Universidade do Minho
Escola de Engenharia

Miguel Jorge da Lomba Magalhães

Augmented Reality over Maps

Junho de 2020

Universidade do Minho
Escola de Engenharia
Departamento de Informática

Miguel Jorge da Lomba Magalhães

Augmented Reality over Maps

Dissertação de Mestrado
Mestrado Integrado em Engenharia Informática

Trabalho realizado sob orientação de

António Luís Pinto Ferreira Sousa
David Manuel Gonçalves Vaz

Junho de 2020

Copyright Notice

To my friends and family.

Statement of Integrity

Braga, 25th June 2020

Miguel Magalhães

Abstract

Maps and Geographic Information System (GIS) play a major role in modern society, particularly on tourism, navigation and personal guidance. However, providing geographical information of interest related to individual queries remains a strenuous task. The main constraints are (1) the several information scales available, (2) the large amount of information available on each scale, and (3) difficulty in directly infer a meaningful geographical context from text, pictures, or diagrams that are used by most user-aiding systems. To that extent, and to overcome the aforementioned difficulties, we develop a solution which allows the overlap of visual information over the maps being queried — a method commonly referred to as Augmented Reality (AR).

With that in mind, the object of this dissertation is the research and implementation of a method for the delivery of visual cartographic information over physical (analogue) and digital two-dimensional (2D) maps utilizing AR. We review existing state-of-art solutions and outline their limitations across different use cases. Afterwards, we provide a generic modular solution for a multitude of real-life applications, to name a few: museums, fairs, expositions, and public street maps. During the development phase, we take into consideration the trade-off between speed and accuracy in order to develop an accurate and real-time solution. Finally, we demonstrate the feasibility of our methods with an application on a real use case based on a map of the city of Oporto, in Portugal.

Keywords: Augmented Reality, Computer Vision, Geographic Information System, Maps

Resumo

Mapas e Sistema de Informação Geográfica (GIS) desempenham um papel importante na sociedade, particularmente no turismo, navegação e orientação pessoal. No entanto, fornecer informações geográficas de interesse a consultas dos utilizadores é uma tarefa árdua. Os principais dificuldades são (1) as várias escalas de informações disponíveis, (2) a grande quantidade de informação disponível em cada escala e (3) dificuldade em inferir diretamente um contexto geográfico significativo a partir dos textos, figuras ou diagramas usados. Assim, e para superar as dificuldades mencionadas, desenvolvemos uma solução que permite a sobreposição de informações visuais sobre os mapas que estão a ser consultados - um método geralmente conhecido como Realidade Aumentada (AR).

Neste sentido, o objetivo desta dissertação é a pesquisa e implementação de um método para a visualização de informações cartográficas sobre mapas 2D físicos (analógicos) e digitais utilizando AR. Em primeiro lugar, analisamos o estado da arte juntamente com as soluções existentes e também as suas limitações nas diversas utilizações possíveis. Posteriormente, fornecemos uma solução modular genérica para uma várias aplicações reais tais como: museus, feiras, exposições e mapas públicos de ruas. Durante a fase de desenvolvimento, tivemos em consideração o compromisso entre velocidade e precisão, a fim de desenvolver uma solução precisa que funciona em tempo real. Por fim, demonstramos a viabilidade de nossos métodos com uma aplicação num caso de uso real baseado num mapa da cidade do Porto (Portugal).

Palavras-chave: Mapas, Realidade Aumentada, Sistema de Informação Geográfica, Visão por Computador

Contents

List of Figures

List of Tables

List of Listings

1 Introduction

"Augmented Reality (AR) was first demonstrated in the 1960s, but only recently have technologies emerged that can be used to easily deploy AR applications to many users. Camera-equipped cell phones with significant processing power and graphics abilities provide an inexpensive and versatile platform for AR applications."
— Schmalstieg et al. (2011)

This chapter introduces the subject of Augmented Reality applied over maps. Moreover, we present the motivation and the goals behind this work. Finally, the document structure is described.

1.1 Motivation

An analogue map has several physical limitations. To name a few, the amount and depth of information, its size constraints, and the inherent readability issues. In order to overcome these issues, we can use Augmented Reality [1] capabilities to display multiple layers of information over a *raw* map. This allows us to interact with the map in ways that were previously inaccessible, *e.g.* by getting detailed information about features of interest, extract several layers of information (*e.g.* distance, landmarks, weather forecast), extended information of points of interest such as the area of lakes (Klinkenberg and Rhind, 2020), among others.

Modern advancements in Geographic Information System [2], visualisation and computer science have made traditional cartographic information ubiquitous. As a consequence, maps and paper-based information have evolved from the analogue to the digital world. A common trend behind this evolution is the constant drive to improve the delivery of information to the end-user.

1 Allows the user to visualize in real time the physical world with an overlay of digital information (Azuma, 1997).
2 Used to collect, analyze, and present information describing the physical and logical properties of the geographic world Shekhar et al. (1997).

Due to the digitalization of geographical visualisation, several groundbreaking innovations democratized the usage of maps. Among these, the smartphone is the most impactful resource for worldwide scaling of such technology. The reasoning behind this dominance is due to two factors. At first, its portability and affordability makes it of common existence among most individuals in a connected world — reaching up to 88% in developed countries and having skyrocketed in many emerging economies in the last years (Poushter et al., 2016). Secondly, the massive processing capabilities of modern devices allows for advanced visualization software that were prohibited until now.

The process of digitalization has been studied extensively, leading to the field of digital cartographic information. Amongst related existing research fields, Augmented Reality is one of the most prominent. The rationale is that such techniques deliver a clean and direct experience to a user, via a visual overlap of extended information on real data (map). The range of existing solutions is extense, and involve technology giants such as Google and Apple (Waldman, 2016). For the sake of completeness, it is relevant to mention two distinct types of maps, the precise and the emphasized. Precise maps are accurate representations of the real world, with accurate scaling and dimensions — mostly visible as cartographic representations. On the other extent, another range of applications is on the augmentation of non-realistic maps with emphasized landmarks, such as attraction map of amusement parks. In such cases, maps are not an accurate cartographic representations of the represented area, yet they are designed in a way to be easy for its users to navigate (Turchi, 2011). Such application of AR to the domain has been already used and shown to improve the customers satisfaction rates (Jung et al., 2015). This is supported by Ghose and Huang (2009), which have demonstrated that the increased availability of modern technologies enables businesses to facilitate quality enhancement through the personalization of its services and products.

However, current technologies applied to the massification of touristical information are, to some extent, limited. Additionally, such solution is of high practicality and usefulness, as it allows for (1) reduction of physical information leaflets; (2) updated real-time touristic routes, events and listings; (3) personalized advertisement; and to some extent (4) an improved touristic experience leading ultimately to higher expenditure on the host country.

With that in mind, the objective of this dissertation is the research and improvement of state-of-art methods for augmented touristic information, via the combination of AR and GIS. The crossing of the two is still on its early days and has not yet been fully explored. Due to the wide extent of potential applications (refer to Schmalstieg et al. (2011) for a review), in the context of this study, we will focus specifically on the application of museums, fairs and expositions with coordinate-based (precise) maps. Nevertheless, the discussed methods are general and may have implementation in other fields, to name a few, in the retail world (store locator), logistics (warehouse orientation) and large cultural events (music festivals).

The detailed schedule of this work is presented in Appendix A.3.

INDUSTRY COLLABORATION The work that follows was developed in a business environment at Ad Evolutio, Lda (Evolutio), a software house specialized on geographical systems. With the company collaboration, this work aims to provide a proof of concept through an end-to-end client-ready product for the final customers. For brevity, details of this industry collaboration are omitted.

1.2 Scope and Objectives

The main goals of this dissertation are: (1) to provide a study of available alternative solutions, identify their (dis)advantages and applications; (2) study AR applications in other scientific areas with common applicability to the current topic of cartography covered; (3) to detail the development of an easily-expandable software solution, allowing for the addition of new features as separate modules and a linear development of different User Interfaces (UIs); and (4) to detail the application of the developed software to a concrete case study, making use of all the features available in the solution.

The implementation effort can be detailed as follows: we provide a solution that displays augmented cartographic information over 2D maps, on both physical printouts and digital screen maps. In order to correctly position the augmented data, we compute the transformation that matches the coordinates of the marker plane to the frame. Moreover, this solution aims to be a generic solution for most of the use cases previously mentioned, with an emphasis on modularity. Our work stands out from existing methods due to following:

- At first, we focus on providing functionality expansion, that facilitate the addition of new features to the platform without the need to reformulate its core;
- Secondly, we allow the map provider to have a minimum amount of maps for its clients. For the sake of comparison, in common solutions, an interactive map can only be operated by one person at a time (Musatov et al., 2001). However, with our AR system we extend this number to everyone with a camera-equipped device with visibility to the map;
- Another feature that regular interactive maps do not possess is the ability to display different objects simultaneously according to the user selection; such objects can range from roads, markers, regions, among others. We aim to provide this capability;
- Finally, our methods allows for fully interactiveness in any type of map — physical or digital — *i.e.* it is possible for one to apply it on a 2D map, without the need to customize the maps beforehand;

1.3 Document Structure

The document is organized as follows. Chapter 2 reviews AR map applications, both on research papers and on commercial products, thus creating a comprehensive survey of the technologies available. This review follows with state-of-art technologies in the areas of AR and GIS, together with its implications on the developed product. Chapter 3 follows with an analysis of the solution challenges, the technologies that should be the support of the AR system, and the design of the modular approach. The detailed development and implementation of the system is described in Chapter 4. Finally, a prototype will be delivered with an application of AR over 2D maps enriched by extra information, not directly displayed on the map, as detailed in Chapter 5. The information displayed through AR includes dynamic AR information, which is interactive from the user perspective. Chapter 6 draws the final summary and conclusions together with viable future work.

State of the Art 2

In the following chapter, we discuss the related work and technologies to the solution. First, we perform a study of relevant solutions developed previously in the field of AR merged with maps. We analyze each solution independently and discuss its advantages and disadvantages. Additionally, we study distinct approaches to different problems and emphasize the most relevant ideas in the context of our problem. Secondly, the involved technologies in the solution development are also analyzed, which includes Computer Vision (CV), AR, Computer Graphics (CG), GIS, cartography and their combination, from both a technical and User Experience (UX) standpoints. Finally, a summary follows with the conclusions and detailed decisive factors on the selection of the methods used in the development.

2.1 Related work

In this section, we discuss existing **AR maps** applications, relevant in the scope of our project.

APPLE / GOOGLE AR MAPS In recent years, technology giants such as Google and Apple brought to the market location-based AR maps (Waldman, 2016; Chang, 2019), where a video-stream, captured from the device's camera, together with the users' geographic position is processed. From this, interest points are displayed on the screen merged with navigation directions to the users' selected destination, if one is configured (Mulloni and Schmalstieg, 2012). This process is illustrated in Figs. 1 to 3.

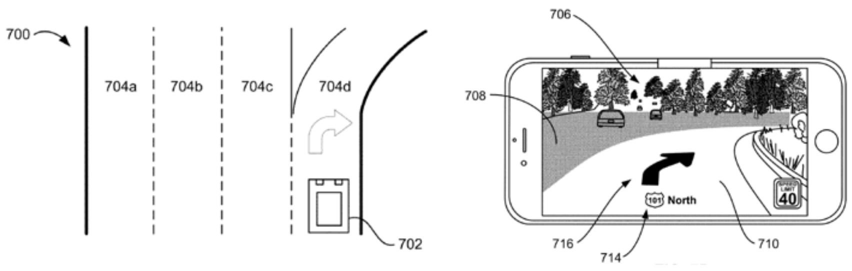


Figure 1: Apple's navigation AR map for cars. Source: Chang (2019)

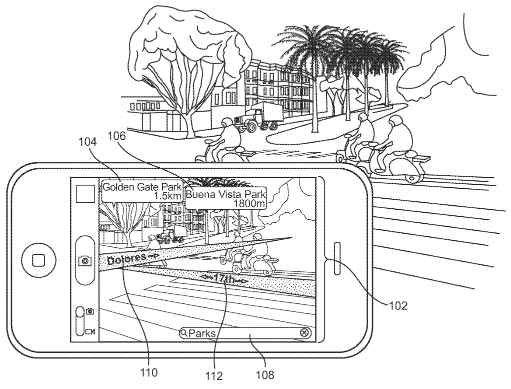


Figure 2: Apple's AR map with labels. Source: Waldman (2016)

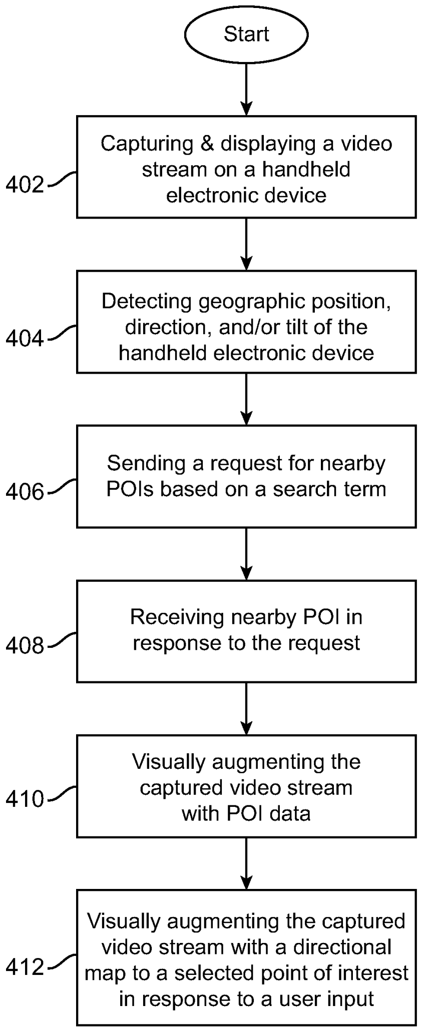

Start
402
Capturing & displaying a video stream on a handheld electronic device
404
Detecting geographic position, direction, and/or tilt of the handheld electronic device
406
Sending a request for nearby POIs based on a search term
408
Receiving nearby POI in response to the request
410
Visually augmenting the captured video stream with POI data
412
Visually augmenting the captured video stream with a directional map to a selected point of interest in response to a user input

Figure 3: Apple's AR map diagram. Source: Waldman (2016)

MAPLENS Developed by IPCity, MapLens (IPCity, 2020) is an application for Symbian phones equipped with camera and Global Positioning System (GPS) (Fig. 4). The functioning is the following: as users look at a paper map through the camera of a mobile phone, the IPCity application augments a map with digital information overlaid on top (Jobst et al., 2012). For example, photos that are connected to certain map locations or the users' current position.

Although possibilities are very vast, there is only one study using this application as an environmental awareness game which combines content from both physical and virtual realities (Morrison et al., 2011). Nevertheless, other possible applications were discussed, such as using bus stops maps that can be augmented with dynamic and personalized digital information.

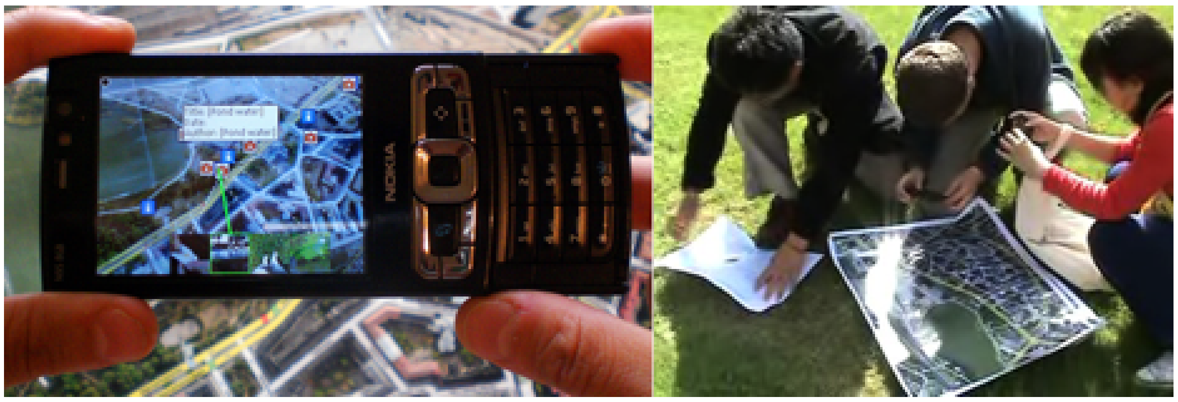


Figure 4: MapLens augments a paper map with icons and labels in live video stream. Left) MapLens user interface. Right) Participants playing an environmental awareness AR game. Source: Morrison et al. (2011)

MAP TORCHLIGHT: A MOBILE AR CAMERA PROJECTOR UNIT Paper-based maps are extensively used in the field of AR due to its printable high-resolution, large-scale adoption by the community, zero power consumption compared to its electronic counterparts, and easiness of information spread at close-to-zero cost per unit. The most common visual augmentation on paper maps is the visualisation of the augmented objects on the device (*phone*) where the image is being captured with. However, this is not the only solution. As an example, Schöning et al. (2009) demonstrates a distinct approach that uses a projector alongside a camera from a mobile device, as illustrated in Fig. 5.

The main advantage of this platform is the ability to add overplayed information over the paper map. Yet, concurrent use of this platform is not possible since this solution can only display a single layer of dynamic information at a time. In addition to that, this solution does not work on outdoor conditions during the day, due to its high sensibility to brightness. Moreover, the need for specialized equipment is also a big throwback in terms of possible user adoption of the platform.

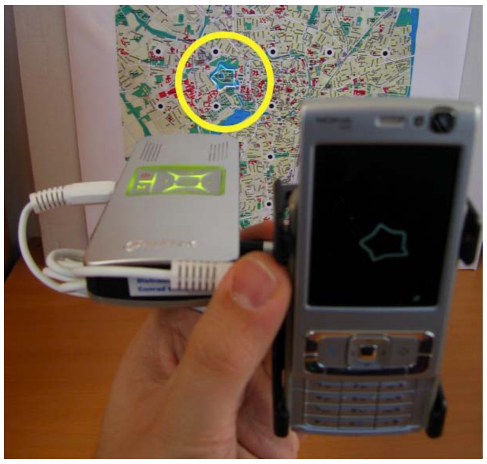


Figure 5: Prototype of Map Torchlight unit highlighting the castle of the City of Münster, Germany. Source: Schöning et al. (2009)

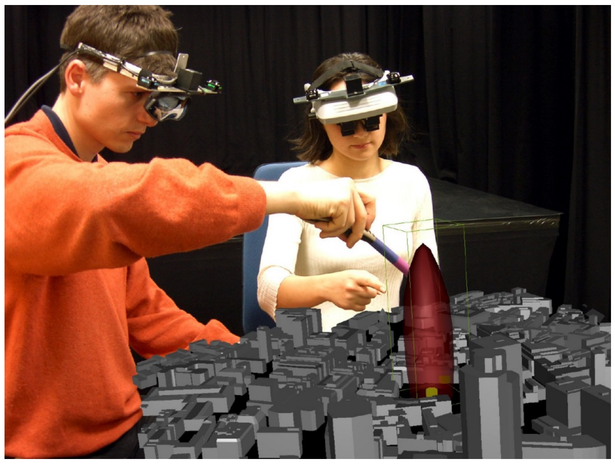


Figure 6: ARTHUR: Viewing and manipulating shared virtual objects in a round table. Source: Moeslund et al. (2003)

ARCH: AUGMENTED REALITY FOR ARCHITECTURE DESIGN The field of architecture takes advantage of AR technologies. Several applications have been demonstrated, applying AR to architecture. To name a few:

- **ARTHUR** (Fig. 6): A Collaborative Augmented Environment for Architectural Design and Urban Planning, used for urban planning and to help the architects' decision-making process making use of Head Mounted Displays (HMDs) together with markers on a plain surface (Broll et al., 2004; Moeslund et al., 2003)

- **ARUDesigner**: Augmented Reality-based Urban Design System, used the same HMDs together with markers (Wang et al., 2008)

- **ARchitectureView**: An Augmented Reality Interface for Viewing 3D Building Information Models, which allows to display architectonic models of buildings, specifically Building Information Model (BIM) (Belcher and Johnson, 2008)

Following all these developments, a new set of solutions called **ARch** (Augmented Reality for Architecture Design) was developed (Lopes et al., 2014; Mendonça, 2014; Miguel, 2014). These solutions bring a new functionality of visualizing and working with three-dimensional (3D) models using the integrated camera from a tablet (Gomes, 2015). Moreover, they were designed to be used by architects during the design process, without the need to have physical scale models (Costa et al., 2017). ARch is composed by these distinct solutions:

- **SeeARch** combines knowledge from several disciplines, including History of Architecture and the City, Building Construction Technologies, Architecture in the broadest sense, among others and enables mobile users to recognize the facade of specific buildings and, in real-time, superimpose relevant associated 3D and multimedia information (Raposo et al., 2017)

- **ARch4models**: a tool to augment the physical scale model, developed in order to improve the experience architects and designers have with physical scale models whilst making use of AR (Costa et al., 2017)

- **ARch4maps** (Fig. 7) recognizes image features on a city paper map and overlays digital content related to the relevant buildings of the city (Gaspar et al., 2016; Mendonça, 2014)

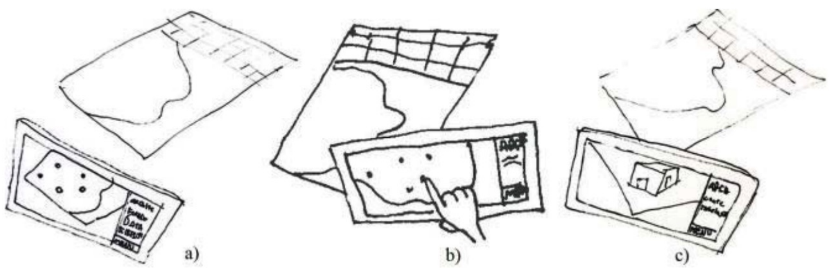


Figure 7: ARch4maps's user experience. a) Map identification; b) Click on building's point; c) Overlay the 3D model on the map. Source: Gaspar et al. (2016)

LOCATION-BASED INDOOR NAVIGATION In Paucher and Turk (2010), an AR system with six degrees of freedom (DoF)[1] has been demonstrated on mobile devices. However, it yields a poor frame-rate and limited usability (Sanches et al., 2019). The approaches for visual augmentation of a scene fall into the following categories:

- **Object recognition**, which uses object recognition to provide information about the recognizable objects viewed by the user, but does not supports AR when there are no special objects in the scene – used in this dissertation as described in Chapter 4.
- **Pose computation**, which determines the six DoF to allow the projection of any 3D virtual object into the image the user is viewing at any location) - the method used in Paucher and Turk (2010).

This application also enables the extraction of the location (and the orientation) from an image with a precision of about 10 to 15 centimeters, as pictured in Fig. 8.

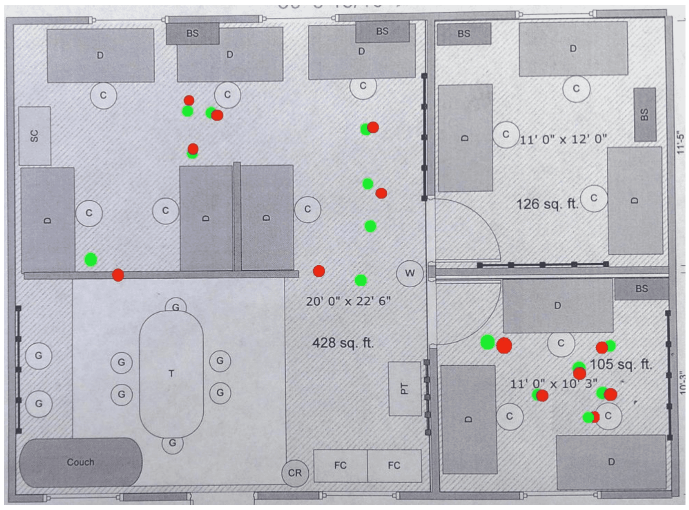


Figure 8: Indoor localization extracted from images (green points are the correct positions and red points are the computed positions). Source: Paucher and Turk (2010)

VR AND AR LOCATION-BASED GAME: POKÉMON GO In recent years, we have observed the release of a multitude of pervasive Mixed/Merged Reality (MR) games — see Montola (2011) for an exhaustive review. Among all the released games, a clear winner stands out: Pokémon Go, a location-based AR game based on the Pokémon franchise — "In Pokémon GO, the players are Pokémon (pocket monsters) trainers who search, capture,

1 Refers to the freedom of movement of a body within a 3D space.

collect, train, evolve, and battle Pokémon creatures. GPS is used to match the player's real-world location with the virtual world. When Pokémon creatures appear in the virtual world, AR is used to overlay the Pokémon on the real-world viewed through a mobile camera. The player's goal is to capture the Pokémon by throwing Poké Balls at them." (Paavilainen et al., 2017).

This technology is illustrated in Fig. 9. This is by itself a very distinct location-based AR application type, and demonstrates that there are numerous possible applications combining AR and GIS.

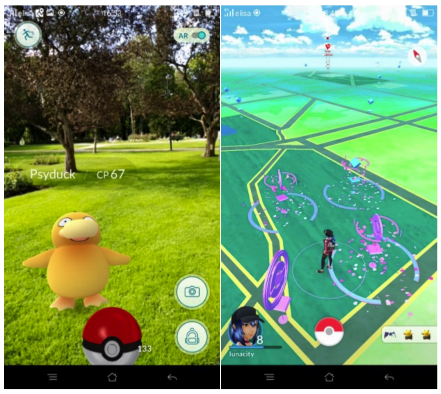


Figure 9: Capturing a Pokémon (left) and moving in the virtual world (right) Source: Paavilainen et al. (2017)

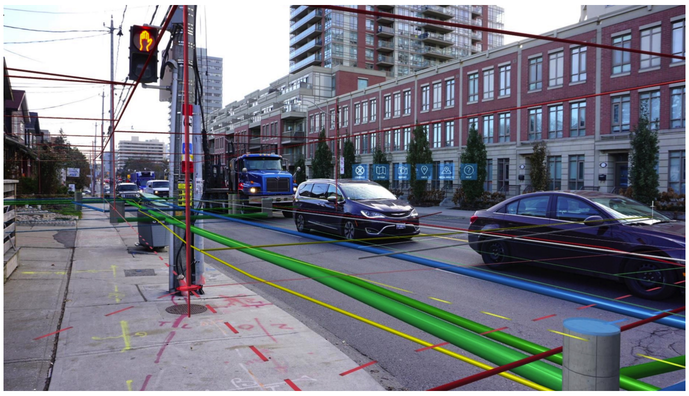


Figure 10: vGIS sample screen with augmented infrastructure. Source: vGIS and Promark-Telecon (2018)

VGIS vGIS is a platform for enhanced utility management with integration of GIS and AR (vGIS Inc., 2020). Beneath the pavement of the streets of our cities and villages, there is a vast number of different networks, from electricity, gas, sewer, water, telecommunication, among others. To visualize all these networks, Meemim Inc. developed a visualization platform for spatially-tagged data, which was designed to augment the workers' view by displaying underground assets as holograms (Pestov, 2018) — as exemplified in Fig. 10.

With this solution, the productivity of the workers was boosted by approximately 10% per job occurrence (vGIS and Promark-Telecon, 2018), which confirms the vast potential of visualization technologies in the locate industry in general. In 2018, Microsoft recognized this solution as an innovative partner (Microsoft, 2018).

2.2 Technology

We have shown in the previous section that existing solutions are vast yet limited and/or outdated. With this in mind, in this section we analyze the required technologies that constitute our solution. We compare them with the previously presented solutions, discuss each concept and its characteristics, and provide a breakdown of advantages and disadvantages.

2.2.1 Augmented Reality

Augmented Reality (AR), also referred to as Enhanced Reality (ER), allows the user to visualize, in real time, the physical world with an overlay of digital information, yielding a *hybrid* visualization of the merge of both real and enhanced world (Fig. 11). Quoting Azuma (1997): "Augmented reality enhances a user's perception of and interaction with the real world. The virtual objects display information that the user cannot directly detect with his own senses.". According to the same source, an AR system needs to fulfill three characteristics:

1. Combine both real and virtual worlds;
2. Be interactive in real-time;
3. Be registered in 3D;

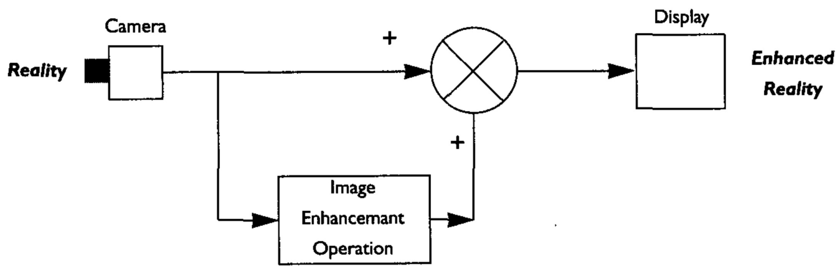


Figure 11: A general model of an AR system. Source: Bowskill and Downie (1995)

Augmented Reality is a variation of a Virtual Reality (VR) / Virtual Environment (VE) (Azuma, 1997). The main difference is on the imaging worlds that are presented to the end user. While on a VR environment the user is immersed on a digital experience (only) without connection to the surrounding reality (Rheingold, 1991). On AR, the user's reality is extended rather than replaced (Kipper and Rampolla, 2012). Well known examples of AR and VR wearable devices are Google Glass (Google, 2020) and Oculus Rift (Oculus, 2019), respectively (Tepper et al., 2017).

Additionally, Augmented Reality also differs from Mixed/Merged Reality (MR), which is a hybrid form of AR and VR. On a MR environment, it is possible to interact with both physical and virtual environments. In 2016, Microsoft released the MR device HoloLens which enables: real-world image augmented visualization; interactive functionalities provided by voice (speech recognition) and virtual representations of objects that react to gestures such as touch, spin and zoom; and extended navigation features with perception of depth, color, and spatial awareness (Evans et al., 2017; Microsoft, 2020).

For clarity, illustrative schematics of the differences between the three modes are presented in Figs. 12 and 13.

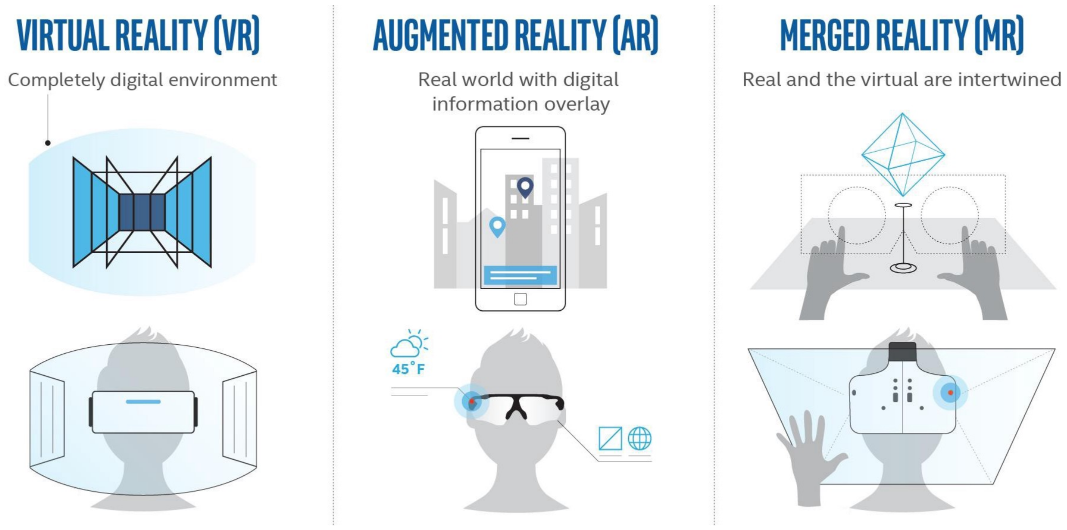

VIRTUAL REALITY (VR)
Completely digital environment
AUGMENTED REALITY (AR)
Real world with digital information overlay
45°F
MERGED REALITY (MR)
Real and the virtual are intertwined

Figure 12: Comparison between virtual, augmented and mixed reality systems. Source: VR Tech Today (2018)

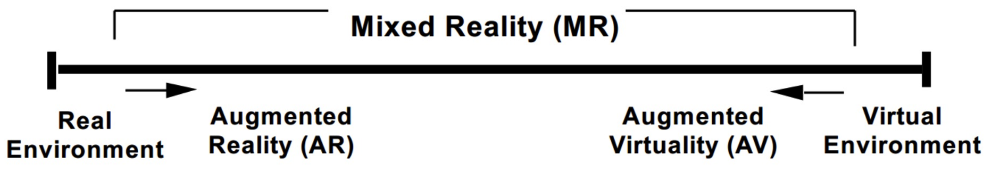


Figure 13: Reality-Virtuality (RV) Continuum. Source: Milgram et al. (1995)

Following the discussion of the reality systems, we now present the most relevant available technologies for Augmented Reality. We start with the description of existing types of AR (Patkar et al., 2013; Henrysson and Ollila, 2004).

MARKER-BASED AUGMENTED REALITY Based on one or more markers which are used to calculate the geometric transformations (translation, rotation, scaling and shear [Narayanan, 2018]) of a camera, in which the origin is the center of a marker itself. This category can be divided into the following sub-categories (Patkar et al., 2013):

- **2D barcodes**, which use square markers consisting of black and white data cells, that can be computed to obtain a final value of a marker (Gupta et al., 2018). This kind of markers are required to remain fairly simple for stable error correction (Patkar et al., 2013). An example of an application is ARUCO (Fig. 14) developed by Munoz-Salinas (2012) (Romero-Ramirez et al., 2018; Garrido-Jurado et al., 2016);

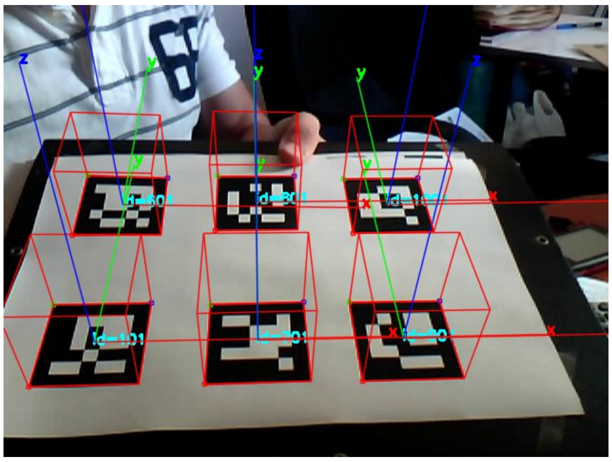
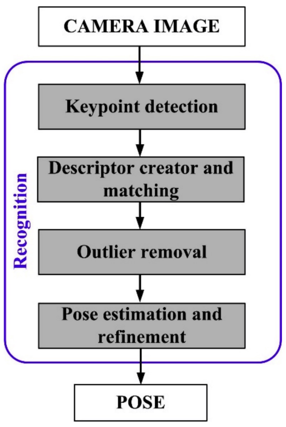
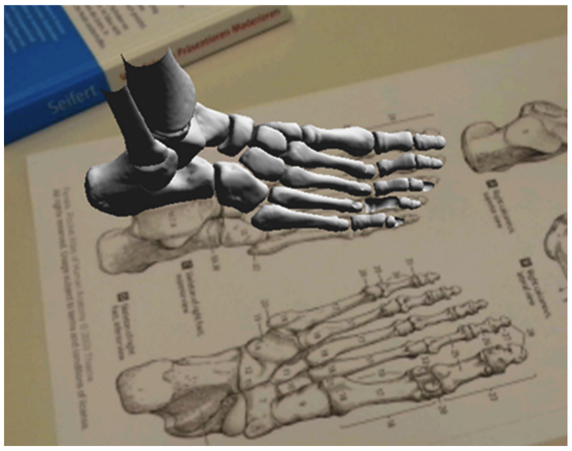


Figure 14: ARUCO markers. Source: Munoz-Salinas (2012)

- **Natural Feature Tracking (NFT)** (Neumann and You, 1999), via image-based markers. Detects and tracks the features that are naturally found in the image (Ćuković et al., 2015), such as corner, edges, blobs, etc. (Gupta et al., 2018). This technology is of high relevance in the context of our project, as it will be later adopted to detect and track maps in our solution. The pipeline workflow and an augmentation example are depicted in Fig. 15.


CAMERA IMAGE
Recognition
Keypoint detection
Descriptor creator and matching
Outlier removal
Pose estimation and refinement
POSE

(a) NFT pipeline. Source: Ćuković et al. (2015)



(b) Augmented scene with NFT. Source: Ćuković et al. (2015)

Figure 15: NFT pipeline and augmented scene.

MARKERLESS AUGMENTED REALITY Based on different types of sensors — such as GPS, gyroscopes, cameras, hybrid vision, accelerometers, and others (Azuma, 1997; Azuma et al., 2001) — that allow the creation of a AR environment (Figs. 16 and 17). As a main feature, it is not limited by the visibility of a marker (Henrysson and Ollila, 2004). Some examples of this includes the ones discussed above such as Apple / Google AR Maps and VR and AR location-based game: Pokémon Go.

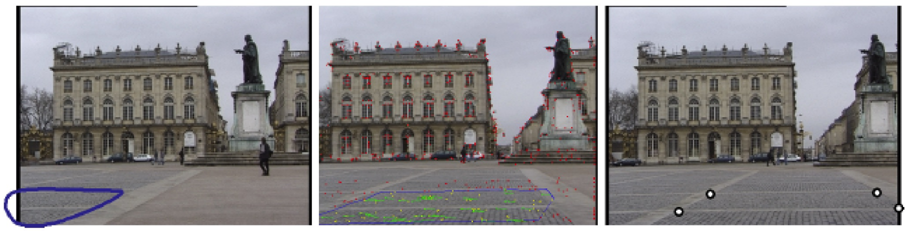


Figure 16: Steps in the algorithm: 1) The plane to be tracked is roughly indicated with the mouse. 2) Feature detection and tracking. 3) Setting the coordinate system. At no point after 1), the camera needs to be stationary, nor tracking interrupted. Source: Simon et al. (2000)

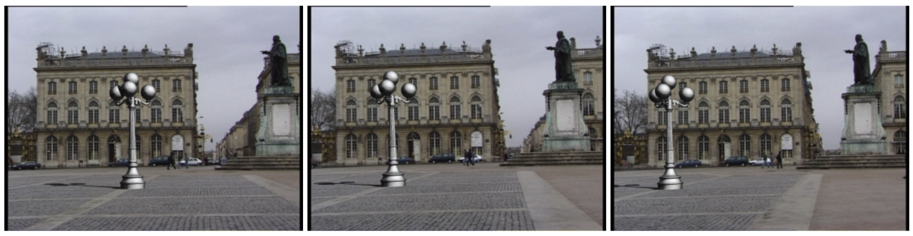


Figure 17: Augmentation results in Stanislas Square (France). Source: Simon et al. (2000)

After presenting the existing types of Augmented Reality, we present the most relevant currently available tools in the following paragraphs. A comparison table will follow in Table 1.

ARKIT In 2017, at the Apple Worldwide Developers Conference (WWDC), ARKit (Apple, 2017) was introduced as the *de facto* framework for AR apps for iOS (working on both iPhone and iPad) (Dilek and Erol, 2018). It performs both marker-based and markerless (Apple, 2018) Augmented Reality. Some of its main features are: people occlusion (AR content passes behind and in front of people in the real world); multiple face tracking (up to three faces, applied in Memoji and Snapchat); and live collaborative sessions between multiple people. Despite its advantages, it is a closed-source solution and not available outside Apple's ecosystem, which is a major drawback for multi-platform development and user adoption.

ARCORE Complementary, Google released ARCore (Google, 2018) in 2018, based on Project Tango (Bhagat, 2016), released in 2014, with fewer features that are not dependent on specialized cameras, thus allowing a larger number of devices to be compatible (Voinea et al., 2018).

VUFORIA Vuforia (PTC, 2019) acquired by PTC, allows for the development of Augmented Reality apps with advanced vision and recognition of a range of everyday images, objects, and environments (Voinea et al., 2018). It is closed source, yet allowing free usage with some limitations (Marto, 2017).

WIKITUDE An all-in-one AR Software Development Kit (SDK) including: (a) object & scene tracking; (b) Natural Feature Tracking (NFT); (c) geo-based AR; (d) multiple targets; and (e) extended tracking beyond recognized target, among others (Wikitude, 2019). It is a cross-platform system that works on smartphones and tablets across Android, iOS, Windows, and smart glasses. Additionally, it supports a variety of development frameworks such as Flutter, Cordova, Unity Xamarin and more (Wikitude, 2020). However, it is closed source and provided as a paid service (Marto et al., 2018).

	ARKit	ARCore	Vuforia	Wikitude
Developer	Apple	Google	PTC	Wikitude
Last Update	19 Sep 2019	28 Oct 2019	10 Dec 2019	13 Nov 2018
Platforms	iOS	Android, iOS	Android, iOS, Win	Android, iOS, Win
Free	Yes	Yes	Limited	No
2D Marker	Yes	Yes	Yes	Yes
3D Marker	Yes	No	Yes	Yes
Markerless	Yes	Yes	Yes	Yes
SLAM	No	Yes	Yes	Yes
GPS	Yes	Yes	No	Yes
IMU	Yes	Yes	Yes	Yes

Table 1: Comparison between AR frameworks (last updated in 28 Dec 2019).

2.2.2 Geographic Information Systems

The previous subsection (2.2.1) described technologies to detect an image and displayed AR content over the real scene. Here, we discuss the different types of data which can be displayed on top of maps.

According to Shekhar et al. (1997), "Geographic information systems are used to collect, analyze, and present information describing the physical and logical properties of the geographic world. Geographically referenced data is the spatial data that pertain to a location on the earth's surface.". Moreover, citing the same source, a typical GIS is composed of four major functional units:

1. **data input unit**, responsible for entering information into the system;
2. **data model**, which is a data abstraction that hides the intrinsics of data storage;
3. **data manipulation**, allowing operations on the data, e.g., distance calculation;
4. **result presentation**, which presents results visually.

The following subsection focuses only on the latter, *i.e.* on how the geodata is presented since it is the only unit relevant to the scope of our project. Usually a GIS presents the results visually in the form of maps, consisting of graphic images composed of raster data (Shekhar et al., 1997) with geodata overlayed. The geodata consists of spatial objects made up of points, lines, regions, rectangles, surfaces, and 3D volumes, with an underlying time-index, when available. It is also common to supplement it with non-spatial attribute information such as feature names, descriptions, photos etc. (Samet, 1990, 1995).

From here, we extract two distinct types of data in a Geographic Information System: (1) map tiles and (2) overlay data, described next.

MAP TILES

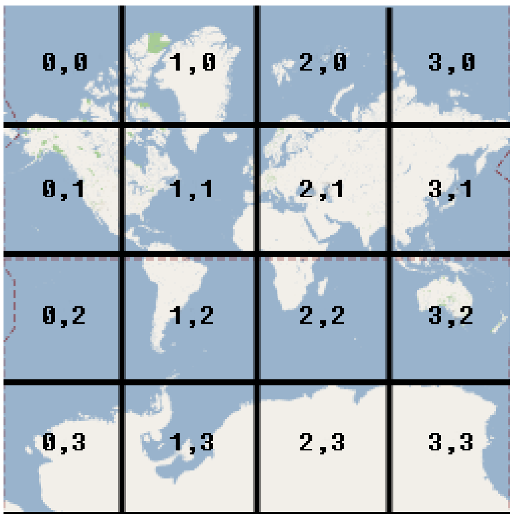
> "**tile** (noun, often attributive) - a flat or curved piece of fired clay, stone, or concrete used especially for roofs, floors, or walls and often for ornamental work"
> — Merriam-Webster Online (2019)

In this section, we describe **map tiles**, which have their nomenclature based on the ordinary "construction" tiles due to their concept similarities.

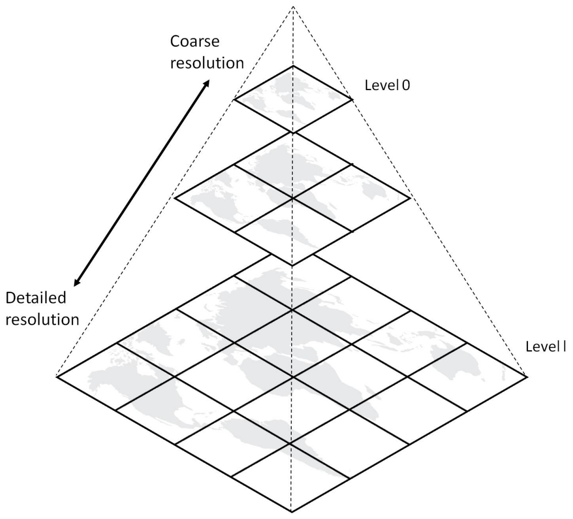
In order to display a map, it is common to divide it in smaller, usually squared, parts that together form the entirety of the map. These tiles are usually represented by a **"XYZ" scheme** (Fig. 18) where the "Z" refers to the **zoom**, while the "X" and the "Y" refer to the tile coordinates (Ramsey, 2019) — not to be confused with the geographical coordinates latitude and longitude. This is the *de facto* industry standard.

As a side note, this standard is not unique, as there are more tiling systems *e.g.* adopted by Microsoft's Bing Maps (Schwartz, 2018), employing *quadkeys* for addressing; or TileJSON, a lightweight JSON description of all the parameters associated with a web map, created by Mapbox (Mapbox, 2012).

The rationale behind the tiling system is to allow the end-user to have a lazy downloading, in which it only downloads the tiles that are actually being shown whilst the rest is delivered



(a) Tile coordinates in a map. A tile is a rectangular subsection of the *XY* coordinate region. Tiles are equally sized and equidistant. Source: Google (2019).



(b) Tile pyramid representation. Source: García et al. (2012)

Figure 18: Map tiles represented by a *XYZ* scheme.

when the position or zoom is changed. However, for partial display of a tile, the whole tile content is required to be downloaded.

Tiles are typically categorized as raster tiles or vector tiles. **Raster tiles** are composed by square images, they are suitable for satellite/aerial imagery[2], and provide low requirements for end-users hardware. These tiles are commonly stored as GeoTIFF (Ritter and Ruth, 1997) and JPEG2000 (Rosenbaum and Taubman, 2003) image formats. On the other hand, **vector tiles** (Mapbox, 2016), a more recent approach, are based on vector representation, instead of pixels. Therefore, provides reduced data size, allows for faster generating time and are easily customizable (Martinelli and Roth, 2016). Since it only stores basic geometries rather than pixel data, this allows for zoom without pixelated results. However, this approach requires more processing power from the end-users in order to display. Additionally, it is not suitable to display satellite imagery[2] (Map School, 2020).

Despite their differences, it is common to combine both approaches. For example, raster tiles with an aerial / satellite view together with vector tiles with only the labels in the user's language (Fig. 19). This technique allows having just one set of raster tiles and several vector

2 Images of Earth or other planets collected by imaging satellites.

tiles in different languages (Janak, 2019). Moreover, this adds the possibility of loading a different language without the need to load the raster tiles, and the other way around.

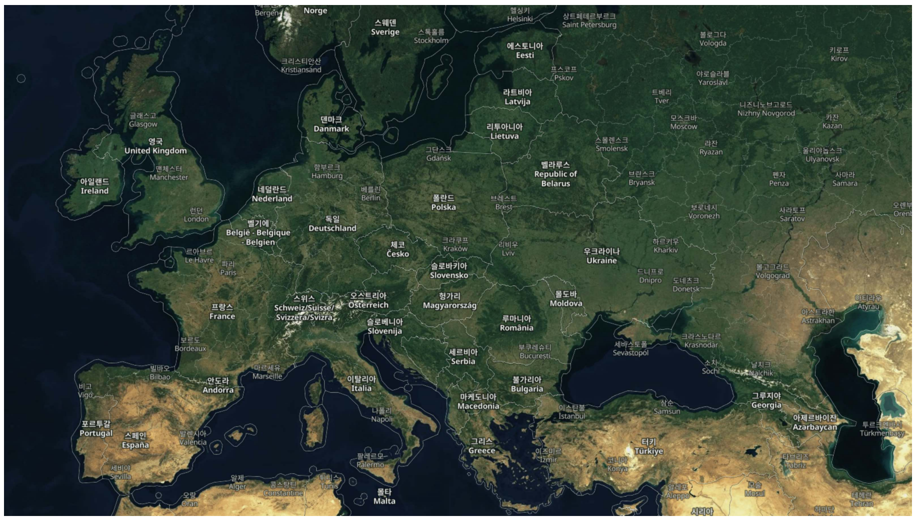


Figure 19: Raster tiles of satellite imagery of Europe overlayed with vector tiles of Korean labels. Source: Janak (2019)

GEODATA OVERLAYS There are several data types capable of representing of spatial elements to overlay data over a map. In the scope of our work, we will focus on:

- **vectorized polygons, lines, and points** which can be used to define areas, paths, and markers, respectively;
- **3D models** for example to display buildings.

In the current work, we focus specifically on the vectorized spatial elements. Traditionally, the most established vector format is the Shapefile (ESRI, 1998), which is limited by a 2GB filesize limit, non human-readable binary format, among others (Map School, 2020). Recently, GeoPackage (Yutzler, 2018) was developed to be the new alternative for Shapefile, but it is still non human-readable and more suitable for large datasets (Rashidan and Musliman, 2015). Nevertheless, modern technologies, particularly web applications, lead to the wide adoption of other formats. The most common is the GeoJSON (Butler et al., 2016) (Geographic JSON), based on the JavaScript Object Notation (JSON) syntax (Charollais, 2013), and adapted to the geographical context and shapes. It allows to describe the geometry: *Point*, *LineString*, *Polygon*, *MultiPoint*, *MultiLineString*, and *MultiPolygon*. For example, the point 10°06'00.0"N

125°36′00.0″E, represented in Fig. 20, is described in GeoJSON in Listing 1. Another possible format to represent this type of data is Keyhole Markup Language (KML) (Consortium, 2015), based on Extensible Markup Language (XML) (W3C, 2013). The representation of the same *Point* in this format is in Listing 2.

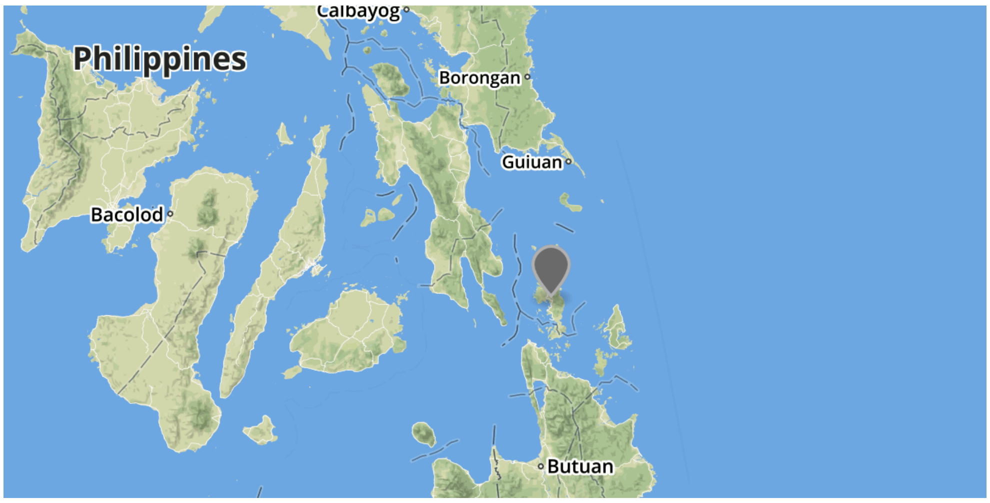


Figure 20: Visual representation of a point in a map.

```json
{
  "type": "Feature",
  "geometry": {
    "type": "Point",
    "coordinates": [125.6, 10.1]
  },
  "properties": {
    "name": "Dinagat Islands"
  }
}
```

Listing 1: Point represented in GeoJSON.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<kml xmlns="http://www.opengis.net/kml
    /2.2">
<Document>
<Placemark>
  <name>Dinagat Islands</name>
  <Point>
    <coordinates>125.6,10.1</coordinates>
  </Point>
</Placemark>
</Document>
</kml>
```

Listing 2: Point represented in KML.

2.3 Project Context

As described in the Related work, there are several possible applications resultant of the crossing between AR and GIS. However, the solutions similar to our work, such as ARch4maps and MapLens have its limitations. Briefly, ARch4maps is directed to architects and designers, and MapLens uses obsolete technologies. As a result, it is clear that there is a gap to be filled.

Moreover, the aforementioned review of technologies allows us to clearly identify the properties of the project that follows. Regarding the AR technology, we adopted a **marker-based AR platform**, more precisely using **Natural Feature Tracking (NFT)** since the target map, is treated as an image.

In terms of GIS, we will foccus solely on the unit for the *result presentation facilities* (Shekhar et al., 1997), referring to the presentation of the geodata. The basemap is the target, on which the AR content is augmented. The geodata comprises the multiple overlays to be augmented over the target map. As a final remark, we use a **vectorial** representation of the geographical data. Yet, 3D models are also a viable extension as later detailed in Section 6.1.

Architecture & Solution Design 3

Based on our research, we divide our problem in two distinct tasks. The first component is the **detection and tracking** of the map from the video feed received from the camera. The second is the **rendering** of the additional data overlaid over the camera feed. To correctly position the overlay, we use the transformation matrix from the tracking component, together with the coordinate transformation from geographic to cartesian systems.

3.1 Challenges

3.1.1 Accurate, Precise and Fast Map Tracking

In terms of **detection and tracking** of the map, we require a highly-precise and accurate transformation matrix, since it allows for the correct positioning of the cartographic overlay information over the video feed. An imprecise matrix leads to the wrong positioning of the geographical data, leading to an erroneous representation of the geodata.

The algorithms used to compute this transformation matrix using image-based markers tend to follow the rule that "haste makes waste", related to the **trade-off between speed and accuracy**. On the other hand, a responsive system is an indispensable requirement for an Augmented Reality application. As a reference, the minimum image refresh rate deemed acceptable on AR applications is 15 Frames Per Second (FPS), in order to provide a smooth video motion to the user. (Craig, 2013). Low refresh rates lead to lapses in the user's visual perception of motion. As a side node, this is significantly below the refresh rate observed in cinema movies, at 24 FPS.

3.1.2 Geographic to Cartesian Coordinates

Another issue arises in the **rendering** component of systems that requires geographic to cartesian coordinate conversion. In practice, the first represents positioning as a tuple of latitude and longitude on the diameter of an oblate spheroid world. On the other hand, the latter represents the coordinate pixel-positioning on the 2D map surface instead. For a visual explanation of the earth geographic coordinate system, refer to Fig. 21.

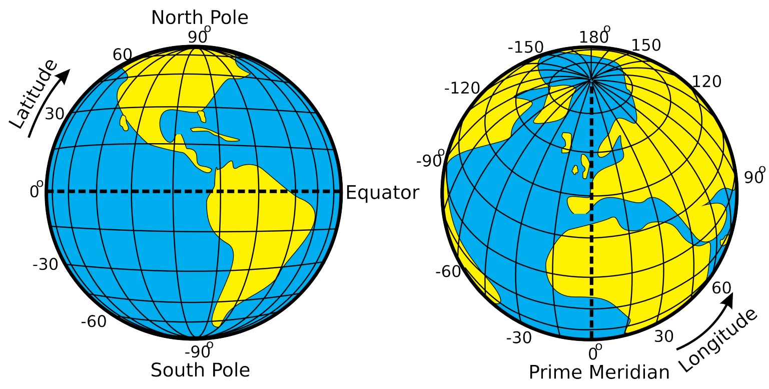


Figure 21: Earth geographic coordinate system.

3.1.3 Production System Limitations

An important set of challenges arises from the objective of the product. Having been performed in an enterprise environment, toward a sellable product, with real users in mind, certain restraints are imposed. These conditions draw a comparison line between the production system that underlies our development. While several existing projects, including the ones mentioned in Section 2.1, due to their scientific exploration nature, allow for a *powerful yet commercially infeasible* outcome (*e.g.* low responsiveness, poor user experience, bad integration with existing libraries, map adaptability issues). However, our work must obey some specifications, to name a few:

- It must be easily configurable to new maps and different input data without the need to make changes to the source code;

- The software and algorithms involved in the platform should be free and open source. With this requirement, patented or closed-source methods will not be used, even if more promising or easy to use;

- It should be modular in order to be easily upgradable and expanded to new use-cases. This directly implies replacement of separate components without the need to be changing the entirety of the source code;

- The technologies used should be widely accepted and used across multiple platforms, which means that any software with low acceptance or limited to a single platform or a single Operating System (OS) is not regarded as viable.

3.2 Framework

As mentioned previously, even though there are several AR frameworks, the majority of them are closed source and/or limited to a single platform. In order to fulfil the promises described previously, the following decisions were made. At first, the core programming language adepted was Python[1]. Secondly, Computer Vision (CV) primitives were extracted via **OpenCV**[2], a free and open-source library of programming functions mainly aimed at real-time computer vision, which is to be used in the video, tracking and rendering modules. Finally, the User Interface (UI) component was created with **PyQt**[3], an open-source widget toolkit for graphical user interfaces and cross-platform applications.

Since a major focus is to develop a public usage system, it needs to be easily configurable in terms of input. For this purpose, it was decided to have a Configuration File in YAML Ain't Markup Language (YAML) (McCombs, 2018), which is a human-friendly, cross-language, Unicode based data serialization language (Ben-Kiki et al., 2005).

In order to represent the vectorial geodata, which overlay the map, the selected option was GeoJSON due to its broad adoption — previously described in Section 2.2.2.

3.3 Approach

With a divide and conquer strategy in mind, besides the two core components described previously — the **tracking** and **rendering** modules — we also need two additional modules: (1) the **video** module, which captures and processes the frames from the camera; and (2) the **interface**, where the user has access to view and interact with the system.

1 https://www.python.org/
2 https://opencv.org/
3 https://riverbankcomputing.com/software/pyqt

The workflow is as follows. The video module communicates the video, by continuously sending frames, to both the tracking and the rendering components. From the received frame, the tracking module computes the transformation matrix and sends it to the rendering module. The rendering module combines both the frame and transformation matrix, and draws the frame with the augmented information overlaid. This augmented frame is forwarded to the interface where the user is able to interact with it, sending the user selection back to the rendering module for highlighting purposes. For clarity, this workflow is illustrated in Fig. 22.

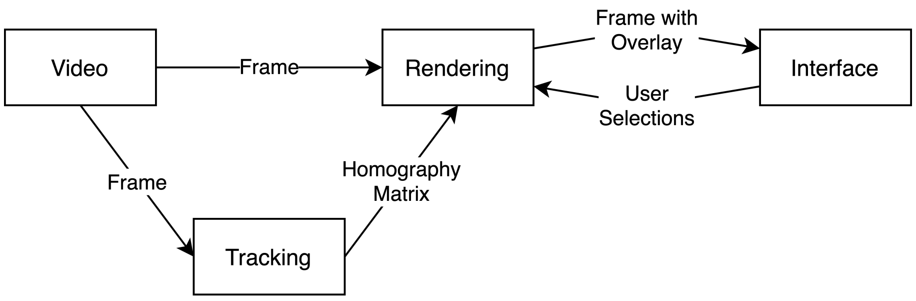


Figure 22: Diagram of the modular approach.

Having defined the main challenges, the framework and the approach, the next chapter introduces the implementation of the system.

Implementation 4

In this chapter, we discuss the implementation of the system. For clarity, we follow the natural order of the frame, from the moment it leaves the camera until it reaches the user. At the end of this chapter, we present a summary of the developed modular implementation, with final remarks on each component together with a diagram overview of the implementation.

4.1 Video

The task of reading and processing each frame of the camera is by definition I/O bound. This bottleneck was overcome by performing camera related I/O operations on a separate thread. In practice, while the main thread handles the processing of the current frame, the additional thread polls the camera for new frames.

It is important to remark that the system is unreliable, in the sense that some frames may be lost. This is the case if the augmentation phase that follows the video input — *i.e.* the tracking and rendering in Fig. 22 — takes longer than the loading time of the next frame. This issue is acceptable and adequate. The rationale is that, similarly to a real life situation, when a sequence of frames is incomplete, only the most recent (last processed) frame is relevant, and the previous are safely discarded. Thus, updated imaging is guaranteed, as only the last collected image serves as input. However, frame duplication can occur. Due to the concurrent producer-consumer approach, from the data loading and augmentation threads respectively, consistent processing of imaging is not guaranteed. To be precise, if the cycle of the augmentation thread is faster than the data loading thread's, the same frame can be processed more than once.

To finalize, the hardware utilized to collect videos was the Logitech C920 PRO, a camera widely used in computer vision projects (Taubin, 2018), enabling Full High Definition (HD) [1] at 30 FPS.

1 Video mode characterized by 1920 pixels displayed across the screen horizontally and 1080 pixels down the screen vertically.

4.2 Tracking

Several planar transformations can occur in 2D homogeneous coordinate vectors (Fig. 23) through a set of 3x3 matrices (Hartley and Zisserman, 2003). These transformations hierarchily-listed in Table 2 form a set of nested transformations, as an example this means that every affine transformation is also a projective transformation, but not the other way around. Additionally, the inverse of each transformation belongs to the same group as the original transformation.

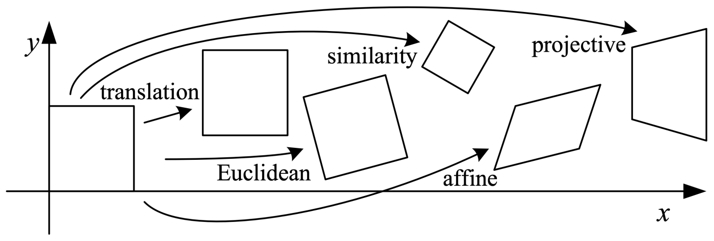


Figure 23: 2D planar transformations. Source: Szeliski (2010)

Transformation	**Matrix**	**#DoF**	**Preserves**
translation	$[I \mid T]_{2x3}$	2	orientation
rigid (Euclidean)	$[R \mid T]_{2x3}$	3	lengths
similarity	$[sR \mid T]_{2x3}$	4	angles
affine	$[A]_{2x3}$	6	parallelism
projective	$[H]_{3x3}$	8	straight lines

Table 2: Hierarchy of 2D coordinate transformations. Each transformation preserves the properties listed in the rows below it. The 2x3 matrices are extended with a third row $[0^T\ 1]$ to form a full 3x3 matrix for homogeneous coordinate transformations (I, R, T and s refer to the identity, rotation, translation and scale transformation matrices respectively). Source: Szeliski (2010)

A projective transformation, also known as a homography or perspective transform, operates on homogeneous coordinates. This transformation preserves straight lines. Nevertheless, it does not guarantee the persistence of orientation, lengths, angles and parallelism (Szeliski, 2010) — Table 2.

The objective of the tracking component is to compute the image transformation required for the coordinate matching of the marker into the video frame (Strang, 2010). The workflow is as follows. A camera is positioned in distinct locations, pointing at an object plane, *i.e.* the target map image (marker). The images (frames) contain the coordinates of the object plane, with a transformation applied to a different point of view. The tracking module reads the input for each pinhole image (frame) and **computes the transformation that matches the coordinates of the object plane to the frame** (camera image). For clarity, this schema is presented in Fig. 24.

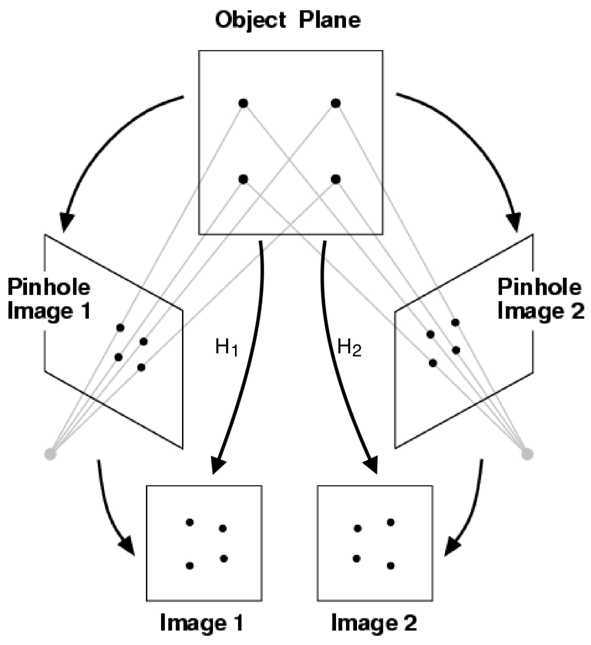


Figure 24: Planar projection diagram with a pinhole camera. In this specific situation the *Object Plane* is the target map (marker) and the *Images* refer to the frames from the camera. Adapted from Collins (2007).

To compute this matrix, two images are required at any given time; one is the target map, the marker which is being searched, and the other one is the frame, which is analyzed in order to find the marker — Fig. 25.

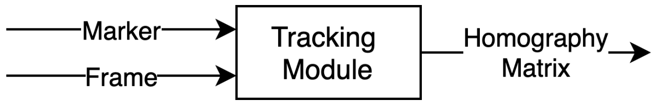


Figure 25: Input-output diagram of the tracking module.

As a side note, there are many distinct homography matrix applications. Among them, it is widely used on generating panorama photos, as a combination of image mosaicing and stitching methods.

4.2.1 Feature Detection

In order to recognize the marker in the frame, characteristics of both images are computed and compared. This kind of image features are to be detected even if the image is scaled, translated, sheared, rotated and even if it has a difference in brightness (Schmid and Mohr, 1997).

A direct, yet naive, approach is to perform a complete pixel-by-pixel comparison of both images. However, this is not resilient to any of the image altercations listed before. Instead, our method is based on the computation of feature points and the matching of its descriptors to both images. These features refer to the "interesting" parts of an image (Baggio, 2012). In fact, a perfect feature detection algorithm will always detect the same features regardless of the applied transformations.

Early feature detection algorithms try to detect corners in an image (Harris et al., 1988; Shi et al., 1994). These algorithms are rotation invariant, *i.e.* we can find the same corners in a rotated image. However, they are not scale-invariant because a corner might result in a curve in a zoomed image. In Lowe (2004), a scale-invariant algorithm is introduced with the name Scale-Invariant Feature Transform (SIFT). This approach is still widely accepted as one of the highest quality available alternatives - at the expense of computational cost (Leutenegger et al., 2011).

In Bay et al. (2006), a faster version of SIFT was published with the name of Speeded Up Robust Features (SURF). Despite being faster than the predecessor, this algorithm was not fast enough for a real-time application. Nonetheless, Features from Accelerated Segment Test (FAST) (Rosten and Drummond, 2006) had a clear advantage in speed, but its approach suffers in terms of reliability and robustness as it has minimal tolerance to image transformations and distortions, specifically in terms of in-plane rotation and scale change (Leutenegger et al., 2011).

In the meanwhile, several alternative algorithms were developed for both the feature points detection and feature descriptors extraction, such as Center Surround Extremas for Realtime Feature Detection and Matching (CenSurE) (Agrawal et al., 2008), Binary Robust Independent

Elementary Features (BRIEF) (Calonder et al., 2010), KAZE [2] (Alcantarilla et al., 2012), Binary Robust Invariant Scalable Keypoints (BRISK) (Leutenegger et al., 2011), Oriented FAST and Rotated BRIEF (ORB) (Rublee et al., 2011), among others. These algorithms were implemented and tested on our specific use case.

Based on previous research by Tareen and Saleem (2018), Karami et al. (2017) and Chien et al. (2016), ORB was the preferred algorithm for the feature detection. The three algorithm characteristics which supported this choice are: (1) the fact that it is not patented and free to use, which is one of the promises referred in Chapter 3; (2) its computational efficiency, needed for a real-time AR application; and (3) the equally important accuracy and precision offered, required for the use case of geographic data — taking into consideration that some algorithm parameterization was added, as described in Section 4.2.6. The application of this method to a sample input map is illustrated in Fig. 26, displaying the features computed through ORB over a the city of Oporto.

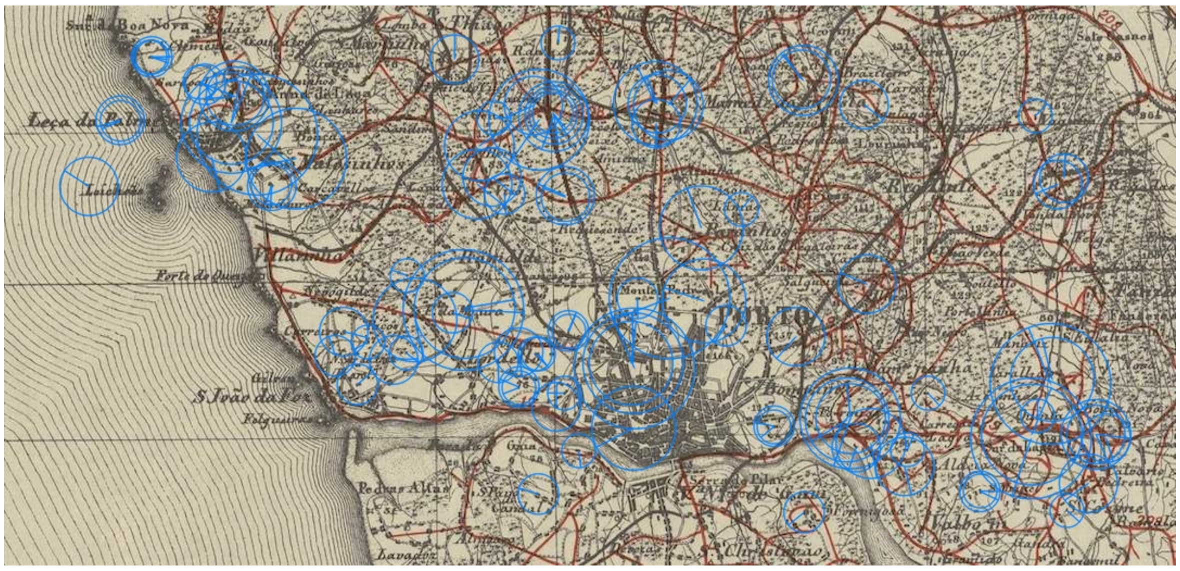


Figure 26: Features computed through ORB drawn over the map of Oporto. Each feature is defined by a center point, radius and orientation. Original map from Direcção Geral dos Trabalhos Geodesicos do Reino (1880).

2 Japanese word for *wind*

In the final solution, ORB uses 5000 features with a scale factor [3] of 1.1, since the default values of 500 features and a scale factor of 1.2 do not provide enough precision for the tracking. Nevertheless, to counterbalance the increase in computation time, the algorithm used to rank features was the *FAST_SCORE* (Rosten and Drummond, 2006), instead of the *HARRIS_SCORE* (Harris et al., 1988). This results in a faster computation time, although the resulting keypoints are slightly less stable.

4.2.2 Feature Matching

After computing the keypoints and its descriptors for both the marker and the frame, they are compared in order to find the matching pairs of keypoints.

There are two distinct approaches to feature matching. The first, via a brute-force algorithm, which exhaustively tries all possibilities to find the best matches. Secondly, through the approximate nearest neighbour algorithm, namely Fast Library for Approximate Nearest Neighbors (FLANN) (Muja and Lowe, 2009). It is a faster method, to the trade-off of not returning the optimal solution, *i.e.* it may not necessarily return the best solution. Performing a brute force matching is not feasible for a real-time application, so FLANN was chosen as the best fit.

The FLANN matcher is configured through an index and a search parameter. Since ORB is being used, the selected indexing algortihm is locality-sensitive hashing (LSH), with 6 hash tables, 10-bit long hash keys and 1-bit shift to check for neighboring buckets (Lv et al., 2007). Moreover, for the search, each tree in the index is recursively traversed 50 times — a higher value would result in a better search precision, but it would also take more time to process (OpenCV Team, 2020).

Throughout the iteration of the matching stage, mismatches may occur. These can be of two types: (1) a *false-positive* match, when the correspondence is wrong, or (2) a *false-negative* match, when a match is absent and both keypoints are visible in both images. *False-negative*

3 Pyramid decimation ratio, which must be greater than 1. For example a scale factor of 2 represents the classical pyramid, where each next level has four times less pixels than the previous one. Such a large value will degrade feature matching scores dramatically. However, if the value is too close to 1, more pyramid levels will be required resulting in a speed decrease. (OpenCV Team, 2020)

mismatches are not recoverable, as they are rejected by the algorithm. However, we can minimize the number of *false-positive* matches in order to reduce the total number of outliers. Quoting Shi et al. (1994):

> "A good discrimination at the beginning of the processing chain can reduce the remaining bad features to a few outliers, rather than leaving them an overwhelming majority. Outlier detection techniques at higher levels in the processing chain are then more likely to succeed."

This rationale is complemented by Lowe (2004), where the **ratio test** was proposed to eliminate the *false-positive* matches. After computing the distance ratio between the two nearest matches of the given keypoints, the algorithm labels it as a good match if this value is below a threshold. This threshold is considered to be approximately 0.75, which means every keypoint with a $ratio < 0.75$ is accepted as a good match (Fig. 27). This forms the basis of our method of **outlier removal**.

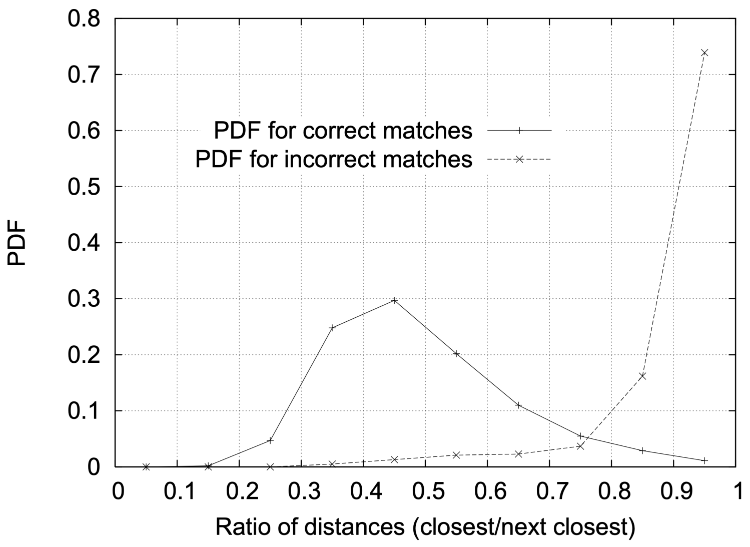


Figure 27: Using a database of 40,000 keypoints, the solid line shows the probability density function (PDF) of this ratio for correct matches, while the dotted line is for matches that were incorrect. Source: Lowe (2004)

The distance between features can be measured using different methods (Haavardsholm, 2019) such as:

- the sum of absolute differences (SAD): $d(f_a, f_b) = \sum |f_a - f_b|$;

- the sum of squared differences (SSD): $d(f_a, f_b) = \sum (f_a - f_b)^2$;

- the Hamming distance: $d(f_a, f_b) = \sum (f_a \oplus f_b)$.

In our implementation with ORB, we utilize the Hamming distance since ORB has binary descriptors (Fan et al., 2013).

As a final remark, the nearest neighbour distance ratio test is one of the most widely used, delivering both fast and good results. However, there are other filtering tests such as the geometric test, which eliminates matches that do not fit to a geometric model. The application of this method is detailed in the next section.

4.2.3 Homography Estimation

Following the extraction of features, their matching to the input images, and outlier removal, we then compute the homography matrix. This is a step of high importance, as an accurate matching process and outlier filtering leads to a better estimation in the homography process.

With this in mind, Random Sample Consensus (RANSAC) is applied in the system. The algorithm finds the best homography matrix by probing subsets of feature points (Fischler and Bolles, 1981). Additionally, it marks each correspondence as either inlier or outlier, taking into consideration the reprojection error for the calculated homography matrix (Fig. 28).

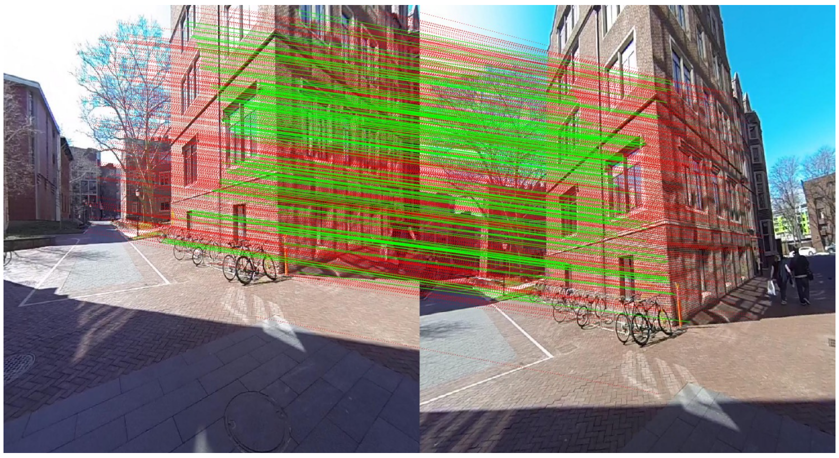
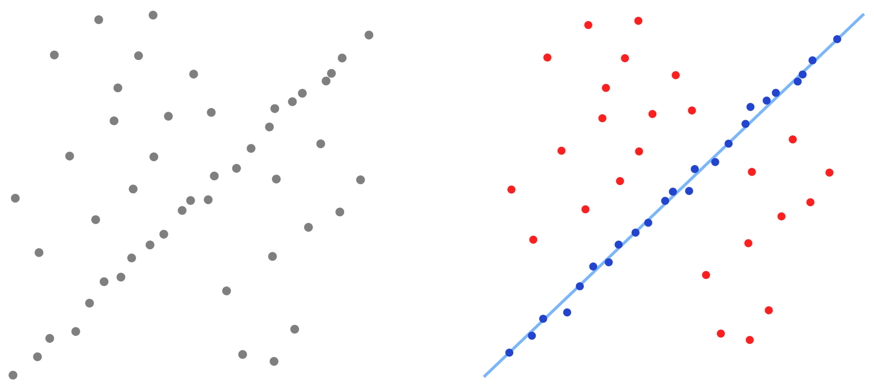


Figure 28: Matched features after RANSAC. (Green: Selected correspondences; Red: Rejected correspondences). Source: Singh (2019)

This is an iterative two-step algorithm. The first is the matching of randomly selected feature point pairs, in order to compute the model parameters. The second step is the calculation of pairs that follow this model and agree with a specified threshold. This step is iteratively executed until the stop condition is verified, *i.e.* when the maximum number of matches is found (Li and Selviah, 2011) (Fig. 29).



(a) Data set with many outliers for which a line has to be fitted.

(b) Fitted line with RANSAC; outliers have no influence on the result.

Figure 29: Robust regression with RANSAC.

The algorithm is designed based on the fundamental assumption that there are enough inliers to agree on a good model. In practice, this means that we need to have enough good keypoint matches, where *enough* is a count above a user-defined value. This issue is not problematic since, if there is not a sufficient number of good matches, it does not make sense to compute the homography as it would result in a distorted transformation and an inadequate estimation. Furthermore, in RANSAC the outliers do not interfere with the estimation — an indispensable requirement — because they will create unwanted distortions to the computed matrix (Chen et al., 2003).

The outcome of the application of the previous algorithm is usually refered as a planar homography. A planar homography relates to a projective transformation between two planes, it is represented as a linear operation on a 3x3 matrix with 8 DoF (as referred in Section 4.2). For completion, the planar transformation H of a point $[x, y]$ into $[x', y']$ can be computed as:

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = H \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

There are 9 variables in H, leading to the misconception that such transformation yields 9 DoF. However, we enforce 8 DoF by setting $h_{33} = 1$ or by imposing a unit vector constraint (Collins, 2007; Szeliski, 2010), in the form:

$$h_{11}^2 + h_{12}^2 + h_{13}^2 + h_{21}^2 + h_{22}^2 + h_{23}^2 + h_{31}^2 + h_{32}^2 + h_{33}^2 = 1.$$

4.2.4 Refined Homography

In order to further improve the precision of the estimated homography, a refinement process (Hadian and Kasaei, 2015) was added. To obtain the refined homography, the frame is initially warped with the inverse of the rough/initial homography (H_1^{-1}). Afterwards, the warped frame goes through the process of feature detection, feature matching and homography estimation, resulting on a second homography (H_2). In brief, the refined homography process is a composition of two homographies and can be formulated as $H = H_1 \cdot H_2$.

In practice, the transformation H_1 performs a rough homography estimation and the second one (H_2) slightly adjusts this transformation into a more precise result (Baggio, 2012).

4.2.5 Optical Flow

The process of feature extraction and matching required for the desired precision has two major drawbacks. At first, it is computationally expensive, and does not allow the algorithm to achieve the minimum of 15 FPS needed for an AR application (Craig, 2013). The second, is that the computed homographies for similar frames can be slightly different in terms of their distortions, even with a static camera. The rationale is that different feature points are detected on each frame, yielding a different set of inliers. This leads the AR content to jitter, which precludes the natural flow of the augmented content over the frames, therefore reducing the augmentation realism (Lepetit et al., 2009).

The proposed solution for both problems is the use of a bi-directional optical flow (Bouguet et al., 2001) applied between frames after the homography estimation. This process measures the 2D image motion between a sequence of frames (Prince et al., 2002; Mooser et al., 2007). This technique has been previously used with success in Chen et al. (2016) and Herakleous and Poullis (2013).

The integration of the iterative algorithm can be summarized as follows. The keypoints of the first frame i are extracted using ORB. Afterwards, using optical flow, we compute the new positions of those keypoints in the following frame $i+1$. In this process, the same set of inliers is used to avoid the jittering (Lepetit et al., 2009). Subsequently, a new homography is computed from the new positions of those feature points. This homography H_{of} represents the applied transformation between frames. Finally, the final transformation of the marker results of the application the subsequent transformations of the previous homography and of the optical flow:

$$H_{i+1} = H_i \cdot H_{of}$$

In Lucas, Kanade, et al. (1981), the Lucas–Kanade (LK) optical flow algorithm was proposed. This method has three base assumptions: brightness constancy, temporal persistence or small movements, and spatial coherence.

BRIGHTNESS CONSTANCY This first assumption implies that, when using the AR system, we maintain the brightness levels quasi-constant throughout the interaction with the system, for the period of analysis. Our implementation guarantees this property, since the user near a marker is usually in an almost static position.

TEMPORAL PERSISTENCE OR SMALL MOVEMENTS Although this algorithm requires small and coherent motion, a high level of motion is characteristc of a real-world situation. To overcome this issue, our method follows the common approach of combining an image pyramid with the Lucas–Kanade algorithm (Bouguet et al., 2001). This tracking method was further improved by customizing the algorithm parameters to allow for a deeper search. Nevertheless, this accuracy improvement was at the cost of increasing the system's execution time (Table 3).

SPATIAL COHERENCE In our implementation, spatial coherence relies solely on the keypoints of the marker, as these are the only elements required to compute the object-aware optical flow. With this approach, the homography is not influenced by the background flow (Bai et al., 2016). Therefore, spatial coherence is guaranteed. Additionally, this implementation is resilient to the situation where only the marker is moved and the camera remains static — otherwise, the static background with a moving marker would tamper with the estimated homography.

When a homography is always dependent on the homography of the previous instances, the accumulated error will subsequently add up throughout the tracking process. A naive approach to solve this problem is to compute a new homography with the "initial" approach every X frames or seconds. In this method, the newly computed homographic transformations are not dependent on the previous ones. However, the drawbacks are that static cameras and markers will blink/flash even without the non-accumulation of errors (which only occurs when there is movement). The proposed approach is to use a technique based on a depreciating value, the decreasing keypoint count. At every iteration, the count remains static in the occurrence of little or no movement, and decreases upon considerable movement. When the count reaches a certain user-defined threshold, we assume that there was enough movement that provides high certainty that there is a certain level of accumulated error/distortion. Therefore, a new homography is estimated using the "initial" approach through the extraction and matching of feature points.

4.2.6 Other Performance Improvements

In order to enhance the overall system's performance, several improvements were added. It is crucial to understand the **speed-accuracy trade-off**, where both aspects must be considered before changing the system.

SINGLE GRAY-SCALE CONVERSION PER FRAME The feature detection and descriptors computation tasks (Section 4.2.1) only use gray images. To avoid a duplicate conversion process, the frames are only converted once at the beginning of the process.

REDUCED IMAGES RESOLUTION Smaller resolution images allow take less time to process. As a result, instead of using the original frames resolution (1080p), scaled frames with a smaller resolution (720p) were used. This resulted in a scaled homography matrix. Additionally, the analogous reasoning was applied to the resolution of the input marker. Nevertheless, this approach must not be excessive, since it will decay the accuracy of the homography estimation.

MARKER FEATURES COMPUTED ONLY ONCE Since the marker (target map) does not change after loading the application, its features are only computed once and stored in memory.

ACCURATE TRACKING Although the Refined Homography was an improvement in terms of accuracy and precision, it slowed down the tracking process. Instead of increasing the accuracy at the expense of the homography estimation — performed for every frame — this computational cost was transferred to the feature extraction and matching phases — only performed occasionally. This was performed through a parameterization of the ORB and FLANN algorithms. This parameterization resulted in a steep increase in the number of features and changes in other configurations, as referred previously in the Sections 4.2.1 and 4.2.2. This resulted in an overall increase in the accuracy and precision without a sharp rise in the computational cost.

In practice, the feature extraction and matching phases can be computationally heavier, since they are only computed once when the image is found. Afterwards, the optical flow reliability is based on this initial precision [4]. Nevertheless, when the marker is not in sight this process is still computed for every frame which results in a low FPS experience. In order to resolve this issue, the heavy operation is performed only once every N frames. For clarity, this results in a delayed start of the tracking at a maximum of $N/Camera_{FPS}$ seconds — we

4 The optical flow algorithm also includes an LK method parameterization, as described in the previous section.

used $N = 10$, resulting in a delay of $10/30 = 0.3(3)sec$ — an insignificant delay for the user, which enables a seamless experience when the marker is not being detected.

THREADED TRACKING MODULE The video / camera module (described in Section 4.1) is on a separate thread which improves the overall speed of the system. With this in mind, the same reasoning was applied to the tracking module by running it on a separate thread. However, it did not work out as expected, since the result was as if a shadow was following the marker and not a realistic augmented overlay of the content over the marker.

In order to further comprehend the speed enhancement (or deterioration, which is required in some cases to improve the tracking accuracy), the system's frame rate was measured on several algorithm iterations within the same conditions by using a pre-recorded video as input for the tests — the results of this analysis are in Table 3 — the experimental setup is described in Section 5.2.

Improvement	**FPS**	**% Change to Previous Row**	**% Change to Original**
Initial version with ORB parameterization	3.56	—	—
Threaded camera module	3.66	↑ 2.81%	↑ 2.81%
Single gray-scale conversion per frame	3.82	↑ 4.37%	↑ 7.30%
Reduced frame resolution (from 1080p to 720p)	4.48	↑ 17.28%	↑ 25.84%
Reduced marker resolution (50%)	5.83	↑ 30.13%	↑ 63.76%
Marker features computed only once	6.38	↑ 9.43%	↑ 79.21%
FLANN matcher (prev. brute force matcher)	9.45	↑ 48.11%	↑ 165.45%
FLANN parameterization	7.39	↓ 21.79%	↑ 107.58%
Optical flow*	24.96	↑ 237.75%	↑ 601.12%
Optical flow LK parameterization*	26.13	↑ 4.69%	↑ 633.99%
Tracking delay of $N/Camera_{FPS}$**	30.92	↑ 318.40%	↑ 768.53%

Table 3: Tracking performance improvements. Each improvement preserves the previous improvements listed in the rows above it. The last two rows represent the final frame-rate of the system. *With a marker in sight. **Without a marker in sight, compared with previous comparable row — FLANN parameterization.

4.2.7 Summary

We have looked on detail into the independent components of our tracking solution. We will now provide a summary of the assemble of those components and the tracking workflow. For completion, the workflow is displayed in Fig. 30.

The tracking workflow starts with the image capture device (camera module). These frames are continuously received by the tracking module. When a frame is received, there is a verification if a pre-existing homography is available. In the affirmative case, the optical flow algorithm is applied. In the negative case, it goes to the delayed start condition ($\#frame \ \% \ N == 0$), which if negative returns a void homography matrix, and if positive proceeds with the tracking. The next step is the feature detection and matching, which is followed by the distance ratio test, for outlier removal. Afterwards, the number of matches is checked to be sure we have enough inliers to continue with the homography estimation. If that is not the case, a void homography is returned. Otherwise, the homography is estimated and forwarded to the rendering module. Additionally, if the homography refinement was added to the algorithm, it would be performed to any valid homography, before the rendering module.

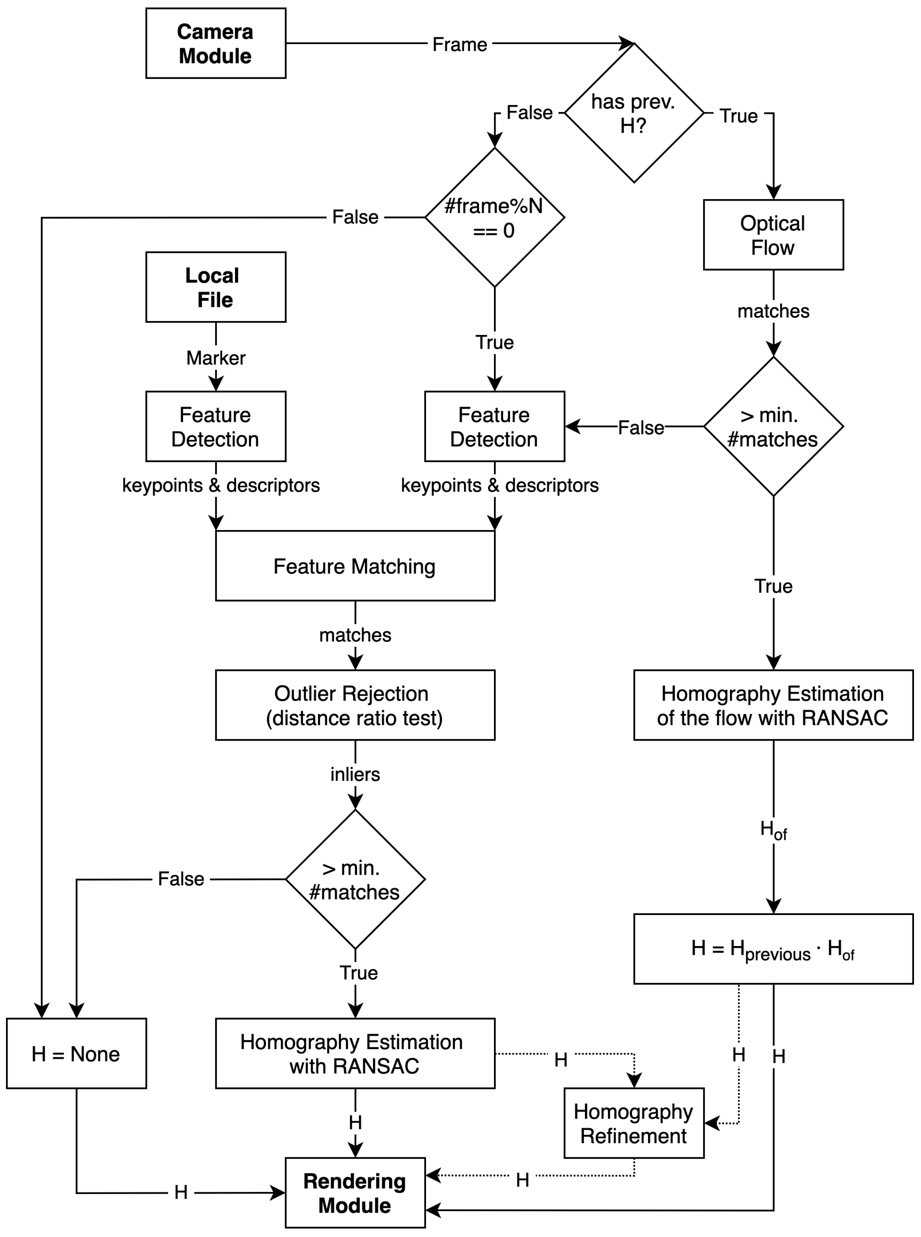


Figure 30: Diagram of the tracking algorithm (dotted line: alternative path).

4.3 Rendering

The Rendering module is responsible for the drawing of the AR content over the frame, positioned through the transformation given by the homography H_{3x3}. If no valid matrix is received from the tracking, then the marker has not been detected. This can occur when *e.g.* marker is not in sight, low light conditions, visual obstructions, etc.

The input-output diagram of the rendering module is illustrated in Fig. 31. This module needs (all) three inputs — scene/frame, overlay content, homography matrix – in order to augment the scene. If it is not possible to perform the augmentation, it returns the received frame.

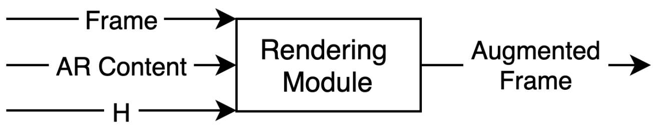


Figure 31: Input-output diagram of the rendering module.

4.3.1 From Geographic to Cartesian Coordinates

Before proceeding with the rendering of the geographical content, it is necessary to convert Geographic to Cartesian Coordinates, *i.e.* transform GIS coordinates into rectangular coordinates, represented as pixels of the marker image.

There are several formulas that may be used on the conversion of geographic coordinates into their cartesian counterparts for a given projection — see Bowring (1976) for a review. However, if we assume the target map to represent a relatively small physical area, which is the case, we can calculate the conversion without taking into account the oblate spheroid shape of the world. The reasoning behind this is that the resulting error of this calculation will not be noticeable enough to have a considerable impact on the final result, therefore, it can be disregarded. In practice, the geographical-to-cartesian conversion is computed as a direct positioning of overlays. Nevertheless, for system use cases that involve a world map, entire continents, or large countries, this approach would have to be adapted and take in the consideration the shape of the world.

For simplicity, this can be considered a planar transformation, which occurs in 2D homogeneous coordinates. More precisely, a similarity transformation, which preserves the angles, parallelism and straight lines, as defined in Section 4.2. This transformation can be represented by a $[sR|T]_{2x3}$ matrix comprising scale, rotation and translation transformations and has a total of 6 DoF (Hartley and Zisserman, 2003).

Since a similarity transformation is a subset of affine transformations (Hartley and Zisserman, 2003), it is possible to compute its matrix using the `getAffineTransform` function (OpenCV Team, 2020) with its input being three matching pairs of coordinates with their position in pixels from the Configuration File. Afterwards, a third $[0^T\ 1]$ row is appended to form a full 3x3 matrix to create a homogeneous coordinate transformation (Szeliski, 2010).

Afterwards, the similarity transformation matrix is used to convert the geographical coordinates from an input GeoJSON to pixel coordinates with the `perspectiveTransform` function (OpenCV Team, 2020).

Since this calculation is not dependent on the homography, it does not change for every frame. Consequently, this is only computed once per GeoJSON at loading time, and it replaces the original coordinates in memory with the newly transformed cartesian coordinates.

4.3.2 GeoJSON Rendering

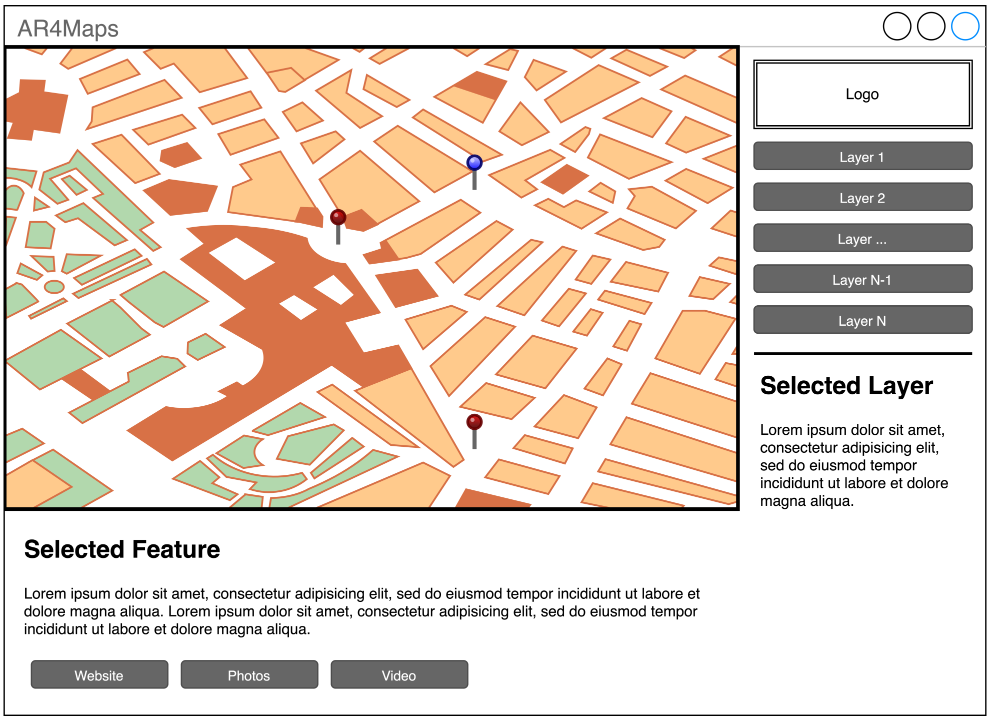
The augmented data is stored in the GeoJSON file format. Each layer is stored on a different file. This file format includes several geometry types, to name a few: `Point`, `LineString`, `Polygon`, `MultiPoint`, `MultiLineString` and `MultiPolygon`. The `Multi*` primitives are represented as a composition of basic geometries (Butler et al., 2016). The primitive shapes can be drawn with the OpenCV built-in drawing functions `circle`, `polylines` and `fillpoly` respectively.

In order to draw them, we first need to perform a `perspectiveTransform` of the pixels that compose the geometric objects using the homography matrix. This results in the objects being now projected over the marker detected in the frame.

Rendering works by continuously drawing every shape at every frame. As a side note, an attempt to improve the performance by drawing shapes once and dynamically warp them (`warpPerspective`) using the projection matrix was experimented. However, the blending process was deemed computationally expensive. Although it was not adopted, we believe this a viable option for large files with many shapes and/or complex shapes with numerous points[5], and we leave this as exploration in future work.

4.4 Interface

At the initial phase, the specifications start with the functional user requirements for the interface, i.e. the user-interaction specification (Al-Sarayreh and Abran, 2010). The minimum viable product (MVP) demands an ability from the user to alternate between geographical layers, to select visible augmented features, and to display information of the selected layer and feature. Based on these requirements, we designed a mockup of the user interface (pictured in Fig. 32) to serve as development guidance.



Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

Figure 32: Application wireframe / schematic.

5 This option is viable, since it enable an independent rendering time from the GeoJSON file size, while being only dependent on the rendering resolution of the overlay.

The desired functioning is as follows. At the right-hand side a listing of layers from the Configuration File is displayed. Upon selection of one of those layers, the corresponding GeoJSON file is loaded, and its textual description is shown under the listing. The central map is then augmented with the layer shapes. Upon the selection of a feature, its title and description are displayed on the bottom part of the interface, which are followed by buttons leading to more information such as the website, photo gallery and video.

4.4.1 Interaction with Augmented Overlay

A complex task is the user interaction with the augmented content. In our model, an user can interact with a feature, which can be either a `Point` or a `Polygon`, by clicking on top of the feature.

As a side note, each independent feature is identified by an universally unique identifier (UUID) (Leach et al., 2005), also know as globally unique identifier (GUID), a 128-bit identifier that guarantees uniqueness across space and time. This uniqueness property allows for efficient assertion of equality across features.

The click workflow is as follows. To begin with, the event is handled on the Interface, where the position in cartesian coordinates (x', y') represent a point over the frame. Nonetheless, these coordinates do not directly allow us to answer the question of which feature was clicked. This is due to the fact that the click coordinates are in the "frame plane" and need to be transformed to the "marker plane" to be comparable with the stored shapes in memory. Therefore, the (x', y') click coordinates are transformed to comparable coordinates (x, y) through the application of inverse of the homography matrix (H^{-1}):

$$(x, y) = H^{-1} \cdot (x', y')$$

Subsequently, we can compare the transformed clicked point with the stored features in memory, since they are now in the same geometric plane. In order to assert if a click was over a point, we check if the euclidean distance between the clicked position p and center q of the circle is less than the circle radius, which represents a `Point`:

$$d(p,q) < c_{radius} \iff \sqrt{(q_1 - p_1)^2 + (q_2 - p_2)^2} < c_{radius}$$

To check if a clicked point is inside a `Polygon`, the `pointPolygonTest` function is executed. This OpenCV function uses the Jordan curve theorem for polygons (Courant et al., 1941; Shimrat, 1962). Briefly explained, if we imagine that every point generates a ray in a fixed direction, not parallel to any sides of the polygon, that point is inside the polygon if it intersects P in an odd number of points, and is outside if it intersects P in an even number of points (Fig. 33).

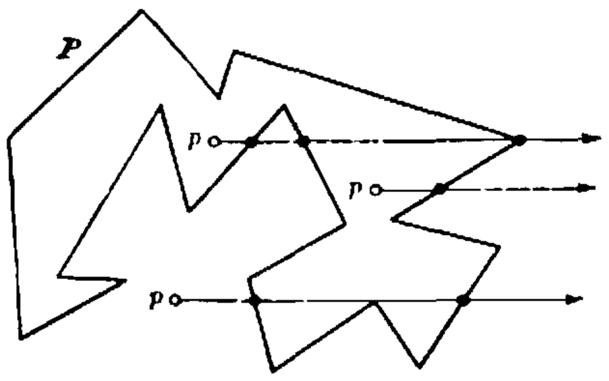


Figure 33: Counting intersections of rays with polygon's line. Each of the three points generate a ray, which cross the polygon's borders. The two rays on the top cross the polygon an odd number of times, which means those points are inside the shape. The bottom ray crosses an even number times, so the corresponding point is outside the polygon. Source: Courant et al. (1996)

4.5 Performance Tracking

To collect performance bottlenecks and analyse resource usage, execution was profiled using the native python profiler. Additionally, to collect metrics related to user experience, we developed a custom class to collect the Frames Per Second (FPS) throughout execution time. Moreover, the profile data visualization tool KCachegrind (2013) was used to further comprehend the data output from the profiler. The graphs pictured in Figs. 34 and 35 represents the calling relationships between subroutines in the application.

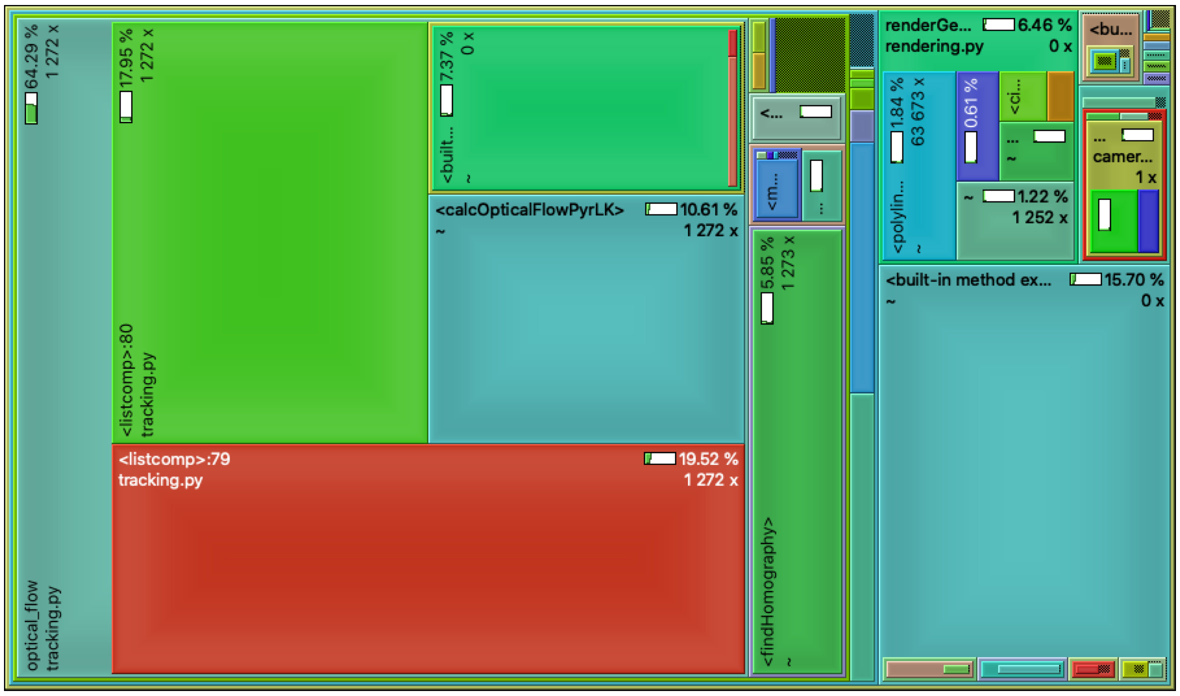


Figure 34: Tree map visualization of application's call hierarchy.

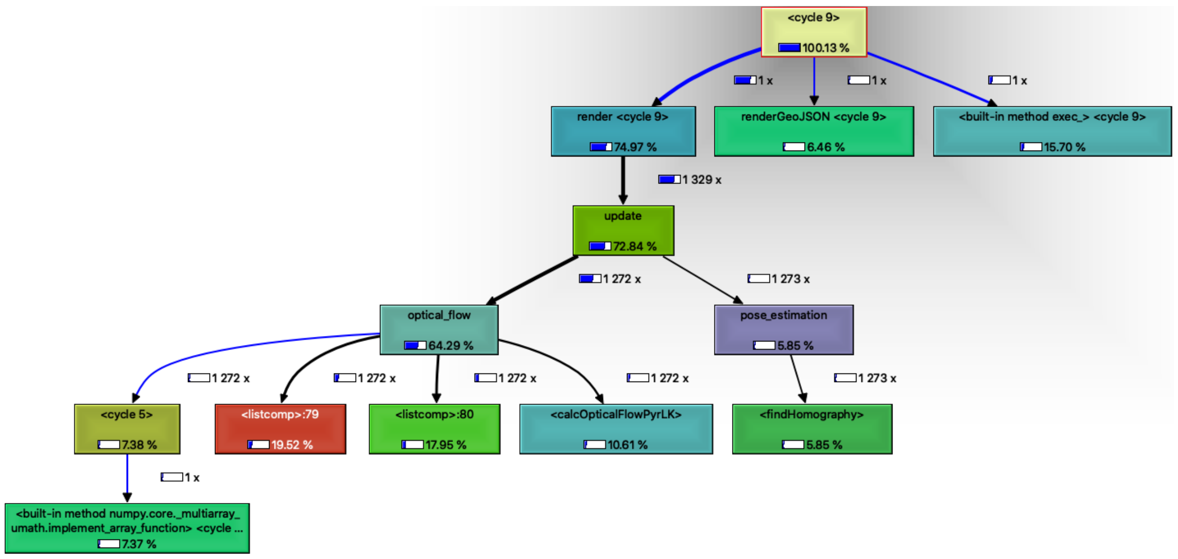


Figure 35: Call graph of the application. Each node represents a function and each edge indicates a function call.

These tools were indispensable to the continuous improvement of the involved algorithms, specially in the tracking process (described in Table 3) since it represents around 75% of the computational time as pictured in Figs. 34 and 35. Most decisions taken during development had their foundation on the trade-off between performance and precision.

4.6 Summary

At the end of this chapter, it was possible to test the assumptions, made prior to the implementation phase, through the development of the minimum viable product (Moogk, 2012).

Moreover, a modular approach, based on the divide and conquer strategy, proved to be the ideal path. This approach allows the system modules to be replaced. In fact, during development, we tested an Interface with *Tkinter*, and 3D Rendering with *OpenGL*. Additionally, the video module can be adapted to other use-cases such as reading the frames from an online stream or file. As a side note, the tracking algorithm proved to be the most challenging. Several distinct techniques were combined in order to create a Natural Feature Tracking (NFT) [6] algorithm which works in real-time and detects the marker with high accuracy and precision, indispensable for our distinct use-case with geographical data.

In Fig. 36, an extended diagram of the implementation is presented. On the top, it shows the modules of the system and their connections, already described in the previous chapter. On the bottom, it displays the local files that serve as static input to the system. For clarity, the marker image is the image of the map, the AR content describes an individual layer to be augmented and the config file defines the interface — sample input files can be seen in Appendices A.1 and A.2.

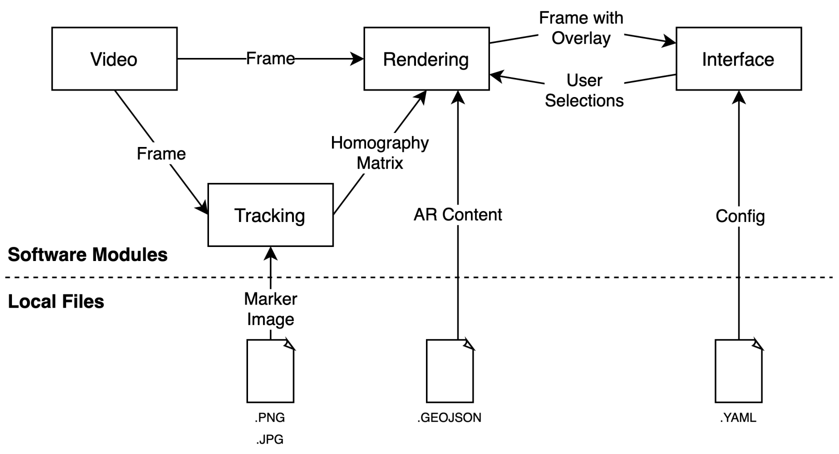


Figure 36: Diagram of the implementation.

6 Described in detail on Section 2.2.1

Results 5

We build a solution to provide augmented information over a map. The methods are applicable to *any* map and data layers. In this chapter, we discuss our solution applied to a real use-case.

5.1 Map Considerations

"No feature-based vision system can work unless good features can be identified and tracked from frame to frame."
— Shi et al. (1994)

This solution was developed to be used together with any map. However, not all maps make good image targets for AR. Since the feature extraction algorithms only use gray images, the markers must have good local contrast (Ćuković et al., 2015), which can be achieved by using maps with several colors yielding different shades of gray (Radkowski and Oliver, 2013). Another key characteristic is the presence of corners (Harris et al., 1988; Shi et al., 1994), since a marker with only a circle will not give us a sense of orientation/rotation.

The markers must be rich in detail, which results in a higher number of features being detected, leading to an increase in the number of possible matches. Moreover, this detail shall be evenly distributed across to allow tracking even parts of a marker without the need to have a specific region in sight (PTC Inc., 2020). In the specific case of a map, this detail can be added with the creation of different types of colored areas, buildings, text with street names, and other pertinent information.

Additionally, it is also vital for the physical marker used to follow some requirements: have at minimum of 12 centimetres in width; be flat; be matt, i. e. any glossy material needs to be avoided (PTC Inc., 2020).

An essential requirement for map markers is the avoidance of repetitive patterns (PTC Inc., 2020). Some cities have repetitive patterns in their planning, for example, the cities which use the grid plan. This type of map is not adequate for a marker because it is complicated to have a perception of the location in sight, since all the blocks look alike — Fig. 37.

PLAN of the CITY of NEW YORK,
REFERENCES
NORTH RIVER
HUDSON
EAST RIVER
(a) Plan of the City of New York, with the "recent" and intended Improvements. Drawn from actual survey by William Bridges City Surveyor. (1807)

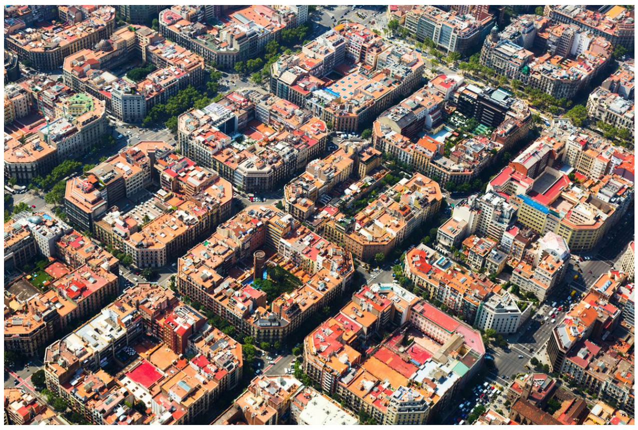

(b) Aerial view of the *Eixample* of Barcelona constructed in the 19th and 20th centuries in a strict grid pattern.

Figure 37: Grid plan examples.

In short, in order to use a specific map as a marker, it must have high local contrast and needs to be rich in detail evenly distributed. In reality, convoluted maps perform pragmatically better than the simpler ones.

5.2 Experimental Setup

We tested our methods on a real use-case with a map of the city of Oporto. Two formats were tested covering the same region: physical analogue printout maps and digital maps running on a Geoserver (Open Source Geospatial Foundation, 2019). The test were sucessfully performed on both smaller and larger mediums — A4/A0 analogue maps and 15"/55" screens for the digital maps.

The augmented scenario configuration is the following. The marker is a map covering an area of roughly 100 km^2. Moreover, there are 5 distinct overlay layers, which represent distinct separate topics. Each layer has different a different level of complexity in terms of the total number of points ranging from 224 up to 7774.

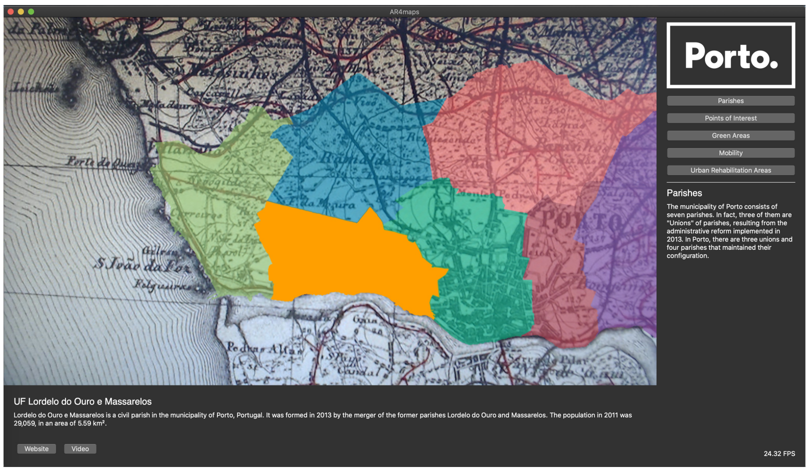
The camera used was the Logitech C920 PRO HD Webcam, which supports Full HD (1080p) at 30 FPS. Additionaly, the computer used for the experiments was the 15-inch MacBook Pro from 2018 with the following characteristics: an Intel Core i9 (I9-8950HK) CPU, 16 GB 2400 MHz DDR4 RAM, 512 GB SSD and a Radeon Pro 555X 4 GB GPU.

The user experience was synthetically induced, as a click at an average of 30 per minute, a value substantially high compared to regular usage. To that extent, we believe the system is reliable even in a worst case scenario. As a final remark, the system performed at 26.13 FPS when the marker was in sight, and at 30.92 FPS when the marker was out of sight, well above the minimum of 15 FPS acceptable for an AR application (Craig, 2013).

The source code necessary to reproduce the experiments is available in Github at: https://github.com/magamig/ar4maps.

5.3 Application

For the chosen use-case, we utilized data from *Câmara Municipal do Porto* (CMP)[1]. The data layers include information regarding the several parishes (*freguesias* in portuguese), the points of interest, the green areas, the mobility interfaces and the urban rehabilitation areas. The configured application is presented in Figs. 38 to 40.



Parishes

The municipality of Porto consists of seven parishes. In fact, three of them are "Unions" of parishes, resulting from the administrative reform implemented in 2013. In Porto, there are three unions and four parishes that maintained their configuration.

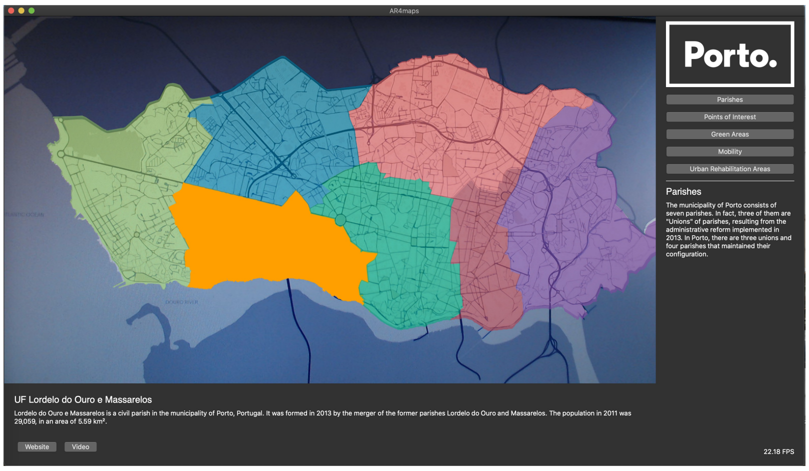
UF Lordelo do Ouro e Massarelos

Lordelo do Ouro e Massarelos is a civil parish in the municipality of Porto, Portugal. It was formed in 2013 by the merger of the former parishes Lordelo do Ouro and Massarelos. The population in 2011 was 29,059, in an area of 5.59 km².

24.32 FPS

Figure 38: Oporto's parishes augmented over a map.

1 http://www.cm-porto.pt/



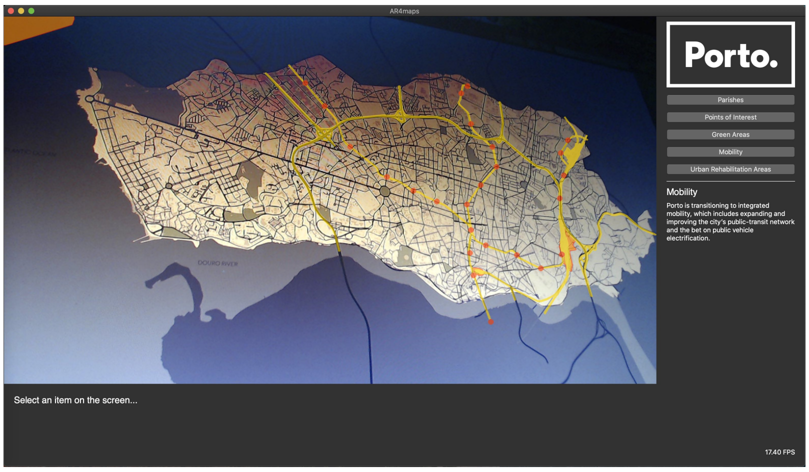
Parishes

The municipality of Porto consists of seven parishes. In fact, three of them are "Unions" of parishes, resulting from the administrative reform implemented in 2013. In Porto, there are three unions and four parishes that maintained their configuration.

UF Lordelo do Ouro e Massarelos

Lordelo do Ouro e Massarelos is a civil parish in the municipality of Porto, Portugal. It was formed in 2013 by the merger of the former parishes Lordelo do Ouro and Massarelos. The population in 2011 was 29,059, in an area of 5.59 km².

Figure 39: Oporto's parishes augmented over a different map.

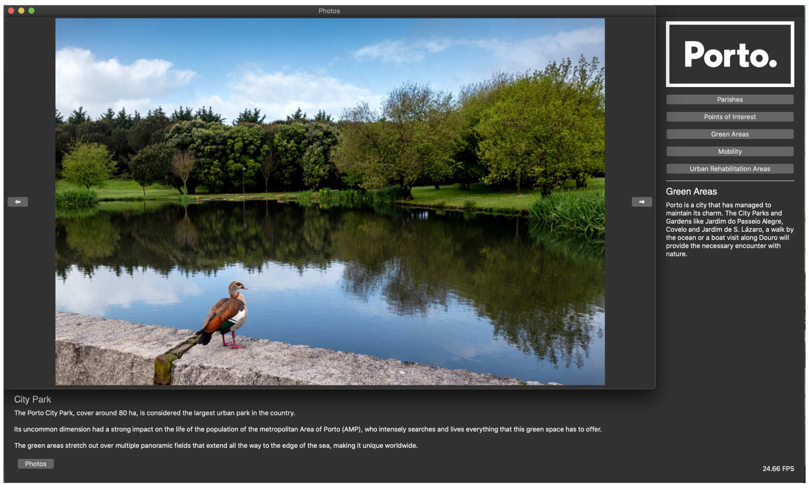


Mobility

Porto is transitioning to integrated mobility, which includes expanding and improving the city's public-transit network and the bet on public vehicle electrification.

Select an item on the screen...

Figure 40: Oporto's mobility interfaces augmented.



Green Areas

Porto is a city that has managed to maintain its charm. The City Parks and Gardens like Jardim do Passeio Alegre, Covelo and Jardim de S. Lázaro, a walk by the ocean or a boat visit along Douro will provide the necessary encounter with nature.

City Park

The Porto City Park, cover around 80 ha, is considered the largest urban park in the country.

Its uncommon dimension had a strong impact on the life of the population of the metropolitan Area of Porto (AMP), who intensely searches and lives everything that this green space has to offer.

The green areas stretch out over multiple panoramic fields that extend all the way to the edge of the sea, making it unique worldwide.

Figure 41: Photo slider of Oporto's city park.

5.4 Summary

We implemented our method on an input data representing high resolution maps of the city of Oporto.

We tested several camera transformations at a high user interaction rate. Our implementation demonstrated high efficiency, even in the synthetically-created experiments of extremely high level of user interaction. The interface was shown to be responsive and no observable delay was noticed due to background computation. We observed no bugs or glitches during the whole duration of the test.

Additionally, tests on the quality of the algorithms were performed. The computed overlays were shown to be precise in most scenarios. In the scenarios where there was not a perfect match between overlay and background marker, the visual discrepancy was observed to be negligible. In the extreme cases of displacement, the tracking algorithm quickly reloaded in order to adjust to the new situation.

Conclusion 6

Detailed conclusions for each part of this work were previously presented in their correspondent sections. In this chapter, the conclusions that can be extracted from the overall of this work are presented. Moreover we describe its contributions to the crossing of Augmented Reality and Geographic Information System technologies, together with feasible future work.

The presented work is intentionally focused over a single knowledge domain, *i.e.* Augmented Reality over maps. Nevertheless, due to its modularity, two implementation components can be extracted and used in different architectures/applications. Specifically, the tracking algorithm can be used for any application which is dependant on Natural Feature Tracking to track any image marker; and, the rendering logic can be used to draw geodata over any image.

The combination of these two very distinct technologies, together with additional logic and interaction connecting them both, allowed the creation of a different and innovative solution demonstrated by the developed minimum viable product.

Moreover, although a trade-off between accuracy and efficiency was required, the final implementation guarantees both at the same time. It displays accurate results to the naked eye, and is responsive as a real-time application.

Furthermore, the developed solution demonstrated high efficiency for all the tested use-cases. This guarantees its applicability to the previously described applications, *i.e.* museums, fairs, exhibitions, among others.

Finally, since a real use-case is provided, this also helps other developers to implement it to similar use-cases and adapt it to other scenarios. Additionally, it also confirms the easy configuration and applicability of our implementation. The solution configured for *Câmara Municipal do Porto* (CMP) is to be displayed at *Le marché international des professionnels de l'immobilier* (MIPIM)[1] - an international real estate event hosted in Cannes, France.

1 https://www.mipim.com/

6.1 Future Work

Although fully functional, the modularity of the developed solution allows it to be extended with new modules and have its current modules improved/replaced.

Despite the tracking algorithm being highly optimized, this is a field in constant development with new algorithms and methods being published continually. Keeping the tracking module up to date with the latest scientific advancements is a must to keep this work relevant.

In terms of the rendering, the developed augmentation feels artificial/unnatural since we are using synthetic vectorial shapes, which are sufficient for the developed minimum viable product. However, this can be improved with the addition of 3D shapes for the markers, lines and polygons. Moreover, geographically referenced 3D models would also create an immersive effect with buildings *coming out of the map*. Furthermore, topographic information can also be added to this solution. Taking into consideration the 3D addition, it is also pertinent to add realistic lighting in order to make the augmentation feel more natural by creating a more immersive experience.

Bibliography

Motilal Agrawal, Kurt Konolige, and Morten Rufus Blas. Censure: Center surround extremas for realtime feature detection and matching. In *European Conference on Computer Vision*, pages 102–115. Springer, 2008.

Khalid T Al-Sarayreh and Alain Abran. A generic model for the specification of software interface requirements and measurement of their functional size. In *2010 Eighth ACIS International Conference on Software Engineering Research, Management and Applications*, pages 217–222. IEEE, 2010.

Pablo Fernández Alcantarilla, Adrien Bartoli, and Andrew J Davison. Kaze features. In *European Conference on Computer Vision*, pages 214–227. Springer, 2012.

Apple. ARKit - Apple Developer Documentation 2020, 2017. URL `https://developer.apple.com/documentation/arkit`. Accessed May 17, 2020.

Apple. *Detecting Images in an AR Experience*, 2018. URL `https://developer.apple.com/documentation/arkit/detecting_images_in_an_ar_experience`. Accessed December 20, 2019.

Ronald Azuma, Yohan Baillot, Reinhold Behringer, Steven Feiner, Simon Julier, and Blair MacIntyre. Recent advances in augmented reality. *IEEE computer graphics and applications*, 21(6):34–47, 2001.

Ronald T Azuma. A survey of augmented reality. *Presence: Teleoperators & Virtual Environments*, 6(4):355–385, 1997.

Daniel Lélis Baggio. *Mastering OpenCV with practical computer vision projects*. Packt Publishing Ltd, 2012.

Min Bai, Wenjie Luo, Kaustav Kundu, and Raquel Urtasun. Deep semantic matching for optical flow. *arXiv preprint arXiv:1604.01827*, 2016.

Herbert Bay, Tinne Tuytelaars, and Luc Van Gool. Surf: Speeded up robust features. In *European conference on computer vision*, pages 404–417. Springer, 2006.

Daniel Belcher and Brian R Johnson. An augmented reality interface for viewing 3d building information models. In *26th eCAADe Conference Proceedings*, pages 561–568, 2008.

Oren Ben-Kiki, Clark Evans, and Ingy döt Net. Yaml ain't markup language - version 1.1, 2005. URL `https://yaml.org/spec/1.1/`.

Rahil Bhagat. Here's what you need to know about google's project tango. *Forbes*, Jun 2016. URL `https://www.forbes.com/sites/rahilbhagat/2016/06/20/just-what-is-googles-project-tango/#5ab0289b3baf`. Accessed January 8, 2020.

Jean-Yves Bouguet et al. Pyramidal implementation of the affine lucas kanade feature tracker description of the algorithm. *Intel Corporation*, 5(1-10):4, 2001.

Bernard Russel Bowring. Transformation from spatial to geographical coordinates. *Survey review*, 23(181):323–327, 1976.

Jerry Bowskill and John Downie. Extending the capabilities of the human visual system: An introduction to enhanced reality. *ACM SIGGRAPH Computer Graphics*, 29(2):61–65, 1995.

Wolfgang Broll, Irma Lindt, Jan Ohlenburg, Michael Wittkämper, Chunrong Yuan, Thomas Novotny, Chiron Mottram, Andreas Strothmann, et al. Arthur: A collaborative augmented environment for architectural design and urban planning. *JVRB-Journal of Virtual Reality and Broadcasting*, 1(1), 2004.

Howard Butler, Martin Daly, Allan Doyle, Sean Gillies, Stefan Hagen, Tim Schaub, et al. The geojson format. *Internet Engineering Task Force (IETF)*, 2016.

Michael Calonder, Vincent Lepetit, Christoph Strecha, and Pascal Fua. Brief: Binary robust independent elementary features. In *European conference on computer vision*, pages 778–792. Springer, 2010.

Jae Woo Chang. Navigation using augmented reality, July 25 2019. US Patent App. 16/251,771.

Patrick Charollais. *Final draft of the TC39 "The JSON Data Interchange Format" standard*. Ecma International, 2013. URL http://www.ecma-international.org/publications/files/ECMA-ST/ECMA-404.pdf.

Haifeng Chen et al. Robust regression with projection based m-estimators. In *Proceedings Ninth IEEE International Conference on Computer Vision*, pages 878–885. IEEE, 2003.

Peng Chen, Zhang Peng, Dalong Li, and Lijuan Yang. An improved augmented reality system based on andar. *Journal of Visual Communication and Image Representation*, 37:63–69, 2016.

Hsiang-Jen Chien, Chen-Chi Chuang, Chia-Yen Chen, and Reinhard Klette. When to use what feature? sift, surf, orb, or a-kaze features for monocular visual odometry. In *2016 International Conference on Image and Vision Computing New Zealand (IVCNZ)*, pages 1–6. IEEE, 2016.

Robert Collins. CSE486 lecture 16: Planar homographies. Accessed April 8, 2020, 2007. URL http://www.cse.psu.edu/~rtc12/CSE486/lecture16.pdf.

Open Geospatial Consortium. Keyhole markup language. *Opengeospatial.org*, Aug 2015. doi: 12-007r2. URL http://docs.opengeospatial.org/is/12-007r2/12-007r2.html.

Fábio Costa, S Eloy, J Dias, and M Lopes. Arch4models: a tool to augment physical scale models. *eCAADe 2017–ShoCK!*, pages 711–718, 2017.

Herbert Robbins Richard Courant, Richard Courant, Herbert Robbins, Ian Stewart, et al. *What is Mathematics?: an elementary approach to ideas and methods*. Oxford University Press, USA, 1996.

Richard Courant, Herbert Robbins, et al. *What is mathematics?* Oxford University Press, 1941.

Alan B Craig. *Understanding augmented reality: Concepts and applications*. Newnes, 2013.

Saša Ćuković, Michele Gattullo, Frieder Pankratz, Goran Devedžić, Ernesto Carrabba, and Khelifa Baizid. Marker based vs. natural feature tracking augmented reality visualization of the 3d foot phantom. In *The International Conference on Electrical and Bio-medical Engineering, Clean Energy and Green Computing (EBECEGC2015)*, 2015.

Ufuk Dilek and Mustafa Erol. Detecting position using arkit. *Physics Education*, 53(2):025011, 2018.

Direcção Geral dos Trabalhos Geodesicos do Reino. Portugal 1: 100,000. sheet 7, 1880. URL `http://1886.u-bordeaux-montaigne.fr/items/show/70166`. Accessed May 2, 2020.

ESRI. Esri shapefile technical description. *Environmental Systems Research Institute, Inc.*, 1998. URL `http://www.esri.com/library/whitepapers/pdfs/shapefile.pdf`.

Gabriel Evans, Jack Miller, Mariangely Iglesias Pena, Anastacia MacAllister, and Eliot Winer. Evaluating the microsoft hololens through an augmented reality assembly application. In *Degraded Environments: Sensing, Processing, and Display 2017*, volume 10197, page 101970V. International Society for Optics and Photonics, 2017.

Bin Fan, Qingqun Kong, Xiaotong Yuan, Zhiheng Wang, and Chunhong Pan. Learning weighted hamming distance for binary descriptors. In *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 2395–2399. IEEE, 2013.

Martin A Fischler and Robert C Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981.

Ricardo García, J Pablo de Castro, Elena Verdú, María Jesús Verdú, and Luisa María Regueras. Web map tile services for spatial data infrastructures: Management and optimization. *Cartography—A Tool for Spatial Analysis*, pages 25–48, 2012.

Sergio Garrido-Jurado, Rafael Munoz-Salinas, Francisco José Madrid-Cuevas, and Rafael Medina-Carnicer. Generation of fiducial marker dictionaries using mixed integer linear programming. *Pattern Recognition*, 51:481–491, 2016.

Filipe Gaspar, Steven Gomes, Ricardo Resende, Sara Eloy, Miguel Sales Dias, Mariana Lopes, and Nuno Faria. Arch4maps: a mobile augmented reality tool to enrich paper maps. In

Proceedings of the Symposium on Simulation for Architecture and Urban Design (SIMAUD), pages 179–182, 2016.

Anindya Ghose and Ke-Wei Huang. Personalized pricing and quality customization. *Journal of Economics & Management Strategy*, 18(4):1095–1135, 2009.

Steven Ramos Martins Gomes. *Realidade aumentada aplicada nos mapas de arquitetura: requalificação do bairro s. Nicolau. Requalificação de Setúbal nascente*. PhD thesis, ISCTE - Instituto Universitário de Lisboa, 2015.

Google. ARCore - Google Developers, 2018. URL `https://developers.google.com/ar`. Accessed January 5, 2020.

Google. Map and tile coordinates (Google Maps platform documentation), 2019. URL `https://developers.google.com/maps/documentation/javascript/coordinates`. Accessed December 28, 2019.

Google. Glass, 2020. URL `https://www.google.com/glass/`. Accessed May 17, 2020.

Ankita Gupta, Kriti Bhatia, Kritika Gupta, and Manu Vardhan. A comparative study of marker-based and marker-less indoor navigation in augmented reality. *Int. Res. J. Eng. Technol.(IRJET)*, 5:1–4, 2018.

Trym Vegard Haavardsholm. Lecture 4.2 feature matching, 2019. URL `https://www.academia.edu/40865474/Lecture_4.2_Feature_matching`. Accessed May 3, 2020.

Mostafa Hadian and Shohreh Kasaei. Fast homography refinement in soccer videos. In *2015 9th Iranian Conference on Machine Vision and Image Processing (MVIP)*, pages 185–188. IEEE, 2015.

Christopher G Harris, Mike Stephens, et al. A combined corner and edge detector. In *Alvey vision conference*, volume 15-50, pages 10–5244. Citeseer, 1988.

Richard Hartley and Andrew Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge university press, 2003.

Anders Henrysson and Mark Ollila. Umar: Ubiquitous mobile augmented reality. In *Proceedings of the 3rd international conference on Mobile and ubiquitous multimedia*, pages 41–45. ACM, 2004.

Kyriakos Herakleous and Charalambos Poullis. Improving augmented reality applications with optical flow. In *2013 IEEE International Conference on Image Processing*, pages 3403–3406. IEEE, 2013.

IPCity. Maplens, 2020. URL http://ipcity.fit.fraunhofer.de/?page_id=288. Accessed June 1, 2020.

Dalibor Janak. What are vector tiles and why you should care, Feb 2019. URL https://www.maptiler.com/blog/2019/02/what-are-vector-tiles-and-why-you-should-care.html. Accessed May 17, 2020.

M. Jobst, Symposium on Service-Oriented Mapping (1, 2012, Wien), and ICA Commission on Map Production and GeoBusiness. *Service Oriented Mapping 2012*. Jobstmedia, 2012. ISBN 9783950203929. URL https://books.google.pt/books?id=Z6TWyObSK-AC.

Timothy Jung, Namho Chung, and M Claudia Leue. The determinants of recommendations to use augmented reality technologies: The case of a korean theme park. *Tourism management*, 49:75–86, 2015.

Ebrahim Karami, Siva Prasad, and Mohamed Shehata. Image matching using sift, surf, brief and orb: performance comparison for distorted images. *arXiv preprint arXiv:1710.02726*, 2017.

KCachegrind, 2013. URL https://kcachegrind.github.io/. Accessed June 8, 2020.

Greg Kipper and Joseph Rampolla. *Augmented Reality: an emerging technologies guide to AR*. Elsevier, 2012.

Brian Klinkenberg and David Rhind. Maps and map analysis, 2020. URL https://ibis.geog.ubc.ca/courses/klink/gis.notes/ncgia/u02.html#SEC2.2.2. Accessed April 29, 2020.

Paul Leach, Michael Mealling, and Rich Salz. A universally unique identifier (uuid) urn namespace. *IETF*, 2005.

Vincent Lepetit, Francesc Moreno-Noguer, and Pascal Fua. Epnp: An accurate o (n) solution to the pnp problem. *International journal of computer vision*, 81(2):155, 2009.

Stefan Leutenegger, Margarita Chli, and Roland Y Siegwart. BRISK: Binary robust invariant scalable keypoints. In *2011 International conference on computer vision*, pages 2548–2555. Ieee, 2011.

Zhaowei Li and David R. Selviah. Comparison of image alignment algorithms. University College London, 2011.

Mariana Lopes, Jorge Silva, Miguel Sales Dias, Sara Eloy, Filipe Gaspar, Ricardo Miguel, and Nuno Mendonça. Sistema de realidade aumentada para apoio ao projeto de arquitetura. *Proceedings of EPCG*, 2014:21, 2014.

David G Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110, 2004.

Bruce D Lucas, Takeo Kanade, et al. An iterative image registration technique with an application to stereo vision (ijcai). In *7th International Joint Conference on Artificial Intelligence (IJCAI '81)*, volume 81. Vancouver, British Columbia, 04 1981.

Qin Lv, William Josephson, Zhe Wang, Moses Charikar, and Kai Li. Multi-probe lsh: efficient indexing for high-dimensional similarity search. In *Proceedings of the 33rd international conference on Very large data bases*, pages 950–961, 2007.

Map School. mapschool: a free introduction to geo, 2020. URL https://mapschool.io/. Accessed March 24, 2020.

Mapbox. How mapbox works, 2012. URL https://docs.mapbox.com/help/how-mapbox-works/#tilejson. Accessed March 20, 2020.

Mapbox. Vector tile specification, 2016. URL https://docs.mapbox.com/vector-tiles/specification/. Accessed May 17, 2020.

Lukas Martinelli and Manuel Roth. Updatable vector tiles from openstreetmap. *Hochschule für Technik Rapperswil, Rapperswil.[online] Available from: https://eprints. hsr*, page 1, 2016.

Anabela Marto, Alexandrino Gonçalves, and A Augusto de Sousa. Dinofelisar: Users' perspective about a mobile ar application in cultural heritage. In *International Conference on VR Technologies in Cultural Heritage*, pages 79–92. Springer, 2018.

Anabela Gonçalves Rodrigues Marto. Realidade aumentada móvel num contexto de herança cultural. Master's thesis, Universidade do Porto, 2017.

Thayne McCombs. Why json isn't a good configuration language, Jul 2018. URL https://www.lucidchart.com/techblog/2018/07/16/why-json-isnt-a-good-configuration-language/. Accessed February 11, 2020.

Nuno Adriano Santos Mendonça. *Expor arquitetura desmontando-a com realidade aumentada*. PhD thesis, ISCTE - Instituto Universitário de Lisboa, 2014.

Merriam-Webster Online. Merriam-Webster Online Dictionary, 2019. URL http://www.merriam-webster.com. Accessed December 3, 2019.

Microsoft. Announcing the 2018 microsoft citynext partner of the year awards, 2018. URL https://cloudblogs.microsoft.com/industry-blog/government/2018/07/17/announcing-the-2018-microsoft-citynext-partner-of-the-year-awards/. Accessed December 19, 2019.

Microsoft. Microsoft hololens | mixed reality technology for business, 2020. URL https://www.microsoft.com/pt-pt/hololens/. Accessed May 17, 2020.

João Ricardo Manarte Miguel. *Realidade aumentada aplicada ao processo de projeto de arquitetura*. PhD thesis, ISCTE - Instituto Universitário de Lisboa, 2014.

Paul Milgram, Haruo Takemura, Akira Utsumi, and Fumio Kishino. Augmented reality: A class of displays on the reality-virtuality continuum. In *Telemanipulator and telepresence technologies*, volume 2351, pages 282–292. International Society for Optics and Photonics, 1995.

TB Moeslund, M Störring, W Broll, F Aish, and Y Liu. The arthur system: an augmented round table. *Computer*, 1(1):277–282, 2003.

Markus Montola. A ludological view on the pervasive mixed-reality game research paradigm. *Personal and Ubiquitous Computing*, 15(1):3–12, 2011.

Dobrila Rancic Moogk. Minimum viable product and the importance of experimentation in technology startups. *Technology Innovation Management Review*, 2(3), 2012.

Jonathan Mooser, Suya You, and Ulrich Neumann. Real-time object tracking for augmented reality combining graph cuts and optical flow. In *2007 6th IEEE and ACM International Symposium on Mixed and Augmented Reality*, pages 145–152. IEEE, 2007.

Ann Morrison, Alessandro Mulloni, Saija Lemmelä, Antti Oulasvirta, Giulio Jacucci, Peter Peltonen, Dieter Schmalstieg, and Holger Regenbrecht. Collaborative use of mobile augmented reality with paper maps. *Computers & Graphics*, 35(4):789–799, 2011.

Marius Muja and David G Lowe. Fast approximate nearest neighbors with automatic algorithm configuration. *VISAPP (1)*, 2(331-340):2, 2009.

Alessandro Mulloni and Dieter Schmalstieg. Enhancing handheld navigation systems with augmented reality. *Service Oriented Mapping*, pages 63–80, 2012.

Rafael Munoz-Salinas. Aruco: a minimal library for augmented reality applications based on opencv. *Universidad de Córdoba*, 2012.

Igor Musatov, Vladimir Popov, and Vladimir Serov. Interactive touch screen map device, November 1 2001. US Patent App. 09/798,976.

Ramkumar Narayanan. Demystifying augmented reality, 2018. URL `https://github.com/ramkalath/Augmented_Reality_Tutorials/raw/master/demystifying_AR.pdf`. Accessed December 11, 2019.

Ulrich Neumann and Suya You. Natural feature tracking for augmented reality. *IEEE Transactions on Multimedia*, 1(1):53–64, 1999.

Oculus. Oculus Rift, 2019. URL `https://www.oculus.com/`. Accessed May 17, 2020.

Open Source Geospatial Foundation, 2019. URL `http://geoserver.org/`. Accessed May 28, 2020.

OpenCV Team. Opencv documentation, 2020. URL `https://docs.opencv.org/`. Accessed June 7, 2020.

Janne Paavilainen, Hannu Korhonen, Kati Alha, Jaakko Stenros, Elina Koskinen, and Frans Mayra. The pokémon go experience: A location-based augmented reality mobile game goes mainstream. In *Proceedings of the 2017 CHI conference on human factors in computing systems*, pages 2493–2498. ACM, 2017.

Raviraj S Patkar, S Pratap Singh, and Swati V Birje. Marker based augmented reality using android os. *International Journal of Advanced Research in Computer Science and Software Engineering (IJARCSSE)*, 3(5), 2013.

Rémi Paucher and Matthew Turk. Location-based augmented reality on mobile phones. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition-Workshops*, pages 9–16. IEEE, 2010.

Alec Pestov. Enhanced utility management with integration of gis and augmented reality, 2018. URL `https://www.geospatialworld.net/blogs/transforming-the-industry-with-esri-gis-and-advanced-visualization-tools/`. Accessed January 12, 2020.

Jacob Poushter et al. Smartphone ownership and internet usage continues to climb in emerging economies. *Pew Research Center*, 22:1–44, 2016.

Simon JD Prince, Ke Xu, and Adrian David Cheok. Augmented reality camera tracking with homographies. *IEEE Computer graphics and Applications*, 22(6):39–45, 2002.

PTC. Vuforia: Market-leading enterprise ar, 2019. URL `https://www.ptc.com/en/products/augmented-reality/vuforia`. Accessed December 7, 2019.

PTC Inc. Best practices for designing and developing image-based targets, 2020. URL `https://library.vuforia.com/content/vuforia-library/en/features/images/image-targets/best-practices-for-designing-and-developing-image-based-targets.html`. Accessed May 27, 2020.

Rafael Radkowski and James Oliver. Natural feature tracking augmented reality for on-site assembly assistance systems. In *International Conference on Virtual, Augmented and Mixed Reality*, pages 281–290. Springer, 2013.

Paul Ramsey. Serving dynamic vector tiles from postgis, Jul 2019. URL https://info.crunchydata.com/blog/dynamic-vector-tiles-from-postgis. Accessed May 3, 2020.

M Raposo, S Eloy, M Sales Dias, and M Lopes. Revisiting the city augmented by digital technologies–seearch tool. In *Proceedings of the 3rd International Conference on Preservation, Maintenace, and Rehabilitation of Historical Buildings and Structures*, pages 637–646. Green Lines Institute for Sustainable Development, 2017.

Muhammad Hanis Rashidan and Ivin Amri Musliman. Geopackage as future ubiquitous gis data format: a review. *Jurnal Teknologi*, 73(5), 2015.

Howard Rheingold. *Virtual reality: exploring the brave new technologies*. Simon & Schuster Adult Publishing Group, 1991.

Niles Ritter and Mike Ruth. The geotiff data interchange standard for raster geographic images. *International Journal of Remote Sensing*, 18(7):1637–1647, 1997.

Francisco J Romero-Ramirez, Rafael Muñoz-Salinas, and Rafael Medina-Carnicer. Speeded up detection of squared fiducial markers. *Image and vision Computing*, 76:38–47, 2018.

René Rosenbaum and David S Taubman. Remote display of large raster images using jpeg2000 and the rectangular fisheye-view. *The 11-th International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision'2003, WSCG 2003*, 2003.

Edward Rosten and Tom Drummond. Machine learning for high-speed corner detection. In *European conference on computer vision*, pages 430–443. Springer, 2006.

Ethan Rublee, Vincent Rabaud, Kurt Konolige, and Gary Bradski. Orb: An efficient alternative to sift or surf. In *2011 International conference on computer vision*, pages 2564–2571. Ieee, 2011.

H. Samet. *Applications of Spatial Data Structures: Computer Graphics, Image Processing, and Other Areas*. Addison-Wesley series in computer science. Addison-Wesley Longman, Incorporated, 1990. ISBN 9780201503005. URL https://books.google.pt/books?id=u9pQAAAAMAAJ.

Hanan Samet. *Spatial Data Structures*, page 361–385. ACM Press/Addison-Wesley Publishing Co., USA, 1995. ISBN 0201590980.

Silvio Ricardo Rodrigues Sanches, Cléber Gimenez Correa, Claiton Oliveira, and Joana Nascimento. Impact of frame rate in augmented reality for mobile devices. *IEEE Latin America Transactions*, 17(02):228–235, 2019.

Dieter Schmalstieg, Tobias Langlotz, and Mark Billinghurst. Augmented reality 2.0. In *Virtual realities*, pages 13–37. Springer, 2011.

Cordelia Schmid and Roger Mohr. Local grayvalue invariants for image retrieval. *IEEE transactions on pattern analysis and machine intelligence*, 19(5):530–535, 1997.

Johannes Schöning, Michael Rohs, Sven Kratz, Markus Löchtefeld, and Antonio Krüger. Map torchlight: a mobile augmented reality camera projector unit. In *CHI'09 Extended Abstracts on Human Factors in Computing Systems*, pages 3841–3846. ACM, 2009.

Joe Schwartz. Bing maps tile system - bing maps, Feb 2018. URL https://docs.microsoft.com/en-us/bingmaps/articles/bing-maps-tile-system?redirectedfrom=MSDN. Accessed December 28, 2019.

Shashi Shekhar, Mark Coyle, Brajesh Goyal, Duen-Ren Liu, Shyamsundar Sarkar, et al. Data models in geographic information systems. *Communications of the ACM*, 40(4):103–111, 1997.

Jianbo Shi et al. Good features to track. In *1994 Proceedings of IEEE conference on computer vision and pattern recognition*, pages 593–600. IEEE, 1994.

Moshe Shimrat. Algorithm 112: position of point relative to polygon. *Communications of the ACM*, 5(8):434, 1962.

Gilles Simon, Andrew W Fitzgibbon, and Andrew Zisserman. Markerless tracking using planar structures in the scene. In *Proceedings IEEE and ACM International Symposium on Augmented Reality (ISAR 2000)*, pages 120–128. IEEE, 2000.

Chahat Deep Singh. CMSC426 Computer Vision: Structure from Motion. Accessed May 3, 2020, 2019. URL https://cmsc426.github.io/sfm/.

Gilbert Strang. 18.06 Linear Algebra. Massachusetts Institute of Technology. MIT OpenCourseWare, 2010.

Richard Szeliski. *Computer vision: algorithms and applications*. Springer Science & Business Media, 2010.

Shaharyar Ahmed Khan Tareen and Zahra Saleem. A comparative analysis of sift, surf, kaze, akaze, orb, and brisk. In *2018 International conference on computing, mathematics and engineering technologies (iCoMET)*, pages 1–10. IEEE, 2018.

Gabriel Taubin. 3d photography: Camera calibration, 2018. URL `http://mesh.brown.edu/3DP-2018/calibration.html`. Accessed April 8, 2020.

Oren M Tepper, Hayeem L Rudy, Aaron Lefkowitz, Katie A Weimer, Shelby M Marks, Carrie S Stern, and Evan S Garfein. Mixed reality with hololens: where virtual reality meets augmented reality in the operating room. *Plastic and reconstructive surgery*, 140(5): 1066–1070, 2017.

Peter Turchi. *Maps of the imagination: The writer as cartographer*. Trinity University Press, 2011.

vGIS and Promark-Telecon. Measuring effectiveness of augmented reality in the locate services industry, 2018. URL `https://www.vgis.io/wp-content/uploads/2019/07/vGIS-Promark-Augmented-Reality-GIS-Locate-Industry-Study.pdf`. Accessed January 13, 2020.

vGIS Inc. Leading augmented reality (ar) platform for bim and gis, Feb 2020. URL `https://www.vgis.io/`. Accessed January 13, 2020.

Gheorghe-Daniel Voinea, Florin Girbacia, Cristian Cezar Postelnicu, and Anabela Marto. Exploring cultural heritage using augmented reality through google's project tango and arcore. In *International Conference on VR Technologies in Cultural Heritage*, pages 93–106. Springer, 2018.

VR Tech Today. Virtual reality vs. augmented reality vs. mixed reality – what is the difference? | vr tech today, 2018. URL `https://vrtechtoday.com/virtual-reality-vs-augmented-reality-vs-mixed-reality/`. Accessed June 8, 2020.

W3C. Extensible markup language (xml) 1.0 (fifth edition), 2013. URL https://www.w3.org/TR/xml/.

Jaron Waldman. Augmented reality maps, November 8 2016. US Patent 9,488,488.

Xiangyu Wang, Ning Gu, and David Marchant. An empirical case study on designer's perceptions of augmented reality within an architectural firm. *Journal of Information Technology in Construction (ITcon)*, 13(33):536–552, 2008.

Wikitude. Wikitude augmented reality: the world's leading cross-platform ar sdk, 2019. URL https://www.wikitude.com/. Accessed December 29, 2019.

Wikitude. Wikitude SDK - Platforms, 2020. URL https://www.wikitude.com/download/. Accessed June 13, 2020.

Jeff Yutzler. OGC® GeoPackage encoding standard-with corrigendum, version 1.2. 175. *Open Geospatial Consortium (OGC)*, 2018.

Acronyms

Symbols | **A** | **B** | **C** | **D** | **E** | **F** | **G** | **H** | **I** | **J** | **K** | **L** | **M** | **N** | **O** | **P** | **R** | **S** | **U** | **V** | **W** | **X** | **Y**

Symbols

2D two-dimensional.

3D three-dimensional.

A

AR Augmented Reality.

B

BIM Building Information Model.

BRIEF Binary Robust Independent Elementary Features.

BRISK Binary Robust Invariant Scalable Keypoints.

C

CenSurE Center Surround Extremas for Realtime Feature Detection and Matching.

CG Computer Graphics.

CMP *Câmara Municipal do Porto*.

CV Computer Vision.

D

DoF degrees of freedom.

E

ER Enhanced Reality.

Evolutio Ad Evolutio, Lda.

F

FAST Features from Accelerated Segment Test.

FLANN Fast Library for Approximate Nearest Neighbors.

FPS Frames Per Second.

G

GIS Geographic Information System.

GPS Global Positioning System.

GUID globally unique identifier.

H

HD High Definition.

HMDs Head Mounted Displays.

I

IMU Inertial Measurement Unit.

J

JSON JavaScript Object Notation.

K

KML Keyhole Markup Language.

L

LK Lucas–Kanade.

LSH locality-sensitive hashing.

M

MIPIM *Le marché international des professionnels de l'immobilier*.

MR Mixed/Merged Reality.

MVP minimum viable product.

N

NFT Natural Feature Tracking.

O

ORB Oriented FAST and Rotated BRIEF.

OS Operating System.

P

PDF probability density function.

R

RANSAC Random Sample Consensus.

RV Reality-Virtuality.

S

SAD sum of absolute differences.

SDK Software Development Kit.

SIFT Scale-Invariant Feature Transform.

SLAM Simultaneous Localization and Mapping.

SSD sum of squared differences.

SURF Speeded Up Robust Features.

U

UI User Interface.

UIs User Interfaces.

UUID universally unique identifier.

UX User Experience.

V

VE Virtual Environment.

VR Virtual Reality.

W

WWDC Apple Worldwide Developers Conference.

X

XML Extensible Markup Language.

Y

YAML YAML Ain't Markup Language.

Appendix A

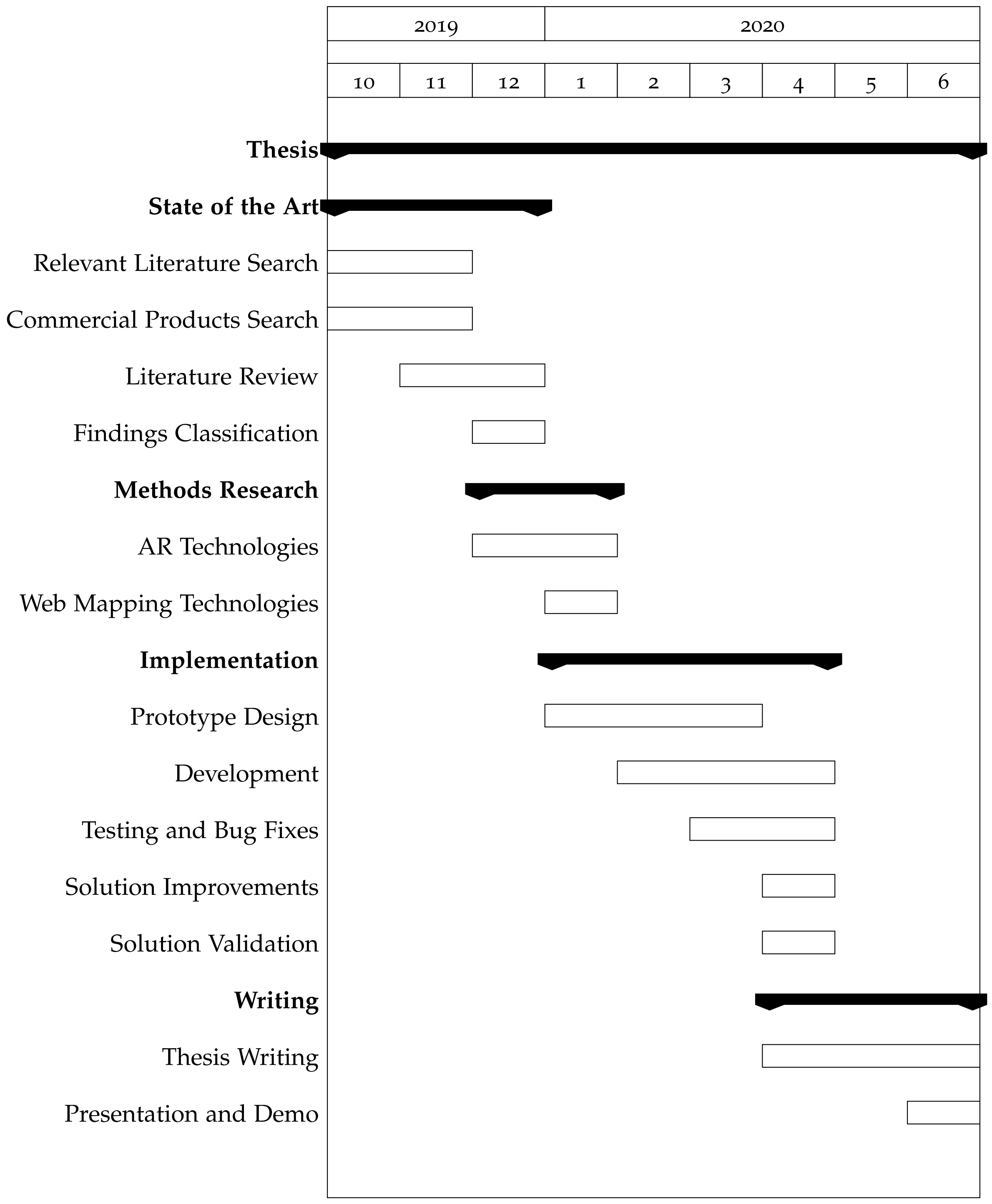
A.1 Configuration File

```yaml
%YAML 1.1
---
title: Porto
logo: images/cm-porto.svg
target: images/cmp.png
coords:
  src:
  - - 41.1464
    - -8.6668
  - - 41.14836
    - -8.64055
  - - 41.17183
    - -8.59542
  dst:
  - - 335
    - 416
  - - 503
    - 400
  - - 792
    - 200
layers:
- name: Parishes
  file: geojson/freguesias.json
  description: The municipality of Porto consists of seven parishes...
- name: Points of Interest
  file: geojson/poi.json
  description: Porto offers a lot of interesting venues to visit...
- name: Green Areas
  file: geojson/green.json
  description: Porto is a city that has managed to maintain its charm...
- name: Mobility
  file: geojson/mobility.json
  description: Porto is transitioning to integrated mobility, which...
- name: Urban Rehabilitation Areas
  file: geojson/arus.json
  description: New designated urban rehabilitation areas are being...
```

A.2 GeoJSON Layer File

```json
{
    "type": "FeatureCollection",
    "features": [
        {
            "id": 131202,
            "type": "Feature",
            "geometry": {
                "type": "Polygon",
                "coordinates": [
                    [
                        [
                            -8.5893,
                            41.16718
                        ],
                        [
                            -8.588871,
                            41.167125
                        ],
                        [
                                -8.622392,
                                41.143101
                        ],
                        [
                            -8.622213,
                            41.142975
                        ],
                        ...
                    ]
                ]
            },
            "properties": {
                "title": "Bonfim",
                "description": "Bonfim is a Portuguese parish, located...",
                "fill": "#b83f4d",
                "video": "https://www.youtube.com/embed/Y4YHnKJ6NeE",
                "website": "http://www.jfbonfim.pt/"
            }
        },
        ...
    ]
}
```

A.3 Schedule


2019
2020
10
11
12
1
2
3
4
5
6
Thesis
State of the Art
Relevant Literature Search
Commercial Products Search
Literature Review
Findings Classification
Methods Research
AR Technologies
Web Mapping Technologies
Implementation
Prototype Design
Development
Testing and Bug Fixes
Solution Improvements
Solution Validation
Writing
Thesis Writing
Presentation and Demo