Copenhagen Mirror
Towards an Improvement of User Experience within Locative Media by the Utilisation of Simulated, On-location Lighting Conditions in a 3d Environment

Louis Flyvholm, Karlo Folmer Kristensen, Christian Graver Larsen,
Marcus Popp Mattsson, Michael Rosendahl Schmidt, and Jonas Wang

2010

Contents

Chapter 1

Introduction

Contemporary smartphone technology allows users to run applications, connect to the internet, use GPS for navigation and built-in cameras to not only take photos but also record video. In short: A plethora of information and possibilities lay at the feet of smartphone users of today. All these technologies combined lends itself well to the field of so-called location-based service (LBS). A lot of researchers have been working on the issue of describing the user experience for users of LBS's. Interaction design is naturally a very influential factor for the user experience and perception of the product. According to Raper et al. (2007) [RGKR07] the research of interaction design within the LBS area is insufficient, and deemed urgent.

The semester theme is audio-visual experiments, which will influence the overall direction within LBS's.

This paper will look at LBS's in relation to the smartphone media and how it can be used to renew the way people are informed about their current locations. This is done with an offset in the historical place of Slotsholmen, the location of the Danish parliament, in Copenhagen. This location is purely chosen out of interest as an example of a place where the theory can be tested. The found results should be independent of the used location.

The paper is structured as follows:

The introduction is continued with previous work within the field of LBS's in "1.1.1 Location-based Services" and "1.1.2 Locative Media". The Gestalt principles are found to be a relevant theoretical foundation for creating location-based services.

The introduction ends with the "1.2 Final Problem Statement" where it is chosen to look at if real world lighting, derived from the gestalt principle of similarity, is an important factor of people's experience with a location-based service.

Chapter "2 Methods and Materials" investigates how similarity"2.4 Designing for Similarity" and real world lighting"2.4.1 Estimating Lighting Conditions" can be designed and implemented in order to be testable on a mobile device which in section "2.2 Technical Platform Evaluation" is decided to be an iPhone 4. The chapter "2 Methods and Materials" continues with a section"2.5 Additional Design Considerations" of how to design and implement the remaining parts of the product including "2.5.2 Design Principles Concerning GUI", "2.5.3 Visual Appearance", and "2.5.1 Designing for Symmetry".

How the final system, combining the real world lighting and the historical visual information of Christiansborg, is built up is presented in "2.6 System Overview". Chapter "3 Final Test" consists of test considerations and the results in "3.4 Results" show that there is almost no difference between people's experience of our location-based service with or without lighting simulating the actual real world lighting conditions.

The results' validity is finally discussed and concluded upon in the chapter "5 Conclusion".

1.1 Previous Work

In the following sections location-based services are categorised leading to the area of locative media. Afterwards design principles others have followed to evaluate location-based services

leads to an approach utilising Gestalt principles, and the final problem statement rounds off the section.

1.1.1 Location-based Services

Location-based services are defined as being "information services accessible with mobile devices through the mobile network utilizing the ability to make use of the location of the mobile device"[Sec01]. The International Open Geospatial Consortium [MBN+05] defines an LBS as "A wireless-IP service that uses geographic information to serve a mobile user. Any application service that exploits the position of a mobile terminal".

Location-based services can be divided into three main categories [SNE06]:

- Location and tracking:
  - Resources tracking with dynamic distribution e.g. vehicle fleet.
  - Resources tracking without privacy controls e.g. tracking postal packages, train carts, etc.
  - Someone or something e.g. Yellow Pages function, turn-by-turn navigation, locating stolen car, phone, etc.
- Proximity-based (push or pull):
  - Notification.
  - Actuation e.g. automatic road or bridge toll payment.
- Locative media e.g. digital media applied to real world places with augmented reality, etc.

As mentioned in the introduction the theme of this particular project is audio-visual experiements. A digital visual experience falls within the locative media category and as such this category will be further investigated.

1.1.2 Locative Media

The focus in this section is on categorising locative media and how a user experience can be described in such products. When an LBS is used to convey a more natural media experience, such as storytelling or with an art installations, it distinguishes itself from a regular location-based service. As mentioned, this category of LBS's is known as locative media.

Established media, such as films, books, papers etc, that are available from mobile platforms at the location of the user are not locative media per say, as they do not create interaction with the location. The interaction with the location is what makes locative media distinct from other types of media.

According to Tuters and Varnelis [TV06] locative media can be categorised according to their mapping:

- Annotative: Virtually tagging the real world.
- Phenomenological: Tracing the action of the subject in the world.

Annotative locative media could be augmented reality, where the media is projected onto the real world. An example could be the Digital Yuanmingyuan project, where a former palace was rebuild for the user with augmented reality [LWL+06]. Phenomenological locative media could be the art project Realtime Amsterdam[PRT06], where the geographical location of the users are mapped onto the virtual world as a visualisation.

Lemos [Lem09] suggests five different categories.

- Electronic Urban Annotations: Generate invisible writing.
- Mapping and geo-localization: Mapping and tracking movements.
- Mobile social networking: Organise meetings and/or the exchange of information.
- Smart mobs: People together to perform an action in public space.
- Pervasive computational games: Games utilizing the locative possibilities.

The first two categories are very similar to Tuter and Varnelis' categories. The following two are based on the interaction between the users of the locative media and the games category is self-explanatory.

Futhermore Epstein [Eps09] suggests the term terratives which is a contraction between terres-

trial and narratives. This term covers stories that are told in tandem with real places. With focus on the historical development at Slotholmens, this category fulfils the needs for this project.

Interaction Design Paradigm of Locative Media

The natural approach to describing the interaction design and user experience could be a traditional human-computer interaction evaluation. But it is important to notice that location-based services are not limited to the interaction we know from traditional platforms. As seen in figure 1.1 the location-to-user and location-to-product interactions are important to consider. This has been widely accepted with mobile HCI research during the last decade [Joh98]. As such other methods for evaluating interaction design has been proposed.

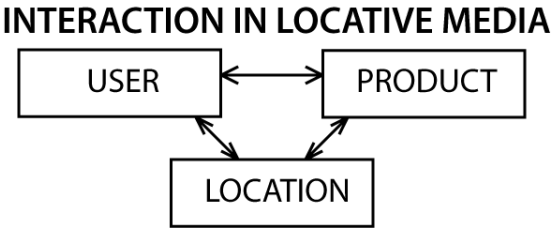

Figure 1.1: Interaction in locative media.

The idea of evaluating LBS's according to Gestalt principles have been proposed by [PK07][PK08]. The Gestalt principles are historically a way of describing the cognitive perception of sensory stimuli suggested by psychologists from the Gestalt School [Wol09]. Humans are able to perceive more than the sum of parts of stimuli, and this have influenced especially visual design for various products throughout history. Gestalt principles have been used in HCI design for traditional products, such as informational screen or graphical user interfaces [PK07], but also multi sensory products as seen in [PK07].

The multifaceted use of Gestalt Principles, which is not only limited to HCI issues, have led to a more general proposition from Oviatt et al. [OCT+03], namely the complete interaction situation. This method to evaluate interaction design is interesting as it can explain the location-to-user experience which is not the focus of traditional HCI. Kjeldskov and Paay have developed projects [PK07] [PK08] that uses the Gestalt principles as guidelines for design and evaluation of the LBS. The systems were evaluated according to the derived Gestalt principles:

- Symmetry: Align representations in the location-based service with the real world in order to obtain a coherent image from which to act.
- Similarity: Grouping specific elements in the system with corresponding elements in the surroundings.
- Proximity: Information on the mobile device screen was seen as belonging to people's current physical location.
- Closure: People relating and making sense of fragmented information and adding the missing bits themselves.
- Continuity: Information in a location-based service does not exist in isolation from people's history of interactions with it.

The categories' implications for locative media are very closely related to the geo-localizational functions. Symmetry, similarity and proximity describes the interaction between location-to-users and location-to-product. As stated the interaction between media and location defines the concept of locative media. Therefore we will align our work closely to the Gestalt framework presented by Kjeldskov & Paay, especially the categories symmetry, similarity and proximity.

The proximity principle is very self evident. According to Kjeldskov & Paay users are judging the relevance of information according to the proximity of the location. The relationship between information and location is so strong that users perceived unrelated information as relevant because of the location where it was presented.

The symmetry principle is situational, but important. Users are looking for symmetry between the LBS and the real world rotating the platform screen to align lines on the screen with lines on e.g. buildings in the real world. Creating symmetry between product and location is

important for the user because it creates comfortable interaction which leads to a better user experience.

The similarity principle may be the most important Gestalt principle for the users who created meaning between actual physical objects and the content onscreen. The users made direct use of distinct features in their environment as anchor points for matching up the system and the world, for example, landmarks, unique patterns and colours on buildings. Physical features can be used to underline the relationship between product and location.

1.2 Final Problem Statement

All together the Gestalt principles have one thing in common: The relationship with the location. They all state that coherence between the physical world and the product helps to improve the user's experience. In this paper the focus will be on that particular subject. To narrow down the focal point of the project we have decided to focus on the lighting conditions in the real world and the coherent lighting conditions in the product. This subject is believed to be important according to the similarity principle, as it is a physical parameter that can be conveyed in the product. The lighting condition is one of the few consistent comparable visual parameters shared throughout the history of Slotsholmen. The use of real world lighting conditions is believed to create a better user experience since it creates a similarity between the product and the location. Furthermore the subject of light is closely related to the semester theme. As mentioned in "1 Introduction" Slotsholmen will be the specific case where the theory will be applied.

The following criteria can now be formulated:

- The product is a locative media based on a smartphone.
- The terrative must convey the historical development of Christiansborg.
- The product must simulate real world lighting according to the similarity principle.

With those criteria in mind, a final problem statement can be formulated:

How is the user experience of a locative media affected if simulated real world on-location lighting is used?

A hypothesis can be made from the final problem statement:

The simulation of real world lighting in the product will improve the user experience.

From the final problem statement and the hypothesis it is clear that the focus will be on creating similarity by applying lighting properties from the real world to the lighting in the developed product. The symmetry and proximity principles will be used as guidelines for the design of interaction, but will not be subjects in the final test.

Chapter 2

Methods and Materials

2.1 Method

This paper will investigate how user experience is affected when one designs a system within the field of locative media with different degrees of the Gestalt principle similarity applied.

We strive to conclude upon the design of locative media in general, but are only implementing and testing on one specific case, which is a locative media conveying the historical development of Christiansborg. The scientific method used is inductive reasoning. An experimental method will be used to compare a control group to an experimental group where certain variables are altered. Within this method, statistics will be the main tool for which to make generalisations from the specific case, hence a quantitative final test on the hypothesis will be conducted as described in "3.3 Statistics of the Test".

To be able to test the hypothesis we will create two versions of the product, one that simulates the current lighting conditions at the user, and one that uses estimated lighting conditions of noon at the user's location. In this way the testing variable will be the lighting conditions and thereby the similarity aspect. The ratings of these two versions can then be compared, and figure 2.1 shows the expected test results if the hypothesis from "1.2 Final Problem Statement" is true.

If the hypothesis is accepted, the rating of the user experience will be similar at noon where the lighting conditions are equal, and the control group's rating of user experience will be lower when the difference between lighting in the real world and in the product are higher, such as at sunrise and sunset.

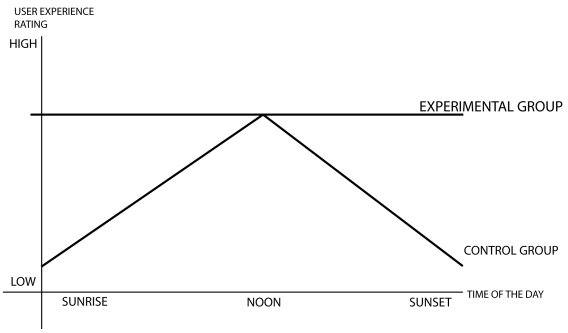

Figure 2.1: Expected results if the hypothesis is accepted.

2.1.1 The Case: Locative Media and Christiansborg

The general idea for our product is a locative media that makes it possible to experience the castles which were once at Slotsholmen in Copenhagen. To do this the area at Slotsholmen will be recreated as a 3d environment, with the old castles, and the user will then be able move and look around in this 3d environment by moving the smartphone or themselves in real life. Only one castle will be shown at a time, and the user should be able to switch between different time periods to see different castles.

As mentioned earlier a smartphone will be the platform and should provide possibilities for the chosen interaction. The interaction control scheme will be based on the Gestalt principle of symmetry, where aligning the user's interaction with the location is important. This will be done by using the real, physical movements of the user and the smartphone in the 3d environment made for the product. This kind of interaction is very similar to other location-based

services such as Layar[1] and is assumed to provide sufficient and understandable controls and interaction for the user.

As seen in figure 2.2 the smartphone can be rolled, pitched and yawed to change the orientation in the 3d environment, i.e. pitching the smartphone lets the user look up and down in the 3d environment, just like if the user were using a camera to look at the world.

Furthermore the user should be able to walk around near the current Christiansborg to move in the 3d environment. As seen in figure 2.3 the light blue area is the actual area around Christiansborg where users are expected to move around.

Figure 2.3: Accessible area for the user on Slotsholmen.

To ensure smooth interaction between physical movement and the modelled environment the system must run in real-time[2].

Furthermore all data collection for the system should be automatically done and should not be dependent on the user.

With a hypothesis claiming a general property of locative media and a specific case to test the claim on, the paper will in the following look into how a product technically can be developed. In the ideal world one would start looking at the requirements for implementing similarity in the system and then decide on the platform. But in order to narrow down the focus of the technical possibilities we start of by determining the platform for the product and development.

2.2 Technical Platform Evaluation

The goal of this section is to provide an overview of the possible target platforms along with an evaluation of them and in particular how development for them occurs. This is coupled with a look at the available software for mobile phone development.

2.2.1 Hardware

The hardware platform for this project will be a smartphone which is able to detect physical movement in the form of roll, pitch and yaw as mentioned in "2.1.1 The Case: Locative Media and Christiansborg".

This project is based on a location-based service and an essential feature of any device to support is therefore GPS or other kinds of positioning systems. Many regular mobile phones do not have this, while almost all modern smartphones have GPS built in[3]. Two of the major operating systems for smartphones are iOS, by Apple, and Android, by Google. The iOS system only runs on Apple devices, specifically the smartphone, iPhone, the portable digital assistant, iPod Touch and the tablet, iPad. Android is open source and therefore runs on a multitude of phones, most notably the smartphones by HTC[4][5]. To create the control scheme mentioned in "2.1.1 The Case: Locative Media and Christiansborg"it is necessary that the smartphone has a built-in gyroscope, since an accelerometer, which is more popular in smartphones these days, is not capable for measuring the yaw orientation. Looking at Android smartphones and the iPhone series, the only device which has a gyroscope, as of fall 2010, is the iPhone 4. This smartphone also fulfils other immediate hardware requirements such as screen size and processing power, since it is built and marketed to

[1] http://d.pr/RvXr
[2] We define real-time as at least 20 frames per second.
[3] http://d.pr/QUSd
[4] http://d.pr/vUo5
[5] http://d.pr/Kz1s

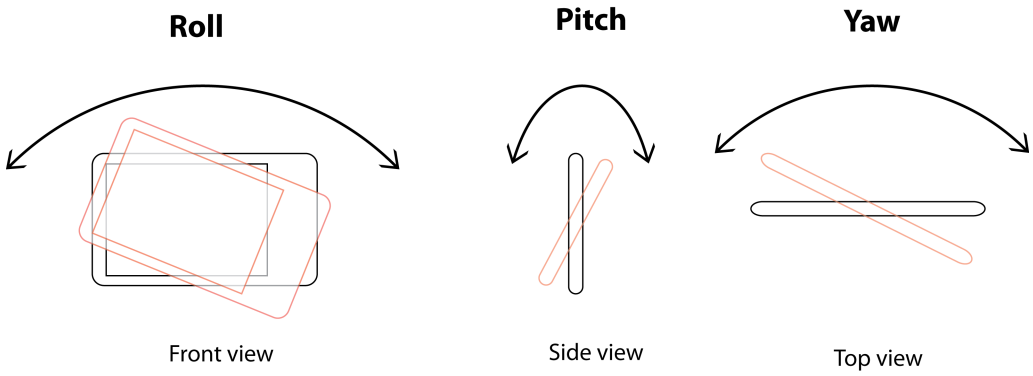

Figure 2.2: Axes of possible movement for orientation in the 3d environment.

be able run a multitude of games[6], and it is therefore chosen as the mobile device used for development.

2.2.2 Software

Apple provides a basic, free SDK[7] for iOS which includes project examples and an iPhone emulator. The language which is interpretable by the iPhone is Objective-C. With Objective-C one can interface with the Cocoa API which provides graphic functionality to applications. Use of the OpenGL ES API is also possible for 2d and 3d graphics.

One piece of middle ware that can deploy to both web, iOS and Android is Unity[8]. This tool is aimed at game development and other applications requiring interaction and visualisation. It is built as an easy to use platform even for beginners and scripting can be done in either JavaScript, C# or Boo[9]. Unity provides full support for iPhone 4, which includes access to the iPhone's GPS, compass, and gyroscope.

Unity appears to offer all the functionality needed without having to deal with time consuming low-level work. Furthermore, the iPhone 4 is chosen as the platform for this project as it comes with a built-in gyroscope, which is needed for the project.

[6] `http://www.apple.com/iphone/features/game-center.html`

[7] `http://developer.apple.com/devcenter/ios/index.action`

[8] `http://www.unity3d.com`

[9] A python dialect.

2.3 Requirements

At this point we can define a set of requirements which will form the basis for the design of the system that will be described in the following sections.

- Developed for iPhone 4
- Developed with Unity
- Real-time interactive 3d environment
  - Allows user to look around
  - Allows user to move around
- Aesthetics
  - Automatic real-life lighting and shadows
- Interface to switch between time periods

2.4 Designing for Similarity

According to "1.1.2 Locative Media" similarity is deemed to be one of the important Gestalt principles when applying these to interaction design. In "1.2 Final Problem Statement" it was chosen to look specifically at the importance of lighting conditions between the physical world and the one in the locative media, since this is an obvious aspect of the similarity principle. This section will cover the process of analysing this principle and designing for it. Specifically we first look in-depth at the principle, then we analyse methods to estimate real-life lighting, look at related areas, and then explore how lighting is created in Unity and with iOS, the chosen

development platforms. Lastly we design and implement the part of the application related to lighting.

2.4.1 Estimating Lighting Conditions

This section will look at the properties of daylight and how we can estimate real-life lighting conditions.

Methods to Estimate Real World On-location Lighting

To simulate the real life lighting conditions on the location of the user, it is necessary to be able to somehow obtain lighting information from this location. External devices, such as a light meter, might be able to obtain detailed lighting information, but since we have chosen to use a smartphone as the platform, we do not wish to impede on the user's experience with the application by adding external devices. However, most cameras have built-in light meters, and the iPhone 4's camera will therefore also be looked at. Lastly other work in the field of estimating lighting will also be analysed.

The Danish Institute of Meteorology (DMI) provides information about the Danish weather on their website, and using geolocation it is possible to fetch localised weather data using screen scraping[10]. Since parts of the weather information (cloudy, sunny, etc.) contain data about the lighting conditions, this would be a possible way to get almost real-time on-location lighting information. However, the official weather information is only updated every couple of hours, but DMI also uses weather information from privately owned weather stations[11]. One of these[12] is located within 3 kilometres from Christiansborg and its weather information is updated every 10 minutes. The weather information consists of temperature, wind, humidity, solar radiation, rain, and more. This should be sufficient information to determine some lighting conditions.

The distance and update rate do, however, mean the lighting conditions may not be the exact same at Christiansborg, so the following section will try to look at alternative means of achieving more detailed information. Another approach is to use the iPhone 4's reflected light-meter in the built-in camera making it possible to extract real-time lighting information from whatever the camera is pointed at. However, a reflected-light meter is not always accurate because all objects do not reflect the same percentage of incident light, especially not if the light is reflected off of big surfaces.

Research have shown that it is possible to estimate the light of a scene using a photography, the camera position and a 3d model of the scene [NSD94][MG97]. These reports use a method of calculating a given number of basic images with different (known) basic light directions. The photography of the scene is compared to the basic images and from a linear combination of the basic images the photography is recreated. This way it can be calculated which lights that are used to recreate the light in the scene. The big difference from these reports to this project is that they both only calculate for a single scene. For our purpose it is necessary to get the light conditions for the entire 3d scene with almost infinite viewpoints. This makes the method unusable as it first of all requires computational power to match the picture to the right spot at the 3d model and secondly requires several basic images for almost infinite viewports.

Another approach to get image based lighting is to use a light probe image [Deb05]. This requires getting high dynamic range (HDR) photographs of the entire sky. The process of getting the light probe image consists of taking several images both to cover the sky and to make the light intensity proportional to the light levels in the scene. It will not be possible to acquire such images and absolutely not in time for something close to real-time lighting and therefore this method is not looked at further.

Conclusively it seems that there is no apparent solution which would give accurate enough information to be sure that the lighting conditions in the 3d environment on the smartphone always correspond exactly to the real-life conditions. Considering the risky results of using the iPhone 4's light meter, the information from the private weather station seems more convenient, but also temporary, since this would only work for the prototype and not for a final applica-

[10] Extracting information from a web site by traversal of HTML code.

[11] `http://d.pr/WDKo`

[12] `http://weather.1806.dk/`

tion that should work in other locations. Furthermore, we would like the application to be as automatic as possible, so the private weather station in combination with DMI will be used to estimate the lighting conditions.

Properties of Daylight

The illuminance and the colour of daylight changes over the course of a day as the sun's elevation changes. Daylight is defined as the irradiance[13] coming from the combined effect of the sun light and sky light.[NW63] Colour temperature is defined as the temperature in kelvin of an ideal black body surface and can generally be viewed as higher temperatures being "cooler" while lower temperatures are "warmer".

The illuminance of daylight rises to about the double at noon of what it is at twilight for overcast days.[Lee94] For clear sky days the illuminance rises to about the triple at around noon.

[Lee94] shows that the colour temperature of a clear sky is 9,000 to 100,000 K while an overcast sky provides a colour temperature in the 6,000 - 9,000 K range. In regards to the change over the day it can be seen that with a clear sky the colour temperatures start out in the range of 30,000 to 100,000 K until noon after which it drops in a semi-exponential manner to the 9,000 to 13,000 K range. For the overcast sky however there is actually a small rise in colour temperature in the last 2 to 3 hours before the sun sets.

Sun Position and Shadows

In order to know where to cast shadows in the 3d environment one will have to know the position of the light source and the position of occluding objects and objects which should receive the shadows. The positions of objects in the scene are handled by Unity and the only light source which has to produce shadows in the scene is the simulation of the sun. The algorithm which calculates the sun position (and sun direction) is obtained from [Mic88] and takes in the time, date and the observers' location on earth as an input. The time and date are easily obtainable from the iPhone's system and the overall location is obtained by the phone's Assisted GPS functionality.

Surrounding Area and Skies

The area surrounding the models in the 3d environment will consist of a textured ground plane. Surrounding buildings and landscape will not be recreated due to the focus on Christiansborg. The skies have influence on the lighting conditions. As mentioned earlier, data for the weather will be pulled from a weather station. This data can only provide an accuracy of three different levels of clouds: Overcast, semi-overcast (sporadic clouds), and clear sky. Renders of these will have to be applied as a skybox, and these will of course also have to look realistic.

2.4.2 Lighting in Unity and iOS

When it comes to lighting, Unity supports Deferred Lighting and Forward Rendering[14][15], Deferred Lighting has the most lighting and shadow fidelity, but unfortunately it is not supported on mobile devices, so Forward Rendering becomes the obvious choice. Forward Rendering supports per-pixel lighting (enabling e.g. normal maps), i.e. the illumination is computed at each pixel of an image, which usually gives images of higher quality. The alternative is vertex lighting, a method which only calculates illuminations at each vertex and then interpolates the values. Forward Rendering also supports real-time shadows from one directional light, and it uses OpenGL ES 2.0, which iOS 4 supports. However, real-time shadows with Forward Rendering are currently not supported on the iPhone, so it is necessary to look at static ways to do the shadows.

Within static lighting the only option is lightmaps, which are pre-baked textures that are added as additional layers on top of the existing diffuse maps. Lightmaps are usually prebaked as they thereby allow for advanced lighting calculations such as Global Illumination (in-

[13] The incoming radiance at the specific point on a surface, measured in w/m^2.

[14] `http://d.pr/iasI`

[15] `http://d.pr/ICqX`

cluding colour scattering[16] and ambient occlusion[17]).

Unity has integrated the Beast lightmapping solution[18] which allows one to do this directly in Unity, alternatively one can bake lightmaps in a 3d modelling application like Maya. Compared to vertex lighting the benefit is that lightmaps scale independently of the mesh composition and thereby becomes much more flexible and allows for adjustable texel density.

Compared to real-time per pixel lighting, lightmaps are significantly faster at render time as it purely is an additional texture pass. Downside compared to both vertex lighting and real-time per pixel lighting is the additional memory required. The biggest issue with lightmaps however is that it is very slow and even a simple test scene in Unity proved to take about a minute to calculate lightmaps for, leaving some challenges for this project, as a real-time synchronised lighting will require a more dynamic technology. A likely option here is to bake a set of lightmaps for each half an hour and then load these in dynamically.

The actual surface light calculations depend on the chosen shader. Unity supports both regular diffuse with a lambertian lighting model[Lam60] and shaders for materials with specular where a Phong-Blinn lighting model[Bli77] is used. This lighting model is characterized by having three terms: Ambient, diffuse and specular and thereby adds a viewer-dependant specular highlight on the surface. The Phong-Blinn model (also called modified Phong) is a revision of the original Phong lighting model[Pho75] with static half-vector added to the equation which speeds up computations with little to no visual impact.

Summary

To sum up the following parameters concerning light and weather are relevant to the final problem statement, and will therefore be in focus during the implementation:

- Light colour
- Light intensity
- Sun position and appearance of the shadows from it
- Appearance of the sky

[16]The effect that a light ray appears to have its colour changed when bouncing of a coloured surface. This results in that nearby surfaces get a little of that colour.

[17]The effect of that corners, holes and other nearby polygons shade each other.

[18]`http://www.illuminatelabs.com/products/beast`

2.4.3 Classes for Lighting

This section will look at the specific design of all classes related to estimating and simulating lighting conditions. These classes are not changing anything during run-time but are purely used for building the correct scene setup when the application launches. The source code for the implementation can be found in the Appendix "A.2 Pseudo Code of Scene Setter Classes".

Getting Weather Information (GetWeatherFromWWW)

In "2.1.1 The Case: Locative Media and Christiansborg" it was determined that the lighting settings in the 3d environment should be done as automatically as possible. The data of the private weather station mentioned in section "Methods to Estimate Real World On-location Lighting" will be used to determine lighting conditions, even though they might not be entirely accurate for the weather at Christiansborg. Listing A.2 will explain the design of this function.

The function was implemented as planned, specifically the website was screen scraped and put into a string, and then this string could be searched through for the proper information, e.g. solar radiation.

Determining Light Colour and Intensity (LightColourDeterminer)

With the information from the **GetWeatherFromWWW** class the light intensity can then be set. The colours are found through a set of equations that have been constructured for this product, based on the research described in "2.4.1 Properties of Daylight". These equations make use of the current date and time of day to set the colour of the sunlight, as seen in listing A.2.

This function was implemented as planned and the intensity was set by taking the solar radiation variable from the **GetWeatherFromWWW** class and then applied a scaling to make the value correspond to useful intensity values in Unity.

Determining Shadows (LightmapChanger)

Since Unity does not support real-time shadows for iOS, it was necessary to use lightmapping to bake shadows, i.e. make new textures with shadows as a part of the texture. This means that a lightmap is baked for every 30 minutes, and the application then needs to change the textures as time goes by. However, the application also needs to know what castle is being shown, and to this it uses a public variable from the GUI function, listing A.3. The design of the LightMapChanger can be seen in A.2

The function was implemented as planned, specifically Unity puts lightmaps into a built-in array, each index linked to specific models, so we simply substituted the proper lightmap textures with the texture in indices corresponding to the correct model. Lightmaps were baked down using Beast, the built-in lightmapper in Unity. Quality was set to a texel value of 10 giving two 1024x1024 sized lightmaps which was found to provide a reasonable compromise between quality and performance on the iPhone.

Determining Skybox (SkyboxChanger)

Three skyboxes were created to simulate different kinds of skies, a clear sky, a sky with some clouds, and an overcast sky. The application needed to be able to switch between these by looking at online information about the current sky. This information comes in the form of an image file on `dmi.dk` which is then analysed.

This function was implemented as planned, a code simply changed the texture of the skybox material. Design can be seen in A.2

Determining Sun Direction and Position (SunDirection)

The sun's direction was used to influence the lightmaps, and during runtime it is used to determine from which side and angle the buildings are lit. This function was implemented as planned, final design can be seen in A.2. The date retrieval functionality is refactored out in a separate class called JulianDate.

The elements in the algorithm will not be explained in detail; most important is that the algorithm, because of computing considerations, treats the sun as orbiting the earth. The output is the sun position given in azimuth angle (horizontal angle) and elevation angle (vertical angle over the horizon). The position is transformed to a Cartesian coordinate system and the sun direction is calculated as the vector from the sun position to the location of the observer. The umbra[19] and the penumbra[20] of the shadows can then be determined by regular light calculations.

2.4.4 Conclusion on Designing for Similarity

The parameters determined in "2.4.2 Lighting in Unity and iOS" were found to be relevant for determining the light and weather conditions. They were all successfully implemented in the respective classes.

2.5 Additional Design Considerations

This section will look at everything concerned with the application that has nothing to do with the similarity principle. Specifically it looks at the symmetry principle, i.e. the control scheme utilising the gyroscope and GPS, the creation of a GUI, and the visual look of the 3d models.

2.5.1 Designing for Symmetry

The Gestalt symmetry principle is basically interpreted to be the control scheme for the application, as mentioned in "2.1.1 The Case: Locative Media and Christiansborg". Specifically

[19] The part of a shadow that is fully shaded.

[20] The partly shaded part of the shadow, the soft shadow component that is.

this means we try to create symmetry by allowing the user to look around in the 3d environment by using the iPhone 4 as a handheld pair of virtual reality goggles, i.e. panning the smartphone to the right means the camera in the 3d environment also pans to the right. Additionally the user can move around in the 3d environment as real-world walking is translated into similar movements on the iPhone.

Design Principles Concerning the Symmetry Principle

This section will look at the human-computer interaction principles and interface design principles used to develop good iPhone applications, specifically with focus on the control scheme using the gyroscope.

Design Principles

The application in this project will make use of what the "iPhone Human Interface Guidelines" [App10] refers to as "direct manipulation" where the user feels they are controlling something tangible, not abstract. This basically means the objects on the screen will remain visible while the user performs actions on them, and that the results of the user's action is immediately apparent, e.g. moving the phone reflects movements in the 3d environment. This also covers another important design principle; feedback, i.e. whenever the user does something the application should give feedback to the user's action.

Ease of Use

One can not assume that users have the time nor inclination to figure out how an application works, so one should focus on making it instantly understandable to users.

The "iPhone Human Interface Guidelines"[App10] does not talk about how to design the control scheme we have in mind, so instead we will look at a popular application, Layar, that use similar control schemes, as seen in figure 2.4.

This application shows you nearby points of interest, e.g. bars or restaurants by finding your location with the Assisted GPS function in the phone. As you move the smartphone around the augmented objects move as well. The difference is that this application just uses the camera to show the real world and then overlays objects,

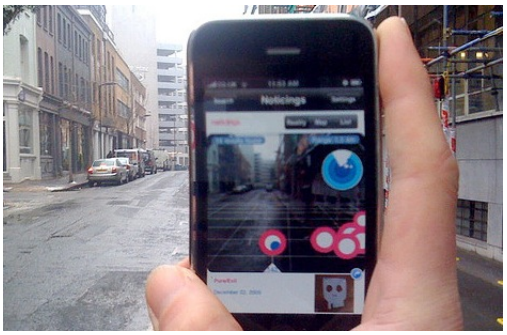

Figure 2.4: The application Layar with overlays of points of interest.

but we want to show a 3d environment instead and then move in this environment with geolocation.

To conclude, the application will use the same control scheme as Layar just in a 3d environment.

Classes for the Control Scheme

This section explains the classes used to create the gyroscope and GPS navigation. The classes are all used during run-time.

Rotation of User's Point of View in 3d Environment (DeviceMotion)

In "2.1.1 The Case: Locative Media and Christiansborg" it was determined that the user should be able to look around in the 3d environment by using the iPhone as a camera into the world, i.e. if the iPhone is panned to the right, the 3d environment moves according to the movement. Listing A.3 will explain the design of this function.

The function was implemented as planned; the rotation of the internal gyroscope was simply applied to the rotation of the camera in the 3d environment.

Translation of User Position in 3d Environment (GetGPSLocation)

In "2.1.1 The Case: Locative Media and Christiansborg" it was determined that the current location of the user would be used to place him in the 3d environment, so by using the

GPS in the iPhone, the user would be able to move around in the 3d environment by physically moving. Listing A.3 explains how this will be designed.

The function was implemented as planned, since we know the latitude and longitude of the user's location, and we know the coordinates of the camera locally in the 3d environment, we can correlate the real-life location with the correct location in the 3d environment and move the camera there.

An overview of how the classes of the entire application communicates can be seen in "2.6 System Overview".

Conclusion on Designing for Symmetry

All classes were implemented as intended. However, currently the product loses precision on the gyroscope within 1-10 minutes of use. To solve this we have added an administrative GUI which allows us to reset the gyroscope and orientation. This is obviously not acceptable for real life usage and therefore has a very high position on the list of things for further development.

2.5.2 Designing the GUI

Design Principles Concerning GUI

This section will look at the interface design principles used to develop graphical user interfaces for iPhone applications. Specifically the GUI should allow the user to go back and forward in time, to see the different buildings which has been located on the site of Christiansborg in the past.

Considerations

Before looking at specific interface considerations, device characteristics that should be kept in mind will be listed. First of all, because of the iPhone's compact screen size it is necessary to focus the user interface on the essentials, e.g. by reducing on-screen information, and keeping navigation restrictive. "HCI Beyond the GUI"[Kor08] also supports this notion.

The "iPhone Human Interface Guidelines"[App10] lists several types of application, and this project's application fits into the "immersive application" category, defined as "offering a full-screen, visually rich environment that's focused on the content and the user's experience with that content." These applications do not display a large amount of text-based information, and "they tend to hide much of the device's user interface, replacing it with a custom user interface that strengthens the user's sense of entering the world of the application".

Ease of Use

To perform the cycling through time periods to reveal the different buildings some sort of practical function which allows that needs to be available to the user. "iPhone Human Interface Guidelines" [App10] mentions two overall ways to do this; by adding some graphical user interface or by using gestures, e.g. by swiping on the screen.
Here the graphical solution is chosen, as a suitable gesture for changing the time periods where not found. Additionally the user would need some information to be able to know that this feature even exists.

Within the graphical solution it is possible to make a time slider, but these, as per "iPhone Human Interface Guidelines"[App10], useful when the user should have fine-grained control over values or some process, and that is not the case for the switching of time periods, since there will not be that many time periods to choose between.

The chosen solution is to simply make two buttons on the screen which goes to the next and previous time periods. The important aspect is that this kind of GUI is non-instrusive and that the user understands their use.

As mentioned, the amount of information and interface needs to be kept at a minimum as the focus should be at the view of models. However, it would be relevant to inform the user both with build year and name of the buildings. This is done to help the user to both differentiate the models from each other, but also to give them an understanding of how vast the jump is from one castle to another.

In a finished, commercial application it might make sense to tell the story behind these buildings and landmarks. In this project, however, the sole focus will be on the visual representation.

This section explains the class used to create the

GUI. The class is used during run-time.

GUI to Switch Between Time Periods (UserGUI)

In "2.1.1 The Case: Locative Media and Christiansborg" it was determined that a simple GUI is needed to let the user switch between the castles that once stood at the location of Christiansborg. Listing A.3 will explain the design of this function.

This function was implemented successfully, a function was made to turn the rendering of the castle models on and off depending on which castle should be shown.

2.5.3 Visual Appearance

This section will look at what buildings should be created, and how the visual appearance of the application's 3d environment, i.e. models, surrounding areas, the sky, etc. should be created. It also looks at shaders and performance related to the creation of the 3d environment on an iPhone 4. Lastly an explanation of the implementation of the chosen models is provided.

Art Direction of 3d Models

Because of the historic aspect it seems relevant to strive for a realistic look for the 3d models in the environment. This art direction was further supported by a meeting with representatives from Christiansborg Slotsforvaltning, who is in charge of Christiansborg Slotsruiner, a tourist venue at the ruins of Absalon's and Copenhagen's Castle.

There have been seven different castles throughout time at Slotsholmen, although it is hard to distinguish each version as they all had improvements and alterations during their lifespan. For the prototype it will not be possible to model all seven versions due to time limitations and it has instead been decided to focus on three versions. The choices are based on wanting to convey the development through time and the change of castles. The following castles have been chosen:

- Absalon's Castle (approx. year 1200)
- Copenhagen's Castle (approx. year 1600)
- First Christiansborg (approx. year 1750)

To achieve the realistic look all models should resemble how they were built originally. Obviously there are no actual pictures of the castles, but several artist and historians have tried to give an estimation of how the castles looked. The models Copenhagen's Castle and the first Christiansborg will be based on wood models made for the Christiansborg Slotsruiner exhibitions. These models can be seen next to their final 3d models in figures 2.6 and 2.7.

For Absalon's Castle there are less materials available. This model will at best be a very rough interpretation of the actual appearance of the castle. Research of the ruins of Absalon's Castle indicates that the castle around 1200 was reinforced by quadratic towers.[21] These towers will also be modelled. Figure 2.5 on page 15 shows the final 3d model and an approximation of the castle from the Christiansborg Slotsruiner venue.

All models should be textured in such way that they resemble the original building materials. This and the other considerations regarding the visual design of the castle models should give them a consistent appearance with a realistic look.

Shaders and Performance

To achieve additional visual fidelity one can use normal maps, an extension of regular bump maps, to give an illusion of depth on an otherwise flat 2d surface. This is done by providing a texture map with each coordinate containing an RGB value which corresponds to the xyz direction and thereby being able to create fake depth through per pixel light modifications. This gives an extra drawcall and extra memory use but on the other hand it provides detail equal to a much higher detail mesh. Specular maps is another way to improve visuals, these are slightly cheaper than normal maps and allows for the illusion of shiny surfaces.

In regards to performance the design of the graphics are of course limited by the fact that the application has to run on an iPhone 4. The

[21] `http://www.kobenhavnshistorie.dk/bog/gtkb/gtkb_bs_lex.html`

final performance is a combination of memory use, triangle count, draw calls[22], shaders and fill rate[23]. There is no clear ceiling for maximum amount of triangles but according to the Unity manual [Uni10] generally one should stay below 10,000 triangles in view. The total scene triangle count is less important as it is at most times not rendered due to the view frustum culling[24] and thereby leaving only a small effect on memory.

According to the Unity manual [Uni10] the key to good performance is to combine as much as possible. The reason for this is that drawcalls can easily be a bottleneck as this number preferably needs to be kept below 30 on the iPhone, there is however not any clear documentation on this. The way to keep this number down is to have as few meshes as possible with as few texture maps as possible. If a mesh is set as 'static' in the Unity editor, then Unity can automatically combine meshes at runtime. This has significant benefit over doing it manually as this batching is done after the visibility determination, it is however still only possible for meshes which share materials.

The iPhone has a total of 128 MiB[25] memory built in, the OS does however take most of this leaving only around 40 MiB for applications [Ali09]. These 40 MiB then have to be shared among the Unity runtime, geometry, animations, textures, code, etc. With textures one can either adjust the amount of textures or the size of textures. Unity provides excellent options for downscaling textures at project compile time and one can therefore easily decrease texture sizes if needed due to memory limitations.

Design and Implementation of Models

The models were created in Maya[26]. The textures were either cut from the pictures of the models or taken from the website CG Textures[27]. They were edited and merged with Photoshop in order to keep the number of texture maps as low as possible. Each castle model was unwrapped so it could be textured from a single texture map of size 1024 x 1024 pixels. Another texture map of same size was used for the ground. Specular maps were created by adding an alpha channel to the textures. Normal maps were either generated from greytone copies of the original textures through the built in function in unity or by the use of the external program CrazyBump[28]. Some surfaces had to have their specular or normal map dropped in order to ensure proper performance.

The final models can be seen in figures 2.5, 2.6, and 2.7 with comparisons to a sketch for Absalon's Castle and wood models for Copenhagen's Castle and Christiansborg.

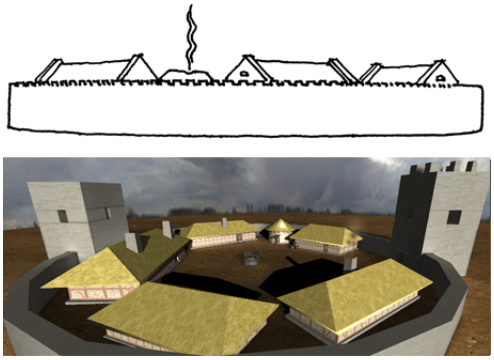

Figure 2.5: Artist's interpretation of Absalon's Castle (top) and its 3d model (bottom).

2.6 System Overview

At this point the entire system has been implemented, and it is now possible to commence testing hypothesis stated in "1.2 Final Problem Statement". First, however, this section will briefly sum up the functionality of the system. It will give an overview of the Unity scene and the classes implemented.

Unity, the chosen development platform, uses scenes and scripts. The scene is the area in which all game objects are placed, and the scripts consists of C# code which control all the underlying functionality.

The following functionality was implemented:

[22] The amount of batches in the render, usually one for each material

[23] The amount of pixels processed during the raster stage

[24] Occlusion of objects outside the field of view

[25] Mebibyte, equal to 2^{20} bytes as opposed to a megabyte's 10^6 bytes.

[26] `http://d.pr/dJih`

[27] `http://www.cgtextures.com/`

[28] `http://www.crazybump.com/`

Figure 2.6: Wood model of Copenhagen's Castle (bottom) and its 3d model (top).

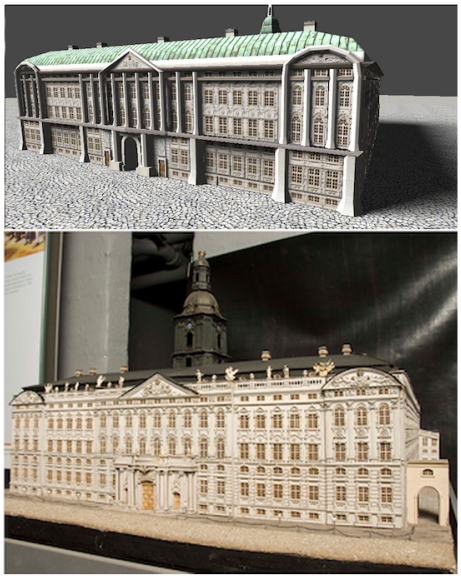

Figure 2.7: Wood model of the first Christiansborg (bottom) and its 3d model (top).

- Rotation of User's Point of View in 3d Environment
- Translation of User Position in 3d Environment
- Getting weather information
- Determining light colour and intensity
- Determining shadows
- Determining skybox
- Determining sun direction and position
- GUI to switch time periods
- GUI for admin purposes

A diagram of how these classes communicate, i.e. an overview of the entire system, can be seen in figure 2.8. The source code can be seen in simplified form in the appendix "A.2 Pseudo Code of Scene Setter Classes" and "A.3 Pseudo Code of Interaction Classes".

The final application in use can be seen in figure 2.9.

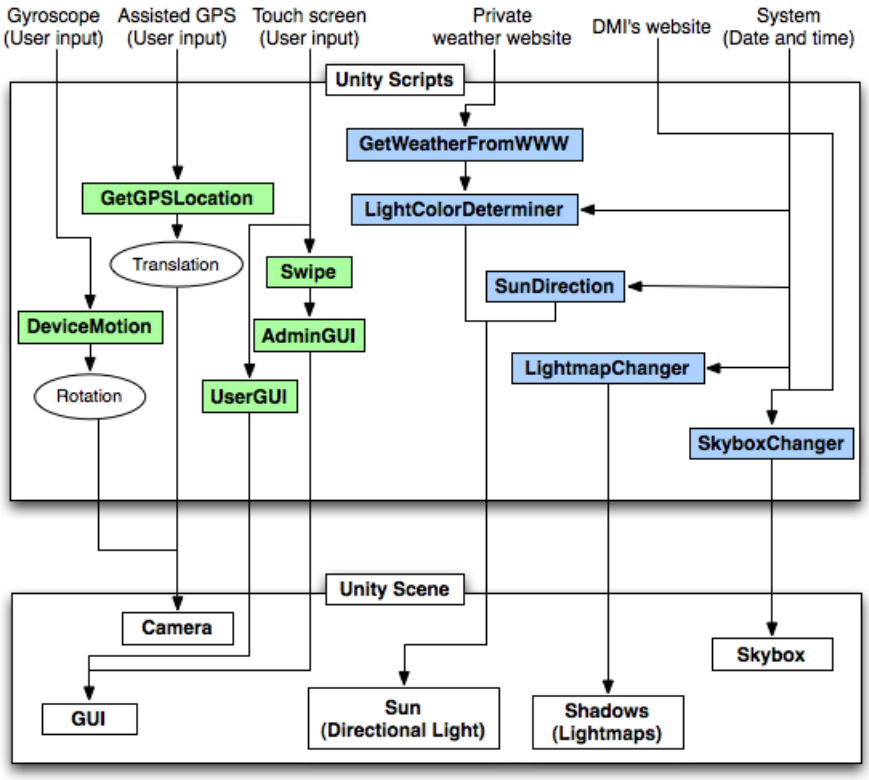

Figure 2.8: Diagram of all the classes in the application.

Figure 2.9: Final application running on an iPhone 4.

Chapter 3

Final Test

The objective of the final test is to accept or reject the hypothesis formulated in "1.2 Final Problem Statement" and is therefore referred to as the hypothesis test. The hypothesis is:

The simulation of real world lighting in the product will improve the user experience.

The chapter will focus on defining user experience in the context of the project and describing the physical test setup and the statistical basis on which the test relies on.

3.1 User Experience

User experience is a term used to describe what a user feels about using a system. Dependent of the kind of system, user experience is tested in many different ways. Although [PK07] propose a way to design location-based services, no framework exists for testing user experience within this field. In order not to invent a testing frame work from the bottom up, terms and testing concepts within the testing of computer game experience is used.

It is reasonable to use methods for testing game experience as the interaction and visual stimuli is relatable. The only mayor difference is the lack of game play within our developed iPhone4 application.

In areas where computer games and the location-based service application have less in common separate evaluation procedures have to be constructed. The evaluation of user experience will use parts of the The Game Experience Questionnaire (GEQ) from [IdKP10]. The questionnaire evaluates game experience within seven different categories:

1. Competence
2. Sensory Immersion
3. Imaginative Immersion
4. Flow
5. Tension/Annoyance
6. Challenge
7. Negative and Positive Affect

As there is no objective in the implemented system the categories about Competence (feeling proud after accomplishment of an objective) and Challenge (how different challenges in the gameplay affected the game experience) are removed. One might argue that Flow can be problematic as it tends to deal with the user forgetting the world around him, something which is positive for a regular computergame but less so for an LBS, as the location-to-user interaction is an important aspect aswell. This category is however still included as the introduction to the questionnaire has informed the user in such way the Flow questions are unambiguous in relation to an LBS.

3.2 Test Setup

The hypothesis test is conducted in the surroundings of Christiansborg. This area is not chosen on behalf of test issues but solely because a place with a high historical diversity was needing in order to make a interesting product in a historical perspective. The test participants are randomly divided in two groups. One group is testing the system with the static lighting conditions and is called the control group. The other half is testing the time dependent generated system and is called the experimental group. Table 3.1 shows the difference in the two systems

Function	Null Control group	Experimental group
Sun position	Noon of test day	Calculated on start-up
Sun direction	Noon of test day	Calculated on start-up
Sun colour	Noon of test day	Calculated on start-up
Sun intensity	Noon of test day	Solar radiation from weather station pulled on start-up
Shadows	Noon of test day	Lightmap chosen on start-up (changed every 30-minutes)
Sky maps	Partly overcast	Weather-data pulled from dmi.dk on start up

Table 3.1: Difference in level of similarity between the two tested systems.

tested.

The participants are not told that there exists two different types of the system and the real testing purpose is thereby never reveiled. In order to produce reliable results only the difference in lighting system is treated as a testing variable. All other test conditions are the same in the two groups.

Where possible the users are put together in small groups of 2-4 people. This is done so the population size is as high as possible with limited testing time and only one iPhone available for testing.

The test is conducted between 10 a.m. and 4 p.m. and is completed over three days. The small groups of testers are scattered throughout the day, a new group starting up every 30 minutes. The two versions of the system are tested on separately days. The aim is to have the same kind of weather on each of the days, in order to eliminate the error produced by letting the weather be a variable in the test. Preferably both systems should have been tested at the same time, but the limited number of iPhones made it impossible.

The reason for testing the system throughout a day is to investigate how the different lighting conditions over a day contribute to the user experience. It is crucial that both populations have the same amount of participants on the same time of day.

The instructions given to the small user groups are kept to a minimum. They are told the following:

1. The application shows historical castles and fortresses on the location of the present Christiansborg.
2. They are allowed to go anywhere they want, but are told that no other parts of Copenhagen are modelled in the application.
3. All members have to try to use the system and thereby be the one holding and orientating the smartphone.
4. They are allowed to keep using the application as long as they like.

Hereafter the group is on its own using the application. In reality the group is not allowed to test as long as they like, but is interrupted after 10-15 minutes of use. If not all members have tried to hold the phone within this time limit they are asked to do so. After the use of the application every participant is asked to fill out a questionnaire. The evaluations of user experience solely rely on the individuals and are not treated in relation with the small groups.

For complete list of questions see “A.1 Final Test Questionnaire”.

3.3 Statistics of the Test

The test data consists of answers on a 5-point Likert scale to 21 questions for every test participant. For every completed questionnaire a mean value for each of the categories is calculated as a mean of all questions in that category. To statistically prove a difference in user experience between the two populations (one testing static light, the other time dependent light) a t-test on the difference between means in all six categories respectively is put up. In order to make a conclusion with a minimum error atleast 30 test participants in both samples is needed [Joh05].

In table 3.2 the null and alternative hypothesis on the hypothesis test on the difference between μ_{gen} and μ_{real} is showed:

To backup the statistical conclusions from the

Category	Null hypothesis	Alternative hypothesis
Flow	$\mu_{gen_f} - \mu_{real_f} = 0$	$\mu_{gen_f} - \mu_{real_f} > 0$
Sensory and Imaginative Immersion	$\mu_{gen_{im}} - \mu_{real_{im}} = 0$	$\mu_{gen_{im}} - \mu_{real_{im}} > 0$
Positive Affect	$\mu_{gen_{pa}} - \mu_{real_{pa}} = 0$	$\mu_{gen_{pa}} - \mu_{real_{pa}} > 0$
Negative Affect	$\mu_{gen_{na}} - \mu_{real_{na}} = 0$	$\mu_{gen_{na}} - \mu_{real_{na}} < 0$
Tension/Annoyance	$\mu_{gen_{ta}} - \mu_{real_{ta}} = 0$	$\mu_{gen_{ta}} - \mu_{real_{ta}} < 0$

Table 3.2: Null and alternative hypothesis of Z-test on the five categories of user experience.

test on the hypothesis, some additional questions are added to the questionnaire. The aim is to prove the level of similarity and symmetry in the system. These qualities where found to be important when enhancing the user experience in an LBS-system. Where the GEQ-questions aim to detect a general user experience, the additional questions are solely put in the questionnaire to show differences between similarity and symmetry in the two instances of the system. All test participants are observed during the test. These observations go in two categories.

- Observation on the route of the testers and the time spent
- Observation on the use of the system

A correlation between the user experience and the time and place spent on testing the product could be found. In order to prove the existence of such a correlation the route and time spend for every user is noted down. In the same way problematic interactions between the user and the system are observed and reported in order to secure that bad interaction design is not in the way of making valid testing results.

3.4 Results

The test was conducted on 9th - 11th of December 2010. On the 9th of December 27 participants tested the static system. The 10th of December 27 new participants tested the time dependent system. The last day five participants completed the test and the day was used to fill out blanks in regards to hour intervals with few test participants. In total 59 people completed the test, 44 men and 15 women. The control group consisted of 29 people, the experimental group of 30. The mean age was 29 years. A picture of the test can be seen in 3.1.

Figure 3.1: A picture of one of the test subjects during the test.

The result on the hypothesis test on improvement of user experience with the use of real world lighting was as follows:

Valid for all tests:

1. Level of significance: $\alpha = 0.05$
2. $z_\alpha = 1.645$

As shown in table 3.3 the five different Z-values are all close to zero. This means that there exists a very small difference between the means off the time dependent version compared to

Category	Test z-value	Reject null hypothesis if	Result
Flow	-0.20	$Z > z_\alpha$	Not rejected
Sensory and Imaginative Immersion	0.09	$Z > z_\alpha$	Not rejected
Positive Affect	-0.35	$Z > z_\alpha$	Not rejected
Negative Affect	0.12	$Z < -z_\alpha$	Not rejected
Tension/Annoyance	1.04	$Z < -z_\alpha$	Not rejected

Table 3.3: Results on Z-test on the two means.

the static, see figure 3.2. In four out of five categories the user experience is rated higher (for Tension/Annoyance and Negative Affect the higher mean the lower user experience rating) in the static version. Only in the category of Sensory and Imaginative Immersion the magnitude of the means are greater for the time dependent version as stated in the hypothesis.

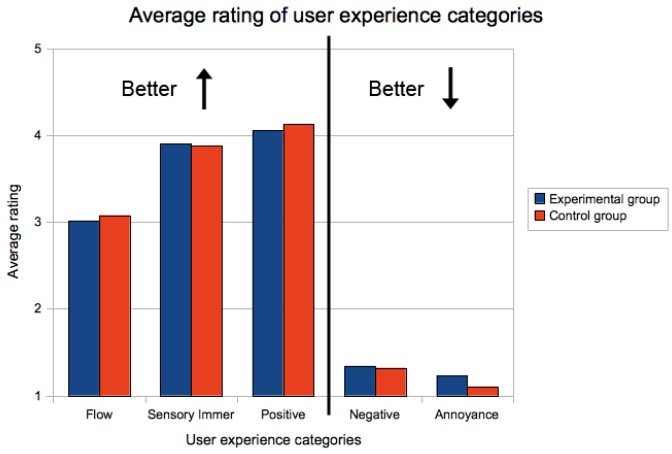

Figure 3.2: Comparison of means of all five categories between the two groups.

Category	Control group	Exp. group
Flow	1.23	1.13
Sens./Imag. Immer.	0.96	1.05
Positive Affect	0.78	0.83
Negative Affect	0.80	0.69
Annoyance	0.34	0.58

Table 3.4: Standard deviation of the five categories.

The distribution of the results on user experience over the span of the test period from 10 a.m to 4 p.m. is showed for each category individually, see figure 3.5 to 3.7. Here all tests done within one hour are grouped together. Be aware that the time intervals do not represent the same amount of test participants. Figure 3.5 show the distribution of tests done during the two days.

Time interval	Control group	Exp. group
9:00-10:00	0	2
10:00-11:00	1	2
11:00-12:00	8	5
12:00-13:00	4	4
13:00-14:00	4	5
14:00-15:00	6	5
15:00-16:00	6	7
9:00-16:00	**29**	**30**

Table 3.5: Distribution of test participants.

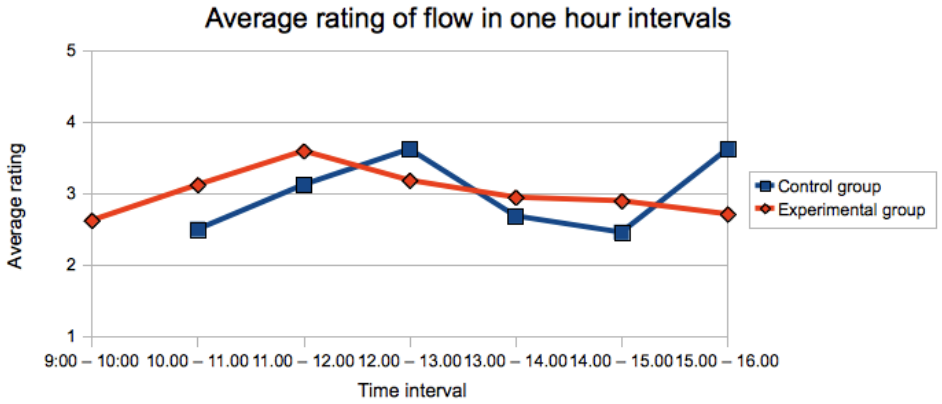

Figure 3.3: Average rating of Flow in one hour intervals.

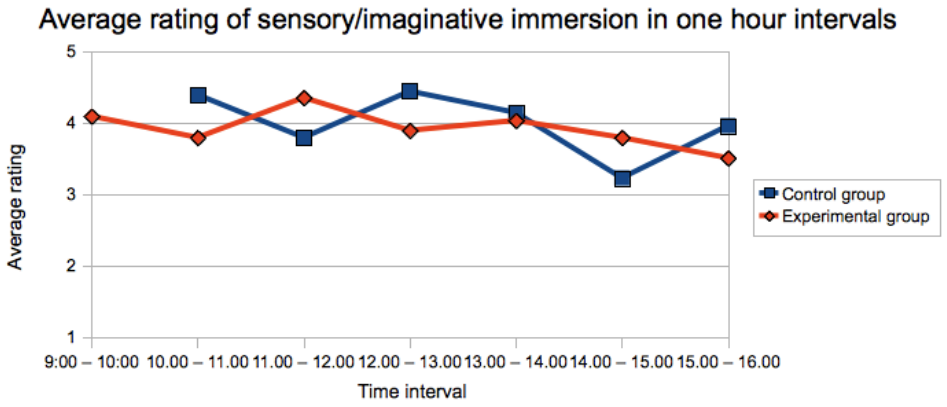

Figure 3.4: Average rating of Sensory/Imaginative Immersion in one hour intervals.

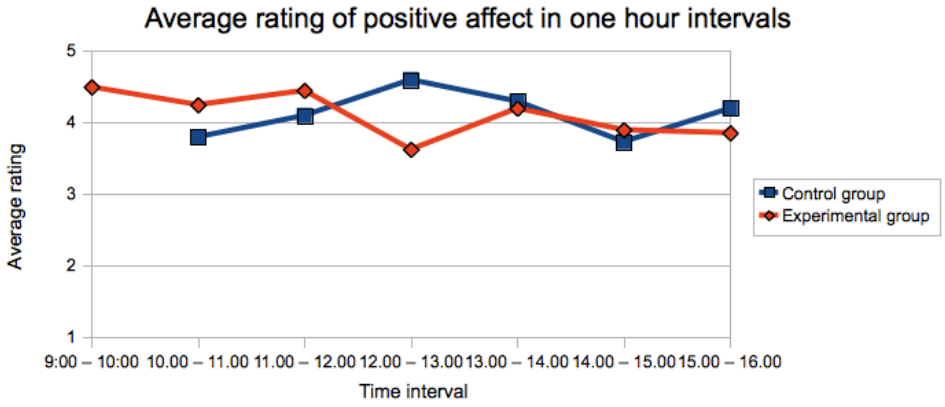

Figure 3.5: Average rating of Positive Affect in one hour intervals.

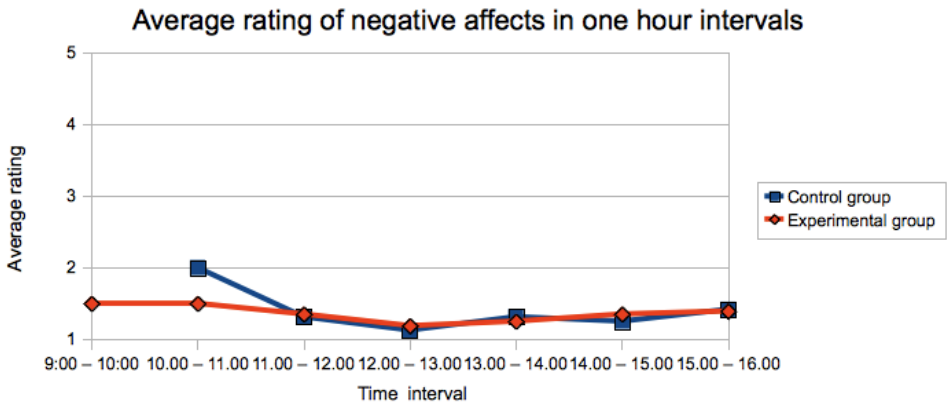

Figure 3.6: Average rating of Negative Affect in one hour intervals.

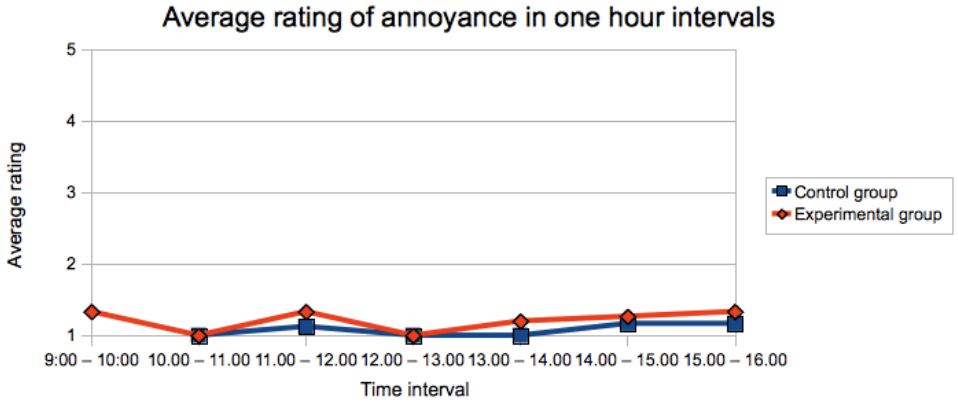

Figure 3.7: Average rating of Tension/Annoyance in one hour intervals

Chapter 4

Discussion

When analysing the results it was discovered that the mean of the ratings for the control group and for the experimental group were similar in each category, as seen in figure 3.2. However, the similarity of the means do not say anything about the shape of the graphs, so visualisations are created in figures 3.3 to 3.7, where it also is obvious that the graphs differ even though the means are similar. These figures are also made to be able to compare the results with figure 2.1, that shows the expected results if the hypothesis is true. Looking at all figures it becomes apparent that none of them come close to fulfilling the expected results.

The graphs for Flow (figure 3.3), Sensory and Imaginative Immersion (figure 3.4) and Positive Effect (figure 3.5) are characterised by the fact that the high rating is alternating between the groups for each hour. It should here be mentioned that to achieve the same amount of statistical validity as in the hypothesis then there should have been 30 test persons for each hour interval, both for the control group and the experimental group. Since there instead was only 1-8 test persons for each hour these results have a certain amount of uncertainty for the time intervals. The control group and experimental group graphs for Negative Effect (figure 3.6) and Annoyance/Tension (figure 3.7) are very similar as opposed to the other three graphs mentioned above.

Since the differences between the mean values of the control group and the experimental group are so small there is not statistical merit for concluding anything on which of the versions that provided the best user experience. This is due to the fact that the differences in mean values in each category are small compared to the standard deviation between the control group and the experimental group.

There are multiple scenarios which can explain these very low differences. Here we describe several possible conclusions that might explain where things have gone wrong in the process of demonstrating the hypothesis to be true, i.e. why the differences are so low.

1) There is no correlation between the similarity principle and user experience in regards to locative media.

The idea that the similarity principle is important for locative media comes from [PK07]. In their projects however, they have not statistically demonstrated that the principle has a direct influence on user experience. Instead they have taken the opposite route to us by creating an LBS and observing it in use. After this they categorised the found issues in categories after the Gestalt principles and conclude that one can avoid issues and thereby increase user experience by designing for these principles. We started by designing specifically with the Gestalt principles in mind with the goal to demonstrate a higher user experience due to this.

If [PK07]'s hypothesis about the Gestalt principles does not hold, i.e. if it has no effect on the user experience, this project is useless as their theory is the basis for our hypothesis. We do however think it is possible that there is a correlation between similarity and user experience and that the results can be explained by looking in other directions.

2) We haven't been able to create two systems with significantly different amounts of our kind of similarity.

The goal of this project was to create a significant difference in similarity for the two versions of the system. We chose to do that using real life

on-location lighting conditions. Specifically we chose to examine the degree those two systems differed in that aspect, by asking test users to rate how much they felt the lighting in the 3d environment corresponded to the lighting conditions in the real world. Here there was almost no difference on the mean values of the answers between the two groups.

A factor that might explain this became apparent during implementation; i.e. that since the testing was conducted during the winter, the difference in lighting between the two versions of the application is minor compared to, say, an average summer day. This might explain the low differences, which means that the similarity principle can still have an effect on user experience, even though our results did not demonstrate this. Lastly we look at possible flaws in our testing method when it comes to evaluating lighting conditions used as the similarity principle.

3) Lighting conditions are such a small part of the user experience that the similarity principle can not be measured in this way.

If one, as discussed in 1), assumes that there exists a connection between the similarity principle and user experience and considers that we have tested on user experience in a broader perspective, because we used a general framework from [IdKP10], then one can conclude that similarity needs to be a more significant factor for user experiences before a statistical difference can be seen in our results. So maybe if a test was designed to look more specifically at the similarity principle the results would be different.

Lastly some considerations on the setup and execution of the test will be discussed.

The two systems should have been tested at the exact same time, not just same time of the day on different days. This stems from the fact that the different weather conditions on the different days might have affected the user experience differently. Although variation in the weather between the days was not pronounced, the small differences must be regarded as uncontrolled test variables. Particularly in this project these variables should have been controlled since they have an affect on the test results because the weather is closely dependent on the focus of the project: producing real world lighting conditions.

In "3.2 Test Setup" it was mentioned that the users should be on their own testing the product. The reason was that this would prevent the people conducting the test (us) to influence the test participants and thereby the test results. This requirement was often violated, resulting in us answering questions and thereby potentially revealing too much of the purpose of the test.

It should also be mentioned the system had some bugs that might have affected the test results. Specifically the gyroscope drifted after some minutes, meaning that the buildings would not be aligned with the real-life buildings. To fix this the administrative GUI was used to reset the calibration and as such it did not prove to be a real problem during testing, but this should obviously be fixed for a more final product.

Chapter 5

Conclusion

In this project a locative media that allowed the user to explore the historical development of Slotsholmen in Copenhagen was designed and implemented. A test was carried out based on the Gestalt principle of similarity in relation to designing a location-based service. Specifically the test was aimed at determining if real world on-location lighting would improve the user experience. To conclude the project the successfulness of the final problem statement will be determined.

The final problem statements is:

How is the user experience of a locative media, that conveys the historical development of Christiansborg's appearance, affected if on-location real world lighting is used?

This was accompanied by a hypothesis containing the assertion that on-location real world lighting would have a positive effect on user experience. This hypothesis was tested on the final product.

The question at this point is, was our implementation actually good enough to provide valid data? As in, does the test results actually show an answer to our hypothesis or are they merely limited by the shortcomings of our implementation? As discussed in "4 Discussion" there were some issues with the implementation but we still believe the results to be valid. However, the comparison of the results from the control group and the experimental group showed very similar graphs. In "4 Discussion" we suggest why we think this happened, and we believe the most important answer is the 2nd suggestion, i.e. the difference between the lighting conditions was too small between the two versions of the product. The Gestalt principles may not be useful in relation with LBS's (suggestion #1) and the test may be flawed (suggestion #3), but we can not conclude that any of these factors affected the test results without first improving the lighting difference between the two application versions.

To sum up, based on the test results we can not conclude whether or not real world on-location lighting affects the user experience. Furthermore we believe the test results are so similar because of an insignificant lighting difference between the two versions of the application.

Bibliography

[Ali09] Maher Ali. *iPhone SDK Programming: Developing Mobile Applications for Apple iPhone and iPod Touch*. John Wiley and Sons, 2009.

[App10] Apple Inc. *iPhone Human Interface Guidelines*, 2010.

[Bli77] J.F. Blinn. Models of light reflection for computer synthesized pictures. *ACM SIGGRAPH Computer Graphics*, 11(2):192–198, 1977.

[Deb05] Paul Debevec. Image-based lighting. In *ACM SIGGRAPH 2005 Courses*, SIGGRAPH '05, New York, NY, USA, 2005. ACM.

[Eps09] Michael Epstein. Moving story. *Media in Transition 6: Stone and Papyrus*, 2009.

[IdKP10] W.A. IJsselsteijn, Y.A.W. de Kort, and K. Poels. The game experience questionnaire: Development of a self-report measure to assess the psychological impact of digital games. *In preparation*, 2010.

[Joh98] P Johnson. Usability and mobility; interactions on the move. *Proc. Mobile HCI 1998*, 1998.

[Joh05] R. Johnson. *Probability and Statistics for Engineers*. Miller and Freund, 2005.

[Kor08] Philip Kortum. *HCI Beyond the GUI - Design for Haptic, Speech, Olfactory, and Other Nontraditional Interfaces*. Morgan Kaufmann, 2008.

[Lam60] J. Lambert. Photometria Sive de Mensura et Gradibus Luminus. *Colorum et Umbrae", Eberhard Klett*, 1760.

[Lee94] Rl Lee. Twilight and Daytime Colors of the Clear Sky. *Applied optics*, 33(21):4629–4638, 1994.

[Lem09] André Lemos. Locative media in brazil, 2009.

[LWL+06] Yue Liu, Yongtian Wang, Yu Li, Jinchao Lei, and Liang Lin. Key issues for ar-based digital reconstruction of yuanmingyuan garden. *Presence: Teleoperators and Virtual Environments*, 15(3):336–340, 2006.

[MBN+05] M. Mabrouk, T. Bychowski, H. Niedzwiadek, Y. Bishr, JF Gaillet, N. Crisp, W. Wilbrink, M. Horhammer, G. Roy, and S. Margoulis. OpenGIS location services (OpenLS): Core services. *OGC Implementation Specification*, 5:016, 2005.

[MG97] Stephen R. Marschner and Donald P. Greenberg. Inverse lighting for photography. In *In Fifth Color Imaging Conference*, pages 262–265, 1997.

[Mic88] Joseph J. Michalsky. The astronomical almanac's algorithm for approximate solar position (1950-2050). *Solar Energy*, 40(3):227 – 235, 1988.

[NSD94] Jeffry S. Nimeroff, Eero Simoncelli, and Julie Dorsey. Efficient re-rendering of naturally illuminated environments. In *Fith Eurographics Workshop on Rendering*, pages 359–373, 1994.

[NW63] Y. Nayatani and G. Wyszecki. Color of daylight from north sky. *J. Opt. Soc. Am*, 53:626–629, 1963.

[OCT+03] Sharon Oviatt, Rachel Coulston, Stefanie Tomko, Benfang Xiao, Rebecca Lunsford, Matt Wesson, and Lesley Carmichael. Toward a theory of organized multimodal integration patterns during human-computer interaction. In *Proceedings of the 5th international conference on Multimodal interfaces*, ICMI '03, pages 44–51, New York, NY, USA, 2003. ACM.

[Pho75] B.T. Phong. Illumination for computer generated pictures. *Communications of the ACM*, 18(6):311–317, 1975.

[PK07] Jeni Paay and Jesper Kjeldskov. *A Gestalt Theoretic Perspective on the User Experience of Location-Based Services*. Association for Computing Machinery, 2007.

[PK08] Jeni Paay and Jesper Kjeldskov. Understanding the user experience of location based services. *Journal of Location Based Services*, 2(4):267–286, 2008.

[PRT06] RM Pulselli, C. Ratti, and E. Tiezzi. City out of chaos: social patterns and organization in urban systems. *International Journal of Ecodynamics*, 1(2):125–134, 2006.

[RGKR07] J. Raper, G. Gartner, H. Karimi, and C. Rizos. A critical evaluation of location based services and their potential. *Journal of Location Based Services*, (1):5–45, 2007.

[Sec01] Second International Conference on. *Developing GIS-supported location-based services*, volume 2, 2001.

[SNE06] Stefan Steiniger, Moritz Neun, and Alistair Edwardes. Foundations of location based services. *University of Zurich*, 2006.

[TV06] Marc Tuters and Kazys Varnelis. Beyond locative media: Giving shape to the internet of things. *Leonardo*, 39(4):357–363, 2006.

[Uni10] Unity Technologies. *Unity Manual*, 2010.

[Wol09] Jeremy M. Wolfe. *Sensation and Perception*. EB Goldstein, 2009.

Appendix A

Appendix

A.1 Final Test Questionnaire

Final test questions answered on the Likert scale.

User experience questions:
Flow:
I was fully occupied with the application
I forgot everything around me
I lost track of time
I was deeply concentrated using the system

Sensory (and Imaginative) Immersion:
It was aesthetically pleasing
I felt imaginative
I felt that I could explore things
I found it impressive
It felt like a rich experience

Negative affect:
It gave me a bad mood
I thought about other things
I found it tiresome
I felt bored

Tension/Annoyance:
I felt annoyed
I felt irritable
I felt frustrated

Positive affect:
I felt content
I thought it was fun
I felt happy
I felt good

I enjoyed it

Additional questions:

I feel that the proportions between the buildings on the iPhone and the real world is believable.
I feel that there is a correlation between my own movements and what I see on the iPhone
I feel that the lighting of the old castles corresponds to the lighting conditions in the real world
I feel that it is easy to navigate around amongst the old castles on the iPhone
I feel that there is a connection between what I see on the iPhone and the place I am standing
I feel that it was interesting to explore the old castles by using the iPhone
I feel that the amount of information about the castles was fitting
Whe the test was done I was curious to know more about Christiansborg
I feel that the iPhone is a a good way to experience the old castles
I feel that it makes sense to experience the old castles on the iPhone today
I prefer to read about the old castles on a poster with pictures as opposed to on the iPhone experience
I feeel that the representation of the old castles was believable
After the test was done I was interested in exploring the old castles on the iPhone further

To how large a degree did you feel that there was a correlation between the real world and the iPhone in regards to:
The weather
Lighting conditions
The sky
The sun
The season
Shadows

If you have comments to the project you are welcome to add them here:

I was interested in exploring the environment.
I want to keep exploring the world.

Location-based Services questions:
I felt a connection between the real world and the historic world.
I did not lose connection with the outside world

A.2 Pseudo Code of Scene Setter Classes

GetWeatherFromWWW

```
GetWeatherFromWWW
{
  if iPhone has access to Internet continue else wait;

    get source code of website with weather information;

    find relevant information in source code;

    extract information;
```

```
}
```

Listing A.1: Getting and using weather information.

LightColourDeterminer

```
LightColourDeterminer
{
  get solar radiation info from GetWeatherFromWWW;

    use solar radiation to set light intensity in 3d environment;

  get current date and time;

    use date and time to set colour of light in 3d environment;
}
```

Listing A.2: Setting light intensity and colour.

LightmapChanger

```
LightmapChanger
{
  get current time and castle;

  apply proper lightmap determined by time and castle;
}
```

Listing A.3: Changing the baked shadow textures dynamically.

SkyboxChanger

```
SkyboxChanger
{
  get online image with weather information;

  analyse part of image where the information about the current sky is;

  set proper skybox determined by image analysis;
}
```

Listing A.4: Changing the skybox dynamically.

SunDirection

```
SunDirection
{
  get current date and time;

  calculate position and direction of the sun;

  set sun position and direction in 3d environment;
}
```

Listing A.5: Setting the direction of the sun.

A.3 Pseudo Code of Interaction Classes

DeviceMotionManager

```
DeviceMotion
{
  each frame
  {
    set camera in 3d environment's rotation to rotation of iPhone's gyroscope;
  }
}
```

Listing A.6: Getting and using gyroscope information.

GetGPSLocation

```
GetGPSLocation
{
  if iPhone has connection to GPS function continue else wait;

  get initial latitude and longitude;

  loop
  {
    get current latitude and longitude;

    calculate distance between initial and current latitude and longitude;

    apply translation to camera in 3d environment;

    set initial latitude and longitude to current latitude and longitude;
  }
}
```

Listing A.7: Getting and using GPS location.

GUIManager

```
UserGUI
{
  create button ''next'';
  create button ''previous'';

  each frame
  {
    if ''next'' is pressed show next castle;
    if ''previous'' is pressed show previous castle;
  }
}
```

Listing A.8: Creating castle switcher GUI.

AdminGUI

```
AdminGUI
{
  create button ''start gyroscope''
  create button ''reset gyroscope''

  if swipe
  {
    toggle display of gui;
  }
}
```

Listing A.9: Creating administrative GUI.