Technische Universität Berlin

Master Thesis

Path-Planning for Optimal Coverage under Security Constraints

Author:
Pere Guillem Mas Gutiérrez
(Universitat Politècnica de Catalunya)

Supervisor:
Dr.-Ing. Ronny Hänsch
(Technische Universität Berlin)

Fakultät IV Elektrotechnik und Informatik
Computer Vision & Remote Sensing

June 30, 2014

Declaration of Authorship

I, Pere Guillem Mas Gutiérrez, declare that this thesis titled, 'Path-Planning for Optimal Coverage under Security Constraints' and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed: _____

Date: _____

Abstract

In recent years, three-dimensional building reconstruction has been an active area of research, partly motivated by the spread of low cost *unmanned aerial vehicles* platforms. These permit exploiting the entire three-dimensional space as long as it is free of obstacles. Current approaches manually plan a set of viewpoints from which to conduct multiple scans of a target building, and then later select the best ones to use in a *structure from motion* system. This procedure often has two problems: some parts are covered with low detail or some parts are evenly uncovered. In these situations, an automatic view planner is necessary; it will completely cover a building surface, while reducing time and cost of the overall process. This thesis presents an automatic view planner for three-dimensional building reconstruction based on dividing edifices into several slices and for each one solve a two-dimensional problem. From a rough model of the environment and a desired detail level, both described in a cost function, the system computes a route in which there is a set of viewpoints to completely cover a target building surface of any shape, taking into account that there may be obstacles in the environment. The final route is proposed to be followed by an *unmanned aerial vehicle* equipped with a digital camera.

Keywords: Computer vision, 3D building reconstruction, view planning, path planning, structure from motion, UAVs.

Zusammenfassung

Die digitale, drei-dimensionale Gebäuderekonstruktion ist ein aktiver Bereich der Forschung, der besonders in den letzten Jahren an Bedeutung zunahm - teilweise motiviert durch preiswerte, unbemannte Fluggerät, welche es ermöglichen eine drei-dimensionale Szene vollständig zu erkunden, solange diese frei von Hindernissen ist. Aktuelle Methoden wählen die Aufnahmepunkte der Kameras manuell aus, und erzeugen aus diesen eine Menge von Bildern vom gewünschten Gebäude. Die besten Aufnahmen können anschließend für ein *Structure-from-Motion* System verwendet werden. Dieses Herangehensweise hat zwei häufig auftretende Probleme: Einige Stellen werden nur detailarm oder gar nicht aufgezeichnet. Für derartige Situationen ist eine automatisierte Aufnahmeplanung nötig; Diese ermöglicht eine komplette Erfassung der Gebäudeoberfläche und reduziert dabei gleichzeitig den Gesamtzeitaufwand und die Kosten des Prozesses. Diese Arbeit stellt einen automatisierten Aufnahmepunktplaner für die drei-dimensionale Gebäuderekonstruktion vor, welcher auf der Aufteilung einer Struktur in mehrer, horizontale Scheiben und dem Lösen des so zwei-dimensionalen Problems für jedes dieser Scheiben basiert. Aus einem groben Modell der Umgebung und einem gewünschten Detailgrad der Aufnahmen - beides in einer Kostenfunktion integriert - berechnet das System eine Route, die es erlaubt die komplette Oberfläche eines Gebäudes, unabhängig von dessen Form, aufzunehmen. Auch die mögliche Anwesenheit von Hindernissen wird durch das System berücksichtigt. Die erhaltene Route dient dann als Flugweg für ein, mit einer Digitalkamera ausgestattetes, unbemanntes Fluggerät.

Stichwörter: Computer vision, 3D-Gebäuderekonstruktion, view planning, path planning, structure from motion, UAVs.

Acknowledgements

First of all I would like to thank my supervisor Dr.-Ing. Ronny Hänsch for his guidance. His valuable advice have been of great help for writing this thesis and I feel really fortunate to have worked with him. Moreover, I appreciate his patience and understanding during our meetings.

Next, I would like to express my gratitude to my home university supervisor, Professor Philippe Salembier, who, despite the distance problem, gave me useful tips. His Image Processing course remains in my mind as the one I most enjoyed during the degree.

Further, my sincere thanks to Toni Martínez del Hoyo and Francesc Capó. They have become more than simple university colleagues after spending a lot of time together, either studying or writing the thesis. With Toni I have shared the experience of being an Erasmus in the best city we could have gone, and besides thanking his countless tips for my thesis, also for teaching me to be ambitious as engineer. Francesc has been a close friend in the last years and our discussions, advices and talks have been helpful and motivating for keep working in hard moments. I also would like to express my gratitude to Nicolás Ribas, Bartomeu Massutí and Bartomeu Monserrat, whose views and suggestions always were useful for my thesis.

Special thanks to Caterina Mas, who despite belonging to another scientific field, always gave me her sincere opinion about my work. She taught me what discipline means and to work hard for my dreams.

Last but not least, I would like to thank my family for supporting me in my studies, first in Barcelona, and this year in Berlin. My parents Isabel Gutiérrez and Guillermo Mas, with their constant support have taught me that an education, not just academic, is the most valuable treasure a person can have and this is something I will never forget. Finally, I want to thank my siblings (and godparents) María José Mas and Antonio Mas for their confidence in me. They were a role model for me since I was a child, and they have a special place in my heart.

≫ *Per acabar, però no menys important m'agradaria donar les gràcies a la meva família per recolzar-me amb els meus estudis, primer a Barcelona, i aquest any a Berlín. Els meus pares Isabel Gutiérrez i Guillermo Mas, amb el seu suport constant m'han ensenyat que una educació, no només acadèmica, és el tresor més valuós que una persona pot tenir i això no ho oblidaré mai. Finalment, vull agraïr als meus germans (i padrins joves) Maria José Mas i Antonio Mas, la seva confiança en mi. Ells van ser un model a seguir per a mi des que era un nen i tenen un lloc especial en el meu cor.* ≪

Contents

List of Figures

List of Tables

Abbreviations

2D	Two-**D**imensional
3D	Three-**D**imensional
DSLR	**D**igital **S**ingle **L**ens **R**eflex
DOF	**D**epth **O**f **F**ield
FOV	**F**ield **O**f **V**iew
GNSS	**G**lobal **N**avigation **S**atellite **S**ystem
GPS	**G**lobal **P**ositioning **S**ystem
GUI	**G**raphical **U**ser **I**nterface
IMU	**I**nertial **M**easurement **U**nit
LIDAR	**LI**ght **D**etection **A**nd **R**anging
LOD	**L**evel **O**f **D**etail
NBV	**N**ext **B**est **V**iew
NBP	**N**ext **B**est **P**ose
SfM	**S**tructure **f**rom **M**otion
SG	**S**cene **G**rid
SR	**S**hot **R**egion
UAV	**U**nmanned **A**erial **V**ehicle
UVS	**U**nmanned **V**ehicle **S**ystem
UGV	**U**nmanned **G**round **V**ehicle
UUV	**U**nmanned **U**nderwater **V**ehicle
VPP	**V**iew **P**lanning **P**roblem

Chapter 1

Introduction

1.1 Motivation

Society continuously faces situations in which buildings are lost, by causes such as fire, earthquake, flood, war, or simply, erosion. Sometimes, a building or a monument which is of special interest for artistic, historical, architectural or religious reasons is partially or completely destroyed, causing a cultural heritage loss (see Figure 1.1). Typically a building of these has been built many years ago and has complex structures and detailed façades. It is inevitable attempting to repair this loss and try to restore or reconstruct these buildings. The building plans and photographic documentation are a helpful tool for the building preservation. With them, sometimes, like in the case of the Dresden Frauenkirche [1], a building that has been destroyed, can be satisfactorily reconstructed. Nevertheless, it is common not to have the plans of a building constructed hundreds of years ago, or in other cases, the building plans may have not been preserved; and even if the plans are available, they only describe course structures and the photographic documentation is often not exhaustive, making it impossible to preserve certain façade details.

Against these situations, nowadays, with the technological advances, another key point for the building preservation has appeared: a Three-Dimensional (3D) model of the building, a representation of the building in a digital space. With a highly detailed 3D model built before the (partial or complete) destruction of a building, an accurate restoration or even a precise reconstruction of it can be achieved.

(A) Frauenkirche in Dresden, Germany.

(B) Port-au-Prince Cathedral, in Port-au-Prince, Haiti.

(C) Alma College in St. Thomas, Ontario, Canada.

FIGURE 1.1: Example of buildings destroyed by: war (1.1a), in February 1945; by an earthquake (1.1b) in January 2010 and by a fire (1.1c) in May 2008. In these images two cathedrals and an old college are shown; other examples of historic buildings to preserve for different reasons could be mosques, synagogues, palaces or castles.

Computer graphics and computer vision are fields of study that produce 3D models in different ways. While the first one produces them by a combination of artistic and technical design processes, the second one does it by measuring the shape and visual properties of real physical objects. For creating a 3D model of a real building, a computer vision approach is needed, which will develop a surface reconstruction of it.

The process of 3D reconstruction consists of four main phases: planning a set of views, taking scans, registering the acquired data in a common coordinate frame of reference, and finally integrating the acquired data into a model. Whereas all steps are important, the first one is of special interest in this thesis. A manual view plan can be the root of an acceptable 3D building reconstruction, but sometimes not. Often, it will not cover completely a complex building surface, or some parts will be covered with a certain detail and other parts with another one, producing a low quality 3D model, which probably will not be enough to restore a detailed façade or reconstruct a complex structure of a building with no plans. As a consequence, new scans from the uncovered or wrongly covered surface will be necessary. Taking these new scans will extend the whole 3D reconstruction process and will increase its cost. For these reasons, an automatic view planner will be necessary.

1.2 Scope of this thesis

This thesis aims to present an automatic view planner of buildings for a later 3D reconstruction of them.

In the past, the coverage of buildings was either restricted by earthbound vehicles or by air-borne sensors to generate only low detailed building models. However, in the last years, the usage of Unmanned Aerial Vehicles (UAVs), commonly known as drones, is growing in this field [2–6] because of their flexibility. UAVs combine the advantages of earthbound vehicles and air-borne sensors, they are able to observe a whole 3D scene and to capture images of a building of interest from completely different perspectives.

On the other hand, when using a UAV, weight, time and security restrictions must be taken into account. The first one is of special interest because it will limit the choice of the viewing sensor since the weight that a UAV can carry is limited. A digital camera seems to be a good solution because of its small size, weight and cost. Although 3D information can be extracted from a single image [2, 7, 8], it is more difficult and solutions are not as robust as in Structure from Motion (SfM) systems, where 3D structures are estimated from Two-Dimensional (2D) image sequences which are coupled with local motion signals.

There are already some works [5, 6], which present 3D building models built from digital images acquired from UAVs. In these works the view planning step is manually designed, having to send occasionally the UAV several times until the complete surface is acquired, with the corresponding increase of cost and time. In addition these works have not computed 3D models from a building with a complex structures. This thesis presents an automatic system that gives a path in which there are a set of viewpoints for covering the complete surface of a target building of which a 3D reconstruction is desired. For this enterprise the system will compute the route from a rough model of the environment (e.g. a model of Level Of Detail (LOD) 1[1]), taking into account security and path length issues.

The detail of the 3D model is important, as discussed above, for correctly restore or reconstruct some buildings. If a 3D building model is computed with images taken from

[1]LOD refers to the different levels of detail in the representation of virtual models in applications such as building 3D reconstruction. Depending on the application, different levels of detail are needed. LOD 1 is according to CityGML [9] a level where only coarse structures are described.

far away of it, the final result will not have enough detail and some façade details could be lost. In a possible application, it could be desired to distinguish the bricks in a brick wall (see Figure 1.2). For this purpose, in the images the brick separation should not be smaller than the size of the smallest element, a pixel, in these images. As it can be appreciated, this constraint directly implies a certain distance to acquire images.

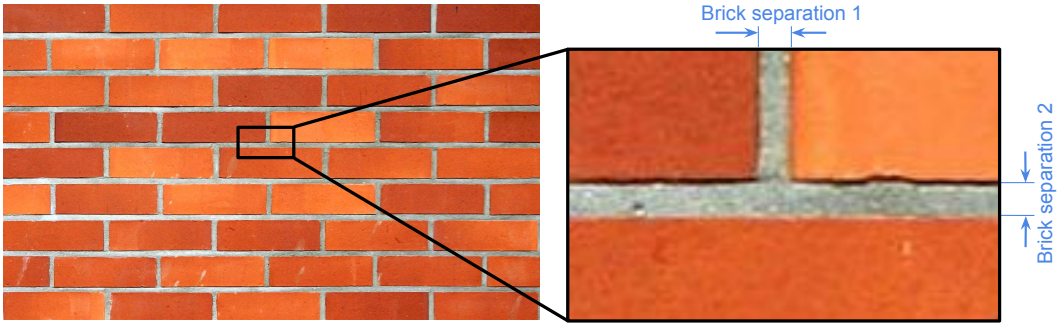


FIGURE 1.2: Brick wall with a detail of the brick separations.

The final route created in this thesis will be designed according to a desired detail of the future 3D model to be built. The result is thought to be used in the generation of highly detailed 3D building models for the preservation of buildings of special interest, city planning, or digital tourism.

1.3 Thesis outline

Chapter 2 gives background information relevant to this thesis. It is divided in three sections: 3D reconstruction from multiple images, view planning and path planning. The goal of this chapter is to give enough information to understand the next chapter.

Chapter 3 is the core of the thesis, describes the system proposed. It begins describing several assumptions that have been taken, after that the overall operation is explained and then its two main blocks, the View- and the Path-planners are described, as well as the approach to cover 3D scenes.

Chapter 4 shows the results of the evaluations of the proposed system in the previous chapter, for 2D and 3D environments; and finally, Chapter 5, gives a conclusion for the thesis, summarising the most relevant achievements, and discussing future work, that can be followed from this thesis.

Chapter 2

Related Work

This chapter gives theoretical background for understanding chapter 3, the core of the thesis, and reviews related work done in the main topics.

The first section explains how 3D information is extracted from multiple images and also which restrictions must these meet for a proper 3D reconstruction.

The second and the third sections review previous methods and work of the main two blocks treated in this thesis: view planning, the problem of covering an area with a sensor or multiple sensors, and path planning, the problem of finding a path from an initial location to a final location.

2.1 3D reconstruction from multiple images

In this section, it is explained how to extract 3D information from multiple images, since it is the background problem. This thesis presents a systems which is thought to be used for 3D reconstruction of buildings from multiple images. It is divided in four subsections, the core of 3D reconstruction from several images, triangulation; then the mathematical basis for the process, after that a brief description of SfM systems, and finally the role of UAVs in 3D reconstruction from multiple images.

2.1.1 Basic procedure

An image is basically a projection of a 3D scene onto a 2D plane, and, as a consequence depth is lost. A 3D point represented in a image is restricted to be in the line of sight. From a single image it is almost impossible to determine which point in this line corresponds to the image point. Some works [2, 7, 8] aim build 3D models from single images, but this is not the focus of this thesis.

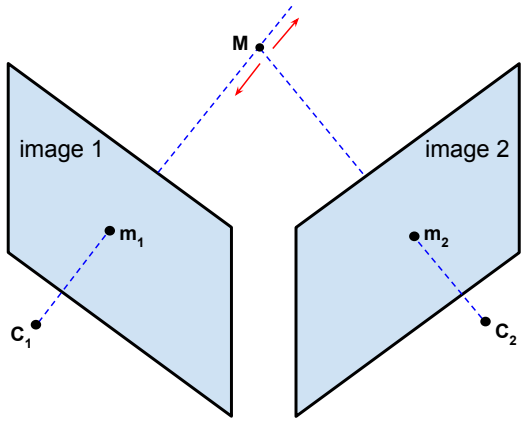


FIGURE 2.1: With two images of a static scene, the location of a 3D point, M, in the scene can be recovered from the projections, m_1 and m_2, in the respective images using triangulation.

On the other hand, if two images are available, the position of a 3D point can be found as the intersection of the two corresponding projection rays (see Figure 2.1). This process is known as triangulation and it requires the equations of the rays and, hence, complete knowledge of the cameras: their relative positions and orientations, as well as their optical settings. The process of determining all the camera parameters is referred to as *camera calibration*. Furthermore, in order to perform the triangulation process, the problem of finding corresponding points in several images needs to be solved. This problem is known as the *correspondence problem*.

According to [10], a distinction of 3D acquisition methods can be done, according to whether or not light sources are controlled. If they are, methods are known as *active*. If they are not, methods are known as *passive*. In this thesis, light sources are not controlled, and thus, when it is said 3D reconstruction or triangulation, it means passive 3D reconstruction or passive triangulation.

2.1.2 Image formation and mathematical basis

The pinhole camera is the simplest model of the image formation process in a camera. In this model, the camera is not more than a black box one side of which has a small hole. The rays of light from the outside world pass through this hole and fall on the opposite side of the box where a 2D image of the 3D environment outside the box (the scene) is formed. This pinhole image is in fact, the photo-negative image of the scene. The photo-positive corresponds to the projection of the scene onto a hypothetical plane that is situated in front of the pinhole camera at the same distance from the hole as the opposite wall on which the image is actually formed. In this section, the term *image plane* will always refer to this hypothetical plane in front of the camera. This plane is preferred to avoid sign reversals in this section. The centre of projection, the hole in the box will be referred to as the *camera centre* and the distance between this point and the image plane is called the *focal length* of the camera[1].

The amount of light that falls into the box through the small hole is limited, but by making the hole bigger, the amount of light can be increased, and as a consequence rays coming from different 3D points can fall onto the same point on the image, causing what is known as blur. A solution for this problem is using lenses, which focus the light. However, even the most perfect lens will come with a limited Depth Of Field (DOF) which means that only scene points within a limited depth range are imaged sharply. Within that depth range a camera with a lens basically behaves like the pinhole model.

A reference frame for the 3D environment containing the scene and fixed to the camera, will be used to simplify the problem. It is a right-handed and orthonormal reference frame whose origin is at the camera center. Its Z-axis is the principal axis of the camera and the XY-plane is the plane through the camera center and parallel to the image plane. The image plane is the one with equation $Z = f$. The principal axis intersects the image plane in the *principal point, p* (see Figure 2.2).

The projections onto the image plane cannot be detected with absolute precision. An image consists of many small elements, known as pixels, which are arranged in a rectangular grid, according to rows and columns. Pixel positions are typically indicated with a row and column number measured with respect to the top left corner of the image.

[1]The focal length is strictly the distance between the camera center and the image plane, when the camera is focused at infinity. A detailed explanation can be seen in Appendix A.

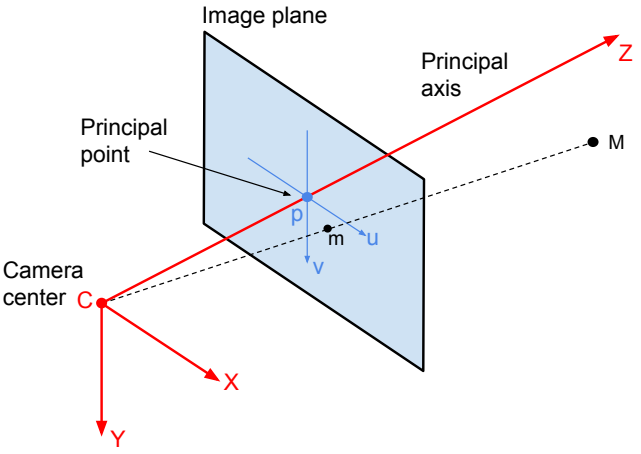


FIGURE 2.2: The camera-centered reference frame is fixed to the camera and aligned with its intrinsic directions. The coordinates of the projection m of a scene point M onto the image plane in a pinhole camera model with a camera-centered reference frame, are given with respect to the principal point p in the image.

These numbers are called the pixel coordinates of an image point. They will be denoted by (x, y), where the x-coordinate is measured horizontally and increasing to the right, and the y-coordinate is measured vertically and increasing downwards.

In a camera-centered reference frame the X-axis is usually chosen parallel to the rows and the Y -axis parallel to the columns of the rectangular grid of pixels, and at the same time, it induces an orthonormal uv reference frame in the image plane. The u- and v-axes induced in the image plane have the same direction and sense as those in which the pixel coordinates x and y of image points are measured. But, while pixel coordinates are measured with respect to the top left corner of the image, (u,v)-coordinates are measured with respect to the principal point, p. The transition from (u,v)- to (x,y)-coordinates for an image point m consists then, into apply offsets, related to the principal point, to each coordinate.

The image of a scene point M is the point m where the line through M and the origin of the reference frame intersects the image plane. If M has coordinates $(X, Y, Z) \in R^3$ with respect to the camera-centered reference frame, then an arbitrary point on the line through the origin and the scene point M has coordinates $\rho(X, Y, Z)$ for some real number ρ, which somehow represents the *depth* of the scene point M. According to [10], the pixel coordinates (x, y) of the projection m of the scene point M in the image are given by:

$$x = \alpha_x \cdot \frac{X}{Z} + p_x \qquad \text{and} \qquad y = \alpha_y \cdot \frac{Y}{Z} + p_y \tag{2.1}$$

where α_x and α_y are the focal length expressed in number of pixels for the x- and y-direction of the image and (p_x, p_y) are the pixel coordinates of the principal point.

More elegant expressions for the projection Equations 2.1 are obtained if the extended pixel coordinates for the image points are used. In particular, if a point m with pixel coordinates (x, y) in the image is represented by the column vector $m = (x, y, 1)^T$, then formula 2.1 can be rewritten as:

$$Z \cdot m = Z \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \begin{pmatrix} \alpha_x & 0 & p_x \\ 0 & \alpha_y & p_y \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} X \\ Y \\ Z \end{pmatrix} \tag{2.2}$$

If one interpretes the extended pixel coordinates $(x, y, 1)^T$ of the image point m as a vector indicating a direction in the world, then, since Z describes the *depth* in front of the camera at which the corresponding scene point M is located, the remaining 3×3 matrix represents the transformation that converts world measurements (expressed in centimeters, millimeters, $\cdots$) into the pixel metric of the digital image. This matrix is called the *calibration matrix* of the camera, and it is generally represented as:

$$\mathbf{K} = \begin{pmatrix} \alpha_x & s & p_x \\ 0 & \alpha_y & p_y \\ 0 & 0 & 1 \end{pmatrix} \tag{2.3}$$

where α_x, α_y, p_x, p_y are the parameters described above. The additional scalar s is called the *skew factor* and models the situation in which the pixels are parallelograms (for example, not rectangular). In fact, s is inversely proportional to the tangent of the angle between the X- and the Y -axis of the camera-centered reference frame. Consequently, $s = 0$ for digital cameras with rectangular pixels, which will be the case in this thesis.

The parameters α_x, α_y, s, p_x, and p_y of the calibration matrix $\mathbf{K}$ describe the internal behavior of the camera and are therefore called the internal parameters of the camera.

Furthermore, the projection Equation 2.2 of a pinhole camera with respect to a camera-centered reference frame for the scene is compactly written as:

$$\rho \cdot m = \mathbf{K} \cdot M \tag{2.4}$$

where $M = (X, Y, Z)^T$ are the coordinates of a scene point M with respect to the camera-centered reference frame for the scene, $m = (x, y, 1)^T$ are the extended pixel coordinates of its projection m in the image, $\mathbf{K}$ is the calibration matrix of the camera. As it has been introduced before, ρ is a positive real number and it actually represents the *depth* of the scene point M in front of the camera. Due to the structure of the calibration matrix $\mathbf{K}$, the third row in the matrix Equation 2.4 reduces to $\rho = Z$. Therefore ρ is called the *projective depth* of the scene point M corresponding to the image point m.

When more than one camera is used, or when a camera is taking several photos, a world frame is needed. In this frame, position and orientation of a camera in the scene are described by a point C, indicating the centre of projection, and a 3×3 rotation matrix $\mathbf{R}$ indicating the orientation of the camera-centered reference frame with respect to the world frame. As C and $\mathbf{R}$ represent the setup of the camera in the world space, they are called the external parameters of the camera. The process of determining a camera parameters, internal and external, is known as camera calibration. Traditional 3D reconstruction techniques had a separate, explicit camera calibration step. Self-calibration techniques use the same images utilized for a 3D scene reconstruction, to extract the internal and external camera parameters.

Finally, the coordinates of a scene point M with respect to the camera-centered reference frame are found by projecting the relative position vector $M - C$ orthogonally onto each of the coordinate axes of the camera-centered reference frame. As a result, Equation 2.4, the projection m of the scene point M in the image is given by the general projection equation:

$$\rho \cdot m = \mathbf{K} \cdot \mathbf{R}^T \cdot (M - C) \tag{2.5}$$

This equation is the key for recovering a 3D information of a 3D point. As it was said in the beginning, with only one view of the point it will not be possible to recover the

position of a 3D point. At least two views will be needed. Equations 2.6 (derived from Equation 2.5), could be the projection equations of two different cameras, such in the situation described in Figure 2.1:

$$\begin{aligned} M &= C_1 + \rho_1 \cdot \mathbf{R}_1 \cdot \mathbf{K}_1^{-1} \cdot m_1 \\ M &= C_2 + \rho_2 \cdot \mathbf{R}_2 \cdot \mathbf{K}_2^{-1} \cdot m_2 \end{aligned} \tag{2.6}$$

This system of equations can be solved, if the internal and external parameters of the cameras are known, for the five unknowns X, Y, Z, ρ_1, and ρ_2. When the cameras are not internally and externally calibrated, then it is not immediately clear how to perform triangulation from the image data alone, although it is possible. In both cases, however, the system is required to be rank-deficient, which is guaranteed if the points m_1 and m_2 are in correspondence (i.e., their projecting rays intersect) and therefore special relations hold (epipolar relations, see Appendix B).

2.1.3 Structure from motion

If the scene to reconstruct is static, like a target static object, the two images could be taken by two different cameras, or by placing the same camera at the two positions, and taking the images in sequence. In this second situation, more than two images could be taken while moving the camera. Such strategies are referred to as SfM. If images are taken over short time intervals, it will be easier to find correspondences, e.g. by tracking feature points over time. Moreover, having more camera views will yield object models that are more complete. Last but not least, if multiple views are available, the camera(s) need no longer be calibrated beforehand, and a self-calibration procedure may be employed instead. These properties render SfM a very attractive 3D acquisition strategy.

Automated systems based on SfM process image sequences to first recover the camera poses and then, a sparse (point cloud) reconstruction of the scene. The situation described in this thesis is suitable for using a SfM system. A 3D reconstruction from a set of images will be desired, while the camera poses will not be exactly known, and thus, an external calibration will be needed.

From the sparse reconstruction computed in a SfM system, dense multi-view stereo algorithms can generate a dense mesh model. Two examples of systems which use large image sequences with SfM approaches to recover 3D structure are [11] and [12].

In [11] an interactive system for modeling architectural scenes from an unordered collection of photographs using SfM is developed. Features are detected in all images, matched across image pairs, then used to robustly estimate the camera poses and the position of points in the sparse 3D point cloud in an incremental fashion. The optimal camera and point parameters are ones that minimize the error between the re-projected 3D points and the detected 2D interest points.

The Photo Tourism System [12] uses SfM to compute the 3D locations and poses of all the cameras taking the images, along with a sparse 3D point-cloud model of the scene. This system browses and organizes large photo collections of popular sites and exploits the common 3D geometry of the underlying scene. To reconstruct the required 3D information, the system handles large photo collections taken by different cameras in widely different conditions, which increases the complexity of the problem. On the other hand, the goal of this thesis is to give a tool for taking a set of images of a concrete building, by the same camera in the (almost) same light conditions.

2.1.4 UAVs and 3D reconstruction from multiple views

According to the Unmanned Vehicle System (UVS) International definition [13], a UAV is a generic aircraft designed to operate with no human pilot onboard. They originally had military applications, but are increasingly applied into civilian fields, such as aerial surveying of crops, aerial footage in film making, search and rescue operations, inspecting power lines and pipelines, counting wildlife, or aerial photography.

UAVs are the aerial version of Unmanned Ground Vehicles (UGVs) and Unmanned Underwater Vehicles (UUVs), which are systems that operate also without an on board human presence, in terrestrial and aquatic applications. Based on size, weight, endurance, range and flying altitude, UVS International defines three main categories of UAVs:

- Tactical UAVs. In this first category, micro, mini or close-range systems are enclosed. Their mass ranges from few kilograms up to 1.000 kg, their range from few

kilometres up to 500 km, their flight altitude from few hundred meters to 5 km, and their endurance from some minutes to 1-2 days.

- Strategical UAVs. Systems included here are high altitude long endurance, stratospheric and exo-stratospheric systems which fly higher than 20.000 m altitude and have an endurance of 2-4 days.

- Special tasks UAVs, like unmanned combat autonomous vehicles, lethal and decoys systems, which have longer endurances to 4 days, and are basically used for military applications.

The principal airframe types are fixed and rotary wings while the most common launch methods are, beside the autonomous mode, air-, hand-, car/track-, canister-, bungee cord launched.

FIGURE 2.3: Octocoper UAV equipped with a DSLR camera used by the company ATEA Data Ltd. [14].

In the last years, micro UAVs[2] have started being used in diverse applications ranging from monument conservation to city planning, for 3D building reconstruction [2–6]. This development can be explained by their flexibility and the spreading of low-cost platforms. Classical tools for acquiring building information consist in using earthbound vehicles or air-borne sensors. While the first ones can cover only building façades, air-borne sensors help to determine the coarse shape of buildings with low detail. UAVs combine the advantages of earthbound and air-borne sensors. Their flexibility allows exploiting the

[2]According to UVS International, a micro UAV is a tactical UAV that weighs less than 5 kg and whose endurance is smaller than 1 hour. However, this is more a guide definition than a strict one.

whole 3D space and their possibility of different imaging positions permits the acquisition of a whole building with an adjustable detail.

For weight constraints, laser scanners, such as Light Detection and Ranging (LIDAR) sensors, can not be mounted on a micro UAV and digital cameras seem to be a good solution because of thir small weight and cost. Digital Single-Lens Reflex (DSLR) cameras[3] are a type of cameras which provide high quality images with a light weight body.

Nowadays, UAVs are usually equipped with Global Navigation Satellite Systems (GNSSs) and Inertial Measurement Unit (IMU) devices [15] and because of this, they can fly along a flight path pre-defined by way points. The flight path usually is planned in advance for shortening the flight time as well as the overall time in the field. This is important, since their time of operation is restricted. With a predefined flight path, images can either be taken at some way points with a desired position and orientation or in dense image sequences using the continuous shooting mode of the camera. However, a navigation system typically carried by a UAV, such as the Global Positioning System (GPS), has an error of 7-8 meters [16]. As a consequence, images will probably be taken in unknown concrete points. For this reason, the acquired images by a UAV are suitable to be used in a SfM (see previous section).

This thesis presents a system which calculates automatically the points where images of a target building should be acquired. Despite extensive literature review no publication on automatic path planning for building reconstruction could be found. Works until now aim to present a complete 3D building reconstruction chain and their focus is not on the path planner but on the overall 3D process [3–6]. As a consequence, they use a manually designed path and against the uncertainty of acquiring the necessary images, those works usually take large image sequences from which a small set of images is selected manually for the reconstruction process. For simple structures it can be a solution, but for complex ones, a manual plan will not cover the complete surface or will do it with a low detail.

An example can be seen in [5]. Here, a 3D reconstruction of the Castle Landenberg in Switzerland using a UAV is accomplished. In this work, two path flights were designed and in those paths image stripes were acquired. One of those flight paths was flown twice to ensure the acquisition of a sufficient number of high quality images. In total,

[3] A DSLR camera is a digital camera combining the optics and the mechanisms of a single-lens reflex camera with a digital imaging sensor, instead of photographic films.

72 aerial images were taken and from them, 18 were selected manually. As a result, 54 useless images were acquired. This thesis aims to give a set of points from which to acquire a set of minimum necessary images for a latter 3D building reconstruction with a desired detail and provide the security that the target building has been completely covered, and thus, eliminate the need to send a UAV more than one time for the same path, with the corresponding increase in cost and time of the overall process.

2.2 View planning

The main problem of this thesis is a visual task of covering or observing a region of interest with a robot, a drone. This becomes a problem of selecting a set of positions of multiple sensors or a single sensor, in our case a digital camera carried by the drone. This problem has been already studied in computer vision and is known as sensor planning problem or View Planning Problem (VPP). It consists of automatically computing sensor positions or trajectories given a task to perform, the sensor features and a model of the environment. An early survey about the VPP was done in [17], where three distinct areas of research were identified according to the vision task: object feature detection, model-based object recognition and localization, and scene reconstruction.

1. Object feature detection. Those methods aim to automatically determine vision sensor parameter values for which particular features of a known object in a known pose satisfy particular constraints when imaged. For example, the features can be required to appear in the image as being visible, in the Field of View (FOV), in the DOF, in-focus, or magnified to a certain agreement. These planning techniques draw on the considerable amount of a priori knowledge of the environment, the sensors, and the task requirements. Because the identities and poses of the viewed objects are known, the sensor parameters are usually preplanned off-line and then used on-line when the object is actually observed.

2. Model-based object recognition and localization. This group assumes some knowledge about the objects that can appear in the scene, and the task is to develop sensing strategies for model-based object recognition and localization. Sensing operations are chosen according to their capacity of identifying an object or determining its pose. The a priori known information about the world in the form

of models of possible objects, sensor models, and information acquired to this point are compiled into recognition/localization strategies. Most approaches in this area follow a common line. Usually, a search is performed in the space of object identities/poses employing a hypothesize-and-verify paradigm:

1) Hypotheses are formed regarding the object identities and poses.

2) These hypotheses are assessed according to certain metrics.

3) New sensing configurations are proposed based on a given criterion until a stopping condition is reached.

In order to limit the search of sensor parameter space in this hypothesize-and-verify paradigm, a discrete approximation of this space is commonly employed.

3. Scene reconstruction. In this third group, methods attempt to build a model of the scene incrementally by successively sensing the unknown world from effective sensor configurations using the information acquired about the scene up to this point. At each step, new sensor configurations are chosen based on a particular criterion. The sensory information acquired at each step is then integrated into a partially constructed model of the scene and new sensor configurations are generated until the entire scene has been explored. While there is no a priori known scene information that could be used in this problem, the iterative sensing strategy is guided by the information acquired to each step.

The authors described the VPP in the initial efforts of researchers to address a problem with many degrees of freedom. It was noted that future work was required for incorporating more realism into the object models and relaxing some of the constraints made on developed systems.

A lot of research has been addressed since then in the three described areas. Scene reconstruction is of special interest for this thesis, as its background problem is 3D building reconstruction. In order to reconstruct an object or scene, a set of views are needed. Finding the minimum number of views which reveal the greatest amount of previously unknown information has been a topic of extensive research in the last years in the sensor planning field. This problem is known as Next Best View (NBV) problem or Next Best Pose (NBP) problem. According to [18], there are two major methods in the literature followed to solve the NBV problem: search based [18–21] and occlusion based methods [22–24]. Search based methods use optimization criteria to search a

group of potential viewpoints for the best view and occlusion based methods use the occlusion boundaries in the image of the current view to choose the next view. Some of this works will be described next.

One of the first publications on the NBV problem is [19]. Here, two algorithms are presented for determining the next best view using partial octree models. The regions that have been scanned were labeled as seen, empty and unseen. In the first algorithm, the next view position is the point at which the greatest surface area of unseen nodes is visible. The second algorithm uses information about the node faces in the octree which are common to both unseen and empty nodes.

In [22], the authors aim to completely acquire an object's surface based on the analysis of occlusions. A priori knowledge of the environment and the sensor is given to the system, and it returns the next view position from which a complete range image of the surface visible to the camera may be obtained. The system then computes new scanning planes for further 3D data acquisition based on the discontinuities (occlusions) in the most recent range image.

Relevant work is presented by Pito in [25]. He presents a solution for the NBV problem for surface acquisition of a priori unknown object. The NBV was determined as the sensor position that maximizes the unseen portion of the object volume; to do that, an objective function is maximized. An objective function is an optimization of the visibility state for all unknown points in the current model. An algorithm is described and implemented as a part of an automated surface acquisition system which used a triangulation based range scanner showing satisfactory results for two or three handspan objects, which can be self-occluded.

An objective function is also used in [18]. Here, it is presented a NBV system which consists of three parts: the Range Scanner, the Model Builder, and the NBV Decision Maker. The NBV Decision Maker examines the unknown information in the volumetric model and uses an objective function to determine the best place to position a new viewpoint. To simplify the problem, all the possible viewpoints are in a view sphere thanks to the assumption that the size and the center of the object to be reconstructed is roughly known. The objective function (see Equation 2.7) uses the unknown information in the current model to calculate the next sensor pose. The NBV is defined as the view from which the largest number of unknown area can be seen. The pose of the next

best viewpoint, $\overrightarrow{P}$, is the position and orientation vector of the best view from which to acquire new data, chosen from the set of potential viewpoints, $\overrightarrow{P_i}$. The updated volumetric model is contained in OG and the visibility state, V, of a given point, n, is the visibility of this point from the current viewpoint.

$$\overrightarrow{P} = max_i \left(\sum^{OG(n)|\overrightarrow{P_i}} V(n, unknown) \right) \tag{2.7}$$

This equation is an interesting concept and it will be used for choosing the viewpoints.

On the other hand, the sensor planning problem has also been studied in computational geometry and robotics with the 2D problem known as *art-gallery problem* [26, 27]. This problem consists in finding the minimum number of guards, which are necessary to cover a whole art gallery. In the computational geometry version of the problem, the gallery is represented by a simple polygon and each guard is represented by a point in the polygon. A set S of points is said to guard a polygon if, for every point p in the polygon, there is some $q \in S$ such that the line segment between p and q does not leave the polygon.

Some interesting 2D path optimization problems resulting from the previous one are the *watchman route problem* and the *zookeeper's problem* [28, 29]. The watchman route problem goal is to compute the shortest route that a watchman should take to guard an entire area with obstacles given only a map of the area. The challenge is to make sure the watchman reaches behind every corner and to determine the best order in which corners should be visited in. The zookeeper's problem is slightly similar (see Figure 2.4).

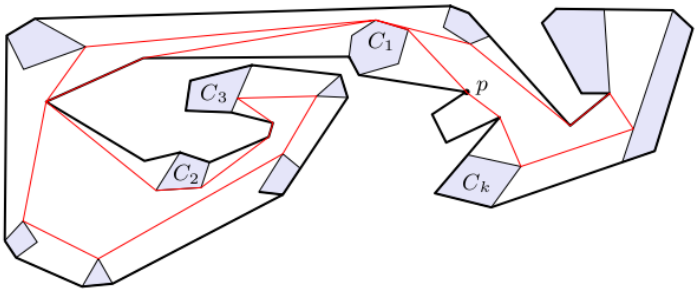


FIGURE 2.4: Zookeeper's Problem. Given a simple polygon (the zoo) with a set of k disjoint convex polygons (the cages, C_1, C_2,..., C_k) inside it, such that each cage shares an edge with the zoo, the goal is finding a shortest route in the interior of the zoo that touches each cage without entering into their interiors starting in a given point, p, and ending in the same point.

The object reconstruction problem is somewhat related to these classic computational geometry problems. Still, there are additional complexities, such as 3D scenarios or restricted visibility.

2.3 Path planning

In robotics, the process of converting high-level specifications of human tasks into low-level descriptions of how to move to a robot or machine, in this case a UAV, is referred as motion planning or path planning. Path planners or planning algorithms are tools that create paths from an initial state to a goal state according to a criteria and a given map of the environment.

In our case, the states will be positions and the singularity of the path to design is that between the starting and ending position, it should go through a set of other positions (the viewpoints). Length and security issues must be taken into account in order to design an acceptable path. An optimal path in terms of length which does not take into account security issues will not be interesting and the same the other way around, and as a consequence there must be a balance.

One option is to use an algorithm that finds a path in a given map. A graph is one way to represent a map, more precisely, it is a representation of a set of objects where some pairs of them are connected by links. The interconnected objects are represented by mathematical abstractions called vertices, and the links that connect some pairs of vertices are called edges [30]. A directed edge has a direction and an undirected edge does not. Graphs can be directed or undirected depending on whether its edges are directed or undirected. If a graph has edges of each kind, it is called mixed. A graph can be weighted if a number, usually called weight, is assigned to each edge. Such weights might represent, for example, costs, lengths or capacities depending on the problem [31]. A graph could be built as to describe a map, in which the weights in the edges represented the cost to move from one position to another. In Figure 2.5, a directed graph with 6 vertices and 10 directed edges is represented.

In graph theory, the *shortest path problem* is the one of finding a path, p, between two nodes in a graph such that the sum of the weights, w, of its constituent edges is minimized (see Equation 2.8).

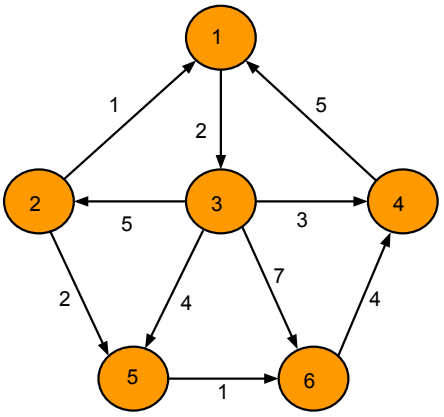


FIGURE 2.5: Directed graph with non-negative edge path costs.

$$min \sum_{p \in P} w(p) \tag{2.8}$$

This problem can be defined for undirected, directed or mixed graphs. There are several algorithms that solve this problem whose solutions can be considered as optimal. Some of them are the Dijkstra's, the Bellman-Ford or the A*search algorithms, among others [32, 33].

Dijkstra's algorithm solves the single-source shortest-paths problem on a weighted, directed graph for the case in which all edge weights are non negative. It is asymptotically the fastest known single-source shortest-path algorithm for arbitrary directed graphs with unbounded non-negative weights [34]. On the other hand, the Bellman-Ford algorithm is a bit slower, but more versatile, as it is capable of handling graphs in which some of the edge weights are negative numbers. Finally, the A*search algorithm uses heuristics to try to speed up the problem.

Those algorithms give optimal solutions in terms of length and as they have been studied since the 1950s, nowadays there are implementations of them in different programming languages. The key point is then to design the graph. For this thesis it will be desired that the graph takes into account the security issue.

On the other hand, *potential field methods* are motion planning approaches that take into consideration the security problem, basically they give the safest path in a environment with obstacles [35]. The robot's configuration is treated as a point in a potential field that combines attraction to the goal, and repulsion from obstacles. While the potential

field principle is particularly attractive because of its simplicity, some problems have been identified, like for example that they can become trapped in a local minima of the potential field, and fail to find a path. Potential field methods inspired the creation of the cost function that will be described in the next chapter.

Chapter 3

Path Planner for Optimal Coverage System

In this chapter, an approach to solve the presented problem is given. It has mainly two blocks, the View-planner, which calculates a set of viewpoints in a 2D map, and the Path-planner, which links the previous ones using a cost function and a planning algorithm.

The first section describes several assumptions that have been taken to solve such a complex problem like the presented one. The second section explains the general proceeding, the third and fourth sections describe the View-planner and the Path-planner, respectively, and finally an approach to 3D is given.

3.1 Assumptions

Four different kind of assumptions will be taken: assumptions related to optimality, to the representation of the scene, assumptions related to the sensor which will acquire information of the scene and finally assumptions related to the viewpoints, which will be the points from which information will be acquired.

3.1.1 Optimality

The title of this thesis is "Path Planning for Optimal Coverage under Security Constraints", but what does *optimal coverage* means? According to the meaning of the word *optimal*, optimal coverage can be understood as the best, most desirable or most favourable coverage of a region with certain purposes (e.g. a 3D reconstruction), under some restrictions. It implies that there is only one single path for the optimal coverage, the other paths which are closer to the optimal one, can be labelled as near optimal paths. In this work, two restrictions are taken into account, the security and the path length and according to this, if a path allows for a desired 3D reconstruction[1], avoids obstacles and is proved to be the shortest one, it will be described as the optimal path, however it is really difficult to measure if the allowed 3D reconstruction is the desired one or if the path is the shortest one. As a conclusion, the resulting paths in this thesis will be labelled as near optimals, and they will aim to allow for the desired 3D reconstruction, while considering security and length issues.

3.1.2 Environment

The final path will be calculated from a rough 3D model of the environment, it means that it will be known where the target building can be, but not precisely. The idea is to use this system iteratively, refining the model until reaching a satisfactory result. Henceforth the part of the rough model that represents the target building will be referred simply as the target building.

This rough model will be described as a 2D/3D array in which each element represents a specific area/volume in the discrete tessellation of the workspace, the *Scene Grid* (SG) from now on. Some works [21, 36–38] use a 3D array with binnary values, an occupancy grid, for describing the scene according if the space is occupied or not. This approximation is useful when there are no obstacles, but in this thesis the presence of obstacles is taken into account and there must be a way to distinguish a possible obstacle (e.g. a tree or another building). This is the reason why the elements of the SG can take three different values, in accordance if they belong to a target building, to an obstacle or to unoccupied space, *free space*. The drone and the camera will be represented as a

[1]The desired 3D reconstrion will be the one computed with a set of images that meet a desired detail.

single point in the SG, the *sensor point*, and the final path will be described as a set of sensor points, indicating which are shot points and which not. The target building will be described as a polyhedron with four main different kinds of surfaces: the façades, the rooftop, terraces and opposed terraces (see Figure 3.1). In a given building, maybe not all the kinds of surfaces are present (see Figure 4.8). Slices from the building are supposed to be polygons of diverse shapes.

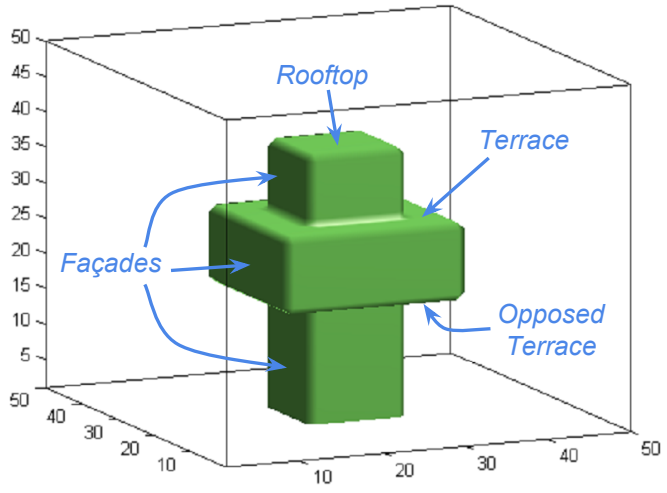


FIGURE 3.1: Different types of building surfaces.

Buildings and obstacles are regions where the UAV can not obviously operate and they can be understood as prohibited regions. In addition, a *security distance* will be defined as a distance to a building or obstacle, beneath of which the UAV will be also not permitted to go. This parameter is used to build the Shot Region (SR) in Section 3.3 and the cost function in Section 3.4.

Finally, an *initial point* will be given to the system, from which all the path will be designed. It will be the initial point, but also the final one. The point of the first shot will depend on this point. From the position and orientation of this point, only the position is of interest for designing the path.

3.1.3 Sensor

The UAV will most likely carry a digital camera such a DSLR one to take shots of a target building. Two assumptions are taken related to the camera, the first one is that the camera will be focused at infinity and the second one is that the optical parameters (i.e, specifications or internal parameters) of the camera will be known, and consequently, it

can be said that the camera will be internally calibrated. Assuming ideal illumination conditions, the parameters of interest in this thesis are the sensor format, the image resolution and the focal length.

- The sensor of a digital camera is a device that converts an optical image into an electronic signal. The image sensor format (shape and size) determines the viewing angle of a particular lens (together with the focal length).
- The image resolution of a camera is the capability of the sensor to observe or measure the smallest object clearly with distinct boundaries, usually it is expressed in pixels.
- The focal length, is the distance between the optical center of the lens to the image sensor, when the camera is focused at infinity (see appendix A). It is a fundamental property of a lens and determines the magnification of the image projected onto the image plane.

3.1.4 Viewpoints

As it has been said, a shot will be taken from some points called viewpoints or shot-points. There are a set of parameters that define a viewpoint, divided in external and internal parameters. The internal parameters are the optical parameters of a camera that have been described in the previous section. The following is a description of the external parameters, the position and the orientation, which are independent of the camera mechanisms.

- Position is given by three positional degrees of freedom of the sensor (x, y, z), it can also be understood as the position vector of the sensor point.
- The orientation is typically given by three degrees of freedom, the roll, pitch and yaw angles (φ, θ, ψ) (see Figure 3.2), but it must be taken into account that the camera will be carried by a drone, which has limitations in terms of orientation since it can only be oriented according to the roll angle. Nowadays when a drone carries a camera, it is common to use a *camera positioner* and depending on it, the viewpoint will be able to have 1, 2 or 3 orientation degrees of freedom.

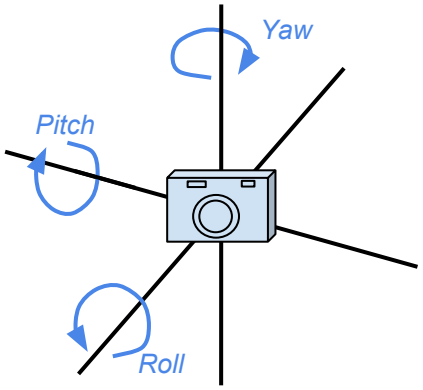


FIGURE 3.2: Yaw, pitch and roll angles. The yaw angle rotates on the camera's up vector. The roll angle rotates on the camera's direction vector. The pitch angle rotates on the cross product of the camera's up and direction vector.

It will be assumed that there will be three possible camera configurations for taking images and in the three cases the viewpoints will have only 1 degree of orientation freedom, the yaw angle. In the first configuration, the roll and pitch angles will be fixed to 0^o, this configuration is thought to be used for covering the building façades. In the second and third ones, the roll angle will be also fixed to 0^o, but the pitch angle will be fixed to a certain angle for covering the rooftops, the terraces and the opposed terraces. For the rooftops and terraces the pitch angle will be fixed to a negative angle, and for the opposed terraces, it will be fixed to a positive one. These configurations can be seen in Figure 3.3.

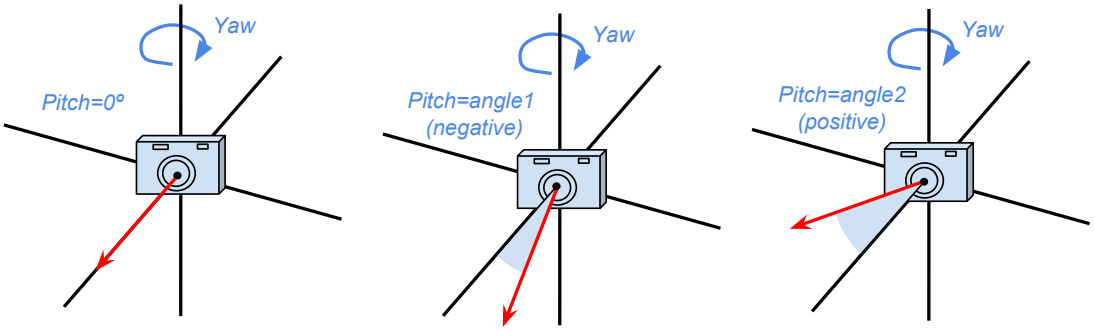


FIGURE 3.3: The three different camera configurations for this thesis.

In this thesis the internal parameters will remain constant in all the viewpoints, and since a path is a composition of positions and orientations, the internal parameters will be used (together with a desired image detail) for calculating the external ones in the final path.

On the other hand, there are several constraints that will constrict the region of acceptable viewpoints. Early studies of these constraints are done in [17, 39–41]. Assuming ideal illumination conditions, the constraints taken into account in this thesis are visibility, FOV and desired detail.

- Visibility. The surface of the target building must first be visible from the sensor viewpoints. It means that all lines of sight from the camera to the building surface are not obstructed by any obstacle. To determine if a point is visible from a viewpoint, all the points between them are checked. If the sum of those values is equal to the free space value, it is considered as visible.

- Field of view. The previous constraint requires that all the light rays from a given point in the target building reach the sensor. The FOV constraint requires that these rays reach the active area of the sensor. In the case that a light ray from a given point lies outside the active area of the sensor, it will not be observable, and thus not recorded. The FOV at a given distance is determined from the focal length and the image sensor size of a camera as it can be seen in Figure 3.4.

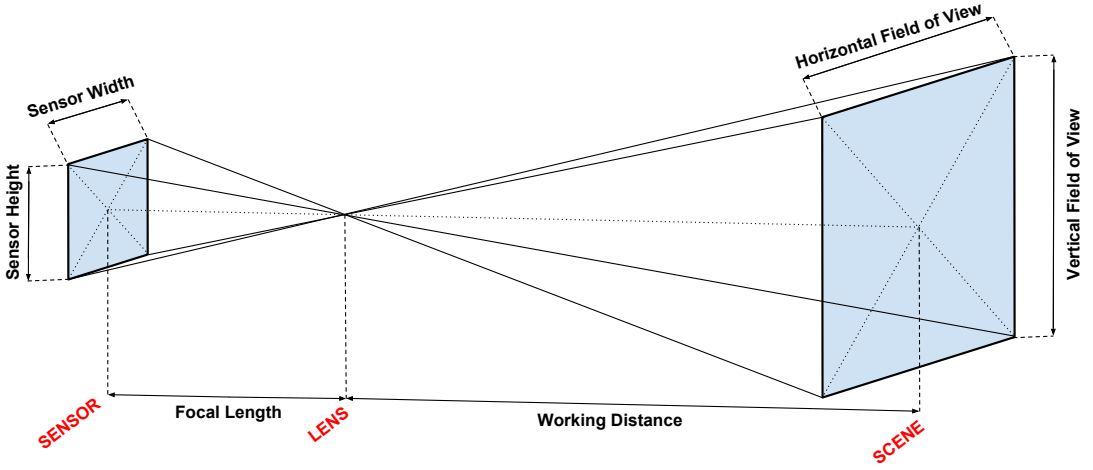


FIGURE 3.4: Relationship between focal length, sensor size, working distance and field of view.

- Desired detail. It describes the desired amount of real distance represented by a single pixel. It will an input together with the rough model of the scene and the camera parameters. As it can bee seen in Figure 3.5, this detail defines a distance, the *optimal distance*, which is the distance between the camera and the target building for taking appropiate photos according to the desired detail. Given this

detail and some camera parameters it is possible to calculate this distance by using the intercept theorem [42].

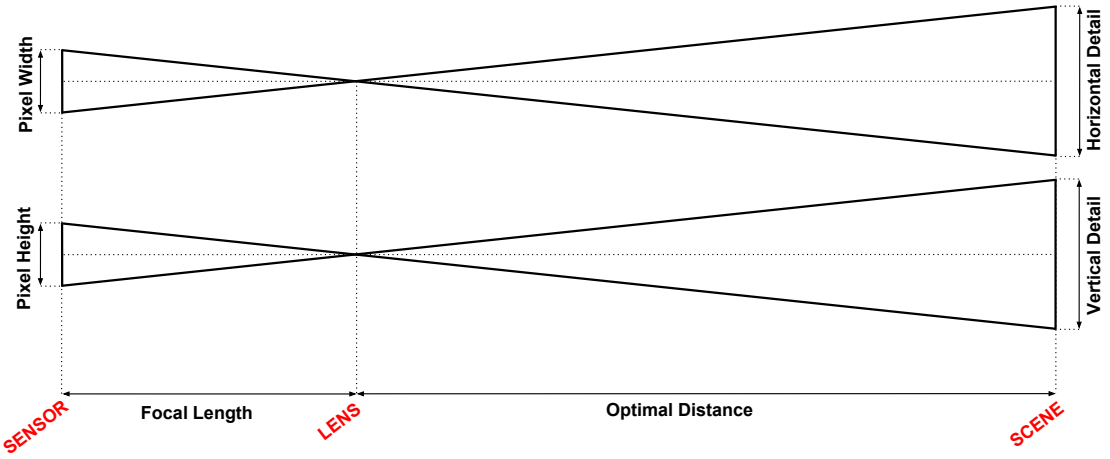


FIGURE 3.5: Relationship between the image detail and the optimal distance.

When a shot at a certain distance is taken, there is only one point which is at exactly this distance. For this reason and also due to the presence of obstacles, it will be impossible to cover all the surface of the target building from the optimal distance, however it will be the reference distance.

3.2 General proceeding

The system, which has been implemented in MATLAB with a Graphical User Interface (GUI), needs a set of inputs. A 2D or a 3D SG, an initial position in the map, a digital camera and a desired image detail. After selecting these inputs, the system computes a path or a set of paths, which are showed visually, together with a road map. The road map describes which points must been followed and in which ones a photo must be acquired and with which orientation.

If a 2D SG is given, the path will only have two positional and one orientation degrees of freedom (x, y, and the yaw angle). On the other side, if a 3D SG is selected, the path will contain three positional and two orientation degrees of freedom (x, y, z, the yaw and the pitch angles).

The basic idea of the system is dividing a building into several slices and for each slice solve a local 2D problem. With this idea, the façades will be covered, but not the rooftop, the possible terraces or opposed terraces; for this reason, in the beginning it is

detected if there is any of these special surfaces and in the case that there are, a path is computed for each one. Finally, a path for the façades is calculated following the slice approach.

Any of those paths is formed by three sub-paths: an initial, a coverage and a final path. The initial path goes from the initial point to the closest point in the SR, which in fact will be the viewpoint where the first shot will be taken. The SR is explained in the next section. The coverage path is the one that covers all the target surface and it is calculated by the View-planner. Finally, the final path links the last shot and the initial point.

3.3 View-planner

The View-planner computes a set of viewpoints from which to take shots of a target building slice given an initial viewpoint. The resulting images taken from those viewpoints will be supposed to be used in a SfM system.

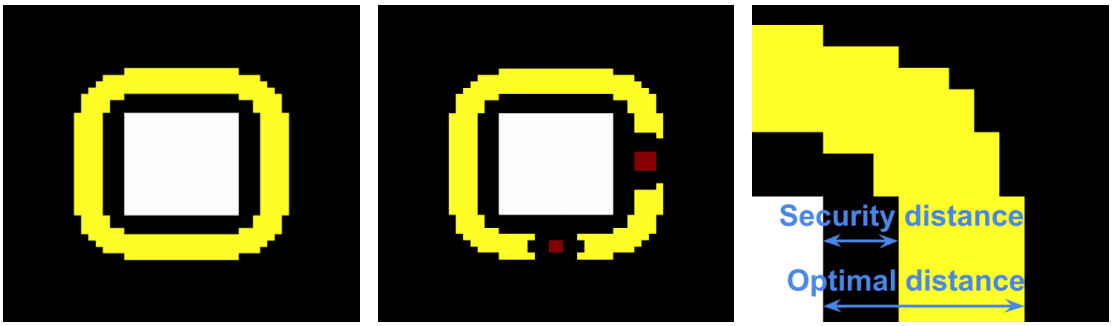
3.3.1 Shot region

First of all, the SR is calculated. It will be the region where the viewpoints will be placed. This area is limited by two distances: the security distance, and the optimal distance. For calculating the SR, Euclidean distances have been used, d_E.

$$\begin{aligned} SR &= A \cap B \\ A &: d_E(T, A) \leq d_{OPT} \\ B &: d_E(TO, B) \geq d_{SEC} \end{aligned} \tag{3.1}$$

Where T represents the target points and TO describes de target and obstacle points.

As explained in section 3.1.2, being under the security distance to the target building or an obstacle, is considered dangerous, this is the reason because this distance is the lower limit of the shot-region. The optimal distance is the ideal distance for taking photos perpendicularly to the target surface, according to the desired detail. It will be the upper limit because farther away to this distance any picture will have a worse detail



(A) 2D scene without obstacles (B) 2D scene with obstacles (C) Limiting distances

FIGURE 3.6: SR (in yellow) for the same 2D SG, without and with obstacles (in red).

compared to the desired one. The view planner looks for viewpoints which are in this region.

3.3.2 Surface organization

From a initial viewpoint, the surface of the target building is organized. This process has three main steps: select a first point, choose a direction and select the rest of points until the last surface point is arranged.

The initial viewpoint is supposed to take an image of a concrete part of the target building, it means that a set of surface points will be seen. From these points, the two extreme ones are detected (see Figure 3.7) as the two points that are only linked to one single surface point imaged in the first shot. After that, one of those two points will be selected randomly as the first surface point. The random selection is reasonable due to the fact that both extreme points are equal candidates to be the first point, it will only affect in the direction of the organization.

The way of selecting the first point implies that the first image acquires a connected set of surface points. If the points are are not connected (e.g. the target itself occludes some points), the system will have problems. After that, the second point is selected as the neighbourhood point that is visible in the first image. The other neighbourhood point is not seen because the first point is one extreme imaged point. As it can be seen, it is assumed that any surface point is connected to only two points, which may lead to problems if the target shape is sharp. It may seem a drawback, but it can be thought that for the same shape, if the resolution is increased, this problem will disappear.

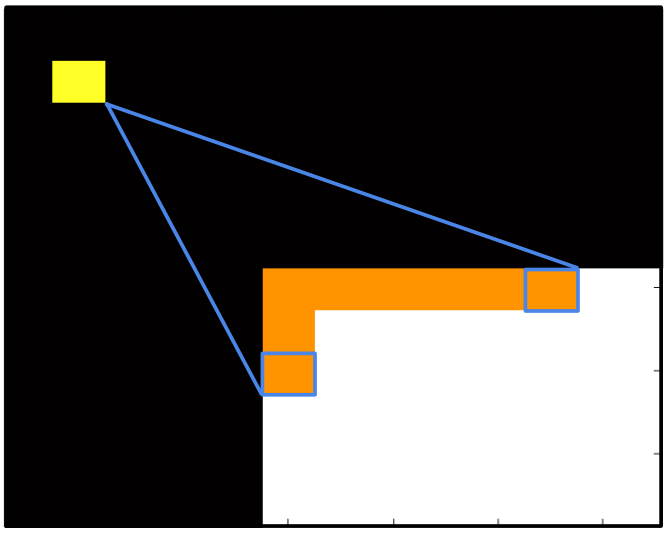


FIGURE 3.7: The first viewpoint (in yellow) takes an image. The two extreme surface points seen in the first image are detected (in blue). The target surface will be organized from one of those two points.

After having selected the first two surface points, the rest of them are selected as the adjacent ones to their previous ones which have still not been organized. As a result, a vector of organize points is built and used by the View-planner to cover exactly all the surface.

3.3.3 Viewpoint selection

After having organized the target's surface, two checking loops are done from the first point until the last one. In the first loop, when an unseen surface point is found, a viewpoint is selected looking at it as the one that sees more unseen surface points. The criteria to choose it follows the idea described by Equation 2.7. After the first loop, all the surface points are seen at least one time. In the second loop, when a surface point seen only one time is found, a viewpoint is selected looking at it as the one that sees more surface points seen only one time. After the second loop, all the surface points are seen at least two times.

In both loops, the potential viewpoints are set inside the SR. In the case that a surface point is unreachable due to the presence of an obstacle, the View-planner will not be able to find a path, due to the fact that it supposes that all the surface points are reachable.

Ideally, in each loop several viewpoints are selected as the ones that see more special surface points (i.e. unseen points or points seen just one time), but because of working

with low resolutions, sometimes, a group of viewpoints see the same amount of special surface points. For this reason, another criteria has been included. In the case that more than one viewpoint sees the same amount of special surface points, the viewpoint which is placed farther away to the previous viewpoint is selected in order to avoid p-outliers (see next chapter).

In the second loop, at times it was found that viewpoints were selected really close to some viewpoints chosen in the first loop, what may produce a low quality 3D reconstruction. To avoid that, after the first loop, the surrounding areas where viewpoints were selected were labelled as prohibited for the second loop.

Finally, the selected viewpoints are arranged because they are disorganized due to the fact that they have been found in two different loops, and the Path-planner needs an organized set of points to link. They are arranged according to the proximity to their previous point, starting with the first shot.

3.4 Path-planner

The task of the Path-planner is linking all the viewpoints. To do that, a path algorithm will be used recursively. The path algorithm will use a graph built from a cost function, which is a representation of the security and the desired image detail in the SG.

3.4.1 Cost function

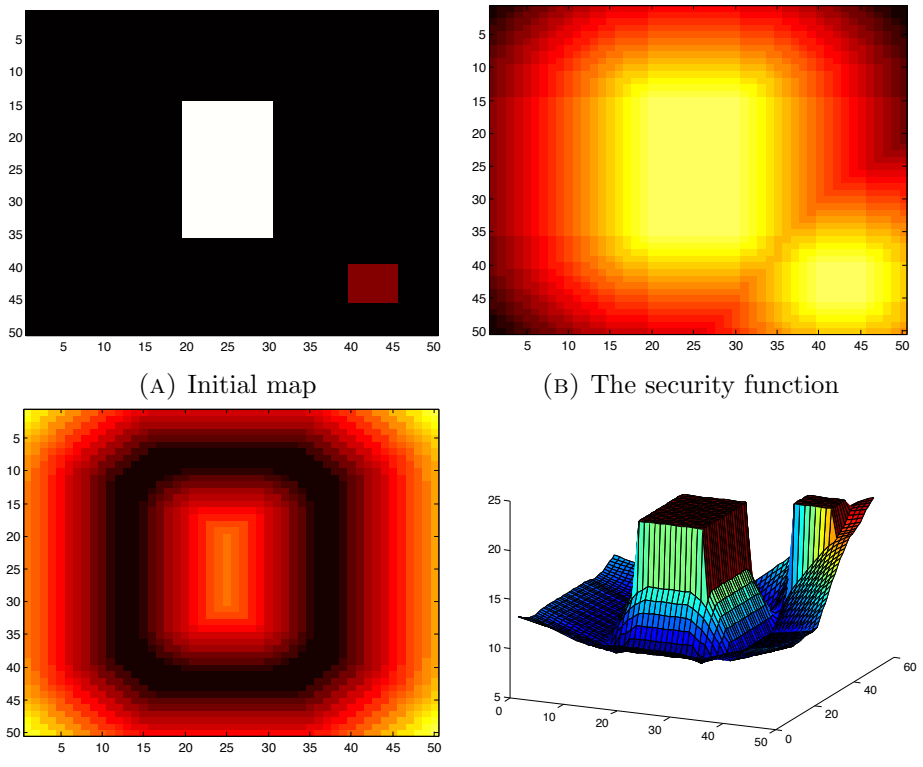
The cost function is a multidimensional array of the same size as the initial SG, in which each element will have a value according to two criteria for describing the space. The first one is that high values represent dangerous area and the second one is that low values represent optimal area to go through. The goal for the Path-planner is to find a path which links the shot points with low values according to the cost function. The area to avoid will be the target building and the obstacles and the optimal area will be zones which are around the optimal distance to take photos. The cost function c, is built from two other functions, a security function s, and an optimal-placement function op:

$$c(x, y, z) = s(x, y, z) + op(x, y, z) \tag{3.2}$$

The security and the optimal-placement functions are multidimensional arrays of the same size of the SG, where each element has a value indicating how close is this element to a building or obstacle in the SG, in the case of the security function; and a value indicading how close is this element to the ideal area to take shots in the case of the optimal-placement function. For that pourpose, as for computing the SR, Euclidean distances have been used, d_E.

$$\begin{aligned} s(x,y,z) &= K - min(d_E(SG_{x,y,z}, TO)) \\ op(x,y,z) &= min(d_E(SG_{x,y,z}, OR)) \end{aligned} \tag{3.3}$$

Where TO are the points that represent the target and the obstacles, OR represents the optimal distance area around the target without taking into account possible obstacles, and K is the maximum distance found in the term $min(d_E(SG_{x,y,z}, TO)$. The usage of K, inverts the shape of the security function, giving high values (K) to the target or obstacles.



(A) Initial map (B) The security function

(C) The optimal-placement function (D) The cost function

FIGURE 3.8: Creation process of a cost function for a 2D scene. In the initial map, the white figure represents a target building and the red one, an obstacle. In the security function, the brighter areas represent not allowed areas (the building and the obstacle). In the Optimal placement function, the darker areas represent ideal areas to take shots.

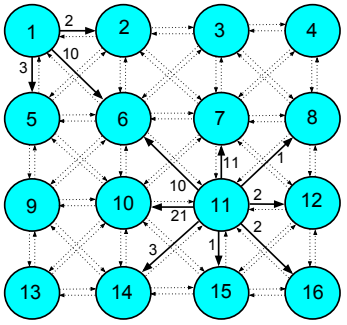
Once both functions are combined, the values of the building, obstacles and regions under a security distance to them, are set to infinite, to ensure that the cost of these regions is really high, so that it is not interesting to go in this areas. The creation process of the cost function for a 2D case is illustrated in Figure 3.8.

3.4.2 Path algorithm

From the cost function, a directed weighted graph is built. In this graph, each node represents a position in the space, in the scene grid. The weights asigned to the edges are values of the cost function.

1	2	3	1
3	10	11	1
1	21	12	2
2	3	1	2

(A) Cost function of size 4×4



(B) Directed graph built from the cost function

Final Node / Initial Node	#1	#2	#3	#4	#5	#6	#7	#8	#9	#10	#11	#12	#13	#14	#15	#16
#1	0	2	0	0	3	10	0	0	0	0	0	0	0	0	0	0
#2	1	0	3	0	3	10	11	0	0	0	0	0	0	0	0	0
#3	0	2	0	1	0	10	11	1	0	0	0	0	0	0	0	0
#4	0	0	3	0	0	0	11	1	0	0	0	0	0	0	0	0
#5	1	2	0	0	0	10	0	0	1	21	0	0	0	0	0	0
#6	1	2	3	0	3	0	11	0	1	21	12	0	0	0	0	0
#7	0	2	3	1	0	10	0	1	0	21	12	2	0	0	0	0
#8	0	0	3	1	0	0	11	0	0	0	12	2	0	0	0	0
#9	0	0	0	0	3	10	0	0	0	21	0	0	2	3	0	0
#10	0	0	0	0	3	10	11	0	1	0	12	0	2	3	1	0
#11	0	0	0	0	0	10	11	1	0	21	0	2	0	3	1	2
#12	0	0	0	0	0	0	11	1	0	0	12	0	0	0	1	2
#13	0	0	0	0	0	0	0	0	1	21	0	0	0	3	0	0
#14	0	0	0	0	0	0	0	0	1	21	12	0	2	0	1	0
#15	0	0	0	0	0	0	0	0	0	21	12	2	0	3	0	2
#16	0	0	0	0	0	0	0	0	0	0	12	2	0	0	1	0

(C) Graph of size 16×16 used by the Dijkstra's algorithm implementation in MATLAB

FIGURE 3.9: Example of the graph creation from a cost function.

As it is explained in chapter 2, a directed edge goes from an initial node to a final one. Those nodes represent an initial position and a final position in the cost function. The value of the end position in the cost function is assigned to the edge weight (see Figure 3.10a).

Finally a path algorithm uses the graph and links the input points (the viewpoints) two on two. The costs are all positive numbers which represent the space mobility, for this reason the Dijkstra's algorithm has been chosen. Its operating system consists in:

1. Assign to every node a tentative distance value: set it to zero for the initial node and to infinity for all other nodes.
2. Mark all nodes unvisited. Set the initial node as current. Creating a set of the unvisited nodes called the unvisited set consisting of all the nodes.
3. For the current node, consider all of its unvisited neighbours and calculate their tentative distances. Compare the newly calculated tentative distance to the current assigned value and assign the smaller one.
4. When all of the neighbours of the current node have been considered, the current node is marked as visited and it is removed from the unvisited set. A visited node will never be checked again.
5. If the destination node has been marked visited (when planning a route between two specific nodes) or if the smallest tentative distance among the nodes in the unvisited set is infinity (when planning a complete traversal; occurs when there is no connection between the initial node and remaining unvisited nodes), then stop. The algorithm has finished.
6. The unvisited node that is marked with the smallest tentative distance is selected, and set it as the new *current node*, then it goes back to step 3.

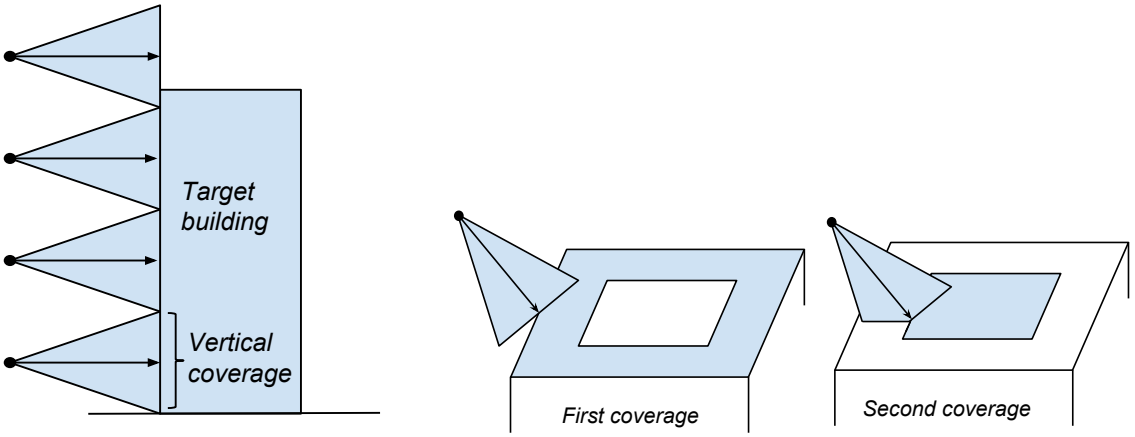
Finally, as it was said before, to create an optimal path, it is important to give to the Path-planner a set of ordered viewpoints, otherwise it would compute a zig-zag path.

3.5 Approach to 3D

Trying to take advantage of the View- and the Path-planners, an approach using them in 3D has been developed. It must be remarked that it is just an rough approach.

3.5.1 Façades

With the internal parameters of the camera and the desired detail, the vertical coverage at the optimal distance can be calculated. Slices separated between this distance are computed until the rooftop is reached. Each 2D SG used in each slice corresponds to a slice of the overall 3D SG. To link the different slices, the movements are completely vertical and once the next level is reached, the closest point in the SR region of this new level is connected. This vertical movements is due to the fact that a graph describing the 3D scene has not been built. It is a line of work that can be conducted in future.



(A) Scheme of façade coverage (B) Example of rooftop covered in two steps.

FIGURE 3.10: Scheme for covering façades and rooftops.

3.5.2 Rooftop, terraces and opposed terraces

For covering these surfaces, a 2D problem is solved iteratively, updating the SG, with a reduced target region, until all rooftop/terrace/opposed terrace points are seen. The target region is reduced proportionally to an assumed distance seen inwards the target. Finally, the resulting paths are fixed at a height related to a fixed pitch angle and the optimal distance.

Chapter 4

Results and Evaluations

The system has been implemented in MATLAB, and thus, the simulations which will be shown are performed in this environment. It is useful for manipulating and representing matrices, which fits well with the volumetric model to represent the space used in this thesis.

In the following simulations, three main criterions are studied, the number of times a surface point is imaged, the distance of the cameras for taking images and finally the angle between image pairs.

1. The first one is the number of times that each surface point of the target building is imaged, *nt-imaged*. Theoretically, each surface point must be imaged at least two times from different perspectives for a proper 3D reconstruction. Imaging a point more than two times is not negative, what is more, this point will be better reconstructed because the proper pair of images for computing triangulation will be able to be chosen.

2. The second is the distance at which each surface point of the target building is imaged, *d-image*. As it has been seen before, this distance is directly related to the image detail and it depends on the application. Generally, a relatively high detail will be desired, since one of the motivations of this thesis is acquiring a highly detailed 3D models. The system proposed gives a path according to a desired image detail, from which an optimal distance can be computed, and it will be desired that the d-image approximates to this distance.

3. The last criterion is the angle between lines of sight of image pairs, the cone angle. When a 3D point location is computed from lines-of-sight forming small angles, the intersection computation of the projection rays is imprecise and the 3D point is likely to be an outlier. In [44], cone angles smaller than a certain angle were discarded. Based in this work, viewpoint pairs with smaller cone angles than 5^{o} will be detected as potentially outliers, *p-outliers*.

On a second level, the total number of shots, the path length and the time of computation will be also studied; and finally, other criterions analysed will be the distance between viewpoints, *db-viewpoints*, the relation of the number shots and the target surface length *p-surface 1*, and the relation of the path length and the target surface length *p-surface 2*.

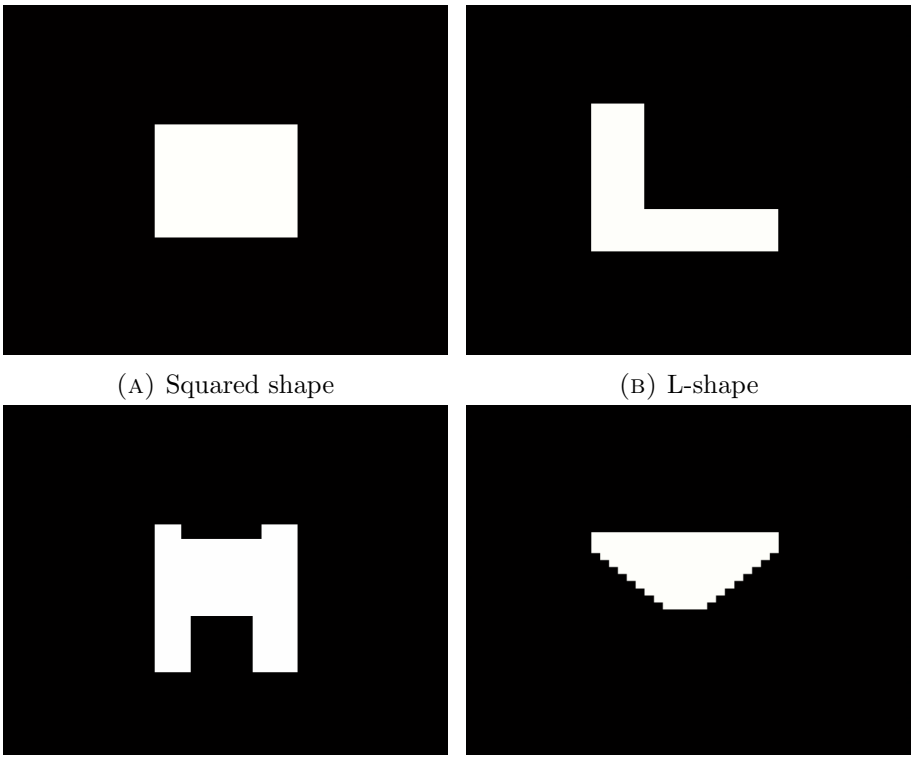
$$\begin{aligned} \text{p-surface 1} &= \frac{\text{number shots}}{\text{surface length}} \\ \text{p-surface 2} &= \frac{\text{path length}}{\text{surface length}} \end{aligned} \tag{4.1}$$

4.1 2D results

In this section, the performance of the system with 2D rough models will be studied, first without the presence of obstacles, and then with it. For that purpose, four 2D SG models with different target shapes (See Figure 4.1) have been created to evaluate the system in different situations. Although, the idea is to use a rough model of the environment and it means that it has low detail, e.g. LOD 1; it is interesting to see the capacity and versatility of the system with models of certain complexity. In fact, a possible future need would be using an automatic view planner several times, updating the current 3D model until reaching a desired accuracy. In this case, the system should be able to adapt to any model shape.

The first 2D SG has a square shape. It is the most simple shape of a building slice. In this first experiment, it is interesting to observe how the system behaves in flat areas and in convex vertices[1]. This SG could correspond to a typical building slice. The second SG is *L*-shaped and is interesting for two reasons. The first one is that it contains a concave

[1] A convex vertex in a polygon is a vertex where the internal angle of the polygon formed by the two edges at the vertex (with the polygon inside the angle), is less than π radians. In other cases the vertex is called concave vertex.



(A) Squared shape (B) L-shape

(C) H-shape (D) Triangulated shape

FIGURE 4.1: Different target building slice shapes which have been evaluated.

vertex, and the second one is that its centroid[2] lies outside the polygon. Earlier strategies for organizing the surface and the shot points were based on the angle to the targets centroid; with that strategy shapes whose centroid lied outside the polygon, such as the L-shape, could not be covered. The third model has a special shape used for seeing the systems behaviour with dead ends, which are regions where the possible viewpoints are limited. This shape also contains convex and concave vertices. Finally, the last SG has a triangular shape. As the grid is visualized as an array of square elements, this model is interesting for seeing the behaviour in oblique surfaces with respect to coordinate axes of the system.

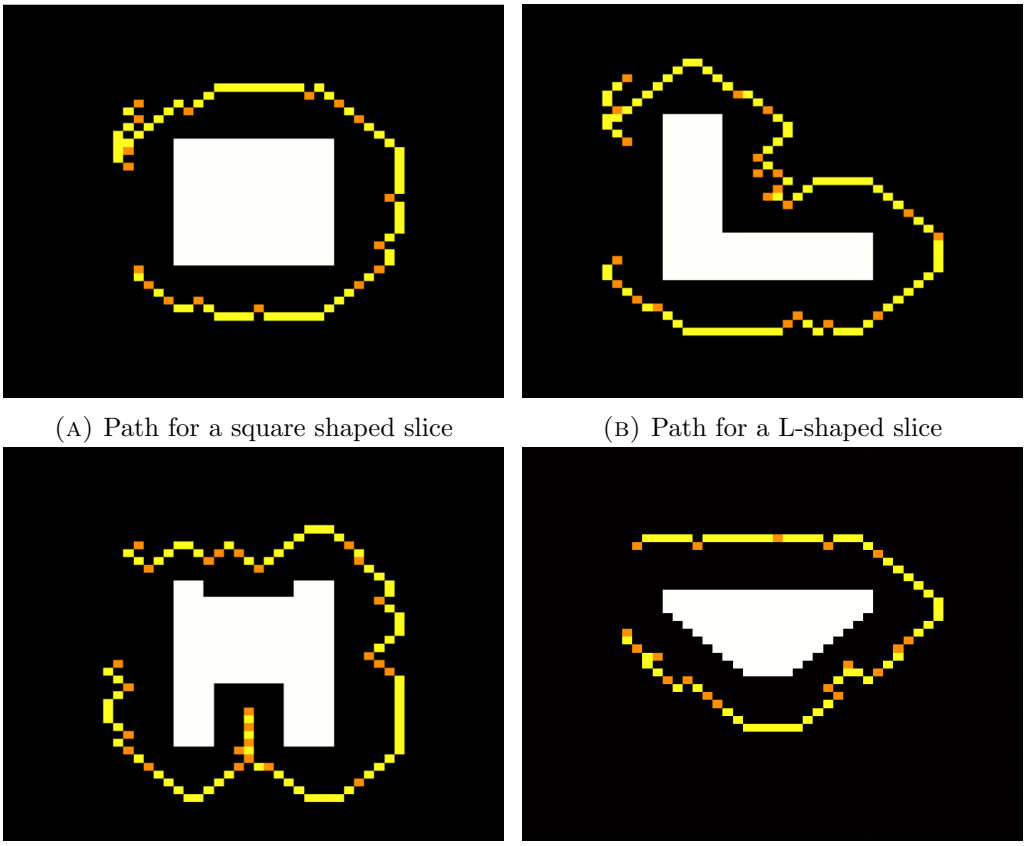
The results are computed from a set of initial parameters which remain the same, without and with obstacles experiments, and for the four SGs, for a better comparison.

- Size of the SG (a square of side M=50 pixels) and initial point.
- Optimal distance = 7 pixels and security distance = 3 pixels.
- Horizontal FOV = 80º.

[2]The centroid of a polygon is the arithmetic mean position of all the points in it.

The initial point may seem like of no importance but from it depends the position of the first shot, and thus, the computation of all the coverage path. It is placed in the upper left corner of the SGs. The next results are related only to the coverage path, which is shown in yellow with the shot points coloured in orange.

4.1.1 2D without obstacles



(A) Path for a square shaped slice

(B) Path for a L-shaped slice

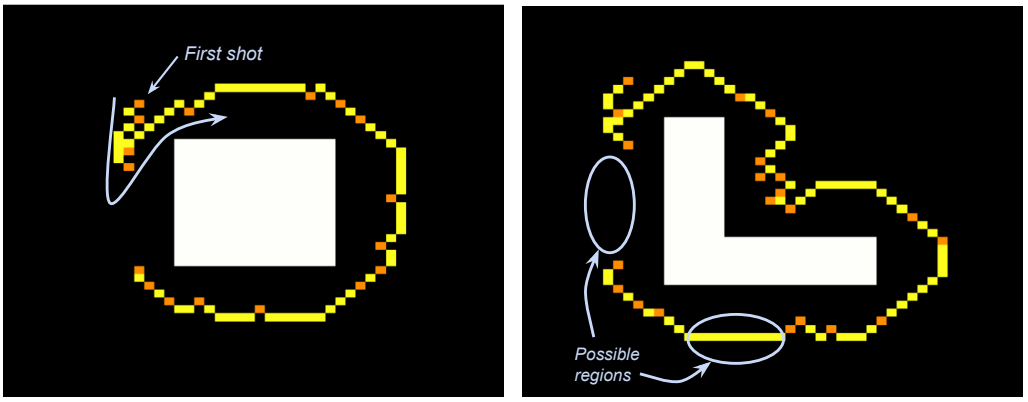
(C) Path for a H-shaped slice

(D) Path for a triangulate shaped slice

FIGURE 4.2: Resulting coverage paths for the four different 2D SGs evaluated.

As it can be seen it Figure 4.2, the paths have approximately a shape that would be drawn by a human designer. However, there are two concrete behaviours that would may have been designed in a different manner. The first one is related to the Path-planner. As described in Figure 4.3a, in some concrete situations the path has a strange design, instead of a single direction path, it has a to and fro design, due to the fact that the shot points are organized according to the proximity to their previous ones. The other special conduct is illustrated in Figure 4.3b, and is related to the View-planner. At times, a set of viewpoints is chosen around a concrete area, and then another regions are empty, it may be more logical to chose viewpoints uniformly. The system chooses

the viewpoints only depending on the amount of information that they can acquire, and usually a point looking in a angled direction to a wall covers more surface.



(A) To and fro behaviour (B) Regions without viewpoints

FIGURE 4.3: Special conducts which may had been manually designed in a different manner.

In the following tables, the information regarding those experiments is given. First, Table 4.1 is given to observe the ranges of the times a surface point is seen, of the image distance, and of the distance between viewpoints.

TABLE 4.1: Ranges of nt-imaged, d-image and db-viewpoints for the 4 experiments without obstacles.

		min	max	mean
PATH OF SG 1	Nt-imaged	2	4	2.5
	D-image (pixels)	4,123	10,295	6,77
	Db-viewpoints (pixels)	2	12,165	5,512
PATH OF SG 2	Nt-imaged	2	4	2,337
	D-image (pixels)	3,605	10,44	6,901
	Db-viewpoints (pixels)	1,414	13,152	4,947
PATH OF SG 3	Nt-imaged	2	4	2,4
	D-image (pixels)	3,605	10,44	6,671
	Db-viewpoints (pixels)	0	14,317	4,206
PATH OF SG 4	Nt-imaged	2	6	2,6
	D-image (pixels)	3,605	10,295	6,591
	Db-viewpoints (pixels)	1,414	12,041	4,398

It is interesting to see that the minimum times a point is seen, is always two times; it is logical according to the system, it checks the surface two times, looking for uncovered points. Also, it is remarking the fact that in the path for the H-shape target, the minimum distance between points is 0, meaning that two points have the same position. After studying the case, it was found that in the below dead end of the SG, two

consecutive points, in the same checking loop[3], were chosen with the same position, but with different orientation. It is logical, because in this region, the possible viewpoints are limited due to the nature of the target shape. Another remarking point of this situation is that the road map result showed only one of those viewpoints, because the Path-planner was built in order to link points with different positions and one of them was erased, causing an error.

Finally, Table 4.2 shows the global evaluation results of the four experiments. In general, all the parameters are satisfactory, except for the p-outliers, which would be desirable to be lower, if possible 0. Specially high is this parameter in the triangulate shape target, due to the surface definition, which is not practical in this situation. It could be said that the best behaviour is found in the L-shape model, where the d-image is the closest one to ideal one while it has the smallest number of p-outliers. Times are quite similar, except for the H-shape, which has the highest one, due to the complexity of the shape; in fact, it has the largest surface length, and more points must been covered.

TABLE 4.2: Complete results of the 4 experiments without obstacles.

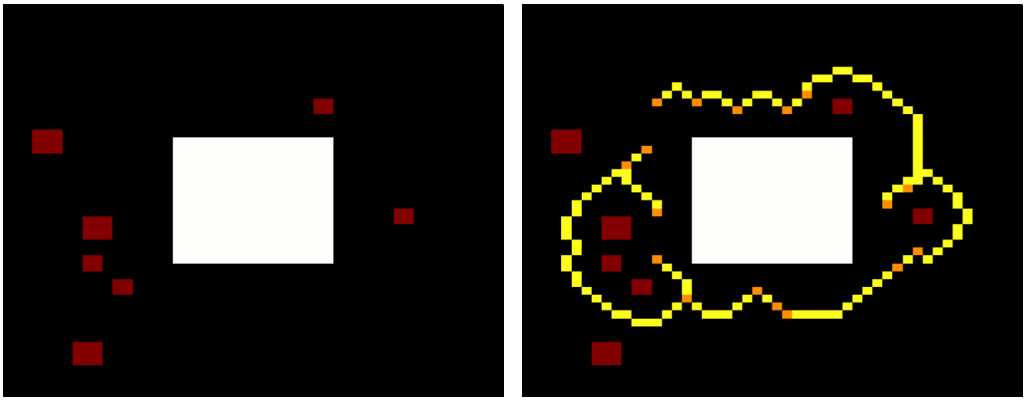
	SG1	SG2	SG3	SG4
Shots	17	21	26	18
Path length (pixels)	79	96	100	71
Time (s)	49,922	65,359	91,302	52,371
(Average) nt-imaged	2,5	2,337	2,4	2,6
(Average) d-image	6,77	6,901	6,671	6,591
(Average) db-viewpoints	5,512	4,947	4,206	4,398
P-outliers	5	3	3	8
P-surface 1	0,283	0,262	0,288	0,3
P-surface 2	1,316	1,2	1,111	1,183

4.1.2 2D with obstacles

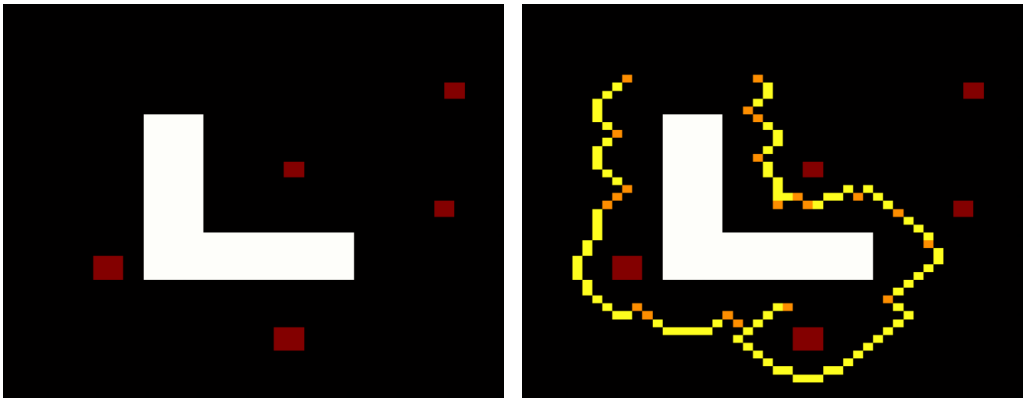
In order to evaluate the systems performance with the presence obstacles, a set of them of different size in random positions have been included in two of the four rough models seen before. For these evaluations the square and the L-shapes have been chosen.

[3]After the first checking loop, the surrounding areas where a viewpoint has been selected are labelled as not possible shot region in the second checking loop, in order to prevent p-outliers. In this case, it was performed in the same loop, and the viewpoint were not p-outliers becuase they had a big cone angle.

Concretely, 3 big (9 pixels), and 5 small (4 pixels), have been placed randomly in SGs with the square and the L-shapes of the same size of the previous experiments (M=50 pixels). With the same initial parameters as the previous experiments, 10 different obstacle configurations for each target shape have been created, producing 10 different paths for each shape.



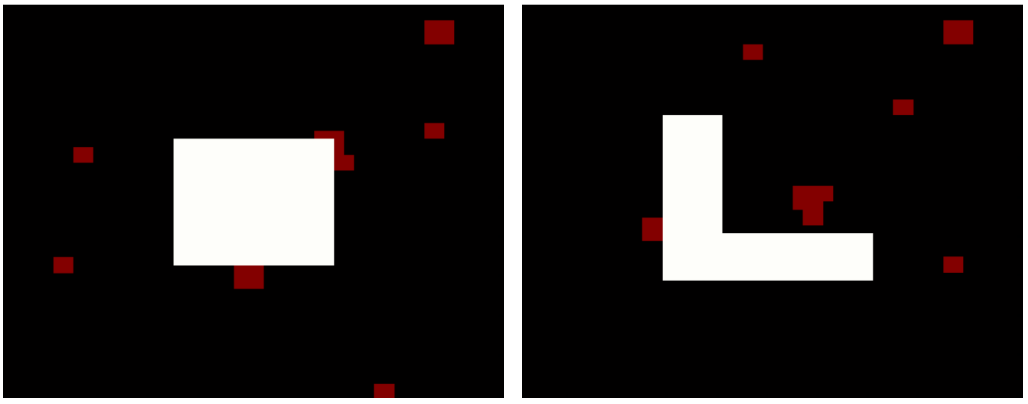
(A) Square shape with obstacles presence (B) Resulting path for the left situation



(C) L-shape with obstacles presence (D) Resulting path for the left situation

FIGURE 4.4: Two results of each collection of experiments.

If the obstacles are placed in areas where they do not cover completely a part of the target building, the system is able to find a path (see Figure 4.4). However, if an obstacle or some obstacles together cover completely an area of the target building (see Figure 4.5), the system can not give a result, because the system checks each surface point and tries to find a viewpoint that covers it. It could be understood as if an obstacle covered completely a part of a building; in this case, this edifice could not be physically covered. The system has been designed assuming that the target building is possible to be covered. One line of future research consists in merging obstacles to the target, if they are too close to it, so as the system can give at least a path, a result (see Appendix C).



(A) Square shaped target with obstacles (B) L-shaped target with obstacles

FIGURE 4.5: Two situations, where the system could not give a result.

Table 4.3 shows the mean and the standard deviation of the parameters described previously for each experiment collection. Those results have been computed from 10 SGs, where the system could find a path, situations where an obstacle covered completly a surface part where discarded.

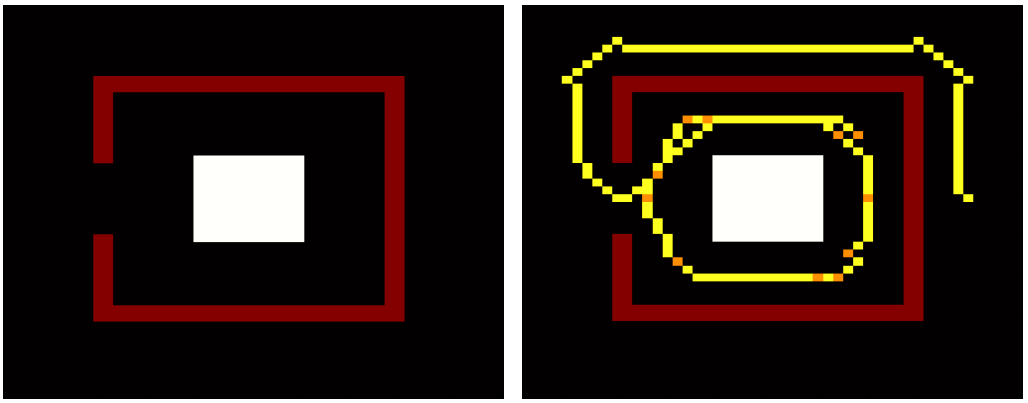
TABLE 4.3: Results of the two collection of experiments. Collection 1 corresponds to the square shape, and collection 2 to the L-shape.

	Experiment collection 1		Experiment collection 2	
	Mean	Std. deviation	Mean	Std. deviation
Shots	18,1	1,044	22,2	1,166
Path length (pixels)	97,4	16,499	110,6	20,367
Time (s)	48,808	2,345	78,483	8,359
Nt-imaged	2,466	0,055	2,463	0,081
D-image (pixels)	6,818	0,129	6,856	0,069
Db-viewpoints (pixels)	5,219	0,393	4,886	0,23
P-outliers	2,6	1,280	3,5	1,565
P-surface 1	0,301	0,017	0,277	0,014
P-surface 2	1,623	0,275	1,382	0,254

These results are analogous to the case without obstacles. The nt-imaged and the d-image are quite satisfactory, but again, the p-outliers is high, and with an elevated deviation compared with its mean, which is a drawback. Also interesting is the time result, it does not differ a lot from the experiments without obstacles. The reason is that the system computes the same amount of operations for finding the viewpoints, but in a more restricted SR limited by the obstacles.

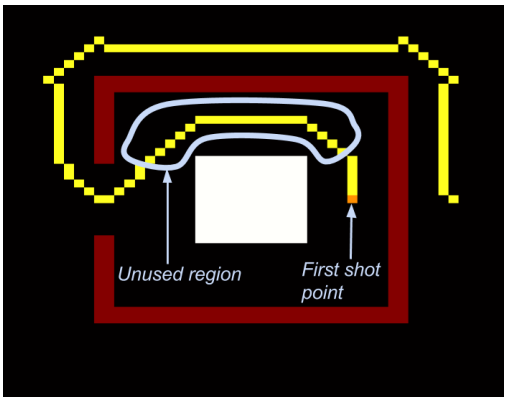
4.1.3 Other 2D results

In order to test the Path-planner, a special situation has been designed, where a target building is completely surrounded, except for a small entrance, by an obstacle. The simulation was performed with an square shape and with the same input values given in experiments described above, but with a different initial position. In purpose, it was given in the opposite site of the entrance to see the behaviour of the Path-planner.

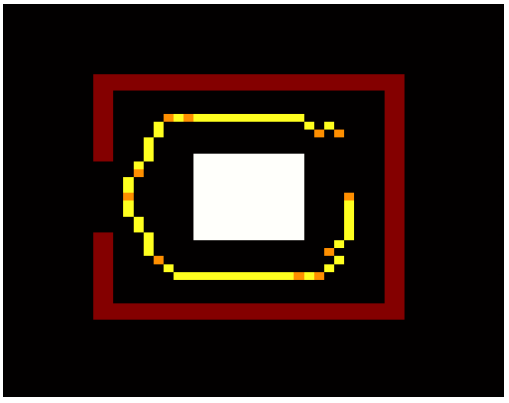


(A) Target surrounded by an obstacle

(B) Total path for covering the left SG



(C) Initial path



(D) Coverage path

FIGURE 4.6: Target building surrounded by an obstacle, the total path to cover it, and the initial and coverage paths.

In Figure 4.6c, it can be seen that in order to reach the first shot point, which is the closest point in the shot region to the initial point, a path is designed with the particularity that for a while it goes alongside the target and some pictures could have been taken. When the first shot point is reached, the coverage path is computed (Figure 4.6d). After that a final path is computed, which goes back to the initial position. If they are treated independently, they can be reasonable paths, but treated together are not efficient because at the end, two rounds are done around the target. This is an issue that must be addressed by modifying the Path-planner.

Finally, the system has been tested with different SG resolutions to see the orders of magnitude of the computation times[4].

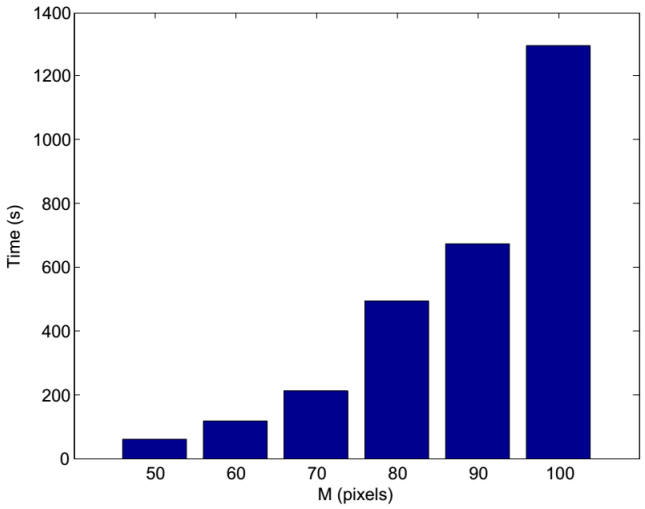


FIGURE 4.7: Time increment, with respect to the SG size increment.

The time increasing has a quadratic form due to the fact that the system must compute a number of operations increased quadratically; in fact, the size of the SG is $M \times M$. Each simulation performed previously in this section has been realised with $M = 50$ pixels, and the order of magnitude for this resolution is around 100 seconds, approximately 2 minutes, which can be considered a good performance. On the other hand, for $M = 100$ pixels, the order of magnitude is around 1.300 seconds, approximately 20 minutes, which is a time that can be considered significant.

4.2 3D results

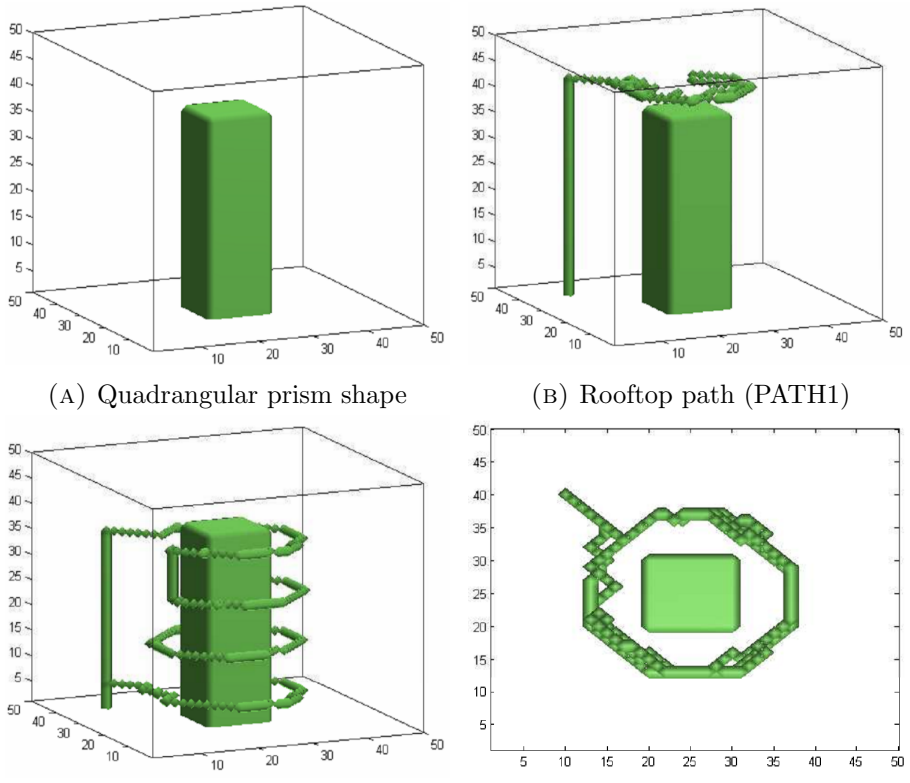
Although most of the work in this thesis has been centred in 2D problems, the final goal was to develop an approach for 3D. In this section, some 3D results of the system are presented visually, for three different building shapes. In the following images, the path is described as a line of points around a main structure which represents a building. Due to representation problems the path and the building could not be coloured in a different manner, which would have been desirable. The images correspond to paths which include an initial, a coverage and a final path to illustrate the movements that

[4]The times presented in this section have been computed with an HP Pavilion g6 Notebook PC, with a processor Intel(R) Core(TM) i5-2430M CPU @ 2,4 GHz, and a memory (RAM) of 4 GB.

a UAV should realize. On the other hand, the corresponding tables are related only to the coverage paths. As in the 2D results, all of these evaluations are performed with the same initial parameters:

- Size of the SG (a cube of side M=50 pixels) and initial point.
- Optimal distance = 7 pixels and security distance = 3 pixels.
- Horizontal FOV = 80º and vertical coverage = 10 pixels.
- Fixed pith angle to one of three possibilities (0º, ± 45º).

The first experiment was realized with a quadrangular prism shaped building. Two paths were computed, one for the rooftop, and another for the façades. It must be said that a slice of this building is a square a bit smaller than the square studied in the 2D section and as a consequence the number of shots or the path length can not be compared.



(A) Quadrangular prism shape (B) Rooftop path (PATH1)

(C) Façade path (PATH2) (D) Façade path seen from above

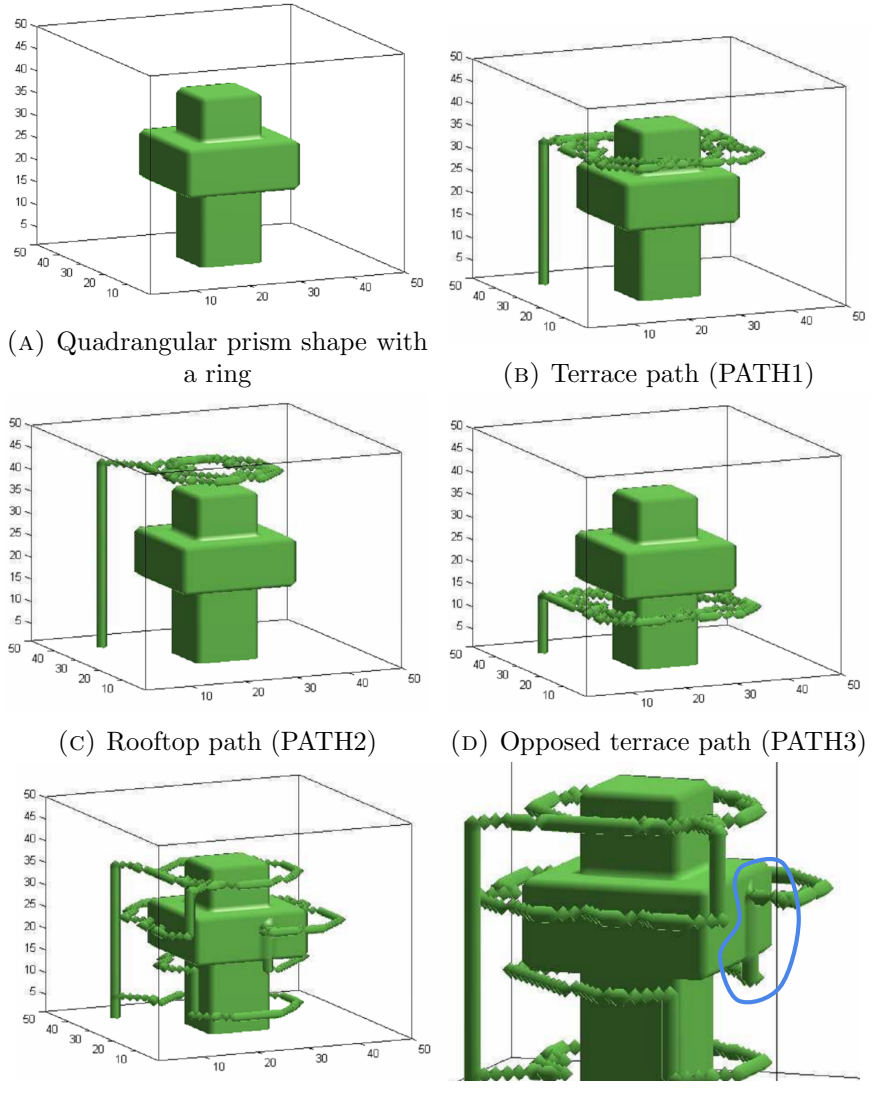
FIGURE 4.8: Resulting paths for a building with a quadrangular prism shape.

The path which covers the façades was built from 4 slices. Information about the two paths can be seen in Table 4.4.

TABLE 4.4: Information of the system result with a quadrangular prism shaped target.

	PATH1	PATH2
Shots	22	45
Path length (pixels)	108	267
Time (s)	50,492	125,537

The second 3D SG tested was the same as the previous one, but with an added ring at half height, creating a terrace and an opposed terrace.



(A) Quadrangular prism shape with a ring

(B) Terrace path (PATH1)

(C) Rooftop path (PATH2)

(D) Opposed terrace path (PATH3)

(E) Façade path (PATH4)

(F) Collision between slices

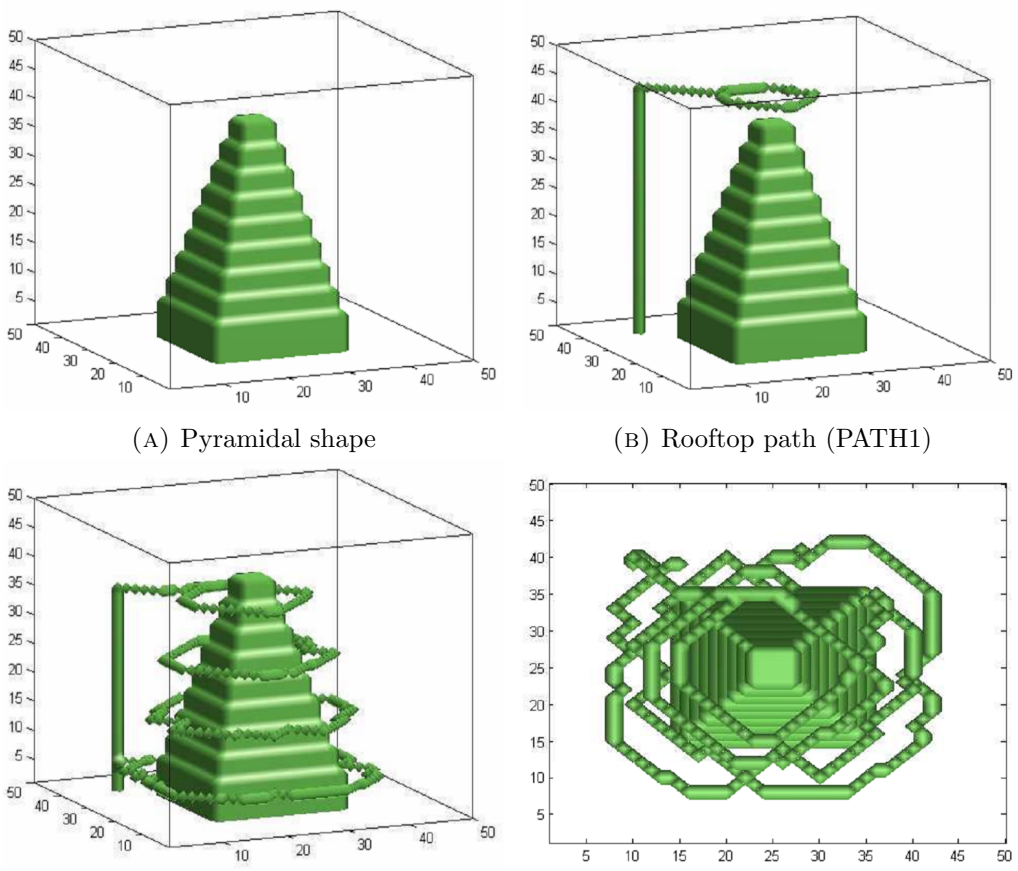
FIGURE 4.9: Resulting paths for a building with terraces and opposed terraces.

In this case, four different paths were computed: one for the terrace, one for the opposed terrace, one for the rooftop and one for the façades. Information about them can be seen in Table 4.5.

TABLE 4.5: Information of the system result with a building with terraces and opposed terraces.

	PATH1	PATH2	PATH3	PATH4
Shots	51	21	51	57
Path length (pixels)	175	94	178	312
Time (s)	126,493	49,243	152,091	192,571

The third 3D SG has a target with a pyramidal shape. Terraces were not detected because the shape changed gradually. As a consequence only two paths were computed, one for the rooftop and one for the façades.



(A) Pyramidal shape (B) Rooftop path (PATH1)

(C) Façade path (PATH2) (D) Façade path seen from above

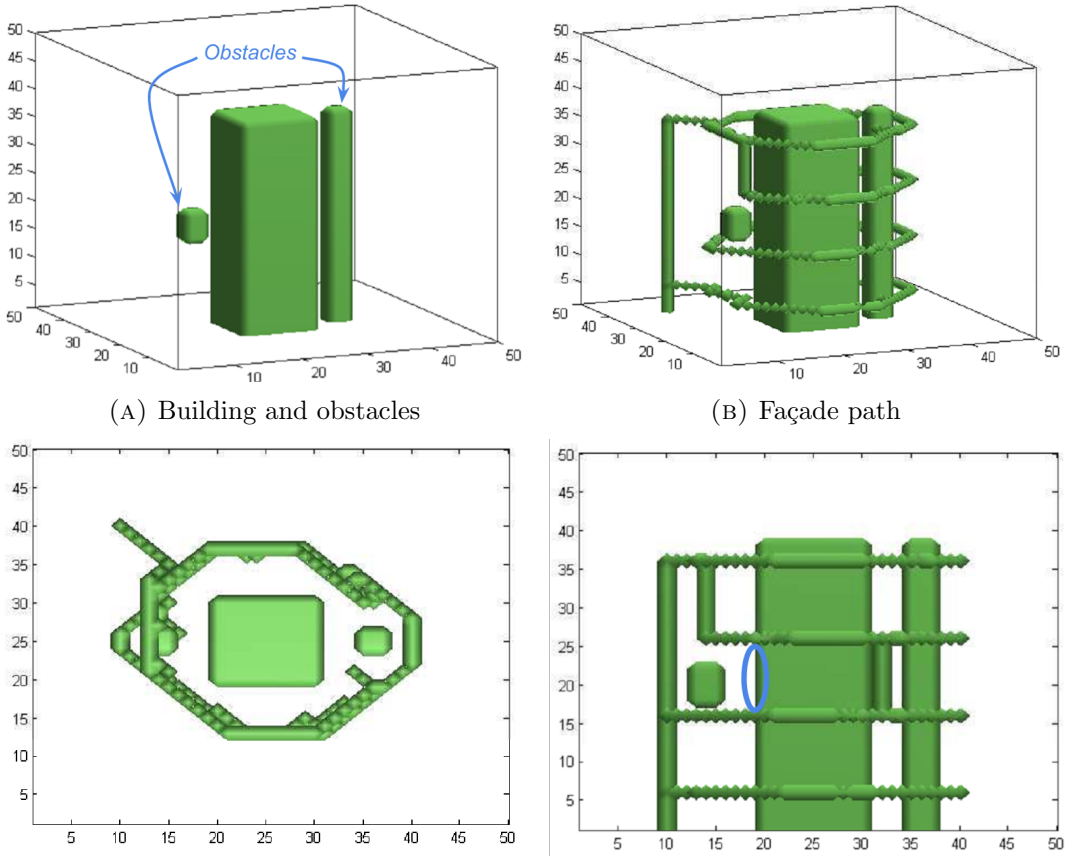
FIGURE 4.10: Resulting paths for a building with a pyramidal shape

As in the quadrangular prism shape, the façade path was also built from 4 slices. Information about the two paths can be seen in Table 4.6.

Finally the system has been tested with a 3D scene with obstacles presence. In figure 4.11, the SG studied is shown, as well as the façade path.

TABLE 4.6: Information of the system result with a pyramidal shaped target.

	PATH1	PATH2
Shots	8	53
Path length (pixels)	27	307
Time (s)	16,112	158,246



(A) Building and obstacles

(B) Façade path

(C) Façade path seen from above

(D) Façade path seen from the side

FIGURE 4.11: A 3D SG with obstacle presence.

As it can be seen visually, the large obstacle is avoided because it remains constant in the vertical axis. However with the small obstacle a problem is presented. It is not a problem of collision, but a coverage one. As it lays in a level between two slices, for them it is not treated. As a result, a building surface area will remain uncovered which is an obvious problem (see circled area in Figure 4.11d).

A last comment is addressed to the time results. The computation time of the façades is directly related to the number of computed 2D slices. In the presented results, only 4 slices are computed because of an assumed vertical coverage of 10 pixels, which in fact, is not realistic, compared to the optimal distance (optimal distance = 7 pixels).

Chapter 5

Conclusions

5.1 Summary

This thesis has proposed an automatic view planner for 3D reconstruction of buildings dividing them into slices and for each slice, solve a 2D coverage problem. Most of the effort has been focused in solving 2D problems with building slices of any shape. The 2D results are promising, but much more effort must be focused in the 3D direction. As a summary, the strong points presented in this thesis are:

- The creation of a system that covers completely a 2D target shape (i.e. a building slice), and with the designed path, a 3D reconstruction from the building area corresponding to it will be able to be performed. The system avoids obstacles and covers a target 2D shape as long as it is physically coverable, taking into account the image detail. An approach for 3D scenes is given using several 2D coverage problems.
- The design of a useful tool, the cost function used by the system, which describes the security and the desired image detail together. It has been developed in 2D and in 3D satisfactorily.
- An implementation of the system in MATLAB, with a GUI, where a user can choose a 2D or 3D scene model, a desired detail and a digital camera, and request for a path that covers the target building of the model. The results are shown graphically and in a road map.

5.2 Further work

This thesis has given a first approach to solve a complex problem with many variables and restrictions. However, it is still not completely solved and future work can be addressed in several directions.

The first direction could be improving the 2D performance. As it has been seen, the 2D results are satisfactory, but against two concrete situations the system has a deficient behaviour. If an obstacle occludes completely a certain part of the target building, so that this part of the surface can not be seen by any viewpoint, the system can not provide a solution due to the fact that it supposes that all the surface is reachable. Some first steps has been addressed in this direction (see Appendix C). The other situation is the shot point linkage, which sometimes leads to a not optimal path shape. The Path-planner links one point to another, without considering the rest. One possibility for improving this problem is taking into account the complete group of shot points when linking them, and not only two at each step. Besides improving the behaviour in those situations, effort must be addressed in reducing the number of p-outliers, because they will probably lead to an inaccurate 3D reconstruction.

On the other hand, a view planning problem for a building reconstruction is obviously a 3D problem. The most amount of work must aim to develop a satisfactory 3D solution. The strategy of dividing a building into slices has sense, but instead of solving a 2D problem for each slice, it could be interesting to confront the problem differently.

One of the first steps could be taking into account two orientation degrees of freedom. With this configuration, a single path able to cover the different kind of surfaces seen in this thesis could be designed, and thus, the UAV could be sended just once. This would have a direct reduction in cost and time of the enterprise.

Developing a graph that describes all 3D space is another key point to link the viewpoints and to give a path in a 3D model. It would give more movement freedom, and obstacles in 3D space could be avoided. This graph would also avoid the collision between slices of Figure 4.9f. To do that, another strategy for creating the graph must be followed.

Further, it will be also interesting to work with SGs with high resolution, so that the results are more realistic. One interesting tool could be using *kd-trees*, which is a space-partitioning data structure for organizing points in a k-dimensional space. It is useful

for several applications, such as searches involving a multidimensional search key, e.g., nearest neighbour searches.

The final step, after having improved properly the automatic view planner, could be acquiring a real building coordinates, create a rough model of it and give this model to the system. After that, transforming the generated path into a navigation system coordinates, e.g. GPS coordinates, so that a real UAV can follow the generated path.

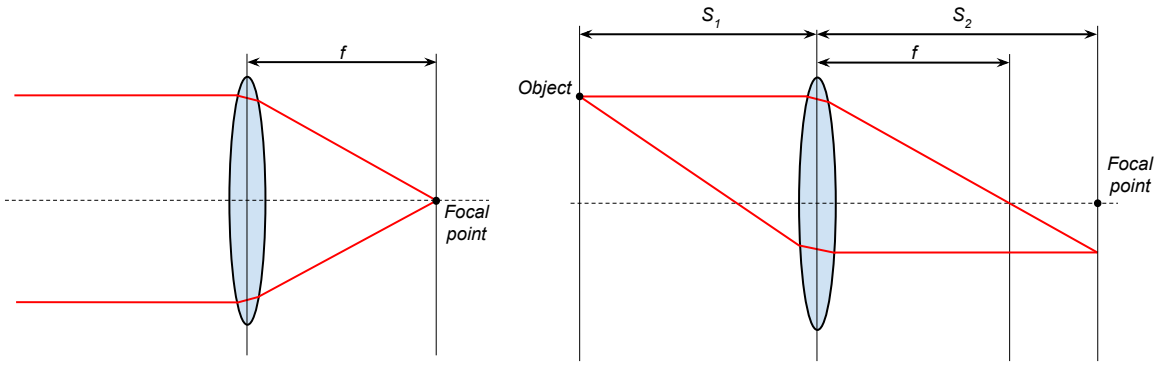
Appendix A

Focal Length in Digital Cameras

A digital camera is an optical device which creates a single image of an object or scene, and records it on a digital image sensor. All cameras use the same basic design: light enters an enclosed box through a convex lens and an image is recorded on a light-sensitive medium. While in principle in a digital camera a simple convex lens will suffice, in practice a compound lens made up of a number of optical lens elements is required to correct the many optical aberrations that arise. This Appendix simplifies the operation of a camera, to the use of a single convex lens.

A convex lens makes light rays passing through it bend inward and meet at a spot just beyond the lens known as the focal point. The image sensor of a camera is situated in the focal plane, a plane perpendicular to the optical axis, which passes through the focal point. In order for a lens to focus rays of light that are not parallel, the lens needs to focus the light. This act of focusing moves the focal point. The image sensor, however, is stationary and does not move, in fact are the lens elements that move away from the image sensor as the lens is focused.

In Figure A.1a, a convex lens focuses a beam of collimated light into the focal point. Collimated light is light whose rays are parallel, which come from infinite. In the case that collimated light comes to a convex lens, the focal length f, is defined as the distance between the optical center of the lens and the focal point. On the other hand, if not collimated light comes to a convex lens, e.g. from an object situated at distance S_1 to the optical center, the focal point will be then situated at a distance S_2 to the optical center. This distance is different to the focal length (see Figure A.1b).



(A) (B)

FIGURE A.1: Focal length in a situation with collimated light (left), and focal length in a situation with light coming from an object at position S_1 (right).

As a conclusion, for the interest of this thesis, when a digital camera is set to infinity, its optical center is separated from the image sensor, by the lens's focal length.

Appendix B

Epipolar Geometry

Epipolar geometry is the geometry of stereo vision. When two cameras view a 3D scene from two distinct positions, there are a number of geometric relations between the 3D points and their projections onto the 2D images that lead to constraints between the image points. These relations are derived under the assumption that the cameras behave like in the pinhole camera model.

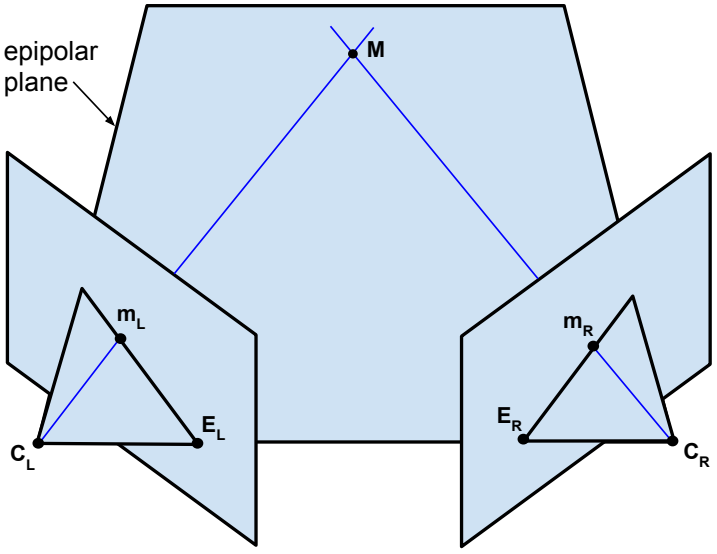


FIGURE B.1: Two cameras are indicated by their centres C_L and C_R and image planes. The camera centres, a 3D point M, and its images m_L and m_R lie in a common plane. A line connecting each camera center, intersects each image plane at the epipoles E_L and E_R. A plane containing this line is an epipolar plane.

If the relative translation and rotation of the two cameras is known, the corresponding epipolar geometry leads to two important observations:

1. If the projection point m_L is known, then the epipolar line E_R–m_R is known and the point M projects into the right image, on a point m_R which must lie on this particular epipolar line. This means that for each point observed in one image, the same point must be observed in the other image on a known epipolar line. This provides an *epipolar constraint* which corresponding image points must satisfy and it means that it is possible to test if two points really correspond to the same 3D point.

2. If the points m_L and m_R are known, their projection lines are also known. If the two image points correspond to the same 3D point M, the projection lines must intersect precisely at M. Triangulation is the process of calculating a point M, from the coordinates of its two image points.

Epipolar constraints are usually described by the fundamental matrix, if the internal camera parameters are unknown:

$$m_2^T F m_1 = 0 \tag{B.1}$$

Or, by the essential matrix, if the internal parameters are known:

$$m_2^T E m_1 = 0 \tag{B.2}$$

Those are 3×3 matrices, which are basically the algebraic representation of the epipolar constraint.

Appendix C

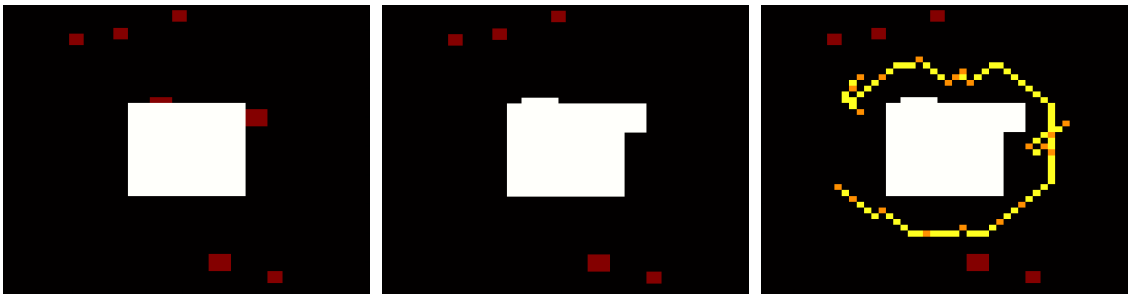
Obstacle Merging

The situation where an obstacle or multiple obstacles cover completely a part of the target building is of special interest. The system proposed assumes that the target building surface is able to be covered. Otherwise, it can not give a path. This can be seen as a drawback, since in the real world any building is surrounded by obstacles such as trees, other buildings, lampposts, etc. Two solutions have been thought to solve this problem:

1. Ignoring the unreachable surface points, and cover the rest of the surface points.
2. Merging the obstacles which are too close to the target into it and solve a problem with this updated target shape.

Initial efforts have been focused in the second direction, obtaining some interesting results. The strategy consists in checking neighbouring points to the target and examine if they belong to an obstacle or not. If they do, they are merged to the target, and the neighbouring points of the new target shape are checked again, until the model is not updated any more. Figure C.1 shows a result of this strategy. Given a scene with obstacles placed too close to the target building, after that they are merged, and finally a path is computed from this updated model. This situation could not have been solved by the systems behaviour by default.

This performance is still not refined. It only merges obstacles which are up against the target or which are at a certain distance. It would be interesting to merge obstacles at



(A) Original SG (B) Updated SG (C) Final coverage path

FIGURE C.1: Situation with two obstacles near a the target building. First, they are merged and as a result a path can be computed.

different selectable distances, and obstacles which cover completely a target surface part but maybe are not to close.

As a consequence, this merging process can lead to a sharped target, which would produce problems in the surface organization step.

Appendix D

Data CD

Contents:

1. Thesis in PDF format.
2. MATLAB folder containing all the necessary functions to run the implementation of the system.
3. File readme.txt. Instructions and indications of how to use the system.

Bibliography

[1] Dresden Frauenkirche. URL `http://www.frauenkirche-dresden.de/wiederaufbau.html`. Last visited: 11/06/2014.

[2] W. Jizhou, L. Zongjian, and L. Chengming. Reconstruction of buildings from a single uav image. In *International Society for Photogrammetry and Remote Sensing Congress in Istanbul, Turkey*, volume XXXV, pages 100–103, 2004.

[3] H. Mayer and J. Bartelsen. Automated 3d reconstruction of urban areas from networks of wide-baseline image sequences. In *International Society for Photogrammetry and Remote Sensing Congress*, volume XXXVII, pages 633–638, 2008.

[4] A. Irschara, V. Kaufmann, M. Klopschitz, H. Bischof, and F. Leberl. Towards fully automatic photogrammetric reconstruction using digital images taken from uavs. In *Proceedings of the International Society for Photogrammetry and Remote Sensing Symposium*, 2010.

[5] H. Puschel, M. Sauerbier, and H. Eisenbeiss. A 3d model of castle landenberg (ch) from combined photogrammetric processing of terrestrial and uav-based images. In *The international archives of the photogrammetry, remote sensing and spatial information sciences*, volume 37, pages 93–98, 2008.

[6] C. Wefelscheid, R. Hänsch, and O. Hellwich. Three-dimensional building reconstruction using images obtained by unmanned aerial vehicles. *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, 38:1, 2011.

[7] D. Rother and G. Sapiro. Seeing 3D objects in a single 2D image. In *IEEE 12th International Conference on Computer Vision*, pages 1819–1826, 2009.

[8] A. Varol, A. Shaji, M. Salzmann, and P. Fua. Monocular 3D reconstruction of locally textured surfaces. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(6):1118–1130, 2012.

[9] CityGML. Common information model and XML-based encoding for the representation, storage, and exchange of virtual 3D city and landscape models. URL `http://www.citygml.org/`. Last visited: 01/06/2014.

[10] T. Moons, L.V. Gool, and M. Vergauwen. 3D reconstruction from multiple images part 1: Principles. *Foundations and Trends in Computer Graphics and Vision*, 4 (4):287–404, 2008. ISSN 1572-2759.

[11] S.N. Sinha, D. Steedly, R. Szeliski, M. Agrawala, and M. Pollefeys. Interactive 3d architectural modeling from unordered photo collections. *ACM Transactions on Graphics*, 27(5):1–10, 2008.

[12] N. Snavely, S. M. Seitz, and R. Szeliski. Photo tourism: Exploring photo collections in 3d. *ACM Transactions on Graphics*, 25(3):835–846, July 2006. ISSN 0730-0301.

[13] UVS International. Non profit organization representing manufacturers of unmanned vehicle systems (UVS), subsystems and critical components for UVS and associated equipment, . URL `http://uvs-international.org/`. Last visited: 04/05/2014.

[14] ATEA Data Ltd. Aerial photography company., . URL `http://www.ateadata.co.nz/home/`. Last visited: 04/05/2014.

[15] C.V. Angelino, V.R. Baraniello, and L. Cicala. UAV position and attitude estimation using IMU, GNSS and camera. In *15th International Conference on Information Fusion*, pages 735–742, 2012.

[16] Department of Transportation. United Stated of America. *Global Positioning system (GPS) Civil Monitoring Performance Specification*, 2 edition, 2009.

[17] K. Tarabanis, P.K. Allen, and R.Y. Tsai. A survey of sensor planning in computer vision. *IEEE Transactions on Robotics and Automation*, 11(1):86–104, 1995.

[18] L.M. Wong, C. Dumont, and M.A. Abidi. Next best view system in a 3D object modeling task. In *IEEE International Symposium on Computational Intelligence in Robotics and Automation*, pages 306–311, 1999.

[19] C.I. Connolly. The determination of next best views. In *IEEE International Conference on Robotics and Automation*, volume 2, pages 432–435, 1985.

[20] H. Zha, K. Morooka, T. Hasegawa, and T. Nagata. Active modeling of 3-D objects: planning on the next best pose (nbp) for acquiring range images. In *International Conference on Recent Advances in 3-D Digital Imaging and Modeling*, pages 68–75, 1997.

[21] J.E. Banta, L.M. Wong, C. Dumont, and M.A. Abidi. A next-best-view system for autonomous 3-D object reconstruction. *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans*, 30(5):589–598, 2000.

[22] J. Maver and R. Bajcsy. Occlusions as a guide for planning the next view. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(5):417–433, 1993.

[23] K.N. Kutulakos and C.R. Dyer. Global surface reconstruction by purposive control of observer motion. 1993.

[24] B. Abidi. Automatic sensor placement. In *Conference on Intelligent Robots and Computer Vision XIV*, volume 2588, pages 387–398, 1995.

[25] R. Pito. A solution to the next best view problem for automated surface acquisition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(10):1016–1030, 1999.

[26] J. O'Rourke. *Art Gallery Theorems and Algorithms*. The International Series of Monographs on Computer Science. Oxford University Press, New York, NY, 1987.

[27] T.C. Shermer. Recent results in art galleries. *Proceedings of the IEEE*, 80(9): 1384–1399, 1992.

[28] Xuehou Tan. Approximation algorithms for the watchman route and zookeeper's problems. In *COCOON*, pages 201–206, 2001.

[29] J. Faigl. Approximate solution of the multiple watchman routes problem with restricted visibility range. *IEEE Transactions on Neural Networks*, 21(10):1668–1679, 2010.

[30] Richard J. Trudeau. *Introduction to Graph Theory (Dover Books on Mathematics)*. Dover Publications, 1994.

[31] P. Fletcher, H. Hoyle, and C.W. Patty. *Foundations of Discrete Mathematics*. Pws Pub Co, 1990. ISBN 0534923739.

[32] S.M. LaValle. *Planning algorithms*. Cambridge University Press, 2006. ISBN 978-0-521-86205-9.

[33] B.V. Cherkassky, A.V. Goldberg, and T. Radzik. Shortest paths algorithms: Theory and experimental evaluation. *Mathematical Progamming*, 73:129–174, 1993.

[34] T.H. Cormen, C. Stein, R.L. Rivest, and C.E Leiserson. *Introduction to Algorithms*. McGraw-Hill Higher Education, 2nd edition, 2001. ISBN 0070131511.

[35] Y. Koren and J. Borenstein. Potential field methods and their inherent limitations for mobile robot navigation. *IEEE International Conference on Robotics and Automation*, pages 1398–1404, 1991.

[36] L.M. Wong, C. Dumonty, and M.A. Abidi. An algorithm and system for finding the next best view in a 3-d object modeling task. Master's thesis, The University of Tennessee, Knoxville, TN, 1999.

[37] D. Rother and G. Sapiro. 3d reconstruction from a single image. 2008.

[38] N.A. Massios and R.B. Fisher. A best next view selection algorithm incorporating a quality criterion. In *Proceedings of the British Machine Vision Conference*, 1998.

[39] C.K. Cowan and P.D Kovesi. Automatic sensor placement from vision task requirements. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 10(3): 407–416, 1988.

[40] C.K. Cowan and A. Bergman. Determining the camera and light source location for a visual task. In *IEEE International Conference on Robotics and Automation*, pages 509–514, 1989.

[41] K. Tarabanis, R.Y. Tsai, and P.K. Allen. Automated sensor planning for robotic vision tasks. In *IEEE International Conference on Robotics and Automation*, pages 76–82, 1991.

[42] I. Agricola and T. Friedrich. *Elementary Geometry*, volume 43 of *Student Mathematical Library*. Publications of the AMS, 2008.

[43] MATLAB. *version 7.10.0.499 (R2010a).* The MathWorks Inc., Natick, Massachusetts, 2010.

[44] N. Salman and M. Yvinec. Surface reconstruction from multi-view stereo of large-scale outdoor scenes. In *The International Journal of Virtual Reality*, volume 9, pages 10–26, 2010.

[45] Dijkstra's algorithm implementation in MATLAB. URL `http://www.mathworks.com/matlabcentral/fileexchange/36140-dijkstra-algorithm`. Last visited: 25/06/2014.