VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMEDIÍ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

AUGMENTED REALITY IN CAVE

DIPLOMOVÁ PRÁCE
MASTER‘S THESIS

AUTOR PRÁCE BC. MICHAL KOLČÁREK
AUTHOR

BRNO 2013

ROZŠÍŘENÁ REALITA V CAVE

AUGMENTED REALITY IN CAVE

DIPLOMOVÁ PRÁCE
MASTER‘S THESIS

AUTOR PRÁCE Bc. MICHAL KOLČÁREK
AUTHOR

VEDOUCÍ PRÁCE Doc. Ing. ADAM HEROUT, Ph.D
SUPERVISOR

BRNO 2013

Abstrakt

Tato práce se zaměřuje na technologii Cave Automatic Virtual Environment a konkrétně pak na využití principů rozšířené reality v tomto prostředí. Dává si za cíl odpovědět na otázku, zdali je možné použít v prostředí CAVE existující frameworky pro rozšířenou realitu, konkrétně ty, pracující na platformě iOS. Hlavní důraz je kladen na rozpoznávání markerů v tomto prostředí a na zvýšení přesnosti jejich rozpoznání. Práce odpovídá na množství otázek z této oblasti, jako jaké markery je vhodné použít, jaké jsou omezení a největší obtíže. Výstupem je demonstrační aplikace, pracující na platformě iOS, která v je prostředí CAVE otestovaná a plně použitelná. Tato aplikace by měla vylepšit uživatelský vjem z prostředí CAVE tím, že mu poskytne dodatečné informace a také základní možnosti interakce se zobrazenými objekty.

Abstract

The aim of this thesis is to develop an application for mobile platform iOS. This application will take advantages of augmented reality and Cave Automatic Virtual Environment. Main target of this thesis is to explore possibilities and obstacles of developing application for iOS platform using standard augmented reality framework in CAVE. Main part is dedicated to experimenting with markers in CAVE, which leads to obtaining some knowledge leading to designing appropriate markers and optimizing the recognition process. Its purpose is the improvement of user experience in this environment and provision of some additional information and possibilities of interaction with the object displayed.

Klíčová slova

iOS, rozšířená realita, CAVE, Cave Automatic Virtual Environment, OpenGL ES, Cocos2D, Vuforia SDK, marker, markerless, uživatelská interakce, rozpoznávání, 3D projekce

Keywords

iOS, augmented reality, CAVE, Cave Automatic Virtual Environment, OpenGL ES, Cocos2D, Vuforia SDK, marker, markerless, user interaction, recognition, 3D projection

Citace

Kolčárek Michal, Bc.: Augmented Reality in CAVE, Brno, FIT VUT v Brně, 2013

Augmented Reality in Cave

Prohlášení

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně pod vedením Doc. Ing. Adama Herouta Ph.D.
Další informace mi poskytli Jussi Kangasoja a Pekka Nisula.
Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

……………………
Bc. Michal Kolčárek
Datum (20. května 2013)

Poděkování

Chtěl bych na tomto místě poděkovat svému vedoucímu doc. Ing. Adamu Heroutovi, PhD. za vedení práce, Jussimu Kangasojovi z Oulu University of Applied Sciences za dohled nad vypracováním práce v Oulu ve Finsku. Dále pak Pekkovi Nisulovi za odbornou pomoc s technologií vývoje pro iOS a s frameworkem Vuforia SDK a také Jannemu Kumpuojovi za pomoc s obsluhou zařízení CAVE.

Table of Contents

1 Introduction

Computer graphics is very progressive area of IT. Power of all computing devices is increasing rapidly and the technology is available to nearly everybody. Huge growth is exploring also segment of mobile devices with tablets and smartphones. Customer can get a mobile device, which is more powerful than standard personal computer was less than ten years ago, for quite reasonable price. The importance of basic mobile phone use, making phone calls, is decreasing. With the new era of devices, mobile phone is more of a fashion accessory, multimedia device and a way to stay connected to the Internet all the time. This demands also increasing graphics power. Mobile phones and tablets have now enormous screen resolution so it is becoming to be quite common, that for example 9,7 inch display on the New iPad has higher resolution than 42-inch television. Handling more than 3 million pixels on this device in real time and high frame per second rate requires multi-core graphics and multi-core CPU. This power allows usage of this device for quite complex graphical operations, including modern sophisticated 3D games.

With 3D games come in hand the possibilities of virtual reality. Computer games of today have graphic interface so close to the real one, that it is sometimes a problem to say whether the image on the screen is a real scene from a real life or an artificially created image. Areas of virtual reality usage are vast, for example nowadays popular virtual tour through world metropolises.

This thesis deals with augmented reality, where virtual and real worlds come together to create a new form of user experience. In the eyes of many, augmented reality has a great future. When the augmented reality meets the increasing power of mobile devices, there is a huge possibility to provide the user an amazing experience without a need for nothing else, just his smartphone or tablet. More information about the foundations of augmented reality and mobile devices will be covered in the Chapter 2.

Another aim of this thesis is the CAVE technology, where the advantages of mobile devices could improve the experience of the user. The CAVE technology is still quite new and not known widely, so there is a closer look to it in Chapter 3. In this part will be also described the specific CAVE, where this thesis was developed, because it is different to conventional CAVE system.

Chapter 4 takes closer look on the frameworks that can be used in mobile devices to provide the user augmented reality experience. Basic information, development philosophy, advantages and disadvantages will be provided. In the end of the chapter, decision about the framework used for developing the application will be made.

In the next chapter, Chapter 5, some detailed information about object detection and markers will be covered. The terms marker augmented reality and markerless augmented reality will be explained. Basic fact about object recognition will be given with special aim to the Vuforia SDK recognition system, also examples of appropriate and inappropriate markers will be shown.

Chapter 6 is the main part of the thesis and it covers the topic of using markerless augmented reality in the CAVE. Series of experiments are described in this chapter. Those experiments reveal some fundamental information about what is and what is not possible in CAVE in terms of marker recognition possibilities and usability of mobile device in the environment. These information provide background knowledge for designing markers usable in the CAVE environment and optimization for increasing the precision or recognition process.

Several issues connected with developing an application using Vuforia SDK will be covered in Chapter 7. Importing models and adding custom markers are the main topics of this chapter. These issues are based on real problems, which the author of this thesis had to deal with, while developing the demonstration application for CAVE use.

In Chapter 8 covers the actual application for CAVE will be described. The main purpose of the application is to demonstrate what is possible to do; the main idea is to provide some possibilities of interaction using touches and gestures to the user to make the model projected into CAVE more interactive for him.

In second to last chapter, Chapter 9, the developed application will be tested and some practical knowledge about possibilities and limitations of the application will be provided. Also some issues connected with CAVE use will be covered.

Chapter 10 will sum the results of this thesis, provides some fundamental knowledge and discoveries about the connection of augmented reality and CAVE advantages and contributions to more user friendly usability of the CAVE. Also the subsequent possibilities to the thesis are mentioned there.

This thesis is subsequent to a Term Project with the same name. From the project are taken chapters 2, 3 and partly 4.

2 Augmented reality

Augmented reality is a variation of virtual reality, where user sees virtual objects placed in the real environment. It supplements reality with some virtual objects, but it does not replace reality completely, so the objects in the view are both real and virtual coexisting together. Virtual objects provide some kind of information, which is cannot be detected directly by user senses. It is advantage when the virtual objects appear in 3D.

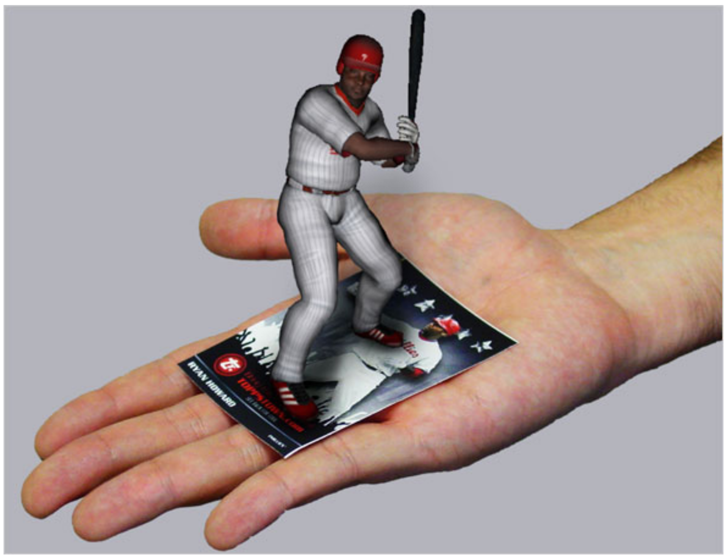


Figure 2-1: Example of augmented reality [3]

In augmented reality the basic space is real, virtual reality takes user to imaginary world replacing the real world. Another term in this area is augmented virtuality, which places real object into a virtual environment. Example of augmented virtuality can be modern television studios, where the reporter stands in a virtual place. The area of augmented reality, augmented virtuality and phases between is sometimes called mixed reality. Figure 2-2 demonstrates the whole situation.

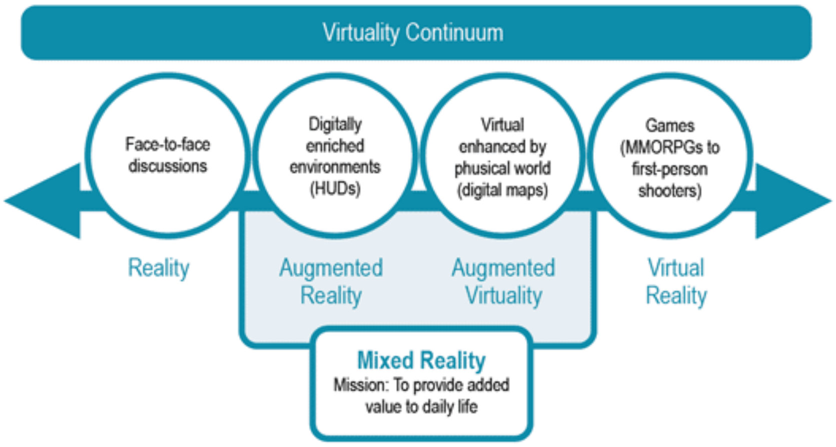

Virtuality Continuum
Face-to-face discussions
Digitally enriched environments (HUDs)
Virtual enhanced by phusical world (digital maps)
Games (MMORPGs to first-person shooters)
Reality
Augmented Reality
Augmented Virtuality
Virtual Reality
Mixed Reality
Mission: To provide added value to daily life

Figure 2-2: Illustration of spectrum between real and virtual world [4]

2.1 History

Augmented reality term has been around since 1990 [5] but its roots have been there for much longer. It all started in 1957 with machine called the Sensorama, which was built by Morton Helig. This machine provided experience for all the users senses. For example it could blow wind, vibrate seat and even project a 3D environment. Originally it should provide an experience of bike ride through Brooklyn, but it was very expensive to make films in 3D for it. It had most of the signs of virtual reality, but there were also marks of the augmented one, because the environment projected to the user was after all recorded reality.

In 1966 one of the key inventions in this area was brought out, because Professor Ivan Sutherland came out with the first model of head mounted display (HMD), but it was too heavy to be mounted directly to someone head, so it had to be suspended from the ceiling. Use of this device was very limited, but it was the first real step to make augmented reality possible.

Professor Tom Caudell established the phase, augmented reality, while working in research facility of Boeing Company in Seattle. He used it to help manufacturing process in aviation industry where it helped mechanics to place the parts of system to certain spot.

In the 1992 two other projects made big steps in augmented reality. First of them was the first working augmented reality project developed for United States Air Force known as virtual fixtures where fixtures were cues to guide the user in the task. The second one used a 3D graphics to show user how to use a printer without a need to read the manual.

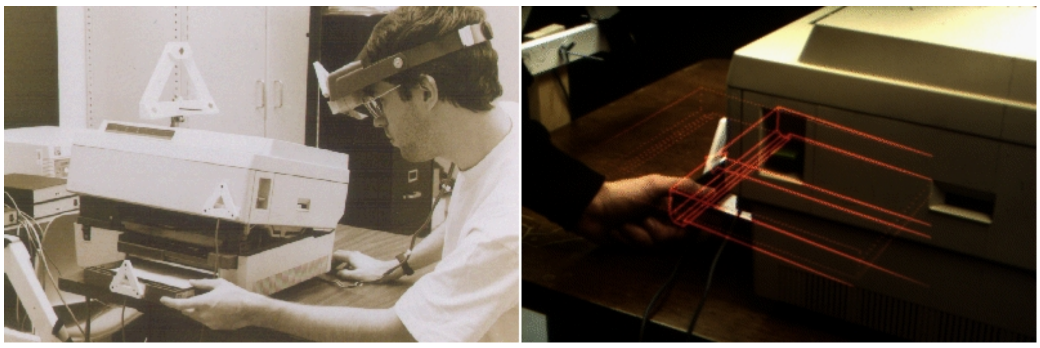


Figure 2-3: The printer maintenance program using AR [5]

In 1999 augmented reality stopped being just new field for scientists, because until then public did not have a clue, that it existed. In this year everything changed, because Hirokazu Kata of Nara Institute of Science and Technology released the ARToolKit to the open source community. This toolkit allowed the users to place virtual 3D objects to the real world video captures and interact with them. A year later the first outdoor mobile AR game named ARQuake was released. This game placed the virtual monsters to the real world. It was also revolutionary, because it did not need any joystick or other controls. On the other hand it needed a lot of hardware, which was not common those days like gyroscope and GPS, so the player needed a backpack and head mounted display to enjoy his game.

The first smartphone applications, which used augmented reality, began to appear in 2008, together with new era of mobile devices, which had everything that was needed to provide a good AR experience. One of the first AR applications on the market was the Wikitude, which provided the user information about nearby point of interest.

The real history of massive use of augmented reality is quite short, so we can expect it to expand in the future even more with more and more people using smartphones.

2.2 Application

Application area of augmented reality is quite vast. In connection with mobile devices, its possibilities are endless. Some of them are summed in this chapter.

Building industry

First area of use is building industry. Augmented reality provides us possibility of placing 3D model of structure in real environment. This can be done for example by pointing a camera to defined spot and display the model on it, so the user is able to observe it from different angles. This can help our imagination of how the structure looks like in reality, so there is a possibility for changes in the schematics, if something looks better on the paper, than it really does in the exact environment.

Games

Augmented reality can take the possibilities of games to brand new level. First huge advantage is, that user can use his real world to provide the area for his game, which provides him absolutely unique feeling about his play. Other advantage is there for game developers, because in some games, they have to put huge effort in making games real, which demand high computing power and developer’s skills. Using real environment make the game undoubtedly as real as possible. Therefore the gaming developers do not have to pay their attention to rendering virtual environment and they can focus to other aspects of game creation like realistic physics or interesting plot. Example of AR fantasy game can be seen in Figure 2-4.

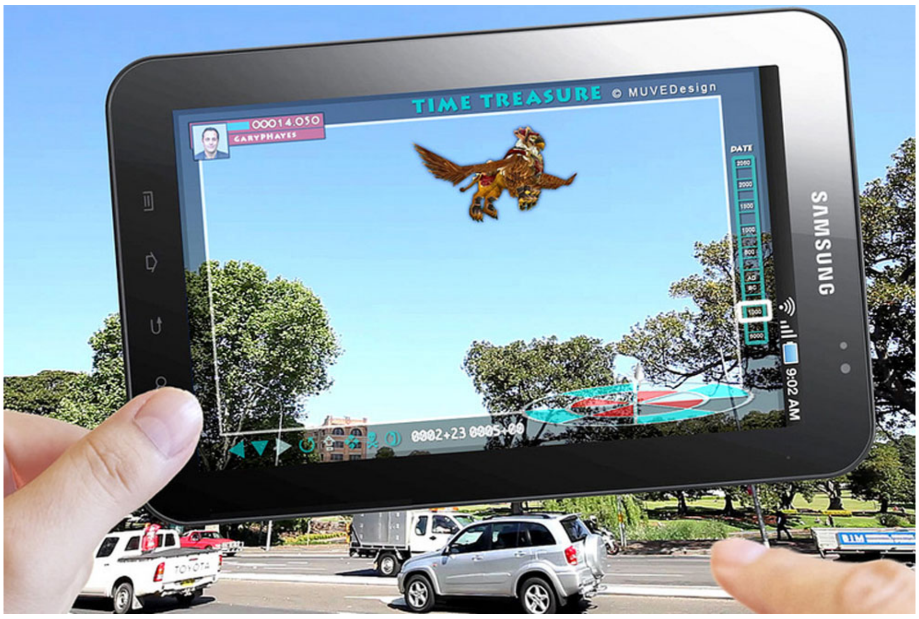


Figure 2-4: Example of augmented reality game [6]

Design

This area is closely connected to the building industry use mentioned above. User can for example find out, how would some particular painting look on a wall in his living room or he can decide better in the question of whether some particular wall paint red, white or put a wallpaper on. Obviously it can be used for designing other objects as well. Painters could use it for example for finding out, whether the idea for a piece of painting in their mind, would not ruin the rest of the canvas, they have already created.

Additional information

For this area augmented reality is used quite a lot these days for example in broadcasting some sport event in television. We can for example project information about football teams directly to field or giving information about swimmer directly to his lane in the swimming pool.

Another example of this area is placing some dynamic content into static environment, like when user points a camera of his smartphone on a newspaper page he can see video instead of static picture. In Figure 2-1 there is an example of city guide developed by Nokia, which provides information about services nearby the position of the user.

There are also some attempts to develop an augmented reality dictionary, which would be able to detect some text, then recognize the language and translate it to the language, the user of the application can understand. For example Frommer's Guidebook Company is taking part in this challenge.

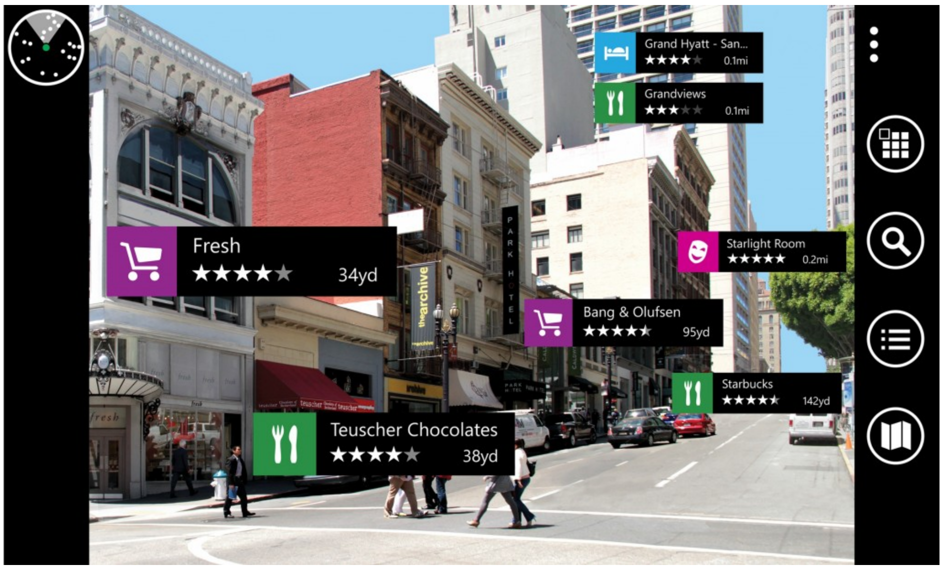


Figure 2-5: Nokia city lens [7]

Simulation

Augmented reality can provide us also to simulate various events in real environment, so we can better understand the results.

2.3 Use on smartphones

Smartphones are definitely kind of device, which make augmented reality much more usable for public. As was mentioned in Chapter 2.1, before the era of smartphones augmented reality could exist only with some complex and expensive hardware and smartphones brought ergonomics and availability for mass use into the augmented world. Only thing that the users have to do is to carry a smartphone with them, which they do anyway.

Modern smartphones have a lot of possibilities that AR benefits from. First and the most important is the camera of course. Without it real-time AR on smartphone would not be really possible. Another very important characteristic is powerful graphics, which is able to display even complex 3D objects in real-time. There are other useful possibilities of smartphones as well, like GPS chip, accelerometer, compass and permanent Internet connection. In the end, also operating phone by touching the screen helps the user to control his AR application.

3 CAVE technology

An Acronym CAVE stands for Cave Automatic Virtual Environment and as can be seen from its name, it is a virtual reality environment, where projectors are directed to the walls of a room-sized cube. The walls are made up of rear-projecting screens and also the floor is made of down-projection screen [8]. The user of CAVE usually wears special glasses to see 3D graphics generated by the CAVE. People there can see the objects floating in the air, can walk around them and this helps them to get a proper imagination of how the objects look like in the reality. This is made possible by electromagnetic sensors. Because of this, the frame of the CAVE is made of non-magnetic stainless steel to prevent interferences. The video is adjusted accordingly to the user moves by projectors and mirrors outside the CAVE. The computers control not only the displaying technology but also the audio aspect via multiple speakers placed into CAVE, providing 3D sound.

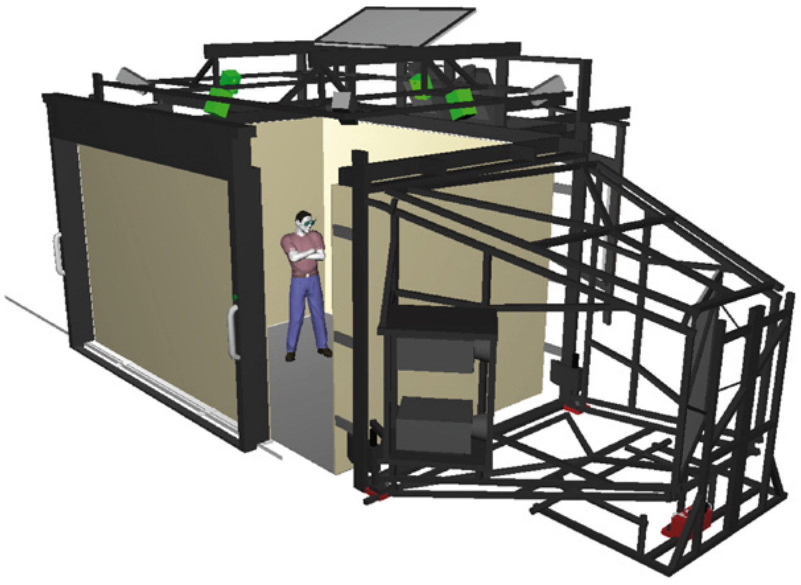


Figure 3-1: The CAVE technology visualization [9]

3.1 History

The first CAVE was developed in the Electronic Visualization Laboratory at University of Illinois in Chicago and was demonstrated in 1992 [8]. It was a response to the challenge to create and display a one-to-many visualization tool that utilized large projection screens. The CAVE answered that challenge and became the third major physical form of immersive virtual reality after goggles'n'gloves and vehicle simulators. Since then, CAVE is a registered trademark of the University of Illinois. Commercial systems based on the concept of the CAVE are available form a handful of manufactures.

The real CAVEs are now all over the world, mostly at the universities. For example there is one at the Czech Technical University in Prague. Other examples in Europe are Association Promouvoir la Realite Virtuelle in Clermont-Ferrand in France, University of Groningen in Germany and also the institution where this thesis application will be developed, the University of Applied Sciences in Oulu in Finland, which is in Figure 3-2.

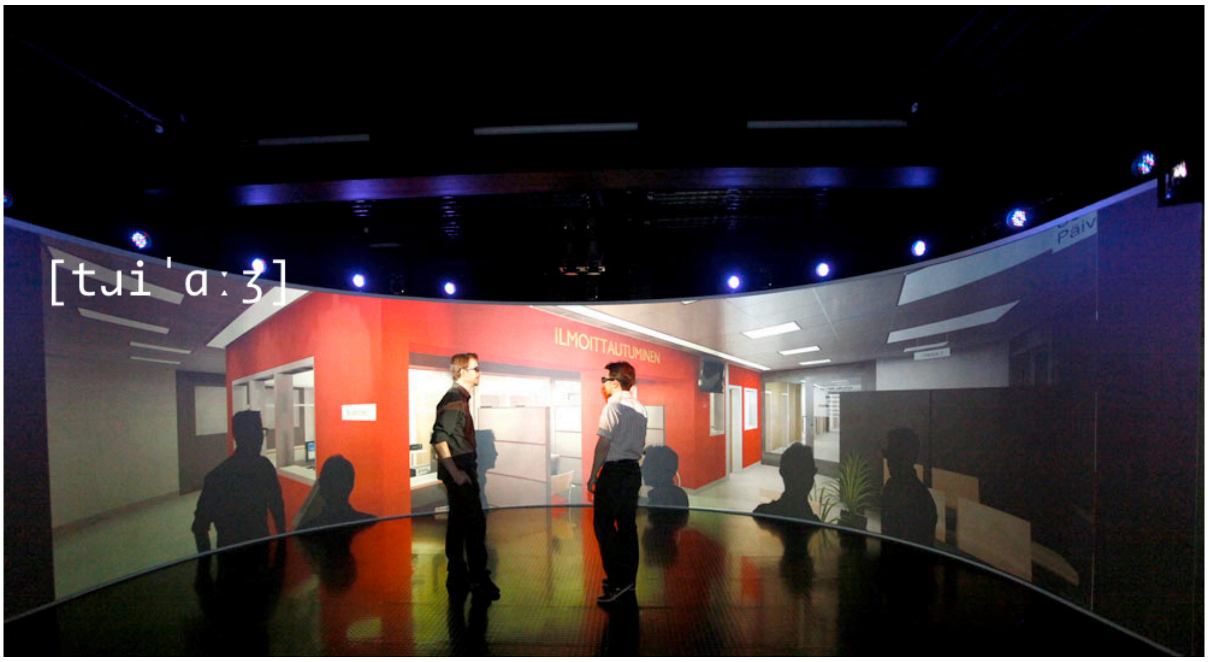


Figure 3-2: CAVE at Oulu University of Applied Sciences

3.2 Principle

The projector positioned outside CAVE and controlled by the real-time user's movements inside CAVE creates the visual display. The computers rapidly generate a pair of images, one for each of the user's eyes based on the user's movements. The glasses, that user wears, are synchronized with the projectors so that each eye sees the correct image. Mirrors are used to reduce the distance required from the projectors to the screens. Because of high computing power demands, clusters of desktop PC's are often used.

There are several libraries and software designed specially for CAVE and there are also several techniques for rendering the scene. Example of such library is CAVElib, which was developed for the first CAVE in Illinois. It was commercialized in 1996 and it provides low-level API for various areas of CAVE related issues.

3.3 CAVE at Oulu University of Applied Sciences

The size of this particular CAVE is 8 m x 8 m x 3.5 m (W x L x H) [10]. Unlike other CAVEs there is not the typical rectangular wall; curved optical projection screen with a front-projection stereo is used instead. Also the floor is front-projected. This was chosen in order to reduce the space needed for CAVE and to achieve sharper images. Curved optical projection screen provides optimal viewing angle and better contrast ratio. The size of the screen is 13 m x 2.5 m and it provides an opening angle of 180º with 3.5 m radius. This screen requires images to be warped before being projected, so special video software is used for the correction. There are also special devices for external control such as lightning, fog etc. 7.1 surround sound system is used for audio. The installation of the CAVE begun in April 2012 and it was finished and opened in September 2012, but it is still on a test run.

Three stereo projectors provide the projection on a curved screen. The schematics can be seen in Figure 2-1. We can see that on some parts of the screen the projection from two different projectors intersects to provide more natural view. This requires precise merging of the two views from two projectors. Also the view from the projector must fade out on the place, where is view projected from two projectors, causing easy recognizable area, where there is a bit darker stripe in the projection. Disadvantage of using front projection instead of rear one is, that if the viewer stands close to the projection, he creates the shadow, because he stands between the projector and the screen and the result of this is violation of the view, which is with use of 3D quite disturbing.

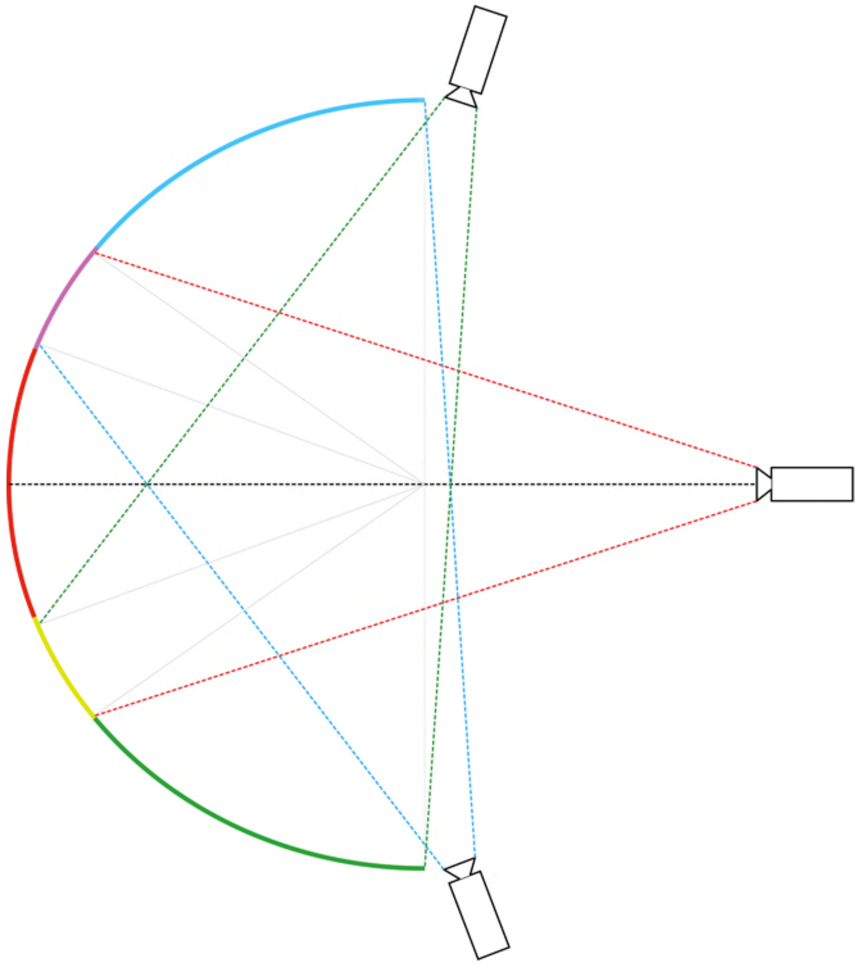


Figure 3-3: Schematics of the OAMK CAVE

For 3D experience in the CAVE is used polarized 3D stereoscopic projection. Every one of the three projectors in Figure 3-3 is in fact a 3D projector, which is composed of two projectors with polarization filters in front of them. Viewing technology is basically the same as with today's 3D televisions, because it uses passive 3D projection. The final view is composed of two partial views, which are mutually shifted for 65 milometers. One of the views uses horizontal ripple and the second one vertical ripple. This means, that the user of CAVE wears special glasses which divides the view back to partial views, one for each eye, so in fact there is one projector for left eye and one projector for right eye. This causes the 3D experience for the viewer. Advantage of this solution is, that it is quite straightforward and easy, but the disadvantage is, that 3D view without glasses is very unnatural. Of course the 3D view can be switched of, so the projection is only 2D and then the viewer does not need the glasses. There are two ways how to create 2D effect. One is very straightforward; close the shutter on one single projector on all the stereoscopic projectors. The second option is to set the shift between the views to 0 millimetres, so both single projectors project to the same place and the views are overlapping. This solution is better, because the image is brighter because there is light from both of the projectors. In the other hand, there can be slight difference between the two views, so it is not exactly at the same spot.

4 Augmented reality frameworks

With growing popularity of augmented reality come new possibilities of mobile application development and work is getting easier with help of graphic libraries and even special augmented reality frameworks. In this chapter we will go briefly through some of them and we will sum up their basic attributes.

4.1 Demands on AR framework

2D/3D objects display

The absolutely essential demand on AR frameworks is displaying objects on a screen. The objects can be in 2D, but 3D objects provide the user much better experience. The definition of those objects should be as easy as possible, but it is quite a common issue, because often must the objects be created as a set of triangles, because the frameworks often use OpenGL as a rendering framework, which needs this representation.

Embedded recognizer

Because the AR application often displays the virtual objects on a defined spots in the reality specified by marker or even better, markerless marker (Chapters 5.1 and 5.2).

User gestures interaction

In some kind of AR application user needs to interact with the displayed virtual object, so he can make it bigger, rotate it and so on. Also control of some real or virtual buttons is useful.

Supported platforms

It is advantage when a framework exists in more versions for different platforms, because it makes transforming application between various platforms easier. We should focus on two leading platforms in this segment, Android and iOS. According to [11], these platforms together have 84% majority in all new smartphones sold in first quartile in 2012. Windows Phone 8 was brought out recently and it has possibility to change the balance in this segment, but we have to wait to see how much. Because of the fact, that the aim of this thesis is augmented reality on iOS devices, we will take closer look especially on the frameworks available for iOS.

4.2 Quartz2D

Quartz2D is a basic graphics library available in the iOS SDK in Core Graphics framework. Its key attributes are:

- Support of 2D environment only.
- Embedded into iOS SDK, so works only on iOS devices.
- Only basic methods for drawing simple objects.
- No real support for augmented reality; developer must handle everything.

- Bad encapsulation, difficult change of the drawn shape.
- Easy to use.

4.3 OpenGL ES

OpenGL ES is a subset of graphic framework OpenGL, which is used on personal computers. Shortcut ES in the name stands for Embedded System, which is area where smartphones and tablets belong. Huge advantage of this framework is that it is available for iOS and Android devices, but it is not yet available for Windows Phone. This is caused by the fact that Windows Phone does not allow unmanaged code to run on this system.

OpenGL ES is much more powerful than Quartz2D, it is capable of creating 3D graphics, animations, different lightning models and so much more. On the other hand it is very low level, so it takes a large amount of time to create basic application and it is not as easy to learn as Quartz2D. All objects in OpenGL ES both 2D and 3D are specified as a set of triangles, which exist in defined coordinate system, so it is quite difficult to create the content and developer can get lost easily. Especially the beginners can find OpenGL ES too hard to learn and confusing. There are so many attributes, that can be set, but average developer does not need to use all of them and they only slow his work down.

It is now clear that OpenGL ES brings more possibilities to augmented reality applications than Quartz2D, mainly because it can render much more sophisticated objects in 3D, use light models and so on. Main importance of this framework is that other high level specialized augmented reality frameworks use OpenGL ES as a rendering engine. So if the platform does not support OpenGL ES, it is hard to find some proper augmented reality framework for it. This is the case of Windows Phone.

4.4 Cocos2D

Cocos2D is primarily an open source gaming framework [2], which works in 2D, as it is clear from its name. Originally it was written in Python, but later it was ported to other languages as well, including Objective-C. Cocos2D uses OpenGL ES because of optimized data structures. It is well connected to the iOS SDK so it can work with camera, accelerometer, Cocoa touch and other features easily. Lot of different things such as basic menus, physics engine, sound support, scene transitions, special effects and much more. Cocos2D provide special template integrated directly to the XCode to make application development easier. Because it does not aim primarily to augmented reality, it does not have marker recognition included, so it has to be done by using third party solution or by implementing own recognizer. Cocos2D can be upgraded with all its features to a 3D environment, using the Cocos3D extension.

4.5 String SDK

The developer of String SDK [13] is String Labs Limited and they claim, that it is the fastest and easiest framework to use SDK for iOS with very low HW requirements. It is very powerful SDK, it provides tracking multiple markers, video capture, and also cooperation with Cocoa Touch, so the user can interact with the objects displayed. The company also claims, that the user of a String SDK does not even have to be a developer, because he is able to create applications without programming. User can also create his own image markers.

All these advantages come with quite a high price. Only the demo version is for free and it cannot be used for Apple App Store release, it can track only one marker and it cannot be interacted by gestures. The cheapest version with the same limitations is free, but it is only for demonstration purposes. Version which can be used for releasing applications is 485 € per application per year. The most expensive version without String advertisements and, unlimited marker count and touch interaction costs incredible 6795 € per application per year. This policy makes the String SDK usable only for big application developer companies with very popular and expensive applications in Apple App Store.

String SDK is not yet available for Android platform, but the company claims, that they are currently porting their SDK to it and it should be out very soon.

4.6 Vuforia Augmented Reality SDK

Vuforia is an augmented reality SDK [14], working in real-time developed by Qualcomm Incorporated. Qualcomm is a global telecommunications equipment company. It also develops its own semiconductors. This system is able to tightly couple 3D virtual objects with real objects recognized in camera view. The virtual objects placed into the view are rendered by OpenGL ES, so support of OpenGL ES is required for Vuforia to run on the device. As a result Vuforia is not available for Windows Phone. Nevertheless there are Vuforia SDK versions for either iOS or Android.

The platform consists of the Vuforia SDK and Target Management System hosted on Internet portal http://ar.qualcomm.at. This portal is used for managing the trackable images. Developer simply uploads the picture, he wants to be detected by his application, and then he downloads target resources which then includes in his application. In case of iOS it is static library (`libQCAR.a`) and on Android it is shared object (`libQCAR.so`).

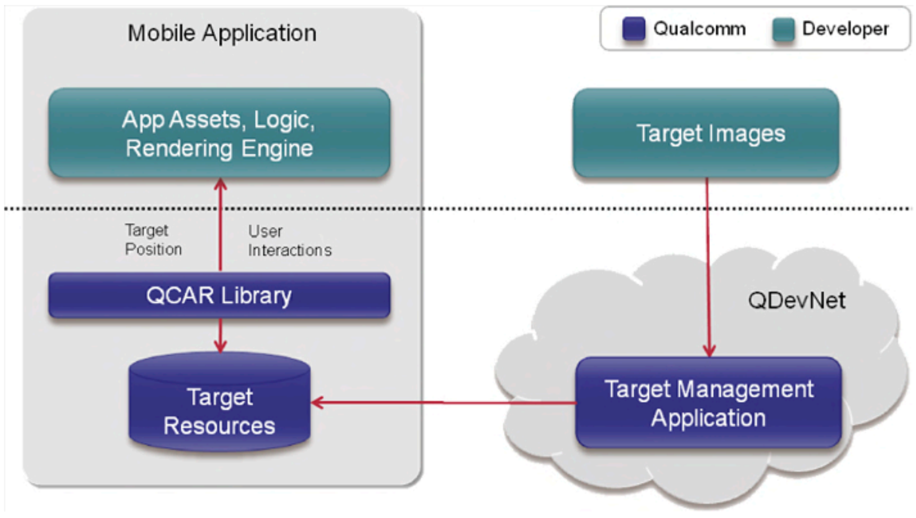


Figure 4-1: Vuforia SDK engine [14]

Vuforia SDK Architecture

Augmented reality application, which is based on Vuforia SDK, has these core components:

- **Camera** – passes the live preview to the tracker. Must be initialized by the developer.
- **Image Converter** – converts between the camera format and format suitable for OpenGL ES and for tracker. Also provides different resolutions of image available in the converted frame stack.
- **Tracker** – contains different computer vision algorithms that detect and track real world objects in the camera view. Results are stored in a state object that can be accessed from the application code.
- **Video background renderer** – renders the camera image stored in the state object. Rendering process is optimized for specific devices.
- **Application code** – in this part all of the above components must be initialized. For each processed frame, the state object is updated and the render method is called. Developer must query the state object for newly detected targets, then update the application logic with the new input data and render the AR graphics overlay.
- **Target resources** – files created by the Target Management System, which contain information that allows the developer to configure trackable features. These assets are compiled into the application installer package.

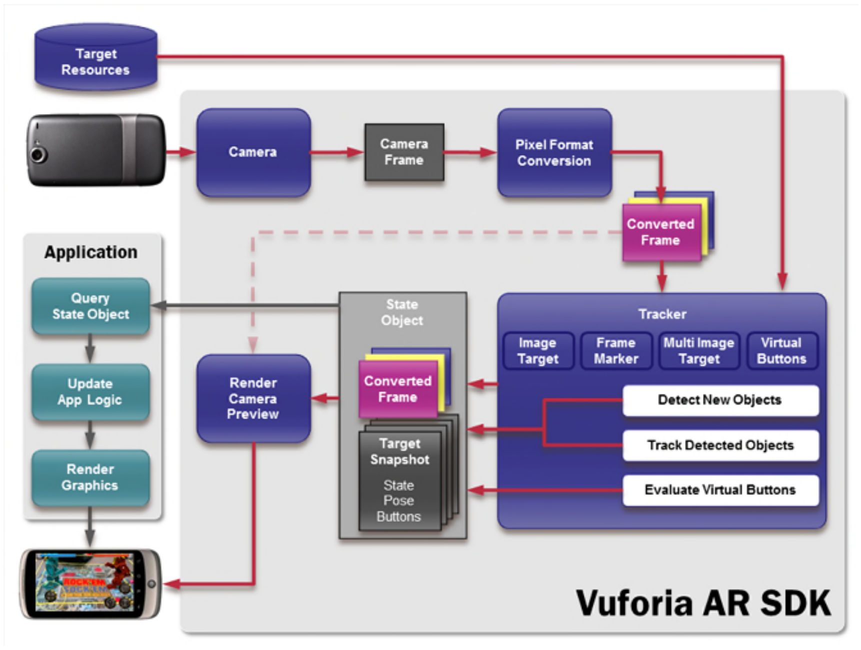


Figure 4-2: Vuforia SDK architecture [13]

Vuforia SDK sample applications

Together when downloading Vuforia SDK the developer gets a several sample applications, which he can use to learn working with Vuforia SDK. In this chapter two main of them will be shown to provide better imagination of what Vuforia SDK allow the user to do.

Background texture access

This basic sample shows us the possibility of displaying a 3D object (teapot in this case) onto a picture, which was proceeded by the Target Management System.

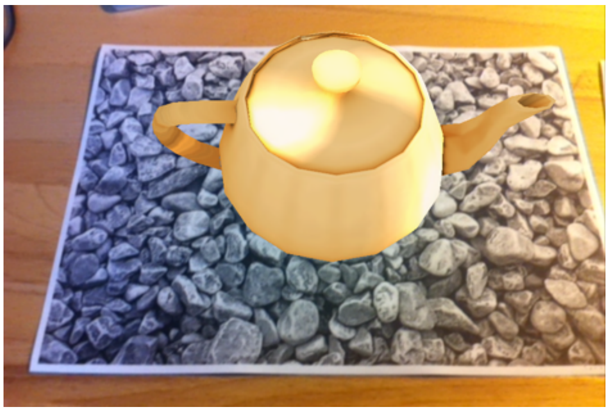


Figure 4-3: Vuforia SDK texture access sample

Frame markers

This sample has the possibility to detect different type of markers and display different type of objects. Result can be seen in Figure 4-4.

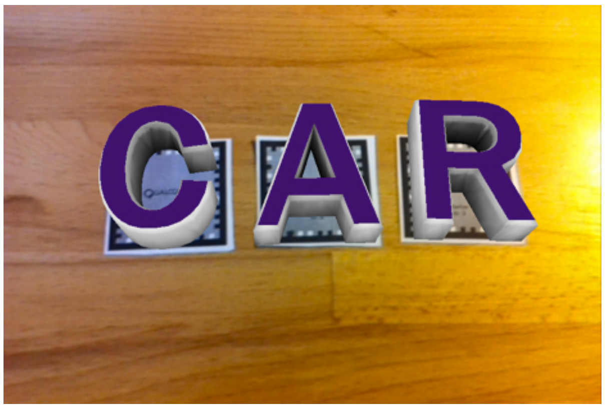


Figure 4-4: Vuforia SDK frame markers sample

4.7 Conclusion on AR frameworks

There are several possibilities of graphic frameworks, which can be used for an application for mobile platforms. Some of them are more specialized for augmented reality and some of them less. The list above does not cover all the existing frameworks; it is more for demonstration of what is possible nowadays. Some of the other frameworks, like Popcode [12], started as ambitious free and wide usable projects, but that changed the policy and become private.

After studying all of the above AR frameworks, it seems, that the best one is the Vuforia SDK, because it is high level, it has embedded marker recognition and is free. String SDK is also very complex and provides even more possibilities than Vuforia SDK, but its brutal licencing policy does not make it available for normal use. Other frameworks do not provide marker recognition or other important features, so it would mean a lot of extra effort to develop an application using them.

5 Markers and object recognition

Marker recognition is an important aspect in creating augmented reality application, because there is almost always demand for recognizing specified objects in the real world and augmenting it. There are two possibilities of recognizable objects and according to that there are two types of augmented reality.

5.1 Marker augmented reality

First possibility of object recognition is to use marker objects. Marker objects are usually 2D objects, which are specially created for purposes of the recognition. Probably the most used marker for the job is QR code. Nevertheless markers can be less sophisticated that QR code and usually some picture with some lines and sharp corners is used. It is easy to recognize, that the object is a marker because it does not naturally belong to the environment. Example of simple marker augmented reality application can be seen in Figure 5-1.

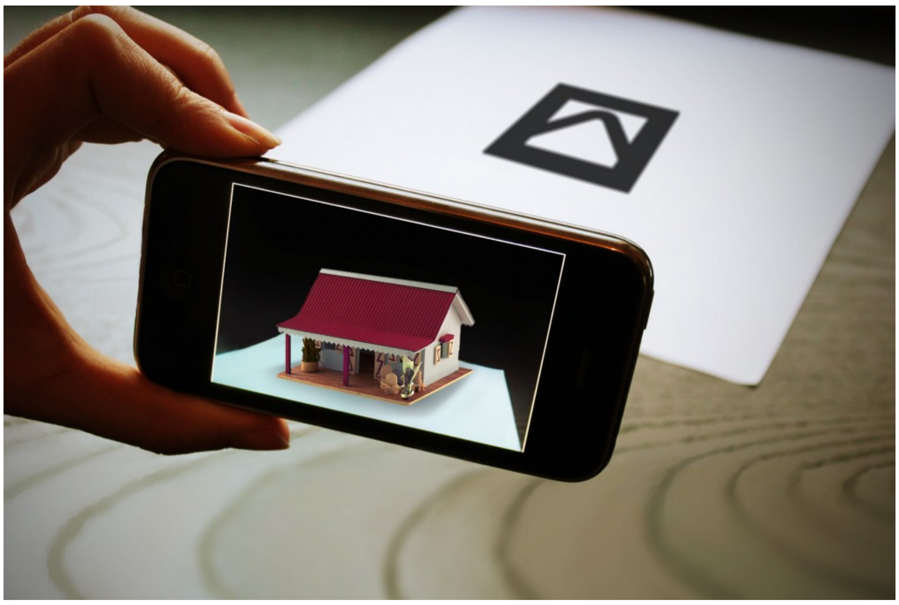


Figure 5-1: Example of marker augmented reality [15]

Advantage of using this kind of markers is that their detection is quite straight forward, because the markers are designed specially for recognizing. This means, that the markers have high contrast between different parts, they are usually black on white background and they have a lot of lines and corners making detection of these markers much easier, so it is quicker, more precise and it has lower requirement for the computing power of the device.

On the other hand these kinds of markers do not provide as good experience of augmented reality, because it is obvious, that some object is a marker. Therefore using this type of markers violates the surrounding environment, which is bad both for using the augmented reality, but mainly for using the real space without augmentation in everyday life.

5.2 Markerless augmented reality

Markerless augmented reality provides the possibility to detect objects, which naturally belongs to the environment. This unfortunately does not mean that we are able to recognize any object; the recognizable object must contain some special features. Often these objects are specially added to the environment, but without it being obvious and violating. It could be for example a picture on the wall. Sometimes also objects, which are already a part of the environment, can be used. For example specified part of a floor or carpet can be used, if it contains enough features. Very suitable natural part of environment is in Figure 5-2.

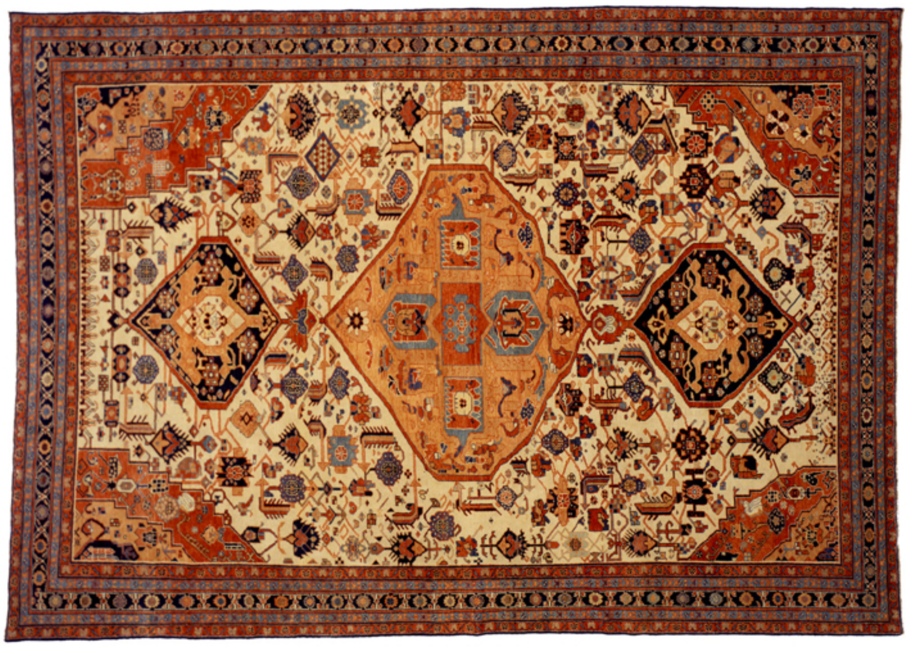


Figure 5-2: Example of natural object suitable for detection

There are several ways, how to recognize these objects, but the conditions, that the object should fulfil are basically the same:

- Good contrast, dark parts and bright parts, difference between colors. For example object with just white a light gray color is not suitable.
- Lot of features like lines, edges, corners.
- The pattern of the object should be unique and should not be repeated on other spot. For example chessboard styled floor is not suitable, because the pattern is periodic and it is impossible to recognize which part of the floor should be detected and which not. Using part of computer keyboard, as a marker is also not a good idea, even though the signs on the buttons are different, because the basic pattern is still the same can also confuse recognizers.

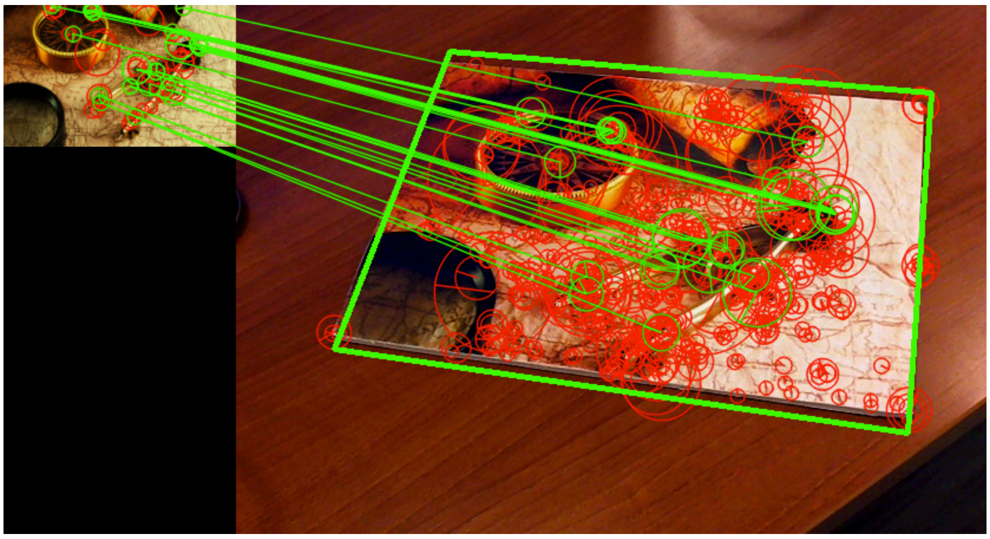


Figure 5-3: Recognizing markerless object [15]

5.3 Markers in Vuforia SDK

Vuforia SDK has implemented feature for recognizing markerless objects. Vuforia SDK uses for recognition corners, so for example circles and other simple round objects are absolutely unable to be recognized. The more features there are in the view, the more precise and quicker the recognition process is. Other important feature of good marker is balanced distribution of the features. Views with all the corners in small part, leaving the rest empty, are not very suitable. If the view is like this, it is better to choose as the marker just the part of it where all the features are. Third aspect of a well recognizable view is good local contrast. This is often difficult to recognize with naked eye, but in general it is not good, when the picture is blurred or high-compressed or has a lot of organic shapes with round details. The last thing about Vuforia SDK suitable objects is to avoid repetitive patterns because of the reason from previous chapter. In this case neither rotating the marker, nor changing scale would help.

Examples of views that can be used as markers

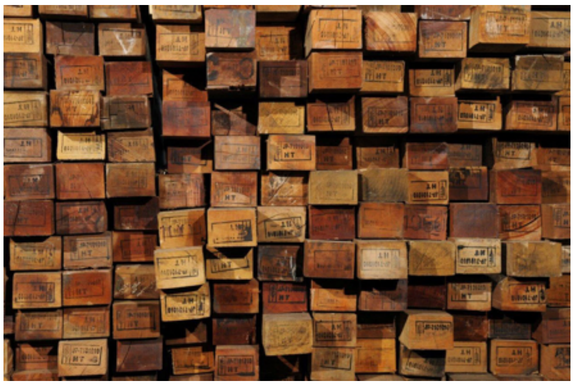


Figure 5-4: Good marker - wood

First example of suitable marker is in Figure 5-4. As we can see, there are lot of edges and corners and also the contrast is suitable. On the other hand the view in Figure 5-5 is poor for recognizing, because there are not enough corners that can be recognized and the edges between different pieces of wood are all the same, so in the view of the recognizer, there is a repetitive pattern. The addition of bad contrast makes this very inappropriate for being used as a marker.

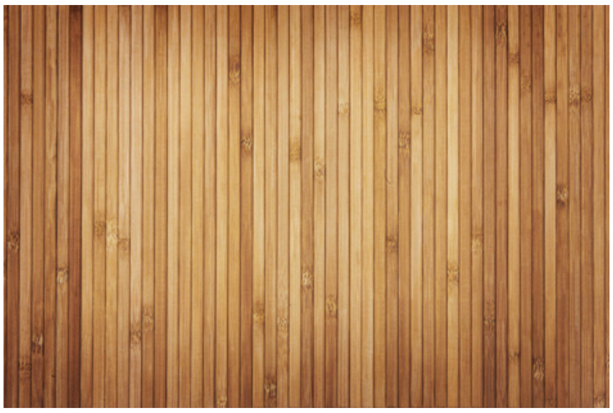


Figure 5-5: Bad marker – wood

Another example of great marker to use with Vuforia SDK is various psychedelic views as the one in Figure 5-6. There are a lot of edges and corners, contrast is good and the pattern is not repetitive.

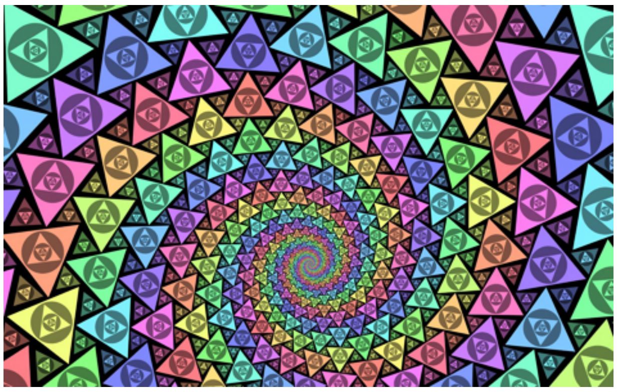


Figure 5-6: Good marker - psychedelic view

The difference between suitable and unsuitable markers can be also explained on classic paintings. In general painting can be quite good markers, but not all of them. For example cubism is good style and cubistic paintings can be used as markers as can be seen in Figure 5-7. It is a famous painting “Guernica” by Pablo Picasso; it contains a lot of features used for recognition, because it has a lot of sharp edges and corners. Also the contrast is great.

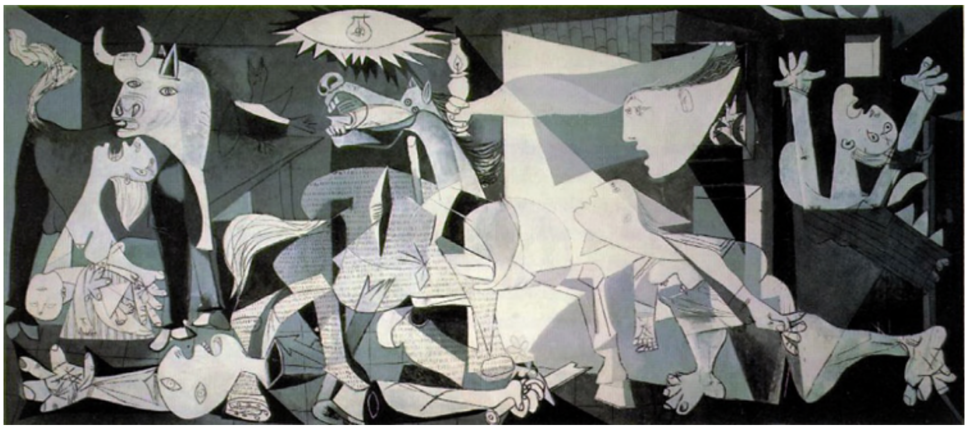


Figure 5-7: “Guernica” by Pablo Picasso (1937) [16]

On the other hand some impressionistic paintings are not so good for recognition. An example can be seen in Figure 5-8, where is a painting “Saint-Georges majeur au crépuscule” by Claude Monet. It is not very suitable to be used as a marker, because it does not have any really sharp edges so the recognizer views it blurry and the contrast is also poor.

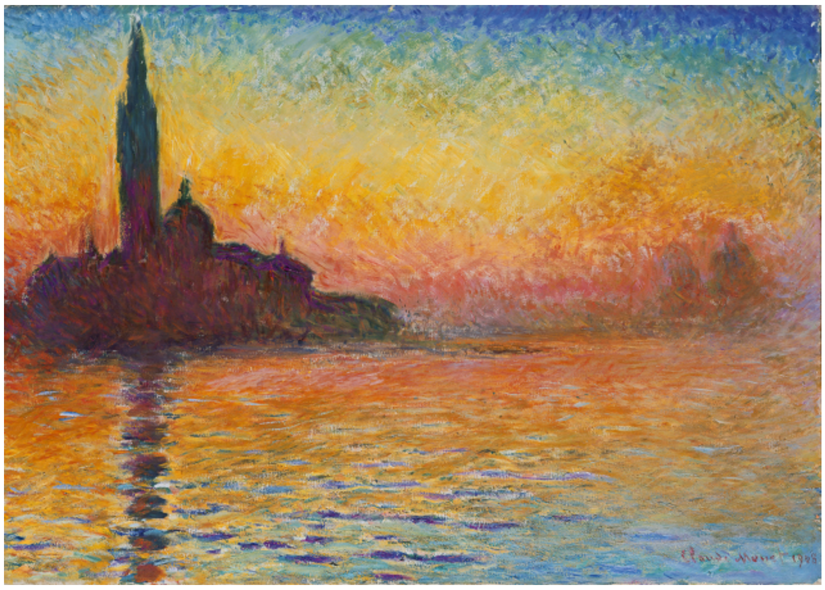


Figure 5-8: “Saint-Georges majeur au crépuscule” by Claude Monet (1912) [17]

Limitation of Vuforia SDK marker recognition and also other engines is, that the markers have to be in 2D. It would be possible to fool the recognizer and use 3D environment as a marker, but since the recognizer is not designed for this, it would work just from one specific angle. If the angle of the camera changes, also the position of the objects in a view changes and therefore the recognizing algorithm would not be able to detect the features in the view and recognize the correct marker. For application in the CAVE environment described in Chapter 3.3, this can cause some problems, because the projection screen is curved and not flat, so it is not really 3D.

Important thing is also, that the marker has to be big enough. The size depends on the distance we want for the marker to be recognized. The further from the marker the camera is, the bigger the marker must be. Fortunately the recognizer can deal with quite wide viewing angle of the marker, so the user does not have to see the marker straight in front of him.

6 Markers in CAVE

Situation with markers in CAVE described in Chapter 3.3 is much more difficult than marker recognition in the real world. This is caused by several factors:

- Projection in OAMK CAVE is not to a flat plane, because the screen is curved. This causes distortion of the image on every place except from the point on which is the projector directly pointed. This can cause problems in both, 2D and 3D projection. The problem can be seen in Figure 6-1, where for example Screen 4 and Screen 5 are almost rectangular. On the other hand, Screen 1 and Screen 8 are more like rhomboids. Although the recognizer can deal with wider angles, the addition of curved surface can cause a big problem for the recognizer.

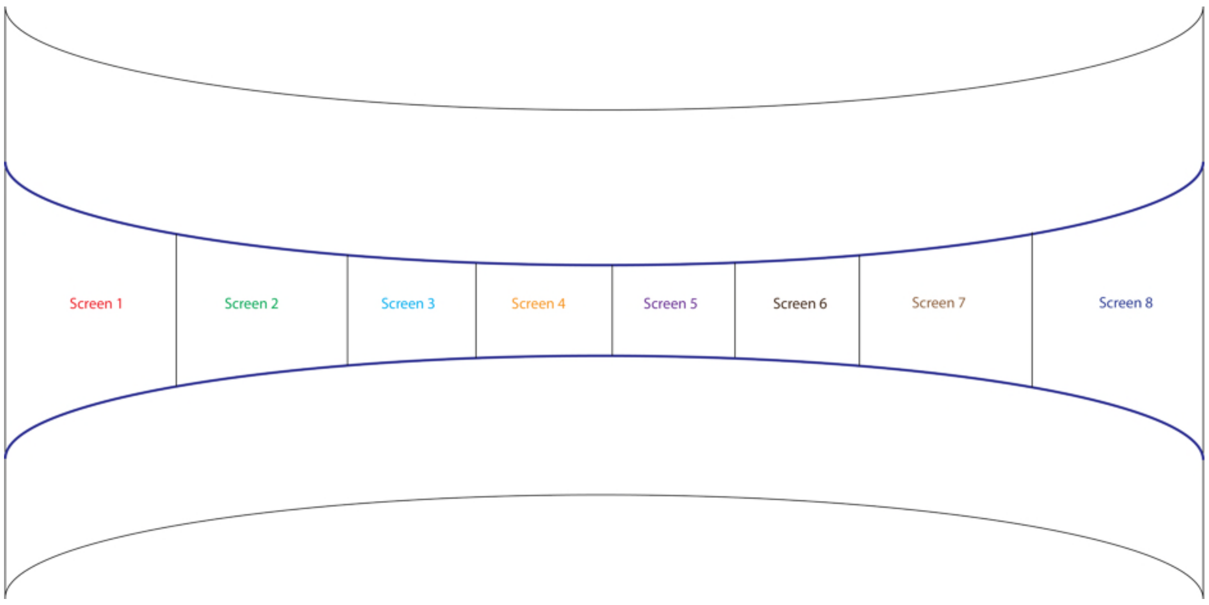


Figure 6-1: Difference in shape of screens in different places on the screen

- In 3D projection there are two shifted views (as is described in Chapter 3.3). Without polarization filters or glasses, the view seems blurry. As is mentioned in Chapter 5.3, this causes a problem for Vuforia SDK recognizer.
- In 3D projection, the shift between the two views varies according to the distance of the projected object. When the object is far in the scene, the absolute distance between the partial views is smaller then when the object is closer. This means, that loading 3D view in one distance does not mean, that it will work in other distances.
- The accuracy of recognizing varies with the size of the object we are trying to recognize. If the user is far from the screen, recognition is worse than when he is close. On the other hand as is clear from Chapter 3.3, when the user is close to the screen, he casts shadows which violate the view. Fortunately the solution of this issue is quite straight forward, because we can just use markers, which are big enough. On the other hand we cannot expect proper

recognition, when the marker is far away in the virtual environment, which is projected in the CAVE.

6.1 First series of experiments

Experiments description

CAVE used in the experiment: CAVE at OAMK (description in Chapter 3.3)

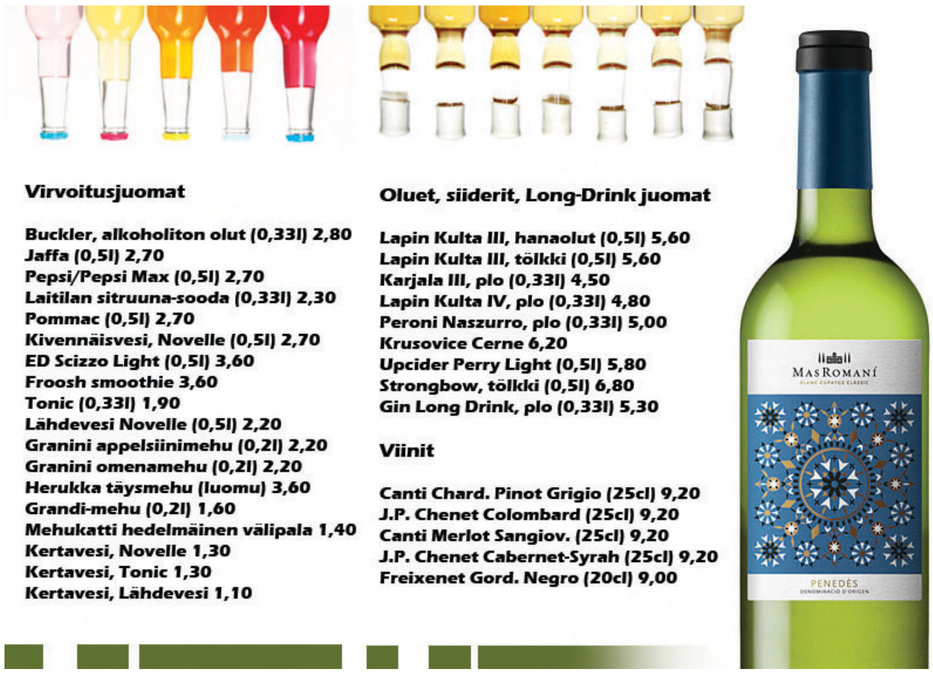
Mobile device: iPhone 4 with iOS 6.1

Application for testing: Vuforia SDK sample application "Used defined targets", which can add markers for recognizing in runtime. The marker is loaded to an application, when it is placed right in the centre of the screen. This means, that it is taken in the time, when the distortion is the least and it is the closest to the square.

Environment projected into CAVE: Model of train carriage, which was created for Finnish railway company VR Group (VR-Yhtymä Oy)

Parts of environments used as markers: Three bitmaps, which are placed in the train model will be used:

1. Beverage list, which is situated in the dining car and also on other places in the carriage. This is in Figure 6-2. We can see, that this marker can be used with Vuforia SDK very well, because it has a lot of non-repetitive features that can be detected.


Virvoitusjuomat

Buckler, alkoholiton olut (0,33l) 2,80
Jaffa (0,5l) 2,70
Pepsi/Pepsi Max (0,5l) 2,70
Laitilan sitruuna-sooda (0,33l) 2,30
Pommac (0,5l) 2,70
Kivennäisvesi, Novelle (0,5l) 2,70
ED Scizzo Light (0,5l) 3,60
Froosh smoothie 3,60
Tonic (0,33l) 1,90
Lähdevesi Novelle (0,5l) 2,20
Granini appelsiinimehu (0,2l) 2,20
Granini omenamehu (0,2l) 2,20
Herukka täysmehu (luomu) 3,60
Grandi-mehu (0,2l) 1,60
Mehukatti hedelmäinen välipala 1,40
Kertavesi, Novelle 1,30
Kertavesi, Tonic 1,30
Kertavesi, Lähdevesi 1,10

Oluet, siiderit, Long-Drink juomat

Lapin Kulta III, hanaolut (0,5l) 5,60
Lapin Kulta III, tölkki (0,5l) 5,60
Karjala III, plo (0,33l) 4,50
Lapin Kulta IV, plo (0,33l) 4,80
Peroni Naszurro, plo (0,33l) 5,00
Krusovice Cerne 6,20
Upcider Perry Light (0,5l) 5,80
Strongbow, tölkki (0,5l) 6,80
Gin Long Drink, plo (0,33l) 5,30

Viinit

Canti Chard. Pinot Grigio (25cl) 9,20
J.P. Chenet Colombard (25cl) 9,20
Canti Merlot Sangiov. (25cl) 9,20
J.P. Chenet Cabernet-Syrah (25cl) 9,20
Freixenet Gord. Negro (20cl) 9,00

Figure 6-2: Beverage list marker (adopted from VR-Yhtymä Oy)

2. Rail map – marker in Figure 6-3 has fewer features than the previous, but in normal conditions it is still good enough. This marker was selected to experiment also with not that good marker inside CAVE.

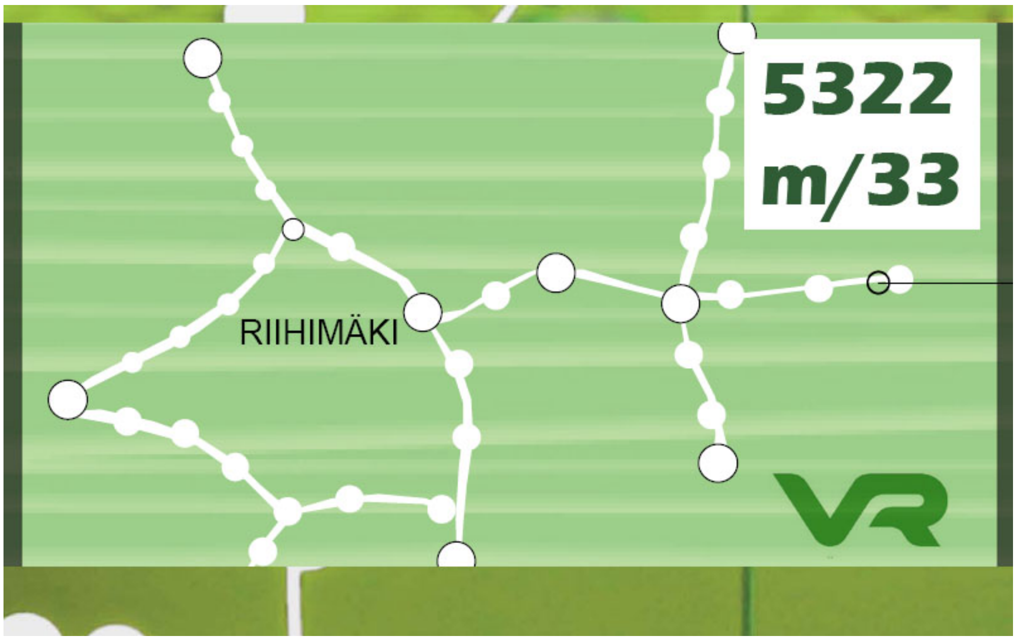


Figure 6-3: Rail map marker (adopted from VR-Yhtymä Oy)

3. Food list – the last marker that we are going to use in the experiments is in Figure 6-4. It is a food list with miniatures of the food. It is a bit similar to the first marker, but this one has more pictures included and less text. Apart from that it has sufficient contrast and a lot of features.

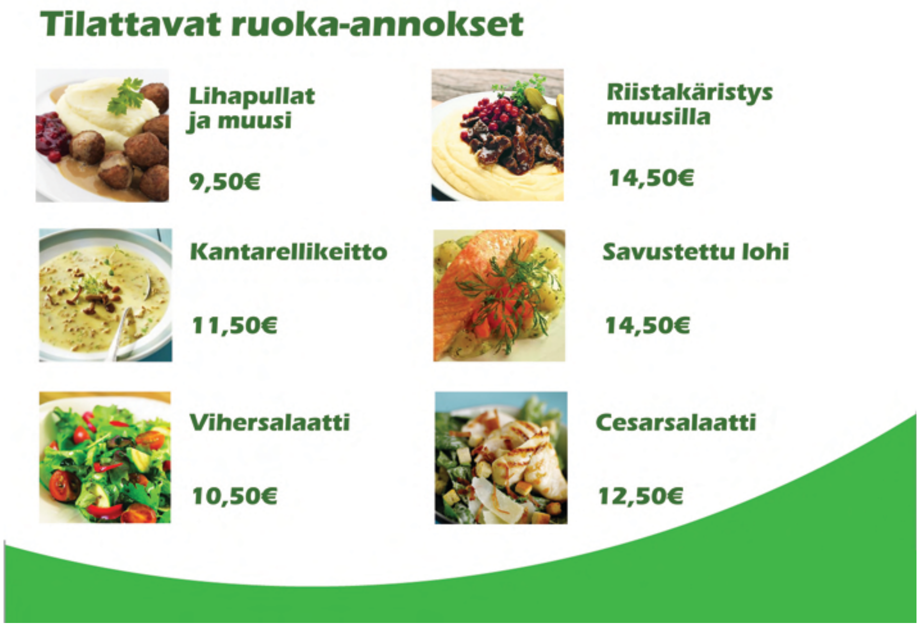


Figure 6-4: Food list marker (adopted from VR-Yhtymä Oy)

Variants of experiments: Recognition will be tested in three ways:

1. CAVE with 2D projection – 3D projection will be shut down. This means, that the user of the CAVE does not need the glasses, but on the other hand the experience will be worse. Advantage is that we are expecting much more precise recognition of the markers.

2. CAVE with 3D projection using polarization glass – in this case 3D projection will be turned on, but we will use polarization glass, so the appearance on the camera is basically the same or at least very similar to 2D recognition. Advantage of this solution is that recognition should work as good as with 2D projection. On the other hand holding glass in front of a device camera can be limiting and the violation of users 3D experience is significant.
3. CAVE in normal use – in this case CAVE will be working normally in 3D and the user will use just his mobile device. This will test the real limitations of Vuforia SDK for use in real CAVE environment.

Angles used for recognition: Part of the experiments will be also testing the view from different angles. It was decided to choose six viewing points, where will be the mobile device situated. In Figure 6-5 can be seen, that we chose three horizontal points, one in the left part of the CAVE, one in the middle and one in the right. In addition we chose two vertical points to try the difference between the middle and the top of the CAVE.

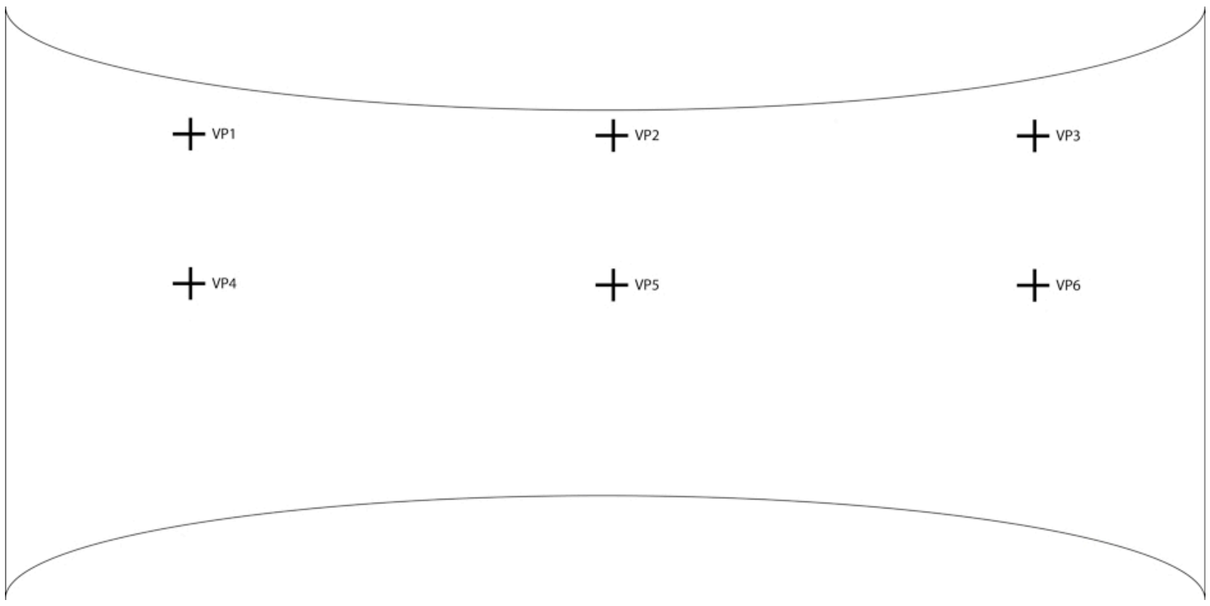


Figure 6-5: Viewing points in CAVE

The other part of viewing angles is the position of marker on the CAVE screen. Because of the curved screen, there is some distortion of the view. In the middle of projection screen, there is just a little of distortion, but the closer is the projected model to the side of the screen, the more are the parts of the model distorted. To bring this issue to consideration, we will also try three points as a placement of the marker on the screen; we will use left, middle and right.

Scale for quantification the experiments: We will use 3 levels for qualifying the recognition:

- Number one for very good recognition. This means, that the recognition is quick and faultless.

- Number <u>two</u> for possible recognition. This means, that the recognition is possible, but with some drawbacks like shivering model, which is placed on the marker, or it takes longer time to detect the marker.
- Number <u>three</u> when the recognition is not possible. This means that the recognizer is unable to recognize the marker at all.

Expected result of the experiment: We expect that the best results will be while using just 2D projection with the view from the middle of CAVE. Another expectation is that the best marker will be the Beverage List marker (Figure 6-2), because it has the most recognizable features. Now we need to find out whether is will be also usable with 3D and different angles and which of the markers will be best for it.

Experimenting in 2D

This experiment should discover the possibilities of object recognition on a curved screen and to find out the abilities of the markers and choose which one of them would be suitable for further experiments. We will do the experiment with all three markers we have chosen.

	Beverage list	**Rail map**	**Food list**
VP1 centre	1	1	2
VP1 left	2	2	3
VP1 right	2	1	2
VP2 centre	1	1	1
VP2 left	1	1	2
VP2 right	2	2	2
VP3 centre	1	3	2
VP3 left	2	1	2
VP3 right	3	3	2
VP4 centre	1	2	2
VP4 left	2	2	3
VP4 right	2	1	2
VP5 centre	1	2	1
VP5 left	1	1	2
VP5 right	2	2	2
VP6 centre	1	3	2
VP6 left	2	1	2
VP6 right	3	3	3

Table 1: Results of 2D experimenting

This experiment revealed the size of the problem with the real shape on the curved screen described by Figure 6-1. In the results we can see, that the best recognition we get when the user is in the middle of the CAVE, so in VP2 or VP5 and when the marker is placed in the middle of the screen. This is the result that we predicted. We can also see, that the best marker of the three used in this experiment is the Beverage List (Figure 6-2) and the worst is surprisingly the Food List (Figure 6-4), so we will not use it in the next experiments. This is caused by not sufficient amount of trackable features in it. Experiment revealed a strong sensitivity to light and contrast, which caused some differences between the recognition precision on the left and right side of the CAVE. The reason is probably some light coming from right side from outside of the CAVE, causing some differences. We can also see, that the difference between VP1 and VP4, VP2 and VP5 and VP3 and VP6 are small. This means, that it does not really matter, how high user holds the camera of his iPhone.

Main obstacle is the angle of the viewing and position of the marker on the screen and when the viewing angle is very sharp, the recognition is almost impossible. We should be able to deal with this issue better, so this experiment provides us a place for further improving.

Experimenting in 3D using polarization filters

In this experiment we will try to use polarization filter. This solution is not very good, because the user has to hold the filter in front of his camera and this can be annoying. On the other hand, the CAVE experience is better then in the previous case. Purpose of this experiment is to find out, whether there will be any difference between 2D projection and this solution. We will not use Food list marker since the previous experiment recovered its inappropriateness.

	Beverage list	**Rail map**
VP1 centre	1	1
VP1 left	2	2
VP1 right	1	1
VP2 centre	1	1
VP2 left	1	1
VP2 right	1	2
VP3 centre	1	2
VP3 left	1	1
VP3 right	3	3
VP4 centre	1	1
VP4 left	2	2
VP4 right	1	1
VP5 centre	1	1

VP5 left	1	1
VP5 right	1	2
VP6 centre	1	2
VP6 left	1	1
VP6 right	3	3

Table 2: Results of 3D with filter experimenting

As we can see from the Table 2, the results are very good and in some cases are even better than in 2D projection. As in previous experiment we can see that there is almost no dependence on high in which the camera is held.

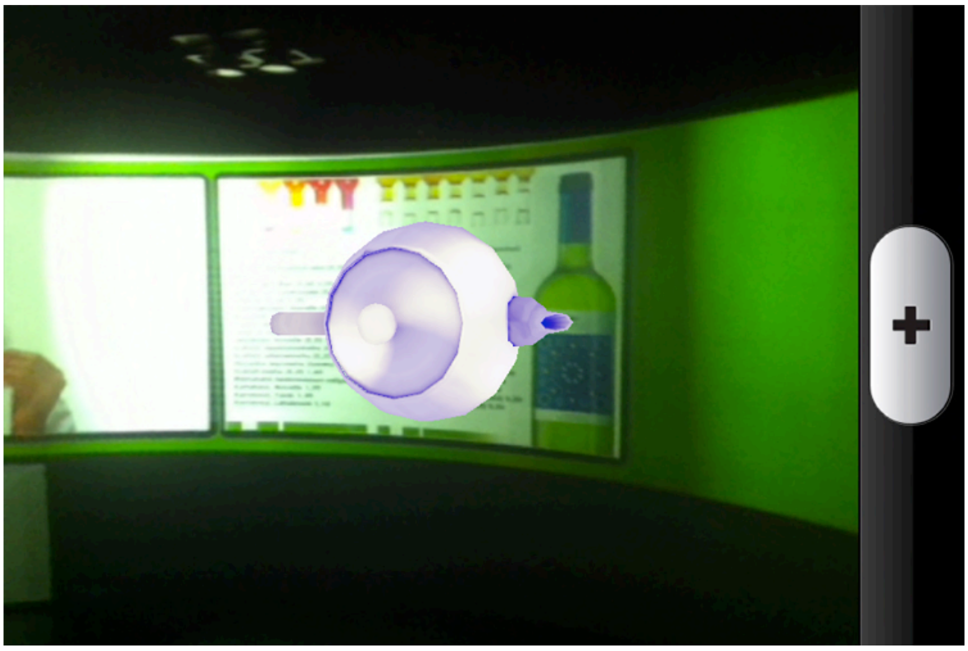


Figure 6-6: Demonstration of 3D projection using polarization filters

Experimenting in 3D without polarization filters

In this experiment we will use CAVE in normal work without any limitations, so with 3D and without using the filter. As in previous case, we will try the Beverage list marker and the Rail map marker. Purpose of this experiment is to find out, what the limitations of Vuforia SDK recognizer in the CAVE are and which kind of marker provide us the best results.

	Beverage list	**Rail map**
VP1 centre	2	2
VP1 left	3	2
VP1 right	2	2
VP2 centre	1	1
VP2 left	1	3
VP2 right	1	2
VP3 centre	2	2

VP3 left	2	3
VP3 right	2	3
VP4 centre	2	2
VP4 left	3	3
VP4 right	2	2
VP5 centre	1	1
VP5 left	1	3
VP5 right	1	3
VP6 centre	2	2
VP6 left	2	3
VP6 right	2	2

Table 3: Results of 3D without filter experimenting

In Table 3 we can see, that results are much worse than in previous cases. This is expected result. Rail map (Figure 6-3) is clearly not very suitable for use in 3D projection, since the recognition is in almost every case impossible; the recognizable features in the view are very blurry.

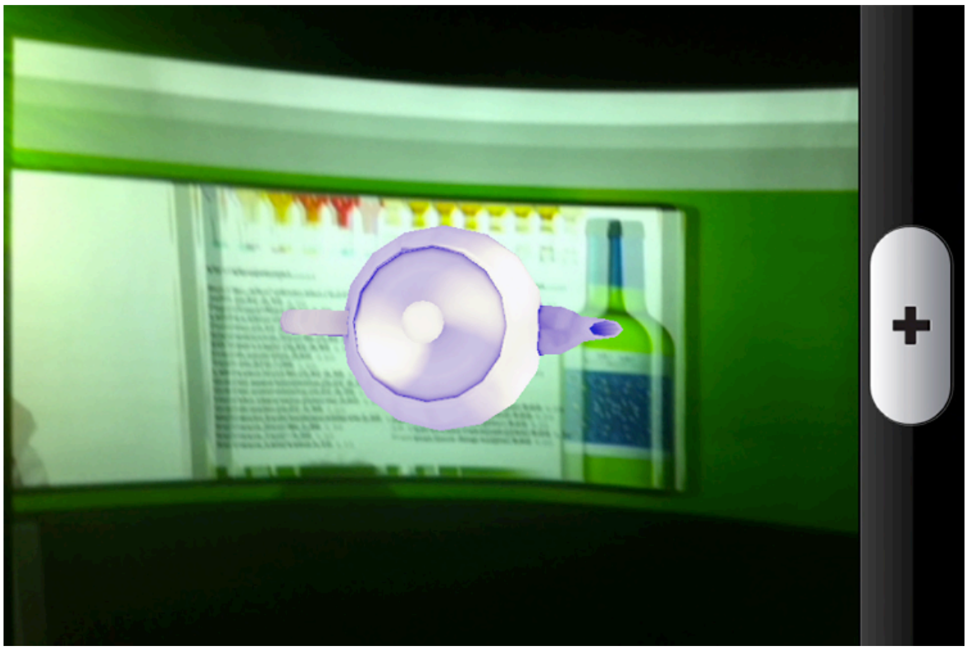


Figure 6-7: Demonstration of 3D projection in CAVE

This experiment revealed much bigger problem with using iPhone in 3D CAVE environment. When camera of iPhone is pointed to the CAVE projection screen, the view, which user sees at the iPhone screen, is not polarized any more. This means that even if the user is wearing the glasses, the image on the iPhone screen looks like if he is watching the projection screen without glasses.

Conclusion on first series of experiments

This series of experiment revealed us important facts about Vuforia SDK recognizer and its usability in CAVE. Probably the main issue is the damaging of 3D view while watching it through iPhone.

This is caused by the way, how 3D projection in CAVE works. As is written in Chapter 3.3, 3D view is based on polarized ripple of both views. Ripple in the view for one eye is horizontal and ripple for the second eye is vertical. However the polarized ripple is damaged by the iPhone and the view going out of the screen is not polarized anymore and the ripple is in 360 degrees range. This problem has basically two possible solutions. The first one is to use CAVE in 2D mode and the second one is to use polarization filters in front of the device camera. Both of these solutions are not very good, because it either does not use full power of the CAVE, or it makes user to hold the filter, which is unnatural. Moreover, the 3D experience on the place where is the device is ruined anyway. Even that 3D recognition in CAVE without polarization filter works quite well, when suitable marker is used, this issue makes it quite useless anyway. Proper solution of this problem would demand change of the hardware of the iPhone, so it would be able to show image, which is polarized again, or to change the viewing possibilities of the CAVE.

Second fact revealed by the experiments is that the recognition is the best, when the marker is in the middle of projection screen. If the marker is loaded into a program as rectangle, its shape on the sides of CAVE is very different, as can be seen in Figure 6-8, and it makes the recognizing process less accurate. Possible solution of this issue is to load the marker into a program three times. Once when it is in the middle, once on the right side and once on the left side. This idea will be tested by experiments later.

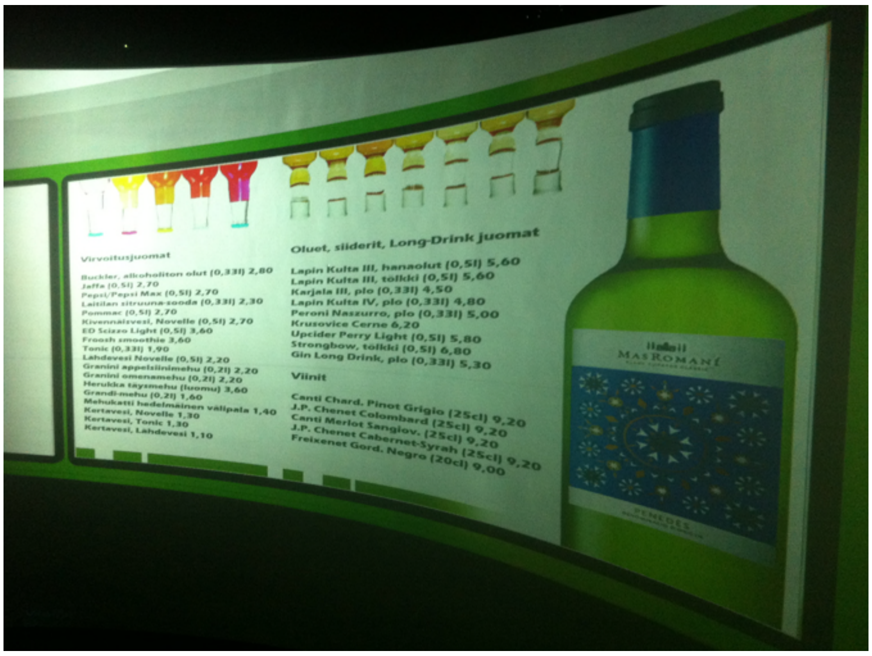


Figure 6-8: Demonstration of non-rectangular marker

All the tests confirmed that there is almost no difference in the height of device that user holds, so for further experiments we will use just one level in which the device will be held. On the other hand very important issue in the recognition process is the size of the marker. When the marker is smaller, recognition is almost impossible. Also when the marker is scanned and loaded into the application it is better when it is bigger.

6.2 Second series of experiments

Using results of previous tests we will now design more experiments to improve the performance of Vuforia SDK recognizer inside CAVE. Using full 3D projection is not a good idea because of facts stated above. Using 2D projection and 3D with polarization filters give almost the same results, so we do not have to perform the tests for both for these possibilities and we will use just 2D projection.

Where we need improvement is recognition of marker on different places on the screen and from various angles. We will try to change loading of the markers into application. In previous tests we loaded the marker just from the central position, when it was almost rectangular. Now we will load it on this position but also on the left and right of the screen as can be seen in Figure 2-1. Important thing is, that the markers are loaded into application while the user stands in the centre of the CAVE.

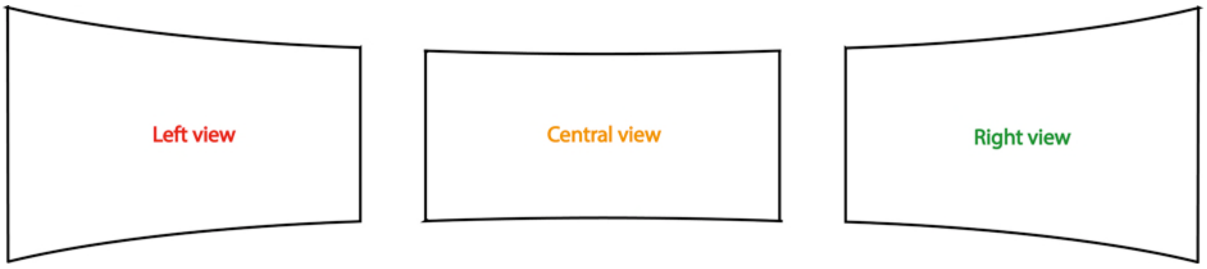


Figure 6-9: Three markers for loading into application

Aim of this experiment is to find out, whether after loading three different markers into application the ability to recognize marker placed anywhere on the screen will improve. We will also change viewing points and also places on the screen in comparison to the previous tests. In Figure 6-10 we can see, that we have viewing points only in one level.

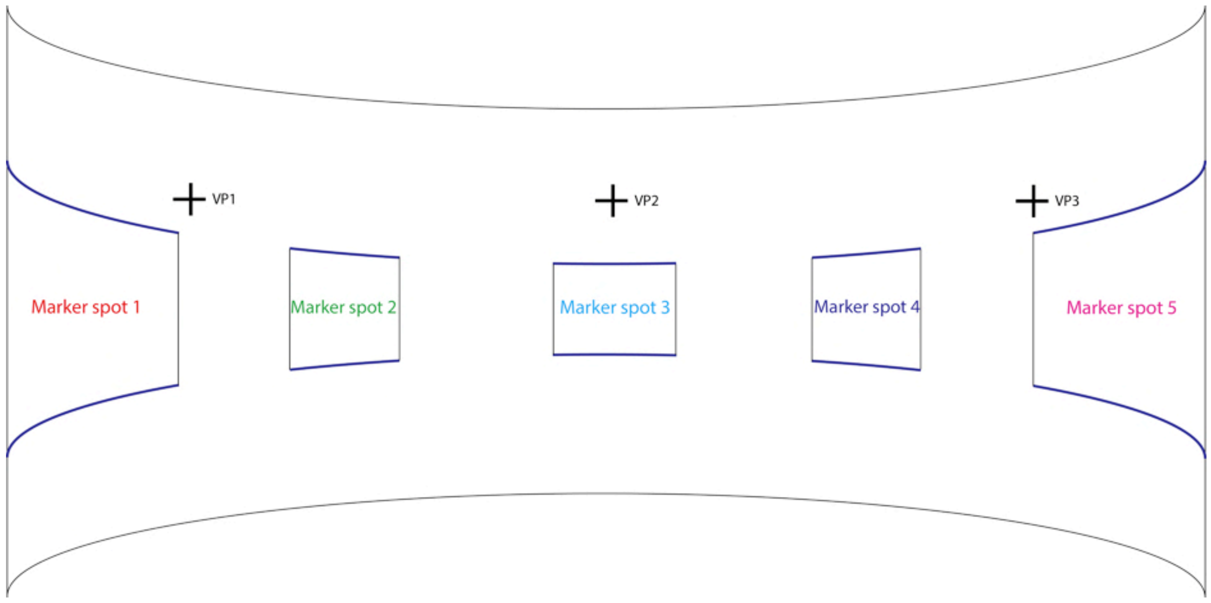


Figure 6-10: Viewing angles and marker spots

Recognition of the markers will occur on five different spots on the screen. The markers will be loaded to the application when being placed on positions Marker spot 1, Marker spot 3 and Marker spot 5. This should help us to discover, whether the recognition will be more precise than previous experiments. Other conditions will be the same as in previous experiments, so the same device with the same operating system and the same application, the same CAVE and the same model with the same markers.

	Beverage list	Rail map	Food list
VP1 MS1	2	2	3
VP1 MS2	2	2	3
VP1 MS3	2	1	1
VP1 MS4	2	1	1
VP1 MS5	2	2	2
VP2 MS1	1	1	1
VP2 MS2	2	1	1
VP2 MS3	1	1	1
VP2 MS4	2	1	1
VP2 MS5	2	1	1
VP3 MS1	2	2	1
VP3 MS2	1	2	1
VP3 MS3	1	1	1
VP3 MS4	1	1	2
VP3 MS5	1	3	1

Table 4: Results of experiments with more markers loaded

Results of the experiment are in Table 4. We can see, that the results are much better than in previous experiments, because almost in every case detection of the marker is possible as is proved by Figure 6-11. Nevertheless during the recognition in minor case occurred the behaviour that it seemed, that the application recognized two different markers and the exact position of the model in the application varies, so the model is skipping from one position to the other. Fortunately this does not happen very often. In this case even the Food list marker does not appear to be as bad as it looked like in previous tests.

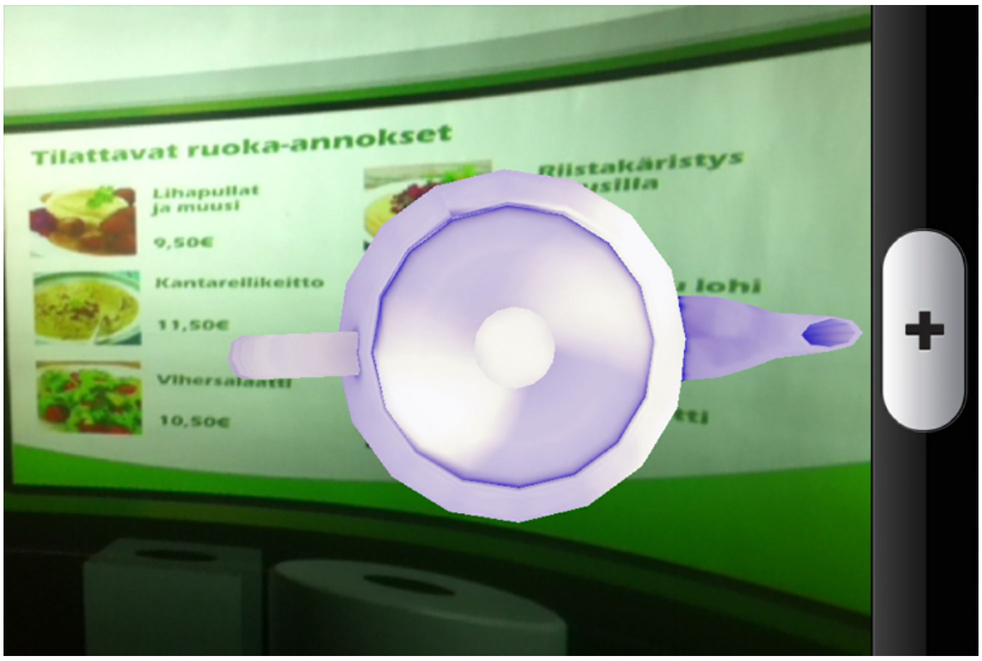


Figure 6-11: Solved problem of non-rectangular marker

6.3 Conclusion on experiments

The experiments revealed us more useful facts about recognizing markerless markers in CAVE. We discovered, that curved screen makes recognizing harder and that if we want recognition also on the sides of the screen, we need to make some adjustments. Those adjustments are, that we will load three markers into the application, one on the left side, one on the right side and one in the centre of the screen. This is easy and quite sufficient solution of this problem. Because of the fact that the recognizer is embedded into Vuforia SDK and it is not open source, it is impossible to change it. The only solution would be to write brand new augmented reality framework from scratch just because of the recognizer that would be working in this specific CAVE. The recognizer would have to be more complex than the one in Vuforia SDK and it would slow down the recognition, so it would not probably be usable in the real-time or it would need higher hardware requirements of the device. The only advantage would probably be, that it would work in this kind of CAVE. If the CAVE would be like most of the CAVEs are, it would be rectangular and not curved, so it would not work there anyway.

The main problem still is the fact, that iPhone corrupts the image and 3D view from the CAVE is no longer available on the screen of the device. As a result it is more appropriate to use polarization filter in front of the device camera or pure 2D projection instead of 3D, because the corrupted 3D view is worse than the 2D and the recognition in 3D is not as accurate as the one in 2D.

We also have some markers that can be actually used. The best one is the Beverage list (Figure 6-2), but also the Food list (Figure 6-4) and Railway map (Figure 6-3) are also sufficient. Since we have the results of experimenting with these concrete markers in CAVE and we have some knowledge about the markers in Vuforia SDK (Chapter 5.3), we would be able to add more markers into the application later.

7 Developing application using Vuforia SDK

Vuforia SDK is very complex framework with architecture, which is not that straight forward. Application can be altered to a programmers needs, but the programmer has to know, where should he make the changes.

7.1 Importing 3D models into Vuforia SDK based application

Vuforia SDK uses OpenGL style for defining 3D objects. This means, that all 3D objects are composed of triangles and the models are imported in a special header file. Creating own complex objects as a set of triangles is almost impossible. The solution of this problem is using some 3D modelling software and that converts the model to a header file, which can be used by Vuforia SDK. As modelling software can be used for example Blender, because it is free to use and also it is able to export models into a Wavefront .obj file.

After having object in `.`obj file we can use for example script created by Heiko Behrens called `obj2opengl` [18]. It is a simple Perl script which converts .obj file into header, which is suitable to use for OpenGL and therefore also Vuforia SDK. It is able to convert also coordinates for normals and textures.

However there can occur a small problem in importing models. By default in Vuforia rendering is enabled back-face culling. This means, that the triangles in OpenGL have only one side, which is considered the front side [19]. All triangles are visible only from one side and not from the opposite one. Difference between front and backside is in defining the coordinates of the triangle. If the coordinates are defined clockwise, the triangle will be visible only from the backside and on the other hand, if the coordinates are defined as counter-clockwise, the triangle will be visible only from the front. Illustration of the problem can be seen in Figure 7-1, where the triangle on the left will be visible only from the backside and triangle on the right just from the front side.

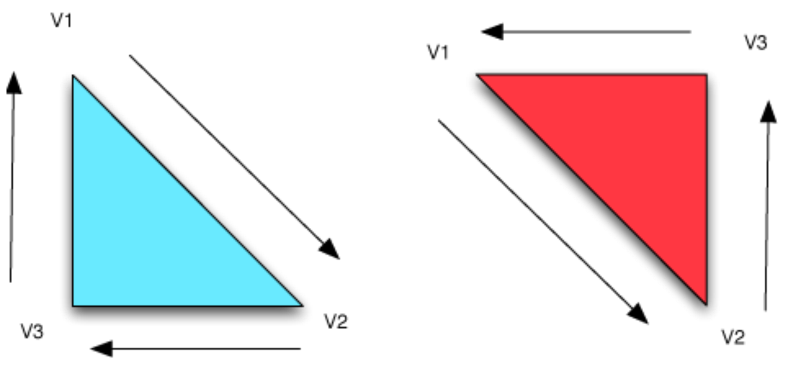


Figure 7-1: Clockwise and counter-clockwise definition of coordinates [19]

Because some models do not consider this problem, there can be some problems in rendering. Result of using model, which does not follow this rule can be seen in Figure 7-2 where can be seen, that only about half of triangles are rendered properly and that the model has holes in it and those triangles, which are not now visible are visible from the backside.

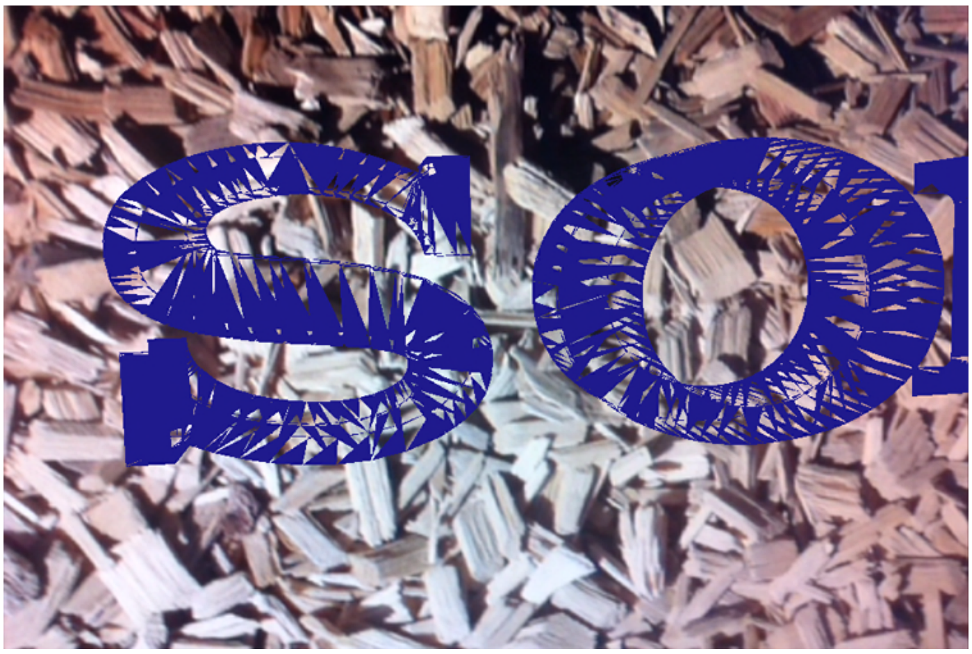


Figure 7-2: Demonstration of back-face culling problem

Fortunately there is quite easy solution to this problem; it is enough, when line of code `glEnable(GL_CULL_FACE)` is replaced by `glDisable(GL_CULL_FACE)`. After that, everything should be fine.

OpenGL ES in iPhone 4 is able to render in real time model, which has tens of thousands triangles without much problems. However there are some limitations. An experiment of importing header file with size of 300 MB with model, which has 460.000 triangles failed, because the device was not able to handle such a complex model.

7.2 Adding custom markers into application

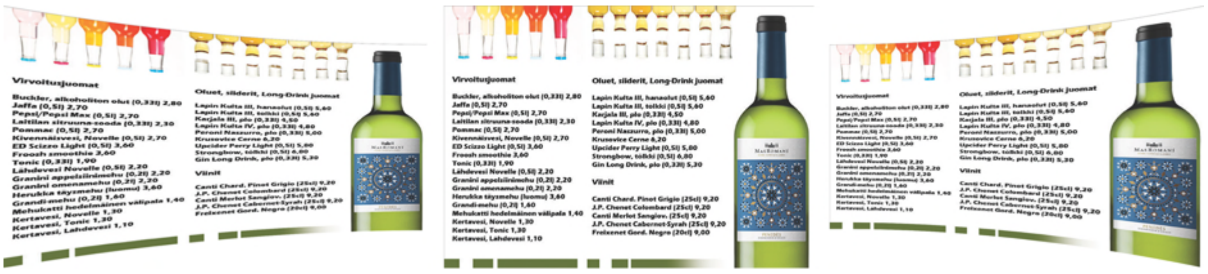
Other important issue about creating own augmented reality application based on Vuforia SDK framework is to add custom marker/markerless markers into it. The way, how to do that is through web service, where registered developer can create his own marker database, import markers in the form of images and after that download .xml and .dat files and import those in the project. Then it is enough just to change the name of this database in the code and in rendering function set what should be rendered on which marker. The whole approach is very straightforward, but on the other hand, all parts of it are embedded into the Vuforia SDK and it is impossible to perform even the slight changes.

7.3 Textures

Vuforia SDK based application enables textures. But there are two issues connected with it. First of all the model has to have the texture coordinates included in the header file. If the coordinates are not right, texture will not be rendered properly. Second limiting fact is, that the size in pixels of the texture must be power of number two, so for example 128x128 or 1024x512. It does not have to be square. If either of these conditions is not fulfilled, the texture will not be rendered properly and the object will be black.

8 AR application for CAVE

In this chapter the actual result of the work is described. Markers, which are used in the final application are described in the Chapter 6.2. That means that in the application are three different markers; Food list, which is in the Figure 6-4, Beverage list in the Figure 6-2 and Railway map in the Figure 6-3. Each one of them is loaded into an application in three different angels. Example of this solution can be seen in the Figure 8-1. The content displayed for every one of this the different representations of the markers is the same.



Virvoitusjuomat

Buckler, alkoholiton olut (0,33l) 2,80
Jaffa (0,5l) 2,70
Pepsi/Pepsi Max (0,5l) 2,70
Laitilan sitruuna-sooda (0,33l) 2,30
Pommac (0,5l) 2,70
Kivennäisvesi, Novelle (0,5l) 2,70
ED Scizzo Light (0,5l) 3,60
Froosh smoothie 3,60
Tonic (0,33l) 1,90
Lähdevesi Novelle (0,5l) 2,20
Granini appelsiinimehu (0,2l) 2,20
Granini omenamehu (0,2l) 2,20
Herukka täysmehu (luomu) 3,60
Grandi-mehu (0,2l) 1,60
Mehukatti hedelmäinen välipala 1,40
Kertavesi, Novelle 1,30
Kertavesi, Tonic 1,30
Kertavesi, Lähdevesi 1,10

Oluet, siiderit, Long-Drink juomat

Lapin Kulta III, hanaolut (0,5l) 5,60
Lapin Kulta III, tölkki (0,5l) 5,60
Karjala III, plo (0,33l) 4,50
Lapin Kulta IV, plo (0,33l) 4,80
Peroni Nazzurro, plo (0,33l) 5,00
Krusovice Cerne 6,20
Upcider Perry Light (0,5l) 5,80
Strongbow, tölkki (0,5l) 6,80
Gin Long Drink, plo (0,33l) 5,30

Viinit

Canti Chard. Pinot Grigio (25cl) 9,20
J.P. Chenet Colombard (25cl) 9,20
Canti Merlot Sangiov. (25cl) 9,20
J.P. Chenet Cabernet-Syrah (25cl) 9,20
Freixenet Gord. Negro (20cl) 9,00

Figure 8-1: Beverage List loaded into application in three different angels

8.1 Displayed content

Because we have three final markers in the application, we will display three different kinds of content on it.

Displayed content for Railway Map

Content for this marker is the simplest one, because it is just display of the picture of the Finnish railway network. Unfortunately it is not as easy as it looks like, because OpenGL ES does not support the direct rendering of pictures. That means, that we have to create a model of a 2D plane and then project the picture on the plane as a texture. In that case we need to fulfil the conditions mentioned in Chapter 7.3.

This model should also enable user interaction controlled with gestures. This means adding gesture recognizer for rotate and pinch, so `UIRotationGestureRecognizer` and `UIPinchGestureRecognizer`. These features will enable the user a possibility of rotating and resizing the picture on the plane. In the Figure 2-1 can be seen the result. There is a map of railways in Finland, which was rotated by two-finger rotation gesture.

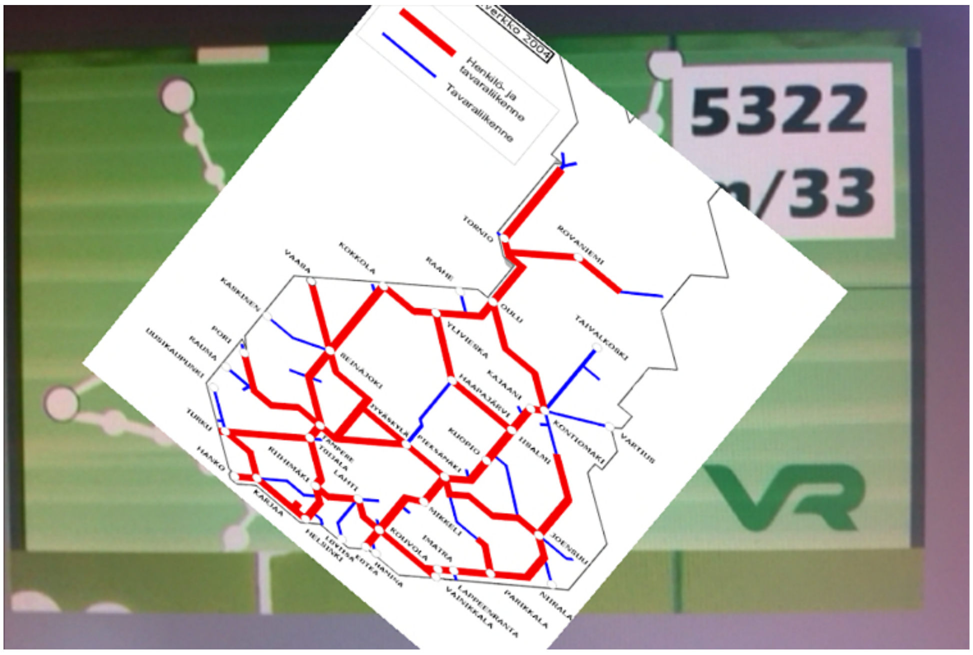


Figure 8-2: Content for Railway map marker showing rotation

Displayed content for Beverage List

It was decided, that some sign with 3D text would be displayed over the Food List marker as can be seen in Figure 8-3. Unfortunately Vuforia SDK uses OpenGL ES rendering engine and it does not have straight support for rendering 3D text. There are several ways, how to deal with this issue. First one is to use different engine or third party solution. Second one would be to create the 3D text in runtime. This would demand importing header files with all the letters models in the alphabet and then connect these models together. The result of it would be flexible text rendering and easy change of the rendered text. Unfortunately it would slow the application down and also all the header files would take up a lot of space of the device. Since there is not actually a need to change the text very often, because the main aim is to experiment with the possibilities of the objects, the easiest way is to create the whole text as a model and import it in the application in one piece. Unfortunately if there would be a demand for changing the text, it would mean create a new model and replace the original one. If the coordinates in the header file would have the same name, it would mean just replacing the header with the new text and rebuild the application. That does not mean that big intervention into the application. To colour the text, a single colour texture is used. As in the previous case, user interaction is enabled, so user can resize the sign. Rotation in this case seems useless, so it is not implemented.

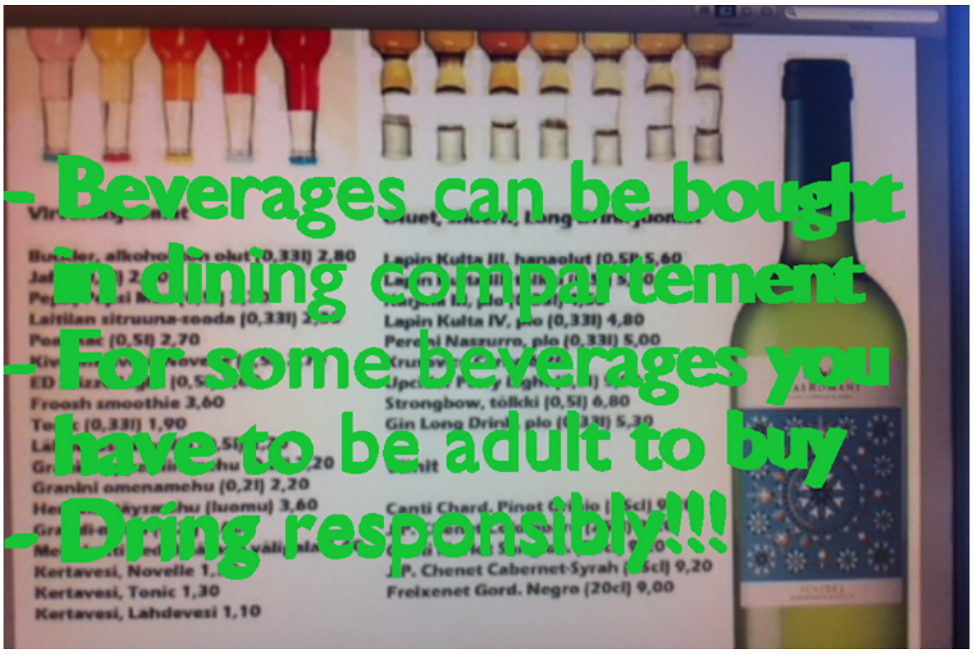

- Beverages can be bought in dining compartement
- For some beverages you have to be adult to buy
- Dring responsibly!!!

Figure 8-3: 3D text sign displayed over the Beverage List marker

Displayed content for the Food List marker

Probably the most interesting content was chosen for the Food List marker. Because the marker contains pictures of six different meals, it was decided that it purpose would be to make an animation of enlarging the pictures of the meals, which will also come closer to the user and enables him to order food. Result can be seen in the Figure 8-4.

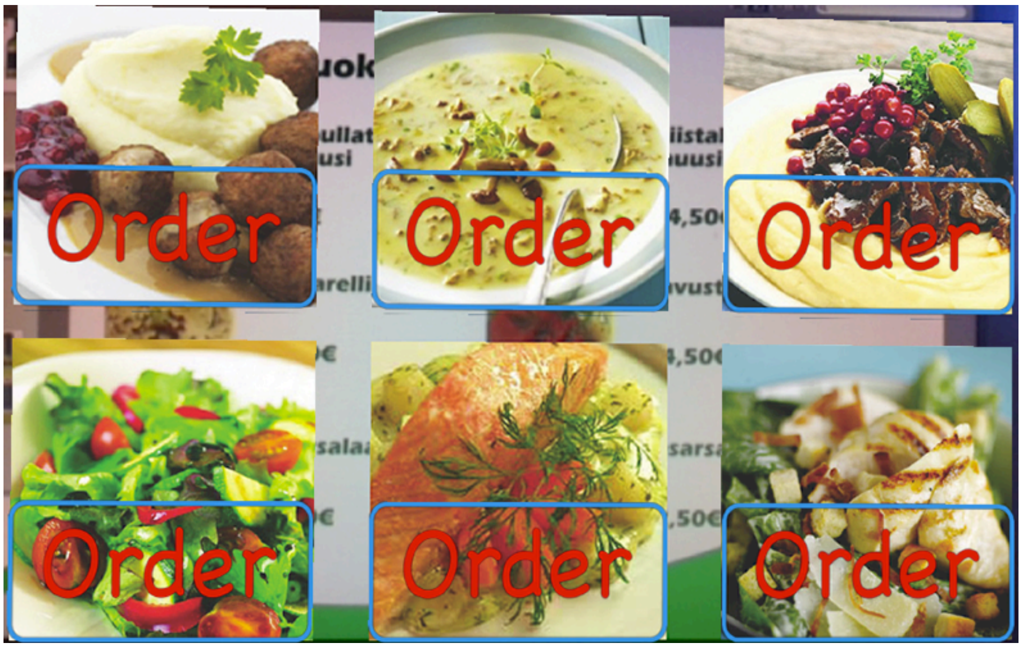
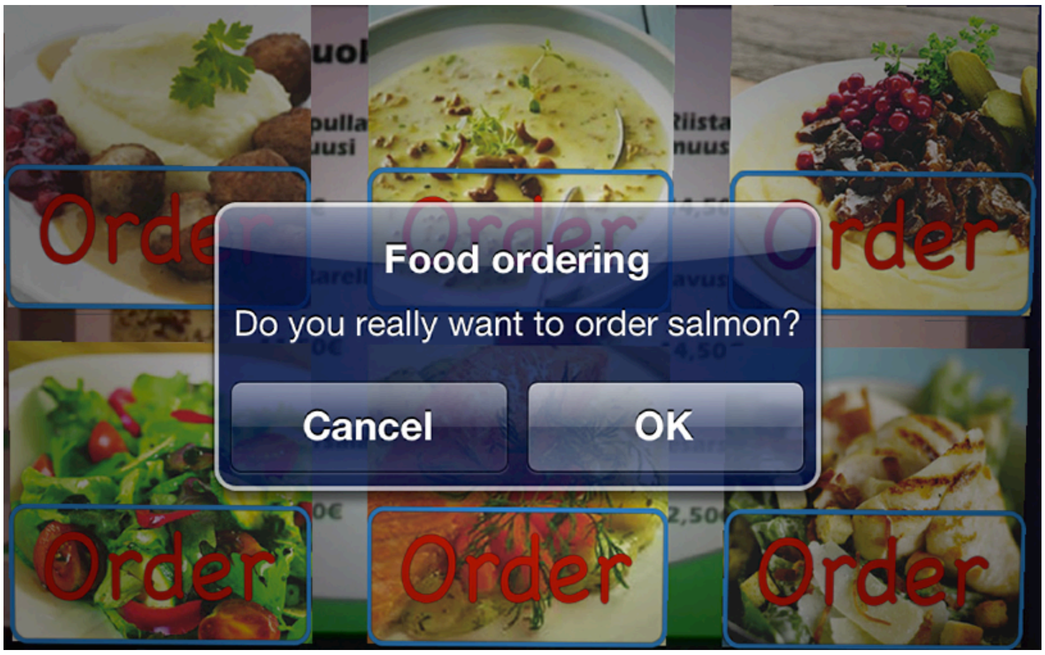


Figure 8-4: Ordering food on Food List marker

In this case also the user interaction is must be solved, because the user should be able to order food, by clicking the "Order" button on the meal. This feature needs some additional programing, because the application have to know, which one of the meals user wants. Unfortunately there is no easy way to decide which object on the screen user touched. This would need an implementation of ray tracing algorithms, which would slow the application down. Another way, how to solve this issue had to be found. Fortunately, there is a way, how to determine which position on the marker user touched. Then the application just have to find out, whether the "Order" button was touched and which food user actually wants to order as can be seen in the Figure 8-5. Knowing the size and the position of the buttons can solve this problem. After choosing the food, user has to confirm his choice to process ordering the food. Now nothing actually happens, but in the real use in the train food compartment, it could send a request to the ordering system server and display a demand for the particular food in the kitchen. It could also return an ordering number to the user and then he can just take his order, after it is prepared.



Food ordering

Do you really want to order salmon?

Figure 8-5: Ordering the food process

Ordering process is also highlighted by changing sign on the "Order" button on selected food as can be seen in Figure 8-6. This change is just for few seconds and that the original "Order" sign appears back.

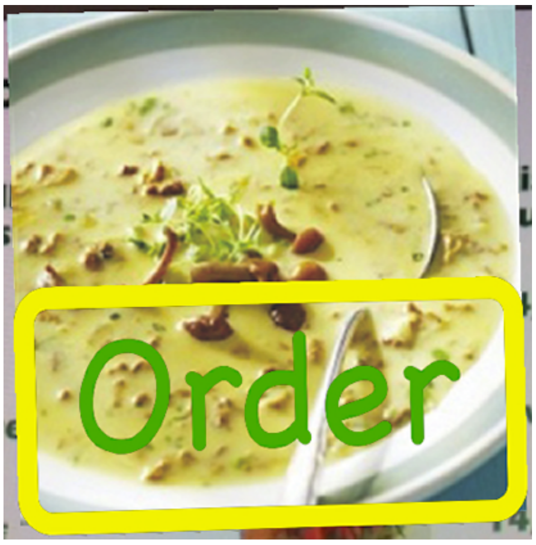


Figure 8-6: Change of sign after the meal is ordered

The actual models displayed on the marker are planes with textures like in the case of Railway Map marker. There are six different planes with six different textures loaded into an application. Over each plane with a picture of meal another plane is rendered with texture of sign with word "Order" on it. Transparency of the texture is enabled by commands `glEnable(GL_BLEND)` and `glBlendFunc(GL_SRC_ALPHA, GL_ONE_MINUS_SRC_ALPHA)`. It is important to disable this function in the right time; otherwise it would destroy all the other textures. Obviously the texture is saved in .png picture file, which has no background color. The result of this solution is that there are 12 models rendered, but since each one of them is created just by two triangles, there are just 24 triangles in total, which has to be rendered. That is reasonable number, so it does not slow the whole application down. Another solution of the situation would be to add the order sign right into the texture with a meal. Advantage of the two overlaying planes solution is that for example if the restaurant is run out of food, it would have possibility to disable the option of ordering the particular meal. Also the change of the texture for the actually ordered meal is possible. In this case also user gestures are enabled, but unlike in previous case with pinching the pictures do not become bigger, they just come closer to the user in sense that the distance between the pictures and the marker gets bigger. Reason for this is, that if the pictures would become bigger, they would start to overlap and it would cause a problem with rendering.

9 Application testing

Application was fully tested in the CAVE environment. By this testing was proved, that the recognition works rather well using various angles. For example in the Figure 9-1 we can see, that the augmentation works even if the marker is not heading straight to the user.

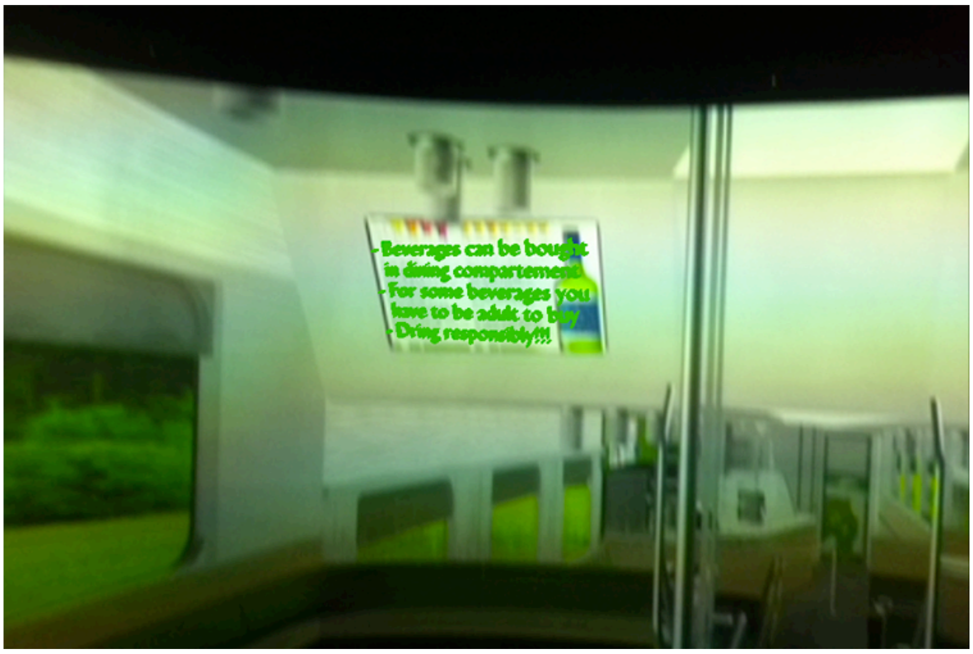


Figure 9-1: Example of different angle

On the other hand, because of the used optimization, sometimes the augmented content was not heading in the correct angle as can be seen in Figure 9-2. On the other hand, usually these troubles were temporary and the angle changed after a while.

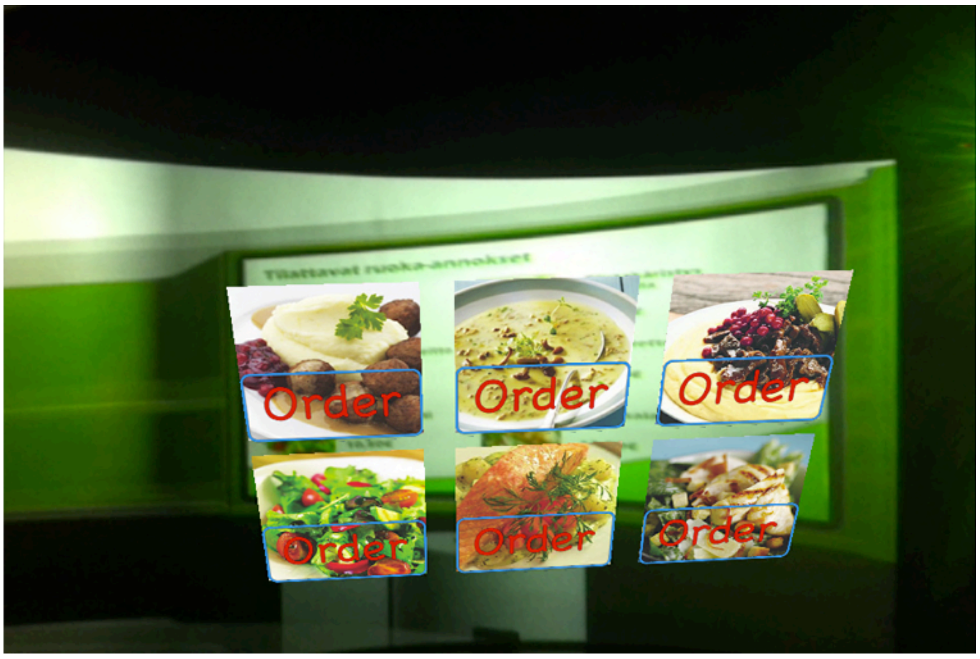
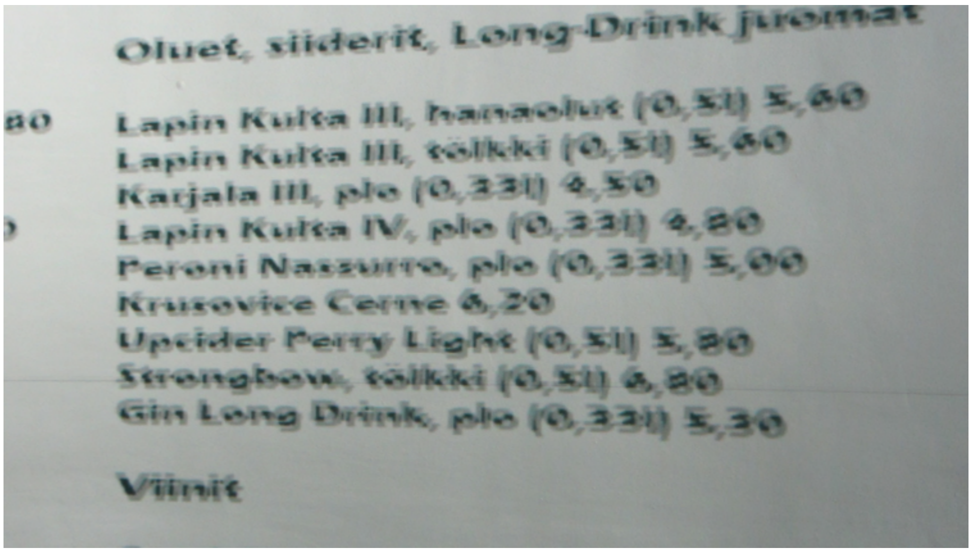


Figure 9-2: Failure of recognition

Quite significant troubles were connected with light in the CAVE. The recognition was optimized for 2D projecting. As is written in Chapter 3.3, there are two ways, how to do it. If we use only one projector, the brightness can be too low for the camera in iOS device and the precision of the recognition might be getting lower. The other solution, projecting two images on the same spot one over the other, can also lead to problems, since sometimes the projection does not work absolutely precisely and the views are not on the same spot as can be seen in Figure 9-3. This is caused by the fault of the projector setting, which has to be retuned. Clearly the precision of recognition is lower, because the view seems blurry and it is harder to recognize key features in it.



Oluet, siiderit, Long-Drink juomat

Lapin Kulta III, hanaolut (0,5l) 5,60
Lapin Kulta III, tölkki (0,5l) 5,60
Karjala III, plo (0,33l) 4,50
Lapin Kulta IV, plo (0,33l) 4,80
Peroni Nazzurro, plo (0,33l) 5,00
Krusovice Cerne 6,20
Upcider Perry Light (0,5l) 5,80
Strongbow, tölkki (0,5l) 6,80
Gin Long Drink, plo (0,33l) 5,30

Viinit

Figure 9-3: Fail of the projection

During developing and testing the application, problems like that occurred quite often mainly because whole CAVE is still in its testing run and needs more development and precise setting of the whole environment. Another problem was connected with different projection behaviour on different spots of the curved projection screen. Problems like this were quite challenging during the whole work. The most problematic area is the one where picture from two 3D projectors, so four single projectors, overlaps, because it was even harder to adjust the projectors correctly. Even when the setting is right, it can be seen clearly, where the overlapping area is, since it is slightly darker than the rest of a screen, as is demonstrated by Figure 9-4.

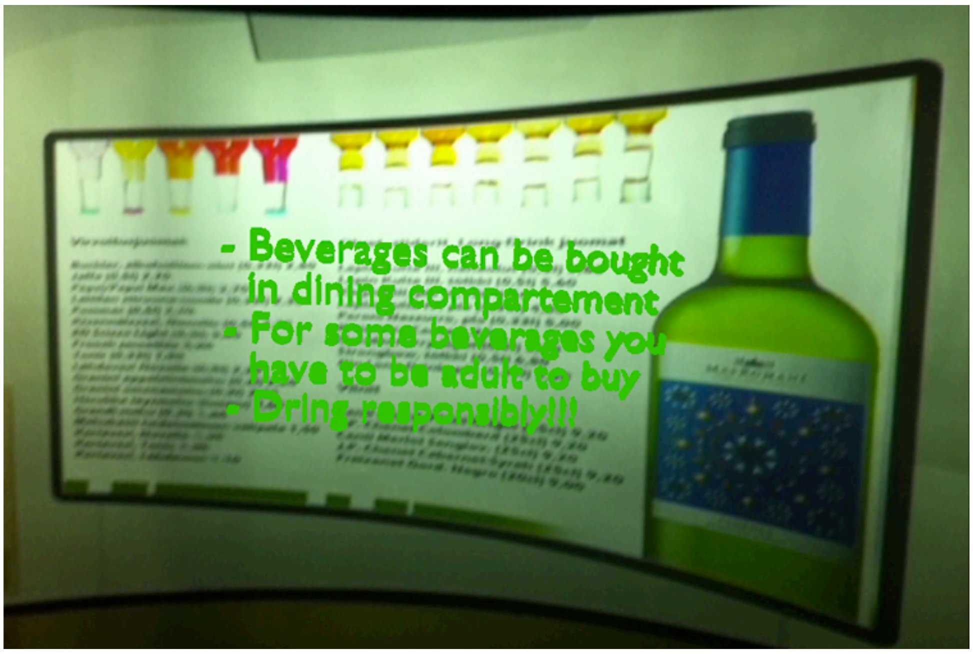

- Beverages can be bought in dining compartement
- For some beverages you have to be adult to buy
- Dring responsibly!!!

Figure 9-4: Dark stripe in overlapping area (in the left part of the picture)

Even though the main purpose was not to develop application for 3D use of CAVE, experiments were done also in this area. The markers loaded into the application were not changed in this case and were just in 2D. Result of this is, that the recognition process does not works as good, mainly because it takes longer time and also is much more susceptible to different angles, small markers and bad lighting. If the markers were loaded in the application in 3D view it would be even better, but because the results from Chapter 6.3 it was not the aim.

10 Conclusion

The aim of this thesis was to discover the possibility of using augmented reality concepts in Cave Automatic Virtual Environment. As a result an iOS compatible application for demonstration this possibilities of use was created. Unfortunately several significant obstacles were discovered. First and the most limiting one is the fact, that because of use of passive 3D projection in the CAVE, the 3D effect is not working anymore after passing through the mobile device. This means that the projection has to use either 2D projection or polarization filters in front of the mobile device camera. Either way, the 3D effect is gone. It would be interesting to see, how would the augmented content affect the 3D view seen by the user on the screen, because it could also look very unnatural.

Because of those reasons, it was decided to focus just on 2D CAVE. Several series of experiments were done to discover the possibilities of object recognition in augmented reality framework Vuforia SDK. The experiments revealed several other issues, like problem with different viewing angles and mainly problems with object recognition on curved projection. Designing an optimization for improving the recognition precision solved these problems. The optimization is based on importing more markers into the application with some distortion, so the shape of the marker matches the shape of the image on curved plane. This solution improved the recognizing process massively.

As a result of the experiments, demonstrating application was developed, which works in specified model projected into CAVE. The projected augmented information was designed to provide the user some added information. This information is not static and the user has possibility to change the content with gestures and touches.

Developed application was tested in CAVE, but with the markers on paper or on computer it works as well. It is crucial to add that the application is designed specifically for the CAVE in Oulu University of Technology, where is used curved screen instead of conventional rectangular shape. Development for curved plane was more challenging because the recognizer in Vuforia SDK is designed for 2D flat recognition. The application works in the mentioned CAVE properly. On the other hand in works only in 2D due to used technology and hardware limitations.

This thesis proved, that using augmented reality in CAVE could be useful, because it provides interaction with the parts of the model to the user. Also the content of the augmented reality mobile application can be changed more easily that the complex model, which is loaded into CAVE. It is possible to detect markers in the CAVE with several limitations. With those in mind, development of such application is possible with quite precise markerless marker recognition process.

The future work continuing this thesis can be of course adding more markers into the model, developing applications for more models, trying to find more interesting content, to be added into the environment or more ways to let the user interact with the content. If those new applications would prove being useful and people would use them, it could be also interesting to develop special framework for augmented reality, which would have special recognizer to be used on the curved screen. This would improve the precision of the recognition, but might also create a demand for more powerful mobile devices since it would have to be much more complex. In area not connected with augmented reality, but with mobile devices application development and CAVE can be very useful to create system for controlling the whole virtual environment with the mobile device, sort of remote control. So there are a lot of possible scenarios, of how to continue with this thesis.

Resources

[1] DUBOIS E., GRAY P., NIGAY L.: *The Engineering of Mixed Reality Systems*. Springer. London, 2010. ISBN: 978-1-84882-733-2.

[2] ROCHE K.: *Pro iOS 5 Augmented Reality*. Apress. New York, 2011. ISBN: 978-1-4302-3912-3

[3] WWW.CIRCUITSTODAY.COM: *Augmenter Reality (AR) Technology* [online] 2011. [Quot. 2012-19-12]. Available at URL: <http://www.circuitstoday.com/augmented-reality-technology>.

[4] WWW.OAMK.FI: *Mixed reality* [online] 2012. [Quot. 2012-19-12]. Available at URL: <http://www.oamk.fi/hankkeet/mixedreality/>.

[5] WWW.POCKET-LINT.COM: *The history of augmented reality* [online] 2011. [Quot. 2012-20-12]. Available at URL: <http://www.pocket-lint.com/news/38803/the-history-of-augmented-reality>.

[6] WWW.OWNI.EU: *Defending and clarifying the term augmented reality* [online] 2011. [Quot. 2012-20-12]. Available at URL: <http://owni.eu/2011/05/06/defending-and-clarifying-the-term-augmented-reality/>.

[7] WWW.NOKIPEDIA.COM: *Maps on Windows Phone 8* [online] 2012. [Quot. 2012-20-12]. Available at URL: <http://www.nokipedia.com/frequently-asked-questions-maps-on-windows-phone-8/>.

[8] WWW.WIKIPEDIA.ORG: *Cave automatic virtual environment* [online] 2012. [Quot. 2012-30-12]. Available at URL: <http://en.wikipedia.org/wiki/Cave_automatic_virtual_environment>.

[9] WWW.JVRB.ORG: *Precise Near-to-Head Acoustics with Binaural Systems* [online] 2006. [Quot. 2012-30-12]. Available at URL: <http://www.jvrb.org/past-issues/3.2006/589>.

[10] WWW.OAMK.FI: *Technical set-up* [online] 2012. [Quot. 2012-30-12]. Available at URL: <http://www.oamk.fi/hankkeet/cave/setup/>.

[11] WWW.BUSINESSINSIDER.COM: *CHART: Android Dominates Mobile Platform Market Share* [online] 2012. [Quot. 2012-29-12]. Available at URL: <http://www.businessinsider.com/mobile-platform-market-share-2012-8>.

[12] WWW.POPCODE.INFO: *What is Popcode?* [online] 2011. [Quot. 2012-30-12]. Available at URL: <http://popcode.info/whatispopcode.html>.

[13] WWW.POWEREDBYSTRING.COM: *String Augmented Reality* [online] 2012. [Quot. 2012-20-11]. Available at URL: <http://www.poweredbystring.com/>.

[14] WWW.QUALCOMM.COM: *Augmented Reality (Vuforia)* [online] 2012. [Quot. 2012-20-11]. Available at URL: < https://developer.qualcomm.com/mobile-development/mobile-technologies/augmented-reality>.

[15] WWW.ARLAB.COM: *Markerless Augmented* Reality [online] 2012. [Quot. 2013-18-02]. Available at URL: <http://www.arlab.com/blog/markerless-augmented-reality/>.

[16] WWW.WIKIPEDIA.ORG: *Guernica (painting)* [online] 2013. [Quot. 2013-26-02]. Available at URL: <http://en.wikipedia.org/wiki/Guernica_(painting)>.

[17] WWW.WIKIPEDIA.ORG: *San Giorgio Maggiore at Dusk (Monet)* [online] 2013. [Quot. 2013-26-02]. Available at URL: <http://en.wikipedia.org/wiki/San_Giorgio_Maggiore_at_Dusk_(Monet)>.

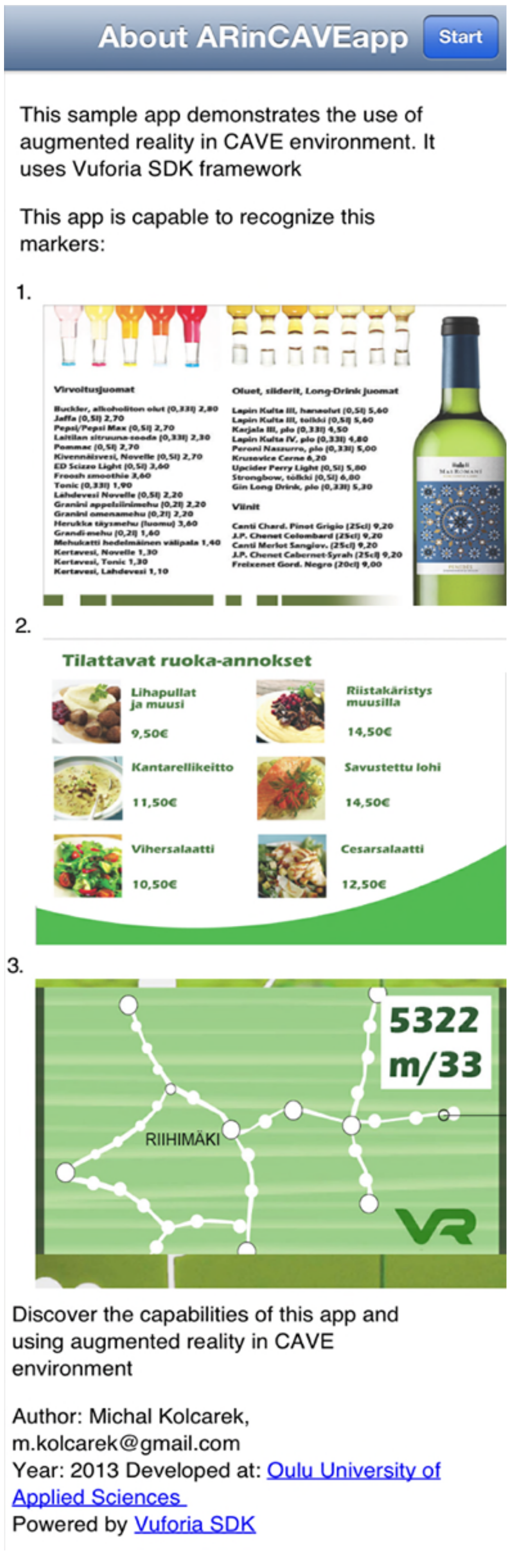
[18] BEHRENS H.: *obj2opengl: convert obj 3D models to arrays compatible with iPhone OpenGL ES* [online] 2009. [Quot. 2013-19-02]. Available at URL: <http://heikobehrens.net/2009/08/27/obj2opengl/>.

[19] LAMARCHE J.: *OpenGL ES From the Ground Up* [online] 2009. [Quot. 2013-04-03]. Available at URL: <http://iphonedevelopment.blogspot.fi/2009/04/opengl-es-from-ground-up-part-1-basic.html>.

List of appendixes

Appendix 1. User-oriented description of the application

Control of the application is very easy and straightforward. User just click on the application icon (Figure 2), after that the splash screen appears (one of those listed in Appendix 3. Application splash screens) and then a screen with application description and mainly description of the markers, that is user supposed to find appear. This help webpage is in Figure 1.



This sample app demonstrates the use of augmented reality in CAVE environment. It uses Vuforia SDK framework

This app is capable to recognize this markers:

1.

Virvoitusjuomat

Buckler, alkoholiton olut (0,33l) 2,80
Jaffa (0,5l) 2,70
Pepsi/Pepsi Max (0,5l) 2,70
Laitilan sitruuna-sooda (0,33l) 2,30
Pommac (0,5l) 2,70
Kivennäisvesi, Novelle (0,5l) 2,70
ED Scizzo Light (0,5l) 3,60
Froosh smoothie 3,60
Tonic (0,33l) 1,90
Lähdevesi Novelle (0,5l) 2,20
Granini appelsiinimehu (0,2l) 2,20
Granini omenamehu (0,2l) 2,20
Herukka täysmehu (luomu) 3,60
Grandi-mehu (0,2l) 1,60
Mehukatti hedelmäinen välipala 1,40
Kertavesi, Novelle 1,30
Kertavesi, Tonic 1,30
Kertavesi, Lähdevesi 1,10

Oluet, siiderit, Long-Drink juomat

Lapin Kulta III, hanaolut (0,5l) 5,60
Lapin Kulta III, tölkki (0,5l) 5,60
Karjala III, plo (0,33l) 4,50
Lapin Kulta IV, plo (0,33l) 4,80
Peroni Nastzurro, plo (0,33l) 5,00
Kruzovice Cerne 6,20
Upcider Perry Light (0,5l) 5,00
Strongbow, tölkki (0,5l) 6,00
Gin Long Drink, plo (0,33l) 5,30

Viinit

Canti Chard. Pinot Grigio (25cl) 9,20
J.P. Chenet Colombard (25cl) 9,20
Canti Merlot Sangiov. (25cl) 9,20
J.P. Chenet Cabernet-Syrah (25cl) 9,20
Freixenet Gord. Negro (20cl) 9,00

2.

Tilattavat ruoka-annokset

Lihapullat ja muusi 9,50€	Riistakäristys muusilla 14,50€
Kantarellikeitto 11,50€	Savustettu lohi 14,50€
Vihersalaatti 10,50€	Cesarsalaatti 12,50€

3.

Discover the capabilities of this app and using augmented reality in CAVE environment

Author: Michal Kolcarek, m.kolcarek@gmail.com
Year: 2013 Developed at: Oulu University of Applied Sciences
Powered by Vuforia SDK

Figure 1: Help web page

After that user just presses button “Start” in top right corner and then he can find the specified markers and enjoy the augmentation.

Appendix 2. Application icon

Together with this application, icon was created in all the needed resolution for iPhone/iPad, which is in the Figure 2.

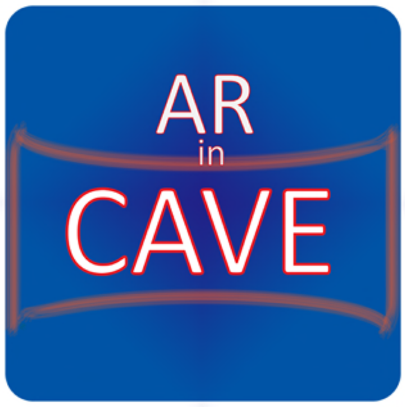


Figure 2: Application icon

Appendix 3. Application splash screens

For better feel about the application also all the demanded splash screens were created. That means splash screen for iPhone (Figure 3), portrait oriented splash screen for iPad (Figure 4) and landscape oriented splash screen for iPad (Figure 5).

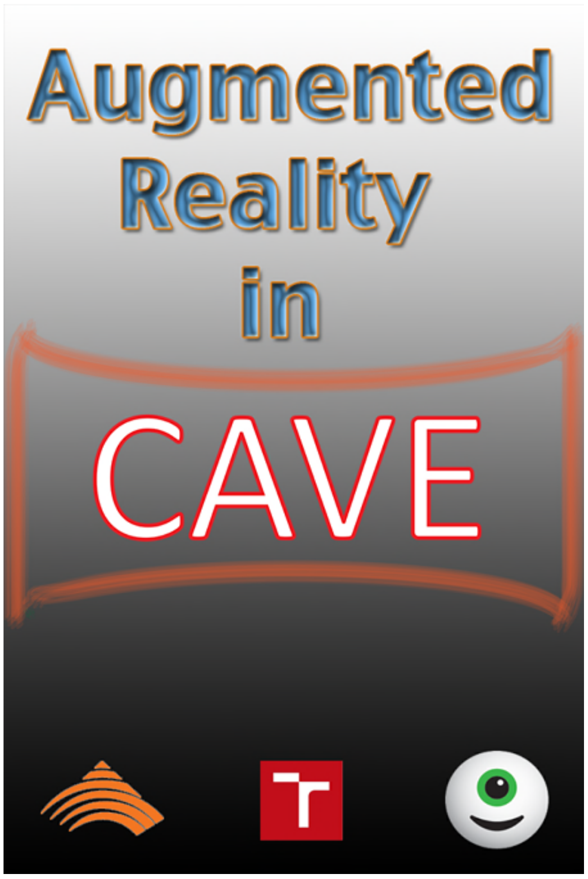


Figure 3: Splash screen for iPhone

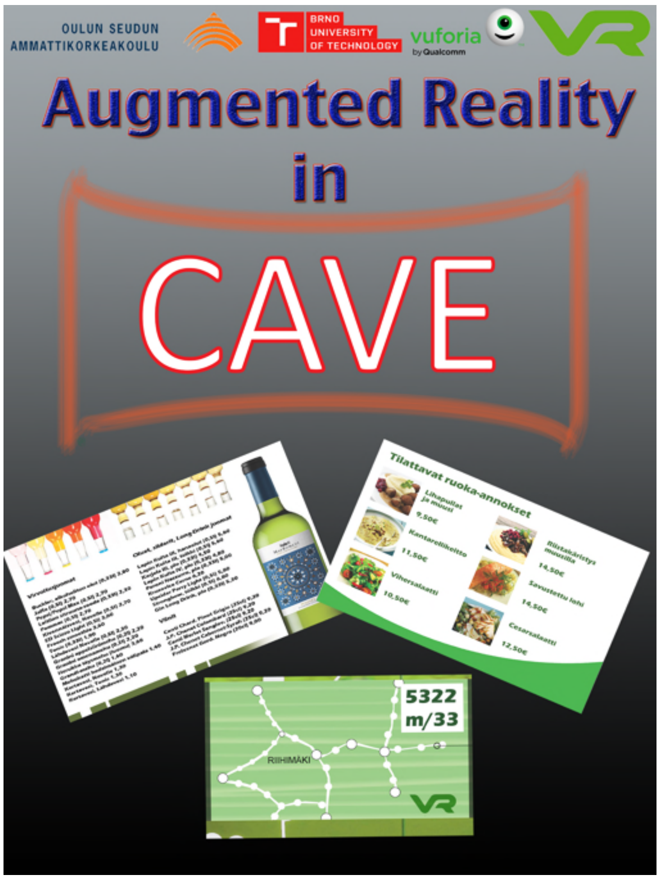

OULUN SEUDUN AMMATTIKORKEAKOULU
BRNO UNIVERSITY OF TECHNOLOGY
vuforia by Qualcomm
VR
Augmented Reality in
CAVE
Tilattavat ruoka-annokset
5322 m/33
RIIHIMÄKI

Figure 4: Portrait oriented splash screen for iPad

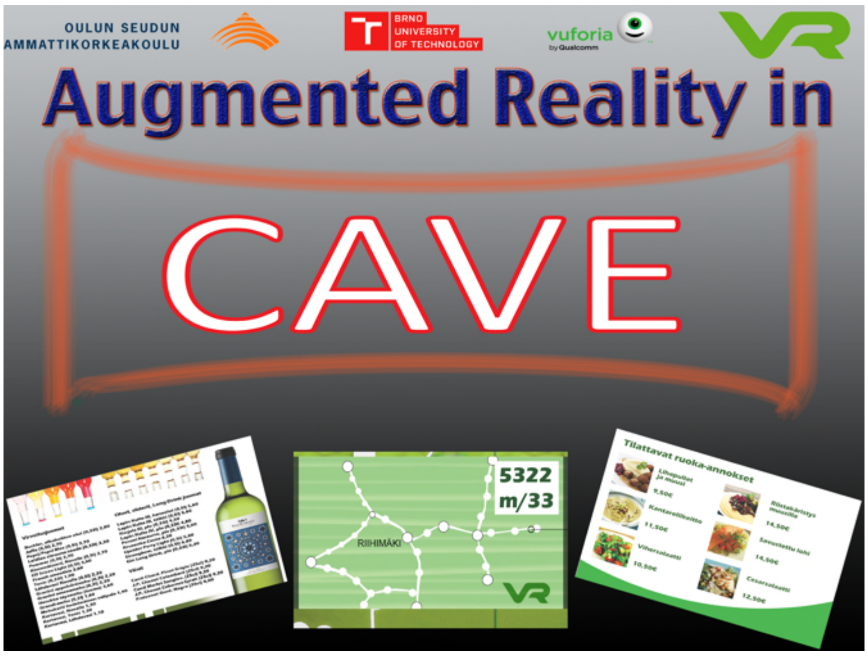

OULUN SEUDUN AMMATTIKORKEAKOULU
BRNO UNIVERSITY OF TECHNOLOGY
vuforia by Qualcomm
VR
Augmented Reality in
CAVE
Tilattavat ruoka-annokset
5322 m/33
RIIHIMÄKI

Figure 5: Landscape oriented splash screen for iPad

Appendix 4. Content of DVD

Application can be run only on iOS devices and it has to built and copied to the device through XCode IDE with the use of Apple Developer Licence. Because of the Apple policies, other way is not possible. Application cannot run in simulator, because it needs the device camera as completely essential thing. To provide better imagination about the demonstration application and CAVE in action, there is one complex demonstration video included also some other raw videos from the testing. Also some screenshots are included.

/Application - folder with the source codes

/Application/source/ARinCAVE/ARinCAVE.xcodeproj – project file to XCode, which enables to build the application

/Markers for test purposes – folder with the .png versions of the markers imported into the application, which can be used for test purposes

/Michal Kolcarek Diploma Thesis.pdf – this full text of the thesis

/Michal Kolcarek Diploma Thesis.docx – this full text of the thesis, which can be edited

/README – readme file for building and running the application, which contains some essential information

/poster.pdf – demonstration poster

/video.mp4 – demonstration video for the thesis

/Visual Demonstration – folder with some raw screenshots, videos and photos of using the application in CAVE and from experimenting with the markers