Bachelor's Thesis
Bachelor Degree in Informatics Engineering
Computation Specialization

360^{o} Indoors Image Processing for 3D Model Reconstruction

Sikander Ali

Director: **Javier Biosca**
Codirector: **Rodrigo Ignacio Silveira**
GEP Tutor: **Marcos Eguiguren**

FIB

UNIVERSITAT POLITÈCNICA
DE CATALUNYA
BARCELONATECH

Acknowledgements

This project is a product of great effort and support from many people. I am grateful to my project management tutor, Marcos Eguiguren, for guiding me how to properly plan the management of this project. I am grateful to my project tutor, Rodrigo Ignacio Silveira, for clarifying many doubts I had relating to academic requirements of the project. I am grateful to my project supervisor, Javier Biosca, for guiding me progress through every small step of the project, from management to research, from prototyping to development.

I am utmost grateful to my family and friends for giving me motivation throughout this project and throughout all these years. I am grateful to my friends for making this competitive yet enjoyable experience.

And I can not express enough with words, how grateful I am to my parents and siblings, who made every moment of my journey possible with their unimaginable support.

I am truly very thankful to all these people.

Abstract

In this modern age of computer technology we are pushing the unimaginable limits of our reality. One of the human desires with these advances is to digitise the vast amount of information that is present in our reality. An important source of information is the 3-dimensional space in which we live. Especially indoors environments that we frequently occupy, for example, living places. With the proliferation of photographing devices, the development of cheap omnidirectional cameras has been one of the interests. So nowadays it is quite easy to obtain spatial data of interior spaces in form of equirectangular images.

In this project we study the problem of 3D Indoors Model Reconstruction from Spherical Images. Though, we study it under perspective based methods as it is possible to perform the conversion from one to other. We first formally specify the problem to be solved. We find many different specifications and describe reconstruction methods for some of them. We choose one specification for our use case. Most of the methods require feature extraction and matching, and then performing multi-view geometry estimation. We continue the study of these methods in the experimentation phase. We propose different hypothesis relevant to different steps, perform experiments and form our conclusions. We finish our work by implementing a very simple system solving this problem, making use of ASIFT feature extractor, FLANN kD-Tree feature matcher, and OpenCV's essential matrix estimation algorithm.

Keywords – 3D reconstruction, indoors image processing, equirectangular image processing, 360 indoors navigation.

Resumen

En esta era moderna de tecnología de computadores estamos empujando los límites inimaginables de nuestra realidad. Una de las aspiraciones humanas con estos avances es la digitalización de la tremenda cantidad de información presente en nuestra realidad. Una de las importante fuentes de información es el espacio 3-dimensional en que vivimos. Especialmente los entornos interiores que habituamos, por ejemplo, las viviendas. Con la proliferación de los dispositivos fotográficos, el desarrollo de cámaras omnidireccionales baratas ha sido uno de los intereses. Por ello, hoy en día es muy fácil de obtener datos espaciales de los espacios interiores en forma de imágenes equirectangulares.

En este proyecto estudiamos el problema de Reconstrucción de Modelos 3D de Interiores desde Imágenes Esféricas. Sin embargo, estudiamos el problema bajo métodos basados en la perspectiva ya que es posible hacer la conversión de uno al otro. Primero especificamos formalmente el problema a resolver. Encontramos distintas especificaciones y describimos métodos de reconstruccions para algunas de ellas. Seleccionamos una especificación para nuestro caso de uso. La mayoría de métodos utilizan *feature extraction*, *feature matching*, y *epipolar geometry*. Continuamos el estudio en la fase de experimentación. Proponemos hipótesis relevantes a diferentes pasos, realizamos los experimentos y sacamos conclusiones. Acabamos el trabajo implementando un sistema simple resolviendo el problema, haciendo uso de ASIFT feature extractor, FLANN kD-Tree feature matcher, y el algoritmo de OpenCV para la aproximación de la matriz esencial.

Keywords – reconstrucción 3D, procesado de imágenes de interiores, procesado de imágenes equirectangulares, navegación 360 de interiores.

Resum

En aquesta era moderna de la tecnologia de computadors estem empenyent els límits inimaginables de la nostra realitat. Una de les aspiracions humanes amb aquests avenços és la digitalització de l'enorme quantitat d'informació present en la nostra realitat. Una de les fonts importants d'informació és l'espai 3-dimensional en el que vivim. Especialment, els entorns interiors que habituem, per exemple, els habitatges. Amb la proliferació dels dispositius fotogràfics, el desenvolupament de càmeres omnidireccionals barates ha estat un dels interessos. Per aquest motiu, avui en dia és molt fàcil obtenir dades espacials dels espais interiors en forma d'imatges equirectangulars.

En aquest projecte estudiem el problema de la Reconstrucció de Models 3D d'Interiors a partir d'Imatges Esfèriques. Tanmateix, estudiem el problema fent ús de mètodes basats en la perspectiva ja que és possible fer la conversió d'un a l'altre. En primer lloc, especifiquem formalment el problema a resoldre. A continuació, trobem diverses especificacions i descrivim mètodes de reconstruccions per algunes d'elles. Seleccionem una especificació pel nostre cas d'ús. La majoria de mètodes utilitzen *feature extraction*, *feature matching* i *epipolar geometry*. Continuem l'estudi amb la fase d'experimentació. Proposem hipòtesis rellevants a diferents passos, realitzem els experiments i extraiem conclusions. Acabem el treball implementant un sistema simple resolent el problema, fent ús de ASIFT feature extractor, FLANN kD-Tree feature matcher, i l'algorisme d'OpenCV per l'aproximació de la matriu essencial.

Keywords – reconstrucció 3D, processat d'imatges d'interiors, processat d'imatges equirectangulars, navegació 360 d'interiors.

Contents

List of Tables

List of Figures

1 Introduction

We begin this document by providing some context as to why we started this project and its need. We describe our original goals for how we wanted this project to be developed and what we wanted to accomplish. Lastly, we give an overview of this document and how it reflects the development of this project.

1.1 Background

The interest in 3D model reconstruction is not particularly new. In fact, the problem has been studied since decades ago within various fields of interest (see [1] and its references). But it is only thanks to the recent technological advances over the last two decades that we are able to apply solutions to this problem in a scalable manner.

The evolution of computers have not only benefited the number crunching tasks, but its side effects can also be seen in information capturing and processing power that they bring. Today we can find very cheap devices to capture the information present in our daily lives, so that we can store it in digital form and better manipulate it to our advantage. One of the recent popular devices for these purposes are the so called 360^{o} cameras. They capture photographs of the view in all directions of our 3D space from a single point and output spherical images. Using these one can ask themselves what sort of information can they capture, process and even recover completely.

For example, spherical videos have found their usage in sharing complete visual experiences using virtual reality devices [2]. But another important and recent interest in the industry has been sharing visual and geometrical information about indoors environments, such as the floorplan, lighting conditions, or even simply spherical images for exploration purposes. Thus in this project we try to take on this challenge of retrieving information, in form of a 3D model, from indoors spherical images, stored in the equirectangular format.

1.2 Project Goals

First of all, this project is developed as part of the Bachelor Thesis of Computation specialisation in Informatics Engineering at the Facultat d'Informàtica

de Barcelona of the Universitat Politècnica de Catalunya. So one of the primary objectives is to put the skills acquired during the studies into practice. But the problem also involves familiarity with topics that were not covered within the programme, and reasonably so due to their mathematical nature. And as such, this is a research and development project.

There are two main goals with multiple sub-goals each for this project:

- Provide a knowledge base for the study of the subject.
  - Use a mathematical framework to express the knowledge.
  - Formally define the problem at hand.
  - Analyse the solutions using appropriate rigour.
  - Formulate conclusions after the development.
- Develop an indoors reconstruction system.
  - Design the system using some solution from the research.
  - Implement the system.

1.3 Document Structure

The document is structured according to the chronological progression of the project as described below:

- **Section 2.** Understanding the problem is part of solving it. In this first section we find formal specifications of the problem.
- **Section 3.** In this section we describe already existing solutions to some of the problem specifications.
- **Section 4.** In this section we describe formal models of digital images and multiple images of the same scene. These models are needed to understand the reconstruction methods.
- **Section 5.** In this section we describe how to process digital image data to actually extract information. Using this information one can obtain a reconstructed model.

- **Section 6.** The experimentation phase is an important step for transitioning from theoretical to a practical solution. In this section we describe the experiments done during the project.
- **Section 7.** In this section we describe our pipeline inspired from the methods discussed in section 3. We describe how the theory from section 4 and 5 is used in conjunction with experiments in section 6.
- **Section 8.** Using the pipeline description we implement a system. In this section we describe the design decisions of this system.
- **Section 9.** In this section we justify the achievement of the technical competences of this project.
- **Section 10.** In this section we include the reports on the planning and management of the project.
- **Section 11.** Finally, we end this document by forming some conclusions and reflecting on possible future derived projects.

1.4 Notation

In this document we use the following notation for the mathematical expressions. A scalar is written as a, b, c etc. and it maybe a letter or a word. A 3D point is written as $\boldsymbol{X}$. A 3D line is written using two point, for example as $\boldsymbol{OX}$. A 3D vector is represented as $\boldsymbol{v}$. A ray is the span of a vector, and is written in lowercase as simply $\overrightarrow{\mathbf{v}}$. A linear transformation or a matrix is written as $\mathbf{T}$. A set is denoted with $\mathbb{S}$.

2 3D Reconstruction Problem

The problem of 3D Reconstruction is an ambiguous one because no general method exists to recreate any given 3D point perfectly from a photograph. Thus it becomes necessary to detail our problem specification. This includes detailing the input, output and any other constraints that may be required to be held. In this section we outline a few well specified reconstruction problems that we thought to be of interest during our research. We describe the inputs[1], outputs and constraints separately, and so a formally specified problem can be stated in terms of a sensible combination of these, e.g., Input Type 1, Output Type 2.

2.1 Input Types

1. An ordered sequence of captured images. The capture locations are very close to each other and high number of images are available within a single room. This is similar to recording a video while walking around a space.
2. A sequence of images. Not particularly ordered. But high probability that two images taken one after another partially share the same view of the space.
3. A set of images. No order assumed. Low number of images per different rooms within an interior space.

2.2 Output Types

1. Each image represents a camera instance. Find the relative position and orientation of different camera instances with respect to each other. Or equivalently find the relative camera motion.
2. Reconstructed 3D points and a highly detailed mesh of the interior space recreated from these points.
3. An approximated planar model to the 3D points of the interior space, ignoring the objects present in the images.

[1]The input is formed of images. Even though our images are spherical, we will ignore this fact for now when studying the solutions to our problem and assume ordinary perspective images. This is because we ended up implementing a workaround at the user interaction level that is discussed separately.

2.3 Constraints

1. The reconstruction process for a pair of images should be able to execute in almost real-time. This means whenever an images is loaded to be reconstructed, it should be perceptively instant to provide feedback from the results. Typically a waiting time of 5-10 seconds is acceptably almost real-time. This falls under a performance constraint.
2. A reconstructed model should have a known ratio. That means we should be able to map measurements in abstract model units to real world metric units.
3. The reconstructed model should be complete and similar. That is, we must only obtain a single model which should contain information retrieved from all input images, and the reconstructed coordinates of a point must at most differentiate by a scalar factor from the reality. This implies the conservation of real-world distances ratios in the model.

2.4 Our Specification

The study of the problem statement was one of the first things we had to work out. This was an initial attempt at formally understanding the problem specification, but despite that, we haven't completely presented a simplified view of the problem. However, with these broken down requirements of the original problem, we can better analyse which of the existing similar but simpler problems and solutions can be applied to give a system with which we are satisfied.

At the beginning of this project we wanted to design our system to work under **Constraint 1** and **Constraint 3** and have the following interface:

$$\textbf{Input Type 3} \longmapsto (\textbf{Output Type 1},\ \textbf{Output Type 3})$$

3 Reconstruction Methods

While we were trying to understand the problem statement in more depth we were also trying to find existing solutions. Using the broken problem specifications from the previous section, we describe some generic algorithmic families, existing software systems and existing research. In the final version of the system we did not implement a single solution completely, but all of the following were relevant considerations and inspirations that guided our research. Thus we also justify the parts we did apply and the parts we didn't apply from the solutions to our system.

3.1 Solution Families

The solution families are result of studying the problem from different point of views, due to different motivations and backgrounds, different domains of application, and different assumptions.

3.1.1 Visual Odometry

Background The purpose of odometry [3] is to perform motion measurements locally. This can be done using data from various sources. Visual odometry [4] in particular makes use of vision based sensors. Its main area of application is in robot navigation. Often low-end vision based system is combined with other sensors to increase their accuracy.

Specification Strong **Constraint 1** (full real-time) and **Constraint 2**.

$$\textbf{Input Type 1} \longmapsto \textbf{Output Type 1}$$

Procedure

1. Apply *feature extraction*.
2. Apply *feature matching* to consecutive images to construct optical flow.
3. Apply *statistical* error filtering on the flow field to remove outliers.
4. Perform *epipolar geometry* estimation encoding the motion.
5. Use vision-external information to obtain the real motion measurement.

Discussion Visual odometry seems a good fit to partially solve our problem if we could require an **Input Type 1**. However, we strongly want to avoid requiring high number of images. We also lack the presence of **Constraint 3** in this method.

3.1.2 Visual SLAM

Background SLAM [5] stands for simultaneous localisation and mapping. It could be seen as a globally consistent version of odometry and thus also holds much more importance in autonomous robot navigation and mapping systems. The objective is to create a map of the environment starting from no data. Thus the robot has to take action and be able to localise itself, that is, estimate its relative motion locally which is then integrated into the global map that is simultaneously being created.

As we can see compared to odometry local errors hold much more importance in SLAM algorithms, because each of them can affect the global map. One of the final steps to refine the map and reduce errors is called loop-closure.

Obviously, visual SLAM [6] is also just a particular case of this algorithmic family using vision based systems to solve the problem.

Specification Strong **Constraint 1**, optionally **Constraint 2** and **Constraint 3**.

$$\textbf{Input Type 1} \longmapsto \textbf{Output Type 1}$$

Procedure

1. Apply *feature extraction*.
2. Apply *feature matching* to consecutive images to construct optical flow.
3. Apply *statistical* error filtering on the flow field to remove outliers.
4. Perform *epipolar geometry* estimation encoding the motion.
5. Use vision-external information to obtain the real motion measurement.
6. Perform loop-closing check and apply global optimisations.

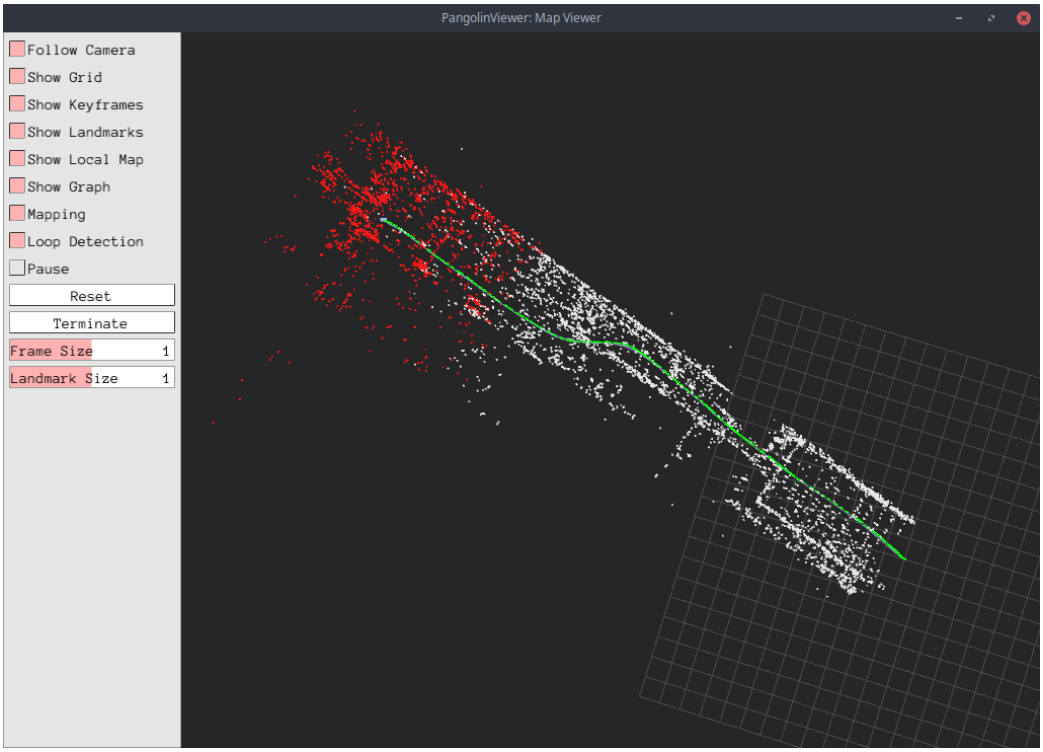

Figure 3.1: Visualised output of OpenVSLAM [7] system on a test environment. [Source: *Compiled by author.*]

Discussion This method is very similar to the previous one. But it does, though, hold the **Constraint 3** we wanted. This is why we decided to try this method out in first weeks of this project. In the figure 3.1 we see a reconstructed trajectory from a test spherical video. This was, in fact, very accurate to our trajectory that we took while recording the video. But of course we are not interested in videos and so we decided to keep investigating this and related methods but for the image inputs.

3.1.3 Structure from Motion

Background This method is intended for using real world objects to create virtual 3D models on object-level detail. Its main area of study can be categorised as photogrammetry [8], obtaining measurements from photographs. It has great interest from some computer graphics applications, because you

can capture and reproduce the details from the real-world in a computer generated environment. Thus, its main focus is on obtaining a fine detailed well integrated complete model of a single object or single set of objects altogether. For this it also requires computing the motion of the camera in between the different shots of the objects. Hence the name Structure from Motion.

Some of the examples include capturing and reconstructing models of important tourist places, e.g., Colosseum, and modeling of real-world places to perform CGI post-processing for cinematic purposes, e.g., Meshroom [9].

Specification **Constraint 1** and strong **Constraint 3**.

$$\textbf{Input Type 3} \longmapsto (\textbf{Output Type 1}, \textbf{Output Type 2})$$

Procedure

1. Apply *feature extraction*.
2. Apply *feature matching* to consecutive images to construct optical flow.
3. Perform *epipolar geometry* estimation encoding the motion.
4. Extract motion from the epipolar geometry.
5. Apply *bundle adjustment* for all images as a whole to obtain an optimally correct motion for all of the images.
6. Apply *multiview stereo* methods to obtain a fine detailed mesh.

Discussion This method seems like a good fit for our case. It obeys the required constraints. The input type corresponds well with our ideal case and the output type is even finely detailed than our ideal output. We could very well replace step 6 from the procedure to improve our performance with a step that gives us an **Output Type 3**. However, there is a small problem with step 5. This step is what ensures that **Constraint 3** is held. As we see, it is a strong constraint, and as such requires all of the images to share at least some partial view of same scene. But in indoors scenes with scarcely taken images it could be difficult to share partial views between all images, but it could very well be the case for small subsets of 2-3 images of the whole set of unordered images. This then requires us a different approach to resolve this little hiccup in adapting this method to our case.

3.2 Existing systems

The goal of this project was to create a knowledge base for reconstruction systems. So it should collect the necessary theoretical concepts needed in presentable form for the purpose of indoors reconstruction. The desired system to be developed should be based on this. So it was not enough to use and extend existing solutions. But despite that, we think that the solution we wanted still differed from what is employed in the available products. In this section we mention some of the existing systems and describe how they differ.

3.2.1 Matterport

Matterport might be the only best service out there which provides very accurate and highly detailed models. They do so by a mixture of LiDAR technology and machine learning with tremendous amount of data collected by them. Thus, their approach although it works nicely, it does not fit our criteria and it is not affordable because of how much data and resources are required.

3.2.2 Benaco

Another service is called Benaco. This seems to solve the problem that is most similar to ours. They do not require any manual input, but instead advise to follow their instructions strictly, which is the point at which our developed system differs. For a good reconstruction they recommend images 1m distance apart. We want to avoid this restriction. Also as they claim on their webpage, their technology is proprietary and so unfitting for the purpose of research.

3.2.3 OpenSfM

OpenSfM [10] is one of the few open-source frameworks that implements a structure from motion pipeline. It gave us promising results during our first weeks of research and testing, and it accepted RGB equirectangulars directly. Even though it performed quite well, there were two downsides to it: inconsistency of outputs between executions with same input due to their usage of randomness, and it was very automised and abstracted, leaving little room for adapting it to our case. The framework is developed by Mapillary, who

describe themselves as providing a service for street map creations similar to ones available in Google Maps. Thus, we see how it is quite similar in conceptual needs for our problem, but is adapted for outdoors reconstructions.

3.3 Existing research

In the paper *State-of-the-art in Automatic 3D Reconstruction of Structured Indoor Environments* [11] they propose the following subdivision for our problem.

1. room segmentation, grouping of images according to their associations of rooms;
2. bounding surface reconstruction, general room layout inference; (*Note: this corresponds to our* **Output Type 3**.)
3. indoor object detection and reconstruction, finding details in the rooms;
4. integrated model computation, merging the solutions of the previous three; (*Note: this corresponds to our* **Constraint 3**.)
5. visual representation generation, interacting with and displaying the model on a screen.

At the start of the project we limited ourselves to finding solutions and developing a system for the subproblems 1, 2, 4 and 5 because it compares with our initial problem specification.

But we noticed that our problem greatly differs in the nature of input data than what is investigated in most of the works mentioned in that paper. We require spherical RGB images captured from sparsely distributed locations. While in table 3.1 (extracted from the paper), most of the methods require single perspective images, dense point clouds, or RGB-D images. However there is one mention from Pintore et al. that suits our needs.

Since the table summarises works for subproblem 2, which is our first priority in this problem decomposition, we decided to investigate the paper from Pintore et al. This is because it coincides with our multiple sparse RGB spherical images input data. We discuss the paper in the next section.

Method	Input type	Input requirements	Output	Priors	Features
Delage et al. [DHN06]	Single RGB	Single pinhole	Floor-wall planes	FW	Vertical-ground fold-lines
Hedau et al. [HHF09]	Single RGB	Single pinhole	Oriented box	CB	Geometric context (GC)
Lee et al. [LHK09]	Single RGB	Single pinhole	IWM planes	IWM	Orientation map (OM)
Furukawa et al. [FCSS09b]	Dense RGB	Multi pinhole	3D mesh; reg. images	MW	VF; FP evidence
Jenke et al. [JHS09]	Dense PC	Two scanners	3D mesh	MW	Cuboids merging
Flint et al. [FMMR10]	Single RGB	Single pinhole	Oriented planes	IWM	C-F homography
Budroni et al. [BB10]	Dense PC	Markers	3D mesh	IWM	Vertical walls via rotational sweep
Flint et al. [FMR11]	Dense RGB	Multi pinhole (video)	Oriented planes	IWM	GR+multi-view features
Turner et al. [TZ12]	Dense PC	Scan positions (per-point)	2D floorplan	VW	Curved walls
Turner et al. [TZ13]	Dense PC	Scan positions	3D mesh	MW	Voxel carving
Bao et al. [BFFFS14]	Dense RGB	Multi pinhole (video)	3D box	CB	GC+OM+multi-view features
PanoContext [ZSTX14]	Single RGB	Single panorama	Oriented box	IWM	GC+OM on panorama
Cabral et al. [CF14]	Sparse RGB	Multi panorama; dense PC	Textured 3D mesh	IWM	C-F homography; FP evidence
Oesau et al. [OLA14]	Dense PC	—	3D mesh	PW	2.5D cell complex
Turner et al. [TZ14]	Dense PC	Scan positions (per-point)	2D floorplan	VW	Triangulation of 2D wall samples
Mura et al. [MMJV*14]	Dense PC	Scan positions	3D mesh	AW	Occlusion-aware; diff. distances
Ikehata et al. [IYF15]	Dense RGB-D	Multi panorama	Structured 3D shape	MW	FS-S evidence
Yang et al. [YZ16]	Single RGB	Single panorama	Oriented 3D facets	MW	GC+OM; 3D facets
Ochmann et al. [OVWK16]	Dense PC	Scan positions (per-point)	3D mesh	AW	Parametric models; thick walls
Mura et al. [MMP16]	Dense PC	Scan positions; oriented points	3D mesh	PW	Fully 3D reconstruction
Pano2CAD [XSKT17]	Single RGB	Single panorama	3D shape	IWM	GC+OM on panorama
Ambrus et al. [ACW17]	Dense RGB-D	—	3D mesh	VW+PW	Artificial scan positions
Mura et al. [MP17]	Dense PC	Oriented points	3D mesh	PW	Artificial scan positions
Murali et al. [MSOP17]	Dense RGB-D	—	3D mesh	MW	Lightweight; cuboids merging
Liu et al. [LWKF17]	Dense RGB-D	Multi panorama; 2D florplan	Labeled 2.5 shape	MW	CNN+IP
Pintore et al. [PPG*18]	Sparse RGB	Single panorama	Textured 3D shape	AW	E2P;C-F homography
FloorNet [LWF18a]	Dense RGB-D	Video	2D floorplan	MW	Hybrid DNN architecture
Pintore et al. [PGP*18]	Sparse RGB	Multi panorama	Structured 3D shape	VW+PW	E2P facets
Yang et al. [YZS*19]	Sparse RGB	Dense point cloud	3D shape	IWM	Curved walls
DuLa-Net [YWP*19]	Single RGB	Single panorama	3D shape	IWM	E2P;C-F homography
HorizonNet [SHSC19]	Single RGB	Single panorama	3D shape	IWM	1D vectors encoding
Ochmann et al. [OVK19]	Dense PC	Oriented points	3D mesh	AW	2.5D cell complex; thick walls
Floor-SP [CLWF19]	Dense RGB-D	Multi panorama	2.5D floorplan	VW	Shortest polygonal loop

Table 3.1: Bounding surfaces reconstruction methods. Solutions for the sub-problem 2. The references to the mentioned works and methods are available in the source paper. Summary of the approaches described in section 6 of the source paper, arranged by chronological order. See the table 3 in the source paper for more details. [Source: *State-of-the-art in Automatic 3D Reconstruction of Structured Indoor Environments* [11]]

3.3.1 3D floor plan recovery from overlapping spherical images

We decided to pick and study the work from Pintore et al. [12]. Their work does not point to any implementation details of such a system. It describes a pipeline to solve subproblem 1, that could be solved via a structure-from-motion step, and the result is exploited for solving subproblem 2 and later subproblem 4.

Thus we see how this work could be a potential fit for our case. It makes use of structure from motion method that we had already strongly considered, but also describes how to extend to exploit it for the **Output Type 3** (via subproblem 2) that we wanted. It also mentions how to divide the reconstruction per room and later integrate that by parts, rather than with a single bundle adjustment step from structure from motion. Hence, it changes the strong **Constraint 3** to a normal **Constraint 3** that is also of interest to us.

After seeing the similarity of the problem explored in this paper, we decided for our work to attempt at starting to implement a solution described in this paper. But for the purposes of TFG and due to time limit, we decided to tackle the first part for now, using an hybrid approach of manual image registration and automated structure from motion, and ended up partly solving the subproblem 4 as well. We describe our pipeline in the section 7.

3.3.2 Floor-plan reconstruction from panoramic images

Another similar research work is contained in the published paper *Floor-plan reconstruction from panoramic images* [13]. Their work employs a similar strategy of semi-automation that we ended up using for our project, but inputs from users require high precision for a more accurate result. This is not appropriate for our purposes because even though we also require user input, we do not demand such precision, and also intend to get rid of this input in the future. Thus not expecting a precise control from the user from the start would lead us to a much more robust system in the future.

4 Geometry of Digital Images

In this section we first explore a formal model of a digital image. This model is also used to describe the feature extraction and feature matching steps of the reconstruction procedures. We then expand the model to more than one image of the same space. In the resulting model we can study the geometry of two or more images and formulate the mathematical relations encoding the relative position of our images or cameras that we want to obtain.

4.1 Digital Image Model

Camera is an invention that behaves similar to human eyes. Usual cameras mostly produce images using the perspective projection of the space. Unlike natural processes, acquisition of digital image requires discretised model of reality, so we also describe its effects in reconstruction systems.

4.1.1 The Pinhole Camera Model

Cameras, like human eyes, produce images by sensing colours from light rays that pass through their lenses. In real life cameras are quite complex, and although the principle of their functionality is simple, the processes involve many parameters. But once a digital image is produced, we can treat them to be originated from a simpler approximated model. The most common model used is the pinhole camera model illustrated in the figure 4.1.

This model first places a reference frame into the scene, where the origin is the camera pinhole and the axis are arranged using a predefined convention with respect to the camera's forward looking direction. What is important here how a certain point in the scene is mapped to the image. The image is represented as the 2D subspace of the 3D scene space, the image plane. This plane has a fixed distance of f from the origin and is perpendicular to the forward direction.

Given any point $\boldsymbol{P}$ in the scene, it's position on the image is given by the intersection of the line $\boldsymbol{OP}$ and image plane. It is apparent from the illustration that this produces reflected images of the space, and so correction is needed. In fact, human brain subconsciously performs this correction.

There are two properties of importance here: each point in the 2D image is represented by the a line (or a ray, modeling the light rays) in the 3D

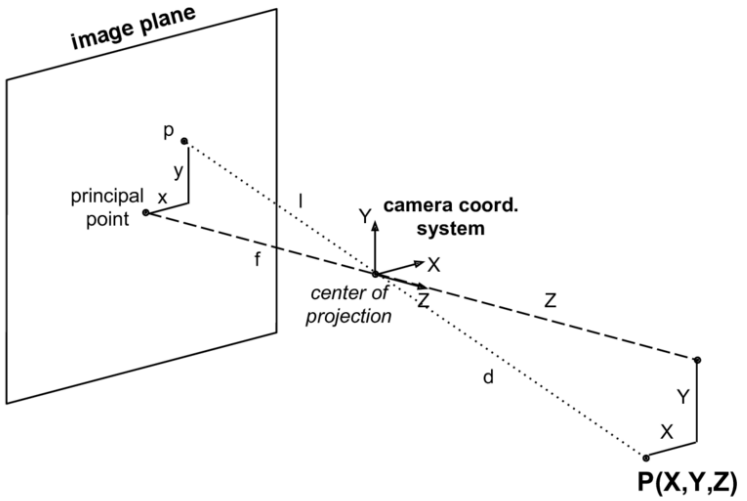

Figure 4.1: Pinhole Camera Model. [Source: *Courtesy of Nick Van Oosterwyck.* [14]]

scene space, and varying the z-ordinate of a point inversely scales the xy-coordinates of the point on the image plane. This latter property is actually the perspective effect.

In mathematics and geometry, the formal model with these properties is called a *projective space*. It is an important theory of perspective projections that one must understand and be comfortable to work with when studying the problems related to perspective images. But we don't think it is necessary to reproduce the explanations in this document as authors with much more experience in the field already have published works of greater quality. Some of these we found to be useful are the book *Multiple view geometry in computer vision* [15] and the lecture notes *Projective Geometry: A Short Introduction Lecture Notes* [16].

4.1.2 Discretisation

Current model is very simple and enough to understand geometry taking into account the perspective property of the images. However, we cannot represent a digital image with the current image plane. Even if the image plane is finite, as illustrated, it can still contain infinite points. And in

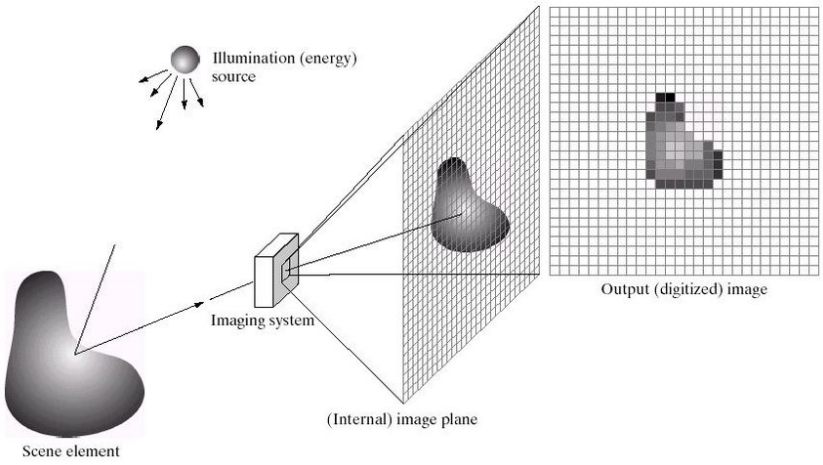

Figure 4.2: Image Discretisation. [Source: *Digital Image Processing 4th Ed. by Gonzalez & Woods* [17]]

real computer hardware it is impossible to store such a mathematical object exhaustively. Thus, normally an image is represented as a discretised grid of values. Each cell[2] in the grid takes on the average of the values of light rays that are projected to it. This process is illustrated in the figure 4.2.

The averaging of values can be an important factor in detecting useful features and reliably able to process them for information inference. Thus, having higher number of grid cells per unit area is much desired. We have not explicitly described a model for spherical imagine, but one can imaging using two hemispheres as a projective surface and merging them into an image. Thus when we resample our images for perspective based reconstruction methods, we first conceptually create a sphere in the 3D space with the equirectangular image projected on to it, which was also discretised, and then we take the image of this sphere using the pinhole model at the center of it. This recapturing also discretises the values once again. Here, we need to be careful, because each time we sample and average values, we lose information. Thus it is important to have as much precision as possible in our images. Fortunately, in this project, we work with very high resolution equirectangular images. Thanks to which we were able to employ a better resampling strategy as we describe in the experiments and later sections.

[2]Usually called a "pixel", derived from "picture element".

4.1.3 Quantisation

Similarly to discretisation, we also have to represent potentially infinite precise values of light intensity with finite resources. Thus, normally the pixel values have the precision defined by the bit depth and data type used. This process is called quantisation.

4.2 Two Image Epipolar Geometry

Now that we have a model for a single image, we are interested in finding a model for two images taken in the same scene space. This model will let us describe relations between two images of the scene, and give us a solution to finding the relative position between camera instances, which is needed for the reconstruction.

In the figure 4.3 we see two images in the same space. We can see many different geometrical elements in the illustration, but notice the points[3] $\overrightarrow{\mathbf{e}}$ and $\overrightarrow{\mathbf{e}}'$. These points are placed on the image planes, and in fact they are the image of one center of projection (the pinholes) as seen from the other. These points are called epipoles. Hence the name, *epipolar geometry*.

4.2.1 Derivation of Essential Matrix

Given the epipolar geometry, illustrated in the figure 4.3 we attempt to derive an algebraic relation between the scene space and its two images.

First, notice that given an arbitrary point $\boldsymbol{X}$ in the space, each image has a corresponding line (representing the light ray), $\boldsymbol{CX}$ and $\boldsymbol{C'X}$. These two lines define the epipolar plane that interesects with the image planes. The intersections are called the epipolar lines, because they always contain the epipolar point of the images. They also contain the image point of $\boldsymbol{X}$, represented by camera local coordinates $\overrightarrow{\mathbf{x}}$ and $\overrightarrow{\mathbf{x}}'$.

Second, we know that there exists a linear transformation that maps a point $\overrightarrow{\mathbf{x}}$ of first image plane to a point $\overrightarrow{\mathbf{x}}'$ of the second image plane. This transformation involves a translation in the direction of $\overrightarrow{\mathbf{t}}$. We can use this transformation to encode the property that $\overrightarrow{\mathbf{x}}$ and $\overrightarrow{\mathbf{x}}'$ lie on the same epipolar plane using the following algebraic formula (section 7.1 from [19]):

[3] An image plane point may be represented by a ray in a camera reference frame.

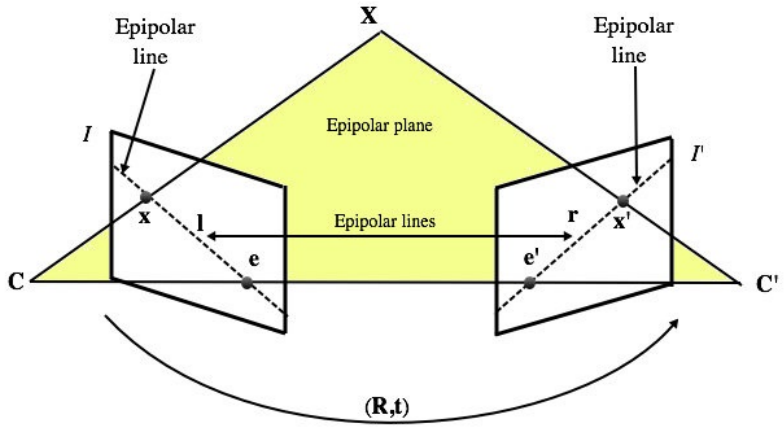

Figure 4.3: Epipolar Geometry of two Images. First image plane on the left. Second image plane on the right. [Source: *Courtesy of Willy Azarcoya-Cabiedes.* [18]]

$$\boldsymbol{x}'^{T}[\boldsymbol{t}]_{\times}\mathbf{R}\boldsymbol{x} = 0$$

This expression is called the *coplanarity constraint*. It involves a matrix that is unique (upto a scalar factor) for a particular configuration of the two image planes, and as such it is called the *essential matrix*: $\mathbf{E} = [\boldsymbol{t}]_{\times}\mathbf{R}$.

The matrix $[\boldsymbol{t}]_{\times}$ denotes the operator that generates the cross product of a vector with the vector $\boldsymbol{t}$. Then the geometric interpretation of the coplanarity constraint is that the expression $\mathbf{E}\boldsymbol{x}$ denotes the normal vector of the epipolar plane in second camera's coordinates, and the expression $\boldsymbol{x}'^{T}(\mathbf{E}\boldsymbol{x})$ confirms that the mapped point $\overrightarrow{\mathbf{x}}'$ lies on this plane, i.e. the normal vector is perpendicular and hence the dot-product is null.

4.2.2 Essential matrix estimation

There exist algorithms based on linear algebra to find solutions to the essential matrix (for example [20]). Then this matrix can be further decomposed into the rotation transformation and a unit translation vector ([21]). The reason for only obtaining a directional vector for the relative positioning of

cameras is because there is no assumption on the scale of our epipolar geometry. The scale is an information that cannot be retrieved purely from photographic data. This is actually the **Constraint 2** from the problem specifications. Some prior assumptions must be made in order to precisely obtain a model with realistic coordinates. Because, otherwise one cannot differentiate between for example images of a dollhouse model and images of a realistic indoors scene. In the dollhouse the images are much more likely to be on centimeter scale, and in realistic indoors environment the images are typically taken a few meters apart.

4.2.3 Statistical error correction

Due to how digital images are stored with discretised information, one cannot perfectly retrieve the essential matrix, because there is a loss of information. This and other factors may cause some feature matches to be erroneous. Thus, normally the implementations try to find a statistically accurate estimation to the essential matrix instead.

Typically, for a statistical optimal solution the algorithm is given an input set rather than a single input. And then it randomly chooses the inputs. For each input it estimates an approximate essential matrix and uses it to compute the values of the coplanarity constraint. If value is not 0, then there is a statistical error present, in this case it is called the *reprojection error*.

Since it is possible to estimate more than one approximate essential matrix $\hat{\mathbf{E}}$, we take our essential matrix to be the approximate that minimises the reprojection error function r:

$$\mathbf{E} = \arg\min_{\hat{\mathbf{E}}} r(\hat{\mathbf{E}})$$

Much more detailed explanation of a popular statistical optimal algorithm RANSAC can be found in [22].

5 Feature Extraction and Matching

Feature processing is an important step in the reconstruction methods that can solely decide the accuracy of the results. Thus it is important to understand how feature extraction and feature matching works so we can ensure the robustness of the pipeline steps involving these.

5.1 Feature Extraction

For a human, features are automatically formed and recognised in images without realising it. A feature lets us attach semantics to the visual data, which we can then use to recall additional information and make comparisons. Us humans can easily adapt to higher level features, like how a face looks like, how a car looks like, etc, via experience. But when trying to write down a clear algorithm[4] for it, it becomes difficult. Instead, there has been decades of work put into detecting simpler features based on single points in the images. These features are called *keypoints* and there exist many algorithms for detecting them and describing them. Here we discuss some that we thought to perform well for our use case.

5.1.1 SIFT

This is one of the popular feature detection method and it works quite robustly in most cases. The keypoint descriptions obtained from this are invariant to rotation, illumination, relative translation and scale changes in the images. Thus it can be an ideal feature detection method in many image processing tasks.

The first step in the detection is to obtain various "scales" of the image. This is done through successive blurring of the image. Normally, it is quite easy to detect keypoints that are very fine detailed and vary over a single pixel. Using this blurring we can simulate the detection of features that are more high level in detail and they are detected by inspecting the change over more than single pixel. Using blurring we can simplify this process.

[4]Note, we are purposefully trying to avoid the discussion of machine learning approaches here. Even though those feature processing algorithms may give much better accuracy, they require additional knowledge of machine learning theory which we did not want to tackle within this project. And the topics discussed here are still essential and serve as the foundation for machine learning computer vision algorithms.

This is also the step that ensures scale invariance. In the next step it uses the *contrast threshold* parameter to eliminate the keypoints that do not have sufficient change around their neighbourhood pixels. Finally, it describes each keypoint using information present in the neighbourhood pixels. This is the key in achieving rotation, illumination and relative translation invariance.

A more technical explanation of SIFT feature detection and description can be found on [23].

5.1.2 ASIFT

All of those invariances are very much needed for our images. When two images of the same scene are taken from two distinct position it is easily possible that an object appears rotated or translated in the images. It is also quite usual that an object appears to have illumination changes because the images are taken at different times, and the lighting varies depending on the factors of natural lighting, artificial lighting and camera's position itself.

However perspective is also present in our images. ASIFT (Affine-SIFT) is a feature detector developed with the goal of combining perspective invariance in the described keypoints. The technical details are quite complex ([24]), but essentially it works by randomly applying rotations to the image plane. By doing so it is able to approximately simulate how different perspective projections of a single point may look like. Thus, it is more likely to have the same description for keypoints in two images representing the same scene point.

5.2 Feature Matching

Given two images, we want to compare the information contained in them. The task can be as simple as identifying if the images share the same view of the scene, or more complex as such as finding the epipolar geometry of the scene. These tasks can be achieved by first finding keypoints in the images, and then matching them instead. We first discuss some matching methods, and then how to filter the matches that have high probability of being accurate.

5.2.1 BFM

Brute-Force Matching is the simplest form of matching algorithm. Given two sets of keypoints, you just match each keypoint in the first set with all of the keypoint in the second set. For each set, the match is selected by exhaustively scoring (usually by some distance function) and selecting the pair with the best score (usually it is the minimum distance).

Let $\mathbb{F}_1$ be one set of keypoint features, and $\mathbb{F}_2$ be another set of keypoint features. Then for some score function s, following is the set of matches:

$$\{ (k_1, k_2) \mid k_1 \in \mathbb{F}_1,\ k_2 = \arg\min_{k \in \mathbb{F}_2} s(k_1, k) \}$$

From this we can see that computing the set of matches using this method has a tight bound of $\boldsymbol{\Theta(|\mathbb{F}_1||\mathbb{F}_2|)}$, a polynomial of second-degree. Thus, it is not ideal to have this asymptotic bound for our system if we are to have many keypoints per image.

5.2.2 FLANN kD-Tree

Another possible matching algorithm is the approximate nearest neighbour[5]. The problem of feature matching can be formulated in terms of a vector space of keypoint descriptions as the points. We can then partition this vector space using a kD-Tree ([25]), where the parameter k is the dimension of the vector space, and determined by the type of descriptor used for the keypoints.

Given two sets of keypoint features $\mathbb{F}_1$ and $\mathbb{F}_2$, we can build a kD-Tree for the second set. Then the match set is obtained using a query function $q : \mathbb{F}_1 \to \mathbb{F}_2$ based on this tree:

$$\{ (k_1, k_2) \mid k_1 \in \mathbb{F}_1,\ k_2 = q(k_1) \}$$

This function, q, for a given keypoint in the first set, should return the most similar keypoint in the second set. Thus, the complexity of comput-

[5]FLANN stands for Fast Library for Approximate Nearest Neighbors: `https://github.com/flann-lib/flann`. And kD-Tree are one of the possible approaches to store and perform its operations.

ing this match set depends on the complexity of q, which on average is $\mathbf{O}(\log|\mathbb{F}_2|)$. Then the complexity of computing this set is $\mathbf{O}(|\mathbb{F}_1|\log|\mathbb{F}_2|)$. In case of only requiring approximately similar result, the actual execution time should be even faster compared to brute forcing. A more correct complexity bound may be found for the approximate search, but the proofs can often be very complex. More information can be found on [26].

5.2.3 Match Filtering

Not each match in the match set is guaranteed to be correct. It could be that a match between two keypoints, actually relates pixels representing different real world points, because different real world points may look similar. Thus, to increase the probability of having correct matches, the match set can be filtered. There are two types of filters that we used in this project.

One filter is the ratio filter. It actually uses a match set with the best and second best matches. Then it compares if the second best match is as good as the first best match, using a ratio threshold value. For example $best\_match \cdot ratio < second\_match$.. The filter then lets pass only those best matches that do not pass this ratio test, because they are the only ones much better than their second best matches, and thus more likely to be uniquely and correctly matched.

Another filter is the cross-check filter. It uses two match sets. The second set is created by swapping the first and second feature sets. Then the filter simply lets pass all those matches in the first set, that are also present in the second match set. Although the matches in the second match set have the swapped pairs and this fact should be taken into account.

6 Experiments

The experimentation phase was an important step for transitioning from research to development. We even had to prolong this phase and repeat some experiments, add new ones and use additional datasets. In this section we include some of the experiments conducted. Experiments relating to the section 5 are of most importance.

All of the experiments were executed on a Linux Kernel 5.11 and on Intel 8th Gen i5 CPU-only laptop, Lenovo Ideapad 330.

6.1 Impact of Contrast Threshold in SIFT

One of the parameters of the SIFT feature detector is the contrast threshold that affects the considered pixels for described keypoints.

Hypothesis

In indoors scenes there are a lot of planar texture regions, e.g. walls. Thus, having lower contrast threshold should yield more keypoints on the walls.

Conclusions

As we can see in the figure 6.1 even though we allow the detector up to 1000 maximum keypoints it is not able to detect many with a contrast threshold 0.1. Decreasing it to 0.01 actually does improve the results, especially around the edges. But still we can the right edge of wall does not have many edges, this is due to how the illumination bounces in indoors spaces. In the end lowering the threshold to 0.001 yields many more keypoints, and even on almost planar texture that the wall has. So in conclusion we think that our hypothesis holds true.

6.2 SIFT vs ASIFT

Different feature detectors trade robustness for execution time and memory. We want to analyse these characteristics to choose the best one for our system. We execute 100 iterations for both of these detectors and descriptors, limiting the maximum n. of keypoints around 20 thousand.

Figure 6.1: An image of a bedroom wall with maximum of 1000 detected keypoints using SIFT. Contrast threshold used, from left to right: 0.1, 0.01, 0.001. [Source: *Compiled by author*]

Hypothesis

Since our images can easily exhibit viewpoint variance, ASIFT should detect better keypoints than SIFT for matching.

Conclusions

In the table 6.1 we first notice how ASIFT is x10 slower than a simple SIFT detector, although both use SIFT descriptors. This is due to how ASIFT simulates viewpoint variations on the images and then applies the detector. We see how the detected matches are also much more significant and more matches are passed through the filters for ASIFT detectors. This implies that viewpoint invariance should in general be a good characteristic to have in our system, as we do have very scarcely captured images in the indoors scenes. In the figure 6.2 we can see our hypothesis held true, especially how matches of ASIFT keypoints are on viewpoint dependent pixels, than those matched with SIFT keypoints, which are detected on almost viewpoint independent pixels.

6.3 Impact of Focused Regions in Feature Matching

We have high resolution spherical images. Thus, we could potentially obtain keypoints from a wide range of regions. And different regions could look similar but have different locations in original scene space. Thus we can resample our high resolution spherical images into focused perspective images

Figure 6.2: Visualisation of matches obtained from keypoints detected with (top) SIFT and (bottom) ASIFT. [Source: *Compiled by author*]

Detector	Average Iter. Time (s)	Pre-Estimate Matches	Post-Estimate Matches
SIFT	0.3549	124	7
ASIFT	3.5512	203	24

Table 6.1: Pre-Estimate matches are computed using FLANN KD-Trees and match filters. Post-Estimate matches are further filtered using statistical reprojection error of less than 0.1%. [Source: *Compiled by author*]

by controlling the desired FoV.

Hypothesis

Manually selecting the regions by controlling the FoV should yield better matching than using a predetermined FoV in the perspective resampling.

Conclusions

In the figure 6.3 we see in the top picture how there are already many matches on the sofa. But it is very clear how even slightly focusing the left image more on the sofa increases the number of matches in total, and in fact more matches are significant. Thus we conclude our hypothesis to be held true.

In this experiment we only tried very simple region restricting technique using zoom in (via resampling using lower FoV). During the project we did consider other sorts of more complex preprocessings to refine and improve our keypoint matches, from simple top-half and bottom-half split of the equirectangular images to region segmentation. But we did not had enough time to experiment with them and we were already satisfied with these results.

6.4 BFM vs FLANN kD-Tree

Similar to feature detectors, feature matching can also trade accuracy for execution time.

Hypothesis

Brute force matching should result in better keypoint matches than approximate matching, i.e, FLANN based kD-Tree matcher.

Figure 6.3: Visualisation of matches obtained from (top) not focused regions (154 matches), i.e, having the same FoV, and (bottom) having approximate focus on the same object (239 matches). The matches are filtered without essential matrix error correction. [Source: *Compiled by author*]

Matcher	Average Iter. Time (s)	Pre-Estimate Matches	Post-Estimate Matches
BFM	8.1918	198	61
FLANN	1.3347	239	62

Table 6.2: Both matchers started with around 20 thousand keypoints per image. Pre-Estimate matches are processed using match filters. Post-Estimate matches are further filtered using statistical reprojection error of less than 0.1%. [Source: *Compiled by author*]

Conclusions

First of all we see in the table 6.2 how brute-force matching is significantly slower than approximate matching. Then we observe that how pre-estimate matching for BFM is a bit more restrictive than FLANN based matching. But then we observe that the number of post-estimate matches are almost the same. From this we conclude that BFM is indeed more accurate by itself. But we observe how both BFM matches and FLANN matches are both reduced to magnitude of 200 after simple filtering, and both result almost in same number of matches after statistical reprojection filtering. These results can be seen in the figure 6.4, and from this we conclude that our hypothesis does not hold true.

6.5 Impact of Coordinate Normalisation in Essential Matrix Computation

Due to how floating-point arithmetic works in computer systems, estimation of essential matrix could result in inaccuracies due to numerical reasons.

Hypothesis

For keypoints using coordinates between [-1, 1] should result in better estimates than using pixel coordinates.

Conclusions

In the figure 6.5 we see our hypothesis in fact does not hold. So we will have to further study our assumptions and the algorithms for numerical details.

Figure 6.4: Visualisation of matches obtained from (top) Brute-Force Matcher and (bottom) FLANN KD-Tree matcher. [Source: *Compiled by author*]

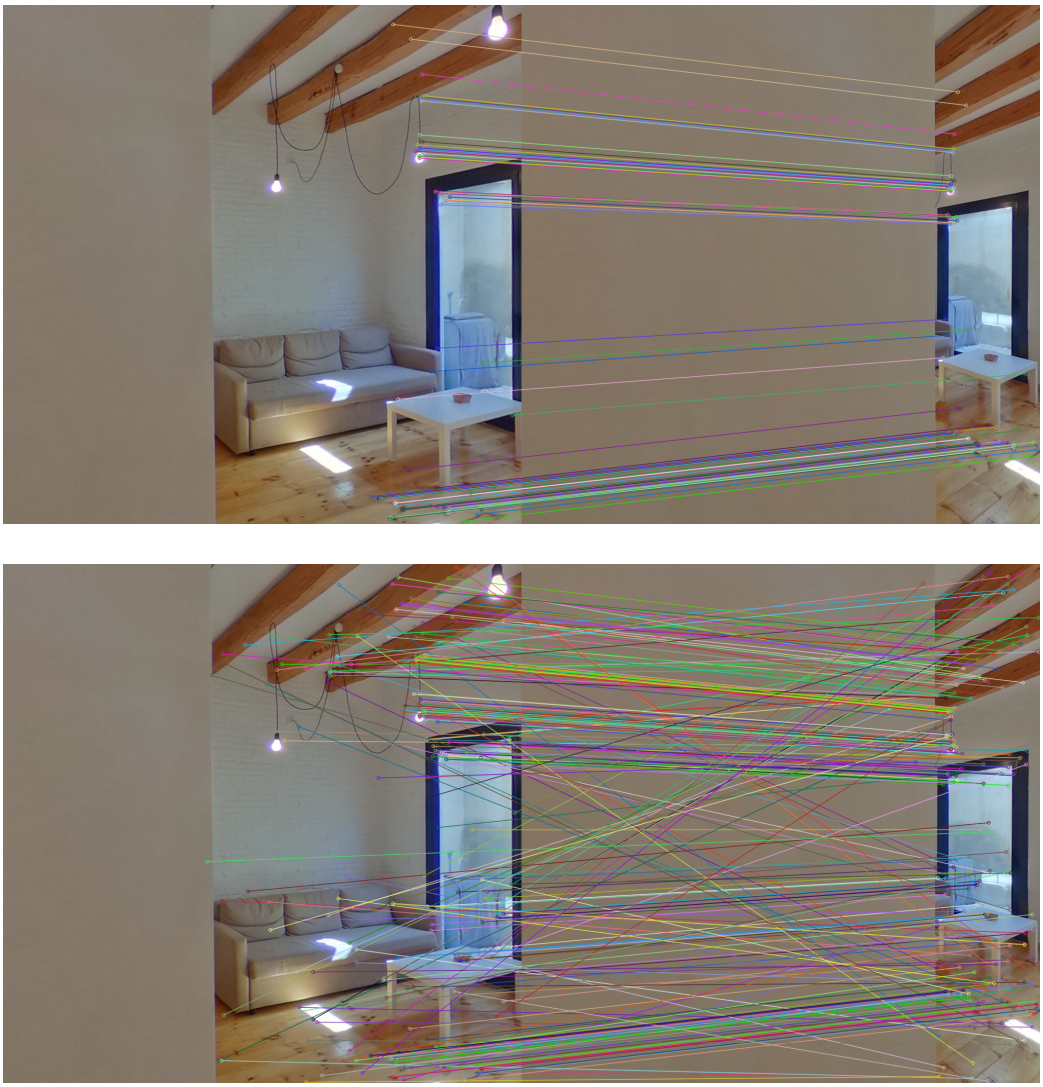

Figure 6.5: Visualisation of matches after essential matrix estimation. Top: non-normalised keypoint coordinates. Bottom: normalised keypoint coordinates. [Source: *Compiled by author*]

7 Pipeline Summary

In this sectiom we summarise the design of our pipeline. Our first specification of the problem was the following under the **Constraint 1** and **Constraint 3**:

$$\textbf{Input Type 3} \longmapsto (\textbf{Output Type 1}, \textbf{Output Type 3})$$

This corresponds to a solution accepting RGB spherical images taken in sparsely distributed locations in the interior scene space, and then outputting the relative positions of the space and an approximate planar 3D model of the scene space.

We then found that a structure-from-motion pipeline works under **Constraint 1** and strong **Constraint 3** and have the following specification:

$$\textbf{Input Type 3} \longmapsto (\textbf{Output Type 1}, \textbf{Output Type 2})$$

But this method is more suitable for highly detailed focused object reconstruction than an indoors reconstruction on the layout level.

In the paper *3D floor plan recovery from overlapping spherical images* [12] the authors describe a pipeline that solves a problem similar to the following under normal **Constraint 3**:

$$\textbf{Input Type 3} \longmapsto \textbf{Output Type 3}$$

However, it partially makes use of structure-from-motion and the **Output Type 1**. We can see the pipeline of this paper in the figure 7.1. The structure-from-motion is used as a sub-step of the first block on the left in the figure and outputs into the fourth block. Since this paper seemed to meet our input and outputs type requirements, we decided to implement a version of this work. As no implementation details were given by the authors, we decided to limit ourselves to the first block for TFG, which has the following interface:

$$\textbf{Input Type 3} \longmapsto \textbf{Output Type 1}$$

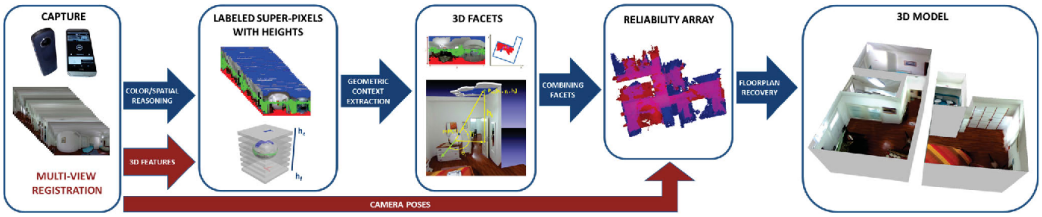

Figure 7.1: A possible 3D Model reconstruction pipeline. [Source: *3D floor plan recovery from overlapping spherical images* [12]

7.1 Pipeline Description

Our pipeline is similar to the procedure of the structure-from-motion in 3.1.3, but without the steps 5 and 6, as they should be taken care of by the other blocks in the extended pipeline[6]. Given the input set, our pipeline is:

1. Manually select two spherical images to be matched.
2. Apply region restriction, as concluded in experiment 6.3.
3. Apply ASIFT feature extractor using contrast threshold of 0.001, as concluded in experiments 6.1 and 6.2. It gives better results for slightly more execution time.
4. Apply FLANN kD-Tree feature matching, as concluded in experiment 6.4. It gives almost the same results for much lesser execution time.
5. Perform essential matrix estimation as described in section 4.2.
6. Decompose the result into relative camera positions and orientations.
7. Visualise the relative positions using directions for navigation.

With this pipeline we do not yet ensure the **Constraint 3** completely. Nevertheless we obtain a graph using the reconstructed pairs. In the future we can investigate how to apply *bundle adjustment* locally[7] on the vertices of this graph, so that we may satisfy the **Constraint 3** completely.

[6]The extended pipeline refers to the pipeline described in *3D floor plan recovery from overlapping spherical images* [12]. The implementation of the rest of the blocks falls outside the scope of this project.

[7]We would be applying the step 5 from structure-from-motion locally. Notice how this is similar to Visual Odometry. This is why we discussed that method in this document.

8 Reconstruction System

Once the problem specification was more clear we decided to implement a simple system that would be very simple and extensible, and solved part of the problem. We used the results and conclusions from the experimentation phase in the implementation of our system.

8.1 Architecture

At the start our objective was to develop an almost real-time reconstruction application for Android mobile devices. To this end we decided to use an existing library implementing generally high performance algorithms for reconstruction algorithms. We thought OpenCV to be a good fit, as it is implemented in the popular zero-cost abstraction but low-level language C++.

In fact it provides with Python API as well which was very helpful in the experiments. Thus we could rapidly experiment and prototype at the same time and all we would have to do is translate our gluing code to C++.

We spent little but some time into investigating compilation for android mobile architectures, but it soon became apparent to us that we would not have enough time to implement this application, because the experimentation phase had been prolonged.

Thus we decided to aim for a server-client architecture. The advantage of this is that we could easily provide frontend in both android and desktop devices, although we had to give up the almost real-time requirement. We had, though, already anticipated this failure to a degree, so we are not completely unhappy with this outcome, because we plan to continue focusing on this requirement later in the future. For now we decided to use the architecture illustrated in the figure 8.1. All of the actual code is available on GitHub: `https://github.com/strexicious/poc-reconstruction-pipeline`.

8.2 Backend API

As it is illustrated in the figure 8.1 we provide a REST API to the core reconstruction logic. However, we only implement the pipeline for feature extraction, feature matching and relative camera motion estimation, and not the steps themselves. The pipeline takes two equirectanulgar images and

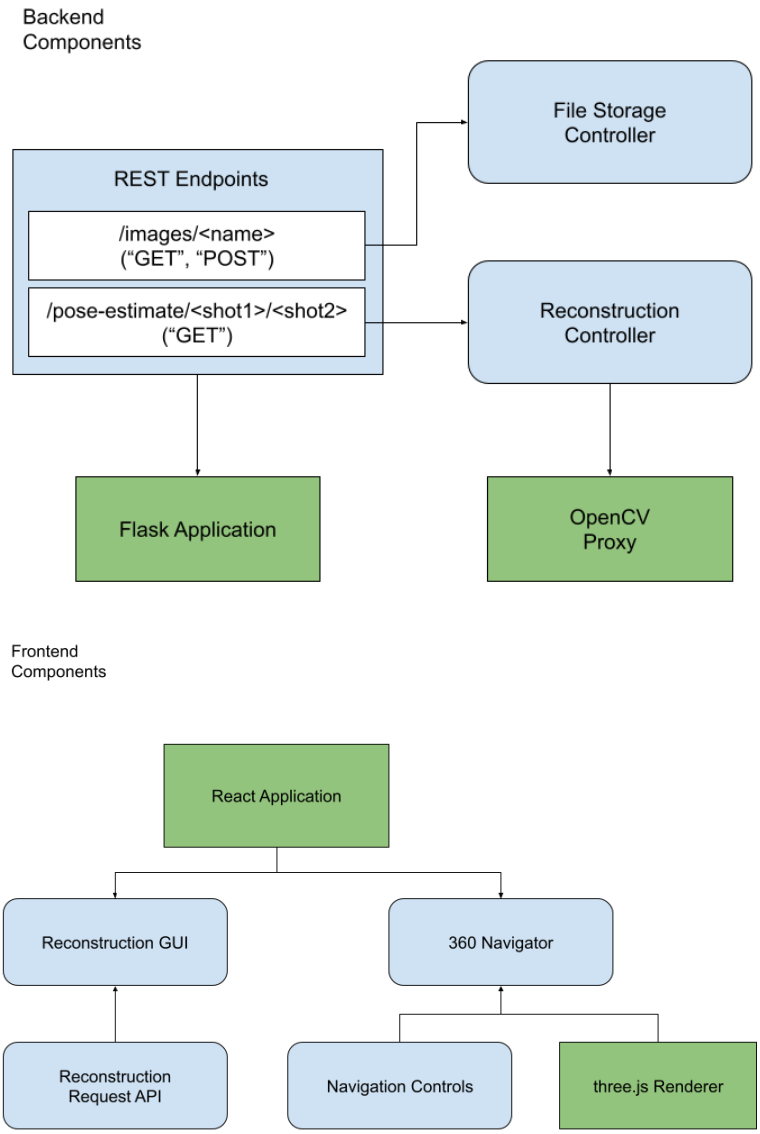

Figure 8.1: Reconstruction system architecture. Top: server components. Bottom: browser based client components. External dependencies: *Flask* [27], *OpenCV* [28], *ReactJS* [29], *three.js* [30] [Source: *Compiled by author*]

resamples one perspective image for each, with the FoV and region provided by the parameters. It applies the feature extraction, feature matching and relative camera motion estimation, taking into account the resampling parameters. The resulted camera motion is returned to the user and can be further processed manually.

8.3 Web 360 Navigator

In the client side we decided to implement a simple user interface using React framework. Its purpose is to provide simple interactions with the backend reconstruction subsystem. The important component of the client is the 360 Navigator, a simple graphics component to let the user explore the spherical images. The navigation environment is shown in figure 8.2.

The idea was to mimic live image capturing, so we instead let the user upload the images one by one. With this we solve the issue of ensuring that no totally unrelated images are being picked for a senseless reconstruction. Here we are delegating the responsibility to the user rather an algorithmic heuristic, to increase the accuracy of image matching. We think that with current implementation the user would be much more faster in doing this task than a computer. Though, in the future we would like to explore automated solutions as well. Machine learning methods may be suitable approaches for this task.

On the other hand we also wanted to let the user explore the reconstruction as she incrementally added the images. To this end we draw clickable arrows to indicate the connected images to the current one. In case there was some error in the reconstruction process, the relative camera motion, we also provide controls for the user to make modification after visually receiving feedback in the environment.

In the figure 8.3 we can see how the user is interactively able to parametrise the resampling of the equirectangular images into region-restricted perspective images. This is precisely the conversion from spherical to perspective that allows us to apply the reconstruction methods based on perspective models.

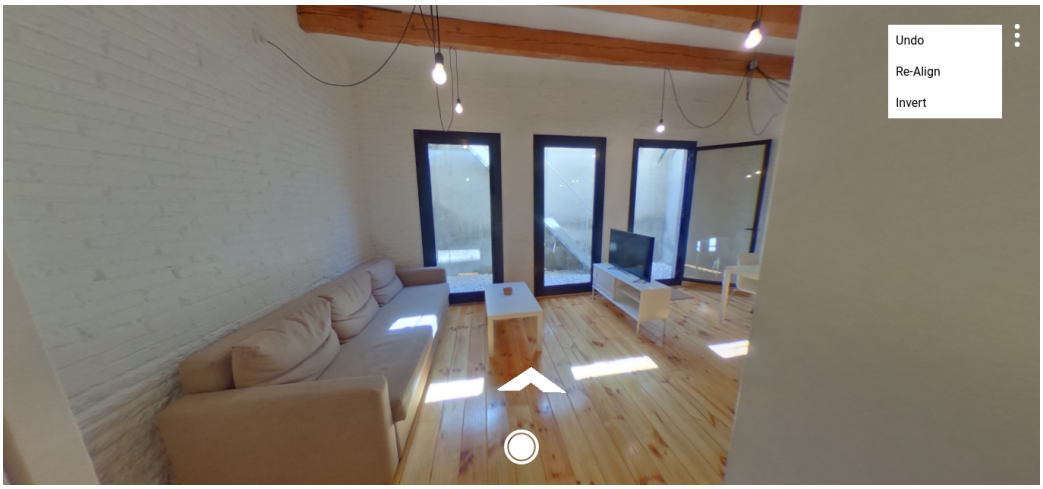

Figure 8.2: 360 Navigator using the reconstructed model. Center: equirectangular image visualiser and adjacent camera location visualiser. Top-right: reconstruction edit controls. Bottom-center: button to add images and connect to the current one. [Source: *Compiled by author*]

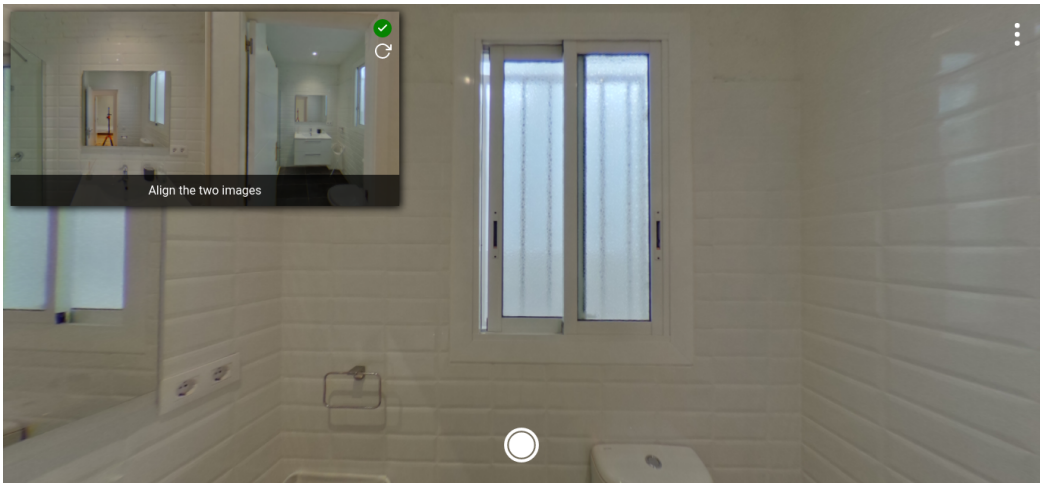

Figure 8.3: Top-left: interactive region restriction parameter controller. [Source: *Compiled by author*]

Android Issues

In theory we should be easily be able to adapt a browser based frontend to an android browser, and thus work on android devices already. But there are two preventing issues here: visually the interface is not adapted to mobile devices, and a small technicality that ordinary mobile devices today work with touch sensors, and not a device like a mouse, for which we have coded our controlling events.

8.4 Native Android System

Analysing the figure 8.1 we see we could easily merge the two client-server subsystems into one, due to their simplicity. The current architecture is very flexible to be able to make it work on many systems. But it has an implicit dependency of a network involved and many abstraction layers, both of which are source of significant overhead in execution time.

In the figure 8.4 we propose a native architecture. OpenCV is already distributed in highly optimised binaries, and if really necessary, we could reimplement bare minimum of the algorithms needed for the reconstruction system. The 360 navigation environment is currently implemented with three.js, which actually makes use of WebGL API for web browser. On android devices there is a native low-level alternative OpenGL ES that can be used directly, which also allows for significant more control and more opportunities for optimisations.

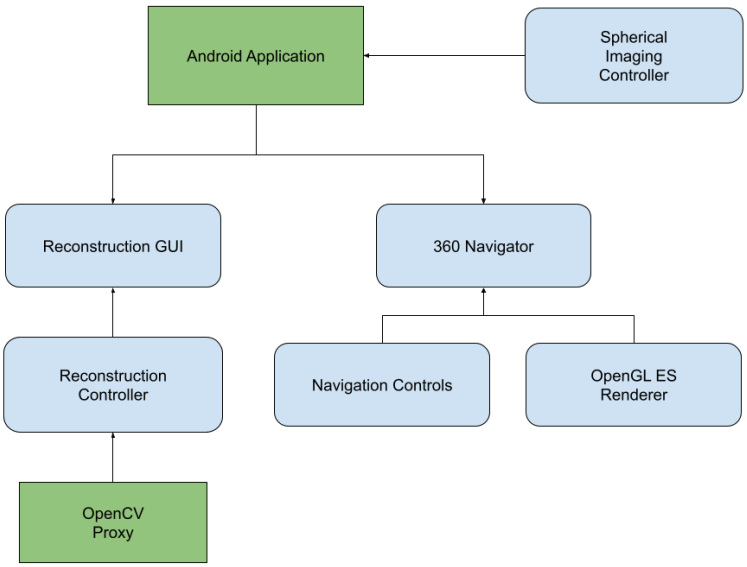

Figure 8.4: Native Android System Architecture. External dependencies: *Andoird API* [31], *OpenCV C++ API* [28], *OpenGL ES API* [32] [Source: *Compiled by author*]

9 Technical Competences

9.1 CCO1

To have an in-depth knowledge about the fundamental principles and computations models and be able to apply them to interpret, select, value, model and create new concepts, theories, uses and technological developments, related to informatics. [Low]

This project did not explicitly required the development or in-detail analysis of algorithmic properties of the solutions, thus we only used little algorithmic analysis under the a generic Turing Machine like model.

9.1.1 CCO1.1

To evaluate the computational complexity of a problem, know the algorithmic strategies which can solve it and recommend, develop and implement the solution which guarantees the best performance according to the established requirements. [Low]

We analysed very vaguely the feature matching techniques using asymptotic bounds to guide which and if they are worth implementing in an almost real-time system.

9.2 CCO2

To develop effectively and efficiently the adequate algorithms and software to solve complex computation problems. [High]

Throughout this whole document we have tried to emphasise on the development of almost real-time and approximately accurate system. Notice how we use the words "almost" and "approximately". This is because we are actually tackling a highly researched and difficult problem which holds much importance in the real world industry.

9.2.1 CCO2.2

Capacity to acquire, obtain, formalize and represent human knowledge in a computable way to solve problems through a computer system in any applicable field, in particular in the fields related to computation, perception and

operation in intelligent environments. [High]

It should be obvious how strong the connection of our desired system is with human vision and perception. Our problem inherently requires tasks related to intelligent detection of features in the images and be able to perform various types of heuristics, developed through human reasoning, to increase the accuracy of image matching. From matched images we use a formal model of camera instances to represent the relative motion in a computeable manner.

9.2.2 CCO2.4

To demonstrate knowledge and develop techniques about computational learning; to design and implement applications and system that use them, including these ones dedicated to the automatic extraction of information and knowledge from large data volumes. [High]

The methods and system studied in this project requires having a decent level of knowledge on extraction of information from high resolution equirectangular image data sets. The most similar computational learning method that we did employ in this project was the FLANN kD-Tree matcher. We have not explicitly talked a lot about machine learning approaches, but they could very well be better alternative methods to solving some of the steps.

9.2.3 CCO2.6

To design and implement graphic, virtual reality, augmented reality and videogames applications. [High]

The final designed system includes a navigation system which makes high use of computer graphics. This system does not support virtual reality hardware, but in concept it is does fall under that category and can be extended to work with virtual reality devices.

10 Project Management

In this section we include the original project planning and comment on the change of plans we had to make over time. This includes the task planning, resource planning and methodology planning. We also include the sustainability report.

10.1 Project Planning

For this project I will be dedicating 40 hours per week, give or take. Furthermore at the time of writing this document at the start of GEP, I expect to defend this thesis during the turn at the end of June. So approximately starting from 23rd of February until 25th of June gives a time period of 720 hours over 18 weeks to complete this project. Taking these facts into account, in the following I describe the planning for this project.

10.1.1 Task Definitions

Here I define the tasks in detail that are to be completed in order to finish the project as a whole. The intention is to completely describe the interdependency, length and work to be done for the tasks. As there are various kinds of tasks involved, they are also grouped at a high level.

Management Tasks

Good management can be the difference between the success and non-success. As such, first kind of tasks are related to the project management. Some of these are to be carried out in the initial phase, so that the project can be developed in an efficient manner and result in success. Thankfully the organisers of GEP already provide us with a blueprint for these tasks.

- **Context & Scope.** Give the context in which the project is developed and limit it's scope. Specify the objectives and the method of development. Contrast with existing solutions and justify. This will give a general overview of the project.
- **Project Planning.** Elaborate a plan for the project; the steps and tasks involved and how to execute these. Estimate the needed resources. This will serve as a rather concrete guide to follow during the development phase.

- **Budget & Sustainability.** Give an estimate of the budget and analyse the sustainability of the project. Inspect the tasks together with the whole project, and their needs, to calculate the budget. Study how the development phase and the completion of the project can impact sustainability. This is perhaps one of the important things to consider in a project but easy to overlook.
- **Final Report.** Refine the gatherings from the previous tasks and finalise them. Fix any mistakes or edges. Incorporate feedback from the directors and GEP tutor. This will document most of the management aspects of the project and should serve as a reference when making any management decisions further along the line (which is not uncommon).
- **Monitoring.** It is very important to keep the director updated in all of the processes. The director as the responsible and mentor figure can give advice and help during these processes so we will hold one meeting per week. To prepare for these I am to give reports twice a week on any sort of work done. A meeting with the codirector will also be arranged towards the end of May.

Research Tasks

There are two types of research tasks. There is some initial research that was done at the start to get familiar with the problem. That is, given a very general and vague description of the problem, find existing work done on it so to guide our own work.

The following research tasks exist to give a theoretical basis for the development part. A thorough understanding of the problem goes great way in solving it. The goal is to be able to describe problems rigorously and prove solutions with high certainty. They will provide with the understanding of concepts which then later can be used in writing the implementation of our system.

- **Mathematical Tools.** Describe basic tools needed from the areas of Linear Algebra, Calculus and Statistics. Elaborate on any advanced concepts and provide with examples.
- **Domain Description.** Using previous tools build a mathematical description of image processing, 3D modeling and reconstruction.

- **Problem Statement.** Restate the problem using the objects and properties from the previous domain description.
- **Solution.** Using the mathematical language describe a general solution. Describe the variations that can take place. Analyse these under some computation model.

Experimental Tasks

Once a theoretical basis is given, new hypotheses can be formulated. The problem exhibits a degree of complexity that may highly benefit from heuristics or any empirical data. The following tasks describe how to gather empirical data and conclusions that will help in writing a robust implementation of our system.

- **Hypotheses.** Give a set of hypotheses on the properties of a solution and its parameters.
- **Experiments.** Design and execute experiments that will let us confirm (or deny) these hypotheses in isolation without needing the final implementation.
- **Conclusions.** Collect the experimental data and give conclusions that can be used in evaluating the decisions of final implementation.

Development Tasks

With concepts well understood and a solution described in abstract, the development phase can start. The experimental data and conclusions may also be used during the implementation. The following tasks outline the general process of writing an implementation.

- **Architecture.** Specify the components and system's architecture.
- **Components.** Specify the interface of the components required.
- **Unit Testing.** Write a unit test suite for the components.
- **Implementation.** Write the implementation for the components. This includes any setup and debugging processes.
- **System Testing.** Test the system using real world data. This may involve going back to implementation tasks.

Documentation Tasks

These tasks are implicitly included in other tasks. All of the **processes, results and notes shall be collected and compiled** in a document as the project progresses. At the end of the project this document will be prepared for the oral defense and presentation will be based on this.

10.2 Resources

As part of this planning it is important to estimate the resources needed, so that later on we don't find ourselves unable to complete tasks due to lack of these. It is also important to not waste any resources either.

10.2.1 Human Resources

The main human resource needed for this task is myself. Mostly the project will be developed in isolation, but as the intention is to later on integrate with an existing system, some coordination with other teams might be necessary. Javier with the help of Rodrigo will be directing the project. Marcos will help in properly managing the project by providing feedback.

10.2.2 Material Resources

- A general PC will be essential for most of the tasks. For some of the tasks a powerful GPU may be necessary.
- An Android device might be needed if we decide to target the platform.
- An infrastructure for production, though there already exists one.
- Ricoh THETA SC 2 to take spherical images and a tripod.
- Physical books and notes for research.

10.2.3 Software Resources

For the technological stack, as it current stands it is expected that some components will require high performance and others reliability. The initial choice of languages are Rust [33] where performance matters and Python for high-level components. For image processing we will use OpenCV [34] (and

dependencies), a robust library with bindings to many languages. For 3D graphics we will be using modern OpenGL [35] (versions 3.3+ or ES 2.0+).

The target platform is yet to be decided, but it will either be Android or Web Browsers. So additionally for the client side we may use either:

- **Android:** Java or Kotlin — to be decided after experimentation.
- **Web Browsers:** HTML5 and JavaScript. To work with OpenGL ES 3.0, three.js [30] might be used (an abstraction over WebGL 2.0).

For documentation we will mostly make use of LaTeX and related tools, Texmaker being our main editor. We will use zoterobib [36] as the bibliography manager. We will make use of GanttProject software for the gantt chart.

For communication and scheduling Atenea, Gmail and Google Calender will be used. For code organisation Visual Studio Code will be used. Git will be our source version control system. Evernote may be used to gather notes.

10.3 Task Summary

All of these tasks are summarised in the table 10.1. For each task we give, the length is estimated by its subjective complexity at the time of writing. In the figure 10.1 this table is illustrated using a gantt chart (the dependencies are omitted as they may be partial).

10.4 Risk management

We need to evaluate some risks that could force us to deviate from this initial planning. In such cases we need to be prepared for alternative plans.

- **Insufficient results [Extreme Risk].** As a research and development project, this situation is not impossible. Our experiments may reveal that a chosen solution is not sufficient. In such case we will have to focus on researching different approaches and potentially give up on the development part. This should not require any more resources than already mentioned as we simply fallback to researching part.
- **Deadline of the project [High Risk].** Due to the inexperience in real world engineering we may have underestimated the complexity of

the problem here, and so the implementation could take longer than expected. In which case we will have to compromise by sacrificing some functionality and give a solution under a specified set of assumptions. For resources we would keep using the ones already mentioned but for an extended period of time.

- **Targeting Android [Medium Risk].** It is desirable to make a solution execute in the mobile Android devices. Due to inexperience with the Android platform it may take longer to write the final implementation. Another risk is the difficulty to optimise for mobile devices. In such cases we will have to settle with an implementation that can run on a more powerful computer using a browser, and optimisating for mobile may become a separate project, potentially developed with the help of an expert, which needs be taken into account as an human resource.

ID	Name	Hours	Dependencies	Resources
T1	Project Management			
T1.1	Context & Scope	24		PC, LaTeX
T1.2	Project Planning	12		PC, LaTeX, GanttProject
T1.3	Budget & Sustainability	16		PC, LaTeX
T1.4	Final Report	20	T1.1, T1.2, T1.3	PC, LaTeX
T2	Monitoring	22		PC
T2.1	Problem Research	80		PC, Search Engine
T3	Theoretical Basis			
T3.1	Mathematical Tools	16		PC, Notes
T3.2	Domain Description	28	T3.1	PC, Notes, Books
T3.3	Problem Statement	5	T3.2	PC, Notes
T3.4	Solution	25	T3.2, T3.3	PC, Notes, Books, Papers
T4	Experimentation			
T4.1	Hypotheses	5	T3.4	PC, Notes
T4.2	Experiments	35	T3.2, T4.1	PC, GPU, Notes, Books
T4.3	Conclusions	8	T4.1, T4.2	PC, Notes
T5	Development			
T5.1	Architecture	6		PC, Notes
T5.2	Components	18	T5.1	PC, Notes
T5.3	Unit Testing	25	T5.2	PC, Notes, Code Editor
T5.4	Implementation	160	T3, T4, T5.3	PC, Code Editor, Material Resources
T5.5	System Testing	32	T4, T5.4	PC, Material Resources
T6	Documentation	176	T1, T2, T3, T4, T5	PC, LaTeX, Notes

Table 10.1: Task summary: ID, name, length, dependencies, resources. Human resources implied as needed. Dependencies may be partial. Notes may be digital or in paper, taken from internet or self compiled, credited appropriately. Implementation and System Testing times take into account debugging processes. Documentation includes preparation of presentations. [Source: *Compiled by author*]

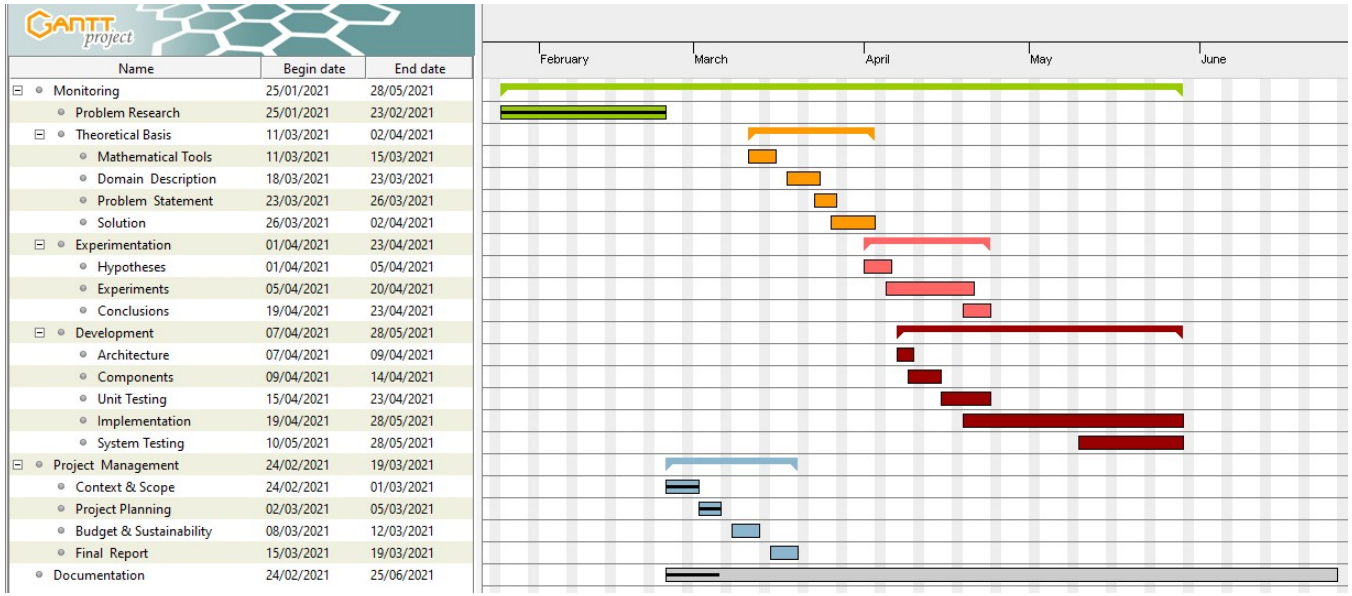

Figure 10.1: Temporal Planning Gantt Chart. The waterfall like organization indicates temporal sequence or dependency of tasks. Tasks overlap indicates the independent progression of tasks in parts. [Source: *Compiled by author*]

10.5 Budget

In the following we analyse the costs for the resources needed to carry out this project. The calculations are then increased to account for deviations and give a maximum budget estimated.

10.5.1 Staff Costs

In the planning section we discussed the human resources that will be needed. For budget estimate it would be preciser to use a more granular analysis of human resources. So the calculations will be based on different roles needed for different tasks. For each task we give the hours spent on it by a particular role. In the end for each of the role we sum total hours spent and multiply by the hourly cost that role is suppose to have. This finer analysis gives a very complete description, so that in case of deviations during the project, we can easily adjust calculations and take measures.

The tasks in this project will be distributed to five general roles. The first one is **Project Manager**, responsible for robust planning, coordinating the development and overseeing the progress of the project. Then we have the **Researcher**, responsible for contextualising the problem and making it concrete, as well as giving a complete description of possible solutions. Related to this we have the **Developer** role who is responsible for the design and implementation of a system solving the problem. Then we also have a **Tester** role, responsible for testing for correct functionality and usability of the system, discovering integration problems and evaluating the API. Finally we have **Technical Writer** for recording all of the processes and important aspects of the project in text.

The costs of these roles per hour are described in table 10.2. The hours dedicated per role to different tasks are described in table 10.3. In the same table we calculate the costs per activity and total staff costs (SC).

10.5.2 Generic Costs

The direct cost of materials include GPU and infrastructure (table 10.4). For the infrastructure we may need servers for hosting and deployment. The PC and Android devices were already available, so their direct cost is assumed amortized and negligible, and only incur electricity costs. The physical books and notes were also available along with the Ricoh Theta SC2 (camera) and

Role	Base Salary (euros)	Total Salary (euros)	Hourly Pay (euros)
Project Manager	2901.15	3771.5	21.43
Researcher	2509.59	3262.47	18.54
Developer	2532.27	3291.96	18.71
Tester	2599.87	3379.84	19.21
Technical Writer	1721.5	2237.95	12.72

Table 10.2: Role Costs. The base salaries are monthly. The base salary ranges were taken from [37] and averaged using $(min \times 0.8 + max \times 0.2)$. The total salary accounts for 30% taxes for social security ([38]). The monthly salary is divided into 176 hours. [Source: *Compiled by author*]

tripod, but their cost information is unavailable. For the camera and Android devices the indirect costs are negligible [39], so we will ignore these.

Electricity costs are result of energy consumption by the material resources. The information needed to do the calculations was extracted from [40], [41] and [42]. Then the formula is $n.\ of\ hours \times kWh \times 0.22\ euros/kWh$. These are summarised in table 10.5.

Travel costs only include metro tickets for me. Assuming that a ticket with 10 trips costs 11.35 euros [45], the equation that gives the average cost for my travels is: $\frac{11.35\ euros}{10\ trips} \times \frac{2\ trips}{day} \times \frac{2\ days}{week} \times 20\ weeks = 90.8\ euros$.

In total the generic cost (GC) sums up to be **2055.55 euros**.

10.5.3 Total Cost

Contingency Costs Our final estimate of our budget consists of the sum of staff and generic costs. But we have to account for any contingencies that could cause our estimate to be very tight. To avoid this we simply add a margin of 30% to the sum of previous two: $CC = (GC + SC) \times 0.3 = 4952.61\ euros$.

Incidental Costs In case of any risky situations take place (see 10.4), we may also give a prior cost estimate to these. For this we simply raise our estimation by 30% for infrastructure costs. The reason being that these costs increase by time, and we hope by adding more time (approx. two months) to the project would resolve those situations. In case we need an Android

ID	Task Name	Total Hours	Hours Distribution					Cost (euros)
			PM	R	D	T	TW	
T1	Project Management							
T1.1	Context & Scope	24	24	0	0	0	0	514.32
T1.2	Project Planning	12	12	0	0	0	0	257.16
T1.3	Budget & Sustainability	16	16	0	0	0	0	342.88
T1.4	Final Report	20	20	0	0	0	0	428.6
T2	Monitoring	22	22	22	22	22	22	1993.42
T2.1	Problem Research	80	10	80	0	0	0	1697.5
T3	Theoretical Basis							
T3.1	Mathematical Tools	16	0	16	0	0	0	296.64
T3.2	Domain Description	28	0	28	0	0	0	519.12
T3.3	Problem Statement	5	0	5	0	0	0	92.7
T3.4	Solution	25	0	25	0	0	0	463.5
T4	Experimentation							
T4.1	Hypotheses	5	0	5	0	0	0	92.7
T4.2	Experiments	35	0	0	35	0	0	654.85
T4.3	Conclusions	8	0	5	3	0	0	148.83
T5	Development							
T5.1	Architecture	6	0	0	6	0	0	112.26
T5.2	Components	18	0	0	28	0	0	523.88
T5.3	Unit Testing	25	0	0	25	0	0	467.75
T5.4	Implementation	160	0	0	160	0	0	2993.6
T5.5	System Testing	32	0	0	0	32	0	614.72
T6	Documentation	176	0	0	0	0	176	2238.72
Total		**713**	**104**	**186**	**279**	**54**	**198**	**14,453.15**

Table 10.3: Staff Costs. Left-most column is task ids. Right-most column is the sum of $role\_per\_hour\_cost \times hours$ for each role. PM: Project Manager, R: Researcher, D: Developer, T: Tester, TW: Technical Writer. For monitoring all roles participate to share the progress. The project manager is partially responsible for correct problem research. [Source: *Compiled by author*]

Resource	Price	Tasks
Infrastructure	1094.78 euros	T5.5
GPU	829 euros	T4, T5

Table 10.4: Generic Direct Costs. GPU cost from [43]. Infrastructure cost are server costs estimated in [44] (converted to euros) and reduced by 50% because this project does not constitute the whole of technological needs. [Source: *Compiled by author*]

Resource	Energy	Time (h)	Cost
PC	0.171 kWh	720	27.09 euros
GPU	0.278 kWh	227	13.88 euros

Table 10.5: Electricity Costs. [Source: *Compiled by author*]

specialist we also add a salary of 6150.18 euros to our estimate ([46]). The calculations are given by: $IC = 1094.78\ euros \times 0.3 + 6150.18\ euros = 7244.96\ euros$.

Final Estimation So our final estimation of the budget is described by the following equation:

$$\textbf{Budget Estimate} = \textbf{SC} + \textbf{GC} + \textbf{CC} + \textbf{IC} = \textbf{28,706.27 euros}$$

10.5.4 Management Control

As is the case for most real world projects, it is highly probable to not fulfill perfectly the initial planning of a project. But as is the purpose of management, we should be able to have as much control as possible so to take appropriate decisions in accordance to our goals of a project. In this section we describe control methods for expenses during the course of development of the project.

Since much of the cost is based on hours spent, we can keep track of how good our estimation actually is. Each time progress is made or change happens in relation to our planning, we can use the following two indicators to reevaluate our budget estimation, where EC are the costs expended at a given moment.

- As first indicator we use $SC + GC - EC$. That is to keep track of our budget if we have had no deviations in the planning of the project. So while the first indicator is positive we should not have to worry too much. If the the indicator turns negative and we are not clearly in emergency situations, then we have to appropriately take decisions to reduce our expenses as well as take into account expenses spent upto that point moving forward.
- As second indicator we use $SC + GC + CC + IC - EC$. This is used to control our maximum expense for the project. If the second indicator also goes into negatives then we have to put on hold the project and immediately reevaluate our planning and use new strategies taking into account all we have learnt upto that point.

If the (supposedly) normal planning without incidents is followed, then for the most part the indirect cost do not hold great importance, so much of the attention should be put into infrastructure costs. These can vary unexpectedly if not carefully managed. Usually the cloud infrastructure providers have a feature to easily monitor and setup alerts for these kind of situations, and to control these costs we will have to be sure that no task that can be run locally needs to be deployed to the live servers. For other types of contingencies and incidents we will simply use the reserves proposed in our budget estimate.

10.6 Changes in Project Planning

With respect to task planning there were virtually no changes. All the initially defined tasks were correct and sufficient to carry out this project. Except for the unit testing, which we did not find appropriate for a small system that we ended up with.

On the resource usage, initially we had over-estimated and anticipated for the needs of deploying the software to production. Though, the research phase ended satisfactorily, and we did develop a proof of concept system, we do not have an application for android yet, and thus no need to spend resources for infrastructure. This is not a failure, but a decision to avoid costs as we will further continue our research beyond TFG.

There were slight alterations in the timeline of the tasks during different phases, but they were handled by compensating the time management towards the end of development and sacrificing some of system testing phase, which shall not cause any harm, as we did not deploy it into production.

The temporal alterations (displayed in figure 10.2) were caused by initial insufficient results. That is why we had to expand our experimentation phase. In the end we were able to successfully find the results we wanted within the time limit we expected. And even so we do not have a fully native android application, the web system developed can be adapted to interface with an android device within a few hours. But, if we wanted a full android solution, and if it is possible, then we can expect it to take from 1-3 weeks more with the help of an experienced developer. This does not add to any costs as we had already highly anticipated the budget needs. A rough estimate for our final costs is then given by the following calculation taking into account that we did not make use of the infrastructure:

$$\mathbf{SC} + \mathbf{GC} - 1094.78\ \textit{euros} \approx \mathbf{15,282.15\ euros}$$

10.7 Methodology and Rigour

10.7.1 Workflow Methodology

Both for research and for development we will use the Kanban methodology. Kanban is designed with the goal of limiting the flow of work by the available capacity. But even with detailed planning it provides control when unforeseen changes appear. So it is appropriate for even a single person projects.

The word Kanban originates from Japanese and may literally be translated as "signboard". The key idea is that we have a board where we can visualise the tasks. The tasks are listed as notes in different stages and make for a visual indicator of progress. The tasks are pulled from one stage to other (the act of progression) at the rate given by the capacity available in each stage and so allows for balancing the focus to a particular bottleneck set of tasks. By focusing on individual tasks it allows for more frequent evaluation of priorities as opposed to Scrum sprints. So in a project with fair uncertainty, like in research and development, it is an appropriate methodology to adopt.

For this project we will use four stages for the tasks:

- **To-do:** Includes all those tasks pending. Will be updated as needed to break bigger tasks into small ones.
- **In progress:** All those tasks being worked or researched on.
- **Testing:** All those tasks which have been considered finished, but need proof reading or testing for correct development.
- **Completed:** All of the tasks finished and tested, and whose dependent tasks may continue their progress.

To apply the concepts from this methodology we will use some online service like Jira to create virtual task boards and configure them as described here. We may vary the stages a bit if needed but in general this template tends to be good choice for starters. An additional variation we will apply to this is that if we have to we will forcefully break a task into sub-tasks so that a unit of work is something that can be done at most in 12 hours.

As is typical of software projects the changes happen quite often. So we will want to use a version control system, particularly git, to manage active research and development. Git has the concept of branches, and without

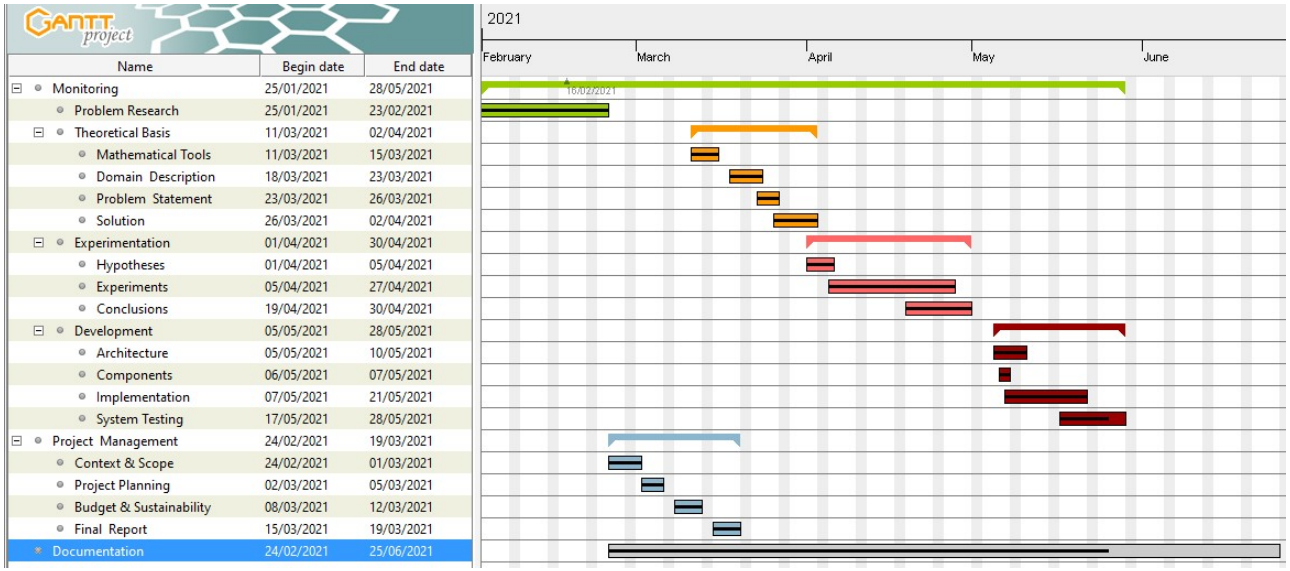

Figure 10.2: Altered Temporal Planning Gantt Chart. The waterfall like organization indicates temporal sequence or dependency of tasks. Tasks overlap indicates the independent progression of tasks in parts. [Source: *Compiled by author*]

digging for details, they help us to employ the Kanban strategy in practice very efficiently, while also giving tools to keep a clean history, so that at a later moment we can easily visualise the journey of the project and go through older versions of our code and documents for any reason we may want to. It also serves as a backup tool because it easily lets us maintain active clones of the history via online services like GitHub, GitLab, BitBucket, etc. or even self-hosted.

For communications many of these previously discussed services provide with comment sections, but emails are one of the oldest and reliable ways to communicate and update. So we will use a combination of these three to keep track and updated on the progress on different aspects of the project, from general objectives to small tasks.

10.7.2 Work Rigour

In spirit of true science and as computer scientists, the choice of using a scientific like method for measuring progress is adequate for a research project. In general we will want to observe a certain aspect of the problem, collect information that is needed to work on that aspect, study the information and apply conclusions in order to continue with this loop of observing and internalising knowledge. This can work for both the research and the development, because engineering builds on top of science.

10.8 Changes in Work Methodology

For the work methodology there were no changes in how it was employed. We following the Kanban strategy to manage our task units. In fact, it was quite helpful in carrying out our research in an organised manner. During the experimentation it helped us stay in control, as we had discussed originally. While we repeated the experiments and made modifications to the parameters of our solutions, it kept our progress in check, because we had organised control over the tasks in the Kanban board. Though, we did not use the **Testing** stage of our task units much because of time restrictions.

10.8.1 Validation

During the research, the validation we needed were to confirm that if a certain paper applied to our solution, or not. To this end the main aspects to speed

up the evaluations have been the studying the abstract and introduction of the papers and the experimental results.

During our experimentation we employed a manual strategy to keep track of the results. We prepared a set of image datasets that are part of our representative set of inputs. Formalising this process is more complex than the intuitive reasoning, and we do not see much value in doing so. So we simply chose a set of images that we think will lead to higher accuracy of a system overall when used with real data. Using this validation data we manually designed and executed the experiments, changing the parameters and the heuristics, and evaluated ourselves the accuracy by experience rather than a statistical test. We think for now this was a sufficient strategy, but in the future if we want to work on a bigger scale we need to develop much more practical yet robust methods.

10.9 Integration of Knowledge

For a functional computer system to be successful, it has to be built using knowledge from different subdomains of computer science. Which is precisely what my training these past four years has been about, and this project is a test of what I have learnt in Informatics Engineering. Following I summarise the bits of knowledge obtained in different courses that helped me through the development of this project.

Fundamentals of Mathematics (FM): To formally study, understand and compile the knowledge base, this was an essential source of tools. Intuition is enforced by good formalism and this course introduced me these topics of set theory. By using these I was able to rigorously manipulate mathematical expressions and use them to my advantage.

Computer Organization, Operating Systems, Computer Architecture (EC, SO, AC): Knowing how computers are structured and organised helped me think of an initial solution for Android. Even though I did not end up fully implementing it, the knowledge obtained in these courses can be used to build a performant system and program it at the lowest levels of abstraction, removing the overheads. They also gave me tools to measure the performance and debug the errors in such environments.

Data Structures and Algorithmics, Algorithmics (EDA, A): I used the knowledge obtained from these courses to organise the data in my

system and process it efficiently. They gave me the tools to conceptualise the algorithms and study them scale independently.

Mathematics I, Mathematics II, Statistics & Probability (M1, M2, PE): Without a doubt these taught the tools necessary for any engineering discipline out there. I used small part of graph theory, but mostly linear algebra and calculus from M1 and M2 to understand and formalise the concepts behind this project. Probability and statistics played an important role in describing the solutions that are useable for real world data filled with noise and uncertainty.

Introduction to Software Engineering, Programming Projects (IES, PROP): The tools obtained from these helped me design the final system and see them to their completion. They gave me a methodical way to carry out this process.

Graphics (G): This is one of the important piece in visualising the results of the reconstructions. Without the tools from this project, we would have no end result to see. Without seeing the reconstructions and and interacting with them, there would be no purpose for this project.

Computer Vision. (VC): This course provides with the essential tools needed to employ the solutions in practice. It provides with techniques that are needed for detecting features, so that they can be described. It provides for feature description, so that they can be robustly compared. It provides for feature comparison, so that they can be matched. Without the tools from this course, we would have an incredibly hard time solving the problem.

10.10 Laws and Regulation

There are no laws and regulations governing this project directly. The project will be made open source in the future and will be appropriately licensed at that moment.

Any research done and collected in this project was obtained from publicly available sources and appropriately cited and credited. The conclusions made on any proprietary software were mere act of formulating hypothesis and making guesses from the information presented publicly.

By using some of third party and open-source software, there are some laws that are applied indirectly. The project uses Python and Javascript. The

usage of the languages is not subject to any license, but programs processing these might be. Since we will be only distributing our code written in these languages, we do not have to provide the license notice for these. The same is true for using public interface of flask and OpenCV. We merely make use of their interface and inform the user of these dependencies (and others similarly). The user themselves will be responsible for obtaining a working copy of these and being able to execute this project.

We will be though, for convenience, redistributing MIT licensed three.js. The MIT license requires us to redistribute this license and credit the original authors, but do not hold them liable or be provided by any warranty, and then we can freely redistribute it with our project.

Another piece of software that we are using is the ex-patented SIFT algorithm implementation in OpenCV. Originally, this keypoint detector and descriptor was patent for over 2 decades, but it expired on 06/03/2020. For now we are not using any actively patented software and we do not intend to use it in the future in this project.

10.11 Sustainability Report

10.11.1 Self Assessment

After going through the survey and thinking about how unsure I was for some the questions posed to me, I realised that even though I had been taught to address the social, environmental, economical sustainability issues, I never did put in practice my training. Part of me wants to justify this by the fact that it is due to not having participated in many impactful projects, and while that maybe true, it does not excuse the fact that if I am to be a professional, I will have to look for these problems actively (as opposed to passively) and find solutions myself, even if I am not directly a main contributor in a project.

Another aspect I had not much thought about previously was how open source software contributed to the society. I always thought a software had social impact only through application level users. It made me realise that open source is also a form of social equity and diversity. Each new open source project presents new opportunities that can spread very widely thanks to the era of internet. It increases the chances of the project being useful by contribution of volunteers who get the chance to make impact. I am not

denying that this could also result in undesirable situations sometimes, but I understand much better now why recently big corporations make so many of their projects open source, even if in part.

I was also not aware that there were established methodical approaches to solving these issues (i.e. [47]). For example usage of indicators to measure and study better the sustainability of a project. It will require little practice and experience, but knowing that people have developed these methods that can guide us to improve our society and planet, even a little each day, then we should all try to make a difference for the better, because it is our duty and none others', as we are also the ones responsible for so many of the problems inadvertently, especially for life on earth.

10.11.2 Environmental Analysis

Project Put into Production

Have you quantified the environmental impact of completing the project?

It is very difficult to strictly quantify the environmental impact of this project. This project was very much digital based and so there is little direct connection with the environment. Quantifying the indirect impact is very overwhelming.

If you redid the project, would be able to complete it with less resources?

I think the resources we ended up using were already minimal. We did not even use the tech infrastructure we predicted to use, and so we think it would be hard to change the environmental impact even more.

Exploitation

What resources do you estimate to be used during the useful life cycle of the project and what is their environmental impact?

The resources needed will be the spherical cameras and general purpose computers. These devices are already heavily use today on common basis so we cannot analyse their impact, but we can say it contributes very little in comparison.

Globally, will the project improve or worsen the ecological footprint?

The project should in general reduce the footprint. It eliminates the need for physical travel, thus potentially reducing CO_2 emissions. Compared to these I would argue that the cost of charging your laptop to open a virtual tour is far less than of those aforementioned.

Risk

Can there be cases that cases that increase the ecological footprint?

We would argue that it is highly improbable, due to the nature of how little resources are needed. And the needed resources are very lightweight and digital.

10.11.3 Economic Analysis

Project Put into Production

Have you quantified the cost of undertaking the project? What decisions did you take to reduce these costs? Have you quantified these reductions?

The budget analysis in the section 10.5 covers the development of this project in isolation. In the project planning report we see our final cost and give and overview of how we reduced the costs.

Did the costs adjust to the initial predictions? Did you justify the differences?

Yes, the costs adjusted very well to our predictions. We also justified the small deviations, or rather savings, we had during the project.

Exploitation

What costs do you estimate that this project will have during its useful life cycle?

There should be a cost for the spherical cameras and other little costs for common used personal computers. These costs are individual dependent and so hard to analyse.

Will there be updating costs?

No, there should not be any usage costs for the rest of its life. But there might be some maintenance costs if required.

Risk

Will there be updating costs?

We do not think there is any scenarios that could affect the viability of the project economically.

10.11.4 Social Analysis

Project Put into Production

Has the project influenced in your personal, professional or ethical opinion of the people involved?

We think there that the pandemic made it very clear a good use case for a project like this one. We ended up developing up a digital solution for a problem which previously required in situ solutions undesireable during the pandemic, or less satisfying digital solutions.

Exploitation

Who will benefit from this project? Could it cause any damage to any collective of people?

This is an open source and publicly available project. As such, it benefits everyone. Its usage or existence do not cause any harm to anyone.

To what degree does the project solve the problem initially posed?

The project indeed does not fully solve the problem at a degree initially desired. But it solve an important part of it that is needed to solve the second part of the problem. It opens the opportunities for further research.

Risk

Can there be situation that would cause harm to a segment of population?

We hardly think this could cause any harm to anyone. If anything, its results should be useful for some people.

Could this project result in any dependency that would affect negatively its users?

The people who wish to use its results and are incapable of obtaining spherical images may not be affected negatively. Otherwise, people who wish to use this system as the sole form of sharing a virtual realistic experience can affect negatively by not providing a physically realistic experience.

11 Conclusions

We started this project with very basic knowledge of digital image processing and computer graphics. The main challenge for us lied in the study of automatic 3D reconstruction from 2D photographs. We immediately started our research by looking up existing methods online and thanks to today's internet, we found an enormous amount of information. But of course not all information is relevant. Some sources were completely unrelated and others were very specialised. We filtered through many sources and collected pieces of information on some general methods in order to understand the fundamentals of the problem.

We think our problem specification decomposition was detailed enough to simplify the analysis of different methods. All of the methods discussed in this document were quite relevant and gave us a good overview of the elements involved in this problem. We think we have only touched the tip of the iceberg and it really is a good beginning into the research of this problem.

This project introduced us to some new subjects of more mathematical nature than studied during the bachelor's degree programme. The epipolar geometry is a general framework which models very well the geometry of the 3D space and 2D images taken in that space. Though, we did not dive in the deeper details of the subject, that may have provided us with insights to improve our results. Overall we are quite satisfied with the knowledge we were able to collect on the subject and it can be further expanded in the future.

In this project we were able to explore beyond the basics of image processing and discover new types of feature processing in digital images. The viewpoint invariance is a very important property that we explored in this project. The similarity of results between Brute-Force Matching and well parametrised FLANN kD-Tree matching in this project surprised us a bit, and from now we will most likely not make the assumption of BFM being better without any empiric knowledge. We came across very basic and application agnostic match filtering of which we had no knowledge before.

We think the experimentation phase was not only successful but actually saved the whole project. It helped us design a system that has an acceptable accuracy. We are very happy with the user interface of our final but proof-of-concept system. It has very simple aesthetics but sufficient functionality,

the system is very interactive and provides good feedback allowing for some manual control. The virtual navigation environment is very easy to use and gives realistic sensation of exploration.

11.1 Future Work

One of the objectives of this project was the compilation of knowledge for further research. Thus we think we did gather a rich collection of knowledge that can be used in the future. Following are some of the topics that we think can be the focus of our research beyond the completion of this project:

- Experiment with the steps involved in reconstruction methods in detail.
- Extend current system to the reconstruction of planar 3D model.
- Do extensive study of multi-view geometry and multi-view stereo reconstruction methods.
- Use the alternative features in the feature processing steps. Mainly we used keypoint features, but we could instead consider holistic features. Features such as boundaries of the room, walls, floor and ceiling. See [48] for more information.
- Explore how to work directly on the spherical image geometry, rather than image planes.
- Research recent neural network based approaches and their impact in the reconstruction methods.
- Employ machine learning to automatically select pairs of images to be reconstructed.
- Research automatic region selection and restriction for feature matching. This could be a case of image segmentation.
- Currently we estimate the relative camera motion under the assumption that we move in the whole 3D space. But perhaps we could use the assumption that we only move and rotate on the horizon plane, as in, the plan spanned by the rays originating from camera center and the horizon line. Doing so may improve the accuracy of the essential matrix estimation. Under this assumption we may even be able to employ heuristics to further reduce our execution time.

References

[1] "The interpretation of structure from motion," *Proceedings of the Royal Society of London. Series B. Biological Sciences*, vol. 203, no. 1153, pp. 405–426, Jan. 1979. [Online]. Available: https://royalsocietypublishing.org/doi/10.1098/rspb.1979.0006

[2] "YouTube Spherical Videos." [Online]. Available: https://creatoracademy.youtube.com/page/lesson/spherical-video

[3] "Odometry," May 2021, page Version ID: 1024369947. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Odometry&oldid=1024369947

[4] "Visual odometry," Jun. 2021, page Version ID: 1026523739. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Visual_odometry&oldid=1026523739

[5] "Simultaneous localization and mapping," Jun. 2021, page Version ID: 1027469336. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Simultaneous_localization_and_mapping&oldid=1027469336

[6] V. O. M. Team | 05/15/2018, "What is Visual SLAM Technology and What is it Used For?" [Online]. Available: https://www.automate.org/blogs/what-is-visual-slam-technology-and-what-is-it-used-for

[7] "xdspacelab/openvslam," Jun. 2021, original-date: 2019-04-25T15:32:08Z. [Online]. Available: https://github.com/xdspacelab/openvslam

[8] "Structure from motion," May 2021, page Version ID: 1021867359. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Structure_from_motion&oldid=1021867359

[9] "AliceVision | Photogrammetric Computer Vision Framework." [Online]. Available: https://alicevision.org/#results

[10] "OpenSfM." [Online]. Available: https://www.opensfm.org/

[11] "State-of-the-Art in Automatic 3D Reconstruction of Structured Indoor Environments." [Online]. Available: http://vcg.isti.cnr.it/Publications/2020/PMGFPG20a/

[12] G. Pintore, F. Ganovelli, R. Pintus, R. Scopigno, and E. Gobbetti, "3D floor plan recovery from overlapping spherical images," *Computational Visual Media*, vol. 4, no. 4, pp. 367–383, Dec. 2018. [Online]. Available: http://link.springer.com/10.1007/s41095-018-0125-9

[13] D. Farin, W. Effelsberg, and P. H. N. de With, "Floor-plan reconstruction from panoramic images," in *Proceedings of the 15th international conference on Multimedia - MULTIMEDIA '07*. Augsburg, Germany: ACM Press, 2007, p. 823. [Online]. Available: http://portal.acm.org/citation.cfm?doid=1291233.1291420

[14] N. V. Oosterwyck, "Real Time Human Robot Interactions and Speed Control of a Robotic Arm for Collaborative Operations," 2018. [Online]. Available: http://rgdoi.net/10.13140/RG.2.2.28723.53286

[15] R. Hartley and A. Zisserman, *Multiple view geometry in computer vision*, 2nd ed. Cambridge, UK ; New York: Cambridge University Press, 2003.

[16] E. Boyer, "Projective Geometry: A Short Introduction Lecture Notes," 2016. [Online]. Available: https://core.ac.uk/display/103469580

[17] R. C. Gonzalez and R. E. Woods, *Digital image processing*. New York, NY: Pearson, 2018.

[18] W. Azarcoya-Cabiedes, P. Vera-Alfaro, A. Torres-Ruiz, and J. Salas-Rodríguez, "Automatic detection of bumblebees using video analysis," *Dyna (Medellin, Colombia)*, vol. 81, pp. 81–84, Oct. 2014. [Online]. Available: https://www.researchgate.net/publication/317498100_Automatic_detection_of_bumblebees_using_video_analysis

[19] R. Szeliski, *Computer vision: algorithms and applications*, ser. Texts in computer science. London ; New York: Springer, 2011, oCLC: ocn462920910.

[20] "Eight-point algorithm," Feb. 2021, page Version ID: 1007524552. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Eight-point_algorithm&oldid=1007524552

[21] "Direct Solution for Estimating the Fundamental and Essential Matrix (Cyrill Stachniss)." [Online]. Available: https://www.youtube.com/watch?v=zX5NeY-GTO0

[22] "RANSAC - Random Sample Consensus (Cyrill Stachniss)." [Online]. Available: https://www.youtube.com/watch?v=Cu1f6vpEilg

[23] "Scale-invariant feature transform - Wikiwand." [Online]. Available: https://www.wikiwand.com/en/Scale-invariant_feature_transform

[24] G. Yu and J.-M. Morel, "ASIFT: An Algorithm for Fully Affine Invariant Comparison," *Image Processing On Line*, vol. 1, pp. 11–38, Feb. 2011. [Online]. Available: https://www.ipol.im/pub/art/2011/my-asift/?utm_source=doi

[25] "k-d tree," Jun. 2021, page Version ID: 1027025564. [Online]. Available: https://en.wikipedia.org/w/index.php?title=K-d_tree&oldid=1027025564

[26] "VLFeat - Tutorials > KD-trees and forests." [Online]. Available: https://www.vlfeat.org/overview/kdtree.html

[27] "Welcome to Flask — Flask Documentation (2.0.x)." [Online]. Available: https://flask.palletsprojects.com/en/2.0.x/

[28] "OpenCV: OpenCV modules." [Online]. Available: https://docs.opencv.org/master/

[29] "React – A JavaScript library for building user interfaces." [Online]. Available: https://reactjs.org/

[30] "Three.js – JavaScript 3D Library." [Online]. Available: https://threejs.org/

[31] "Application Fundamentals." [Online]. Available: https://developer.android.com/guide/components/fundamentals

[32] "OpenGL ES - The Standard for Embedded Accelerated 3D Graphics," Jul. 2011. [Online]. Available: https://www.khronos.org/opengles/

[33] "Rust Programming Language." [Online]. Available: https://www.rust-lang.org/

[34] "OpenCV," Jan. 2021, page Version ID: 997730600. [Online]. Available: https://en.wikipedia.org/w/index.php?title=OpenCV&oldid=997730600

[35] "OpenGL," Mar. 2021, page Version ID: 1009883998. [Online]. Available: https://en.wikipedia.org/w/index.php?title=OpenGL&oldid=1009883998

[36] "ZoteroBib." [Online]. Available: https://zbib.org/

[37] "Función y Sueldo." [Online]. Available: https://tusalario.es/carrera/funcion-y-sueldo

[38] "Dime tu situación laboral y te diré qué impuestos deberás pagar," Aug. 2019. [Online]. Available: https://www.ennaranja.com/economia-facil/que-impuestos-deberas-pagar-trabajador/

[39] A. Kingsley-Hughes, "Here's how much it costs to charge a smartphone for a year." [Online]. Available: https://www.zdnet.com/article/heres-how-much-it-costs-to-charge-a-smartphone-for-a-year/

[40] "How much power does a computer use? And how much CO2 does that represent?" [Online]. Available: https://bit.ly/3ti3gbW

[41] "ASUS ROG GeForce GTX 1080 Ti Poseidon Review." [Online]. Available: https://www.guru3d.com/articles-pages/asus-rog-geforce-gtx-1080-ti-poseidon-review,8.html

[42] "Energy costs in Spain." [Online]. Available: https://www.expatica.com/es/living/household/energy-costs-108518/

[43] J. Penalva, "Nvidia GTX 1080 Ti, análisis: ¿cuánto mejora el rendimiento y fps de los juegos pagando 250 euros más que por la GTX 1080?" Feb. 2018. [Online]. Available: https://www.xataka.com/p/199176

[44] "Sponsored post: How much are startups spending for their top needs?" [Online]. Available: https://social.techcrunch.com/sponsor/brex/how-much-are-startups-spending-for-their-top-needs/

[45] "Preu targeta T-casual metro bus Barcelona | Transports Metropolitans de Barcelona." [Online]. Available: https://www.tmb.cat/ca/tarifes-metro-bus-barcelona/senzills-i-integrats/t-casual

[46] JobFluent, "Android Salaries in Barcelona." [Online]. Available: https://www.techsalarycalculator.com/android-developer-salary/barcelona

[47] G. Lami, F. Fabbrini, and M. Fusani, "A methodology to derive sustainability indicators for software development projects," in *Proceedings of the 2013 International Conference on Software and System Process*, ser. ICSSP 2013. San Francisco, CA, USA: Association for Computing Machinery, May 2013, pp. 70–77. [Online]. Available: https://doi.org/10.1145/2486046.2486060

[48] "Holistic 3D Reconstruction @ ICCV 2019." [Online]. Available: https://holistic-3d.github.io/iccv19/