Project - Asteroid Pong

MAMN01 - Advanced Interaction Design

Group: Elon Musk

Hampus Balldin	<dat12hba@student.lu.se>
Anton Lin	<dic13ali@student.lu.se>
Johan Månsson	<ama10jma@student.lu.se>
Viktor Selleby	<tna12vse@student.lu.se>

Supervisor: Günter Alce

October 19, 2016

Division of Labor

The group's project plan was to always have something new to show at the weekly meetings with our supervisor. This meant several iterations of the prototype, leading up to the final exhibition, where each group member was responsible for implementing functionality needed for that weeks meeting. The following were the group's division of labor.

Table 1: Division of Labor

Tasks:	**Hampus:**	**Anton:**	**Johan:**	**Viktor:**
Idea Brainstorming:	*X*	*X*	*X*	*X*
Low-Fi Prototype:	*X*	*X*	*X*	*X*
Graphical Profile:			*X*	
Navigation:			*X*	
Network Model:	*X*	*X*		
Wifi Direct Module:	*X*	*X*		
Accelerometer Controller:	*X*			
Voice Recognition Function:				*X*
Software Testing:	*X*	*X*	*X*	*X*
Usability Tests:	*X*	*X*	*X*	*X*
Game Trailer:			*X*	
Report Writing:	*X*	*X*	*X*	*X*

Abstract

Smartphones have transcended the notion of being less powerful computers with phone capabilities. Using the smartphone's many input sensors, one can obtain a user experience which is more integrated with the surrounding physical world, pushing the user to interact beyond the screen. This is something that most applications of today have yet to accomplish.

A big challenge, given the constrained hardware present in current smartphones, is how to detect and measure sub-meter movement of a smartphone device. In this project, a new type of accelerometer reader has been created in order to track a smartphone device's lateral motion. As a proof of concept it has been integrated into a game application as to showcase its accuracy and responsiveness. Additionally, the game application is also a means to demonstrate the potential and excitement there can be in something as "trivial" as the measuring of a smartphone's sideways movement and how it can encourage the user to actually socialize with its immediate surroundings through the game application.

Contents

1 Introduction

This report describes the software application project in the course Advanced Interaction Design, MAMN01. The purpose of the project is to find new creative ways to utilize sensor data from a smartphone device in order to interact with an application.

1.1 Background

Over the last decade smartphones have become an integral part of everyday life for billions of people. Initially, the smartphone could be likened to a primitive laptop where the main use cases (besides the usual phone use cases) were to browse the internet and take photos. As smartphones have become increasingly technologically advanced, the potential for applications and services that fall outside of the usual "stationary computer" domain has grown, i.e. the many different sensors that can be found in a smartphone separates it from a traditional computer. These sensors pave the way for applications that can be more closely and seamlessly integrated with the physical world.

However, looking at the majority of available smartphone applications and the common usage of smartphones, the market has still not made full use of the functionality available in the modern smartphone. Most smartphone applications requires the full attention of the user, meaning that all of the focus is aimed towards the smartphone screen. This makes the realm of the smartphone application completely separated from the physical world. Our aim is to make an application that does not treat the application's virtual domain and the physical reality as mutually exclusive, but instead integrates the two and encourages the user to interact beyond the screen.

2 Method

In the following sections the time line of the design and implementation process will be described, along with theory behind our design decisions.

2.1 Idea Brainstorming

The first step taken in the design process was to generate ideas for the project. Starting off, the group participated in an idea brainstorming workshop to find and discuss new and innovative ideas. To prevent project members from influencing each others' thought processes, the group had agreed upon to not share any ideas about the project before the brainstorming session. During the workshop, the group members initially wrote down different individual ideas on post it notes before discussing the different aspects of the ideas together in the group. A photo from the brainstorming session can be seen in figure 1. A method called bodystorming [1, p.1] was also used to perform simulated interactions with the ideas. The workshop session resulted in four different application ideas presented in appendix section 6.

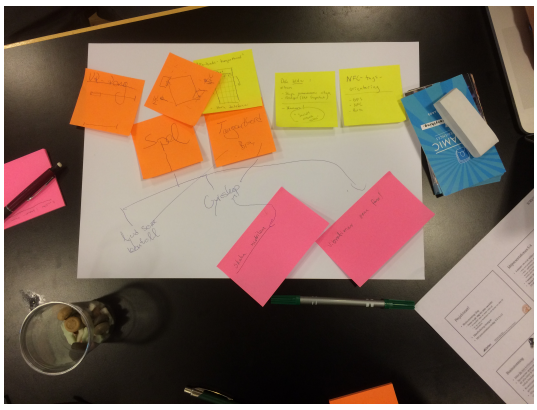


Figure 1: The end result of the initial bodystorming session

In the first meeting with the project mentor and a couple of other project groups, two of the ideas were presented and discussed to get further feedback. During this meeting the group, with help of given feedback, decided upon developing a simple Pong game that uses sensor data to control the game.

2.2 Idea

The groups' idea is a simple 2D-game inspired by the classical game Pong, that uses sensory input data like the device's accelerometer for control input. Using this type of sensory data enables the user to interact with the game in a more physical and direct way compared to using traditional screen input. Additionally, the game is to include multiplayer capabilities such that two players can easily connect and play against each other. Preferably, the users should be in the same room while playing as this should further decrease their perceived gap between virtuality and reality. The ball along with two pong paddles will be rendered, one paddle on each short side of the screen. Each user's device will be mapped to its respective paddle, such that when the device is subject to a lateral motion its digital counterpart will be moved accordingly.

2.3 Vision

In order to get as close to perfection as possible, setting up a clear vision without the need for total realism and consideration for limitations such as time and resources is likely to be beneficial. Further, thinking about what could be and pushing the ideas to their limits helps us explore the boundaries of the interaction, thus providing us a better understanding of our problem domain.

A Tangible Controller
Tangible interaction is a broad term which among other things encompasses "physical embodiment of data" and "the embedding of the interface and the users' interaction in real spaces and contexts" [2]. To create such a mapping between the physical and the virtual world, our vision is to have a one-to-one correspondence between the device position, and of the paddle position on the screen. Ideally, the mapping should be such that when the phone is physically in one corner of the table, the paddle should be in the corresponding corner on the screen of the phone. The virtual paddle is then moved by moving the physical device between the corners of the table. This mapping should also provide a strong and intuitive metaphor for new users as to how to control the phone. Important features for the mapping to exhibit are examined below.

Single purpose mapping
The controller is used to control one thing only. This distinguishes it from a typical general purpose computer mouse in an important way. In the thesis "Graspable User Interfaces", an important distinction is made between the physical tangibility of a device (e.g. an ordinary computer mouse) and that of a single purpose controller which is also tangible (e.g. a midi control) [3, sec. 1.3-4]. The latter is considered to be a stronger, more advanced form of graspable device and the thesis goes on to show that a stronger form implies both better performance and ease of learning [3, sec. 6.1.4].

One-dimensional motion, similarity, and linear response
The movement is similar and one-dimensional; the phone moves on a line and the virtual object moves on a line with the same orientation. Further, the mapping is linear, i.e. moving the phone some distance in the physical world always gives the same amount of change in the virtual world. According to Norman [4, p. 117], this type of spatial mapping usually provides the best controls to problems which are similar to our (Norman uses industrial control problems as an example).

The issue of (non-) visibility
The greatest challenge that our controller faces (if onw sets implementation issues aside) is that of visibility, what Norman refers to as discoverability [4, p. 116]. The phone does not itself show any affordance as to suddenly being able to control something and so is invisible to the user. Thus, ways of improving the discoverability of the control should be a priority during the design process and should be evaluated at during usability testing.

Augmented reality
If an accurate and fully responsive tracking of the device's motion could be obtained, an augmented reality version of the game could be achieved. This version would be in full 3D, with the physical world as the playing field presented in the phone. The control scheme of the virtual paddle is the same; sideways movement of the phone to move the paddle sideways. Figure 2 depicts what a possible

augmented reality pong could look like. Each player's device represents the virtual paddle in the physical world and by aiming the device camera towards the virtual ball's location within the physical world, the ball is visualized on the device screen.

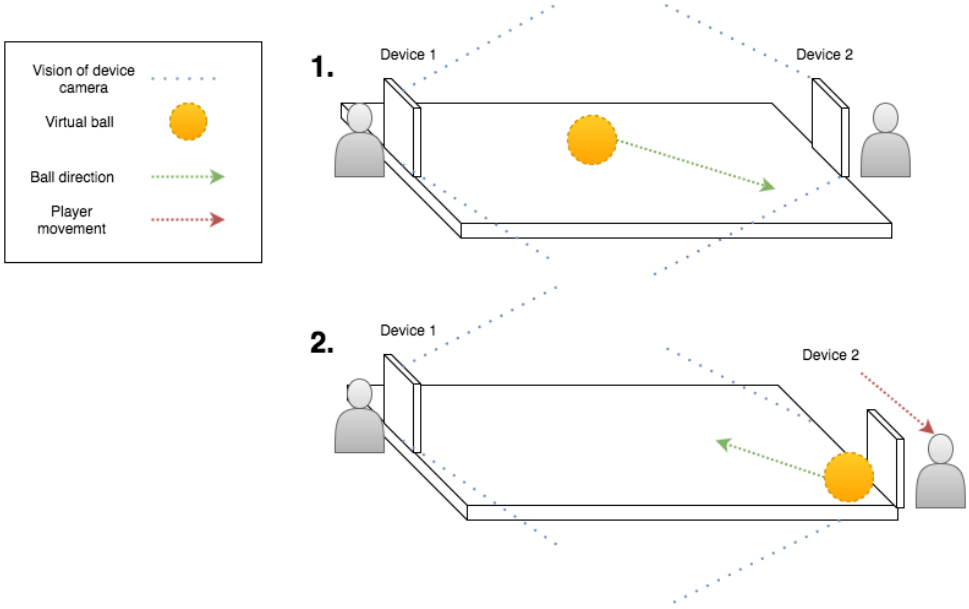


Figure 2: Augmented reality version of Asteroid Pong

Gameplay features
When it comes to gameplay implementations the following could be considered:

Ball direction affected by velocity of paddle	When ball is hit by a paddle in lateral motion, friction will cause the ball's direction to change
Several Balls	As a game round progresses more balls are put in to play in order to make the game more challenging
Power Ups	Power ups will materialize and a player can hit these with a ball to get the upper hand against the opponent
Multiple opponents/Teams	More than two players can participate in a game and form teams

2.4 Conceptual Model

Since the original Pong game is so well known it also serves as the project's conceptual model. The original Pong game, made by Atari in 1972 [5], is by many seen as the first successful arcade game. The game is limited when it comes to graphical components and consists of a moving ball and two pong paddles on a dark background. The original game's conceptual model drew inspiration from a traditional Ping Pong game where a ball is hit as long as possible between two players. The group's idea of using the device's accelerometer for control input, is believed to further enhance the original game's conceptual model since the player has to physically move the device similar to a real world Ping Pong game.

2.5 Low-Fi Prototype

The low-fi prototype is comprised of rudimentary sketches of the graphical playing field with the ball and the two player paddles, which can be seen in figure 3. Since the control scheme of the paddle (lateral motion of the phone to move the paddle sideways) is basic in its concept and integral to the game idea, no further low-fi prototyping was done of the gameplay itself.

Scenario and use case driven design [6, p.3] was used when composing the game menus and the different views. The main use cases are multiplayer oriented, this is reflected by the initial main menu where the first two options are associated with multiplayer gaming. Since single player is seen as a minor use case, it is put as the third and last option in the main menu. The user interaction from starting the application to a running multiplayer game were discussed between the group members and visualized on a whiteboard seen in figure 3 in the top right corner. A reworked model of the menu scheme based on the initial sketches were then made and can be seen in figure 3.

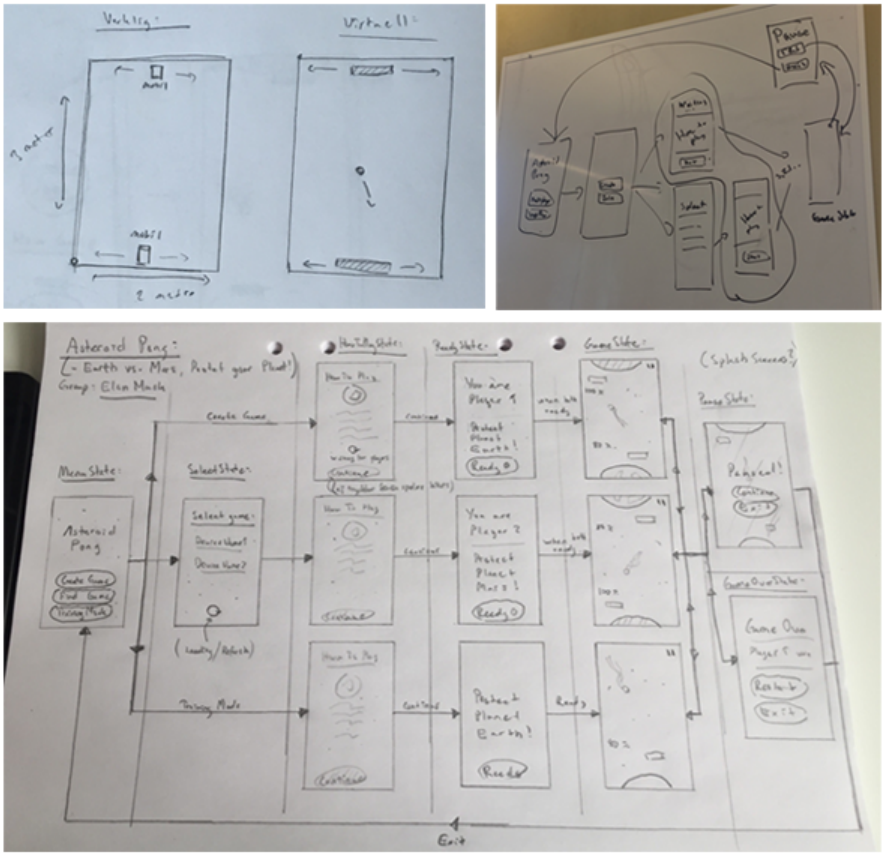


Figure 3: Top left shows sketch of control scheme, top right shows early sketch of menu scheme, bottom shows formal sketch of menu scheme

2.6 Graphical Design

The original game Pong is very limited and monotonous in its graphical design. To make the gameplay visually interesting and intriguing, a decision was made to make the game follow a space theme (as a subtle reference to Elon Musk's SpaceX program [7]). In this context, the ball is represented by an asteroid in space and the players have to protect their respective planet with a game paddle and survive as long as possible. Every time a player's planet is hit by an asteroid the planet's "life" decreases 25 percentage units. Shortly thereafter a new asteroid are displayed at a random position on the screen with a random direction. At 0% the planet explodes and the game is over. An initial sketch, seen in figure 4, shows how the gameplay could be visualized.

The space Pong idea means that a lot of graphical elements have to be created for the game. In addition

to the gameplay, a user must also be able to navigate between different menu states to setup and launch a multiplayer game. Other graphical components for the application includes titles, information texts and a launcher icon etc.

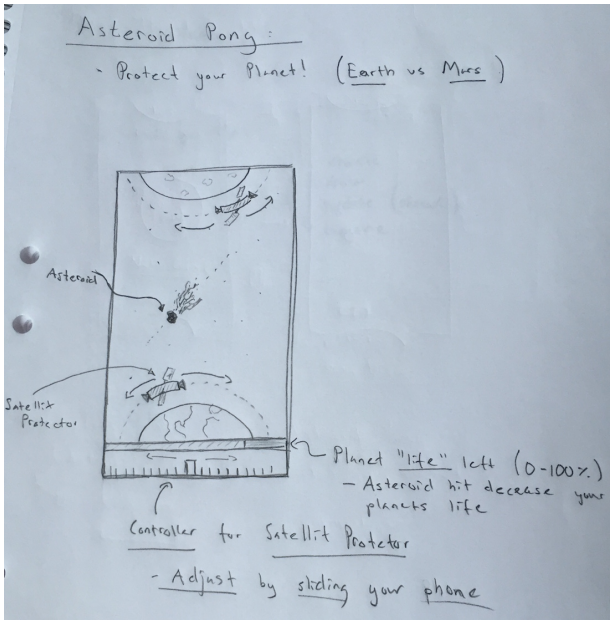


Figure 4: Initial sketch of the gameplay

2.7 Iterative Design Process

The design process as such was iterative (see figure 5), meaning a number of working prototypes were created and tested throughout the project.

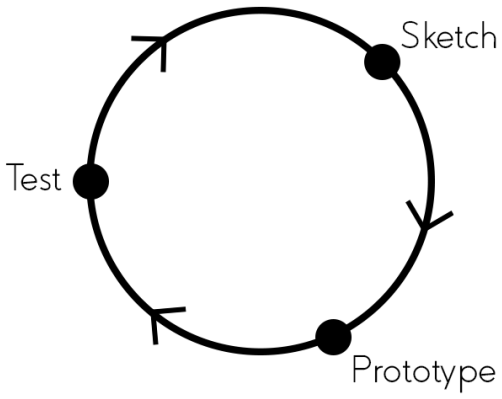


Figure 5: The iterative design process.

The iterative design process allows the design to be continuously refined based on feedback. Developing in increments allows the developers and users to quickly understand each other's mental models of the product to be developed, and the constant feedback given from the tests can help steer the developers in the "right" direction [8].

2.8 Supervisory Meetings

Throughout the project, a number of supervisory meetings were held. The meetings took place in smaller groups (consisting of three project groups) and were supervised by Günter Alce. Every group brought a working prototype of their application to be tested and reviewed by the other group members as well as Günter. The feedback received was then used in a new iteration to further improve the prototype.

At the end of each meeting, goals were also set for the next iteration. As such, the supervisory meetings were a crucial part of the iterative design process as every meeting marked the end of one iteration and the beginning of the next. The supervisory meetings also gave the group a chance to discuss different design paths and bounce ideas with the other groups along with Günter.

2.9 Usability Test

Aside from the feedback received from the supervisory meetings one minor, informal usability test was also held. In the test, users in pairs of two were asked to play against each other in an early prototype of the application. After finishing the game, they filled in a questionnaire designed to evaluate the user experience. The questionnaire used can be found in figure 18 in the appendix. The questions were mainly concerned with the gameplay aspect of the game since the game menu and its scheme had yet to be implemented.

2.10 Evaluation Test

An evaluation test involving a large number of testers was also conducted at the application exhibition. A user evaluation form was drawn up containing questions about the game menu navigation, setup of a multiplayer game and the user experience during gameplay. The test procedure was divided into the following steps:

- Two users are to participate in the test simultaneously.
- One user is to act as host of multiplayer game.
- One user is to join host's game.
- The users initiate and play x number of multiplayer game rounds.
- The users are asked to fill out the evaluation form.

Moreover, the test instructors were advised to only inform the testers of the major tasks. While the tasks were being executed, the instructors were to remain silent as to evaluate whether the developers' conceptual model corresponded to the one of the users'. The questionnaire used in the evaluation test can be viewed in figure 19 in appendix.

2.10.1 Risk Analysis

To get a better understanding of what functionality is most important to implement early and to make a better effort estimation a risk analysis is made. The total risk is calculated as importance (1 - 10) multiplied by the probability of failure (1 - 10)

Table 2: Risk Analysis table

Functionality	Importance (1-10)	Probability of failure (1-10)	Risk (1 - 100)
Accelerometer	10	8	80
Multiplayer	8	6	48
Graphic Design	6	3	18
Speech Input	2	5	10

The analysis showes that the accelerometer is the functionality with the highest risk. Because of this it should be the first functionality to be researched and implemented.

3 Results

3.1 Implementation

This section describes the implementation work done in the design iterations. The team worked according to general agile software development guidelines [9], working in iterations where each iteration would produce new or re-implemented functionality and be concluded with user testing.

3.1.1 Project Structure

The structure set up by the Asteroid Pong developers follows the traditional MVC-architecture. The general layout and dependencies can be seen in figure 6. The model contains the game state, coordinate system, game element positions, paddle velocity etc. Moreover, the respective models found at the server and client device have a network communication set up between each other, communicating their current game state. The view is in charge of populating the mobile device screen with the model data. All input registered by the mobile device is fed through the controller, where the input data is processed and then passed forward to the model. Different controllers used include:

- Speech input and recognition to stop and play the game loop.
- Acceleration signal to control the player position.
- Standard text input to select game name for multiplayer.
- Standard touch input to navigate in menu system.

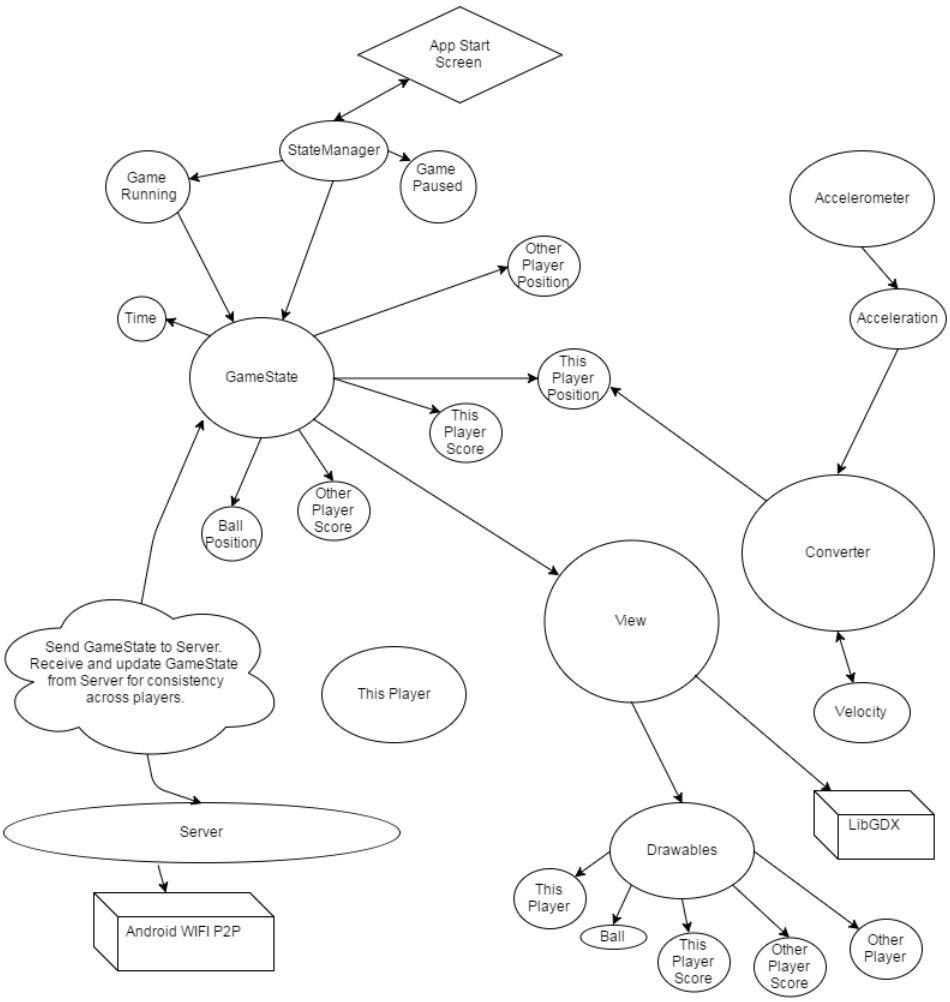


Figure 6: Structure of the project

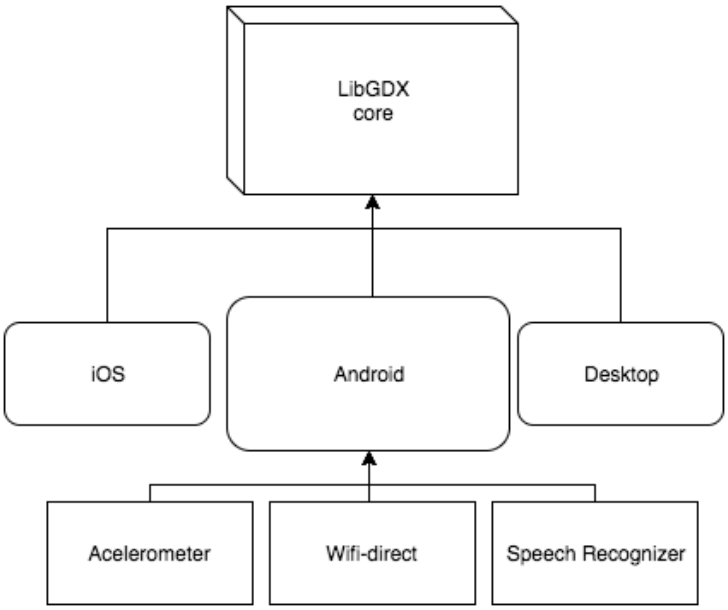


Figure 7: The different modules communicating with the LibGDX core

3.1.2 LibGDX

LibGDX [10] is a framework written in Java, built upon OpenGL, which can be deployed to Android, iOS and desktop. LibGDX acts as a layer between the developer and OpenGL, providing functionality that would be much more tedious to implement using OpenGL directly. Furthermore, since LibGDX is a cross-platform framework, it has support for a wide range of typical smartphone user interactions. For example, this means that one need not be concerned with platform specific code to handle swipe, tap, or other common touch input. The framework in turn, communicates with the native API of the platform that the application is being deployed on.

The typical structure of a LibGDX project can be divided into distinct modules. The "core" module containing all game logic and views, and then one module for each platform. As previously stated, LibGDX supports a wide range of sensor inputs, regardless of the underlying platform. But since Asteroid Pong utilizes a complex accelerometer, speech recognizer and a network solution which contains a lot of dependencies to the native Android API, this functionality needs to be placed directly in the Android module. Thus, one needs to write a new Accelerometer, speech recognizer and network solution for the remaining platforms in order to make the game available for iOS and desktop.

Regarding the pure gameplay development, LibGDX follows the standard concepts of game programming. A game loop, textures, states and actors were all included in the framework. This means that a minimal amount of effort was needed to setup a basic local Pong game without any formal graphic elements.

3.1.3 Wifi-Direct

Wifi-direct is a technology supported by Android devices where two or more devices can set up a direct wifi connection between one another without an external access point [11]. This technology is used in conjunction with the network module of the project. The wifi direct technology's main task is to set up a direct connection between two devices with the desire to initiate a multiplayer game of Asteroid Pong.

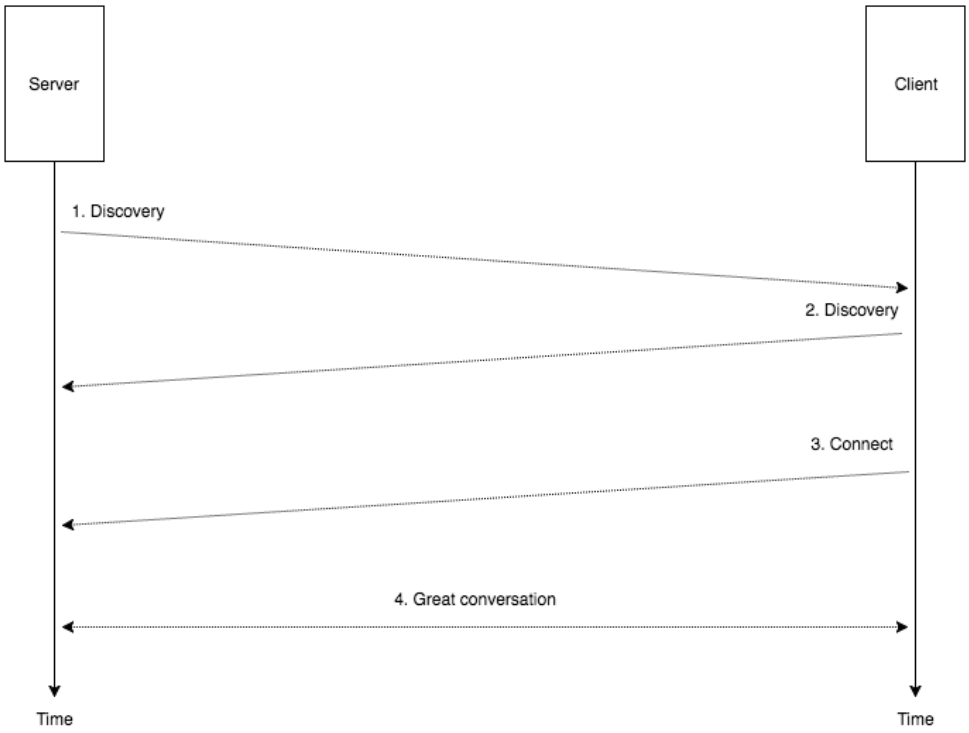


Figure 8: The wifi direct setup procedure between two devices running Asteroid Pong

The different steps of the connection setup can be seen in figure 8. To clarify, figure 8 depicts the wifi connection setup procedure specific for Android Pong. The Wifi direct technology itself is not dependent on a defined client, server relationship. The setup procedure described in more detail:

Discovery	The two devices goes into discovery by broadcasting their name while simultaneously listening for other devices. This step can happen in any particular order. The client does not necessarily have to wait for the server to start its discovery, it can go into discovery mode at any point in time.
Connect	The connection between two devices can only be initiated from one side. A simultaneous connection from both devices will cause the wifi direct procedure to break down. This is the reason why Asteroid Pong will only allow the client device to initiate connection to the server device and not the opposite.
Great Conversation	The great conversation is handled by the network module of the project. It is a conversation of great depth and importance. Although, maybe a bit one sided.

3.1.4 Network

As the wifi-direct communication solution removes the need for a third party access point, a need to promote one of the phones to server, and one to client arises. In order to resolve synchronization issues which come up as a result of delay between the two peers, it was decided to let the server hold the model and make all decisions regarding logic such as ball collision with paddle, which peer won the game, ball positioning and velocity, etc. The model used can be seen i figure 9.

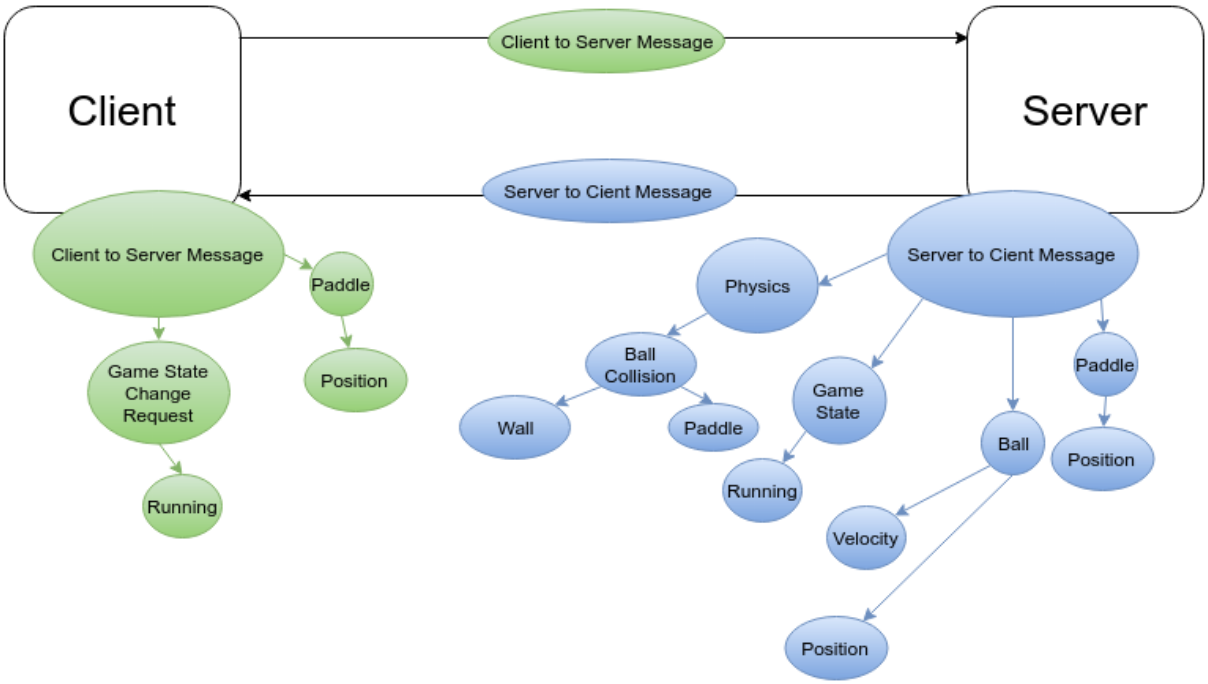


Figure 9: The network structure for messages between the client and server

To get a network experience as lag free as possible, a number of design decisions had to be made for the network design.

Transmission Protocol
The client and server where implemented using UDP instead of the more common protocol TCP. UDP is a connectionless protocol first defined in [12]. It mainly provides port multiplexing, and most importantly (compared to TCP) does *not* provide ordering of packets or retransmissions of lost packets. Ordering of packets would cause delayed packets to block non-delayed ones causing lag, and the real-time nature of the application causes old packets to contain false information meaning retransmissions would be harmful.

Keeping information fresh
As UDP does not provide ordering of packets, one might in the case of delays still receive packets containing information which is no longer valid. Thus, in each packet sent, a timestamp is included, and only packets received with a timestamp which is more fresh than the previous one will be accepted.

Ghost update and adjust
As the server holds the game model, for a long time (in the world of computers) the client will sit without information. Instead of simply waiting for new information from the server, the client runs the same game logic and updates the ball position as the server. This might be referred to as a ghost update, since the information is not used to drive the game model, but is there to make the game look smoother than it actually is. When the client finally receives a new message from the server, it overwrites all values, correcting any potential errors introduced between this and the previous update.

3.1.5 Sensor Input

Accelerometer
As moving the phone is used to control the user paddle, a way to register small changes in phone positioning is required. Tracking indoor positioning is a very hard problem, especially at high resolution. For example, in 2015 a new method in [13] is suggested for obtaining reliable indoor positioning in an Android phone down to "sub-meter accuracy", a resolution which is still far too poor for our controller. Most current systems for tracking indoor positioning uses the wifi signal of some nearby access point. Interestingly, in [14], a thesis on embedded systems, compares attempting to extract positional change from the acceleration data to that of wifi signal tracking. As expected, it finds that the tracking of wifi

gives a much more accurate result, however for low precision applications (such as our game) it does indeed seem that using the built-in accelerometer to register positional change is a viable option. In order to make a usable controller, the problem domain needs to be further understood.

The fundamental problem
Elementary calculus explains how the change in position over a time interval T corresponds to the accumulative velocity (i.e. the integral) over T. Further, the change in velocity over the same time interval is the accumulative acceleration over T. Thus, to calculate the change in position we need to integrate twice. This will inevitably lead to an error which, due to the accelerometer's inaccuracy, will propagate quickly with time (see for example the dashed lines in figure 10 showing how the velocity quickly goes to an inaccurate stationary point even though the phone is not moving).

Filtering
The acceleration signal provided by the android system is riddled with noise and therefore somewhat inaccurate. To remove the unwanted noise, the signal is low-pass filtered using a very short (number of samples used is $N = 8$) Fast Fourier Transform (FFT) low-pass filter. The low-pass implementation uses a Gaussian shape for the low-pass as suggested by [15]. The FFT implementation used sits in the Apache Commons Math library [16] (an open source project licensed under the Apache License, 2.0). To further make the curve more distinct, one looks at the rate of change of the acceleration and, if the average of the six last rate of change samples are less than some threshold, the acceleration value is set to zero.

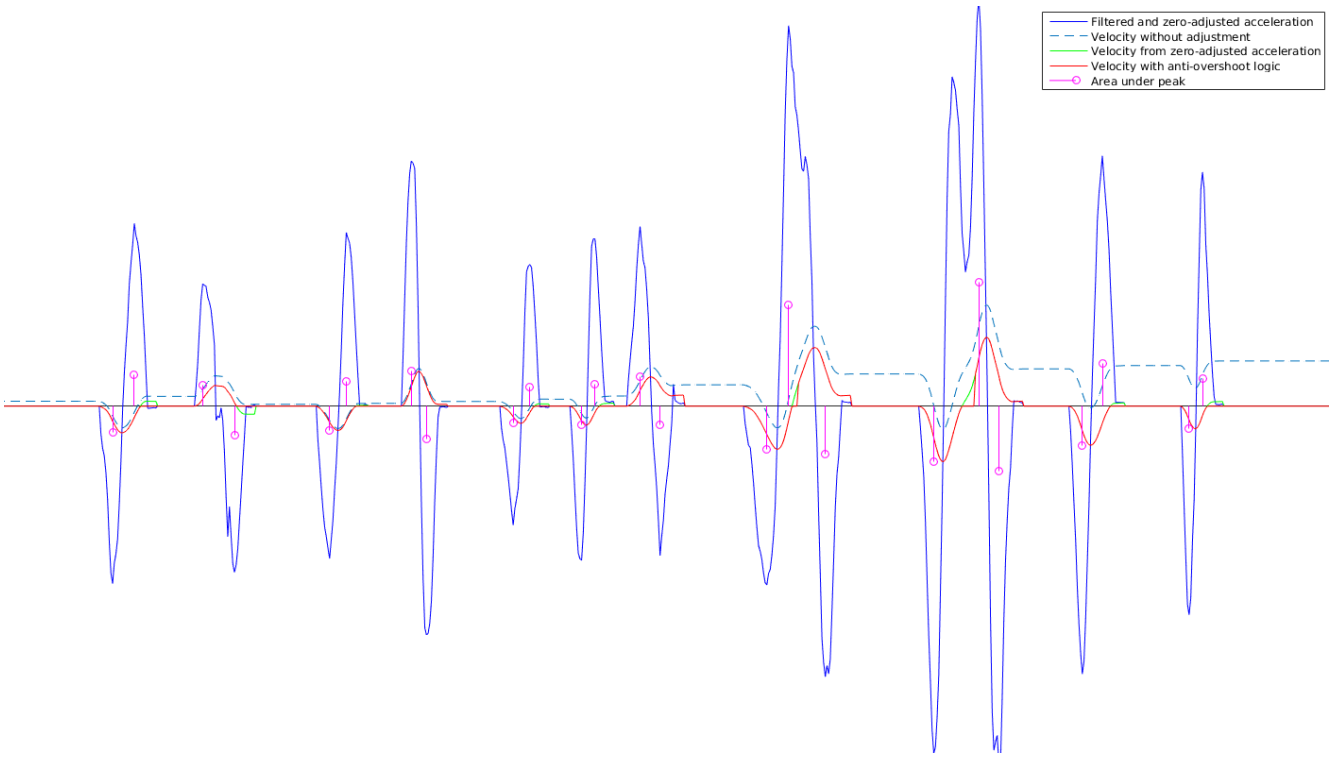


Figure 10: A visulization of actual real time data gathered from the Android application, and processed in Matlab

Looking at the unaltered velocity (the dashed line in the figure 10) it is apparent that there is a need to force the velocity back to zero. The easiest, and chosen way to achieve this goal is to set the velocity to zero when the acceleration returns to zero, resulting in the green line in figure 10 (note that the green line for the most part is covered by the red line when they coincide). Using this strategy has the downside of not allowing constant velocity.

Understanding user controller motion signature
A typical motion of an experienced user has the signature shown in figure 11a. The pattern is produced by moving the phone in a single direction and then stopping. It can be seen that ideally, this motion

gives an initial increase in velocity (while the acceleration is positive) and then the velocity is supposed to return to zero (the velocity goes back to zero as the acceleration shifts to a negative value). Due to inaccuracies in the received acceleration signal, the positive and negative parts of the acceleration will however not cancel out. Instead too much of either positive or negative is received. A small increase in velocity in the intended direction is no problem in our application since it is very hard for the user to notice the difference. However, an error in the opposite direction will cause the paddle to jump into the opposite direction (referred to as an overshoot) from how the user intended for it to go (notice how the green signal in figure 11a goes from positive to negative before returning to zero).

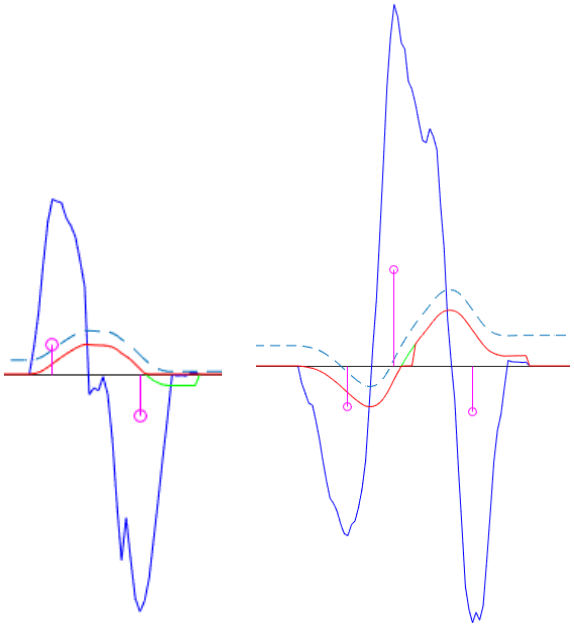
Counteracting the velocity overshoots
A matlab script (available at [17]) was written and used to aid in building control logic. Conceptually, the control logic works by:

1. Identifying when the user moves left or right
2. Not allowing velocity in the opposite direction until the phone has stopped.

As may be seen in both figure 11a and 10, the resulting red velocity signal goes to zero where the uncontrolled green velocity signal overshoots.

Allowing rapid movement back and forth
An issue with the selected solution is this; what if the user performs a back and forth motion? Then the control logic created will lock and not register the opposite (but very much intended) motion. This is solved by adding additional control logic:

1. Measure area under each peak and use as measure for peak size (height of the purple spikes in figures 10 and 11).
2. Check if the opposite acceleration size "dominates", i.e. is substantially larger than the initial one, then allow overshoot.
3. Temporarily disable the overshoot control logic to allow further rocking motion without the need for "domination".



(a) Single direction motion signature (b) Back and forth motion signature

Figure 11: The two most common user motion signatures

The result may be seen in figure 11b. It may be noted that the control logic initially forces the red velocity signal to zero, but as soon as the opposite acceleration "dominates" the initial one, the red signal is allowed to be non-zero again. The small gap produced by the slight delay is not noticeable to the user.

Hardware inconsistencies
The accelerometer exhibits slightly different behavior across different hardware platforms. So far it has been tested on a Nexus 4, 5, 5X, and Samsung Galaxy S7. The slight variations have been small enough to be ignored.

Microphone
The group quickly realized that it would be difficult to pause the game while in the middle of game play as the user has their hands busy with controlling the paddle. Because of this, it was decided that the microphone should be used to enable speech input from the user.

The Android Speech API [18] was used to translate the spoken input to a string. A method in the AndroidLauncher.java class then compares the resulting string with a few key words. For example the words "pause" and "stop" pauses the game while the words "resume", "play" and "start" resumes the game. To achieve a higher success rate a few close varieties of each word were also added as key words.

Continuous speech recognition
The Android Speech API did not really seem to be intended for continuous speech recognition which was crucial for the application, as a ”tap to listen” scheme would defeat the purpose of being able to pause the game without interacting with the screen. The main problem with continuous recognition was the fact that the service played a ”beep” sound every time it started listening (method startListening() in the ListenerActivity.java class) as well as when speech input was given. This resulted in a ”beep” sound close to once per second which became unbearable for the user.

As the API did not provide any solution or possibility to mute this sound the whole sound stream had to be muted. This did however create a new problem as it prevented the group from having any form of audio output as well as muting the audio from other applications. To minimize the effect of this inconvenience, the sound stream was only muted at the very time that the sound of the recognition service played. Although this still does not completely solve the problem as any background music becomes staccato, it allows shorter sounds to play most of the time.

3.1.6 Sensor Output (Feedback)

Sound
As the speech recognition service made it unfavorable to play any sound longer than a second the group was unable to implement sound output to any greater extent. The group experimented with different audio feedback whenever a planet was hit as well as when a game was finished with poor results due to staccato.

The group did however implement audio feedback when the ball collided with either a wall or one of the paddles. The sound was chosen to reassemble the sound a ball makes when it hits another object to further reinforce the conceptual model.

Vibration
Just as the audio feedback was constricted by the speech recognition the vibration feedback was constricted by the accelerometer. Initially vibration feedback was given whenever the ball collided with the users paddle to make it more clear that the ball hit the paddle properly. As the accelerometer implementation became more accurate the vibration feedback begun to interfere with the accelerometer data and thus made the paddle move unintentionally. It was then decided that the game play benefited more from reliable control of the paddle and the vibration feedback was removed. The application does however still provide vibration feedback whenever a game is finished. This vibration is almost a second in length and does further enhance the feeling of the game being over.

3.1.7 Graphical Design

The game’s graphical profile is somewhat inspired by classic arcade games mainly because the game idea is taken from the original Pong game. For example the title ”Asteroid Pong” (which is actually a combination of two classic arcade games, Asteroids and Pong) is displayed in a arcade inspired font on the main screen.

The game includes a few animations to enhance the visual feedback. For example When a second player is searching for created multiplayer games a ripple circle animation is displayed before the user can select a found game. This view can be seen in figure 12. The animations are rendered at a low frame rate which makes them feel more like they were displayed on a older arcade machine which fits very well with the games graphical profile.

The background layer is a dark blue color. Text, buttons and other game elements are white to maximize the contrast between background and foreground. They are also quite large to make them easy to see and use. All buttons (except the game pause button) are positioned at the lower half of the screen to facilitate one hand interaction with the device. Every time the application initializes 30 stars are rendered as small squared shapes at random positions on the screen. To make the background feel more alive stars are also removed and added to the screen in given intervals.

All graphical elements in the game are either predefined Shape objects automatically rendered by the LibGDX-framework or Texture objects which are graphical files placed in the Android assets folder. Moving game objects like the ball and the particles behind it are rendered as Shape objects with a fixed color. Other more visual objects like buttons, animations and planets are png-files created in Photoshop before added to the project. An example of how to code the render method to display a button as a Texture object can be seen below.

```
Texture button = new Texture("button.png");

public void
render(SpriteBatch spriteBatch, ShapeRenderer shapeRenderer) {
   spriteBatch.begin();
   spriteBatch.draw(button,
(Gdx.graphics.getWidth()/2), (Gdx.graphics.getHeight()/2));
   spriteBatch.end();
}
```

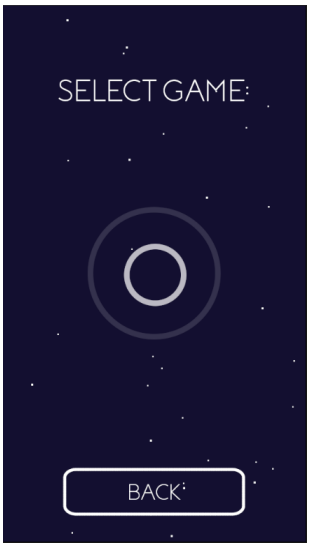


Figure 12: Screenshot of the view when searching for created multiplayer games

During the gameplay the user needs some kind of feedback representing the game score. In Asteroid Pong the score is the planet's "life". In an early iteration of the game, the planet life was displayed in digits above each planet. During one of the supervisory meetings where the prototype was tested, the group discussed the possibility to change the visualization of the planet life from digits to some kind of colored life meter. To avoid covering unnecessary game area, the group decided that it would interesting to place the life meter as a stroke on the planet's border. In figure 13 three different states are shown. The one on the left indicates 100% of life remaining set in a green color. The one on the right indicates 25% life remaining set in a red color

Figure 13: Different states of the planets life meter

The buttons are designed so the overall surface are transparent so they don't cover more than necessary of the background. Basically they only consists of text and a surrounding border. From the main menu the user has three different options; "Create Game" at the top, "Find Game" in the middle and "Training Mode" at the bottom. The buttons are placed in the order that a user would most likely use them (the most likely at the top). To setup a multiplayer game, consisiting of two players, one player must first create a new game with a chosen game name. The second player can thereafter search for the created game and select it to setup a connection between the devices. How to set up a new multiplayer game acting as a server can be seen in in the user flow diagram in figure 14 and includes the following steps:

1. Press the "Create Game" button.
2. Type your game name and wait until a second player has connected. Read the instructions and thereafter press the "start game" button when possible.
3. The game is running, survive as long as you can!

4. You survived longer than your opponent and therefore won. Either press the ”Exit” button to return to the main menu or press the ”New Game” button to start a new game.

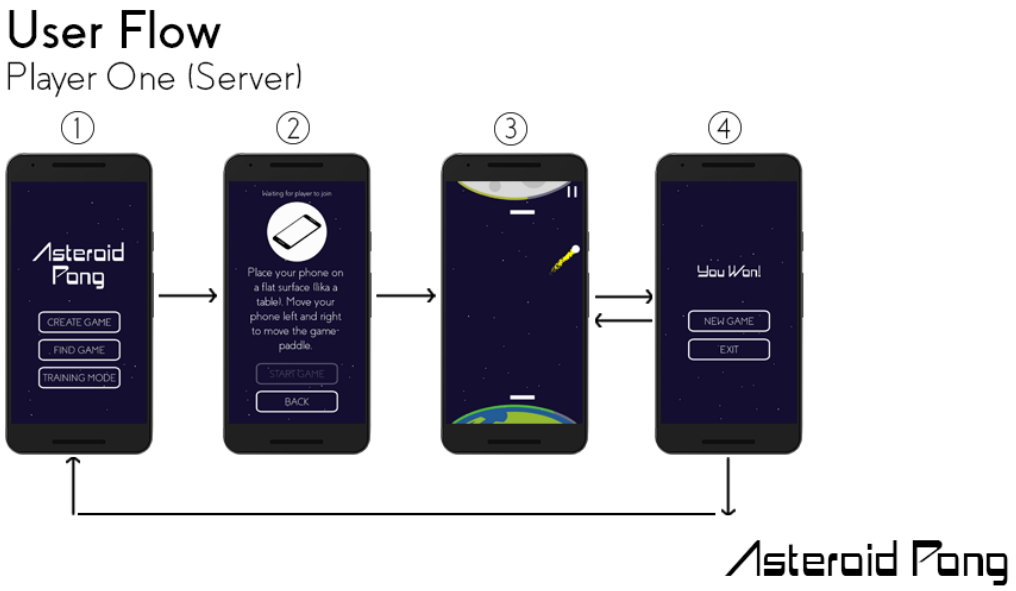


Figure 14: User flow diagram of how to create a multiplayer game

The second player can thereafter search and select a created game as a client seen in the user flow diagram in figure 15 and includes the following steps:

1. Press the ”Find Game” button.
2. Wait until your device has found a game. Thereafter press the shown game name to connect.
3. Read the instructions and thereafter press the ”Start Game” button when possible.
4. The game is running, survive as long as you can!
5. Your planet was destroyed and therefore you lost and the game is over. Either press the ”Exit” button to return to the main menu or press the ”New Game” button to start a new game.

Figure 15: User flow diagram of how to connect to a created muliplayer game

When initializing a multiplayer game it is important that both players are ready to play when the game starts. Therefore a synchronized countdown is displayed on both devices to give visual feedback on when the game actually starts. To accomplish this, we used a countdown as as can be seen in figure 16. The countdown is displayed with one digit at a time in order 3-2-1 before the game starts and the asteroid starts moving.

Figure 16: Synchronized countdown displayed before the game starts

The most common way to interact with smartphone games is through screen input. By moving away from this, Asteroid Pong faced the challenge of controller discoverability. Therefore, prior to starting the game, the user is given information to place their device on a flat surface and to move it left and right to move the game paddle. A simple animation of a phone moving left and right is also provided for clarity. This is depicted in step two in the user flow diagram in figure 14. Likewise, at step three in the user flow diagram in figure 15.

3.2 Usability Test Output

The results of the usability test showed that although all of the users were overall satisfied with the prototype a few areas could be improved. In retrospect the test did not provide that much more feedback than the usual supervisory meetings.

3.3 Evaluation Test Output

The results from the evaluation test questionnaire can be seen in figure 20 and figure 21. The evaluation test provided the group with a lot of valuable feedback. As it was the first time the application was tested on a large scale (30 answers received) it became clear what functionality had to be improved and what was good. For example it became clear that, even though it only took at most a second to for example start a game after the ”start game” button had been pressed, most users pressed the button two or three times. Despite the fact that only one user commented on this phenomenon it became clear to the group that some of the buttons would benefit from more *clickable affordance* as well as to give feedback when clicked. The group also noticed that some users were eager to play and therefore skipped the given information on how to control the game paddle which lead to confusion how to control the game.

Figure 17: Evaluation test on the final exhibition

4 Discussion

4.1 Usablility Tests

The group had a clear vision of the basic functionality that had to be present in the application and limited amount of time available to implement it all, thus initially a minimal amount of effort was put into testing the application and the supervisory meetings were used as the main compass for guiding the iterations. The first usability test was an informal one with only four persons. Although questions with graded answers were provided in the first usability test, no sound statistical basis of the game performance could be achieved since the number of testers were too low. One could argue that a paper prototype should have been prepared for the game menus along with a more in depth questionnaire to allow earlier stage usability testing. Conversely, the second usability test was more formal in nature. A higher number of testers were available and a larger set of questions were presented in order to evaluate as many aspects as possible of the application presented during the final exhibition. The output from the second usability test was of great value to the group. The application presented was much more fleshed in comparison to the one presented during the first usability test, which made a larger number of features available for testing. Furthermore, the test preparation was more thorough than the first one. All this resulted in a very successful usability test during the final exhibition. One thing that could be improved in both the usability test and the evaluation test is to include testers from a more diverse background. More or less all the people that tested the application can be assumed to come from a homogeneous background when it comes to technical ability. If more time had been given, the group could have benefited from more usability tests with a more diverse test population.

4.2 Graphical Design

Although the game is an Android application, the menu design does not follow Google's navigation guidelines for pure Android applications [19]. The reasoning behind this is that Asteroid Pong is a cross platform game application rather than a traditional smart phone application. The android navigation bar, usually located at the bottom of the screen has therefore been hidden. The game is presented in absolute full screen mode, and navigating back and forward is done through the game menu buttons.

The result from the usability test at the final exhibition, see appendix figure 20 and 21, indicated that the majority of the people testing the application did not have any problem navigating and setting up a new multiplayer game. Furthermore, in the overall feedback people commented that the graphical design was very aesthetically pleasing.

Although the overall result was generally positive some graphical design changes were suggested. For example the buttons should give feedback when pressed to avoid users pressing the buttons repeatedly. This could be fixed by changing the buttons' color momentarily, or adding sound feedback when pressed.

Improvements on the screen with information on how to play before starting the game was also discussed at the final exhibition. One suggestion was to let the player somehow try the controller before starting the game. One idea was to replace the animation on the screen with a smartphone icon with arrows on each side and when a user moves the device physically the smartphone icon on the screen would move in the same direction.

The "Start Game" button also confused some players hosting a game leading them to believe that they had to press it to start the server and receive a connection from another player searching, creating an unintentional *logical constraint*. Therefore this button should be changed from "Start Game" to "Ready" to better communicate the conceptual model of mutual agreement for starting the game.

4.3 User Interaction

4.3.1 Interaction Between Users

Even though a user is bound to keep their eyes on the screen during gameplay in order to keep track of the ball, we observed that the players started interacting on a meta level. For example, friendly taunting, laughing, and a will to beat your opponent which really only comes to life when you compete face to face (paddle to paddle).

4.3.2 Speech Input

Additionally apart from the menu screens a user can play the game without giving any input through the touchscreen. One possible improvement that would completely remove the touch interaction would be to implement speech recognition in the menu screens as well. However, as the Android Speech API was not as good as the group had hoped this would require a third party API.

4.3.3 Input Controller

An implementation choice for the controller was made to set the velocity to zero when the acceleration was zero. Using this strategy however has the downside of not allowing constant velocity. As seen when analyzing the movement signatures, this is not a major issue for experienced players as a constant velocity movement is not a typical movement signature for this type of user. However, some users, especially those with poor motor skills or non-frequent mobile use seemed to initially drag the phone at slow and constant speed. Thus, for later iterations of the application this is an area that should be focused on to make it easier for new players to pick up the controller. This is further implied by the fairly large evaluation performed at the fair (30 answers with only uninitiated users) in which the users rated the controller's ease of use at approximately 2.5 out of 5. Observing such a large number of people attempting to control the paddle led to a couple of conclusions:

- Users immediately understand the concept of moving the phone to move the paddle.
- Users find it initially hard to control the paddle, and in a few *extreme cases*, the users felt absolutely unable to make it move to their desire, frustratingly shaking the phone from side to side without it responding.
- After a tip from the developers on how to control the paddle, most users quickly became "experts" at controlling the paddle and found it quite easy and intuitive.

Thus, the controller brings with it an intuitive conceptual model (move the phone to move the paddle) but is not sensitive enough to allow non-expert use. However, the very rapid increase in skill with additional information on how to best move the controller shows that becoming an expert is generally not very hard if one knows the proper way to move the paddle. A minimal acceptable improvement would thus be to include some tip on the start screen on how to move the controller, or perhaps create a small tutorial to help new players get initiated. Ideally though, a way to allow constant acceleration might mitigate this need and would substantially increase the usability of the controller.

5 Conclusion

The group's objective was to utilize the various sensors found on an Android device in order to achieve beyond the screen interaction. During the brainstorming session, it became apparent that it was far more difficult than expected to come up with innovative and functional ways to interact with a smartphone application. Many imaginative ideas were generated during the brain storming session, but only a few of them proved implementable due to hardware and time constraints. When deciding on a Virtual Pong game, the control scheme (utilizing the accelerometer) felt like a novel way to interact with a smartphone game application. Moreover, the initial scope of the project seemed well defined since not an overwhelming number of features had to be present in order to demonstrate a working proof of concept. The iterative design process proved successful. Having each iteration end with a supervised meeting pushed us to present new features each week. With the final prototype, we managed to fulfill most of the goals set up during the project start. Even though we did not achieve the one-to-one mapping, we were still very pleased with the final outcome. The response to the final exhibition prototype was overwhelmingly positive. This confirmed our belief that the controller was not ideal, but that the overall application experience is one that we can be very proud of.

References

[1] Brian K Smith. Bodystorming mobile learning experiences. *TechTrends, Volume 58, Issue 1*, pages 71–76, 2014.

[2] Tangible interaction (interaction design foundation). https://www.interaction-design.org/literature/book/the-glossary-of-human-computer-interaction/tangible-interaction visited 2016-05-25.

[3] Graspable user interfaces. http://www.dgp.toronto.edu/~gf/papers/PhD%20-%20Graspable%20UIs/Thesis.gf.html#RTFToC6 visited 2016-05-26.

[4] Donald A. Norman. The design of everyday things. Basic Books, 2013.

[5] Atari history. https://www.atari.com/history/1972-1984-0 visited 2016-05-25.

[6] J.M. Carroll. Five reasons for scenario-based design. *Interacting with Computers 13*, pages 43–60, 2000.

[7] Space x. http://www.spacex.com/about visited 2016-05-26.

[8] Y Rogers, H Sharp, and J Preece. *Interaction Design*. John Wiley and Sons Ltd, 3rd edition, 2011.

[9] Roger S. Pressman. Agile manifesto. http://academic.brooklyn.cuny.edu/cis/sfleisher/Chapter_03_sim.pdf visited 2016-04-19.

[10] Bad Logic Games. Libgdx. https://libgdx.badlogicgames.com/ visited 2016-05-26.

[11] Wi-Fi Alliance. Wi-fi direct. http://www.wi-fi.org/discover-wi-fi/wi-fi-direct visited 2016-05-25.

[12] User datagram protocol, rfc 768. https://www.rfc-editor.org/rfc/rfc768.txt visited 2016-05-25.

[13] P. Pajuelo, M. C. Perez, J. M. Villadangos, E. Garcia, D. Gualda, J. Urena, and A. Hernandez. Implementation of indoor positioning algorithms using android smartphones. *2015 IEEE International Conference on Cyber Technology in Automation, Control & Intelligent Systems (CYBER)*, page 1, 2015.

[14] Angel Rodriguez Ubejd Shala. Indoor positioning thesis. http://hkr.diva-portal.org/smash/get/diva2:475619/FULLTEXT02.pdf visited 2016-05-23.

[15] Stack overflow; gaussian fft filter. http://stackoverflow.com/questions/28814547/matlab-low-pass-filter-using-fft visited 2016-05-23.

[16] Apache commons math library. http://commons.apache.org/proper/commons-math/download_math.cgi visited 2016-05-23.

[17] Code for analyzing the controller. https://github.com/HampusBalldin/AccelerationToPositionFFT visited 2016-05-24.

[18] Google. Android speech. https://developer.android.com/reference/android/speech/package-summary.html visited 2016-05-24.

[19] Android design guidelines; navigation. https://developer.android.com/design/patterns/navigation.html visited 2016-05-24.

6 Appendix

Idéer, interaktionsdesignprojekt:

- *En-hands-tangentbord:* - Enkelt att skriva text med hand. Välj första bokstav genom tryck eller röst och välj autogenererade ord med gester (accelerometer).
- *NFC-tags-orientering:* - Gör utmaningar (typ tipsrundafrågor) som ”taggas” på en karta. För att ”låsa upp” frågan måste en NFC-tag hittas och läsas av.
- *Delade fotoalbum:* - Tjänst för att enkelt skapa delade fotoalbum mellan användare med Snapchat-inspirerad bildspelsfunktion.
- *Virtuell Pong-spel:* - Två spelare med telefoner fungerar som ”verkliga markörer”. Två andra ser spelplanen och den virtuella bollen och styr spelarna genom anrop. Spelplanen är rummet som mäts upp i rummet eller bara ett bord. Alternativt används ett bord som spelplan.

Figure 18: The questions for the first usability study

Evaluation Form - Asteroid Pong

Data gathering during the exhibition in MAMN01 - 24th of May, 2016

How easy is it to setup a new multiplayer game?

	1	2	3	4	5	
not easy	○	○	○	○	○	very easy

How easy is it to understand how to win the game?

	1	2	3	4	5	
not easy	○	○	○	○	○	very easy

How easy is it to understand how to control your paddle?

	1	2	3	4	5	
not easy	○	○	○	○	○	very easy

How easy is it to control your paddle?

	1	2	3	4	5	
not easy	○	○	○	○	○	very easy

How fun is the game to play?

	1	2	3	4	5	
not fun	○	○	○	○	○	very fun

How stressful is the game to play?

	1	2	3	4	5	
not stressful	○	○	○	○	○	very stressful

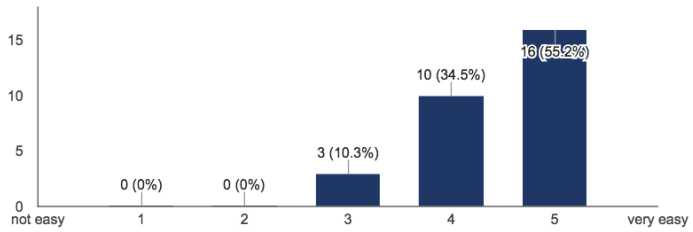
Additional feedback?

Your answer

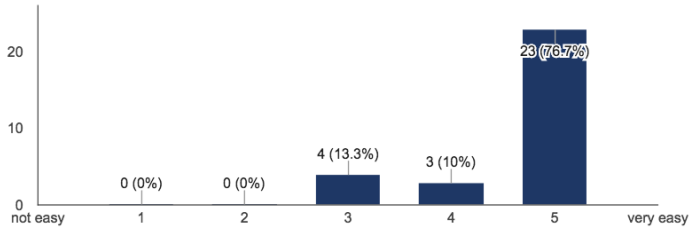
SUBMIT

Figure 19: The questions for the final usability study

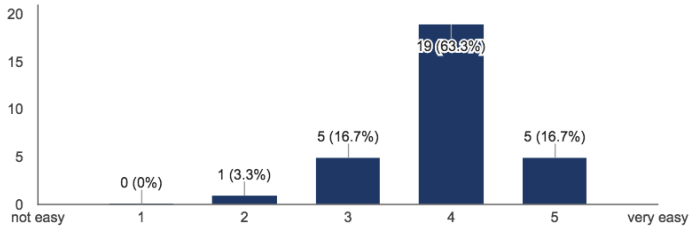
How easy is it to setup a new multiplayer game? (29 responses)



How easy is it to understand how to win the game? (30 responses)



How easy is it to understand how to control your paddle? (30 responses)



How easy is it to control your paddle? (30 responses)

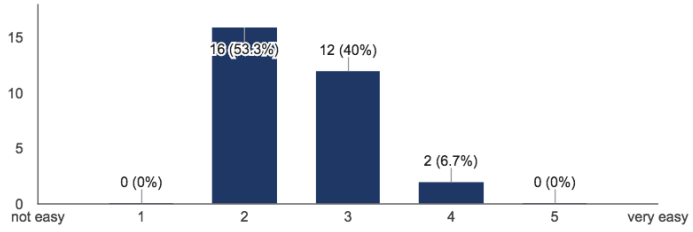
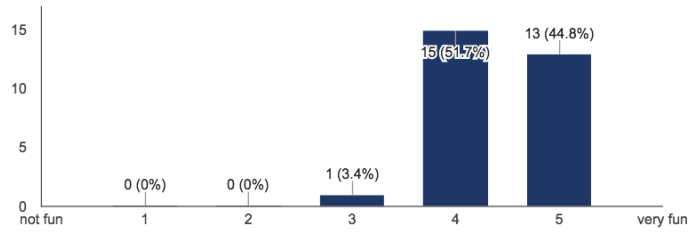
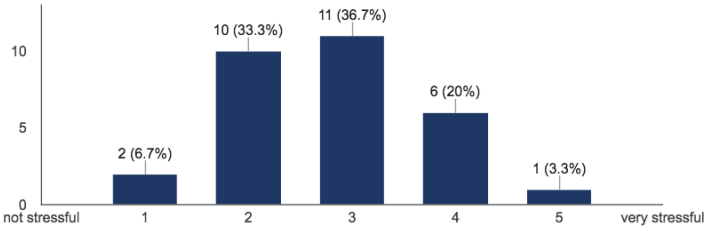


Figure 20: Part 1 of the result from the evaluation test

How fun is the game to play? (29 responses)



How stressful is the game to play? (30 responses)



Additional feedback? (17 responses)

Snygg design
Snygg design
Snabbare boll. Kanske upplysa om den nödvändiga snärtigheten? Ändra Start game till Ready.
Väldigt snyggt. Kul innovativ styrning som gör spelet unikt.
Mycket snyggt GUI
Skulle kunna ha poängräkning så man vet hur mycket man vinner med ;)
Snygg app! Diggar att livet visades på planeterna. Skulle dock kännas mer naturligt att ha telefonen liggandes, som en paddel, men då blir det ju också svårt att visa spelplanen.
Asbra
Fin grafik
Roligt men komplicerat att flytta racket. Kräver en del utrymme på bordet och svårt att få jämna kontrollerade rörelser.
In the intro screen maybe add an example for how to properly move the paddle
Grym
Snyggt jobbat!
Skitsnyggt! Och kul💆
Lite svårt att styra ibland. Kanske borde öka hastighet om ett spel tar lång tid
Snyggt!!!
Snyggt, bra idé!

Figure 21: Part 2 of the result from the evaluation test