JOÃO ANDRÉ ROSA PEREIRA

SMART AUGMENTED REALITY APPLICATION FOR ENHANCED MUSEUM EXPERIENCE

Mestrado em Engenharia Elétrica e Eletrónica
Especialidade em Tecnologias de Informação e Telecomunicações

Trabalho efetuado sob a orientação de:
Prof. Dr. João Rodrigues
Prof. Dr. Pedro Cardoso

UNIVERSIDADE DO ALGARVE
Instituto Superior de Engenharia
2017

SMART AUGMENTED REALITY APPLICATION
FOR ENHANCED MUSEUM EXPERIENCE

Declaração de autoria de trabalho

Declaro ser o autor deste trabalho, que é original e inédito. Autores e trabalhos consultados estão devidamente citados no texto e constam da listagem de referências incluída.

I hereby declare to be the author of this work, which is original and unpublished. Authors and works consulted are properly cited in the text and included in the reference list.

(João André Rosa Pereira)

Resumo

Os espólios dos museus contêm inúmeros objetos, tornando-se difícil escolher quais as obras a visitar e apreciar. Quando um utilizador entra num museu, geralmente depara-se com um guia e/ou rotas predefinidas, que frequentemente não são adequadas às suas necessidades e preferências. Esta dissertação foca o desenvolvimento de uma *framework* de realidade aumentada e de uma aplicação inteligente para multiplataforma, que pode ser usada como guia de museu e auxiliar de navegação.

O trabalho foi dividido em 3 módulos principais: (a) um sistema de cálculo de rotas inteligentes, (b) uma interface adaptativa de utilizador e (c) uma *framework* de reconhecimento de imagens com realidade aumentada. Também é apresentada a integração dos módulos acima mencionados numa aplicação.

O primeiro módulo, o (a) módulo do sistema de cálculo de rotas inteligentes, representa uma solução para um problema "comum" dos museus: as rotas existentes nos museus não tomam em consideração as limitações físicas, morais ou psicológicas do utilizador e/ou suas preferências. O problema em causa consiste em calcular uma rota, visitando uma e só uma vez cada ponto de interesse existente (mas não necessariamente todos os disponíveis no museu), percorrendo o menor caminho possível, e estendendo ao máximo o tempo de visita aos objetos do museu. Neste caso, foi usada uma adaptação de um algoritmo

de *Ant Colony Optimization* para calcular o melhor caminho, considerando as preferências e limitações do utilizador. Este problema foi formulado como um problema de otimização multi-critério.

Ainda nesta temática, (b) foi desenvolvida uma interface adaptativa de utilizador, que se ajusta de acordo com as preferências e condições deste. Este módulo é constituído por um sistema modular de cartões os quais são dividido em estrutura e conteúdos. Foi escolhido este sistema pois permite que uma interface complexa possa ser dividida em sub-módulos mais simples, que podem ser usados noutras partes da aplicação ou mesmo noutra aplicação completamente distinta. Idealmente, cada utilizador teria uma interface com estrutura e conteúdos distintos. No entanto, diferentes utilizadores podem partilhar a mesma estrutura/layout apenas modificando o conteúdo apresentado. Assim, este modulo permite criar facilmente diferentes interfaces para os diferentes utilizadores, quer modificando apenas os conteúdos apresentados ou também toda a sua estrutura.

Relativamente ao módulo de realidade aumentada (c), foi desenvolvido uma *framework* de reconhecimento de imagens com realidade aumentada (MIRAR - *Mobile Image Recognition and Augmented Reality*) para dispositivos móveis. O objetivo deste módulo é reconhecer e fazer o restreamento dos objetos do museu recorrendo ao dispositivo móvel do utilizador. A *framework* desenvolvida é baseada no reconhecimento de marcadores e apesar deste acontecer no cliente (dispositivo móvel) é necessário um servidor para guardar os marcadores pré-processados. Estes são, posteriormente, acedidos pelos dispositivos móveis à medida que os utilizadores navegam pelo museu. A localização do utilizador dentro do é calculada através de um sistema de *beacons* bluetooth a qual é transmitida para o servidor, que, por sua vez, envia os marcadores correspondentes a essa localização para o dispositivo do utilizador.

Finalmente, a integração dos módulos supra-mencionados é apresentada nu-

ma versão alfa da aplicação móvel, bem como testes e resultados para cada módulo.

Palavras-chave: Interface de Utilizador Adaptativa, Aplicação Multiplataforma, Realidade Aumentada, Reconhecimento baseado em Marcadores, Interação Homem-Máquina, Navegação, Cálculo de Rotas, Otimização.

Abstract

Museums' collections can be almost endless, with countless objects, making it challenging to choose which ones to visit and appreciate. When a user enters a museum he usually encounters a guide or a predefined route to aid him, which more often than not is not suitable for his necessities and preferences. This dissertation focus on developing a mobile augmented reality framework and intelligent multiplatform application, that can be used as a museum guide and navigation helper. The work was divided into 3 main modules: (a) an intelligent routing system, (b) an adaptive user interface, and (c) an image recognition and augmented reality framework. Also presented is the integration of the above modules in an application.

The first module, (a) intelligent routing system module, poses a solution for a "typical" museum problem. Museums routes do not take into account the physical, moral or psychological limitations of a user and/or their preferences. It resembles the traveling salesman problem where a route is calculated, only visiting once each point of interest, diminishing as much as possible the "walking" time, and extending the time spent admiring the museum's objects. An Ant Colony Optimization algorithm was used to handle the calculations and compute an optimal walk, rendering the user's preferences and limitations. This problem was formulated as a multi-criteria optimization problem.

Also focusing on adapting the application for the user, (b) an adaptive user interface was developed, which adapts the application's user interface on-the-fly, according to the user's preferences and conditions. This module is built upon a modular card system which is divided into structure and contents. It relies on a modular system in the sense that a complex interface can be divided into simpler and more manageable sub-modules, which can be used in other parts of the application or even in a completely different one. On an ideal application, each user would have a distinct interface/structure and contents. Nonetheless, different users could share the same interface structure only modifying the contents. The adaptive user interface is capable of (as the name implies) adapting itself to the user, either by changing both its structure and contents or only the contents displayed to the user.

Regarding the augmented reality module (c), a mobile image recognition and tracking framework (MIRAR) was developed. The purpose of this framework is to recognize and track the innumerous objects of the museum in a mobile device. This framework is a marker-based augmented reality framework and even though the recognition happens on the client (mobile device) a server is required to keep the packaged markers accessible for the clients. These markers are preprocessed in the server and grouped by section. As the user navigates through the museum, an indoor beacon location system calculates his current position that is transmitted to the server which, in turn, sends the correct markers for that section to the mobile device.

Finally, the integration of the above modules is presented in an alpha version of a mobile application, as well as tests and results for each module.

Keywords: Adaptive User Interface, Multiplatform Application, Augmented Reality, Marker-based Recognition, Human-Computer Interaction, Navigation, Routes Calculation, Optimization.

Acknowledgments

This thesis was accomplished with the direct or indirect contribution of many people without whom this work would not be possible.

To my parents, brother, sister-in-law, grandparents, and more recently my niece my recognition for their unconditional support and endless encouragement, with a special notice for my grandmother that without her I would not be here. And now in Portugues: "*Para os meus pais, irmão, cunhada, avós e, mais recentemente, a minha sobrinha o meu agradecimento pelo todo apoio incondicional e incentivo, com uma menção especial para minha avó, sem a qual eu não estaria aqui.*"

To all my friends who accompanied me on this journey. Special thanks are due to my (recent and old) lab colleagues for the support, knowledge, and for withstanding with my "madness", also I could not forget *João Parente Silva* for accepting me as I am and who as "walked" with me since Day 1 of our academic journey.

Last but definitely not least, a unique and very big thanks to my advisors, Prof. *João Rodrigues* and Prof. *Pedro Cardoso*, for their invaluable support, availability, knowledge, and bearings provided, that allowed me to complete this thesis. I will not easily forget the day I almost lost all the work...

Thank you!

Table of Contents

List of Tables

List of Figures

List of Abbreviations

AAT	*The Art & Architecture Thesaurus*
ACO	*Ant Colony Optimization*
AGI	*Automatic-Generation of Interfaces*
AI	*Artificial Intelligence*
APP	*Application*
AR	*Augmented Reality*
AUI	*Adaptive User Interface*
BRIEF	*Binary Robust Independent Elementary Features*
CONA	*Cultural Objects Name Authority*
CPU	*Central processing unit*
CS	*Candidate Set*
DB	*Database*
EUI	*Emotional User Interface*
FIFO	*First In First Out*
HCI	*Human-Computer Interaction*
HD	*High Definition*
ICT	*Information and Communication Technology*
IUI	*Intelligent User Interfaces*
IxD	*Interaction Design*

JSON	*JavaScript Object Notation*
KNN	*K-Nearest Neighbors*
M5SAR	*Mobile Five Senses Augmented Reality System for Museums*
MIRAR	*Mobile Image Recognition based Augmented Reality*
ORB	*Oriented FAST and Rotated BRIEF*
PDTTSS	*Portable Device for Touch, Taste and Smell Sensations*
POI	*Point of Intereset*
RANSAC	*Random Sample Consensus*
RGB	*Red Green Blue*
SDK	*Software Developlment Kit*
SURF	*Speeded-Up Robust Features*
TGN	*Thesaurus of Geographic Names*
UI	*User Interface*
ULAN	*Union List of Artist Names*
UM	*User Modeling*
UX	*User Experience*

1

Introduction

Imagine a visit to a museum where you can see, hear, feel, smell and maybe even taste what existed when the museological pieces were developed or what is represented in them. This would be a step towards a more immersive experience when compared to the traditional guides existing in most museums. This dissertation proposes the development of a multiplatform Augmented Reality (AR) application (App) for mobile devices (Android, iOS, and Windows) that seeks to use four out of the five human senses (sight, hearing, touch, and taste). The application intends to be used within buildings and aims to serve as a guide in cultural, historical and museological events. The application is integrated in the M5SAR project: Mobile Five Senses Augmented Reality System for Muse-

ums (for more details see Sec. 1.1).

The mobile device App, when using AR, will enhance the user experience adding a variety of levels of information or categories such as practical, theoretical, aesthetical and symbolic, which could not be previously accessed. The application will make a real-time bi-dimensional (2D) and three-dimensional (3D) recognition of the objects in the museological institution. It will be a connection between the object and the senses of the user, enhancing its experience.

Augmented reality applications for mobile devices are not new to the market (Yovcheva et al., 2013; Garau, 2014; Jung et al., 2015; Museum of London, 2017). Many of these applications are based on markers (e.g., tags or images; a.k.a. marker-based), others in geographic location information (e.g., GPS; a.k.a. markerless), and some combine the two types. However, they mainly try to explore the sense of vision and even those who try to value the hearing sense do not do it with the best quality, providing only some complementary information.

This dissertation explores the use of the marker-based technology inside a building/museum, using real life objects (e.g., paintings or statues) as markers. The main intention is to build a "smart" application that adapts to the user while also exploring the hear, touch and taste sensations, besides the already mentioned sight.

Various attributes will be explored to give "intelligence" to the application, in particular (a) the detection of objects/markers (2D and 3D) from various "optimized"/selected angles and scales (distance from the user to the marker) using the mobile device camera, as well as, the selection of specific regions in those objects in the AR, returning extra information about those regions. (b) Adaptive Navigation (AN) and the visualisation of the information using (c) Adaptive User Interfaces (AUI) will also be addressed.

As mentioned, the application intended to perform, in real-time, the visual

recognition of a work of art, usually also mentioned by the museum experts as "object", designation equally adopted in this thesis, and access its specific information, such as its author, techniques, collections, historical context, amongst others. It is also desired to give the visitor the possibility of choosing which information is relevant to him. The visitor decides, at any time, which topics pleases him the most, or as an alternative, the application suggests which route within the museum space would be most appropriate to the visitor's preferences. The application should also adapt to the user's profile, providing him with a different (adaptive) UI with intelligent navigation options. Additionally, the application must also provide generic museum information like: a museum map (allowing visitors to know their location as well as points of interest such as entrances, exits, café, restaurant, souvenir shop, bookstore, etc.), information about the museum itself (e.g., history, programs, schedules, cultural events, etc.), and itineraries (advising for each different visitor profile on routes designed by experts with comments and access to specific content information).

The application should also provide extra functionally for featured works such as: zoom (see an artistic object in detail), highlights (displays unique information on established pictorial elements in selected works), and merchandising information (after identifying an artistic object, emphasize what associated products/souvenirs are available). These featured works can be further classified as masterpieces (for the main works-of-art on display) where personalized content, such as images, videos, 3D contents, and infographics can be developed to facilitate comprehension and reading of the work. Throughout the application, concepts such as gamification, design thinking, design sensitive and responsive design will be used for representing all of the information required. Special care will be taken for not overlapping the information with the works, as it happens in some augmented reality application (Bell et al., 2001).

It is important to stress at this point, that the work presented in this the-

sis is around 16 months of the M5SAR project (see Sec. 1.1), not all the above-mentioned functionalities are already implemented, but all the main foundations for those functionalities are concluded.

1.1 Scope of the Thesis

This thesis is integrated (i.e., is one of the products) in the project M5SAR: Mobile Five Senses Augmented Reality System for Museums, funded by Portugal2020, CRESC Algarve 2020 I&DT, n° 3322, promoter SPIC - Sonha Pensa Imagina Comunica, Lda.[1] and co-promoter University of the Algarve [2]. The project started in Jan. 2016 and will finished in Oct. 2018.

In summary, the project aims to develop an AR system, consisting of an application platform and a device (usually referred to as "gadget") to be integrated with mobile devices (phablets and tablets) that explore the 5 human senses (5S) (vision, hearing, touch, smell, and taste). The system focuses on being used as a guide in cultural, historical and museum events. The system will consist of three products, integrated or used individually: (a) Application for mobile devices (software) that will be available in the "app store" or on a website, which will work on the latest phablet and tablet devices, focusing on 4 of the 5 senses: sight, hear, touch and taste. The second product consists of (b) the device (hardware) that can be sold separately or rented at the museum. This device focuses on 3 of the 5 senses: touch, smell and taste. (c) Both the App, (a), and the device, (b), may be associated, deepening the user experience. For this, the hardware device has to be able to attach itself to the mobile device, as well as to communicate and be integrated with the App. Together they will allow the full integration of the 5 senses in the AR. Each of these products individually and especially together will be without a doubt a novelty in the market, because nothing exists

[1] http://spic.pt/
[2] http://www.ualg.pt/

with these settings/specifications. The final product allows a more immersive interaction with the surrounding space than the existing AR systems. For this goal to be achieved it is necessary to combine research and development in various areas such as ICT (Information and communication technology), electronics, emotions' psychology and design, with applications and equipment for Smart Cities, creative industries and media, tourism, and natural and cultural heritage. By this via, an innovative product for the global market will be created. More information and publications about the project can be found on the website of the project[3].

The work presented in this thesis focus on product (a), and is the work done from March 2016 to June 2017.

1.2 Objectives

As mentioned above, the main target of this thesis is to develop a mobile device application that can promote 4 out of the 5 human senses and capable of working in real time on a mobile device. As such, the objectives of the thesis are to develop a multiplatform application for Android, iOS, and Windows with the following main features/modules:

1. A mobile application to be used in a museum space;
2. An intelligent navigation module based on the user's preferences and characteristics;
3. A user interface module based on "Adaptive Cards";
4. A marker-based image recognition and augmented reality framework;
5. A communication protocol between all the modules.

[3]https://goo.gl/z4WqAS

At this point, it is important to stress how the mentioned senses were addressed in this thesis. The two main focused senses, "Sight" and "Hearing", were addressed using AR and AUI. The remaining three senses ("Taste", "Smell", and "Touch") were addressed using an hardware device attached to the mobile device. For more information about this hardware device please see (Rodrigues et al., 2017a) and (Sardo et al., 2017), since the hardware device is out of the focus of this thesis.

1.3 Structure of the Thesis

The writing structure of this thesis consists of a author's papers compilation, where the above subjects are deepened. In Sec. 5.3 are listed other papers produced with the author's participation during its master program. Therefore, each of the main chapters of this thesis is composed of a paper that is either published or accepted for publication, having its own introduction, state of art, methods' description, and conclusions. The bibliography was removed from each individual paper to simplify the reading and is presented at the end of the thesis instead. Furthermore, please note that since the work in this thesis is part of the work developed in the M5SAR project, as are the presented papers, the introductions in each chapter have partial similar contents.

To clarify the author's work, below is described its overall contribution to each of the 3 papers that correspond to the 3 main chapters of this thesis. In the following order:

- Chapter 2 presents the paper *"A Cultural Heritage and Points of Interest Multi-Criteria Router Supported on Visitors Preferences". In Proc. of the 7th International Conference on Software Development and Technologies for Enhancing Accessibility and Fighting Info-exclusion (DSAI 2016). ACM, New York, NY, USA, pp. 392-399.*

The main contribution of the thesis author to this paper/chapter was the design of an ant colony optimization algorithm to solve a problem comparable to the travelling salesman problem to compute the tour of the visitor inside museum using its preferences (see Sec. 2.2) and its respective implementation (see Sec. 2.3 except method A) as well as comparative testing.

- Chapter 3 presents the paper *"Adaptive Card Design UI Implementation for an Augmented Reality Museum Application", accepted for 11th Int. Conf. on Universal Access in Human-Computer Interaction, integrated in the 19th Int. Conf. on Human-Computer Interaction, 9 - 14 July, Vancouver, Canada.*

  In this paper/chapter, the main author's contribution was the development of the Adaptive User Interface based on a modular card UI system (see Sec. 3.2) where the App interface is being generated in real time as the App is used ("not compiled").

- In Chapter 4 is presented the paper *"MIRAR: Mobile Image Recognition based Augmented Reality Framework", accepted for Int. Congress on Engineering and Sustainability in the XXI Century, 11 - 13 October, Faro, Portugal.*

  The author's contributions for this paper/chapter are present in the entire document, as he presents its finished image recognition and augmented reality module work (previously initialized), apart from the environment detection module (see Sec. 4.4).

In Chapter 5 (Conclusions), the App is briefly presented, i.e., the integration of the above 3 modules into a single mobile multi-platform application for museums (see Sec. 5.1).

1.4 Overview of the thesis

In summary, the present chapter introduced the thesis theme as well as the main goals, contributions and scope, consisting on three different accepted or published papers related to the intelligent navigation system, adaptive user interface, and augmented reality framework for a mobile application. Chapter 2 presents a solution for the navigation module inside a museological space that dynamically calculates a route based on the user's preferences, limitations, and available time. Chapter 3 presents the adaptive user interface module which is based on modular adaptive cards to create the application UI. It also introduces the first approach to the image recognition and augmented reality module for mobile devices. Chapter 4 continues this work improving the accuracy of the image recognition system as well as implementing the augmented reality component. Last but not least, Chapter 5 suggests a complete system integration, concluding the work done in the previous chapters along with the future work.

2

A Cultural Heritage and Points of Interest Multi-Criteria Router Supported on Visitors Preferences

Abstract

Cultural heritage is experienced differently by different visitors. The more erudite know beforehand what they intend to explore when visiting a monument, a museum or a city. On the other side, least erudite visitors usually know and are capable of expressing some of their preferences but do not realize exactly

what to see and explore. Physical and psychological limitations also make cultural products not interesting or suitable for all people. In this sense, this work proposes a set of methods which take into account the network of the local to be explored (e.g., museum or a city) and the classification of the Points of Interest (PoI) to build a set of optimal walks translating the users' preferences and limitations. The problem is formulated as a multi-criteria optimization problem and a set of results shows, for a museum and an urban space, the validity of the proposed methods.

2.1 Introduction

Technology is changing the way cultural heritage is experienced. Traditional visits to museums, cities and other spaces include a predefined walk, or set of walks, which does not translate the majority of the users' real preferences and needs. Many times the number of Points of Interest (PoI) are also large, making impossible to experience them all in a limited time window, and therefore necessary to proceed to a careful selection of what is going to be explored. In parallel, enhancing accessibility and fighting info-exclusion is another vector which should be explored, using for instance some classification systems to include features which reflect the degree of impairment of the visitor along with its preferences. For example, painting PoI probably are not adequate for a blind person or some predefined walks include stairs which are not transposable by people in wheel chairs or other mobility disability.

This routing based on users' preferences is being studied for some time. The Rijksmuseum Amsterdam offers a real-time routing system that implements a mobile museum tour guide for providing personalized tours tailored to the user's position inside the museum and interests (van Hage et al., 2010). The system includes tools for the interactive discovery of user's interests, semantic

recommendations of artworks and art-related topics, and the (semi-)automatic generation of personalized museum tours. In Benouaret and Lenne (2015) is proposed a recommender system for mobile devices. The system adapts to the users' preferences and is sensitive to their contexts, building tours on-site according to their preferences and constraints. A state-of-the-art in the field, proposing a classification of mobile tourism recommender systems and providing insights on their offered services, can be found in Gavalas et al. (2014). The CHESS project (CHESS, 2017) researches, implements and evaluates both the experiencing of personalized interactive stories for visitors of cultural sites and their authoring by the cultural content experts. Spatially broader, the RoutePerfect (2017) allows to easily plan a trip in Europe based on the traveler preferences, budget and personal style. Several other works can be found in literature such as Garcia et al. (2011); Verbert et al. (2012); Wang and Xiang (2012).

In this chapter we propose a multi criteria (Deb, 2001) formulation for a router recommender system. Besides restrictions, as the maximum allowed time for the visit, three goals were devised with the objective of optimizing: (i) the user's preferences, (ii) the number of visited PoI, and (iii) the time spent exploring PoI. To solve the optimization problem, a set of methods to design the visit of the users through some space were designed. The methods are supported on the Ant Colony Optimization (Dorigo and Stützle, 2004) algorithms and a weighted function strategy. As input the methods require a network with a set of PoI categorized according with some classification system and information from the user's preferences over the same classification system. Some results are presented for a network which intends to represent a museum and for the network of Faro city (Portugal).

The remaining document is structured as follows. The next section presents our formulation of the problem. Section 2.3 describes the proposed methods and some results are explored in the section 2.4. The last section presents some

conclusions and future works.

2.2 Preliminaries and Problem Definition

This section presents some preliminaries and defines the problem to be address in this work.

Let N_0 be a network that represents a space to be visited. For instance, in Fig. 2.1 left is sketched a (non-real) museum where each node is a PoI and on the right is sketched a network of the city of Faro (Portugal). In both cases, the color nodes are PoI (blue nodes) and the white ones are edge intersections which we will call auxiliary nodes. The main difference between the presented networks is the presence of those auxiliary nodes, which are not presents in the museum network. Each edge of the network as associated a traversing time – the time to go from one node to an adjacent one. On the other hand, auxiliary nodes have visiting time equal to zero and the PoI have a pre-determined visiting time, which will be considered if the visitor is to explore the PoI.

Depending on the type of network, each PoI is categorized according with some classification system which also depends on the type of space and the type of users that use the application. For instance, a PoI in a city, like Faro, can be categorized as juvenile, shopping, science, museum, church, theater, monument, kids-park, edification, sightseeing, etc. In the case of a geographical region, a more complete classification system can be derived from the GeoNames geographical database where each feature is categorized into one out of nine feature classes and further sub-categorized into one out of 645 feature codes (Geonames, 2017; Coughlan et al., 2015). If we consider a museum (or similar) other classification system are adaptable to our system as Iconclass which is a hierarchically ordered collection of definitions of objects, people, events and abstract ideas that serve as the subject of an image. Art historians, researchers and curators use it

to describe, classify and examine the subject of images represented in various media such as paintings, drawings and photographs (Iconclass, 2017; Couprie, 1978; Isemann and Ahmad, 2014). Also the The Art & Architecture Thesaurus (AAT), the Getty Thesaurus of Geographic Names (TGN), and the Union List of Artist Names (ULAN) are structured vocabularies that can be used to improve access to information about art, architecture, and material culture. The Cultural Objects Name Authority (CONA) is currently in development. It compiles titles, attributions, depicted subjects, and other metadata about works of art, architecture, and cultural heritage, both extant and historical; metadata is gathered or linked from museum collections, special collections, archives, libraries, scholarly research, and other sources (Getty, 2017; Baca and Gill, 2015). Not used at this point of development, the previous classification system can easily adapted and included in the proposed work.

The overall system can be designed to enhance accessibility and fighting info-exclusion, as the previous PoI classification systems can be extended to include features which reflect the degree of impairment of the visitor. For instance, painting PoI probably are not adequate for a blind person and therefore their classification relative to blind people would be very low. Furthermore, the network can also be designed with the impairments in mind, including information of the edges and PoI that are possible to be used by the visitors (e.g., the edges will not include stairs if the user uses a wheelchair or has some other mobility difficulties).

In resume, the network is a structure $N_0 = (V_0, E_0, d_0, t, C)$ where V_0 is the set of nodes which can be PoI or auxiliary nodes, $E_0 \subset V_0 \times V_0$ is the set of edges each connecting two nodes, $d_0 : E_0 \to \mathbb{R}_0^+$ is a function that associates to each edges its traversing time, $t : V_0 \to \mathbb{R}_0^+$ associates to each node the expected visit time and, since each PoI can be categorized in more than one class, $C : V_0 \to \{0, 1, \ldots, 5\}^m$ classifies each node according with m classes in a scale of 0 to 5.

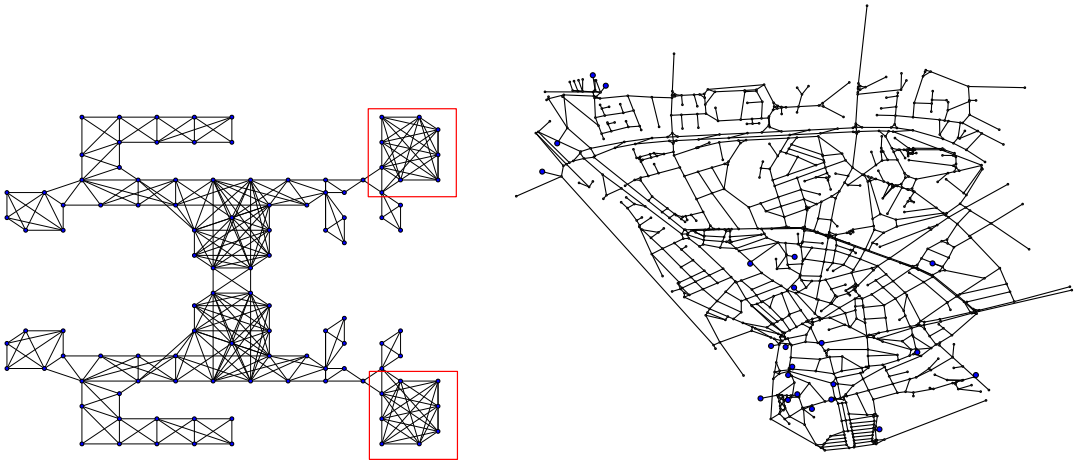


Figure 2.1: Examples of networks: a museum where every node is a PoI on the left and the map of the city of Faro (Portugal) having PoI (blue nodes) and auxiliary nodes.

Given a maximum total visit time (T) and a vector of user's preferences ($UP \in \{0, 1, \ldots, 5\}^m$) with each component associated to each classification class, the main problem is to discover a optimal walk $R = (n_s, n_1, n_2, \ldots, n_k, n_t)$, where n_s is the starting node, n_t is the ending node, and n_i $(i = 1, 2, \ldots, k)$ are the PoI to be visited. Just recall, a walk is defined as any route through a network from node to node along edges which can end on the same node on which it began or on a different node and can travel over any edge and any node any number of times. At this point, we should mention that, to avoid unnecessary computation, the algorithms will use a network $N = (V, E, d, t, C)$ derived from N_0, where V is the set of PoI and possible start and end nodes, $E \subset V \times V$ is the shortest path between the nodes in N_0, and $d : E \to \mathbb{R}_0^+$ is the length of those shortest paths.

The problem is intrinsically multi-objective (Deb, 2001; Miettinen, 1999) where we can optimize several goals, such as: the total walked distance, the number of PoI visited, the total visit time, the time spent while observing PoI, the validity of the walk in terms of observing the users' preferences, etc. Given a walk $R = (n_s, n_1, n_2, \ldots, n_k, n_t)$, this work addresses three objectives which are to be

minimized, namely:

- the *user's preferences cost* given by

$$W_1(R) = 1 - \frac{\sum_{p \in R - \{n_s, n_t\}} \lambda(p)}{|R| - 2},$$

where

$$\lambda(p) = \frac{\sum_{i=1}^{m} C_i(p)^{UP_i}}{\sum_{i=1}^{m} 5^{UP_i}}. \tag{2.1}$$

C_i is the classification value for the i-th category and UP is a vector of user's preferences. If a PoI satisfies the users' preferences, then λ will be approximately equal to 1. If all PoI in the visit satisfy the users' preferences then $\frac{\sum_{p \in R - \{n_s, n_t\}} \lambda(p)}{|R| - 2}$ will also be approximately equal to 1 and therefore $W_1(R)$ will be approximately equal to 0.

- the *time spent observing PoI cost* given by $W_2(R) = 1 -$ (time visiting PoI)/(total visit time), where the time spent visiting PoI is given by $\sum_{i=1}^{|R|-1} t(R_i)$ and the total visit time is the time spent visiting PoI plus the time to go from PoI to PoI, i.e., $\sum_{i=1}^{|R|-1} t(R_i) + \sum_{i=0}^{|R|-1} d(R_i, R_{i+1})$. If the time spent walking from PoI to PoI is low then (time visiting PoI)/(total visit time) is approximately 1 and therefore $W_2(R)$ is approximately 0.

- the *diversity cost* given by $W_3(R) = 1 - \frac{|R|-2}{\text{total number of PoI}}$. This objective is related with the percentage of PoI visited. A larger number of visits will return $W_3(R)$ near 0.

The solution of a multi-objective problem is a set of trade-off solutions called Pareto set (Deb, 2001; Miettinen, 1999). In the Pareto (or efficiency) order relation, a solution R is said to dominate another solution S , $R \prec S$, when R is not worse than S for all objectives and there is at least one on which it is strictly

better, i.e., considering the 3 objectives (W_1, W_2 and W_3),

$$R \prec S \Leftrightarrow \begin{cases} \forall_{i \in \{1,2,3\}} : W_i(R) \leq W_i(S) \\ \exists_{i \in \{1,2,3\}} : W_i(R) < W_i(S). \end{cases} \tag{2.2}$$

A single "optimal" walk can be obtained by computing the entire Pareto set and then selecting an element from that set. However, the computation of the Pareto set is in general extremely expensive which implies that the end user might by satisfied with an approximation to the Pareto set and in particular he can be pleased with a single solution that observes its interests.

A simpler way to compute a single "optimal" can be achieved by transforming the original multi-objective problem into a single objetive problem by, for instance, redefining the objective function as a weighted sum function. The solution obtained with the weighted sum method is known to be Pareto optimal (Miettinen, 1999). In our case, we designed a slightly different (single objective) weighted function, defined as $F(R) = \omega_1 \times W_1(R) + \omega_2 \times \frac{W_2(R)+W_3(R)}{2}$, where $\omega_1, \omega_2 \in [0,1]$ are weights that can be used to give more importance to one of the objectives and $\omega_1 + \omega_2 = 1$. Notice that cost W_2 and W_3 are correlated and therefore were associated in a single summand.

2.3 Algorithmical Approach

This section explains the algorithmic approach used to design the walks. At this stage, an Ant colony optimization (ACO) algorithms (Dorigo and Stützle, 2004) was selected. ACO algorithms are meta-heuristics based on the collective behavior of the majority of the ant colonies, where sets of agents compute new solutions based on artificial pheromone trails left by the previous agents. Technically, those pheromone trails are numerical values reflecting the best solutions found so far. ACO algorithms have a background of success solving many

Algorithm 1 Ant Colony Optimization Algorithm

Ensure: : (Aproximation) to the optimum solution
1: Set parameters
2: Set pheromone trail, $\tau = [1]$
3: **repeat**
4: **for all** ants **do**
5: Build a solution using pheromone trails and heuristics
6: Apply local search to the solution *# Optional*
7: **end for**
8: Update the pheromone trail using the solutions obtained in step 5
9: **until** stopping criteria is met
10: **return** best achieved solution

multiple objective optimization problems (Cardoso et al., 2011; García-Martínez et al., 2004; Mohan and Baskaran, 2012). The general process can be described as follows. During a set of cycles, a set of solutions based on the pheromone matrices and possible heuristics are computed. These solutions are then evaluated and used to update the pheromone matrices for the next cycle. The overall procedure is supported by the positive and negative feedbacks generated by pheromone updating strategies. Algorithm 1 sketches a general ACO.

The process described in Algorithm 1 is common to the majority of the ACO implemented solutions, varying mainly in step 5. Our approach includes two methods to compute a solutions as explained next.

Build a solution – Method A The first method – Method A – considers a starting (n_s) and an ending (n_t) node to define an initial walk, $R = [n_s, n_t]$. In the next step, for each non visited PoI p the best position k (in terms of walking time) in R is found, and the pair (p, k) is kept in a candidate set, CS, if the total time constraint T is not violated by pushing p into position k of R. Now, (a) if the candidate set is not empty, select the next node and position (p, k) to be placed in the walk (and push it into R) according with the following formula

$$(p,k) = \begin{cases} \arg \max\limits_{(p,k)\in CS} \left[\tau(R_k,p)\tau(p,R_{k+1})\right]^{\alpha} \lambda(p)^{\gamma} & if q < q_0 \\ (p',k') & if q \geq q_0, \end{cases} \tag{2.3}$$

where:

- $\tau(x,y)$ is the amount of pheromone in the path $x \rightarrow y$;
- λ was defined in Eq. (2.1);
- α and γ - are control parameters which allow to give more importance to the pheromone and/or preference factors. For instance a large α will emphasize the use of the pheromone while a large γ will emphasize the users' preferences;
- (p',k') is a node and position pseudo-randomly selected from the candidate list using the probability function

$$P(p',k') = \frac{\left[\tau(R_{k'+1},p')\tau(p',R_{k'+1})\right]^{\alpha} \lambda(p')^{\gamma}}{\sum\limits_{(r,k)\in CS} \left[\tau(R_k,r)\tau(r,R_{k+1})\right]^{\alpha} \lambda(r)^{\gamma}}. \tag{2.4}$$

After inserting the PoI in the walk, reset the candidate set and repeat the previous steps. Otherwise, (b) if the candidate set was empty then walk R is returned, since there is no admissible insertion of a PoI into R, and the method stops. Algorithm 2 outlines the described process.

Build a solution – Method B The second method has similarities with Method A. The method begins by defining an initial walk, $R = [n_s, n_t]$, given a starting (n_s) and an ending (n_t) node. Then, for each non visited PoI p, all admissible insertions position k in R are found, and the pairs (p,k) are kept in a candidate set, CS. (a) if the candidate set is not empty, then select the next node and position (p,k) to be placed in the walk (and push it into R) according with the following

Algorithm 2 Solution computation – Method A

Require: n_s (starting node), n_t (ending node), maximum allowed visit time (T), α, γ, q_0, set of PoI
Ensure: A walk
1: $R = (n_s, n_t)$ *# initial walk*
2: $T_R = d(n_s, n_t)$ *# initial traversing time*
3: **while** True **do**
4: $CS = \emptyset$ *# Candidate set*
5: **for all** not visited PoI, p **do**
6: Find position, k, on the walk that minimizes the total traversing time if the PoI, p, is to be inserted in that position, i.e., $k = \arg\min_{i \in 0,1,\ldots,|R|-1} T_R - d(R_i, R_{i+1}) + d(R_i, p) + d(p, R_{i+1})$
7: **if** inserting p in R does not exceed the maximum visit time **then**
8: $CS = CS \cup \{(p, k)\}$ *# keep the candidate and the position*
9: **end if**
10: **end for**
11: **if** $CS \neq \emptyset$ **then**
12: Use Eq. (2.3) to choose a node p, and respective position k (obtained in step 6), between the candidates in CS.
13: Push p into position k of R
14: Update the traversing time, T_R
15: **else**
16: **return** R
17: **end if**
18: **end while**

formula

$$(p,k) = \begin{cases} \arg\max\limits_{(p,k) \in CS} \frac{[\tau(R_k,p)\tau(p,R_{k+1})]^{\alpha} \lambda(p)^{\gamma}}{[d(R_k,p)d(p,R_{k+1})]^{\beta}} & if q < q_0 \\ (p',k') & if q \geq q_0 \end{cases} \tag{2.5}$$

where β is a control parameter which allows to emphasize an heuristic which favors the insertions of nodes closer to the nodes already present in the walk R. Furthermore, (p', k') is a node and position pseudo-randomly selected from the candidate list using the probability function

$$P(p',k') = \frac{\frac{\left[\tau(R_{k'+1},p')\tau(p',R_{k'+1})\right]^{\alpha}\lambda(p')^{\gamma}}{\left[\left(R_{k'+1},p'\right)\left(p',R_{k'+1}\right)\right]^{\beta}}}{\sum\limits_{(r,k) \in CS} \frac{[\tau(R_k,r)\tau(r,R_{k+1})]^{\alpha}\lambda(r)^{\gamma}}{[d(R_k,r)d(r,R_{k+1})]^{\beta}}}. \tag{2.6}$$

The remaining parameters were already introduced in Eq. (2.3). Now, as in the previous method, after inserting the PoI in the walk, reset the candidate set and repeat the previous steps. Otherwise, (b) if the candidate set was empty then walk R is returned, since there is no admissible insertion of a PoI into R, and the method stops. Algorithm 3 outlines the described process.

Pheromone update The pheromone represents a central role in any ACO algorithm. Used in the building of the solutions, the pheromone trail is updated after each cycle according to formula

$$\tau(e) = \rho\tau(e) + \Delta(e), e \in E,$$

where (i) $\tau(e)$ is the pheromone associated to path e; (ii) $\rho \in [0,1]$ is called the persistence factor ($1-\rho$ is the evaporation factor). The smaller the values of ρ are, the smaller quantity of information, used in one cycle, is transmitted to following cycle; (iii) $\Delta(e)$ is the pheromone reinforcement associated to path e and is computed using formula

$$\Delta(e) = \sum_{S_e} \frac{Q}{F(T)},$$

where S_e are the computed solutions containing path e and Q is a value with the same magnitude of the solutions.

2.4 Results

A set of test were run varying the parameters such that[1] $\alpha, \beta, \gamma \in \{0,1,3\}$, $q_0 \in \{0, 0.75\}$, 2-OPT local optimizer (Croes, 1958) turned on and off, $\rho = 0.9$, and $\omega_1, \omega_2 \in \{0, 0.5, 1\}$ ($\omega_1 + \omega_2 = 1$). For each set of parameters 25 runs were made

[1] β is not used in Method A

Algorithm 3 Solution computation – Method B

Require: n_s (starting node), n_t (ending node), maximum allowed visit time (T), α, γ, q_0, set of PoI
Ensure: A walk

1: $R = (n_s, n_t)$ *# initial walk*
2: $T_R = d(n_s, n_t)$ *# initial traversing time*
3: **while** True **do**
4: $CS = \emptyset$ *# Candidate set*
5: **for all** not visited PoI, p **do**
6: **for** k = 0,1, . . . , |R|-1 **do**
7: **if** $T_R - d(R_k, R_{k+1}) + d(R_k, p) + d(p, R_{k+1}) < T$ **then**
8: $CS = CS \cup \{(p, k)\}$ *# Check if p can be placed at position k without exceeding the maximum visit time and update CS*
9: **end if**
10: **end for**
11: **end for**
12: **if** $CS \neq \emptyset$ **then**
13: Use Eq. (2.5) to choose a node p, and respective position k (obtained in step 8), between the candidates in CS.
14: Push p into position k of R
15: Update the traversing time, T_R
16: **else**
17: **return** R
18: **end if**
19: **end while**

with 25 cycles of 25 ants over the network present in Fig. 2.1 left. Each PoI in the network was classified according with eleven categories. The network presents two areas, inside the red rectangles, which were categorized as highly adequate for senior people and also with high classifications in "sixties photography". The remaining categories and PoI where classified randomly. The visit time (VT) was also generated randomly, except for two of the previously classified PoI (one in inside each red rectangle) which were defined has having a large visit time.

Since it is impracticable to present all results, Tab. 2.1 resumes the best mean (and standard deviation) results for the W_1, W_2, and W_3 cost functions, presented in Section 2.2. Furthermore, besides the methods parameters ($\alpha, \beta, \gamma, q_0, \omega_1, \omega_2$ and 2-OPT on/off), the aggregated cost function (F), the number of visited PoIS and time spent observing PoI (from 90 time units) are also presented. The last

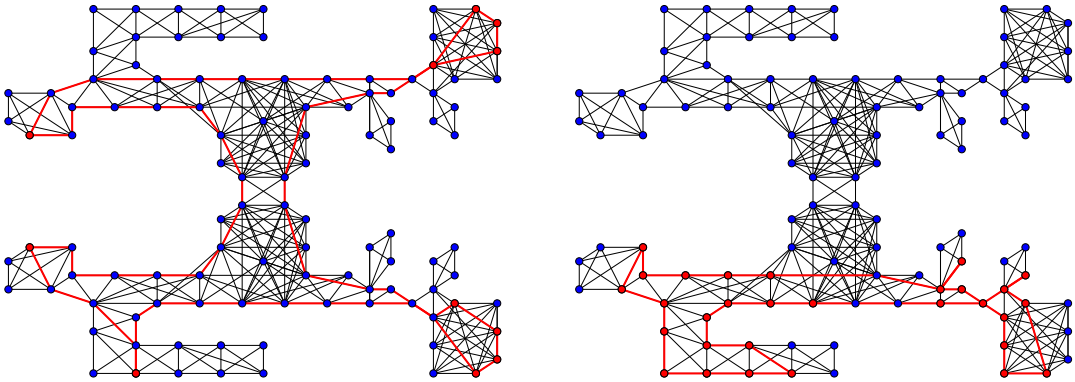
	Method	W_1	W_2	W_3	**PoI VT**	No. PoI	F	α	β	γ	q_0	**2-OPT**	ω_1	ω_2
Best results for W_1	B	,002(,00)	,484(,04)	,899(,01)	39,3(2,5)	11,0(,8)	,002(,00)	1	0	3	0,75	True	1	0
	B	,002(,00)	,491(,03)	,899(,01)	39,1(1,9)	10,9(,7)	,002(,00)	3	0	3	0,75	True	1	0
	B	,002(,00)	,568(,02)	,900(,01)	38,8(1,7)	10,8(,8)	,002(,00)	1	0	3	0,75	False	1	0
	B	,002(,00)	,568(,02)	,899(,01)	38,7(1,8)	10,9(,8)	,002(,00)	1	0	1	0,75	False	1	0
	B	,002(,00)	,572(,02)	,902(,01)	38,4(1,6)	10,6(,6)	,002(,00)	3	0	3	0,75	False	1	0
	B	,002(,00)	,440(,02)	,907(,00)	37,0(,0)	10,0(,0)	,002(,00)	0	0	1	0,75	True	1	0
	B	,002(,00)	,454(,03)	,907(,00)	36,8(,4)	10,0(,0)	,002(,00)	0	0	3	0,75	True	1	0
	B	,002(,00)	,588(,00)	,907(,00)	37,0(,3)	10,0(,0)	,002(,00)	0	0	1	0,75	False	1	0
	A	,003(,00)	,310(,00)	,833(,00)	61,8(,0)	18,0(,0)	,003(,00)	0	–	3	0,75	True	1	0
Best results for W_2	B	,783(,02)	,110(,02)	,702(,00)	75,0(,7)	32,2(,5)	,423(,00)	1	3	0	0,75	True	0	1
	B	,823(,04)	,114(,03)	,720(,01)	72,2(1,3)	30,2(,6)	,431(,00)	0	1	0	0,75	True	0	1
	B	,822(,03)	,116(,03)	,706(,01)	74,4(1,1)	31,8(,8)	,427(,00)	1	1	0	0	True	0	1
	B	,827(,04)	,120(,02)	,697(,01)	76,2(,8)	32,7(,7)	,419(,00)	1	3	0	0	True	0	1
	B	,825(,04)	,121(,02)	,711(,01)	71,7(,8)	31,2(,8)	,428(,00)	0	3	0	0	True	0	1
	B	,833(,05)	,123(,02)	,703(,01)	76,1(1,2)	32,1(,7)	,421(,00)	1	1	0	0,75	True	0	1
	B	,815(,05)	,128(,02)	,700(,01)	76,4(,6)	32,4(,8)	,420(,00)	3	3	0	0	True	0	1
	B	,816(,05)	,134(,02)	,709(,01)	75,7(1,2)	31,5(,8)	,425(,00)	3	1	0	0,75	True	0	1
	B	,811(,03)	,135(,01)	,703(,01)	75,5(,8)	32,0(,7)	,424(,00)	3	1	0	0	True	0	1
Best results for W_3	A	,757(,02)	,173(,01)	,688(,01)	73,8(1,0)	33,7(,7)	,431(,00)	3	–	0	0	False	0	1
	A	,764(,04)	,166(,01)	,690(,01)	74,0(1,2)	33,4(1,2)	,431(,00)	3	–	0	0	True	0	1
	A	,770(,03)	,167(,01)	,692(,01)	74,2(1,1)	33,3(1,0)	,430(,00)	1	–	0	0,75	True	0	1
	A	,759(,03)	,176(,01)	,694(,01)	73,9(,9)	33,0(1,0)	,434(,00)	1	–	0	0	False	0	1
	A	,792(,03)	,174(,01)	,696(,01)	73,9(1,1)	32,9(1,0)	,434(,01)	1	–	0	0,75	False	0	1
	B	,827(,04)	,120(,02)	,697(,01)	76,2(,8)	32,7(,7)	,419(,00)	1	3	0	0	True	0	1
	A	,775(,03)	,220(,02)	,697(,01)	69,6(1,5)	32,8(,9)	,459(,01)	3	–	0	0,75	True	0	1
	A	,788(,04)	,227(,02)	,697(,01)	69,1(1,4)	32,7(1,3)	,463(,01)	3	–	0	0,75	False	0	1
	B	,815(,05)	,128(,02)	,700(,01)	76,4(,6)	32,4(,8)	,420(,00)	3	3	0	0	True	0	1

Table 2.1: Best results for costs W_1, W_2, and W_3 (mean and standard deviation) and VT the visit time.

two values are shown since they are more "legible" values. Finally, we should recall that all costs were to be minimized.

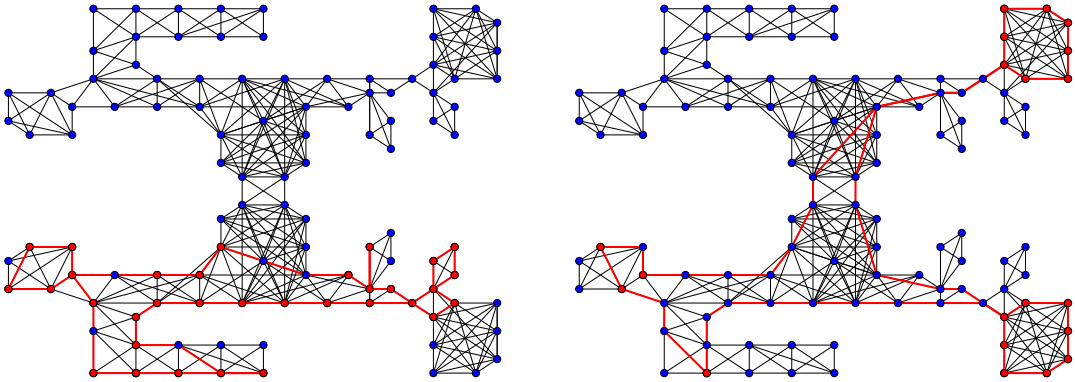
From Tab. 2.1 we can draw some conclusions. The best result for W_1 show solutions where the mean number of visited PoI is equal to 11 (Fig. 2.2a shows a typical result for the best set of parameters). On the other hand, if we choose the best results for W_2 or W_3 the mean number of visited PoI raises to over 30 (Fig. 2.2b and Fig. 2.3a show typical results for the best sets of parameters). Similar, the visit time expended observing PoI is much smaller (a mean around 39.3 of the 90 times units) when we consider the best results of W_1, against the best results of W_2 and W_3 (a mean around 70 of the 90 times units).

The best results for the W_1 cost were naturally obtained for $\omega_1 = 1$ and $\omega_2 = 0$. On the other hand, it was also natural that the best results for the W_2 and W_3 costs were obtained for $\omega_1 = 0$ and $\omega_2 = 1$. In this sense, Tab. 2.2 presents the best values for the aggregate cost function F when a balanced preference set was



(a) Walk for best W_1 (b) Walk for best W_2

Figure 2.2: Examples of a walk in a museum for the best results on (left) W_1 and (right) W_2



(a) Walk for best W_3 (b) Best walk when $\omega_1 = \omega_2 = 0.5$

Figure 2.3: Examples of walks in a museum for the best results of (left) W_3 and (right) $\omega_1 = \omega_2 = 0.5$

considered between the the costs weights, i.e., $\omega_1 = \omega_2 = 0.5$. When compared with the previous results, the resulting walks are more balanced in the sense that the mean number of visited PoI is around 20 and the PoI visit time around 60 time units of the maximum 90 allowed. Figure 2.3b shows a typical results for the best set of parameters.

As final observations, the 2-OPT local optimizer presents a important role as the majority of the best solutions were obtained when it was active. Tables 2.2 and 2.1 also show that Method B appears with more frequency in the best results, although the difference in terms of costs were not expressive. Never-

Method	W_1	W_2	W_3	**PoI VT**	No. PoI	F	α	β	γ	q_0	**2-OPT**	ω_1	ω_2
B	,019(,02)	,234(,05)	,822(,01)	62,6(1,9)	19,2(,9)	,283(,01)	1	1	3	0	True	0,5	0,5
B	,013(,01)	,253(,03)	,825(,01)	61,5(2,3)	18,9(,9)	,283(,01)	1	0	3	0,75	True	0,5	0,5
B	,030(,01)	,238(,05)	,817(,00)	64,2(,9)	19,8(,5)	,285(,00)	1	1	3	0,75	True	0,5	0,5
A	,023(,01)	,278(,02)	,817(,00)	64,8(2,1)	19,7(,4)	,285(,00)	1	–	3	0	True	0,5	0,5
A	,021(,01)	,280(,02)	,819(,01)	64,7(2,2)	19,6(,6)	,285(,00)	1	–	3	0	False	0,5	0,5
A	,014(,01)	,292(,02)	,820(,00)	63,6(2,0)	19,4(,5)	,285(,00)	1	–	3	0,75	False	0,5	0,5
A	,003(,00)	,316(,00)	,824(,00)	61,2(,2)	19,0(,0)	,286(,00)	0	–	3	0,75	True	0,5	0,5
A	,003(,00)	,317(,00)	,824(,00)	61,3(,3)	19,0(,0)	,286(,00)	0	–	3	0,75	False	0,5	0,5
B	,016(,01)	,229(,07)	,839(,01)	57,7(3,4)	17,4(1,3)	,286(,00)	0	1	3	0	True	0,5	0,5

Table 2.2: Best results for costs F when $\omega_1 = \omega_2 = 0.5$ (mean and standard deviation) and VT the visit time.

theless, method B is computationally more demanding than method A which might pose a doubt which method to use in a real-time application with many accesses.

The same algorithms were applied to the city of Faro, using the best parameters for F and $\omega_1 = \omega_2 = 0.5$, i.e., $\alpha = \beta = 1$, $\gamma = 3$, $q_0 = 0$, and 2-OPT activated. The network has 24 PoI, classified between 1 and 5 in 11 categories, e.g., "shopping", "museum", "church", "theater", or "monument". The visitor was characterized has looking for "churches" (church preference was set to 5 and the others between 0 and 2) and having 1500 time units to spend. Figure 2.4 shows the results for a single run of each of the methods: method A on top and method B on bottom. The results for both methods are similar with Method A obtaining $W_1 = 0.040, W_2 = 0.417, W_3 = 0.636$ and $F = 0.284$ and Method B $W_1 = 0.040, W_2 = 0.409, W_3 = 0.636$ and $F = 0.281$. In both cases, the solutions starts in the same node and passes through six PoI classified as churches.

Figure 2.5 presents the objective values of all obtained solutions (in blue) and the red points form the set of non-dominated solutions in the objective space (recall Eq. (2.2))

2.5 Conclusions

The way people experience cultural heritage is changing. Traditional visits where everyone, despite their interests or limitations, have to follow a predetermined

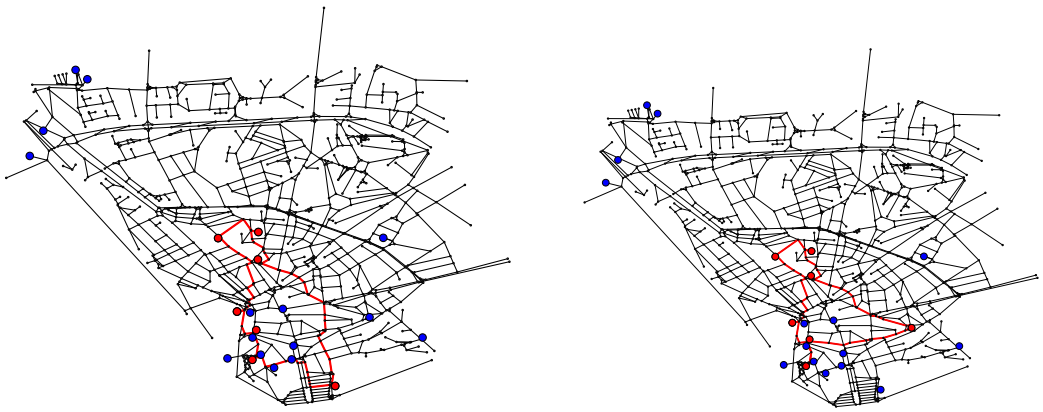


Figure 2.4: Example of walks in the city of Faro for the best parameters (for F and $\omega_1 = \omega_2 = 0.5$): method A on top and method B on bottom.

route are not the best way anymore. Furthermore, many times the number of Points of Interest is also large and the time available to explore them is limited, making impossible to experience them all. Another important objective is to enhance accessibility and fighting info-exclusion.

This chapter proposes a problem formulation for the routing inside a network with a set of PoI and explores two methods supported on the ACO algorithms to build, in near real time, walks which translate the user's preferences and limitations. In this sense, both the network and the users' preferences have to be designed with the same insight. The results shown good solutions which translate the data and the preferences.

As future work, naturally, many things can be further developed and tested. For instance, it can be useful to carry out exhaustive stress test like larger networks, larger number of categories, implementation of the methods in computational devices with less capacity (like mobile devices), etc. The features aspect can also be further explored. One idea is to adapt the walk as the users navigates the network. For instance the user might spend more time near a particular piece, or simple make a pause, which, given the limited visit time, must be reflected in the proposed walk. The walk should also adapt to users way of exploring the space. If the users spends much time near a piece which is classified

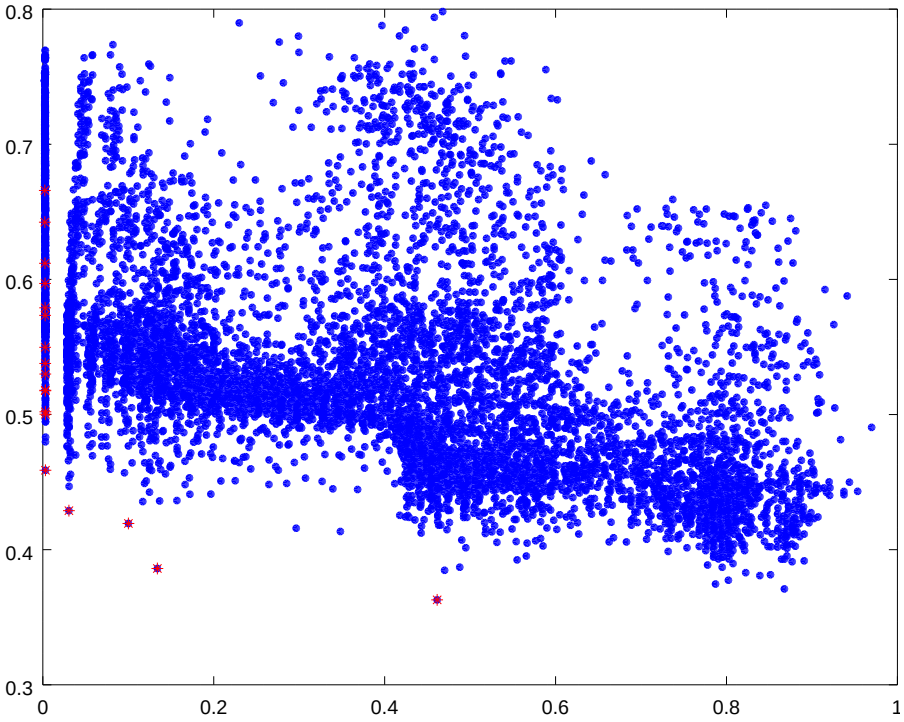


Figure 2.5: Objective values of all obtained solutions.

"outside" the initial preferences, maybe the application should suggest other pieces of that type. Other features include the constructions of the walks based on the expected "occupation" of the PoI, reservation of time slots to the more wanted PoI or past information collected from users with similar interests.

3

Adaptive Card Design UI Implementation for an Augmented Reality Museum Application

Abstract

Museums are great places where visitors can see, hear, touch, feel and experience interesting things. The visit is even better when visitors can select what they want to see and have ways to enhance their experience. Many museums have a huge amount of collections and objects, selecting which ones to see is

sometimes difficult. A system that adapts on the fly to the user's preferences, suggesting objects that he might want to see, paths he would like to follow in their visit, as well as the complementary information he needs about each object, will be of fundamental importance. Smartphones, with their Apps are the best solution to help enhance the museum experience, nevertheless, most of the time they fail, because their user interface (UI) does not adapt to the user's preferences. This chapter presents: (a) an initial framework for a museum application where augmented reality and gamification are connected with an adaptive UI, (b) an adaptive card implementation to realize the UI, and (c) an initial fast object recognition implementation for the markers used for the augmented reality.

3.1 Introduction

M5SAR (Mobile Five Senses Augmented Reality System for Museums) project aims at the development of an Augmented Reality (AR) system, which consists of an application (App) platform and a device ("gadget" - hardware), to be connected to mobile devices, in order to explore the 5 human senses (sight, hearing, touch, smell and taste). The system is to be a guide in cultural, historical and museum events, complementing or replacing the traditional orientation given by tour guides, directional signs, or maps.

The number of mobile Apps, including the ones that use AR, are increasing due to the popularity of built-in cameras and global positioning systems. The massive availability of Internet connections on mobile devices also enables, the construction of personal context-aware cultural experiences (Jung et al., 2015).

In the present and in the future, User Interfaces (UI) is a fundamental research area, where at least four (sub-)areas interconnect: Human-Computer Interaction (HCI), Artificial Intelligence (AI), User Modeling (UM) and Interaction Design (IxD). The core of the investigation in the near future should fall,

most probably, in the usually called Intelligent User Interfaces (IUI) or Adaptive User Interfaces (AUI) and on the Automatic-Generation of Interfaces (AGI), connected with the best practices of IxD, user experience (UX) and Emotional UI (EUI). AUI should be enhanced with accessibility features, and can also be enhanced with AR and Gamification features.

The UIs traditionally follow a one-size-fits-all model, ignoring the needs, abilities and preferences of individual user's. However, research indicated that visualization performance could be improved by adapting aspects of the visualization to the individual user (Steichen et al., 2014). As Conati et al. (2015) stated, intelligent user-adaptive interfaces and/or visualizations, that can adapt on the fly to the specific needs and abilities of each individual user, is a long-term research goal. This is due to two main reasons: (a) the difficulty of extracting information about the users needs and abilities, and (b) the implementation of the UI that can adapt/change "itself" on the fly. Alvarez-Cortes et al. (2009) define IUI as a sub-field of HCI with the goal of improving the HCI by the use of new technologies and interaction devices, including the use of AI techniques that allow adaptive or intelligent behavior. Akiki et al. (2014) presented a study about adaptive model-driven UI development systems. Gajos and Weld (2004) proposed an automatic system for generating UI, i.e., solution based on treating interface adaptation as an optimization problem.

Reinecke and Bernstein (2013) refer that a modular UI, that allows a flexible composition from various interface elements, increases the number of variations of the interface to the power of the number of adaptable elements. Thus, instead of designing each interface from scratch, a modular user interface approach is a possible good solution, since it allows achieving many more versions with less design and implementation effort. Equally important is to adapt the UI to users with different visual, auditory, or motor impairments. Unfortunately, because of the great variety of individual incapabilities among such users, manual mod-

ular designing interfaces for each one of them is impractical and not scalable (Gajos et al., 2008; Rodrigues et al., 2016). However, the modular and/or adaptive generation of UI offers the promise of providing personalized interfaces on the fly, but this does not mean that the user will be satisfied with his/her personalized App. According to Zhao et al. (2012), the psychological process behind satisfaction is highly complex and requires a differentiation between transaction-specific satisfaction and cumulative satisfaction. Nevertheless, mobile Apps should move towards completely personalized experiences. These experiences usually are built from the aggregation of many individual pieces of content.

Having all the above in mind, at least three main challenges arise in the UI design and implementation: (a) how to harvest the necessary information about each user preferences and skills (without asking them to fill any form). (b) From the acquired information/data, how to give "intelligence" to the UI to adapt on the fly to the user's changes (e.g., to the user mood). (c) How to develop this adaptive UI, even a modular UI, without being necessary to develop a huge amount of different (sub-)modules, and at the same time still optimize the user experience (UX) and the main principles of interaction design (IxD), i.e., how to implement Automatic-Generation of Interfaces. One way these challenges can be addressed is as cards (Adobe, 2016; Babich, 2016) based UI. Card-based interaction model is not new and is now spreading pretty widely in most of the recent Apps.

This chapter also focus in the implementation of AR App. The present solution is an AR marker-based method, often also called image-based (Cheng and Tsai, 2013). AR markers-based allow adding preset signals (e.g., paintings, statues) easily detectable in the environment and use techniques of computer vision to sense them. The use of AR in museums is not new, including the implementation of head-worn displays (HWD) (Vainstein et al., 2016). Other AR solutions

are also available see e.g. Rodrigues et al. (2016). There are many commercial AR toolkits (SDK) such as Vuforia (2016) and AR content management systems, e.g. Catchoom (2016), including open source SDKs, probably the most know is ARToolKit (2016). Each of the above solutions has pros and cons, some are quite expensive, others consume to much memory (it is important to stress, that our application will have many markers, at least one for each museum piece), others take too much time to load in mobile devices, etc. Here, we also focus on the initial development of an image marker detector, which will be based on the ORB binary descriptor (Rublee et al., 2011).

The main contribution of this chapter is a framework for the adaptive on the fly card-based UI construction, where the development of the cards has a modular architecture. In addition, an initial patch-based marker architecture for fast AR is also presented.

3.2 Adaptive card implementation

One of the objectives of this work is to develop a methodology to UIs that can adapt on the fly to each user. In particular, this section presents the architecture to create the card-based UI on run-time.

To have a full adaptive UI, we could have (at the limit) a different layout and content for each UI view and user. Nevertheless, different users could have the same layout or at least partial similarly layouts. The same layout and structure can also be used in multiple views (e.g., when showing information about different paintings to the same user), usually, in this case, the only thing that could change are the contents to display to the user. Of course, even when the layout is the same for different users the content could be different.

In this context, and with the principle of adapting the UI on the fly, makes no sense in terms of App memory and CPU optimization, to build each layout

(or partial layout) from scratch every time it is required. If a layout (or partial layout) is created once, and expected to be used more than one time, this should be kept in memory instead of creating it when needed (it is important to stress that the methodology presented here was tested and developed using Unity (2017) development platform).

To achieve this, we decided to separate a *view* in (A) structure/layouts and (B) contents. This means that, the application will no longer create views but will instead make card-layouts and place different contents on the (same) card-layout at different execution points, since the (different) layouts and structures are used multiple times.

To build the structure/layout (A), an engine was created to assemble the card-layout data structure. The engine uses as input a “layout-tree" data structure, where the basic layout units, called *content format*, are joined together in *cells*, which could be joined (again) as *templates*. Both cells and templates are joined together until a card-layout is formed. Thus, each card-layout is composed by one or several cells, plus zero to several templates, that can be used in different card-layouts of the same App (the template has one or several cells, and each cell has one or several content format).

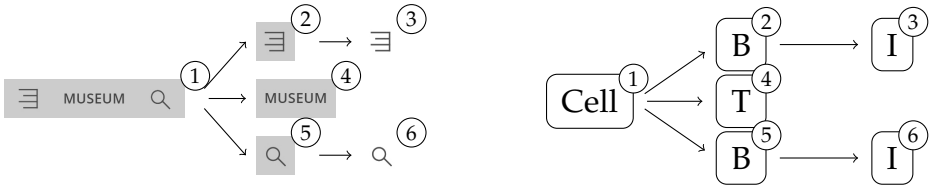
In more detail, the card-layouts are assembled in a tree structure since they represent a parent-child relationship. Figure 3.1a sketches the disassembled view of a card layout data structure, and the corresponding block diagram in Fig. 3.1b. In the figure every box represents a node, and the number in the top right corner its identifier.A tree node can be from one of three categories: (a) a content format, (b) a cell or (c) a template. Common to both the content format and the cell categories are some *properties*, like the dimensions of the node and its responsiveness behavior.

A content format (a) represents the formatting of a content (the basic unit of the card-layout), be it a text, an image, etc. Each specific content has its own

properties. For example, a *text content format* (represented as T in Fig. 3.1b) has properties that define the font, the line spacing, the text color, etc. They also define the location where the content will appear. An *image content format* is represented by an I, and a *button content format* by a B in the same figure. A cell (b), or stack layout (Xamarin, 2016), is a node that, unlike the content formats, does not convey any information to the user, as each cell is used to organize the contents. A cell divides the children into a single line that can be oriented horizontally or vertically and gives the appropriate spacing between them. A cell child can be any of the categories aforementioned.

A template node (c), is a special node that integrates another preexisting template, i.e. a group of already structured cells and contents. This node is useful in situations where a determined structure is repeated several times, like for example the menu template shown on Fig. 3.1a and used in Fig. 3.2 (highlighted in blue). Each template can be used in any card-layouts of the App.

In the construction of the card-layout (see Fig. 3.2), inside the tree terminology, two terms are important: the root node (the node of the tree from which all other nodes - children - descend) and the leaf node (a node that has no children). Two rules were established and must be followed while creating a layout tree: (i) the root node ("view") must always be a cell; (ii) a leaf node must be a content or a template. Regarding (i), the root node can not be a template node, because this would mean that the new tree would be a copy of the referenced template. The root node also can not be a content format, since each template and the final card-layout should be an agglomerate of multiple contents that are organized in some shape or form. Concerning (ii), a leaf node cannot be a cell node, because its (cell) purpose is to arrange its children (if it has no child then it makes no sense to have it sinse it would be to add excessive information that needs to be sent and processed). Finally, it is important to stress the specificity of the button content format, the button itself only represents the click action and requires a



(a) Disassembled view. (b) Block diagram.

Figure 3.1: Menu tree diagram.

child to provide visual elements to the user. These elements can be any of the categories mentioned before.

When assembling the card-layout we opted for a depth-first approach. With this approach, the card layout build engine was implemented to work in a recursive manner: (1) create the node and set its properties; (2) processes each child node (step 1) and, (3) establish the parent-child relationships (we stress again that this process was tested and developed using Unity (2017) platform).

When a view, a card-layout with contents is needed, the application simply adds to the card-layout already instantiated the contents (B). If another view requires the same template it uses the same card-layout in memory and just changes the contents.

Figure 3.2 illustrates the build process of a card-layout. In this case, the root node of the tree is a cell that is divided vertically and whose children are represented by the dashed lines. The first child of this view is the *Menu template* as displayed in Fig. 3.1, but with different contents. The Menu template is assembled as follows (see Fig. 3.1): start by instantiating the root cell horizontally divided (node 1) and define its properties like the horizontal alignment. Next create the button content format (represented by B on node 2), followed by its child image (I on node 3). At this point the relationships are established, node 3 defines its parent as node 2, and node 2 its parent as node 1. Next, moving to the 2nd child of the Menu template root node, which in this case is a text content format (node 4), create it, specify its attributes and then set its parent as node 1.

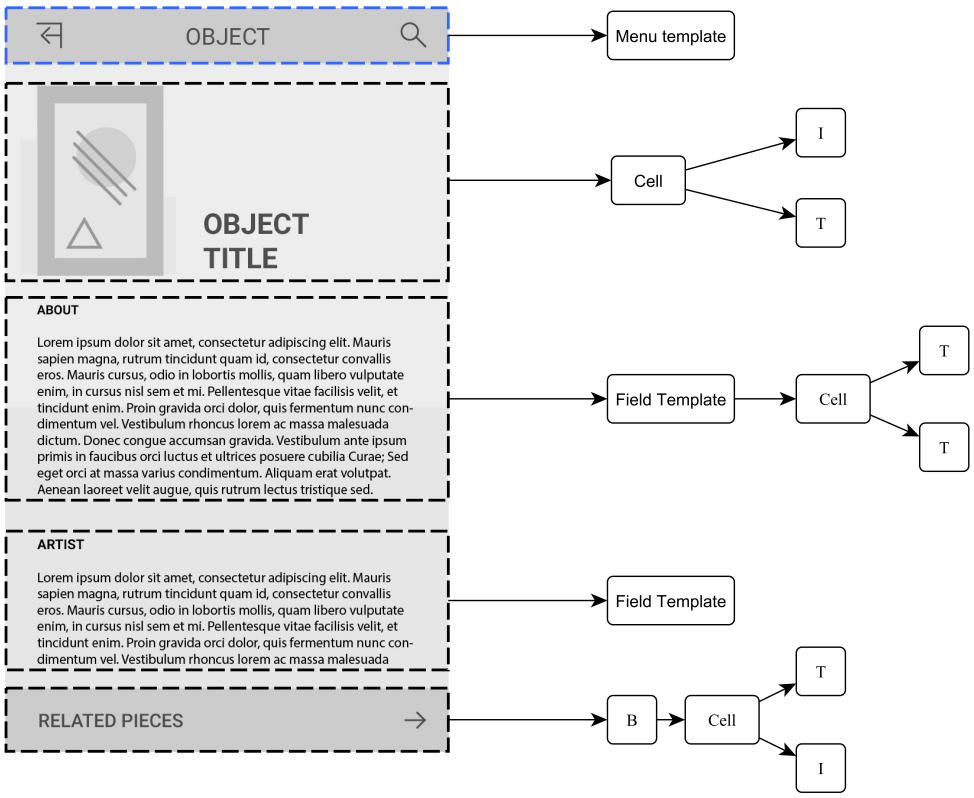


Figure 3.2: Example of a museum object view.

Lastly, nodes 5 and 6 are processed in a similar fashion to nodes 2 and 3. This template is now ready to be used at any time. The next node of the card-layout is a new cell node that contains an image and a text content format. The next two nodes both contain a *Field template*. This Field template is a simple template that includes two text content format arranged vertically. Finally, there is a button whose child is a cell that has a text and an image. Each of these cells follow the building principles, which were early explained.

It is very important to stress the function of the database (DB) as a fundamental component of this system, since it is where ("harvested") user information/specifications are kept, that are then converted (not presented or discussed in this chapter) to the correspondent specifications for each user card-layout and card-contents (also stored in the DB). In this chapter we only focus on the part of the DB related to the card-layout. The database for the card-layout implemen-

tation follows the exact same tree architecture and it can be subdivided in three major layers: (a) components, (b) formats and (c) structure.

The components layer (a) is the simplest one, where we define basic properties like colors, fonts, shadows, outlines and backgrounds. The formats layer (b) is where we indicate the type of content to be used in a child node and where we store the information related to that specific type of content, whether it is an image, a text or a button, using previously created sets of component properties. Here we can also override a particular property if needed. Then, there is the structure layer (c), where the parent-child relationships of our tree architecture are defined, node by node. It is also used to save data regarding layouts and cells, like orientation and spacing. All this information can be aggregated by templates, therefore they may be reused later, optimizing the process of creating new views.

When a new view has been added to the database, we need to convert it to a JSON format and store it, so that it can be requested by the application installed in the mobile device (see Fig. 3.3). For the JSON generation process we are running a script on the server side that receives the new template index and then connects to the database to build up the entire tree. It navigates from table to table, node by node, in the same manner that it was described for Fig. 3.1b. At the end of the process, the script saves the file in the DB with a time stamp, this way the App can determine whether or not that is the most recent version for that template, and if it is not, it can simply download the new JSON document. The simplified block diagram for the UI generation can be seen in Fig. 3.3.

3.3 Fast mobile object detection and tracking

In the present App a huge number of cards, the ones that describe museum objects (e.g. paintings or statues) only appear in the presence of the object, i.e.,

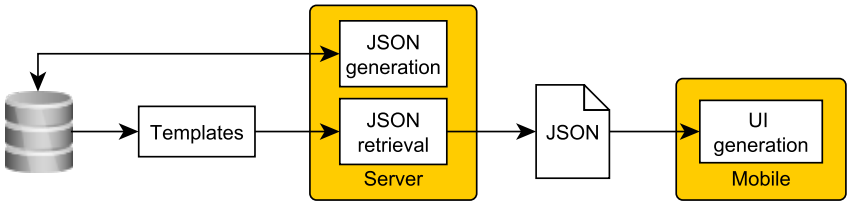


Figure 3.3: UI generation overview.

when the camera is pointed to the object. Here, we focus on object detection, recognition and tracking, with the purpose to call the respective card view.

There are many solutions (see Sec. 3.1) to detect a museum object and deploy the AR respective "card". Those solutions use what is called "markers" (ARToolKit, 2016), which in a simplified way, are photographs (one or more) from the original object, that work as a template (see below). By using Computer Vision algorithms, they are compared with the frames captured by the mobile camera and trigger (when recognized) the identifier for the object as well as its position on the mobile screen. Here, we focus on a solution, with three goals: (a) speedup the process of downloading the markers into the mobile device, (b) do all the recognizing process in the mobile, reducing the server requirements, and (c) try to minimize power and memory consumption when doing the recognition.

Before applying our mobile object detection algorithm, museum objects were photographed and stored in a server using high quality Full HD images. Those are called image *templates* for the object. While for paintings a single photograph was used, for statues several photographs were used to represent the object. For the marker recognition implementation, it is required a reliable and fast descriptor since we aim to compute it on a mobile device. For that reason, we have opted to use the ORB descriptor (Rublee et al., 2011) for object recognition.

Before we start to explain the algorithm, we define *patch* as a section of the original image, of size $N \times M$ pixels (px); see Fig. 3.4 top-right row. The algo-

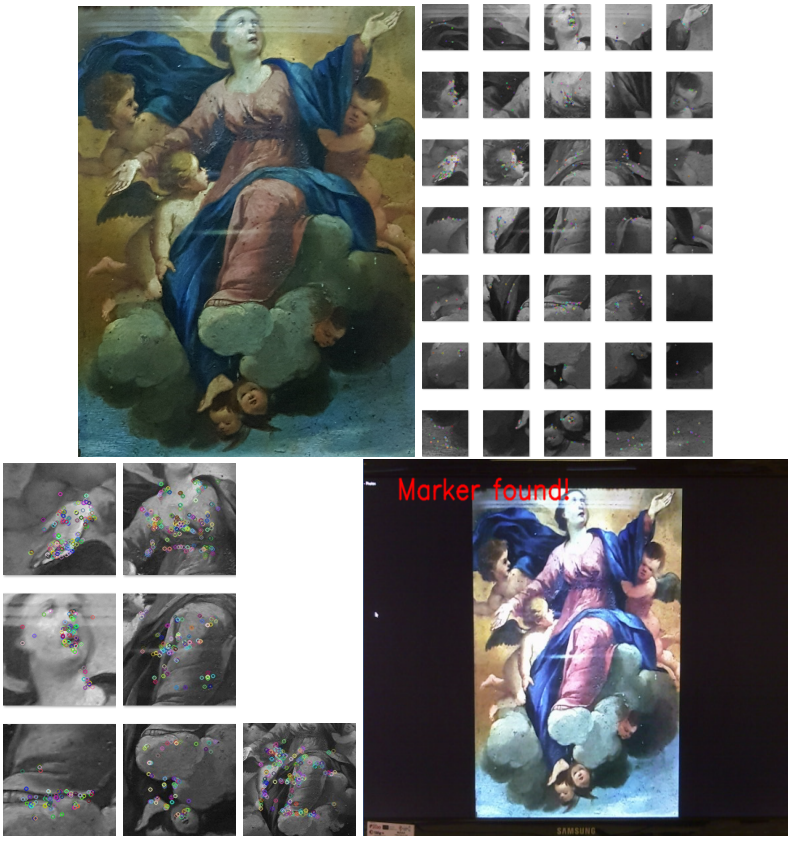


Figure 3.4: Marker template and its patches extracted, and a marker template matched. Top to bottom, left to right: Marker template (low res.), original image divided in patches, 3 most relevant patches from the full-size image, the 1/2 and from 1/4 size image. Bottom-right, object matched.

rithm works as follows: (i) Over the template. (i.1) Compute ORB descriptor (Rublee et al., 2011); (i.2) Divide the template in patches, and extract patches with keypoints and respective descriptors; (i.3) Sort the patches by keypoint/descriptor importance, and select the K most relevant extracted patches - those are to be used as *marker patches*; (i.4) repeat steps (i.1) to (i.3), now with the image size divided by 2 and by 4 (3 scales). (i.5) Group and sort the patches from the different scales, with total number of marker patches per template, $\gamma \leq 3 \times K$ ("</=", depends on the original size of the template).

(ii) On the object recognition and tracking: (ii.1) Acquire a frame from the mobile camera and apply the ORB descriptor; (ii.2) Test the frame using the *most relevant marker patch* from the 3 scales grouping for each of the available

templates (see step i.5); (ii.3) Select the template based on best classification; (ii.4) Test the object recognition using all (γ) patches from the correspondent template, matching "template-frame"; (ii.5) Object is recognized, when ratio validation threshold is verified; (ii.6) Flag if object found. After this point the object is only tracked (not tested for recognition). (ii.7) Track object based only on a valid marker patch from selected template; (ii.8) Restart the recognition process again if tracking time threshold is met.

As mentioned, in the initial step (i.1) for each object and its respective template(s), the ORB descriptor is applied for each of the 3 template scales (Full HD, $1/2$ and $1/4$ size). (i.2) Then, starting from the middle of the template image, each template is divided in patches (best results were obtained for $N = M = 200$ px); see Fig. 3.4 top-right row. The reason for starting the template division in patches from the centre, is because there is a higher probability it will have richer keypoints regions. If necessary, border regions from the template image are ignored. (i.3-4) The extracted patches are then sorted by the number of keypoints in each patch, in descending order, until $K = 5$ patches are stored per scale. This is repeated for the template with $1/2$ and $1/4$ of the size. This process allows farther and shorter validation distances when targeting the mobile camera onto an object. (i.5) All marker patches (γ) from the 3 scales are then grouped in descending order based on the keypoints count and stored on a object template dataset. On the mobile device, each time a frame is captured by the mobile camera (ii.1) ORB descriptor is applied, after which is matched against the object template dataset, that contains all (γ) marker patches grouped. The classifier matches the frame with the most relevant patch of each marker, (ii.2) to get the match count of similar keypoint descriptor. The marker template which has the highest match count is validated against the threshold of minimum match count ($T_{mc} = 1$) required (ii.3) for advancing to the following stage.

(ii.4) After 1 marker descriptor patch validated, all patches from the selected

marker template are matched, (ii.5) and a ratio is calculated between the number of patches validated (MP_v) and the total length (γ), i.e., $r = MP_v/\gamma$. (ii.6) On ratio validation threshold validated, $T_{rv} = 0.1$, the object is recognized. After the object being recognized, and while it is in the field of view of the camera, we only need to tack it. This is a process less CPU demanding than recognition. Now, for each frame acquired, (ii.7) we only match the frame with a single valid marker patch. This steps continues until the object disappears from the field-of-view for more than $T_t = 1$ second of the camera. If this occurs then the tracking step stops (ii.8) and the recognition process starts again (steps ii.1 to ii.6).

3.4 Conclusions

In this chapter we present an initial framework for the development of an architecture capable of producing an adaptive UI (for a museum application), the focus was on the creation process of a card-based UI, where the development of the cards has a modular architecture. In addition, it was also presented a patch-based marker architecture for mobile object recognition with application in the realm of AR.

Despite both systems being still in an initial stage of development, both present satisfactory results. For future developments, we will focus on how to harvest the necessary information about each user preferences and skills, and from the acquired information/data, how to give "intelligence" to the UI to adapt on the fly to the user's changes. In the case of the mobile object recognition system, it can at the moment achieve real time recognition of 50 different objects, being the goal in the future to achieve at least 100 objects recognition in real time.

4

MIRAR: Mobile Image Recognition based Augmented Reality Framework

Abstract

The application of the most recent technologies is fundamental to achieve sustainability in tourism, as well as in other economic sectors. This chapter presents the initial architecture of MIRAR, a Mobile Image Recognition based Augmented Reality framework, which aims at the Augmented Reality for mobile applications development. The MIRAR framework allows the development of a system which uses mobile devices to interact with museum's objects. By using the mo-

bile device's camera, the system recognises and tracks on-the-fly, on the client side (mobile), museum's objects. In addition, the initial discussion on how the MIRAR framework detects environments allowing the superimposition of information over those is presented.

4.1 Introduction

The World Tourism Organization (UNWTO, 2017) defined sustainable tourism as "Tourism that takes full account of its current and future economic, social and environmental impacts, addressing the needs of visitors, the industry, the environment and host communities." The only way to achieve this definition is to apply new technologies to this economic sector.

Augmented Reality (AR) is a technology that, thanks to the mobile devices increasing hardware capabilities, quickly evolved in the recent years, gaining a huge amount of users (Azuma et al., 2001). AR empowers a higher level of interaction between the user and real world objects, extending the experience on how the user sees and feels those objects, by creating a new level of edutainment that was not available before. The M5SAR: Mobile Five Senses Augmented Reality System for Museums project (Rodrigues et al., 2017b) aims the development of an AR system to be a guide in cultural, historical and museum events. This is no novelty since, almost every known museum has its own mobile applications (App), e.g. InformationWeek (2017); TWSJ (2017). The use of AR in museums is much less common, but it is also not new, see e.g. HMS (2017); Qualcomm (2017); SM (2017); Vainstein et al. (2016). The novelty of the M5SAR project is to extend the AR to the human five senses, see Rodrigues et al. (2017b) for more details.

This chapter focus on MIRAR, Mobile Image Recognition based Augmented Reality framework, one of M5SAR's modules. MIRAR focus on the develop-

ment of a mobile multi-platform AR (Azuma et al., 2001) framework, with the following main goals: (a) perform "all" computational processing in the client-side (mobile device), minimizing, this way, costs with server(s); (b) use real world two- and three-dimensional (2D and 3D) objects as markers for the AR; (c) allow to project contents (e.g., text and media) on the objects displayed in the mobile device's screen, as well as enhance the object's displayed contents, by touching regions on the device's screen; (d) recognise environments, allowing the projection of contents in them; (e) detect human shapes and allow projection of contents on those shapes; (f) use the mobile device's RGB camera to achieve these goals. A framework that integrates all these goals is completely different from the existing (e.g., SDK, frameworks, content management) AR systems such as ARToolKit (2016); Catchoom (2016); Kudan (2016); Layar (2016).

The MIRAR module for object recognition (b) presented in this chapter is an AR marker-based, often also called image-based (Cheng and Tsai, 2013). AR image-based markers allow adding pre-set signals (e.g., from paintings, statues, etc.) easily detectable in the environment and use computer vision techniques to sense them. There are many image-based commercial AR toolkits (SDK) such as Catchoom (2016) or Kudan (2016) , and AR content management systems such as Catchoom (2016); Layar (2016), including open source SDKs (ARToolKit, 2016). Each of the above solutions has pros and cons. Between other problems, some are quite expensive, others consume too much memory (it is important to stress that the present application will have many markers, at least one for each museum piece), and others take too much time to load on the mobile device.

This chapter presents an initial version of the MIRAR's object recognition module, which allows smartphone/tablet users to interact with the museum's innumerous objects (integrating MIRAR goals (a) to (c)). By using photographs (images) from the museum's objects, the framework allows the development of AR applications that use the mobile device's camera to recognise and track on-

the-fly, in the client-side, the museum's objects, minimising servers and communications requirements. Also presented, is the initial discussion of the MIRAR's module to detect room shapes and project (on the mobile screen) contents adapted to those shapes (MIRAR goal (d)). Not presented nor discussed is the detection of persons in the environment (MIRAR's goal (e)). These last two features are important to integrate museum's objects in the epoch and environment they were created. It is important to stress, that the MIRAR framework operates in any indoor or outdoor AR applications.

The main contribution of this chapter is the MIRAR framework architecture, which can deal with innumerous objects and users without the need of powerful servers, as the processing demands are client side (mobile device) distributed.

The document is structured as follows. The MIRAR framework and architecture is introduced in Sec. 4.2. Section 4.3 presents the main module of MIRAR, namely the image-based AR module and the communications between mobile and server, and respective tests in a museum environment. The Environments Detection module is discussed in Sec. 4.4. The chapter concludes with a final discussion and future work, Sec. 4.5.

4.2 MIRAR framework

Before detailing the MIRAR framework it is important to give a brief overview of the M5SAR system, shown on Fig. 4.1 top. On the figure's left side, the basic communications flow between the server and mobile device is schematized (a detailed description is out of the focus of this chapter) and, on the right side, the simplified diagram of the mobile App and the devices "connected" (via bluetooth) with the mobile device is shown. The displayed Beacons (Estimote, 2017) are employed in the user's localisation (see Sec. 4.3.1) and the Portable Device for Touch, Taste and Smell Sensations (PDTTSS) (Sardo et al., 2017) used to en-

hance the five senses.

As already mentioned, the MIRAR framework intends to do as much as possible processing in the mobile device, minimising server costs, communications between server and mobile, and the mobile's battery consumption. The MIRAR, in the context of M5SAR, has four main features: (a) detect and recognize museum objects triggering a card in the (M5SAR) App (Rodrigues et al., 2017b); (b) detect, recognize and track objects as the user moves along the museum, allowing him to touch different areas of the objects displayed in the mobile screen and show information about that region of the object; (c) detect and model the museum walls, and project in the detected walls information (e.g., images, movies, text) related with the recognized object's epoch; (d) detect persons that are moving in the museum, and for instance dress them with clothes from the object's epoch.

The M5SAR App architecture is divided into three main modules: Adaptive User Interfaces (AUI), see Rodrigues et al. (2017b); Location module, here addressed only in the MIRAR's scope; and, MIRAR's module (see Fig. 4.1 bottom). As already mentioned, the MIRAR's goal is to be a fast (process the requests in real time), reliable and efficient mobile (image-based) AR framework. This said, MIRAR was divided into four main modules: (i) object detection, recognition, and tracking (labelled as "Object Detection" to simplify the figure's design), features (a) and (b); (ii) environment detection, feature (c); (iii) person detection (not discussed in this chapter), feature (d); and (iv) communications with the server and other devices.

The object detection module (i) is the only module that needs to communicate with the server, i.e., the MIRAR module sends to the server the user position, based on the previous object detections and the localisation given by the Beacon's signals. From the server, the MIRAR module receives a group of object markers (image descriptors; see next section), here called bundles, that contains

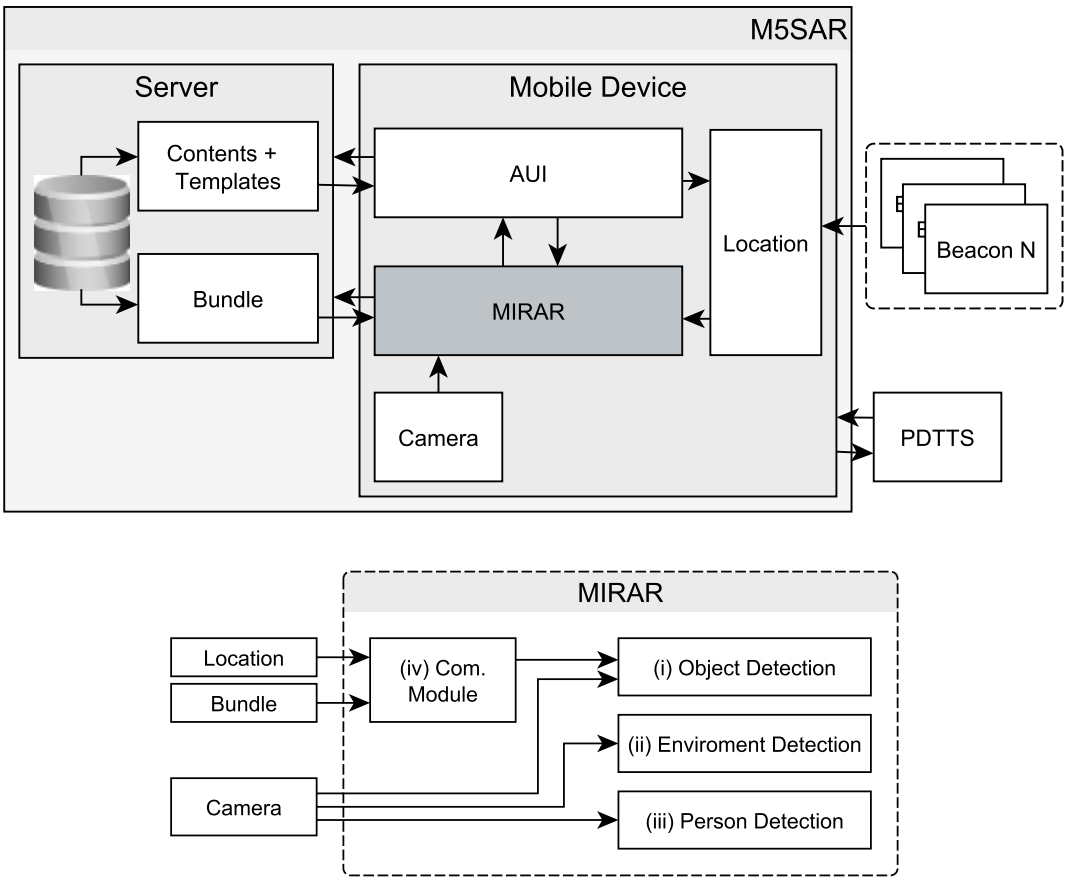


Figure 4.1: Top: overall simplified system architecture. Bottom: MIRAR block diagram.

all the objects available in the located room or museum section. In a way to minimise communications, the App stores in memory (limited to each device memory size) the bundles from the previous room(s), museum section(s), and as soon as it detects a new beacon signal it downloads a new bundle. Older bundles are discarded in a FIFO (first in, first out) way.

It is also important to stress that, since the sensor used to acquire the images from the environment is the mobile's camera (of course!), in order to save battery, the camera is only activated when the AR option is selected in the UI. When the activation occurs, the user can see the environment in the mobile screen and perpetrate the previously mentioned actions. As an additional effort to save battery, the device will enter a low-power state if the user turns the phone upside

down, by dimming the phone's screen and interrupting the processing.

As final remarks, the App was implemented using Unity (2017), the computer vision algorithms was deployed using the OpenCV (2017) library (Asset) for Unity, and tests and results consider that the mobile device is located inside a museological space. Next section will present the object detection and tracking module.

4.3 Object detection, recognition and tracking module

The object detection, recognition and tracking module algorithm was divided into 2 components: (a) detection and recognition, and (b) tracking. While the recognition is intended to work on every museum object, the tracking will only work in masterpieces[1]. The masterpieces tracking allows to place contents in specific parts of the UI and user clicks on chosen areas, in order to give more information about that particular region of the detected object.

Before describing this module in further details, it is important to distinguish from templates and markers. Here, templates are images (photographs) of the objects stored in the server's database (DB), see Fig. 4.2. On the other hand, a marker is the set of features (keypoints) with their respective (binary) descriptors for a template. The authors' implementation uses the ORB descriptor for keypoint detection and descriptors implementation (Rublee et al., 2011), as it is a good alternative to other known descriptors, e.g., BRIEF or SURF (Figat et al., 2014). ORB is a reliable fast binary descriptor that works very well on mobile devices. Examples of keypoints can be seen in Fig. 4.3 top, corresponding to partial views of the bottom two templates on Fig. 4.2.

[1] Masterpieces are objects that have an enlarged (historical and cultural) value in the museum's collection.

Figure 4.2: Example of existing objects in Faro Museum.

A generic image recognition and tracking algorithm for AR has the following main steps: (1) from a template image(s) extract the markers (keypoints and descriptors); (2) from a query image (i.e., for each mobile device camera's frame) extract keypoints and compute descriptors; (3) match the descriptors of both the template and query; and, when needed, (4) calculate the projection matrix to allow perspective wrapping of images, videos, and other contents.

An initial recognition algorithm was presented in Rodrigues et al. (2017b). In that initial version, the markers were divided into patches which are sorted descendently by the number of keypoints present in each one, and the top 5 patches are selected. The patch with more keypoints is chosen as the *most relevant marker patch* for each template. This procedure is not done on the frames acquired by the mobile device camera, since keypoints are simply extracted from those query images and their respective descriptors computed (again, using the ORB descriptor). The matching phase compares the query image's descriptors with a template's descriptors. A valid match occurs when there is at least 1 keypoint matched. This minimum is very low because 3 methods are used to remove false-positives (outliers) from matched points (Baggio, 2012). These are rigorous methods, only returning matches that are correctly identified. For more

details please refer to Rodrigues et al. (2017b).

The performed tests showed excellent results, with the correct detection of 99% of the experimented objects of Faro Museum (see details of the test site and analysed objects in Sec. 4.3.2). Nevertheless, only the smallest patch was generically used to compare with the query image, and more importantly due to the implemented restriction (false-positives filtering) it was not possible to correctly detect the axis, i.e., to detect with precision the place where the user pressed in the object when a masterpiece is presented.

Thus, the implemented method was changed to a more "standard" AR algorithm (Baggio, 2012), which comprises the following steps. In Step (1), instead of using patches, the whole image is utilized to extract keypoints and compute descriptors. The exception are the borders (e.g., painting frame) which where removed, once usually there is no relevant information in that areas. Nevertheless, the templates are processed in different scales (image sizes): starting at the pre-defined camera frame size (640×480), the templates are scaled up and down (by a $1/3$), resulting in a total of 3 scales per template. To further increase the framework performance, these markers continue to be created on a server and sent to the client (mobile device) on demand, to be deserialized (see details in Sec. 4.3.1).

Step (2) of the algorithm remains equal to the previous implementation, i.e., from the frame acquired by the camera, the keypoints are simply extracted and their respective descriptors computed (using the ORB descriptor).

Regarding Step (3), the query image descriptors are (3.1) brute-force matched, using K-Nearest Neighbours (KNN), with $K = 2$, against the descriptors of the available markers. Next, (3.2) the markers descriptors are matched to the query's descriptors. Following, (3.3) a ratio test is performed, where if the two closest neighbours of a match have close matching distances (65% ratio) then the match is discarded (Baggio, 2012), because this would be an ambiguous match.

This ratio evaluation is the test where most matches are removed. For this reason, this test is performed first to improve performance later on. Then, (3.4) perform a symmetry match where only the matches resulting from the KNN in (3.1) that are present in (3.2) are accepted. After this, a (3.5) homography refinement is applied. This refinement uses the RANSAC method to verify if the matched keypoints in the query image maintain the same configuration between them (same relative position) as they had in the template image. If any of the keypoints stay out of this relation then they are considered outliers and removed from the match set. (3.6) If after all these refinements there are at least 8 matches, then it is considered as a valid classification. In the (3.6.i) classification stage the query image is compared (brute-force) to all marker scales for each of the available templates. This, in turn, returns a classification based on the count of (filtered) matches, when there are at least 8 descriptors matches. The marker that retrieved the most number of matches is considered the *template to be tracked*. Afterwards, if the (3.6.ii) tracking stage is necessary, i.e., if a masterpiece is present, the matching only occurs with the markers of the 3 scales of the *template to be tracked* previously selected in classification phase. If the object (*template to be tracked*) is not visible in the scene for 1 second then it is considered lost and the recognition process initiates again.

Last but not least, Step (4) of the generic algorithm is done using perspective wrapping (pose estimation) in order to place content on the same plane as the detected image (marker). To accomplish this, points in the camera's 2D space must be projected into the world's 3D space. This is achieved by computing

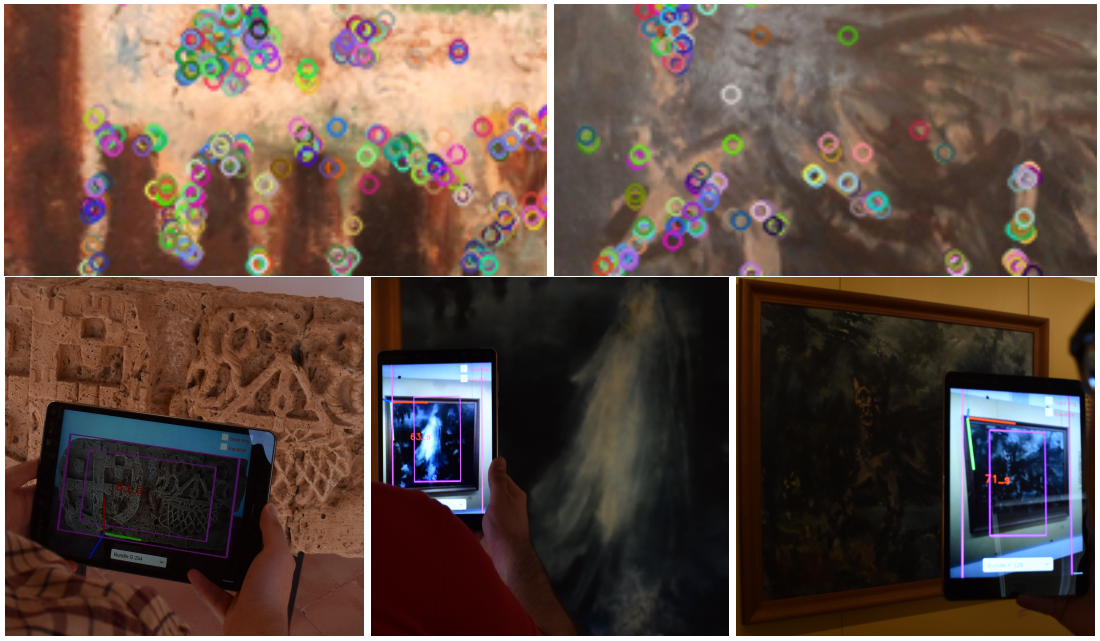


Figure 4.3: Top: two examples of descriptor detection. Bottom: examples of a detected and tracked marker with the axis.

P_{cam}, with $P_{cam} = A \times [R|T] \times P$, or in matrix (expanded) form:

$$\begin{bmatrix} x_{cam} \\ y_{cam} \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} R_{11} & R_{12} & R_{13} & T_1 \\ R_{21} & R_{22} & R_{23} & T_2 \\ R_{31} & R_{32} & R_{33} & T_3 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}, \tag{4.1}$$

where P_{cam} represents a point in the camera's 2D space, A denotes the internal (intrinsic) calibration matrix, $[R|T]$ denotes the external (extrinsic) calibration matrix and P represents a point in the world's 3D space.

The intrinsic calibration matrix A, also called the *camera calibration matrix*, is calculated from the camera internal parameters. This allows for a better projection and 3D scene reconstruction. It is composed of the camera's horizontal and vertical focal length represented (in pixels) as f_x and f_y, respectively. The c_x and c_y values are the camera's centre point (principal point) that are used to move the origin to the upper-left corner of the image. These parameters are unique to

each camera-lens configuration and once calculated remain unchanged for that combination.

Regarding the extrinsic matrix ($[R|T]$) it is derived from the scene, i.e., for each frame this matrix is different and must be computed. To calculate this at least 8 matched points (2D coordinates and their respective 3D position) are required (Laganière, 2014). Once this matrix is calculated, it can then be applied to other points to be placed (projected) in the scene. Getting this set of 8 points with their 2D coordinates and 3D space equivalent is an easy task within this module.

Since this module manages 2D images/views (e.g., paintings) and in templates the moulding is not considered, it is safe to assume that all keypoints from the marker (template) are on the same plane ($Z = 0$). After the completion of the matching step (Step 3), there is a set of matches where each one has a keypoint of the template and its correspondent keypoint from the query image. This given, one can get a set of 2D points from the query's keypoints and their complementary 3D position using the templates keypoints (assuming $Z = 0$, flat plane). With this, the extrinsic matrix is computed and later used to calculate 2D points to be projected in the UI. Even though the 3D coordinates do not represent (directly) any real world measurements (e.g. meters) that is not an issue because the 3D points are measured from the same origin using the same unit (in this case, pixels).

An illustrative example for the complete algorithm for a real museum object, is as follows: the template is shown in Fig. 4.2 bottom right, for the keypoints and descriptor for a single scale shown in Fig. 4.3 top right. For those markers, when limiting the ORB keypoint detection to a maximum of 512 keypoints, the (3.1) KNN and brute-force matching return 510 matches, the ratio test (3.3) returned only 82 keypoints, after the symmetry test (3.4) there were only 71 and, finally, the homography refinement (3.5) obtained a final count of 52 keypoints.

Steps (3.6) and (4) are shown if Fig. 4.3 bottom right, where the classification number is shown in red and where the axis (red, green and blue lines) have been projected using the technique explained in Step (4). The intrinsic matrix was previously calculated. The extrinsic matrix computed as described above and applied to a "3D object" (axis) so that it could be correctly placed in the UI.

More examples of steps (3.6) and (4) results are shown in the same figure bottom row. In the next section we address the Communication module.

4.3.1 Communication module

As mentioned in the previous section, one fundamental aspect of the App is to detect the objects on-the-fly. Therefore, it is mandatory that the App has available in memory the necessary markers for the comparisons. The Communication module is responsible for the information transmission between server, beacons, and client (mobile device). Its main goal is to, using a beacon indoor location system and the last detected museum object (see also Sec. 4.2), send to the server which room or museum section the user is in, making the necessary calls so that the mobile device receives the required data (e.g., markers, contents, etc.).

To establish the communication between server and client a communication protocol was settled. Since the protocol should enable an expedite recognition process, the templates are processed into markers on the server side, extracting all the required information and saving it to subsequently be sent. That said, the markers needed to be kept in some sort of file/database that could be easily sent to the mobile device. The adopted solution was to serialise the markers and send them like that to the client, in order to speed up the recognition process. This relieves the mobile device from creating the templates on-the-fly only needing to deserialise the previously processed files.

As a first approach, and because it was already used on the developed Adap-

```
table Keypoint {
    x:float;
    y:float;
    size:float;
    angle:float;
    response:float;
    octave:int;
    classId:int;
}
table Descriptor {
    columns:int;
    rows:int;
    cvType:int;
    data:[ubyte];
}
```

```
table Marker {
    name:string;
    keypoints:[Keypoint];
    descriptor:Descriptor;
}
table Bundle {
    markers:[Marker];
}
```

Figure 4.4: FlatBuffer schema used in MIRAR.

tive UI (Rodrigues et al., 2017b), the authors used a JSON (2017) "encoding" to serialise (on the server) and deserialize (on the client) the required data to deliver the image recognition. However, initial testing showed a poor performance on the downloading and data parsing, with unpractical times and file sizes, voiding the purpose of a fast framework. Thus, serialisation through JSON was abandoned and the FlatBuffers (2017) library was selected for the (de)serialisation purposes.

FlatBuffers is a multi-platform binary serialisation library developed by Google, which is compatible with Unity (2017). When using FlatBuffers, a schema (in human-readable form) is defined for the information to be sent, being compiled to structures that are then used in the serialisation (binarization) and deserialization process. The FlatBuffers serialisation is order dependent because it is made up of binary data and memory offsets. This said, saved data must be created in a bottom-up methodology. To deliver fast serialisation times and small file sizes, the FlatBuffers library lacks on data types available, implying that more (simpler) structs must be created in order save the required data for the image recognition. The FlatBuffer schema used in MIRAR is shown in Fig. 4.4.

In summary, each bundle corresponds to a FlatBuffer binary file. The files are automatically downloaded while the user navigates within the museum. The correct bundle is acquired by the location of the user inside the museum. This location is provided by a set of bluetooth beacons that will map the user's location inside the museum allowing the App to download (in background) the correct bundle for the section the user is in. Due to small file sizes, FlatBuffers proved to be an efficient solution when compared to JSON in terms of downloading and deserializing contents. The download and deserialization times are presented in the Sec. 4.3.2.

4.3.2 Tests and results

A set tests were performed at Faro Municipal Museum[1] using 3 different mobile devices (Samsung S7 Edge, Asus Zenpad 10 3S and Windows Surface 3). From the museum's permanent exhibitions, a selection of objects was made and divided into 8 bundles, named A through G; see Fig. 4.5. The selection contains a variety of different objects, mainly painting (bundle A to F) and stone engraved coats of arms (bundle G). The museum floor plan can be seen in the same figure with the corresponding bundles' identification. Objects in bundle A to D have dim artificial light, with exemplifying pictures in the images from the top row 1st column, and bottom row 1st and 2nd column of Fig. 4.2. In bundle C the pictures are very large compared with the room size, becoming almost impossible to frame a whole painting in the mobile device screen. Bundle D contains octagonal portrait paintings with very little contrast (see Fig. 4.2 top row, 2nd column). Bundle F pictures are all well lit, both from natural and artificial lighting. Examples of pictures from Bundle F can be found in Fig. 4.2 bottom row, 3rd and 4th column. Bundle G is an outdoor space in the museum cloister, see top row, 3rd and 4th columns of Fig. 4.2.

[1] http://www.cm-faro.pt/pt/menu/215/museu-municipal-de-faro.aspx

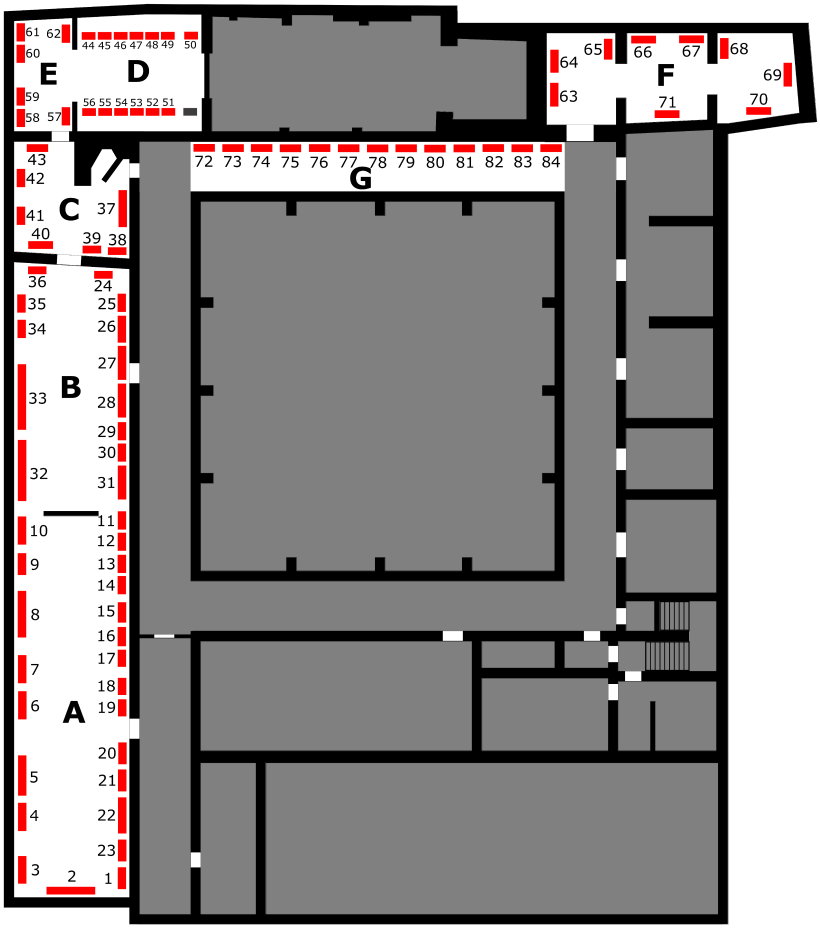


Figure 4.5: Faro museum 1st floor plan, with the bundle designation and object IDs.

A preliminary set of tests were conducted on the Samsung S7 Edge using 128, 256, 512, and 1024 keypoints as a maximum number of keypoints (for the ORB keypoint detector). As the limitation of 128 keypoints showed good results to detect "all" objects and project the axis with precision, tests with the remaining devices were done using that number of keypoints for the markers. Table 4.1 shows, grouped by bundle, the results for the number of markers per bundle, and matches correctness (true positives and false positives). For this cases, the last column (ALL) shows the sums for all bundles. The table's 5th and 6th rows show the average time in milliseconds for classification (Class.) and tracking (Track.) processes, and the corresponding last column represents the averaged

Table 4.1: Samsung S7 Edge results for all bundles.

Bundle		A	B	C	D	E	F	G	ALL
Number of markers		23	13	7	13	6	9	13	84
Correct matches (true positives)		21 (91%)	12 (92%)	7 (100%)	12 (92%)	6 (100%)	9 (100%)	13 (100%)	80 (95%)
False positives		2 (9%)	0 (0%)	0 (0%)	0 (0%)	0 (0%)	0 (0%)	0 (0%)	2 (2%)
Times (ms)	Class.	694	413	172	332	186	223	392	345
	Track.	59	63	44	45	62	54	44	53
	Deser.	125	85	37	56	53	53	86	70

time from all bundles. It is important to stress that, the false- and true- negatives were not shown in the table because it were always 0, and, in addition, the presented times are considering 2 scales and are the average from at least 10 classifications or tracking per object.

For Bundle A there were 2 false-positives since this bundle contains two identical pieces (one is a replica). In bundle B, MIRAR could not identify the painting behind a glass, due to the reflections. Regarding bundle C all objects were correctly identified, albeit the large paintings. Furthermore, the 345 ms average time taken to identify the objects does not affect the UI in modern devices, keeping a stable and fluid appearance, as this process happens in the background. It can also be concluded that the classification, when increasing the number of objects time is almost linear, i.e., it grows in averaging 28 ms per marker (regarding the Samsung S7 Edge).

In the case of the Asus Zenpad 10 3S, the results were not the best, probably because the device's camera does not change its parameters to counterbalance the lighting conditions in the museum. So, this device could only detect 6% of the objects from bundles A to E. In bundles F and G, where the lighting condition are optimal, the device detected correctly 95% of the objects, with an average time of 315 ms. As one interesting note, this device is less "powerfull" than the Samsung S7 but produced faster times, even though the quality of the camera

is much worse. This can be due to several factors like the different versions of Android OS they were running (version 6 on Asus vs. version 7 on Samsung S7) or a lower number but faster processors on the Asus. Further testing is needed to evaluate the cause. Compared with the Asus Zenpad, in the case of the Windows Surface 3 (tablet version, not the Pro version) the results were much better with a correctly detection ratio of 95% for the objects from bundles A to G and an average time of 240 ms.

Last but not least, regarding the combination of all devices, the communication with the server takes on average 249 ms to download, being this time highly dependent on the wifi connection. FlatBuffers achieved an average of 16KB per marker in a binary file and it takes an average of 69 ms to deserialize (Deser.) each bundle, again an average value for all bundles in the 3 devices; e.g., see bottom row of Tab. 4.1 for the Samsung S7 results.

4.4 Environments detection module discussion

In this section it is discussed the present/future implementation of the environment detection (not yet fully implemented in the present version). Regarding the detection of the environment's shape, the framework should be able to find the walls, replacing them with other contents, such as video or images, amongst other features.

There has been an increased necessity for environment detection in the last years, from robotics to different environments, buildings, and objects detection (Zhang et al., 2014). The normal approach for image acquisition involves the use of RGB-D devices or LIDAR's sensors (Hulik et al., 2014; Ring, 1963; Xiao et al., 2013). Considering the main target of this framework, none of those methods can be used, since MIRAR should only use the camera available on the mobile device (single RGB camera). The solution found was a variation of a method

proposed by one of the authors, already used for wall/door/side-walk detection and blind navigation (Moreno et al., 2012; Serrão et al., 2015).

As mentioned by Serrão et al. (2015), a man-made environment is characterised by the existence of numerous parallel lines and orthogonal edges from which one can retrieve the geometry of any said (man-made) space. As a rule, considering that our input will come from mobile cameras, all the environments will be captured from a perspective view, where all the receding lines of said perspective will converge into a unique point in the horizon that is called the vanishing point (Duan, 2011). From there it is possible to determine the different planes (wall, floor, ceiling) of the environment, discarding irrelevant information (Moreno et al., 2012).

With the desired final objective of this module being the possibility of environments wall detection and projection of content over it in real-time, the authors' implementation had to expect some additional challenges: this application must run on mobile devices; the user will be moving continuously and changing between different rooms; the quality of the input image will vary from device to device; and some rooms may not present the desired linear perspective. A first step for the initial development was to find the perspective lines on a simpler static image, as can be seen in Fig. 4.6, i.e., from a perspective view only four situations can occur: the frontal wall (top-left); no intersections with other walls (top-right), which will present its lines converging to the vanishing point; the intersecting walls (bottom-left); or three walls intersecting (bottom-right), with a frontal wall without perspective.

Having all the above in mind, the initial algorithm is as follows:

1. Read the input frame;

2. Apply a Gaussian blur (to remove small details/noise);

3. Apply Canny (1986) edge detection;

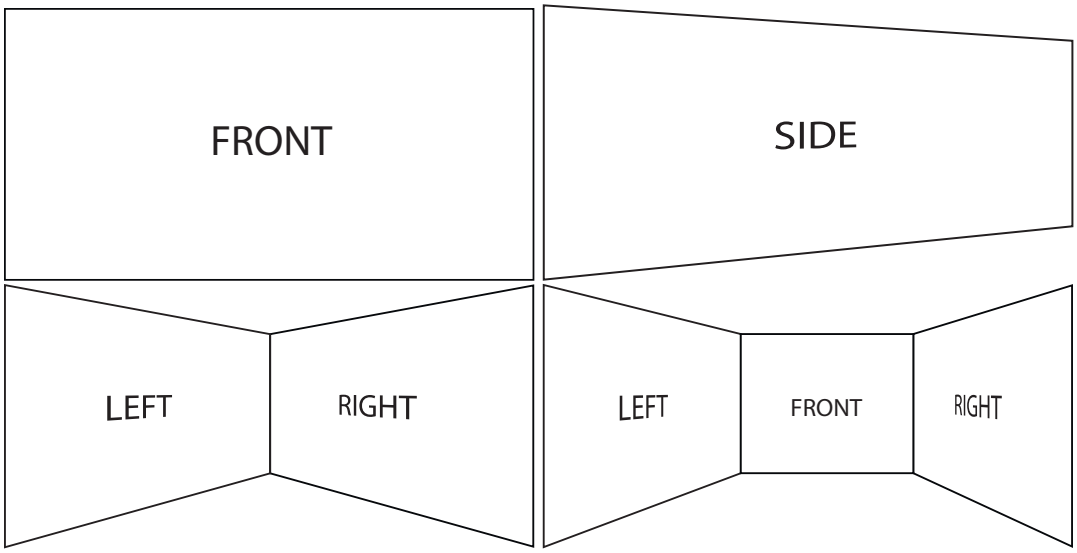


Figure 4.6: Expected shapes of the environment. On the top-left, the frontal plan, right the side plane. On the bottom-left, intersecting planes and three planes' perspective.

4. Apply the Hough (1962) Transform;

5. For each returned line, $y = m_i x + b_i$, average the ones that share a m_i and b_i with a 5% deviation;

6. Calculate all intersection points between the remaining lines and with the frame edges;

7. Evaluate each point for their proximity to the expected perspective lines;

8. Select the corner points (for each wall);

9. Apply perspective wrapping to a content with the corners points;

10. Replace the new wrapped pixels in the original frame;

In Fig. 4.7, the top 4 rows shows the illustration of these main steps with a synthetic image, and the 2 bottom rows depict two images/frames from the museum and respective edge detection by Canny.

Regarding this module, there is still further work required to achieve the final desired algorithm. Considering that the input images will come from the

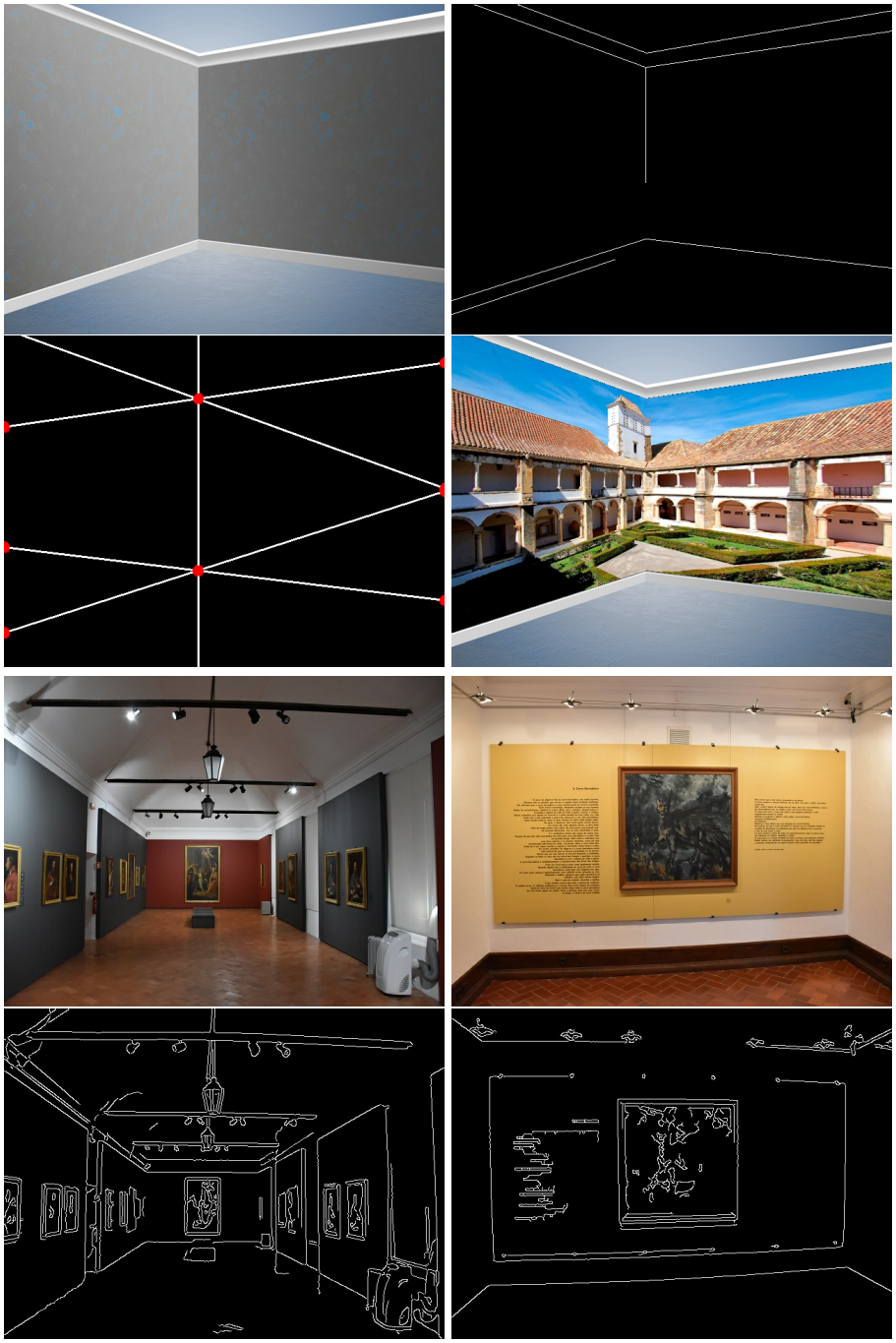


Figure 4.7: Top to bottom, left to right: Synthetic image, Canny edge detection applied over the previous image, Hough lines and points of intersection obtained from the previous image and the projection of a Faro museum picture over the detected "wall", two real pictures from the Faro museum, and the corresponding Canny edge detection.

a mobile device camera, normally achieving 30 frames per second, even if only a small percentage of frames are used, there is still additional information that comes from them. For instance, with the movement of the user, even when

trying to stay still, the perspective lines obtained from previous frames can be matched to the perspective lines newly detected. Therefore, as future work, it will be necessary a tracking system which can be implemented using the already developed tracking system.

4.5 Discussion and future work

This chapter presents the initial Mobile Image Recognition based Augmented Reality (MIRAR) framework architecture. Even in its initial state, MIRAR has already presented good results in the object detection, recognition, and tracking module. Nevertheless, tests and adaptations have to be done focusing mainly in the 3D museums objects. This kind of work/technology is fundamental for the current and future economic, social and environmental impacts, once it addresses the needs of visitors, the industry, the environment and host communities

For future work, as mentioned, recognition of 3D objects is an immediate focus, also the refinement of the object recognition and tracking module. This can be achieved by refining the matches with homography and try to find an optimised set of keypoints from multiple scales. One important step is to test several pre-processing image processing schemes to the frame resulting from the mobile device before applying the presented algorithms, in an attempt to solve the problem of dark/bright frames acquired by the camera of some mobile devices (such as ASUS Zenpad).

The Environments detection module has to be tested in real environments. If the present algorithm does not show satisfactory results, other solutions can be applied, such as segmentation, which could also allow for an easier plane detection and reduction of processing time. The presented geometric algorithm could be replaced with a corner detection module, but a performance analy-

sis would be necessary before further implementation. When the final module is accomplished, the user must be able to simply navigate with the projected content being streaming, from an outside source, without ever covering the museum pieces, further enhancing its surroundings. The person detection module is being now implemented.

In conclusion, the MIRAR shows, even in this initial stage, promising results and it is expected to be an excellent tool to give a more impactful relation between the museum user and the museum's objects.

5

Conclusions

This thesis presents the steps for developing an intelligent augmented reality mobile application to enhance the user's experience in museums. Three different modules were introduced: a route computation system, an adaptive user interface, and a marker-based recognition and augmented reality framework. This chapter concludes the thesis by presenting the integration of the above modules in the App as well as the final conclusions.

5.1 System Integration

The three modules mentioned above (Chapters 2-4) were presented independently of each other, but (of course!) they can be/are integrated in a single ap-

plication. Due to the architecture and methodology of development, several different application and UIs can be developed by using those modules.

Here, one App possible integration is presented based on the initial design (mockup) of the M5SAR project. It is important to stress that, at the time this thesis was written, a new design for the M5SAR App was already presented and the respective App is under development.

Currently, the M5SAR alpha version of the App has an interface as shown in Fig. 5.1. When starting the App the user is presented with a selection of available museums (see Fig. 5.1 top row 1st column). After choosing a museum (in this example, Faro Municipal Museum) the UI changes to exhibit the generic information about it (see Fig. 5.1 top row 2nd column). At this point, it is important to state that the UI is generated at runtime by the Adaptive User Interface module (see Chap. 3), providing a seamless navigation to the user interface, that can easily be changed according to the user's needs. Next, the user can see the navigation map (Fig. 5.1 top row 3rd column) where a path computed by the routing system is presented (see Chap. 2). The UI shows a 3D view of the museum's floor where the closest room to visit appears in green, the farthest in orange, and yellow is used to indicate the rooms in between. The floor plan of the closest room is also displayed with the objects (PoI) to visit and the suggested route between them. The user can now decide whether he will view a generic information of a piece by touching it on the map (see Fig. 5.1 bottom row 1st column), or if he will enter the Augmented Reality component (see Fig. 5.1 bottom row 2nd column) where it identifies and obtains more (detailed) information about that piece (see Chap. 4). When the object is recognized, it triggers the AUI to show the information about it (see Fig. 5.1 bottom row 3rd column). At any time the user can switch between the AR and the navigation map.

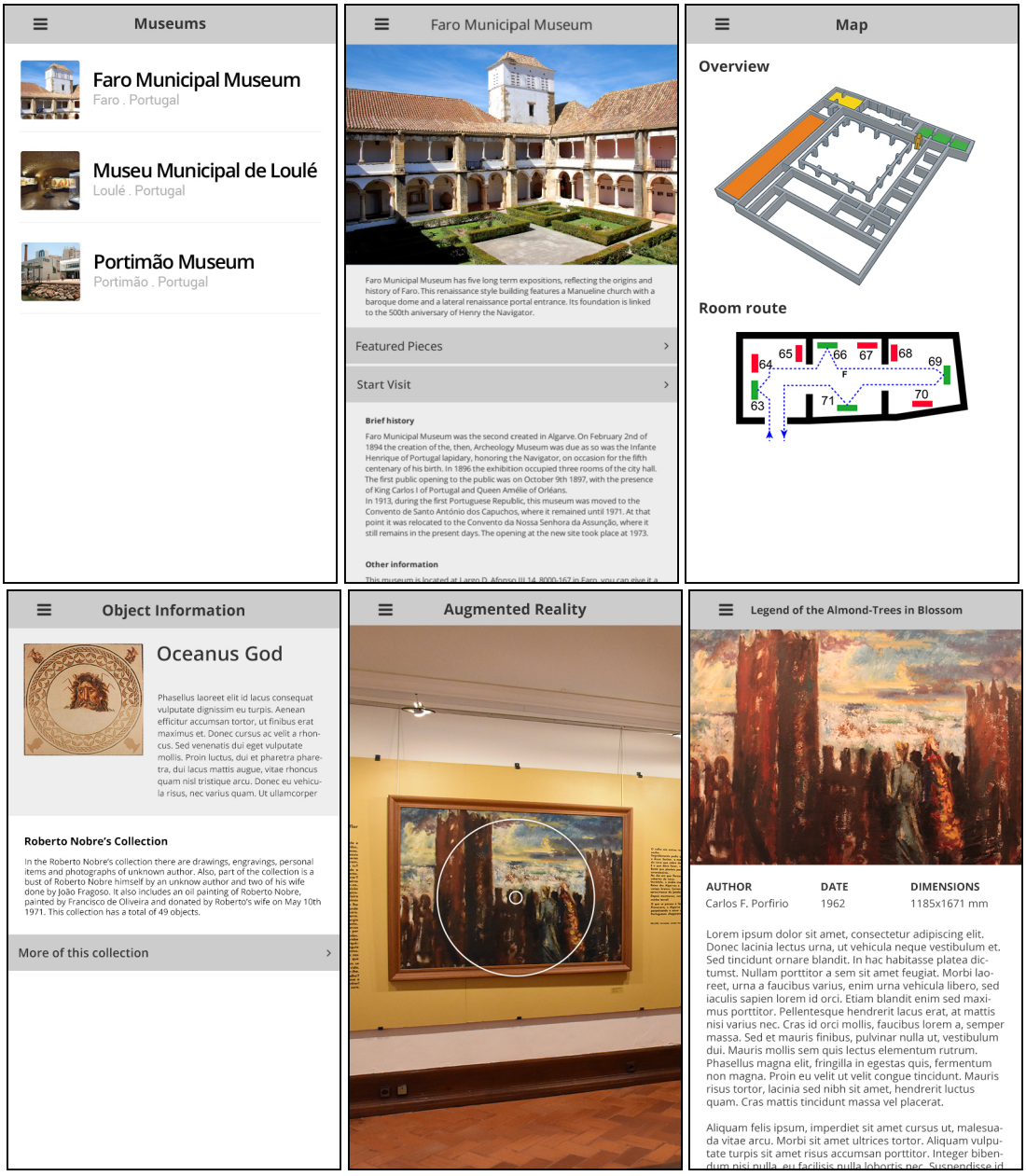


Figure 5.1: Modules integrated in the APP. Top to bottom, left to right: list of available museums, specific museum information, museum map with a route calculated, example of a card piece, image recognition (AR), and information about the detected piece.

5.2 Final Conclusions and Future Work

As mentioned throughout the text, this thesis presented the foundations for a framework to generate "intelligent" AR Apps, in the present case with the functionalities requested for the M5SAR project. Some of the partial conclusions

were already presented in each chapter.

Regarding the purposed objectives for this thesis:

(a) The development of an intelligent navigation module was accomplished by formulating a multi-criteria optimization problem. Therefore, two methods, built upon the ACO algorithm, were tested. To compute and deliver the best route, a network of PoIs (where the nodes are the PoI and the edges are the the distances between them) was generated. The algorithm developed delivers a walk that accommodates the user's choices and adapts to its limitations. The details of this module and tests are shown on Chapter 2. The main contribution in this chapter is the calculation of dynamic routes that can adjust themselves based on the options and restrictions of the user;

(b) The development of the user interface module based on "Adaptive Cards" was completed with the creation of a modular card system. This system allows the generation of smaller and simpler cards that when assembled together enables the creation of more complex interfaces. These cards can be adapted to the users by changing, for example, the text font size or by presenting more images. The UI is generated at run-time as the App is being used and even though it slightly affects the initial boot time, future changes in the UI layout (even while running) are easily made. The implementation details are shown on Chapter 3. The main contribution of this chapter is the developed framework for adapting the UI on-the-fly with a card-based modular architecture;

(c) The development of a marker-based image recognition and augmented reality framework was accomplished. This module is capable of recognizing the various museum objects by extracting features of the works-of-art and comparing them on real-time. Since this module's intention is to run on the client's mobile device, it would be impossible to execute the recognition in real-time if all the processing was done on the mobile. As such, the museum objects are captured in advanced, pre-processed, and saved in a server. This server does

not need a high performance since the processing of the images only happens once. The (minimum) required information is saved on the server and later sent to the client as he navigates through the museum. The tests performed showed promising results for the image recognition speed and robustness. The full detailed explanation, as well as test results, are shown on Chapter 4. This chapter improves the current state-of-the-art by dividing the image-recognition process between the server and client thus decreasing the costs on the server side as the processing demands are distributed through each client;

Last but not least, also already mentioned, (d) the development of the mobile application to be used in a museum space was fulfilled by creating an App that combines the three modules (presented in Chapters 2-4). This integration is shown in Sec. 5.1.

In term of future work, being the main focus to continue the incorporation and improvement of the three (separate) modules into a unique application, several aspects can be improved in each module, as will be mentioned below. Nevertheless, a fourth module must be developed to collect the necessary information about each users' preferences, limitations, and skills. This could be done by crawling through the user's social profiles and collecting the required data, at the first time the App is used. This module would give "more intelligence" to the systems, communicating with both the AUI and the routing system. During the use of the App, this module would acquire more information about the user's navigation pattern and how he uses the UI, updating the routes and the AUI dynamically.

Regarding the routing system, it should use the information of the aforementioned module to adapt to the user's way of traveling through the museum. More exhaustive tests are required using a real-life museum with a larger network and consequently more PoI and more categories. As for the Adaptive User Interface, the main focus of the future work goes by using the acquired user in-

formation and changing the interface to ease its use and accommodate for the user necessities.

For the marker-based recognition framework improvements must be made in several subjects. Even though it showed pleasing results when working with 2D museum objects (e.g. paintings), changes must be made to allow the detection of 3D objects. Also, some refinements to improve the accuracy and precision when tracking objects are essential for the augmented reality component. In order to further increase the performance of the framework, the extracted keypoints of the various scales must be somehow optimized. Real-life tests should be conducted on the environment detection module to see if the proposed algorithm has to be adapted.

All in all, the objectives of the thesis were achieved, including several publications (see Sec. 5.3), but for the final M5SAR project application more work needs to be done in the following months.

5.3 Publications

During the time of the master's degree, six peer review manuscripts were published and/or accepted for publication: five on international conferences and a book chapter. In preparation until the end of the year, is a short paper for the RECPAD 2017, and a Journal article for International Journal of Virtual and Augmented Reality (IJVAR).

The following list enumerates the published works, where the first three documents are the ones presented in this thesis, and the last three present complementary information or work (not placed in the thesis to avoid the exaggerated increase of the thesis size).

- **Pereira, J.**, Nogin, S., Cardoso, P.J.S., Rodrigues, J.M.F. (2017) ***A Cultural Heritage and Points of Interest Multi-Criteria Router Supported on Visi-***

tors Preferences, In Proc. of the 7th International Conference on Software Development and Technologies for Enhancing Accessibility and Fighting Info-exclusion (DSAI 2016). ACM, New York, NY, USA, 392-399. DOI: https://doi.org/10.1145/3019943.3019999

- Rodrigues, J.M.F., **Pereira, J.A.R.**, Sardo, J.D.P., Freitas, M.A.G., Cardoso, P.J.S., Gomes, M., Bica, P. (2017) ***Adaptive Card Design UI Implementation for an Augmented Reality Museum Application***, In M. Antona and C. Stephanidis (Eds.): Universal Access in Human-Computer Interaction 2017, Part I, LNCS 10277, pp. 1–11, 2017. DOI: 10.1007/978-3-319-58706-6 35

- **Pereira, J.A.R.**, Sardo, J.D.P., Freitas, M.A.G., Veiga R., Cardoso, P.J.S., Rodrigues, J.M.F. (2017) ***MIRAR: Mobile Image Recognition based Augmented Reality Framework***, accepted for Int. Congress on Engineering and Sustainability in the XXI Century, 11 - 13 October, Faro, Portugal

- Rodrigues, J.M.F, **Pereira, J.A.R.**, Cardoso, P.J.S., Lessa, J., Sardo, J.D.P., Freitas, M., Semião, J., Monteiro, J., Ramos, C.M.Q., Lam, R., Esteves, E., Figueiredo. M., Gonçalves, A., Gomes, M., Bica, P. (2017) ***An Initial Framework to Develop a Mobile 5 Sense Museum System***, accepted Chapter 5 in Technological Developments for Cultural Heritage and eTourism Applications, IGI Global. ISBN: 978-1-5225-2927-9

- Cardoso, P.J.S., Rodrigues, J.M.F., **Pereira, J.A.R.**, Sardo, J.D.P. (2017) ***An Object Visit Recommender Supported in Multiple Visitors and Museums***, In M. Antona and C. Stephanidis (Eds.): Universal Access in Human-Computer Interaction 2017, Part I, LNCS 10277, pp. 1–12, 2017. DOI: 10.1007/978-3-319-58706-6 24

- Sardo, J. D. P. , Semião, J., Monteiro, J. M., Esteves, E., **Pereira, J.**, Freitas,

M., Rodrigues, J. M. F. (2017) ***Portable Device for Touch, Taste and Smell Sensations in Augmented Reality Experiences***, accepted for Int. Congress on Engineering and Sustainability in the XXI Century, 11 - 13 October, Faro, Portugal

References

Adobe (2016). XD Essentials: Card-based user interfaces. https://goo.gl/gg8qUM. Retrieved: Nov. 16, 2016.

Akiki, P. A., Bandara, A. K., and Yu, Y. (2014). Adaptive model-driven user interface development systems. *ACM Computing Surveys (CSUR)*, 47(1):9.

Alvarez-Cortes, V., Zárate, V. H., Uresti, J. A. R., and Zayas, B. E. (2009). Current challenges and applications for adaptive user interfaces. In *Human-Computer Interaction*. InTech.

ARToolKit (2016). ARToolKit, the world's most widely used tracking library for augmented reality. http://artoolkit.org/. Retrieved: Nov. 16, 2016.

Azuma, R., Baillot, Y., Behringer, R., Feiner, S., Julier, S., and MacIntyre, B. (2001). Recent advances in augmented reality. *IEEE computer graphics and applications*, 21(6):34–47.

Babich, N. (2016). Designing card-based user interfaces, smashing magazine. https://goo.gl/AM46gT. Retrieved: Nov. 18, 2016.

Baca, M. and Gill, M. (2015). Encoding multilingual knowledge systems in the digital age: the getty vocabularies. *Knowledge Organization*, 42(4):232 – 243.

Baggio, D. L. (2012). *Mastering OpenCV with practical computer vision projects*. Packt Publishing Ltd.

Benouaret, I. and Lenne, D. (2015). Combining semantic and collaborative recommendations to generate personalized museum tours. In *East European Conference on Advances in Databases and Information Systems*, pages 477–487. Springer.

Canny, J. (1986). A computational approach to edge detection. *IEEE Transactions on pattern analysis and machine intelligence*, (6):679–698.

Cardoso, P., Jesus, M., and Márquez, A. (2011). ϵ – DANTE : an ant colony oriented depth search procedure. *Soft Computing*, 15(1):149–182.

Catchoom (2016). Catchoom. http://catchoom.com/. Retrieved: Nov. 16, 2016.

Cheng, K.-H. and Tsai, C.-C. (2013). Affordances of augmented reality in science learning: Suggestions for future research. *Journal of Science Education and Technology*, 22(4):449–462.

CHESS (2017). CHESS – cultural heritage experiences through socio-personal interactions and storytelling. `http://www.chessexperience.eu/`. Retrieved: June 29th, 2016.

Conati, C., Carenini, G., Toker, D., and Lallé, S. (2015). Towards user-adaptive information visualization. In *AAAI*, pages 4100–4106.

Coughlan, T., Carletti, L., Giannachi, G., Benford, S., McAuley, D., Price, D., Locatelli, C., Sinker, R., and Stack, J. (2015). Artmaps: Interpreting the spatial footprints of artworks. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, CHI '15, pages 407–416, New York, NY, USA. ACM.

Couprie, L. D. (1978). Iconclass, a device for the iconographical analysis of art objects. *Museum International (Edition Francaise)*, 30(3-4):194–198.

Croes, G. A. (1958). A method for solving traveling-salesman problems. *Operations Research*, 6(6):791–812.

Deb, K. (2001). *Multi-objective Optimization using Evolutionary Algorithms*. John Wiley & sons.

Dorigo, M. and Stützle, T. (2004). *Ant Colony Optimization*. MIT Press.

Duan, W. (2011). *Vanishing points detection and camera calibration*. PhD thesis, University of Sheffield.

Estimote (2017). Create magical experiences in the physical world. `https://goo.gl/OHW04y`. Retrieved: April 04, 2017.

Figat, J., Kornuta, T., and Kasprzak, W. (2014). Performance evaluation of binary descriptors of local features. In *Int. Conf. on Computer Vision and Graphics*, pages 187–194. Springer.

FlatBuffers (2017). Flatbuffers documentation. `https://goo.gl/kAiptk`. Retrieved: April 04, 2017.

Gajos, K. and Weld, D. S. (2004). Supple: automatically generating user interfaces. In *Proc. Int. Conf. on Intell. User Interfaces*, pages 93–100. ACM.

Gajos, K. Z., Wobbrock, J. O., and Weld, D. S. (2008). Improving the performance of motor-impaired users with automatically-generated, ability-based interfaces. In *In Proc. SIGCHI Conf. on Human Factors in Computing Systems*, pages 1257–1266. ACM.

Garau, C. (2014). From territory to smartphone: Smart fruition of cultural heritage for dynamic tourism development. *Planning Practice & Research*, 29(3):238–255.

Garcia, I., Sebastia, L., and Onaindia, E. (2011). On the design of individual and group recommender systems for tourism. *Expert Systems with Applications*, 38(6):7683 – 7692.

García-Martínez, C., Cordón, O., and Herrera, F. (2004). An empirical analysis of multiple objective ant colony optimization algorithms for the bi-criteria TSP. *ANTS Workshop, Lecture Notes in Computer Science*, 3172:61–72.

Gavalas, D., Konstantopoulos, C., Mastakas, K., and Pantziou, G. (2014). Mobile recommender systems in tourism. *Journal of Network and Computer Applications*, 39:319 – 333.

Geonames (2017). GeoNames. `http://www.geonames.org/`. Retrieved: June 29th, 2016.

Getty (2017). Getty. `http://www.getty.edu/`. Retrieved: June 29th, 2016.

HMS (2017). Srbija 1914 / augmented reality exhibition at historical museum of Serbia, Belgrade. `https://vimeo.com/126699550`. Retrieved: April 04, 2017.

Hough, P. V. (1962). Method and means for recognizing complex patterns. Technical report.

Hulik, R., Spanel, M., Smrz, P., and Materna, Z. (2014). Continuous plane detection in point-cloud data based on 3D Hough transform. *Journal of Visual Communication and Image Representation*, 25(1):86–97.

Iconclass (2017). Iconclass. `http://www.iconclass.nl/`. Retrieved: June 29th, 2016.

InformationWeek (2017). Informationweek:10 fantastic iphone. `https://goo.gl/rYnhm29`. Retrieved: April 04, 2017.

Isemann, D. and Ahmad, K. (2014). Ontological access to images of fine art. *J. Comput. Cult. Herit.*, 7(1):3:1–3:25.

JSON (2017). JSON documentation. `http://www.json.org/`.

Jung, T., Chung, N., and Leue, M. C. (2015). The determinants of recommendations to use augmented reality technologies: The case of a korean theme park. *Tourism Management*, 49:75–86.

Kudan (2016). Kudan computer vision. `https://www.kudan.eu/`. Retrieved: Nov. 16, 2016.

Laganière, R. (2014). *OpenCV Computer Vision Application Programming Cookbook Second Edition*. Packt Publishing Ltd.

Layar (2016). Layar. `https://www.layar.com/`. Retrieved: Nov. 16, 2016.

Miettinen, K. (1999). *Nonlinear Multiobjective Optimization*. Kluwer Academic Publishers.

Mohan, B. C. and Baskaran, R. (2012). A survey: Ant colony optimization based recent research and implementation on several engineering domain. *Expert Systems with Applications*, 39(4):4618 – 4627.

Moreno, M., Shahrabadi, S., José, J., du Buf, J. H., and Rodrigues, J. M. (2012). Realtime local navigation for the blind: detection of lateral doors and sound interface. *Procedia Computer Science*, 14:74–82.

Museum of London (2017). Street museum. `http://www.museumoflondon.org.uk/Resources/app/you-are-here-app/noflash/no-flash.html`. Retrieved: Fev. 20th, 2017.

OpenCV (2017). OpenCV. `http://opencv.org/`. Retrieved: April 04, 2017.

Qualcomm (2017). Invisible museum. `https://goo.gl/aS0NKh`. Retrieved: April 04, 2017.

Reinecke, K. and Bernstein, A. (2013). Knowing what a user likes: A design science approach to interfaces that automatically adapt to culture. *Mis Quarterly*, 37(2):427–453.

Ring, J. (1963). The laser in astronomy. *New Scientist*, 18(344):672–673.

Rodrigues, J., Lessa, J., Gregório, M., Ramos, C., and Cardoso, P. (2016). An initial framework for a museum application for senior citizens. *In Proc. 7th Int. Conf. on Software Development and Technologies for Enhancing Accessibility and Fighting Info-exclusion*.

Rodrigues, J., Pereira, J., Cardoso, P., Lessa, J., Sardo, J., Freitas, M., Semião, J., Monteiro, J., Ramos, C., Lam, R., Esteves, E., M., F., Gonçalves, A., Gomes, M., and Bica, P. (2017a). An initial framework to develop a mobile 5 sense museum system. *Chapter 5 in Technological Developments for Cultural Heritage and eTourism Applications (in Press)*.

Rodrigues, J., Pereira, J., Sardo, J., Freitas, M., Cardoso, P., Gomes, M., and Bica, P. (2017b). Adaptive card design UI implementation for an augmented reality museum application. *In Proc. 11th Int. Conf. on Universal Access in Human-Computer Interaction, integrated in the 19th Int. Conf. on Human-Computer Interaction*.

RoutePerfect (2017). Route perfect. `https://www.routeperfect.com/`. Retrieved: June 29th, 2016.

Rublee, E., Rabaud, V., Konolige, K., and Bradski, G. (2011). Orb: An efficient alternative to SIFT or SURF. In *Proc. Int. Conf. on Computer Vision*, pages 2564–2571. IEEE.

Sardo, J., Semião, J., Monteiro, J., Pereira, J., Freitas, M., Rodrigues, J., and Esteves, E. (2017). Portable device for touch, taste and smell sensations in augmented reality experiences. *Accepted for InCREASE*.

Serrão, M., Shahrabadi, S., Moreno, M., José, J. T., Rodrigues, J. I., Rodrigues, J. M. F., and du Buf, J. M. H. (2015). Computer vision and gis for the navigation of blind persons in buildings. *Universal Access in the Information Society*, 14(1):67–80.

SM (2017). Science museum - atmosphere gallery. `https://vimeo.com/20789653`. Retrieved: April 04, 2017.

Steichen, B., Conati, C., and Carenini, G. (2014). Inferring visualization task properties, user performance, and user cognitive abilities from eye gaze data. *ACM Tr. on Interactive Intelligent Systems*, 4(2):11.

TWSJ (2017). The wall street journal: Best apps for visiting museums. `https://goo.gl/cPTyP9`. Retrieved: April 04, 2017.

Unity (2017). Unity 3D. `https://unity3d.com/pt`. Retrieved: Nov. 10, 2017.

UNWTO (2017). Sustainable development of tourism. `http://sdt.unwto.org/`. Retrieved: April 04, 2017.

Vainstein, N., Kuflik, T., and Lanir, J. (2016). Towards using mobile, head-worn displays in cultural heritage: User requirements and a research agenda. In *Proc. 21st Int. Conf. on Intelligent User Interfaces*, pages 327–331. ACM.

van Hage, W. R., Stash, N., Wang, Y., and Aroyo, L. (2010). *Finding Your Way through the Rijksmuseum with an Adaptive Mobile Museum Guide*, pages 46–59. Springer Berlin Heidelberg, Berlin, Heidelberg.

Verbert, K., Manouselis, N., Ochoa, X., Wolpers, M., Drachsler, H., Bosnic, I., and Duval, E. (2012). Context-aware recommender systems for learning: A survey and future challenges. *IEEE Transactions on Learning Technologies*, 5(4):318–335.

Vuforia (2016). Vuforia. `https://www.vuforia.com/`. Retrieved: Nov. 16, 2016.

Wang, D. and Xiang, Z. (2012). *The New Landscape of Travel: A Comprehensive Analysis of Smartphone Apps*, pages 308–319. Springer Vienna.

Xamarin (2016). Stack layout - Xamarin. `https://goo.gl/i7LhG9`. Retrieved: Nov. 18, 2016.

Xiao, J., Zhang, J., Adler, B., Zhang, H., and Zhang, J. (2013). Three-dimensional point cloud plane segmentation in both structured and unstructured environments. *Robotics and Autonomous Systems*, 61(12):1641–1652.

Yovcheva, Z., Buhalis, D., and Gatzidis, C. (2013). Engineering augmented tourism experiences. In *Information and Communication Technologies in Tourism 2013*, pages 24–35. Springer Berlin Heidelberg.

Zhang, Y., Song, S., Tan, P., and Xiao, J. (2014). Panocontext: A whole-room 3D context model for panoramic scene understanding. In *Proc. European Conf. on Computer Vision*, pages 668–686. Springer.

Zhao, L., Lu, Y., Zhang, L., and Chau, P. Y. (2012). Assessing the effects of service quality and justice on customer satisfaction and the continuance intention of mobile value-added services: An empirical test of a multidimensional model. *Decision Support Systems*, 52(3):645–656.