the towers, a set of UAV images were obtained. We obtained 3D reconstructions of associated terrain by using different 2D descriptors in order to compare and evaluate the advantages of each one.

7.1 Related work

One of the most remarkable application of UAV imagery is 3D reconstruction from multiple 2D images. First, a cloud of points can be obtained from a set of 2D images. After that, a polygonal mesh can be obtained from the cloud of points. This process has been used to reconstruct environment from a set of disordered pictures in reasonable time [70]. Some representative works of this application are the building reconstruction as shown in [71, 72, 73] as well as urban terrain reconstruction as presented in [74] in which dynamic programming and parallel processing using graphics processing unities (GPUs) are used to reduce computing time. There exist other approaches of 3D reconstruction such as the evaluation of the road conditions as shown in [75, 76].

Small-scale aerial vehicles equipped with high resolution cameras are useful to acquire images from different points of view in short time. These images can be used for the generation of digital superficial models (DSM). Nevertheless, due to the power restriction of these devices the capture planning is essential to perform a suitable sampling during the light time [77]. In the context of simultaneous localization and mapping (SLAM) the process is known as the Next Best View (NBV) [78, 79].

7.1.1 Auto-calibration

The results of computing the structure from motion are more useful when a metric reconstruction is obtained. This means that the parallel lines remain parallel, the orthogonal walls should present rectangular angles, and finally the reconstructed model will be a scaled version of the reality. Several techniques for self calibration have been developed in order to convert a projective reconstruction into a metric one. This is equivalent to retrieving the unknown calibration matrix associated to each image [80].

7.1.2 Structure from motion

Structure From Motion SFM is a technique for estimating the 3D structure of a scene based on different views that can be related to 3D positions. The SFM process serves well when the 3D positions of objects are unknown since the 3D structure can be retrieved by means of the image correspondences.

SFM allows, to obtain the projection matrix and the 3D correspondent points in each view. This can be defined as n projected points u_{ij}, $i \in 1...m$, $j \in 1...n$ in m images. The goal is to find both projection matrix $P_1...P_m$ and the consistent structure (of 3D points in homogeneous coordinates) $X_1...X_n$. Bundle Adjustment (BA) makes an estimation using a non linear estimation that minimizes the cost function. This function corresponds to the weighted sum of the square of the retro-projection error.

7.2 3D Reconstruction process

The 3D reconstruction process consists of a set of stages that begin with the detection of keypoints. After that the descriptor extraction is performed and later the matching process is carried out. The latter is required for the bundle adjustment. Finally, a sparse model is obtained. This can be improved to generate a dense model. The different stages of the 3D reconstruction process are shown in Figure 7.1.

7.2.1 Descriptor extraction

In this work some of the 2D descriptors mentioned in chapter 2 SIFT, BRISK, ORB and FREAK are evaluated in the context of 2D reconstruction of electrical towers. In the first place the keypoints are detected and after that the descriptors are computed into keypoints.

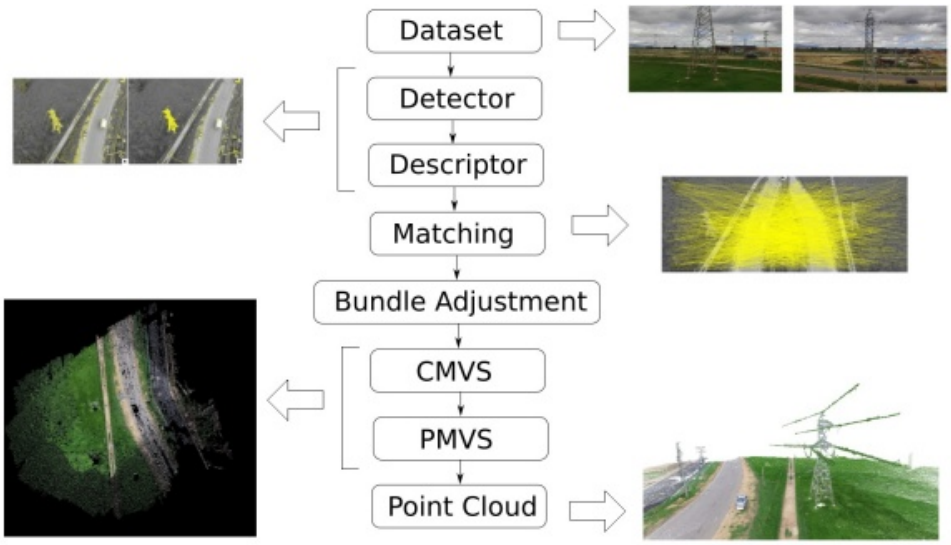


Figure 7.1: Toolchain of point cloud generation.

7.2.2 Matching

An algorithm based in a searching tree is used to determine the feature points that match in between the images by using the information of 2D feature descriptors in a radius of 0.6. In Figure 7.2, the matching results with different algorithms are shown.

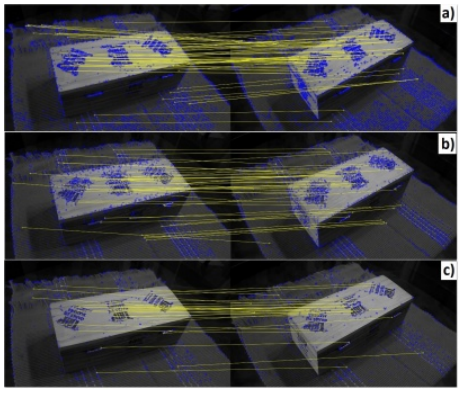


Figure 7.2: Matching with different descriptors. a) SIFT, b) BRISK, c) FREAK.

7.2.3 Bundle adjustment

The bundle adjustment of the image u_{ij} SFM has an initial estimation of the projection matrix P_i and 3D points X_j. It is necessary to improve this estimation using a non linear iterative optimization. This function is related with a weighed summation of square re-projection errors. Using Gauss-Newton it is possible to achieve a fast convergence [81].

The main goal of bundle adjustment is to determine an optimal estimator from a set of parameters θ due to a set of noisy predictions. If a set of predictions $x(\theta)$ and a set of respective observations is $z(\theta)$. The prediction residual error Δz is given by:

$$\Delta z = \bar{z} - z(\theta) \tag{7.1}$$

For the bundle adjustment stage of SFM we use the matching information in the software *Bundler* [82]. The cloud of points obtained (sparse model) consist of basic information with the principal structure which is obtained in an incremental way [83].

7.2.4 CMVS

Many algorithms for Multiview Stereo (MVS) have not a good performance with a high number of input and have a high computational cost in memory and resources. For this reason some approaches as mentioned in [84], used clustering of smaller groups of images in order to improve the performance. This process is called CMVS and is a preprocessing stage to the PMVS stage.

7.2.5 PMVS

The PMVS (Patch Multiview Stereo) algorithm takes a set of images and the camera parameters to perform a 3D reconstruction ignoring the non rigid objects. The output of the PMVS is a set of oriented points into a polygonal model with its 3D coordinates and a surface with a normal in each point [85].

7.3 Experimental setup

The experiments have the purpose of evaluating the performance of different 2D descriptors over different kind of objects.

The process is divided in 3 experiments. First, reconstruction of regular 3D objects by using the 2D descriptors SIFT, BRISK, FREAK and ORB. In this case we, evaluate the descriptors by comparing them with the geometrical model using the RMS (Root Mean Square) error. Second, reconstruction of electrical towers using the same 2D descriptors. Third, reconstruction of the associated terrain using UAV images.

For this process, a standard DJI phantom platform was used in order to obtain different views of an electrical infrastructure place. For the processing, we used two computers. First, a computer with Intel Core i5-5200 U (Broadwell) a 2.2GHz processor and RAM 8GB DDR3L memory. Secondly, a computer with Intel Core i7 4210U processor and DDR3L 1600 MHz SDRAM 8 GB SDRAM of memory.

7.3.1 First test: Regular basic objects

In first instance, the goal is to compare the clouds of points of geometric shapes with a parametric model. For this purpose the RANSAC (RANdom SAmple Consensus)

method [86][87], that is provided by the PCL (Point Cloud Library) [88] is used. RANSAC find the parameters of the 3D model from a set of points. Finally the RMS error obtained between the cloud of points and the desired 3D model is used to evaluate the performance of the different descriptors in the 3D reconstruction.

$$RMS = \frac{1}{n}\sum_{i=1}^{n}(\bar{Y}_i - Y_i)^2, \tag{7.2}$$

where, $\bar{Y}_i$ are the data of the obtained cloud of points of n elements, with SFM and Y_i are the data of the RANSAC generated model.

In the error analysis of the cloud of points with spherical geometry, the coefficients of the 3D model obtained with RANSAC are the radius r and the center coordinates (xc, yc, zc). The error for each point of the generated cloud of points is determined by computing the distance from the given point to the center. The total mean square error of the sphere is E_s.

$$\begin{gathered} \bar{r} = \sqrt{(x-xc)^2 + (y-yc)^2 + (z-zc)^2} \\ E_s = \tfrac{1}{n}\textstyle\sum_{i=1}^{n}(\bar{r}_i - r_i)^2 \end{gathered} \tag{7.3}$$

In the analysis of the cloud of points of the box it is necessary to evaluate the error of each plane. The total error in the box is the result of the total mean square error of each plane E_p.. The coefficients of each plane are obtained with RANSAC.

$$\begin{gathered} \bar{Y} = aX_{sfm} + bY_{sfm} + cZ_{sfm} + d \\ Y = aX_{rns} + bY_{rns} + cZ_{rns} + d \\ E_p = \tfrac{1}{n}\textstyle\sum_{i=1}^{n}(\bar{Y} - Y)^2 \end{gathered} \tag{7.4}$$

7.3.2 Second Test: Linear objects

In order to evaluate 3D models that contain several linear parts such as the electrical towers an interactive program for evaluating geometric surfaces by using a cylinder as a bounding volume is developed. The program separates the desired points of a cloud which corresponds to a linear part of the tower.

The 3D model of a cylinder have to be aligned to the desired linear part of the tower to be evaluated. The line is defined with two points: the initial $P_1 = (x_1, y_1, z_1)$ and the final $P_2 = (x_2, y_2, z_2)$. In this case the vertical axis is Z.

The length of the line L is obtained as the magnitude of the vector (from P_1 to P_2), $L = \sqrt{(x_2 - x_1) + (y_2 - y_1) + (z_2 - z_1)}$.

The translation parameters are (x_1, y_1, z_1). The rotation angles in the axis XZ and YZ are defined in the Equation 7.5.

$$\theta_x = atan\left(\frac{y_2 - y_1}{z_2 - z_1}\right)$$
$$\theta_y = atan\left(\frac{x_2 - x_1}{z_2 - z_1}\right). \tag{7.5}$$

Finally, the restriction that determines if a point (x, y, z) of the cloud is inside of the cylinder of radius r_c and height L, is defined as Equation 7.6.

$$\begin{aligned} 0 &\leq z \leq & L \\ -r_c \leq & \sqrt{x^2 + y^2} & \leq r_c. \end{aligned} \tag{7.6}$$

7.3.3 Dataset

The dataset is composed of two sets of images: The first consists of images of objects with basic regular shapes such as a parallepipedal (box) and a sphere (basketball). This set consists of 200 images of 1280 × 720 resolution (see Figure 7.3). The second set consists of a set of 34 images of 1280 × 720 resolution taken with the UAV in continuous flight. In Figure 7.4 the used dataset is shown.

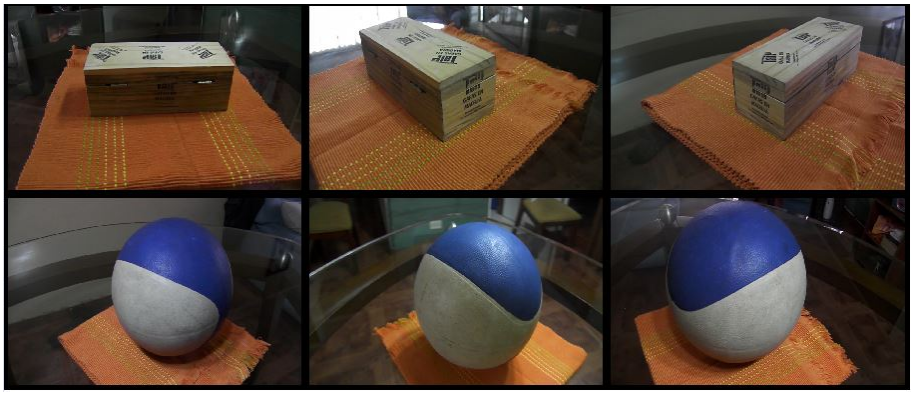


Figure 7.3: Images from dataset of regular objects.

Figure 7.4: Images obtained with an UAV.

7.4 Results

The results were divided into three tests. The first one aimed to evaluate the descriptors with regular objects. The second test to evaluate different 2D descriptors in the 3D reconstruction of electrical towers. Third test is a 3D reconstruction of the associated terrain obtained with different 2D descriptors and methods.

The time of descriptor extraction and matching is defined as T_{pm} and the time for obtaining the sparse model is defined as T_{sm}.

7.4.1 First Test Results

The number of points per view ($Npview$) is the number of keypoints detected of the set of 200 images in each view. This number provides relevant information about the performance of the methods SIFT, BRISK, ORB and FREAK used as a detectors as shown in Table 7.1.

Detector	Plane	Sphere
SIFT	7838.5	3769.7
BRISK	1357.35	662.22
FREAK	1956.11	1251.3
ORB	6457.24	2366.02

Table 7.1: Number of points per view ($Npview$) in 200 images

When the SIFT method is used as a detector, the number of relevant keypoints detected is bigger compared to the ORB. Since ORB uses the FAST method as a detector it presents keypoints detected in the corners. BRISK uses a modified version of FAST. FREAK obtain a less number of points that are more distributed in the overall image than ORB.

The number of matched points ($Npmatch$) If it is possible to match a point in three or more views this is considered, a stable point in the 3D structure. The number of stable points during the 3D reconstruction is part of the final cloud of points. Table 7.2 shows the number of matched points in 200 images. The number of coincidences obtained with SIFT is remarkable since the results shown more uniform 3D models in the sphere, this mean that is the best descriptor in the evaluated cases. For the planar objects the results are similar except with ORB.

Descriptor	Plane	Sphere
SIFT	38132	64674
BRISK	35012	9685
FREAK	35072	9753
ORB	12954	4327

Table 7.2: Number of matched points in 200 images.

Although 3D reconstruction is possible with all evaluated descriptors, the best results in terms of regularity and uniformity of the obtained clouds of points are obtained with SIFT because the other detectors and descriptors have more coincidences concentrated in the corners of the objects.

It is remarkable, that the reconstruction of spherical objects is more detailed with SIFT than using others descriptors. For planar objects FREAK present good results.

The results of the computing time for detecting and describing the keypoints in the set of images is shown in Table 7.3.

Descriptor	Plain T_{pm} (min)	Sphere T_{pm} (min)
SIFT	161.134/130.600	182.234/119.433
BRISK	45.345/28.356	33.133/16.342
FREAK	112.043/89.342	95.333/68.271
ORB	152.356/139.3	173.88/110.345

Table 7.3: First PC/Second PC: time in minutes for each descriptor

The reconstruction time of the sparse model varies depending on the number of matched points. The duration of this reconstruction depends on the obtained data on previous stages. In Table 7.4, the results of duration of this reconstruction process are reported.

Descriptor	Plain $T_{sm}(sec)$	Sphere T_{sm} (sec)
SIFT	215.0/184.0	230.0/192.00
BRISK	49.0/31.0	43.0/33.0
FREAK	203.0/175.0	199.0/169.0
ORB	170.0/152.0	151.0/133.0

Table 7.4: First PC/Second PC: time in seconds for the cloud points generations (model sparse)

From these results, we can observe that the SIFT detector/descriptor offers a bigger computational cost, followed by ORB, FREAK and BRISK. The scene geometry presents a better uniformity which improve the quality of the visualized model. BRISK reduced the computation time.

Keypoint detection is obtained with different algorithms as it is shown in 7.5.

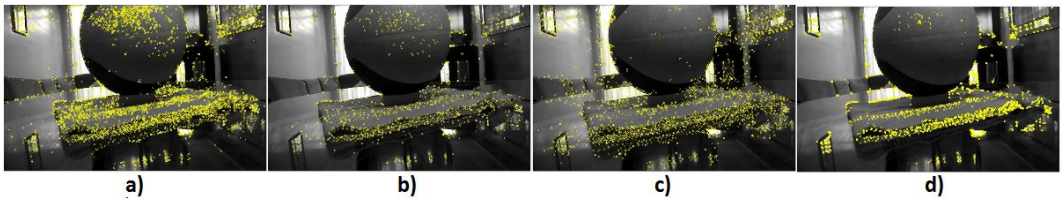


Figure 7.5: Keypoint detected with different algorithms: a) 5024 with SIFT, b) 3034 with BRISK, c) 3221 with FREAK, d) 2894 with ORB.

In figure 7.6, the results of 3D reconstruction by using different feature descriptors.

The results of the RMS error of the 3D model obtained compared with the parametric model are shown in Table 7.5. It is good to mention that the error obtained in the cloud of points using SIFT is bigger than using the other descriptors because this contains more points. Considering this SIFT presents better performance than the other evaluated descriptors.

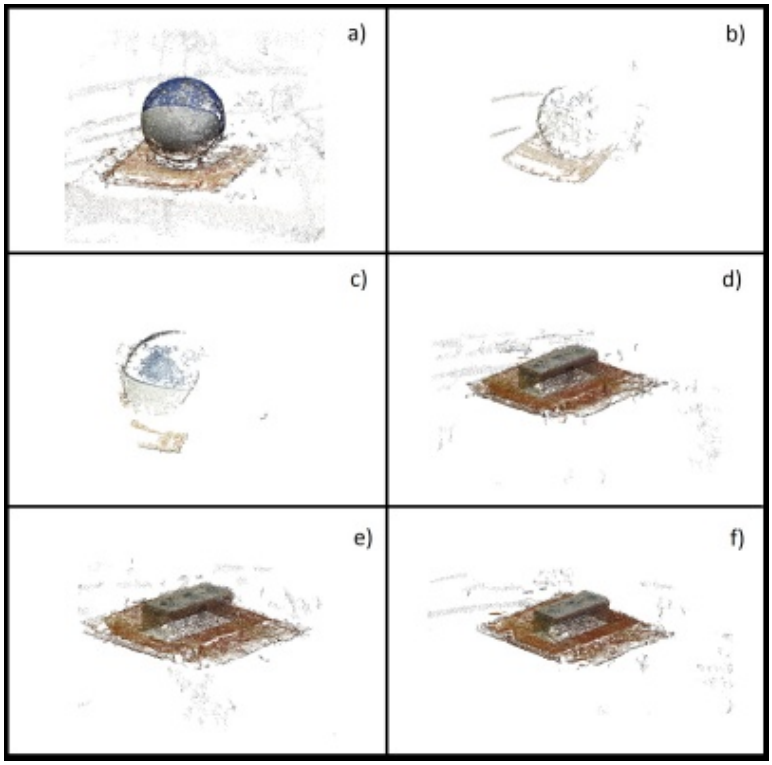


Figure 7.6: Cloud of points obtained with different feature descriptors: a) using SIFT, b) using BRISK, c) using FREAK, d) using SIFT, e) using BRISK, f) using FREAK.

Descriptor	Plane	Sphere
SIFT	1.03986 e^{-002}	7.15924 e^{-005}
BRISK	3.25866 e^{-002}	2.54412 e^{-003}
FREAK	1.06139 e^{-002}	1.58424 e^{-003}
ORB	NA	NA

Table 7.5: Root mean square error for parametric models.

The quality of the SFM obtained model compared with a parametric one shown that there exists less variations when SIFT is used. However, FREAK provides an alternative for generated models of planar objects with shorter time of computing.

7.4.2 Second Test Results

For the electrical tower case the number of points per view ($Npview$) with different descriptors are shown in Table 7.6. The results of number of matches ($Npmatch$) are shown in Table 7.7.

In this case the average of number of points detected per image for the evaluated descriptors is similar but with BRISK is higher. However the cloud of points obtained with SIFT or FREAK present an uniform distribution of points. The number of effective points in the resultant cloud is bigger also with SIFT but this takes more time than with other descriptors.

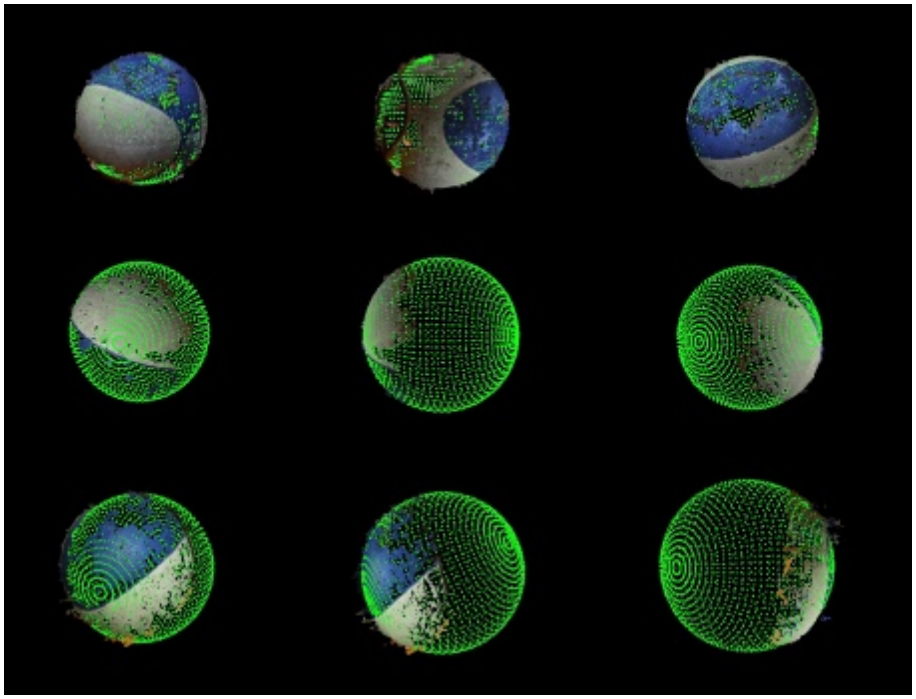


Figure 7.7: RANSAC model visualization of a sphere for the clouds of points with different descriptors. First row SIFT. Second row BRISK. Third row FREAK.

Detector	Npview	Npmatch
SIFT	5,744.5	210,337
BRISK	6,968.6	200,888
FREAK	6,636.9	174,184
ORB	5,589.7	116,529

Table 7.6: Number of points per view ($Npview$) in 33 images of towers.

There exist a significant reduction of computing time when FREAK or BRISK are employed.

The keypoint detection stage using different detection algorithms is shown in Figure 7.9. In this case 6841 with SIFT, 5598 with ORB, 5912 with FREAK and 6285 with BRISK.

The result of matching of electrical towers with different descriptors. SIFT, ORB, FREAK and BRISK is shown in Figure 7.10. The number of matches are shown in Table 7.8.

In Figure 7.11 the dense models obtained as a result of 3D reconstruction of a tower by using different feature descriptor (SIFT, BRISK and FREAK) are shown. The results in terms of computing time are shown in the Table 7.9.

Finally a quantitative evaluation of a 3D model based in the amount of stable points. The results were obtained in a radius od 1.00 around the vector defined by two points (P_1 and P_2) as is presented in Table 7.10.

In Figure 7.12 the manual selection of two points (P_1 and P_2) for the generation of the cylinder is shown. For this process a vertical support of the tower is selected in a

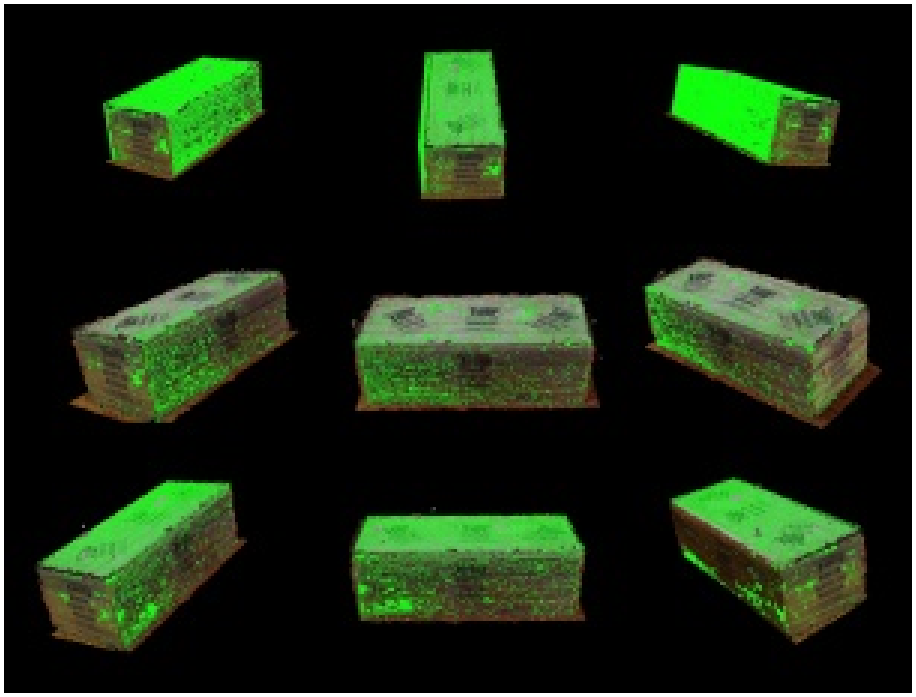


Figure 7.8: RANSAC model visualization of plane for the clouds of points with different descriptors. First row SIFT. Second row BRISK. Third row FREAK.

Descriptor	Plane	Sphere
SIFT	38132	64674
BRISK	35012	9685
FREAK	35072	9753
ORB	12954	4327

Table 7.7: Number of matched points ($Npmatch$) in 33 images.

good quality model of the tower obtained with an evaluation version of software pix4d (that uses SIFT descriptor and a manual adjustment process) in the left side of the Figure. In the right side the same vertical support is selected. The purpose of this selection is comparing the performance of the different descriptors quantitatively.

In the proposed cylinder, SIFT present more number of points inside.

7.4.3 Third Test Results

The reconstruction of associated terrain with different descriptors is shown in Figure 7.13. In this case, the SIFT descriptor provides the most detailed cloud of points.

Finally, Figure 7.14 shows different views of the 3D reconstruction obtained of associated terrains to electrical infrastructure. This reconstruction was obtained by using the software pix4d in an evaluation version which uses SIFT as a descriptor and a manual adjusting process for matching that improve the results.

In the video video (see `https://youtu.be/tEz1cv59i5Y`) the deveoped process is shown.

Figure 7.9: Keypoint detection with different algorithms a) SIFT, b) ORB, c) FREAK, d) BRISK.

Descriptor	Number of matches
SIFT	1319
FREAK	773
FREAK	737
BRISK	1506

Table 7.8: Matching results with different descriptors.

7.5 Conclusions

In this chapter, the process of SFM was used to obtain 3D reconstruction of electrical towers by using different 2D feature descriptors.

It is possible to reduce the processing time of descriptor extraction for a 3D reconstruction by using different descriptors than SIFT such as FREAK and BRISK. The reduction is close to 40%.

The 3D models obtained with this kind of descriptors are good to reconstruct regular objects since it is possible to obtain uniform cloud of points. For electrical towers the obtained cloud of points reveals important details of its rectilinear elements.

Although it is possible to obtain cloud of points by using BRISK and FREAK as a feature descriptors in shorter time than using SIFT. The developed test shown that the SIFT offer better results in the obtained meshes since the reconstructed objects do not present evident distortions and contain more details (relevant points).

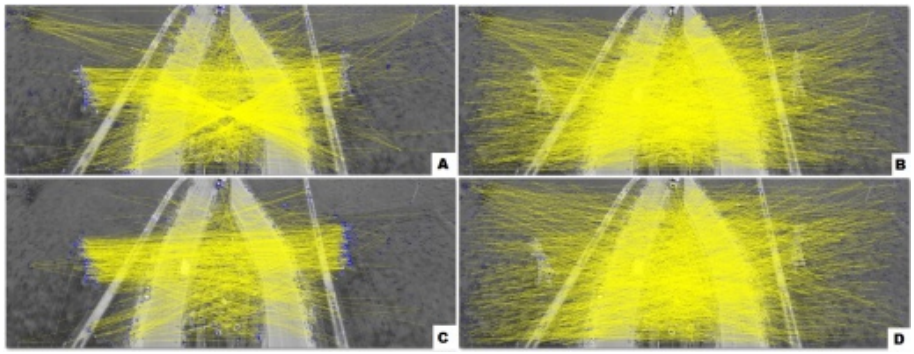


Figure 7.10: Matching of towers with different descriptors a) SIFT, b) ORB, c) FREAK and d) BRISK.

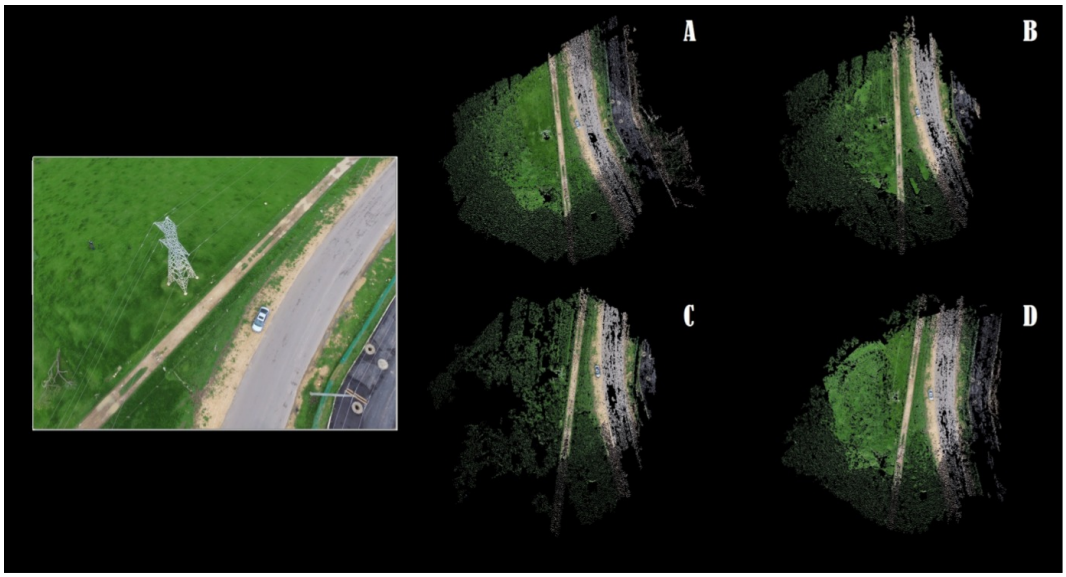


Figure 7.11: Results of tower reconstruction: a) using SIFT, b) using ORB, c) FREAK and d) using BRISK.

Descriptor	Computing descriptor time	SFM
SIFT	136.283/112.210 min	376.00/330.00 seg
BRISK	22.050/10.00 min	391.00/308.00 seg
FREAK	30.250/20.00 min	493.00/380.00 seg

Table 7.9: First PC/Second PC: time for the cloud points generations (model sparse) of electrical towers.

Descriptor	Np Model	Np SFM
SIFT	5541	549
ORB	6375	211
FREAK	6273	166
BRISK	6446	461

Table 7.10: Root mean square error for parametric models.

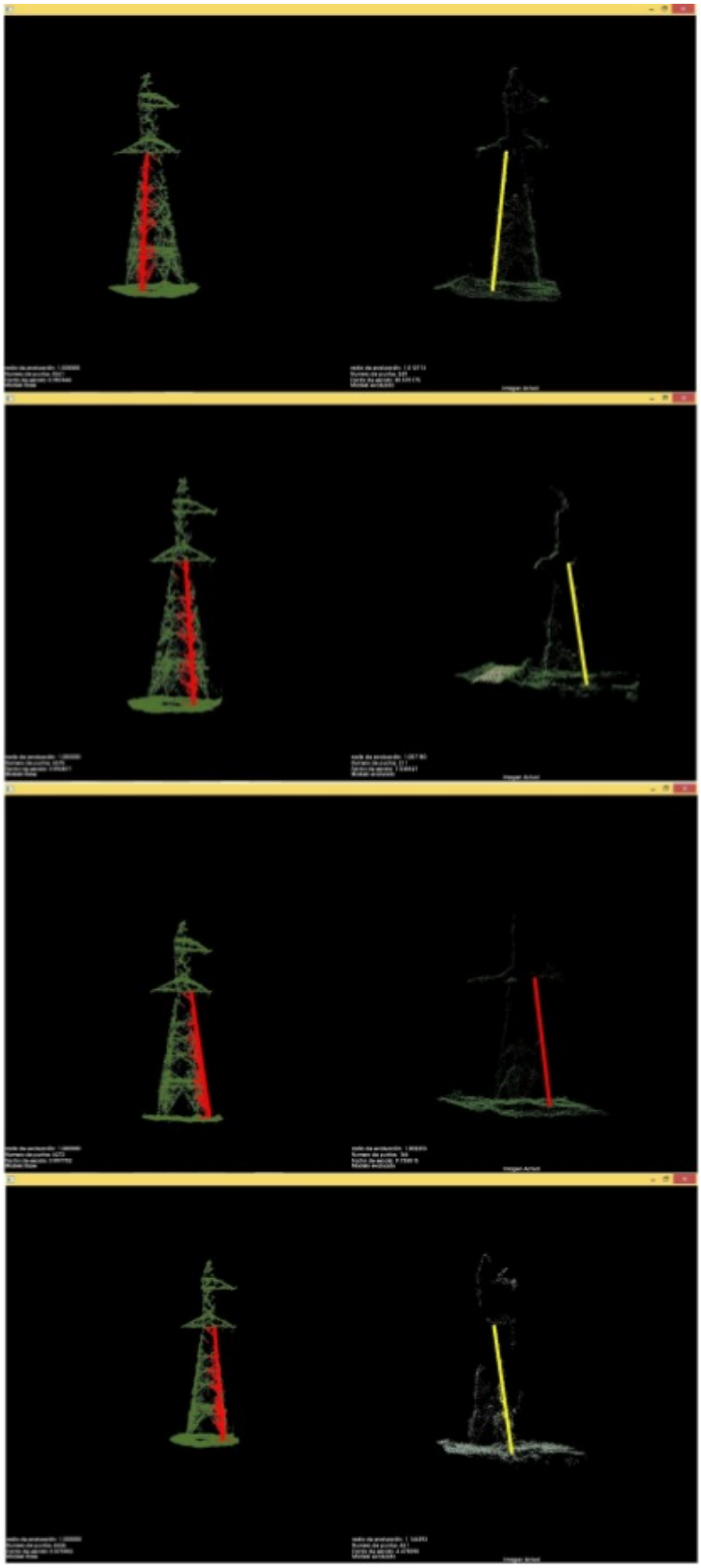


Figure 7.12: Linear elements selection. First row using SIFT, second using ORB, third using FREAK, and fourth using BRISK.

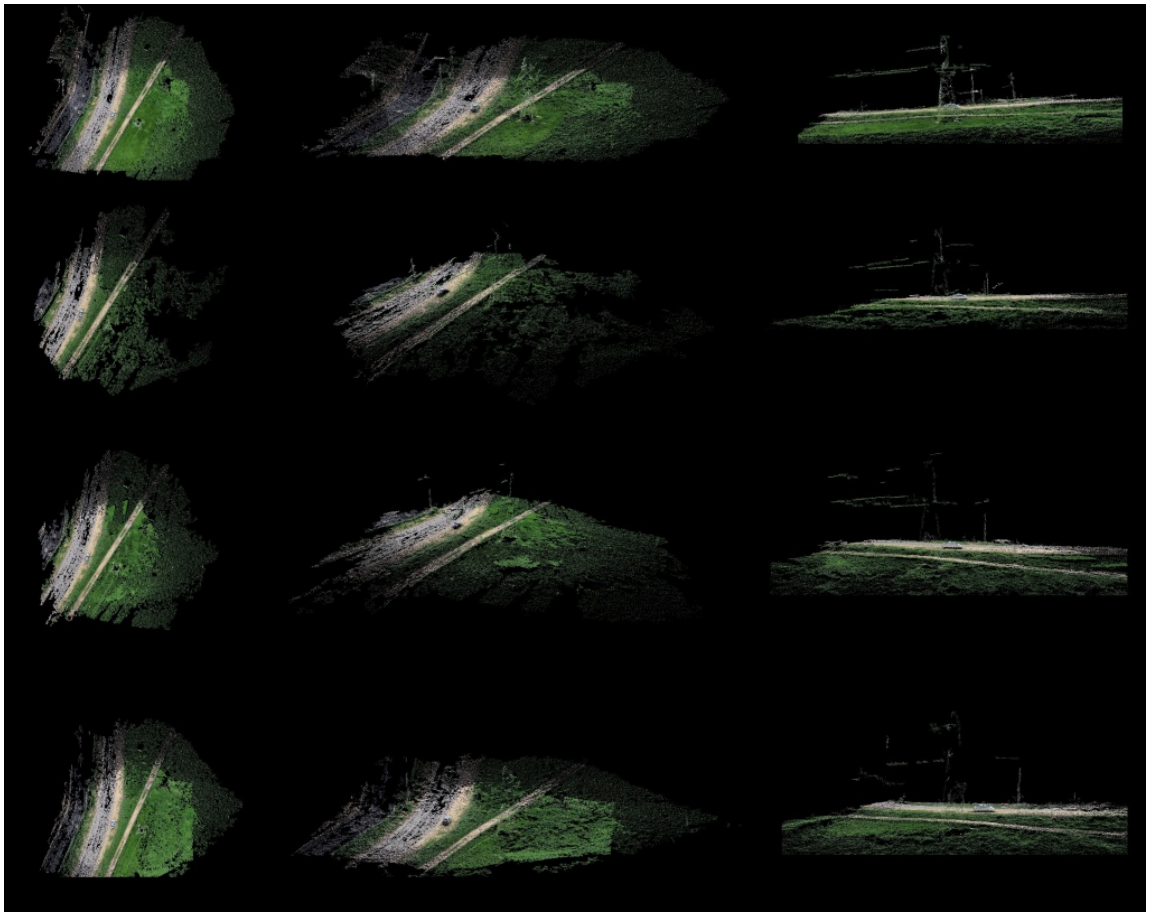
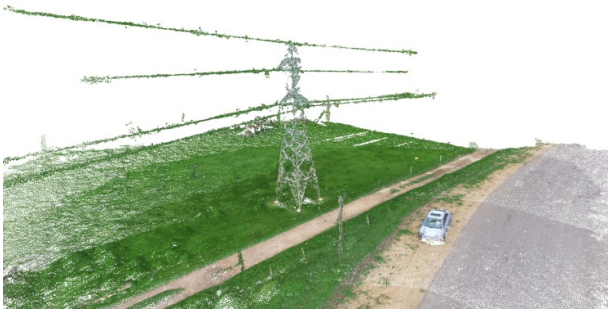


Figure 7.13: Results of terrain reconstruction. First row using SIFT, second using ORB, third using FREAK, and fourth using BRISK.



(a) view 1

(b) view 2

(c) view 3

Figure 7.14: 3D Terrain reconstruction.

CHAPTER 8

Conclusions and Future Work

This thesis has addressed the problem of onboard visual based navigation for UAVs in the context of electrical infrastructure. The endeavours were mainly focused on developing strategies based on geometrical concepts and 2D feature descriptors for detecting the visual features of scenes with power lines, catenaries and towers. The main aspect considered is the detection of power lines and towers.

The line detection method CBS detected power lines with good performance in comparison to the state of the art approaches. Its computing time proven to be efficient regarding other methods evaluated through the data set.

The proposed approaches were assessed using real data obtained in power line inspection tasks and with UAV platforms. 2D descriptors such as SIFT, SURF, ORB, FREAK can be used in a machine learning-based object detection system by using the GRID approach developed in this thesis.

Although the object detection method was developed for electric towers detection (it provides good results when detecting structures composed of many linear segments in combination with 2D descriptors), it is expected that it can be used for other similar objects, especially objects that are composed of linear segments and corners, such as those present in some machine vision systems.

The complexities of UAV autonomous navigation make it necessary the development of different flight architectures in real time simulation systems using virtual environments. This is very useful to plan flight strategies in order to avoid risk of collision or damage of the UAV with the electrical infrastructure.

Flying over an electrical infrastructure represents a particularly complex task since main elements such as power lines are thin and the background diversity makes them difficult to detect. The CBS line detection method that was developed in this thesis was used in three applications: line detection, tower detection and navigation.

The studied descriptors that were useful for object detection process were evaluated in the context of three-dimensional reconstruction of towers and associated terrain. The results show that it is possible to obtain clouds of points with different kinds of 2D feature descriptors using techniques of SFM.

8.1 Perspectives

Power line scenarios where power lines are located are very complex. Many of the lines presented in the scene are not power lines. This problem can be tackled by using computational intelligence approaches. For example, it is expected that the detection of objects or features in electrical infrastructures could be extended in a high level system based on semantic information is implemented.

Research on scene understanding is of great importance due to the many applications that include information retrieval, robot navigation and security among others. The first stage consists on the scene analysis in order to determine the type of scene. After that, it is required to identify the elements that are present in the scene by using object detection methods such as the mentioned in chapter 4 of this dissertation. An approach used for this is called Geometric Scene Categorization (GSC) [89, 90]. A method for power line navigation based on the scene understanding for UAV navigation could be an interesting topic for future work.

Since line detection methods have been used for power line detection and tower detection, it is interesting to evaluate them in the context of tower reconstruction in order to obtain relevant areas for the matching process of the 3D reconstruction. A study of relevance of different line detection methods for the correspondence problem in 3D reconstruction could be an interesting topic.

A new method for tower detection was developed in this thesis. The implementation on an onboard vision system for autonomous navigation based in tower detection for power line environments is considered as a future work.

A combination of power line detection from a top down view, catenary and tower detection can be implemented in a robust UAV navigation system.

Another part that could be considered in a computer vision system for power line and tower detection is the skyline detection, as the background of the sky is different than the soil and vegetation requires different considerations. This can be useful for improving the detection of the electrical infrastructure elements.

Several descriptors that use parallel process and present less time of response have been developed. This has boosted the use of General Purpose Graphics Processing Unit GPGPU for onboard computers with graphic processing units in order to accelerate the computer vision techniques. Architectures such as CUDA (Compute Unified Device Architecture) are useful for parallel programming different computer vision methods. For this reason it is expected the development of different methods for feature extraction, line detection and object detection using GPU.

3D models of electrical infrastructure such as electrical towers have been obtained by using SFM techniques. Future work could be focused on the development of 3D methods for cloud of points processing that allow to detect and recognize different elements that are part of the electrical infrastructure such as the power lines, the electrical tower and the elements that are not part of the electrical infrastructure. This by using techniques that include 3D shape descriptors as mentioned in a previous work [28].

Also, an interesting work is to validate different methods for power line following on other kinds of infrastructures and with other kinds of flight platforms such as octocopters and fixed wing planes. This can be implemented using sensor fusion.

This work can be extended to other applications such as the surveillance of outdoors areas from different points of view. Also other interesting applications can be focused in people detection and identification, and oil pipeline inspection with UAVs by using computer vision methods.

Bibliography

[1] Kenzo Nonami, Farid Kendoul, Satoshi Suzuki, Wei Wang, and Daisuke Nakazawa. *Autonomous Flying Robots*. Springer, Tokio, 2010.

[2] Zhengrong Li, Rodney Walker, Ross Hayward, and Luis Mejias. Advances in Vegetation Management for Power Line Corridor Monitoring Using Aerial Remote Sensing Techniques. In *Proceedings of the First International Conference on Applied Robotics for the Power Industry (CARPI)*, pages 1–6. Ieee, October 2010.

[3] Dewi Jones. Power line inspection - a UAV concept. In *Autonomous Systems*, 2005.

[4] Serge Montambault, Julien Beaudry, Kristopher Toussaint, and Nicolas Pouliot. On the application of VTOL UAVs to the inspection of power utility assets. In *2010 1st International Conference on Applied Robotics for the Power Industry (CARPI 2010)*, pages 1–7. Ieee, October 2010.

[5] Binhai Wang, Xiguang Chen, Qian Wang, Liang Liu, Hailong Zhang, and Bingqiang Li. Power line inspection with a flying robot. In *2010 1st International Conference on Applied Robotics for the Power Industry (CARPI 2010)*, pages 1–6. IEEE, October 2010.

[6] Philipp Heer. Framework for Vision-Based Power Line Inspection with an UAV. Technical report, 2012.

[7] Ming Lu, Gehao Sheng, Yadong Liu, Xiuchen Jiang, Shiguang Nie, and Guangyu Qu. Inspection Based on Unmanned Aerial Vehicle. In *Power and Energy Engineering Conference (APPEEC)*, 2012.

[8] Dewi Jones. Aerial inspection of overhead power lines using video: estimation. *IEE Proceedings Vision, Image & Signal Processing*, 147(2):157–166, 2000.

[9] Stephan M Weiss. *Vision Based Navigation for Micro Helicopters (PhD Thesis - Weiss 2012)*. PhD thesis, ETH, 2012.

[10] Davide Scaramuzza, Friedrich Fraundorfer, and By Davide Scaramuzza. Visual Odometry Part I: The First 30 Years and Fundamentals. *IEEE Robotics & Automation Magazine*, (December):80–92, 2011.

[11] Friedrich Fraundorfer, Davide Scaramuzza, and By Friedrich Fraundorfer. Visual Odometry Part II: Matching,Robustness, Optimization, andApplications. *IEEE Robotics & Automation Magazine*, (June):78–90, 2012.

[12] Jorge Artieda, Pascual Campoy, Juan F Correa, Carol Mart, and Miguel Olivares. Visual 3-D SLAM from UAVs. *Journal of Intelligent & Robotic Systems*, 55, 2009.

[13] Iván F. Mondragón. *On-board visual control algorithms for Unmanned Aerial Vehicles*. PhD thesis, Universidad Politecnica de Madrid, 2011.

[14] Dongjin Lee, Yeongju Kim, and Hyochoong Bang. Vision-Based Terrain Referenced Navigation for Unmanned Aerial Vehicles Using Homography Relationship. In *ICUAS*, 2012.

[15] Paul Hough. Method and means for recognizing complex patterns. US Patent: 3,069,654., 1962.

[16] Yuee Liu, Luis Mejias, and Zhengrong Li. Fast power line detection and localization using steerable filter for active uav guidance. In *In 12th International Society for Photogrammetry & Remote Sensing (ISPRS2012)*, volume XXXIX, pages 491–496, 2012.

[17] Yuee Liu and Luis Mejias. Real-time Power Line Extraction from Unmanned Aerial System Video Images. In *2nd International Conference on Applied Robotics for the Power Industry (CARPI)*, number September, pages 52 – 57, 2012.

[18] Xiwen Yao, Lei Guo, and Tianyun Zhao. Power Line Detection Based on Region Growing and Ridge-Based Line Detector. In Zengqi Sun and Zhidong Deng, editors, *Chinese Intelligent Automation Conference*, volume 255 of *Lecture Notes in Electrical Engineering*, pages 431–437, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg.

[19] Alexander Ceron, Flavio Prieto, and Ivan Mondragon. Power line detection using a circle based search with UAV images. In *2014 International Conference on Unmanned Aircraft Systems (ICUAS)*, 2014.

[20] Biqin Song and Xuelong Li. Power line detection from optical images. *Neurocomputing*, 129:350–361, April 2014.

[21] Zhengrong Li, Troy S Bruggemann, Jason J Ford, Luis Mejias, and Yuee Liu. Toward Automated Power Line Corridor Monitoring Using Advanced Aircraft Control and Multisource Feature Fusion. *Journal of Field Robotics*, 29(1):4–24, 2012.

[22] Jared Heinly, Enrique Dunn, and Jan-michael Frahm. Comparative Evaluation of Binary Features. In *Computer Vision ECCV*, pages 759–773, 2012.

[23] Marius Muja and David G. Lowe. Fast Matching of Binary Features. *2012 Ninth Conference on Computer and Robot Vision*, pages 404–410, May 2012.

[24] Baojie Fan, Yingkui Du, and Yandong Tang. Efficient Registration Algorithm for UAV Image Sequence. In *Information and Automation (ICIA)*, number June, pages 111–116, 2011.

[25] Wengang Cheng and Zhengzheng Song. Power Pole Detection Based on Graph Cut. *2008 Congress on Image and Signal Processing*, pages 720–724, 2008.

[26] Carol Martinez, Carlos Sampedro, Aneesh Chauhan, and Pascual Campoy. Towards Autonomous Detection and Tracking of Electric Towers for Aerial Power Line Inspection. In *International Conference on Unmanned Aircraft Systems (ICUAS)*, 2014.

[27] Steven J. Mills, Jason Ford, and Luis Mejias. Vision Based Control for Fixed Wing UAVs Inspecting Locally Linear Infrastructure using Skid-to-Turn Maneuvers. In *International Symposium on Unmanned Aerial Vehicles*, 2011.

[28] Alexander Ceron and Flavio Prieto. Evaluating and comparing of 3d shape descriptors for object recognition. In Springer, editor, *Proceedings of ISVC 2013*, volume 8034 of *LNCS*, pages 484–492, 2013.

[29] Alexander Ceron, Ivan F. Modragon, and Flavio Prieto. Towards visual based navigation with power line detection. In Springer, editor, *Proceedings of ISVC 2014*, volume 8887 of *LNCS*, pages 827–836, 2014.

[30] Alexander Ceron, Ivan F. Modragon, and Flavio Prieto. Visual-based navigation for power line inspection by using virtual environments. In *Proc. SPIE 9406, Intelligent Robots and Computer Vision XXXII: Algorithms and Techniques*, 2015.

[31] David G Lowe. Distinctive Image Features from Scale-Invariant Keypoints. *IJCV*, pages 1–28, 2004.

[32] Herbert Bay, Tinne Tuytelaars, and Luc Van Gool. SURF: Speeded Up Robust Features. In *ECCV*, 2006.

[33] Michael Calonder, Vincent Lepetit, and Christoph Strecha. Brief: Binary robust independent elementary features. In *11th European Conference on Computer Vision*, 2010.

[34] Ethan Rublee, Vincent Rabaud, Kurt Konolige, and Gary Bradski. ORB: an efficient alternative to SIFT or SURF. In *ICCV*, 2011.

[35] Stefan Leutenegger and Margarita Chli. Brisk: Binary robust invariant scalable keypoints. In *IEEE International Conference on Computer Vision (ICCV)*, 2011.

[36] A. Alahi, R. Ortiz, and P. Vandergheynst. FREAK: Fast Retina Keypoint. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 510–517. Ieee, June 2012.

[37] Edward Rosten and Tom Drummond. Machine learning for high-speed corner detection. In *ECCV'06 Proceedings of the 9th European conference on Computer Vision*, pages 430–443, 2006.

[38] Paul L. Rosin. Measuring corner properties. *Computer Vision and Image Understanding*, 73(2):291–307, 1999.

[39] Elmar Mair, Gregory D. Hager, Darius Burschka, Michael Suppa, and Gerhard Hirzinger. Adaptive and generic corner detection based on the accelerated segment test. In *Proceedings of the European Conference on Computer Vision (ECCV'10)*, September 2010.

[40] Rafael Grompone von Gioi, Jérémie Jakubowicz, Jean-Michel Morel, and Gregory Randall. LSD: a fast line segment detector with a false detection control. *IEEE transactions on pattern analysis and machine intelligence*, 32(4):722–32, April 2010.

[41] Brian Burns, Allen R. Hanson, and Eduard M. Riseman. Extracting Straight Lines. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-8(4):425–455, July 1986.

[42] Cuneyt Akinlar and Cihan Topal. EDLines: A real-time line segment detector with a false detection control. *Pattern Recognition Letters*, 32(13):1633–1642, October 2011.

[43] Rafael Grompone von Gioi, Jérémie Jakubowicz, Jean-Michel Morel, and Gregory Randall. LSD: a Line Segment Detector. *Image Processing On Line*, March 2012.

[44] Richard O Duda, Peter E Hart, and Menlo Park. Use of the Hough Transformation To Detect Lines and Curves in Pictures. *Graphics and Image Processing*, 15(1):11–15, 1972.

[45] John Canny. A Computational Approach to Edge Detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-8(6), 1986.

[46] Zhengrong Li, Yuee Liu, Ross Hayward, Jinglan Zhang, and Jinhai Cai. Knowledge-based power line detection for UAV surveillance and inspection systems. *2008 23rd International Conference Image and Vision Computing New Zealand*, pages 1–6, November 2008.

[47] Jingjing Zhang, Liang Liu, Binhai Wang, Xiguang Chen, and Zheng. High speed Automatic Power Line Detection and Tracking for a UAV-Based Inspection. In *International Conference on Industrial Control and Electronics Engineering*, 2012.

[48] Cihan Topal and Cuneyt Akinlar. Edge Drawing: A combined real-time edge and segment detector. *Journal of Visual Communication and Image Representation*, 23(6):862–872, August 2012.

[49] Donald Hearn and M. Pauline Baker. *Computer Graphics*. Prentice Hall, 1996.

[50] Young-Ho Choi, Tae-Kyeong Lee, and Se-Young Oh. A line feature based SLAM with low grade range sensors using geometric constraints and active exploration for mobile robot. *Autonomous Robots*, 24(1):13–27, October 2007.

[51] Su-yong An, Jeong-gwan Kang, Lae-kyoung Lee, and Se-young Oh. SLAM with Salient Line Feature Extraction in Indoor Environments. In *11th Int. Conf. Control, Automation, Robotics and Vision*, number December, pages 410–416, 2010.

[52] W.T. Freeman and E.H. Adelson. The design and use of steerable filters. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(9):891–906, 1991.

[53] C. Steger. An unbiased detector of curvilinear structures. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(2):113–125, 1998.

[54] I. Rock and S. Palmer. The legacy of gestarl psyology. *Scientific American*, 263:84–90, 1990.

[55] Daniel Crevier. A Probabilistic Method for Extracting Chains of Collinear Segments. *Computer Vision and Image Understanding*, 76(1):36–53, October 1999.

[56] Paul Viola and Michael Jones. Rapid Object Detection using a Boosted Cascade of Simple Features. In *IEEE CVPR*, 2001.

[57] Navneet Dalal and Bill Triggs. Histograms of Oriented Gradients for Human Detection. In *Computer Vision and Pattern Recognition, CVPR*, 2005.

[58] Pedro F Felzenszwalb, Ross B Girshick, David Mcallester, and Deva Ramanan. Object Detection with Discriminatively Trained Part Based Models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(9):1627–1645, 2010.

[59] Tomasz Malisiewicz, Abhinav Gupta, and Alexei A Efros. Ensemble of Exemplar-SVMs for Object Detection and Beyond. In *IEEE International Conference on Computer Vision (ICCV)*, pages 89–96, 2011.

[60] Ian Golightly and Dewi Jones. Corner detection and matching for visual tracking during power line inspection. *Image and Vision Computing*, 21(9):827–840, September 2003.

[61] B. Cetin and M. Bikdash. Automated Electric Utility Pole Detection from Aerial Images. In *Southeastcon, 2009. SOUTHEASTCON '09. IEEE*, pages 44–49, 2009.

[62] Jittichat Tilawat, Nipon Theera-umpon, and Sansanee Auephanwiriyakul. Automatic Detection of Electricity Pylons in Aerial Video Sequences. 1(Iceie):342–346, 2010.

[63] Paolo Piccinini, Andrea Prati, and Rita Cucchiara. Real-time object detection and localization with SIFT-based clustering. *Image and Vision Computing*, 30(8):573–587, August 2012.

[64] Jianguo Li and Yimin Zhang. Learning SURF Cascade for Fast and Accurate Object Detection. *2013 IEEE Conference on Computer Vision and Pattern Recognition*, pages 3468–3475, June 2013.

[65] Alexander Ceron, Augusto Salazar, and Flavio Prieto. Relevance analysis of 3D shape descriptors on interest points and regions of the face. *Int. J. Signal and Imaging Systems Engineering*, 5(1):0–10, 2012.

[66] Johannes Meyer, Alexander Sendobry, Stefan Kohlbrecher, Uwe Klingauf, and Oskar Von Stryk. Comprehensive Simulation of Quadrotor UAVs Using ROS and Gazebo. In *Simulation, Modeling, and Programming for Autonomous Robots*, pages 400–411, 2012.

[67] Stefano Carpin, Mike Lewis, Jijun Wang, Stephen Balakirsky, and Chris Scrapper. USARSim: a robot simulator for research and education. In *IEEE International Conference on Robotics and Automation*, pages 1400–1405. Ieee, April 2007.

[68] Nick Dijkshoorn. Integrating Sensor and Motion Models to Localize an Autonomous AR.Drone. *International Journal of Micro Air Vehicles*, 3(4), 2011.

[69] Nick Dijkshoorn. Simultaneous localization and mapping with the AR.Drone. 2012.

[70] Sameer Agarwal, Noah Snavely, Ian Simon, Steven M Seitz, and Richard Szeliski. Building Rome in a Day. In *International Conference on Computer Vision*, 2009.

[71] Jan Bartelsen Helmut Mayer. Automated 3d reconstruction of urban areas from networks of wide-baseline image sequences. In *ISPRS08*, pages 633,638, 2008.

[72] Steffen and F Wolfgang. On visual real time mapping for unmanned aerial vehicles. In *21st Congress of the International Society for Photogrammetry and Remote Sensing, Beijing, China*, pages 57–62, 2008.

[73] Cornelius Wefelscheid, Ronny Hänsch, and Olaf Hellwich. Three-Dimensional Building Reconstruction Using Images Obtained by Unmanned Aerial Vehicles. In *Conference on Unmanned Aerial Vehicle in Geomatics*, 2011.

[74] Huei-Hung Liao, Yuping Lin, and Gerard Medioni. Aerial 3D Reconstruction with Line-Constrained Dynamic Programming. In *ICCV*, 2011.

[75] Chunsun Zhang and Ahmed Elaksher. 3D Reconstruction from UAV-acquired Imagery for Road Surface Distress Assessment. In *31st Asian Conference of Remote Sensing*, 2010.

[76] Chunsun Zhang and Ahmed Elaksher. An Unmanned Aerial Vehicle-Based Imaging System for 3D Measurement of Unpaved Road Surface Distresses. *Computer-Aided Civil and Infrastructure Engineering*, 27:118–129, 2012.

[77] Christof Hoppe, Andreas Wendel, Stefanie Zollmann, Katrin Pirker, Arnold Irschara, Horst Bischof, Stefan Kluckner, and Siemens Corporate Technology. Photogrammetric Camera Network Design for Micro Aerial Vehicles. *Computer Vision Winter Workshop*, 2012.

[78] Enrique Dunn, Jur Van Den Berg, and Jan-Michael Frahm. Developing visual sensing strategies through next best view planning. *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4001–4008, October 2009.

[79] Enrique Dunn and Jan-Michael Frahm. Next best view planning for active model improvement. *Procedings of the British Machine Vision Conference 2009*, pages 53.1–53.11, 2009.

[80] Richard Hartley and Andrew Zisserman. *Multiple View Geomtry in computer vision*. Cambridge, 2003.

[81] Bill Triggs, Philip F. McLauchlan, Richard I. Hartley, and Andrew W. Fitzgibbon. Bundle adjustment - a modern synthesis. *Lecture Notes in Computer Science*, 1883:298–372, 2002.

[82] Noah Snavely, Steven M. Seitz, and Richard Szeliski. Photo tourism: Exploring photo collections in 3d. *ACM Transactions on Graphics (TOG) - Proceedings of ACM SIGGRAPH*, 2006.

[83] Noah Snavely, Steven M. Seitz, and Richard Szeliski. Modeling the world from internet photo collections. *International Journal on Computer Vision*, 2007.

[84] Yasutaka Furukawa, Brian Curless, Steven M Seitz, and Richard Szeliski. Towards Internet-scale Multi-view Stereo. In *CVPR*, 2010.

[85] Yasutaka Furukawa and Jean Ponce. Accurate, dense, and robust multiview stereopsis. *IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE*, 2010.

[86] H Cantzler. Random Sample Consensus (RANSAC) Subsampling of the input data. *Journal of Computational Biology*, 124(8):2–5, 1981.

[87] H Cantzler. Random sample consensus(Ransac). *Institute for Perception, Action and Behaviour, Division of Informatics*, pages 2–5, 2005.

[88] Radu Bogdan Rusu and Steve Cousins. 3d is here: Point cloud library (pcl)@ONLINE, May 2011.

[89] Vladimir Nedović, Arnold W M Smeulders, André Redert, and Jan-Mark Geusebroek. Stages as models of scene geometry. *IEEE transactions on pattern analysis and machine intelligence*, 32(9):1673–87, September 2010.

[90] Chanho Jung and Changick Kim. Real-Time Estimation of 3D Scene Geometry from a Single Image. *Pattern Recognition*, 2012.