Final report for the Mixed Reality WS2017/2018 final project: Dragonhood

Akash Castelino
Saarland University
s8akcast@stud.uni-saarland.de

David Liebemann
Saarland University
s8dalieb@stud.uni-saarland.de

ACM.
Saarbrücken, Saarland University
ACM Unknown.
Unknown

Abstract

.

Author Keywords

Final report; Mixed Reality; Augmented Reality; Interactive video game; Puzzle game; Cooperative game;

ACM Classification Keywords

H.5.m [Information interfaces and presentation (e.g., HCI)]: Miscellaneous; See [http://acm.org/about/class/1998/]: for full list of ACM classifiers. This section is required.

Introduction

The final project, currently named Dragonhood and developed for the Mixed Reality seminar, is a Mobile Game with Augmented Reality aspects, set in the real world neighbourhood of players. The real aspects are enhanced with fictional scenarios players have to cooperatively solve to win the game.

Previous projects

The project is built on basis of two prototypes developed over the course of the seminar, which were combined, enhanced and improved in order to build a complete experience.

The "Travel Guide" mobile application enables users to connect with nearby users in order to determine the next travel

goal. Users can view their current surroundings on a map and experience the position of certain Points of Interest by seeing the direction they are looking at in the real world. The map view and the display of the real world rotation were used in the final project.

The "Point and Click Neighbourhood Adventure" prototype is the main precursor of the final project. It enabled users to connect to other nearby users and solve two scenarios, one of which could only be solved cooperatively.

Figure 1: Vuforia-marker for the scenario “Puzzle-box”.

Figure 2: Vuforia-marker for the scenario “Dragon”.

In this paper a *scenario* is defined as a sequence of tasks which players have to successfully solve in order to gain a reward. A scenario is activated by scanning a so called “Vuforia Marker”, a certain image printed on paper.

Vuforia is an Augmented Reality Software Development Kit for mobile devices that enables the creation of Augmented Reality applications. It uses Computer Vision technology to recognize and track images in real time. [16] Examples for Vuforia Markers can be seen in figure 1 [19] and figure 2 [18].

All aspects of finding and solving scenarios during a game session will further be called an *adventure*. The prototype focused on the gaming aspect of such an adventure and offered only limited cooperative play. The final project enhances this prototype by providing a Creation Mode and mandatory cooperation between players in order to win the game.

Final project
The newly added Creation Mode allows players to split up each scenario marker into multiple pieces and set the GPS-position for each of them. To choose GPS-positions they need to move to a location in the real world, select a scenario type and confirm the position selection. Once all marker pieces are set, the information can be saved to file.

The improved Play Mode allows users to load save-files created during Creation Mode and share the scenario positions with other players over the network. Scenario positions can then be viewed on an Augmented Reality map, which is displayed when scanning the corresponding Vuforia Marker.

While the Puzzle-box scenario received improved usability and touch controls, the Dragon scenario was enhanced with mandatory cooperative gameplay elements – just as with the Puzzle-box scenario, players now have to work together to complete the Dragon scenario.

The project was implemented using Unity3D [10], a cross-platform 3D engine mainly used for video game development.

Motivation
In modern day life, especially in metropolitan areas, people often do not know the names of their neighbours or have never explored their neighbourhood. It is hard for parents to motivate their children to go outside and play there – the perspective of staying at home and playing video games is more appealing. On the other hand, exploration of the neighbourhood is sometimes discouraged by parents, if dangerous areas are nearby.

This game motivates players to explore their surroundings while looking for good spots to create scenario locations during Creation Mode or while searching for scenarios during Play Mode. It demands cooperative play in order to accomplish tasks, players will have to find people nearby to play the game with. To successfully finish adventures, players have to speak with each other in the real world, conveying team play and communication skills.

Children can be motivated to play outside and engage in physical activity due to the mix of having to search for scenarios combined with virtual gameplay on the mobile device. Parents can create adventures for their children to either have some fun family time or keep their children away from dangerous locations.

Requirements for using the application
The application will be usable on mobile phones running Android: OS 4.3 or higher or iOS 7.0 or higher [7]. The phone will need the following sensors and features for the application to work correctly:

- Gyroscope
- Accelerometer
- GPS
- Camera
- Mobile Hotspot creation [9]

The application has been successfully tested on a Samsung Galaxy S3 (2012) with Android: OS 4.3, a Samsung Galaxy S8 (2017) with Android: OS 7.0 and an iPhone 6 with iOS 11.2.2 installed.

Related work

Geocaching Mobile App
The Geocaching® Mobile App [2] is built similarly to the presented project. Creators hide containers, called “geocaches”, at certain locations. They mark the GPS-positions of a cache in the application, add a picture and a small description hinting at the rough location of the cache.

Players can then start looking for those caches, based on the information given in the app. Because simply navigating to a GPS-position is easy, caches are usually well hidden – not only to provide a challenge for players, but also to hide the caches from random bystanders. Caches usually contain a block of paper and a pencil, containing the names of all players that found the cache.

The Geocaching Mobile App works with the same concept of creational freedom for Creators as the Dragonhood application, especially in terms of the Creation Mode and the navigation during Play Mode.

Pokémon Go
Positive experiences with Pokemon Go

Todo: introduce Geo Caching and Pokemon Go. Our application is a mix of both

Choice between Creation Mode and Play Mode

On start-up the player is presented with a simple menu, in which he can choose between entering Creation Mode or Play Mode.

Creation Mode

The Creation Mode allows players to create an adventure, adjust it to their neighbourhood and save the adjustments to file.

Entering Creation Mode
Upon entering Creation Mode the player is presented with a map depicting the real world surroundings, centred on the current GPS-position of the players phone. Map textures are downloaded from Google Maps [3] using the Google Maps Developer API [4].

The player position and view direction are represented by a small player model and a blue coloured pointer. The UI shows buttons to set new markers, save the adventure, adjust the zoom level or exit the Creation Mode. The save but-

ton is disabled in the beginning, indicated by a red colouring.

By holding the zoom button and tilting the phone, players are able to zoom in and out of the map.

An example scene upon entering the Creation Mode can be seen in figure TODO.

Setting Scenario markers
Scenario markers display GPS-locations of scenarios in the real world. They are represented by small models depicting a Puzzle-box or a dragon on the map. Upon entering the Creation Mode there are no markers present. Players have to determine scenario-locations by moving to a point in the real world and selecting the button for setting new markers.

On the first “Set marker” button press, the player is presented with a menu to select the amount of pieces each of the two Vuforia Markers is split up into. Players may choose to use between one and four pieces for each scenario.

On subsequent button presses, the menu will allow players to either set a “Puzzle-box” or a “Dragon” type scenario, up until the previously selected amount of pieces was created. Once the maximum number of markers of a certain type is reached, the selection button for that type is deactivated, indicated by a red colouring.

After choosing a scenario, a corresponding model will appear on the current GPS-location of the player. Players now have to select the corresponding, printed Vuforia Marker piece and place it at the chosen real world position.

An example-view of the display after a player set several markers is depicted in figure TODO.

Saving the adventure
After all markers are set by the player, the “Save” button becomes available. Pressing this button enables the player to store the current positions of the markers and assign a name to the created adventure. After completing this process, the adventure can be loaded up and played during Play Mode.

Exiting Creation Mode
Players can choose to exit Creation Mode at any time. Unsaved progress during creation is lost upon exiting, only saved adventures are available to play.

To create a new adventure, players need to exit and reenter Creation Mode.

Play Mode

The Play Mode allows players to experience previously created adventures. They need to work cooperatively in a group of two people to successfully complete an adventure. To achieve this, players have to navigate towards the GPS-locations of each scenario marker in the real world, gather all the Vuforia Marker pieces and then complete both scenarios.

Entering Play Mode
Upon entering Play Mode, the player is presented with the current video input of the phones camera and a range of UI-elements, as depicted in figure TODO.

Network connection
To play cooperatively, players have to connect their applications over a Wi-fi network. All players need to have the current version of the application running on their mobile phones. One player starts a Software Hotspot [9], the other players then connect to that hotspot.

Once everybody is connected, one of the players has to press the “Host” button and will subsequently be denoted as Host. Every player connected to the hotspot then automatically connects to the Host in the application.

The Host can now choose between starting the class selection and loading an adventure by pressing the corresponding buttons. Using the “Class selection” or the “Load adventure” button is only available to the Host player.

Class selection
A player-class represents a certain type of role, a player has to fit in. It equips players with special abilities and unique tasks they have to fulfil over the course of the adventure.

After a click on the “Class selection” button by the Host, every player is presented with a choice between one of the two following classes:

- *Puzzle-master:* Required to solve the Puzzle-box. Can carry a bell.
- *Fighter:* Required to win against the Dragon. Is able to carry a feather.

Each class will have to be picked by one player in order to complete the adventure. The chosen class type will be displayed in the ?? corner as depicted in figure TODO and on the map, as discussed in the section “Navigation Scroll”.

Loading an adventure
After pressing the “Load” button, the Host is presented with a list of saved adventures. After choosing one, the names and GPS-locations of the scenario pieces are distributed to each connected player and made visible on the Navigation Scroll.

Navigation Scroll
The Navigation Scroll represents the possibility for players to locate the scenario marker positions in the real world. By scanning the corresponding Vuforia Marker with the phones camera, the Navigation Scroll is activated and shown as an Augmented Reality object on top of the marker.

The Navigation Scroll view is similar to the view used during the Creation Mode: As a base plane, a Google Maps image of the real world surroundings is displayed. Players can see their real world position and rotation represented by a small player model and a blue pointer. Scenario markers are represented by either a small Puzzle-box model or a small Dragon model. By moving their phones closer or farther away from the Vuforia Marker, players can naturally zoom in or out of the map.

To find a scenario marker, players will have to move to one of the marked locations on the Scroll. The change of position and rotation is projected in real time on the Navigation Scroll, enabling players to track their progress and head in the intended direction.

Starting a scenario
Before attempting to start a scenario, players will need to find all Vuforia Marker pieces of a single type in the real world and put them together to form a complete image.

Players can scan and thus activate a scenario by pointing their camera towards the complete Vuforia Marker. Once the application has identified a marker, the corresponding scenario will be shown as an Augmented Reality object on top of the marker, as depicted in figures TODO and 3. Players can then start solving the scenarios.

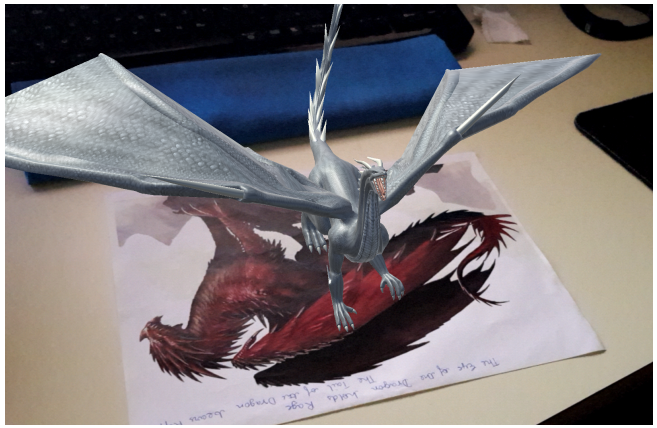

Figure 3: Sample view of the Dragon as an Augmented Reality object.

Puzzle-box

The Puzzle-box is the first of two scenarios the player has to solve in order to successfully complete an adventure.

The Puzzle-box consists of five visible sides with different numbers depicted on each side, similar to an ordinary dice. The number on each side corresponds to the number of fingers the player has to touch it with – f.e. if a side shows the number three, the player has to touch the side on the screen with three fingers.

If the correct amount of fingers is used, the side turns yellow to signal a correct input – it is now seen as "solved". Otherwise it will turn red for a second and then return to the original colour, in order to signal incorrect input.

Once a side turned yellow, the player who solved that side can not solve any other side of the box. He needs to wait for another player on the network to solve the corresponding side, which is defined as the opposite side. A light effect and glowing particles, visible to both players, give a hint at which side the player has to solve. Once that side is solved by the other player, both sides turn green, they are now marked as "finally solved". Otherwise all sides on the Puzzle-box turn red and the Puzzle-box is reset to its original state.

Players have to work together, as the Puzzle-box can not be solved by one person alone. All players are able to see whether or not a side has been solved, but only the Puzzle-master is able to see the numbers depicted on the sides of the box, giving him the most important role in this scenario. The Puzzle-master has to communicate in the real world with the Fighter in order to tell him, which number is depicted on the side the Fighter intends to solve.

Completing the scenario
The scenario counts as completed, once all five sides of the Puzzle-box have been finally solved and turned green. The scenario then presents the rewards as Augmented Reality objects on top of the Puzzle-box: A feather and a bell, which are both required to solve the Dragon-scenario.

Players can pick up the item corresponding to their class and store it in their inventory, as depicted in figure TODO, number 3.

Dragon

The fight against the Dragon can only be started once the Puzzle-box scenario was solved and both players picked up the items they received as a reward. Each player adopts a specific role based on the item they acquired from the Puzzle-box.

The Fighter with a feather has to perform an action sequence on the Dragon – examples for actions and the cor-

responding hints can be found in figures TODO and TODO. Actions can only be performed from behind the Dragons back. Initially, the dragon will face and follow the Fighter, making it impossible to perform the required gesture. The Dragons movements and animations are replicated on the network, so both players can see the current state of the fight.

The Puzzle-master has to shake his mobile device to produce a bell ring . Once the bell ring action is completed, the Dragon will face and follow the Puzzle-master who rang the bell, which allows the Fighter to move behind the Dragon.

Once the Fighter looks at the Dragon from behind, the action-gestures to be performed become visible and appear as written hints above the dragon. A successfully performed action forces the Dragon to look at the Fighter again, an incorrectly performed gesture leads to the dragon blowing fire.

Repeating this sequence – ringing the bell and performing the actions successfully in coordination – leads to the Dragons defeat. It then flies away, leaving a model of a treasure chest, full of gold.

Implementation

The following section will discuss the implementation of the presented concepts. Naturally the project can be divided into two parts: A creation section and a play section. Each part is contained in its own Unity-Scene [15] and gets loaded up when the corresponding button in the selection menu is pressed.

Creation Mode

The Creation Mode can further be subdivided into the rendering of the Google Maps texture, the setting of the scenario markers and the save file creation.

Google Maps

The Google Maps image displayed in both Creation Mode and Play Mode has to be downloaded at runtime to display the current surroundings of a player. The script for downloading the image was taken from a public git repository [6] and adjusted to fit the projects requirements.

Parameters like the centre GPS-position, texture size and zoom level are converted into a web request, which returns the image information that can be loaded into a texture usable by Unity3D. This texture is then displayed on a primitive plane inside the scene. The centre GPS-position is by default set to the devices GPS-position on start-up.

As the player model represents the current position and view direction of the player, the devices GPS-position and rotation is checked regularly and updated on the map. The necessary information is gained from the devices GPS and Gyroscope sensor, provided by Unity3Ds built-in Input framework [11]. As GPS-position and Unity3D-position definitions differ, a conversion – based on the map-plane centre and Google Maps zoom level [20] – is performed.

Setting marker locations

The UI interactions are implemented by using Unity3Ds built-in UI. Once the amount of pieces each scenario marker should consist of was chosen, the controller script stores that information. On every subsequent “Set Marker” instruction, the controller script updates the information both internally and visibly in the UI. If the maximum amount of pieces of a certain scenario type was set, the corresponding UI button is disabled. Once the maximum amount of pieces for both types was set, the save button gets activated.

Additionally, on each “Set Marker” instruction a Unity Prefab [14], corresponding to the scenario type chosen, is spawned at the devices current GPS-position and thus

made visible on the screen.

Saving an adventure
To save an adventure, the serialization options of C# [8] are being used. Once the Save button is pressed, the relevant information of each marker – scenario-type-ID, name and the GPS-position – is written into a serializable data-object and stored in a C# List [5]. This list is then serialized by a binary formatter [1] and stored as an ".adv" file on the local machine data storage.

Such a file can then later be deserialized again into a list of data-objects containing the stored information.

Play Mode
The Play Mode implementation can be further subdivided into network, Navigation Scroll, Puzzle-box and Dragon implementation.

Network
For establishing a connection between players, the Unity3D Networking framework [12] is being used. At the start of Play Mode the application starts listening to messages being broadcasted over the local area network. If a player decides to become a Host, the application stops listening for broadcast-messages and starts broadcasting itself.

Any player on the local area network, who is not a Host, receives the broadcasted messages and uses the transmitted IP address to establish a connection to the Hosts server. Every connected player – further called client – can now send and receive messages over this connection. The Host application is a client and a server running on the same device – the client and server part of the application are disconnected from each other. The only difference between the Host and a normal client application is the fact that the Host device can start the "Class selection" and "Load adventure" process.

For class selection, a certain message type without any further information is sent to all client devices, which starts up the UI for class selection. The application locally stores the chosen class information.

A "Load adventure" message contains the list of scenario marker information which was deserialized from the selected save file. The message content is transferred to the navigation logic to be displayed on the Navigation Scroll.

Navigation Scroll
The Navigation Scroll reuses most of the Creation Mode control logic, like the display of the Google Maps image and the player model. It is a child of the Vuforia Image Target Prefab [17], which activates once Vuforia recognizes the corresponding Vuforia Marker from the camera video input.

It also processes input received from the "Load adventure" message by spawning scenario marker prefabs according to the information contained in the message. GPS-position to Unity-position conversion is the same as performed during Creation Mode.

Puzzle-box
Similar to the Navigation Scroll, the Puzzle-box object is a child of the Vuforia Image Target Prefab and activates, once the corresponding Vuforia Marker was recognized. For input information, the Unity3D Input framework is used.

The Puzzle-box consists of five Puzzle-faces processing input independently. If a face is touched with the wrong amount of fingers, it locally marks itself as wrongly touched for a second, then returns to its original state, without any other effect. Once a face recognizes valid input, it trans-

mits that information to a controller script, which then sends a message containing the Puzzle-face ID to all connected clients.

The clients controller script then marks the correctly touched face in yellow and activates the hint effect on the opposite face. It also blocks all Puzzle-box input if the message was sent by the same device – forcing the other player to solve the opposite face. An exception is made if the top Puzzle-face was correctly touched: the face is simply seen as finally solved on all connected devices and marked in green. If the application receives a message with the ID of the opposite Puzzle-face, both faces get marked as finally solved. Otherwise a reset message is sent over the network, returning all faces to their original state.

After every “Puzzle-face Touch” message, the controller script checks, if all Puzzle-faces were finally solved. If that is the case, the reward-objects – a feather model for the Fighter, a bell model for the Puzzle-master – become visible. A click on them will mark them as collected by the player and display the corresponding UI information.

Dragon
Similar to the Puzzle-box, the Dragon object is activated once the corresponding Vuforia Marker was recognized. The Unity3D Input framework is used to obtain the number of touches as well as start and end position of each touch on the mobile devices.

The difference between start and end positions of a touch is the “gesture-distance”. Additionally, the difference between start and end time forms the “gesture-time”. With these values the speed of a gesture can be calculated. In order for a gesture to be successful, it must be performed with the correct number of simultaneous touches and a speed above the set threshold value.

The order of the actions-sequence gestures, performed by the Fighter, are determined by boolean variables that represent each gesture. They are set to false by default. Once a gesture is successfully executed, the variable is set to true.

The bell ring action, performed by the Puzzle-master, uses the accelerometer data of the players mobile device, provided by Unity3Ds Input framework. Once the devices acceleration is larger than a fixed value, the bell ring audio clip is played for feedback.

For synchronization, messages containing the state of animations, particle systems and gestures are sent to all connected clients after an action was performed. To synchronize the movement of the dragon, the position of each players device relative to the dragon gets replicated on the network, using the Unity3D Networking framework [13].

Conclusion and future work
TODO

Differences to original concept
In a preceding concept document, mandatory and optional features of the final project were presented and discussed.

The finished application implements every mandatory feature as planned, except for the navigation aspect during Play Mode. As discussed in the concept document, the implementation of the optional feature of splitting up Vuforia Markers into multiple pieces forced the development of a new navigation method, as discussed in the section “Play Mode”.

The adjustment lines up well with the Augmented Reality aspect of the application, builds a highlight of the Play Mode and thus is considered as an improvement over the

original concept.

Possible problems out of developers reach
As the creation of adventures is community driven, real world aspects of the game – like the correct positioning of Vuforia Markers in the real world – can not be controlled by developers. Experiences with related applications, like the Geocaching® Mobile App [2], support the assumption that creators will act with caution while creating an adventure, as it is in the best interest of everyone using the application.

Future work
To mitigate the effects of imprecise GPS positioning, creators could be able to add a picture of the scenario location and a small hint to the scenario marker. This extension is inspired by the Geocaching® Mobile App [2], which uses similar mechanics to aid people in finding certain, hidden spots in the real world. This addition, combined with a number of new scenario types would lead to more and diverse adventures for players, increasing the amount of fun even after multiple replays. Additional player classes would enable bigger groups to work together, increasing the cooperative complexity.

Not every end product user will be willing to set up a hotspot to connect to other players. The connection should either be set up automatically, without having to deal with external requirements, or via mobile data.

REFERENCES

1. 2018. BinaryFormatter Class. (2018). https://msdn.microsoft.com/en-us/library/system.runtime.serialization.formatters.binary.binaryformatter(v=vs.110).aspx
2. 2018. Geocaching® Mobile App. (2018). https://newsroom.geocaching.com/
3. 2018a. Google Maps. (2018). https://www.google.de/maps
4. 2018b. Google Maps API for developers. (2018). https://developers.google.com/maps/?hl=de
5. 2018. List<T> Class. (2018). https://msdn.microsoft.com/en-us/library/6sh2ey19(v=vs.110).aspx
6. 2018c. Load a Google Map into Unity3D. (2018). https://gist.github.com/derme302/d57a395b711b4ecf100f
7. 2018a. Requirements to run Unity3D. (2018). https://unity3d.com/de/unity/system-requirements
8. 2018. Serialization (C#). (2018). https://docs.microsoft.com/en-us/dotnet/csharp/programming-guide/concepts/serialization/
9. 2018. Software Hotspot (Wi-Fi). (2018). https://en.wikipedia.org/wiki/Hotspot_(Wi-Fi)#Software_hotspots
10. 2018b. Unity 2017.3. (2018). https://unity3d.com/de
11. 2018c. Unity Input. (2018). https://docs.unity3d.com/ScriptReference/Input.html
12. 2018d. Unity Networking Overview. (2018). https://docs.unity3d.com/Manual/UNetOverview.html
13. 2018e. Unity NetworkTransform. (2018). https://docs.unity3d.com/Manual/class-NetworkTransform.html
14. 2018f. Unity Prefabs. (2018). https://docs.unity3d.com/Manual/Prefabs.html
15. 2018g. Unity Scenes. (2018). https://docs.unity3d.com/Manual/CreatingScenes.html

16. 2018a. Vuforia explanation. (2018). https://en.wikipedia.org/wiki/Vuforia_Augmented_Reality_SDK
17. 2018b. Vuforia Image Targets. (2018). https://library.vuforia.com/articles/Training/Image-Target-Guide
18. 2018. Vuforia-marker image used for the dragon scenario. (2018). https://benwootten.deviantart.com/art/Red-Dragon-118573547
19. 2018. Vuforia-marker image used for the Puzzle-box scenario. (2018). http://wallpaperswide.com/question_marks-wallpapers.html
20. 2018d. What ratio scales do Google Maps zoom levels correspond to? (2018). https://gis.stackexchange.com/questions/7430/what-ratio-scales-do-google-maps-zoom-levels-correspond-to