Augmenting Ubiquitous City Elements for Mobile City-Wide AR Games

Milan Bombsch

Bachelor Thesis
September 2014

Supervisor: Fabio Zünd, Dr. Marcel Lancelle
Professor: Prof. Dr. Markus Gross

ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

cgl
Computer Graphics Laboratory ETH Zurich

Abstract

The goal of the thesis is to demonstrate the possibilities of citywide Augmented Reality (AR) games when using ubiquitous city elements as playing locations. This category of games is really new and has great potential for taking mobile gaming to a new level. The task was to combine an AR experience with the aspect of exploring/walking around a city. This was done by implementing a trading game. The game can be played on mobile devices (Android and iOS) all around a city. We used newspaper boxes as playing locations and each location has an unique behaviour which forces the player to visit multiple locations. As there are very few similar games the feedback from the user study was very important. Would people enjoy playing the game when they are forced to walk/travel around and not be able to play the game at a stationary location as is the fact with nearly all mobile games so far. An overall positive feedback showed that City-Wide Augmented Reality Games can provide an unique and entertaining gaming experience.

Zusammenfassung

Diese Arbeit zeigt die Möglichkeiten von City-Wide Augmented Reality Spielen, wenn stark verbreitete Stadtelemente als Orte zum spielen benutzt werden. Diese Kategorie von Spielen ist sehr neu und eröffnet ganz neue Möglichkeiten bei tragbaren Geräten wie Handys oder Tablet Computern. Die Aufgabe hier bestand darin Augmented Reality, das heisst virtuelle 3D Objekte korrekt in das Kamerabild einzublenden, mit dem Erkunden der Stadt zu kombinieren. Dies wurde am Beispiel eines Handelsspieles gemacht. Das Spiel kann auf tragbaren Geräten, welche Android oder iOS als Betriebssystem benutzen, in der ganzen Stadt gespielt werden. Wir benutzten Zeitungsboxen als Orte für das Spiel, welche im Spiel alle verschiedene Eigenschaften haben und so den Spieler zum herumlaufen oder fahren zwingen. Da es noch sehr wenige ähnliche Spiele gibt, war das Feedback der Testpersonen sehr wichtig. Macht es den Leuten Spass ein Spiel zu spielen, das sie zwingt herumzulaufen und es ihnen nicht erlaubt wie bei herkömmlichen Handyspielen das Spiel an einem Ort zu spielen? Ein durchwegs positives Feedback zeigte, dass City-Wide Augmented Reality Spiele eine einmalige und unterhaltsame Erfahrung bieten können.

ETH zürich

Computer Graphics Laboratory ETH Zurich

Bachelor Thesis Project Proposal

Augmenting Ubiquitous City Elements for Mobile City-Wide AR Games

Milan Bombsch

February 2014

Description

Within the Future Internet Content initiative, the AR team at CGL is targeting a pervasive games platform that blends multiplayer mobile gaming with 3D virtual environments to create new immersive worlds. This project aims at tracking visually prominent ubiquitous objects that appear throughout the city, such as newspaper boxes, ticket vending machines, or city maps, and integrating them into city-wide Augmented Reality games for mobile devices.

Expected Results

Along with GPS information the tracked objects facilitate the localization and pose estimation of the mobile device and enable augmenting the specific location with game content. The developed tracking and augmenting framework should be demonstrated with a simple game. Various games should be considered, e.g. taking care of your plants, playing pinball or pong at the current location.

Schedule

Task	Due date
Launch	**03-Mar-2014**
Literature review, decide which algorithms, methods and toolboxes to use	**17-Mar-2014**
General system description and requirements defined	**31-Mar-2014**
First working (alpha) game prototype for evaluation	**02-Jun-2014**
First complete thesis draft	**19-Aug-2014**
Final game and hand in thesis	**03-Sep-2014**
Thesis presentation	**TBA**
Hand in printed thesis and CD	**TBA**

Required Skills

- Good programming knowledge in C#, Java, JavaScript, or C++
- Basic knowledge in Computer Graphics
- Independent worker
- Experience in game programming is a plus!
- Experience in programming for Unity3D, Android, or iOS is a plus!

Grading

The grade of the thesis is based on the performance demonstrated throughout the thesis work, the written document, as well as the final presentation. The grading scheme is as follows:

Grade	Requirements
6.0	Extraordinary quality, the results are much higher than expected.
5.5	Thesis results are very good; student expanded on the original theme.
5.0	Thesis meets expectations.
4.5	Thesis partially meets expectations and has minor deficits.
4.0	Thesis meets minimum quality requirements; but has deficits and is below expectations.

Remarks

The thesis is overseen by Prof. Markus Gross and supervised by Fabio Zünd (fzuend@inf.ethz.ch) and Marcel Lancelle (marcell@inf.ethz.ch), Institute of Visual Computing.

1. Acknowledgement

Fabio Zünd did a great job in supervising my bachelor thesis. I would like to thank him for all the effort he put in on his part, he really showed me how to tackle such a project and which steps are most important to concentrate on.

I also thank Dr. Marcel Lancelle who implemented the Leaderboard [Lea14] and helped me making the right decisions during this project.

Alessia Marra created the graphic assets for the final version. They look amazing. Thank you!

Isabelle Roesch and Tizian Zeltner did a similar Bachelor Thesis at the same time and I would like to thank them for a lot of helpful discussions on the topic of city-wide AR games.

And of course I thank all the people who participated in the play-testing and gave me feedback on the work.

Last but not least I would like to thank the ones who proofread this thesis.

1. Acknowledgement

Contents

List of Figures

List of Tables

1

Introduction

Now a days almost everybody has a smartphone or tablet and has it with him all day. These devices typically have a big touchscreen, are equipped with a camera, a location sensor (such as GPS) and an active internet connection. Combining these features for mobile games can lead to completely new gaming experiences. City-Wide Augmented Reality Games. These games use the camera of the device to augment the real world with virtual 3D objects, the location info to adapt the game to the users current location and the internet connection to allow interactions between different players. As part of the Future Internet Content initiative [FIC14] , the city-wide games team at CGL [Cit14] is targeting a pervasive games platform that blends multiplayer mobile gaming with virtual 3D environments to create new immersive worlds. This bachelor thesis contributes to this platform by demonstrating the possibilities of citywide AR games using ubiquitous city elements.

1.1. Problem Statement

This project uses visually prominent wide spread city elements as locations to play a mobile game. The game should also make use of the users location and the camera image to augment the users world with virtual 3D objects the user can interact with. Possible city elements are newspaper boxes, ticket vending machines and city maps. Several techniques to track the real world objects should be considered for the AR experience.

1.2. Overview

The thesis is structured in 3 main parts. In the first part the game is set into context and described in detail. What game was implemented and why? Followed by a description of the implementation. What tools were used and what kind of problems occurred during the development? Rounding it up are the results of our work. What did the users say in the Play-Testing session and what are possible improvements for future work? The appendix includes a guide for new users of the mobile app GnomeTrader, which was developed during this project, and the questionnaire people were given after one week of testing the app.

2

Related Work

City-wide AR games are a relatively new topic, because the devices used to play such games need to be powerful enough to process the data for the AR part on the fly. They also need to be aware of their current location and are ideally connected to the internet.

One already existing example is 'Ingress' a game developed by Google [Ing14]. The player has to walk around the city to collect points with which he later can capture real world objects for his team and fight others. The AR part in this game is using the users GPS location to display a map of his current location. This map is then augmented with the game elements. We wanted to provide a better AR experience and use the devices camera to actually put 3D objects into the real world.

Junaio [Jun14] is AR browser for mobile devices from Metaio Inc. It provides a platform for developers to create AR games and general AR content for mobile devices. They use their own framework (Metaio SDK). The Junaio browser is fine for small applets, but in our case we wanted to provide a full game. Building an app on our own gave us the possibility to fine-tune each step.

In [LDNB13] Gun A. Lee et al. use the Mobile Outdoor AR Framework [mob14] to provide the user with a virtual reality tour through the antarctica. For this they use the devices orientation and location. They replaced the cameras image with images from antarctica, so people got the feeling they were really there. For our program we wanted an AR experience which adapts to the users current location. It should not take the user to another place but rather show him new things at his current location.

3

The Game

As described in Chapter 1 the game should be playable all around the city and provide the user with an Augmented Reality experience at different locations.

3.1. Brainstorming Ideas

A brainstorming session led to a lot of different game ideas, but of course some of them were not feasible. Three ideas are listed here:

- Taking care of your plants: The user can plant seeds all around the city. He needs to take care of them in order to let them grow and prosper. Users can see the plants of other users at their correct GPS location and interact with them.
  The problem was that the GPS of mobile devices is not accurate enough to be able to put a plant into the camera pictures of multiple users at the same place. The plants would be seen at slightly different positions by different users.
- Collecting coins from houses: The idea is that every window of a house transforms into a coin. Players have to collect these coins. If a player collects the coins from a house they are inaccessible for other players for a specific amount of time. The aim is to collect as many coins as possible.
  The problem was that detecting arbitrary windows on house walls is very difficult and definitely not realizable in the scope of a bachelor thesis.
- GnomeTrader: Each newspaper box (there are plenty of them in Switzerland) houses a little gnome who either sells or buys a specific resource (such as nuts or peas). The aim of the game is to make as much money as possible by trading with gnomes.

The GnomeTrader was the one we implemented.

3.2. Deciding on and developing of the final idea

GnomeTrader seemed to be the most portable idea. The idea started as a simple game in which people should be able to trade resources (e.g. wheat and wood or nuts and peas) at variable prices at different trading locations (the newspaper boxes). At the start there was no limit to the amount of resources a user could carry. But it makes the game much more interesting if users are limited in the amount of resources they can carry, which we implemented as a bagSize of the user (increasing in steps of 10 and starting at 10). After the play-testing phase players even said that a restriction on the amount players can buy at once would improve the gameplay even more. After a working basic version we defined the buyer- and seller-gnomes more clearly. Buyers should consume their resource, meaning from time to time resources are removed from their stack. Sellers act the opposite way. They produce resources from time to time, thus their stack gets refilled if users buy resources from them. Towards the end of the development an experience and level calculation for the users was added, in order to keep them motivated in the competition to beat others on the newly introduced leader board. Another thing which was added to keep players motivated was the daily-reward. Every time a player visits a trading location for the first time that day, he receives free resources. The daily-reward consists of 10% of the users bagSize in the resource the gnome at the visited trading location does not trade with. In the first version the game was only playable at newspaper boxes of the 20-Minuten newspaper [20M14]. The game was at this point called 20MinTrader and only later renamed to GnomeTrader. In the final version the game is also playable at the Blick-Am-Abend [Bli14] newspaper boxes.

3.3. The Story

In each newspaper box there lives a little gnome. Some gnomes want to get nuts or peas to eat, they are called Family Gnomes. Others want to get rid of their nuts or peas, because they produce them. These are called Gardener Gnomes. Each newspaper box houses either a Family Gnome (who buys peas or nuts) or a Gardener Gnome (who sells peas or nuts). In order to get the gnomes ecosystem to work, the player needs to buy resources from the Gardener Gnomes and sell those resources to the Family Gnomes. But the player needs to be aware of the price the gnomes demand/pay. These prices depend on the amount of resources a gnome has. The more resources he has the less he will pay/demand from the player. If the player makes wise decisions with which gnome he trades he will make faster progress and earn more money to buy a bigger bag and trade more efficiently.

3.4. How to play

A simple starters guide can be found in Appendix A.

3.4.1. Start playing

To get a first impression on how the game works, the player walks around the city and tries to find a trading location which sells resources. He then buys as much resources as he can and tries to find another trading location where he can sell them at a better price than he initially spent. By continuing to do so the player gets more and more money. But in order to be really successful he needs to take a deeper look at how the in-game money and resource system works.

3.4.2. In-Game money and resources

The in-game money is called gold. Each player starts with 100 gold and no resources. With this gold he can buy resources at the trading locations or he can increase the size of his resource-bag, which is done in the Menu at the BagView (see Figure 4.1). As the user can trade more efficiently with a bigger resource-bag, the price of the next bigger one increases each time. Each bag costs 50 gold more than the next smaller one, starting at 50 for a bag of size 20.

Having enough gold for the next bigger resource-bag is really important for a fast progress. Let us have a look at the resource system and how it should be used to earn a maximal amount of gold.
There are 2 resources: Nuts and Peas. They are treated equally, meaning they do not have any special attributes. For both resources the prices are calculated the same way: (3.1) and (3.2). Figure 3.1 shows the prices graphically.

$$BuyPrice = 2.506 * 10^{-7} * amount^2 - 0.001252 * amount + 2.001 \quad (3.1)$$

$$SellPrice = BuyPrice + 0.1 \quad (3.2)$$

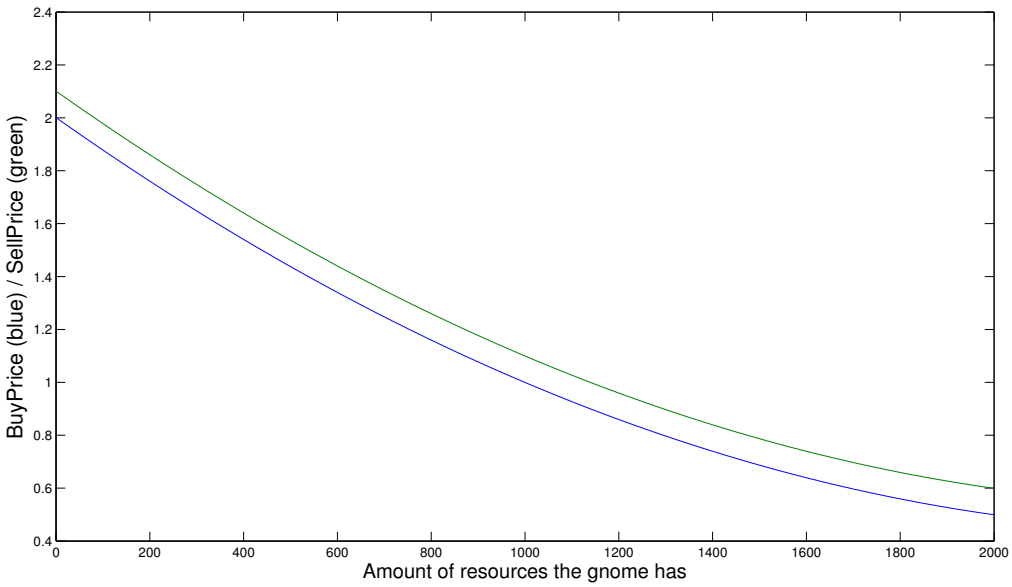


Figure 3.1.: *The BuyPrice (blue) and the SellPrice (green) for resources the user has to pay.*

The *BuyPrice* is the price for which the user can buy resources from the gnome. The *SellPrice* is the price for which the user can sell resources to the gnome. Both prices are in gold per resource and are between 0.43 and 2 for the *BuyPrice* and 0.53 and 2.1 respectively for the *SellPrice*.
Having more than one resource makes the game much more interesting, because now there are 4 different kind of gnomes (nut-selling, nut-buying, pea-selling, pea-buying). If a player has his bag full of peas, he needs to find a gnome who buys peas. To avoid such a situation it is best to always have peas and nuts in the resource-bag. On the other hand to make a maximal amount of money, the player should always either buy as much as he can or sell as much as he can. Trying to find the right balance makes the game challenging. As described in Section 3.2 the player does not only earn resources through buying them, but also through visiting new boxes and receiving the daily-reward.

3.4.3. Advanced playing

Walking around town and buying/selling resources whenever one can, is a good starting point to play the game. Since gnomes selling resources tend to have more resources (since they are producing them) than the ones buying them (they consume them) the player gains money most of the time. But sometimes, when a trading location, selling resources, is in a frequently visited place, the price for buying them increases and the player loses money if he has to resell them at a lower price. To give advanced players a place to look for good prices on the 'market' we integrated an overview over all available trading locations and their prices (Figure 3.2 and Figure 3.3).

Using these tables players can have a look at where the best prices are and travel there to get the maximum amount of money for their resources. The 2 views are websites and their links are available within the GnomeTrader app.
Generating enough earnings is not everything a good player needs to consider. There is also a leader board, on which players can compete against each other. The score used for the leader board entry is the experience of a player (see how it is calculated in Section 3.5). Since the calculation of the experience includes the size of the resource-bag, the amount of gold transferred and the amount of trading locations visited, the player should also try to visit as many trading locations as possible.

3.5. Score and Level progress

To give the player a view of their progress in game, we introduced levels and with it the experience measure. A player starts at level 1 with 0 experience and progresses from then on with no level cap. But of course progressing from level 1 to level 2 is much easier than from level 10 to level 11. Players need more experience points in order to progress at higher levels. The experience is a measure containing the size of the resource-bag (bs), the amount of gold the player transferred with his trades (gt) and the number of times he visited a trading location (tv). The formula for calculating the experience is given in (3.3).

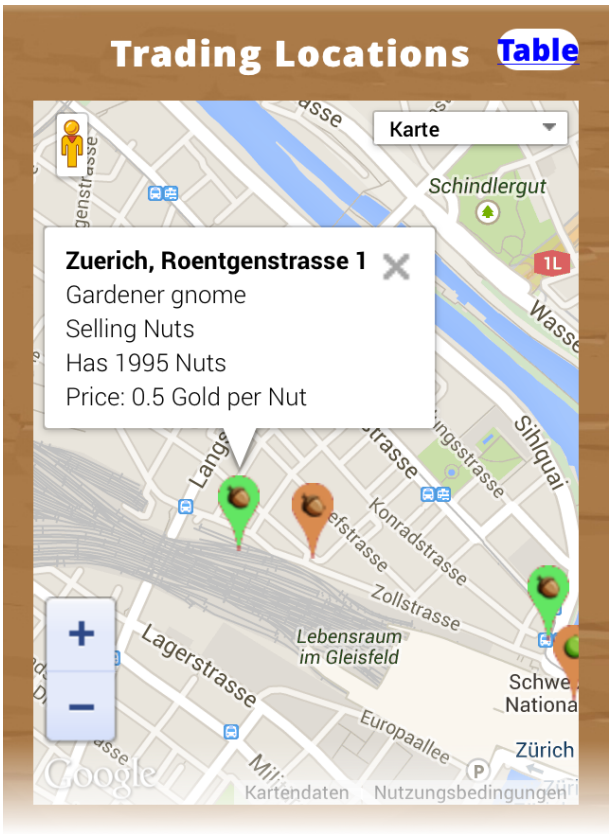


Figure 3.2.: *An overview of all available trading locations as a table*

Figure 3.3.: *An overview of all available trading locations as a map*

$$Experience = 20 * (bs - 10) + 2 * gt + 10 * tv \tag{3.3}$$

The numbers in (3.3) were found empirically. The size of the resource-bag has the biggest impact on the experience. The reason behind this is that for buying a bigger resource-bag the user needs to get more gold and to get more gold he needs to trade a lot, which is the aim of the game.
The level of a player can be directly computed with his experience (3.4). A graphical representation can be found in Figure 3.4. As one can see it takes longer and longer to reach the next level, which is a typical aspect in game design.

$$Level = \left\lfloor \left(\frac{Experience + 57.67}{57.67} \right)^{\frac{1}{2.415}} \right\rfloor \tag{3.4}$$

Again the numbers in (3.4) were found empirically.
The level and experience determine the rank of the player on the leader board.

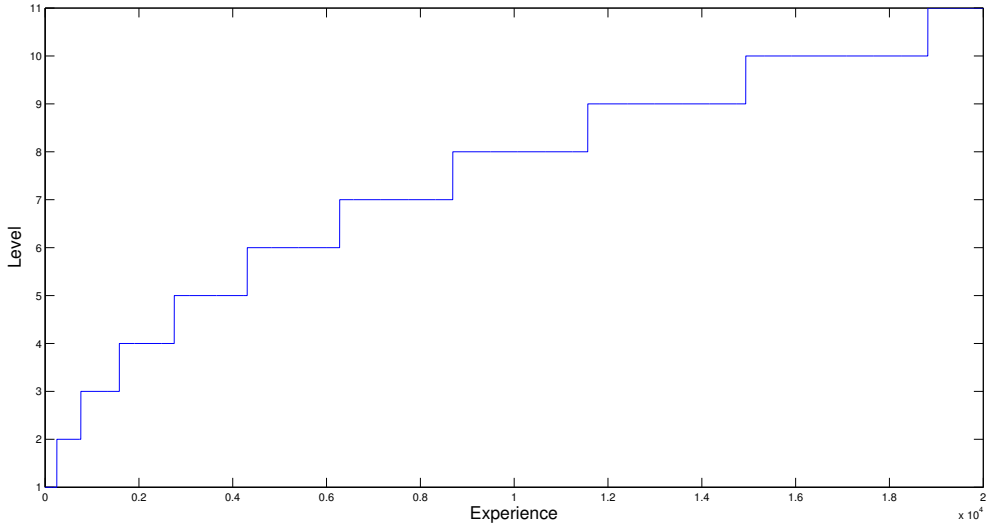


Figure 3.4.: *Levels plotted against the experience needed to reach them.*

3.6. Graphics

In order to refine the tracking we used the 'Toon Baby Dragon' by Remi Storms (available on the Unity Asset Store free of charge) as augmented 3D object and for the UI a simple black texture. The dragon was really good for this task, because small changes in the rotation or the size could easily be detected. Some impressions of the early GUI can be found in Figure 3.5 and Figure 3.6.

In Version 0.4 of the GnomeTrader the GUI was redesigned and the 'Toon Baby Dragon' was replaced by the originally intended gnome. For this version Alessia Marra created professional 3D and 2D assets. We put the gnome behind a door to improve the feeling that the gnome actually lives in the newspaper box. The board next to the gnome was put on the back of the door, so the controls to trade are only visible when the door is open. For the 2D GUI we split the menu into different views, so everything looks more organized. The new GUI can be seen in the GUI overview in Figure 4.1. We needed to keep in mind that the GUI should fit smartphones as well as tablets. For the 3D part this is easy to realize, since the size of the gnome does not depend on the device but on the newspaper box, which is always the same size. The 2D GUI is scaled up for tablets, which works fine.

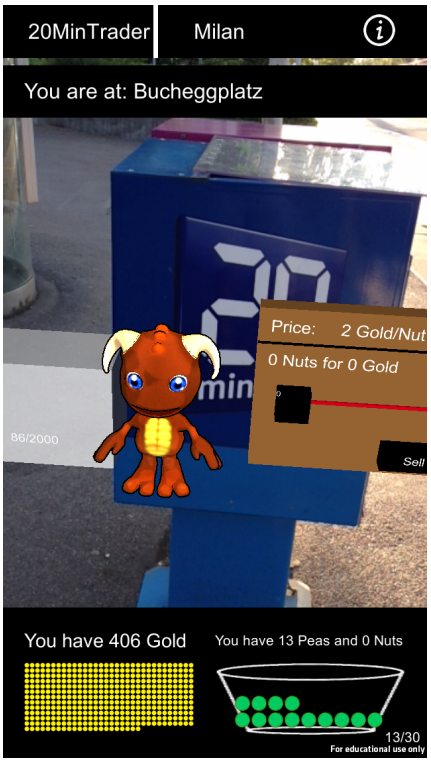


Figure 3.5.: *An early version of the Main View with a dragon as dummy gnome.*

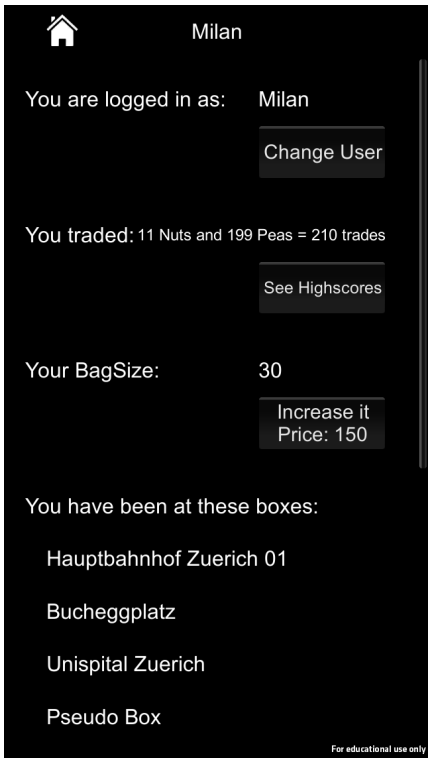


Figure 3.6.: *In previous versions the Menu was one single view.*

4

Implementation

This chapter describes how the game idea is realized and which challenges we were facing during the development. It is split in two parts: Client and Server. The client is the mobile application on the users device, the server is a central linux computer which handles the database and synchronizes the game data between the clients.

4.1. Client

The client in our case is the application on the mobile device. This mobile device can be a smartphone or a table running iOS or Android. For the app to properly work, the device needs to have GPS, mobile data and a back facing camera. An overview of the GUI is given in Figure 4.1. The client is programmed with Unity. We will not talk too much about the code here, because it is well documented itself (as XML comments inside the C# files). The whole Unity project can be found on the complementary CD to this thesis.

4.1.1. Versions (Client)

The client is programmed with Unity using C# as scripting language. The following tools and libraries were used for developing the client app.

Development Environment Unity Pro 4.5.2f1 (Education version)

Augmented Reality Library Qualcomm Vuforia 3.0.6 (Unity Extension)

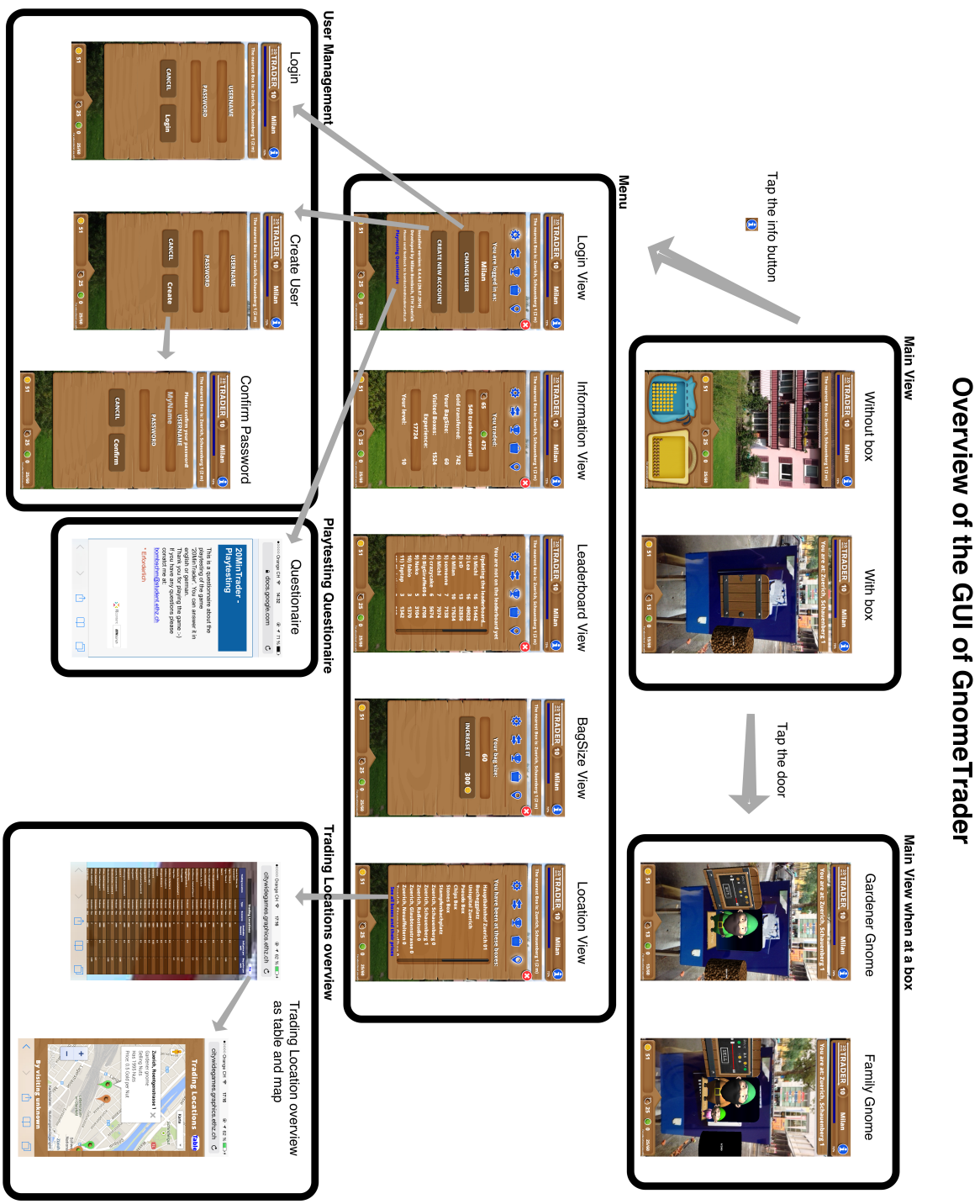


Figure 4.1.: *Overview of the clients GUI. The grey arrows show were the clicks on the according buttons leads you.*

JSON Library 'JSON Object' by Defective Studios from the Unity Asset Store (free)

The app is tested and works for Android 4.4.4 and iOS 7.1.2

4.1.2. Tracking real world objects and augmenting them

Augmented Reality (AR) is nothing new, but with mobile devices getting more and more powerful AR can provide the user with a completely new experience. In case of this project we concentrated on combining AR with mobile games. To be able to produce a functional game within the scope of this thesis we could not write our own AR library but had to rely on others. There are multiple libraries which provide a good basis for tracking and augmenting real world objects with virtual ones on mobile devices. The most popular being Qualcomm Vuforia, Metaio and ARToolkit (by Inglobe Technologies). Qualcomm Vuforia is the only one completely free of charge. It also provides a very good Unity integration and runs on Android and iOS. We chose Vuforia because it provides a good basic tracking and we could easily augment predefined markers with 3D objects.

4.1.2.1. Vuforia

With Vuforia one can either track a marker, which is a 2D image, or a 3D object like a cube or a cylinder. Since we intended to track newspaper boxes in their entire, a cube tracker would have been the most fitting. But after testing and comparing the results from a 2D tracker image of the newspapers logo and a 3D tracker of the whole box, we decided to stick with the 2D tracker, since it lead to much better results. The fact that the newspaper boxes look the same from both sides makes it hard for the algorithm to distinguish them and augment the 3D content correctly. The missing texturing is also a problem since the algorithm needs a lot of edges and corners to correctly place the virtual objects. Looking at a plain blue (in case of the 20 Minuten boxes) part of the picture the algorithm cannot decide in which direction the camera is currently pointing. Let us have a closer look at the algorithm Vuforia implements.
The first thing a developer must do is upload an image he wants to track to the Vuforia website, for preprocessing. This process searches for feature points in the picture. Vuforia transforms the picture into a black and white one and afterwards searches for corners. A good marker image should therefore also be a good marker if you use a black and white filter on it, thus it should have high contrasts. But having lots of corners is even more important. If Vuforia cannot find any corners, say in a picture of a circle, it is impossible for the algorithm to track the object. Thankfully the logos on the newspaper boxes mostly consist of text, which naturally has lots of corners. We therefore decided to use the logos at the side of the newspaper boxes as markers for our augmentation. But we still had problems detecting the box. In the beginning the rate of detecting a box successfully when standing in front of it was 1 to 10. This was due to different lighting conditions, reflections on the box and stickers people had put on it. The solution was to add many different pictures of the boxes as markers and tell Vuforia to augment one at a time. With this approach we were even able to detect boxes which were covered by poles (See Figure 4.2).

Figure 4.2.: *A Blick-Am-Abend Box surrounded by poles augmented with the gnomes door and his container*

More information on how Vuforia tracks an image can be found at: `https://developer.vuforia.com/resources/dev-guide/natural-features-and-rating`

Augmenting 3D objects on top of images is very easy with Vuforia and Unity. One can simply add the 3D objects to the image target and they appear when the image target is recognized by the algorithm. Since we chose to have many different image targets of the same real object, they all run the same script (`GnomeTrackableEventHandler.cs`) so that their behaviour towards the user is exactly the same.

4.1.3. Database Communication

The client is mainly a viewer. It displays the data fetched from a database to the user (done in `DatabaseCommunicationScript.cs`). Everything from the user account informations to the trading locations and their properties is stored in a database (see Section 4.2.2). This database is accessed by the client through its Web API (see Section 4.2.3). The client fetches the user data of the currently logged in account and information about all available trading locations. It permanently displays the users account information and when the player arrives at a trading location and points his device towards a newspaper box, the corresponding information about this trading location are used to calculate the gnomes inventory and the price he demands/pays. If a user visits a trading location or buys/sells resources the client tells the database to perform this action and the database returns the new values which are then displayed to the user. If anything fails during the process of communicating with the database or the client has outdated

data the database will simply ignore the action and the client needs to update his data and retry the action.

4.2. Server

The server is a remote linux computer which stores the database and runs several scripts. It also acts as a web-server, so users can browse websites containing information about the Gnome-Trader game. The behaviour of the server is described in detail in the next chapters.

4.2.1. Versions (Server)

The following programming languages were used in the development of the server: MySQL, PHP, HTML, Java Script
The following tools and libraries were used for developing the server part.

OS Ubuntu 13.10.1

Database MySQL 5.5.37

Web server Apache 2.0 with PHP 5.5.3

Leaderboard FI-Content Leaderboard component [Lea14]

Maps API Google Maps Javascript API [Map14]

Transportation API Swiss public transport API [Tra14]

4.2.2. The Database

At the heart of the game is a database of all user accounts, trading locations, resources and how they are connected to each other. A diagram of the database model can be found in Figure 4.3

The three main tables are Users, TradingLocations and Resources. All the others connect those tables and state the properties for their connection. For example the table UserHasResource stores how much the user with UserID has of the resource with ResourceID and the time when this amount changed last. No entry for user with UserID indicates that the user does not have any resources. One table stands out. This is the UserHasAchievement table. We planned on integrating achievements into the game, but they couldn't be implemented so far. This remains for future development.
Some of the tables have triggers. Let us have a closer look at them. First there is the trigger in the Users table. It changes the ModifiedBagSize column every time the BagSize column is updated, this happens when the user buys a bigger bag. It is implemented as a Before Update Trigger. This trigger was introduced in order to understand if the game works as intended or if users can buy new bags far too often or almost never. The triggers in UserHasResource and TradingLocationHasResource change the Modified entry every time the entry changes.

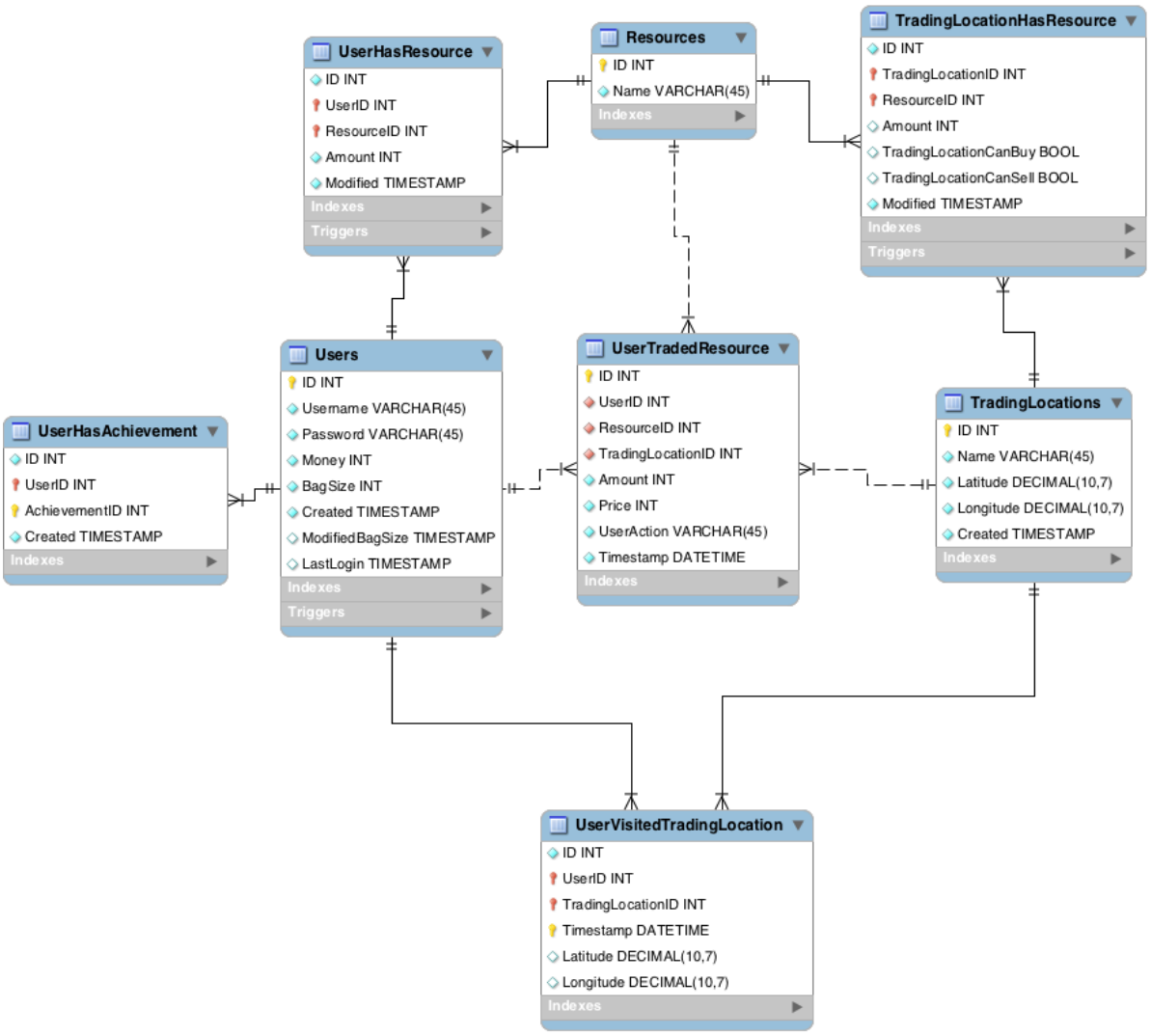


Figure 4.3.: *The model of the database.*
(yellow or multiple red) key = primary key, turquoise diamond = not null, white diamond = nothing, red indicates a foreign key

Since it is nearly always the Amount entry that is responsible for a change, we can get a better overview of the trading system by checking this column regularly.
To get a view of how the database evolves in size over time, we show in Table 4.1 the number of entries per table. These numbers were taken after running the game for about 2 months.

Table	Number of Entries
Users	50
TradingLocations	218
Resources	2
UserHasResource	67
TradingLocationHasResource	217
UserTradedResource	3533
UserVisitedTradingLocation	6499
UserHasAchievement	0

Table 4.1.: *Number of entries per database table. This gives an overview of how the system evolves over time. The data was taken after 2 months of running the system.*

4.2.3. The Database API

The API for the client to access the database is defined in `api.php`. All the corresponding logic is implemented in a separate file called `logic.php`. For a client to make a call to the database, he needs to call the web address of the `api.php` file (in our case `citywidegames.graphics.ethz.ch/trader/api.php`) and add the function name and the input arguments as parameters into the url. A skeleton for this is

```
http://yourURL.com/api.php&action=functnName&param1=xx&param2=yy
```

The return value is either an error string in the form of "ERROR: description here" or a JSON object containing the information of a user, resources or trading locations.

The database API defines the following functions:

```
User createUser (username, password, version)

User loginUser (username, password, version)

User increaseBagSize (userID, newBagSize, totalPrice, password,
    version)

User addVisited (userID, tradingLocationID, password, version)

List<Resources> getResources (version)

List<TradingLocation> getTradingLocations (version)

(User, TradingLocation) buyResource (userID, tradingLocationID,
    resourceID, amount, totalPrice, password, version)

(User, TradingLocation) sellResource (userID, tradingLocationID,
    resourceID, amount, totalPrice, password, version)
```

Notice that all functions have an input parameter called 'version'. This is the version of the client app making the request. This was introduced to avoid erroneous clients from taking part in the game. For example client version 0.3.2 has a big bug enabling users to buy resources for no cost, the bug was fixed fixed in version 0.3.3 and the API from then on reject calls from clients with a version lower than 0.3.3.
Most of the functions also have a parameter called 'password'. This is the MD5 hash of the password to the corresponding username or userID, ensuring that only authorized users can interact with the system.

4.2.4. Gnomes

The gnomes do not just sell and buy resources, but they also consume and produce them. Family gnomes buy resources from the user and consume them. Gardener gnomes sell resources to the user and produce them. If the gnomes only sold or bought resources, at some point the sellers would not have any more resources to sell and the buyers container (the box the gnomes use to store their resources) would be full. To enable this behaviour of the gnomes we set up a php file (`produceAndConsumeResources.php`) which when called reduces the amount of resources of the family gnomes and adds resources to the containers of the gardener gnomes. To be able to let the system run by itself we setup a cronjob which calls this php file once per hour.

4.2.5. Bots

We introduced bots to see how the system behaves, before testing it with real players. The behaviour of a bot is defined in the following way:

1. Check if time is between 8:00 and 20:00
2. Draw a random number between 1 and 130 (later changed to 260). If it is a 1 continue, otherwise abort
3. Choose a random trading location
4. Buy/Sell as much resources as possible
5. If enough money to buy new resource-bag, buy it

This procedure is called by a cronjob once every minute.
The first step is to simulate real players more accurately, who typically play at day times. In the second step we introduced some randomness. Real players will not play always at the same time. Choosing a number between 1 and 130 lets bots trade about every 2 hours. As real players joined the system we slowed down the bots (trading every 4 hours) to lower their impact on the game. After this step a bot chooses a random trading location and buys or sells as much as he can. If he has enough money he can increase the size of his resource-bag. When first testing the system with bots, they made reasonable progress. Not too slow and not too fast, so the price calculation ((3.1) and (3.2)) and the level (3.4), experience (3.3) formulas were a good choice. For our testing we used 10 bots. A bot is defined as a user whose username starts with 'Bot'. Such accounts cannot be created inside the mobile app. Adding another bot is as easy as adding the necessary user entry in the database.

4.2.6. Generating new Trading Locations

One of the biggest problems while programming the game was: How could the game get to know all newspaper boxes in Zürich or even in Switzerland? We did not want to simply ask the newspaper companies for a list of GPS coordinates of their boxes, but rather choose an approach which could also work with any other widely spread marker. We started by logging the locations where players saw a newspaper box with the app. Every time the app recognizes a newspaper box (marker) it tells the database its current GPS coordinates. Resulting in a lot of points on a idealized 2D plane. Now we calculate the best fitting circle. We do this with a brute force approach since we do not have that many data points and our script runs on the server and not on the mobile devices. The algorithm works as follows:

1. Set a fixed radius
2. For each pair of points calculate the number of enclosed points for the two circles touching those points
3. Take the circle which covers the most points and return their average Latitude and Longitude

In our case the fixed radius is 20 meters, which is roughly half the distance between two newspaper boxes, which are most of the time at tram stations. For each pair of data points there are two circles with radius r which touch those points. If we consider all those circles, we considered all relevant circles. The reasoning behind this is that if we have a circle we can move it slightly so it touches two points and still covers the same amount of points. Running this

algorithm takes $O(N^3)$ time since there are N^2 pair of points which circles we need to test, to see how many points out of N they cover.
Now we run the algorithm on all points which are more than 20 meters away from an existing trading location and find the center of the circle covering the most points. But should we now just create a trading location at this place? Probably not. Otherwise we would have lots of faulty created trading locations, which are not actually there but only created because of deviation in the GPS of the mobile devices. To avoid this we set a threshold of 10 covered GPS points from at least 2 different users to actually create a new trading location at a certain point. After finding a point satisfying these conditions we create a new trading location there which houses the gnome who is missing the most in the area around the new box. We take from the four combinations (Family Gnome with nuts, Gardener Gnome with nuts, Family Gnome with peas, Gardener Gnome with peas) the one who is furthest away. The whole procedure is stored in a PHP file (`newLocations.php`) and called by a cronjob once per hour.
With this algorithm the system can build up a database of all trading locations (newspaper boxes) by itself and can easily be adapted to other objects/markers. During our beta testing we managed to create 218 trading locations (see Table 4.1).

4.2.7. Web content

The server also acts as a web server to display webpages. There are three webpages available: `showPrices.php`, `showPricesMap.html` and `showPlayers.php`. The `showPlayers.php` is only for internal debugging/surveillance purposes, showing the progress and state of all players in the system.
The two others (`showPrices.php` and `showPricesMap.html`) give the players an overview of the trading locations system. There they can see which trading locations are available and what they charge at the moment for buying/selling resources. The two websites are pictured in Figure 3.2 and Figure 3.3. The table view is a simple HTML table with the content from the database. For the map view we used the Google Maps Javascript API [Map14].

4.2.8. Leaderboard

The leaderboard was added to the game to keep players motivated and to introduce a competition between players. We used the leaderboard component of the Pervasive Games Platform from the FI-content [Lea14]. The players experience is used as the score entry for the leaderboard.

5

Play-testing

This chapter describes the play-testing we did near the end of the project.

5.1. Approach

We invited 10 people to play the game for one week and fill out a questionnaire at the end. Each participant received a cinema coupon after filling in the questionnaire. 8 people followed our request. The participants received version 0.4.2.10 of the GnomeTrader app for testing. For Android the `apk` file could be downloaded from our web server and for iOS we used Test Flight [Tes14] to distribute the `ipa` file.

5.2. Results

All testers could successfully install and play the game on their Android and iOS devices. We will now go through the different questions in the play-testing questionnaire. The whole questionnaire can be found in Appendix B.

Gender We had 5 female and 3 male testers.

Age They were aged between 17 and 26, which is the typical smartphone user. See Figure 5.1

Experience with computer games The experience was mixed, but as expected the men said to have a little bit more experience than the women. See Figure 5.2

Experience with AR games 2 players had some experience with AR games. The others

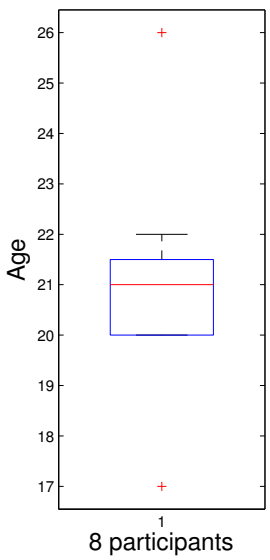


Figure 5.1.: *Testers age*

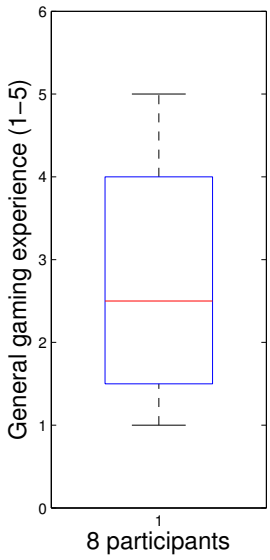


Figure 5.2.: *General Experience*

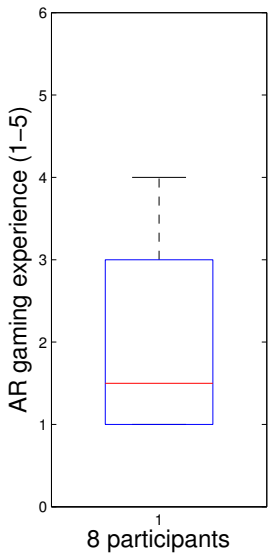


Figure 5.3.: *AR Experience*

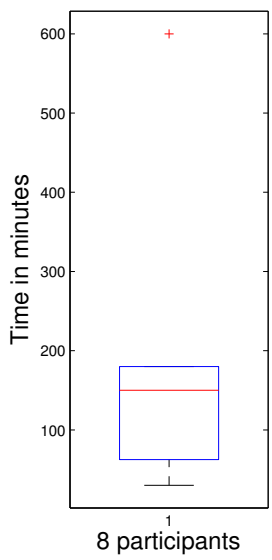


Figure 5.4.: *Time played*

had none. See Figure 5.3

Device 4 iPhones (from 4 to 5s) and 4 Android devices.

GPS Everybody enabled their GPS (In this version it was not mandatory)

Internet Connection Only one player mainly used Wi-Fi for connecting to the internet and mainly played at home in front of a print out of a newspaper box. All the others used their mobile internet connection.

Version All testers used version 0.4.2.10 of the game

Time played All players played the game for around 2 hours spread over the course of 1 week, which makes sense since playing the game at one trading location only takes about 5 minutes. See Figure 5.4

The controls The controls were rated good (nearly self-explaining (1 = self-explaining, 5 = confusing)) by all players. See Figure 5.5

Impression The first impression among all players was: confusing. This is probably due to the fact that this is an AR game and people are not that familiar with these kind of games. While playing the game most people said that they began to understand the mechanism and the aim of the game better and better, so it was still fun and challenging to play.

Frustration The main point mentioned here was the 3D slider which specifies the amount to sell or buy. We fixed this in version 0.4.2.12 by making the slider a lot slower. Another aspect was that people complained that they were not able to find boxes which buy/sell nuts/peas. This emphasizes the importance of the player having to travel to different locations, using the whole city. You should not be able to play the game at one single location.

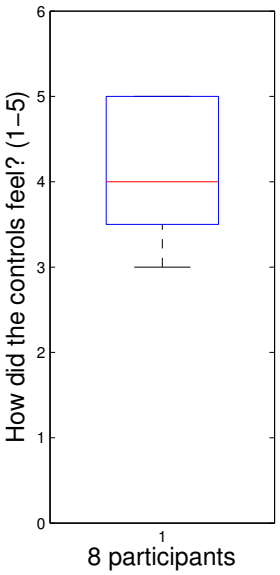


Figure 5.5.: *Rating of the controls*

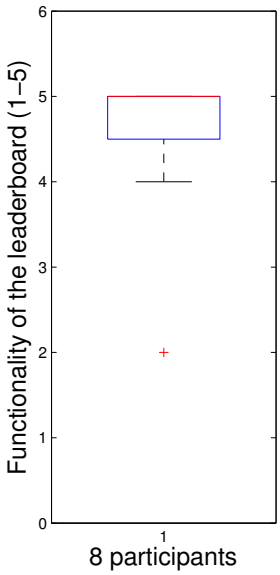


Figure 5.6.: *Rating of the leader board*

Did it drag on? The game gladly did not drag on after a week, but some testers mentioned that it probably would after another week because of missing changes in the game.

Leaderboard The leaderboard did its job, all agreed that it was easy to use. See Figure 5.6

Objective Was clear to all participants: Trade with the gnomes, progress by buying bigger resources-bags and try to be on top of the leaderboard.

Winning strategy The strategy which was used the most is visiting many boxes even if you cannot buy or sell, because those visits also count towards the experience (see (3.3)).

Loopholes The biggest one was definitely the inaccuracy of the devices GPS. Some even turned it off to be positioned completely offside. We fixed this by disabling the game when GPS is disabled, but the bad accuracy remains. People could switch between boxes by just staying in front of one location and with the fluctuation of the GPS signal changing boxes frequently. This problem is not fixed yet.

Play Matrix We could not evaluate the results from the play matrix since they were not saved correctly.

Something missing? The testers would have liked to be able to trade between users and missed some challenges or special boxes to keep the game interesting.

What to improve? Again trading between users was mentioned. The testers said that they would also slow down the process of generating new trading locations. This is fine since we reduced the amount of visits needed for creating a new trading location to get a good start in the beta test. After the testing phase we slowed down the generation process again.

Giving away Most people would give the app to a person who is around 20 years old and

spends a lot of time traveling with the public transport system.

All in all the play-testing was very successful and gave us great feedback of what to improve and change. We are also happy about the fact that some players kept playing the game in their spare time after the end of the play-testing phase.

6

Conclusion and Outlook

6.1. Conclusion

We successfully implemented a city-wide augmented reality game using ubiquitous city elements as playing locations. The dynamics introduced with the trading game were very effective and people had fun walking and driving around the city playing the game. We were able to refine the tracking and the interaction with the 3D objects to provide a good gaming experience. The fact that players see the effects of other players buying and selling resources to the gnomes adds a multiplayer aspect to the game. All in all we can say that city-wide augmented reality games have great potential if they can be played at many locations which each have different properties.

6.2. Future Work

There are lots of possible improvements for the game. On one hand there are still some inconveniences with the current version and on the other hand there are lots of improvements to be done to increase the entertainment value of the game.

6.2.1. Remaining Problems

The biggest problem is still the bad accuracy of the GPS sensors in the mobile devices. To reduce the impact of one measure, one could again increase the number of points needed for a new trading location, but this would also need more players to work. The GPS location used

to determine the nearest trading location, could also take an average of the last 10 measures to avoid fast switching between boxes.
The accuracy of the augmentation should also be improved. More often than not the virtual 3D objects are slightly rotated and incorrectly align with the newspaper box. To improve on this one could take the gravity measure of the devices gyroscope (most of them have one) and align the 3D objects to it. Since the newspaper boxes are always align to gravity this would probably improve the correctness of the augmentation.
The starting process of the app should be improved significantly since the current version (0.4.4.0) takes about 10 seconds to load due to Vuforia loading many images to track (all the newspaper logos from different angles and lighting conditions).
The frame rate of the running app should also be improved for older devices (very bad performance on iPhone 4 with iOS 7).

6.2.2. Future Features

A feature we wanted to implement was achievements. This would help players who play a lot stay motivated. Those could look something like this: Buy 1000 peas in total, visit 1000 boxes, visit all boxes in Bern, spend all your gold, etc.
Another feature is user trading, which was requested by a lot of testers. The idea is that users who are next to each other in real life can exchange resources and gold. If one player lives near boxes who sell peas and another one lives near boxes buying peas, they could trade among each other.
The view showing the player where he already visited boxes is a table at the moment. This should be changed into a map, which possibly also shows other not yet visited boxes. This would give players a much better overview of the system. The map currently available on the web, should be integrated into the app.

A

User Guide

GnomeTrader - User Guide

In this game you can experience a new world with the 20-Minuten-Boxes and the Blick-Am-Abend-Boxes, which are distributed all around Switzerland.

WHAT YOU NEED

- An Android or iOS Phone or tablet
- A mobile data connection
- GPS on your device

INSTALLATION

Android:

Download the app at:
http://citywidegames.graphics.ethz.ch/trader/GnomeTrader.apk

iOS:

Create an account at testflightapp.com from your device and ask me for an invitation (bombschm@student.ethz.ch). Afterwards you will receive a link to install the app on your device.

GETTING STARTED

The first thing you should do is create an account by first tapping the  and then the „Create New Account“ Button.

THE GAME

In each 20-Minuten-Box lives a little Gnome.

Some gnomes are Gardeners.  They produce either Nuts  or Peas  and you can buy these products from them.

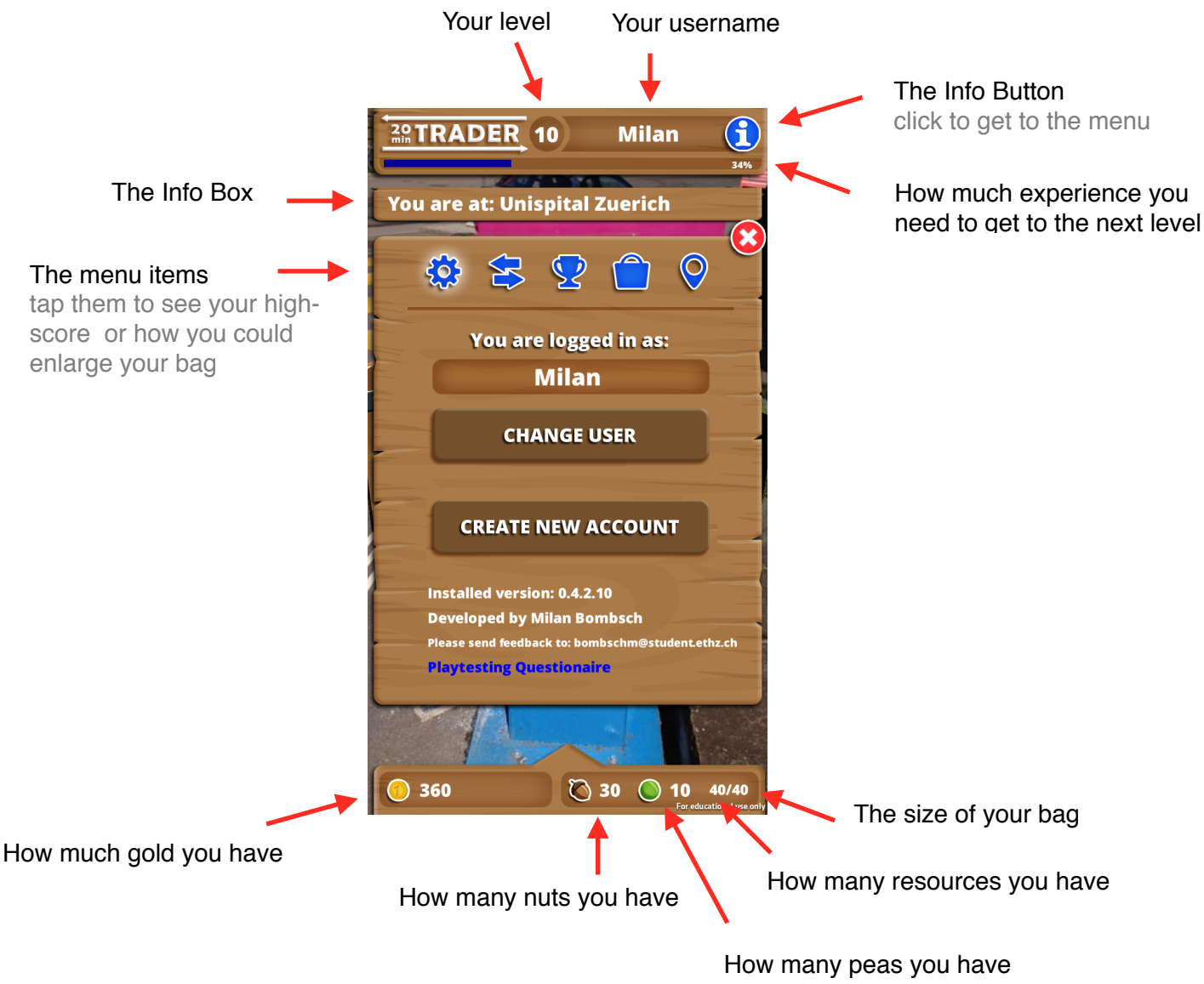
Some gnomes are Family Gnomes  and they need Nuts or Peas to survive. You can sell these to them and get gold  in return.

Your task is to buy goods from the Gardeners and sell them to the Families. When you get better and better you can buy a bigger bag  to trade more efficiently.

BE THE BEST

The amount of money you transfer by buying and selling peas and nuts, the size of your bag and how much 20-Minuten-Boxes you visit add up to your experience. If you reach a certain amount of experience you get to a higher level.
You can compete with other players on the leader board  and see how good they trade.

THE CONTROLS



At a 20-Minuten-Box:

Move the slider to change how much you want to buy/sell

The gnomes themselves

The gnomes blackboard:
Can you buy or sell something to him?
For how much?

The gnomes basket and how full it is

Tap this Button to perform the purchase or sale

After playing for a while it would be nice if you could give me some feedback through the **Playtesting Questionaire**

ETH zürich

FIcontent

B

Play-testing Questionnaire

20MinTrader - Playtesting

This is a questionnaire about the playtesting of the game '20MinTrader'. You can answer it in english or german.
Thank you for playing the game :-)
If you have any questions please conatct me at: bombschm@student.ethz.ch

* Erforderlich

About you

Your gender? *

- ◯ Male
- ◯ Female

Your age? *

How would you rate your experience with computer games in general? *

	1	2	3	4	5	
none	◯	◯	◯	◯	◯	a lot

How would you rate your exeprience with augmented reality games? *

	1	2	3	4	5	
none	◯	◯	◯	◯	◯	a lot

About your device

Which device did you use to play the game? *
e.g. Apple iPhone 4, Samsung Galaxy S3, etc.

Did you allow the game to use GPS? *

- ◯ Yes

- ◯ No
- ◯ I don't have GPS

How did you connect to the internet? *

- ◯ Mobile Data
- ◯ Wi-Fi
- ◯ Tethering from another device
- ◯ I did not connect to the internet

About the game

Which version of the game did you play? *
The version can be seen when you press the 'i' in the upper right corner

How long did you roughly play the game overall? *

How did the controls feel? Did they make sense? *

	1	2	3	4	5	
confusing	◯	◯	◯	◯	◯	self explaining

What was your first impression? *

How did that impression change as you played? *

Was there anything you found frustrating and if yes what? *

Did the game drag on at any point and if yes when? *

How do you rate the functionality of the leaderboard? *

	1	2	3	4	5	
hard to use	○	○	○	○	○	easy to use

Describe the objective of the game: *

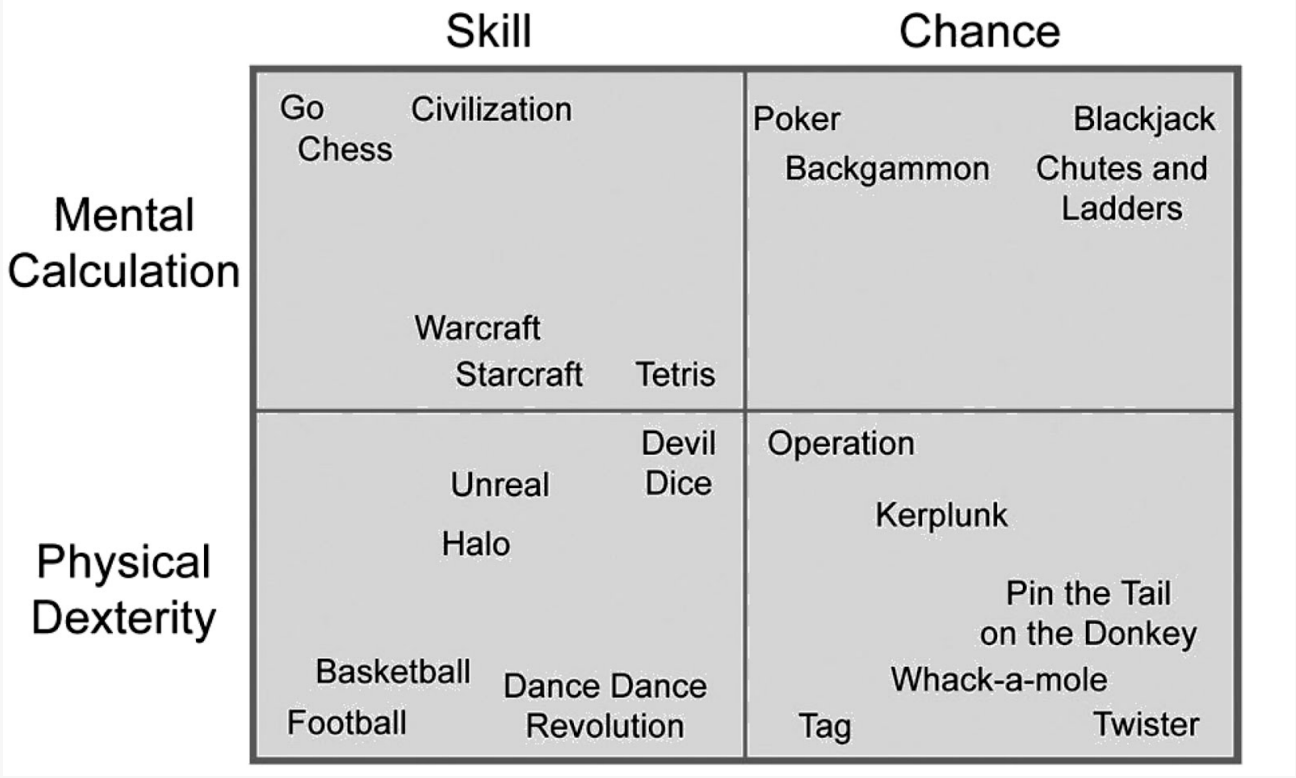
Was the objective clear at all times? *

- ○ Yes
- ○ No

What was your strategy for winning? *

Did you find any loopholes in the system and if yes which? *

Play matrix



Where would you put the game in the play matrix?

	Skill				Chance
Mental Calculation	○	○	○	○	○
	○	○	○	○	○
	○	○	○	○	○
	○	○	○	○	○
Physical Dexterity	○	○	○	○	○

What was missing from the game? *

If you could change just one thing, what would it be? *

If you were to give this game as a gift, who would you give it to? *

Thank you for helping me to improve the game

Bibliography

[20M14] 20-Minuten Zeitschrift. http://www.20min.ch/, August 2014.

[Bli14] Blick-Am-Abend Zeitschrift. http://www.blickamabend.ch/, August 2014.

[Cit14] City-Wide Augmented Reality Games group at ETH Zürich. http://citywidegames.graphics.ethz.ch, August 2014.

[FIC14] Future Internet Content a project of FI-PPP supported by the European Commission. http://mediafi.org/, August 2014.

[Ing14] Ingress - A City-Wide Augmented Reality Game developed by Google. https://www.ingress.com, August 2014.

[Jun14] Junaio - an augmented reality browser by metaio. http://www.junaio.com, August 2014.

[LDNB13] Gun A. Lee, Andreas DuÌĹnser, Alaeddin Nassani, and Mark Billinghurst. Antarcticar: An outdoor ar experience of a virtual tour to antarctica. *IEEE International Symposium on Mixed and Augmented Reality 2013*, 2013.

[Lea14] Future Internet Content - Pervasive Games Platform. Leaderboard component. Programmed by Marcel Lancelle. http://mediafi.org/?portfolio=leaderboard, August 2014.

[Map14] Google Maps Javascript API. http://maps.googleapis.com/maps/api/js, August 2014.

[mob14] Mobile Outdoor Augmented Reality Framework by Human Interface Technology Laboratory New Zealand. http://www.hitlabnz.org/mobileAR, August

2014.

[Tes14] TestFlight is used to install GnomeTrader on iOS devices. `https://testflightapp.com`, August 2014.

[Tra14] Swiss public transport API. An Open Data project. `http://transport.opendata.ch`, August 2014.