Digital Learning Games

Digital Technologies

Tallinn University

CONCEPTUAL DESIGN AND DEVELOPMENT OF A SERIOUS GAME FOR LEARNING TOPOGRAPHICAL MAPS

Master thesis

Author: Indrek Kaine

Instructor: Martin Sillaots

Author: …………………………………………………………… ,, ……….. ,, 2018

Instructor: ………………………………………………………… ,, ……….. ,, 2018

Rector of the Institute: ……………………………………………… ,, ……….. ,, 2018

Tallinn 2018

Declaration

I hereby declare that I have written this thesis by myself and without support from any other person or source, and I have only used the materials and sources indicated in the list of work cited. Neither I myself nor any other person has submitted this to any other institution for a degree or for publication.

Indrek Kaine

……………………

Table of contents

_Toc513180146

Abstract

There is too little or no digital learning material about topographical maps which use techniques of repetition and instant 3D feedback. This paper introduces a digital learning game prototype that satisfies these needs and also provides description of the building processes.

Introduction

During the course of Serious Game Workshop, Digital Learning Games (DLG) curriculum in Tallinn University, there was an assignment to pitch ideas for games with serious content or learning purpose. Ilya Dotšar, a DLG student, presented a problem that it was hard for him to learn the symbols and *contour lines* (the lines which connect the points that have the same height) of topographical maps in school, and thought that a digital learning game would help solving the problem.

Kaarel Kallas (2015) stated the same problem in his research "The Implementation of Orienteering in the National Curriculum of Elementary and Secondary Schools" where he conducted an interview with gymnastics teachers of primary and high schools. His research states that 22% of the questioned teachers think that pupils do not acquire enough theoretical knowledge about using a compass and a map in geography lessons to support conducting orienteering lessons in gymnastics.

To solve this problem, a concept was formed for the digital learning game to teach topographical symbols, scaling and how to orientate using the map. Its main idea was that the player himself creates the map and by doing repetitive actions like drawing the contours and adding different symbols, the meanings of the markings would be memorized.

The concept was called "Topographics", and it won the 4th prize in Cross Motion Tallinn Conference and Hackathon 2016, the meeting place for media and gaming industries with the health, tourism and educational sectors to spark ideas how to bring innovative gaming and audiovisual solutions to the service of the mentioned sectors (Cross Motion, 2016).

Since the concept got positive feedback, it was decided to conduct a design-based research to develop a game prototype which should effectively teach the topic. The goal of the research was to find out how to build the game so that it would efficiently teach the symbols and scaling of topographical maps. The Spiral methodology (Bohem, 1988) was used to set deadlines for the research, enhancing the concept and production. The

MoSCoW (Clegg & Barker, 1994) method was used to set priorities for the first prototype.

The research was conducted in five cycles: 1) finding similar interactive learning materials and software and adjusting the existing concept; 2) determining teachers' interests if they used the game in their lessons; 3) collecting existing data and code samples to start building the prototype; 4) test the gathered data, and; 5) develop the first prototype.

During these phases, various tools were required to reach the goals. To keep track of the whole process, the progress was recorded to a diary. A research group was formed whose responsibility was to interview the teachers of Geography and National Defense. A usability scenario was formed to help explaining what the game's functions are and how to use it. Meetings were arranged in order to gather topographical data. A prototype was developed using game development software Unity and if any problems occurred, PDSA cycle (Deming, 2000) was used to solve them.

The personal goal of this study is to enhance practical skills on building digital learning games. While developing the prototype, the finalized product can be useful to "The Estonian Lifelong Learning Strategy 2020" and its program "A digital focus in lifelong learning" for the encouragement about creating digital learning materials and making them available in order to support the teachers' and students' digital competences (Rosenblad et al., 2017).

1. Theoretical background

The purpose of the current research is to find out how to build a digital learning game that teaches the symbols of topographical maps effectively. In order to achieve this, firstly there was a need to understand what determines the quality of teaching the topic and how to implement the knowledge into the design. Secondly, develop the design according to the target group. Thirdly, find out the most efficient way to build the prototype.

1.1 Research

The game could be a solution to the problem which Kaarel Kallas (2015) points out: 22% of his questioned teachers, who teach gymnastic in primary and secondary schools in Estonia, state that pupils do not acquire enough theoretical knowledge about using a compass and a map in order to conduct orienteering lessons. Furthermore the majority of the interviewed teachers (68.4%) wish to receive schoolings in computer programs which teach orienteering and 25.3% of all the teachers wish to have trainings on programs that can be used to map the surroundings of their school, because if the school has orienteering maps of the surrounding areas, they would be outdated. (Kallas, 2015) Thus, as a future development for the prototype, the drawn maps could be printed out.

“Learning and Teaching Maps” (Wiegand, 2006, p. 59) restates that “the teaching of a contour-map reading is generally regarded as one of the most difficult problems the geography teacher has to face”. To explain this, Wiegand points out the Liben and Downs (1989) report which brings out that the performance was generally poor, when “children in grades 1 and 2 (Y2 and Y3) were asked to match flags positioned at a number of points on the model with small stickers on a map which showed the same area but at a smaller scale”. Furthermore, Wiegand explains that “although children at about age 11 begin to interpret simple, discrete landforms, the difficulties increase when the contours are open and unresolved at the map edge or where the landscape does not match a known ideal.”

Similar statements and articles, that describe the current situation about topographical knowledge among pupils in middle and high school of Estonia, were searched for using keywords “topography”, “orienteering” and “maps”, but nothing relevant was found.

1.2 Other games

Currently the most prominent available interactive learning materials on the topic are divided into two: focusing on orienteering or teaching the symbols, lines and scaling of a topographical map.

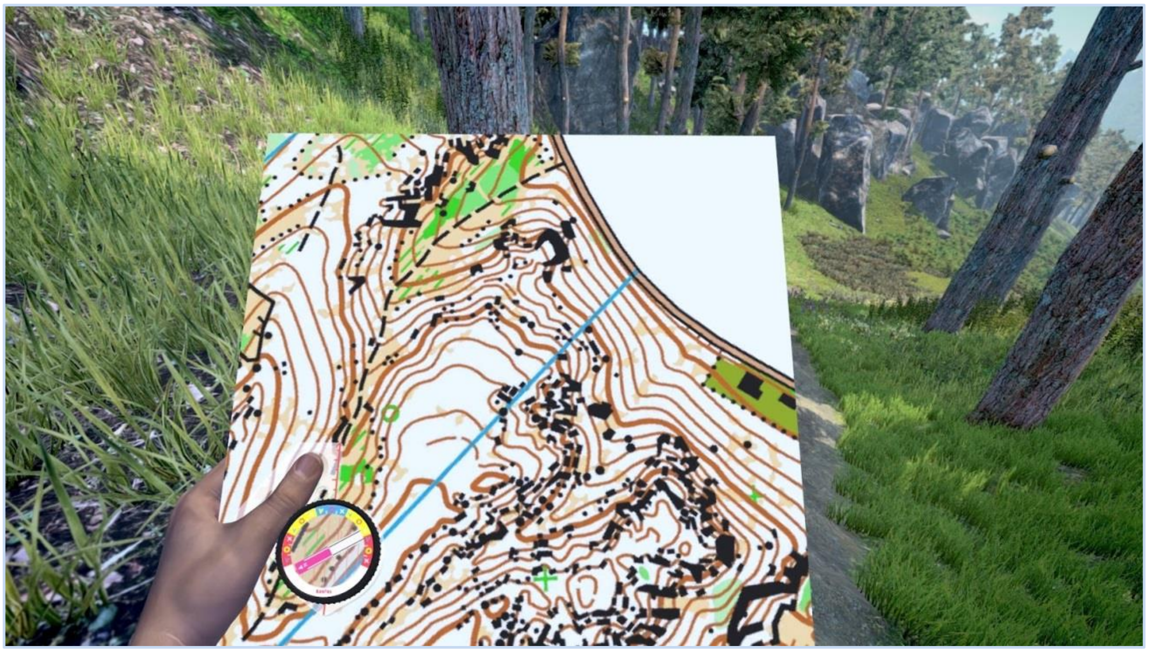
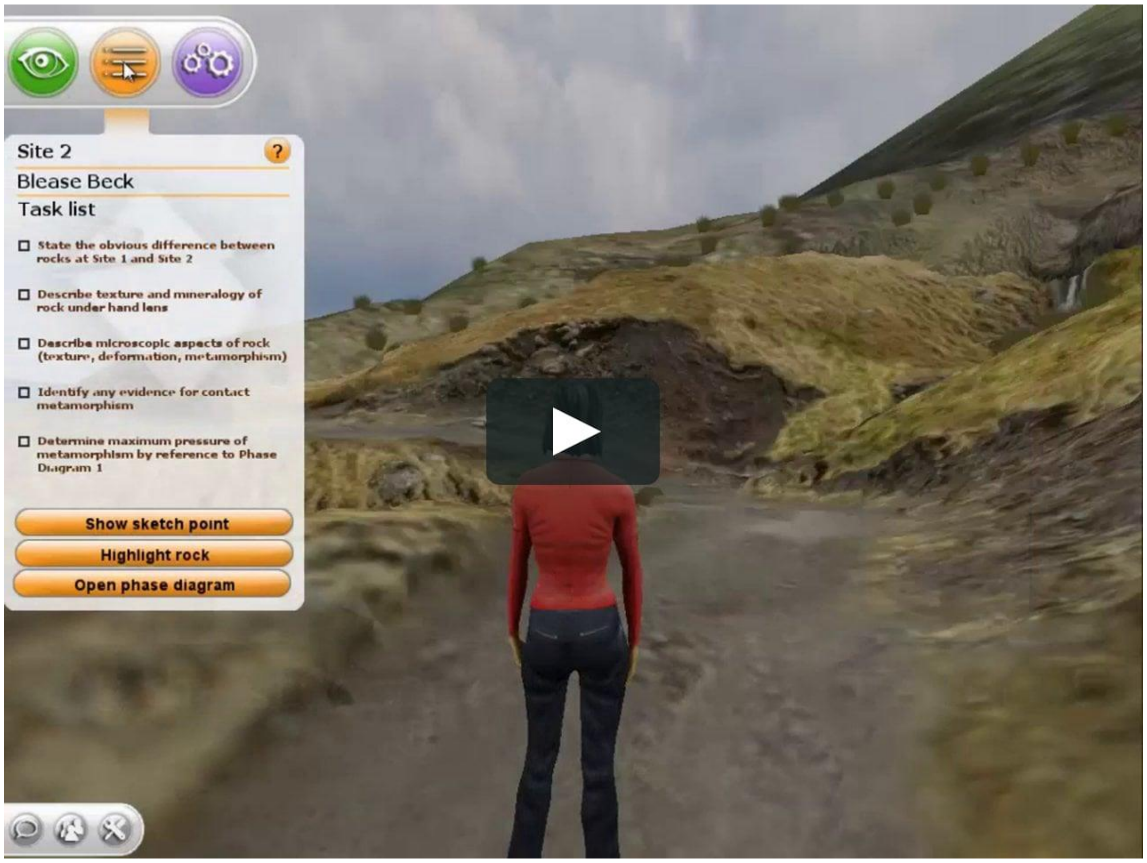


Figure 1. Screenshot of “VirtualO” (Contours, 2016)

In example “VirtualO” (Contours, 2016), “Suunnistussimulaatori” (Pulli, 2014), “Catching Features” (“Catching Features,” 2008) and “Virtual Skiddaw” (Argles, Minocha, & Burden, 2015) simulate orienteering in 3D environment, but expect the user to already know the basics. “Virtual Skiddaw” differs from the others, because its purpose is to simulate a geology field trip: i.e. gather geological data of soil and rocks in the given area to solve tasks using the gathered info. Orienteering from one checkpoint to another can be skipped completely. “Catching Features”, “Suunnistussimulaatori” and “VirtualO” have the same purpose: provide an orienteering experience while using a topographical orienteering map and a compass to navigate between checkpoints. The time to finish the course is being recorded and is dependent on which route the player chooses to reach the next checkpoint, because running in water, uphill or in hardly passable terrains like swamps slows the player down. “VirtualO” is the newest developed orienteering game with the most advanced aesthetics, but does not yet have the multiplayer functionality like “Catching Features” and “Suunnistussimulaatori” have.



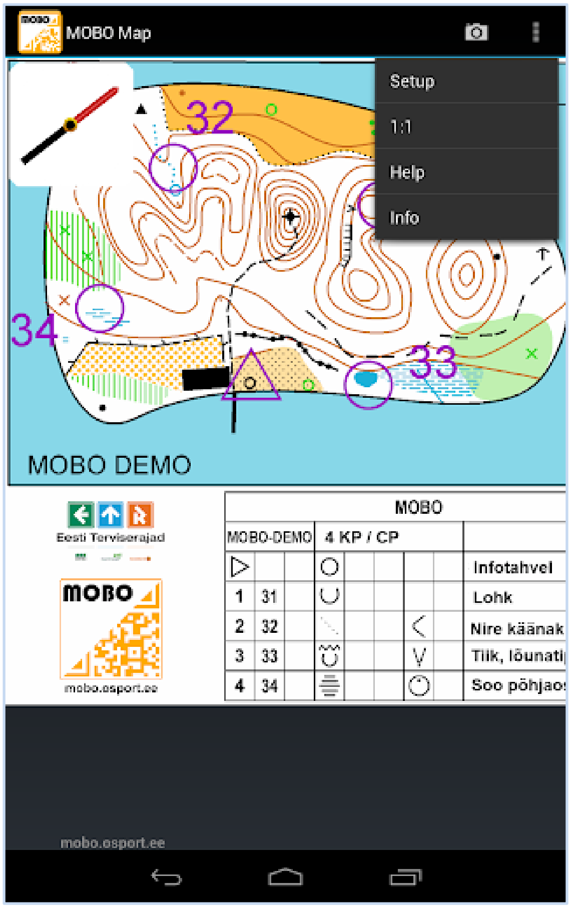
- State the obvious difference between rocks at Site 1 and Site 2
- Describe texture and mineralogy of rock under hand lens
- Describe microscopic aspects of rock (texture, deformation, metamorphism)
- Identify any evidence for contact metamorphism
- Determine maximum pressure of metamorphism by reference to Phase Diagram 1

Figure 2. Screenshot of the introducing video of “Virtual Skiddaw” (Argles et al., 2015)

The learning materials that teach markings, lines and scaling, such as “Reading a Map” (WebRangers, n.d.), use 2D images and animations to pass the idea how the contour lines work and how to use the legend to find out the meanings of the symbols, but the interactive part is a quiz; or, considering younger generations, completely leave out the formation of landscape and scaling to only concentrate on passing the general idea how a map works: “Map Maker” (Riley, 2008).

1.3 Existing softwares

During the research a few noteworthy softwares were found that might be useful to anybody interested in topographical maps or orienteering. “MOBO” (Klaar, n.d.) is a free orienteering software for mobile phones which contains topographical orienteering maps of real-life existing places. Using the mobile’s in-built compass and the map on the screen, the user can orientate through the course’s checkpoints and post the results online. Another program is “OCAD” (AG, 2018), a commercial software developed to draw topographical, orienteering, regional and other types of maps. The software has widely used symbols and even can be used to create custom map markings. Usage of this product however expects the user to know how a topographical map works and does not render a digital model of the *terrain* according to the created map, thus it might not be a suitable tool for learning.



MOBO								
MOBO-DEMO			4 KP / CP					
▷			○					Infotahvel
1	31		∪					Lohk
2	32		⟍			<		Nire käänak
3	33		∪			V		Tiik, lõunati
4	34		≡			◌		Soo põhjao

Figure 3. Screenshot of “MOBO” (Klaar, n.d.)

2. Methodology

This research bases on the design-based research methodology (Brown, 1992; Collins, 1992) and the Spiral model (Boehm, 2007) was used to develop the prototype. Then during the production phase, the MoSCoW method (Clegg & Barker, 1994) was applied to set priorities for the prototype, and many tools were useful throughout the research.

2.1 The Design-Based Research methodology

In order to get more information and practical knowledge on creating a digital learning game; find out if and how the needs of the stakeholders shape the outcome; and see which patterns help developing a digital game, the design-based research methodology (Brown, 1992; Collins, 1992) was chosen for this study.

Design-based research is an emerging paradigm for the study of learning in context through the systematic design and study of instructional strategies and tools (Collective, 2002). In this research, the context was dictated by the interests of stakeholders, existing interactive programs and learning materials and the samples of authentic areal data. With these influences, a concept was formed which is the first step towards the design of the game. While developing the prototype according to the design concept, it was studied how the problem solving tool works for creating the functionality.

2.2 The Spiral model

The development phase was decided to be conducted by a team of one person, the author, who has almost no experience with the chosen game development platform or C#. Due to this, there was a high risk of not being able to deliver the prototype in time, or the prototype would lack significantly in core functionality. Although some risk was reduced by the author's programming skill JavaScript programming language, the risk level remained high since if the game had to be reprogrammed, it would take immense amount of time to make the changes. Thus there was a strong need to prioritize and form a strategy in order to reduce risk level and the most suitable methodology to use was the Spiral Model of Software Development (Boehm, 2007).

The Spiral model (see Figure 4) describes software development as cycles of similar actions. The first cycle starts with determining objectives, alternatives and constraints.

Then proceed to evaluate the info and form a risk analysis, according to which the concept of the first prototype can be formed. This concept can be shaped into a paper prototype, wizard of Oz experiment, or similar simple tool that can be tested on the target group to verify if the concept satisfies the objectives that were formed at the beginning of this cycle.

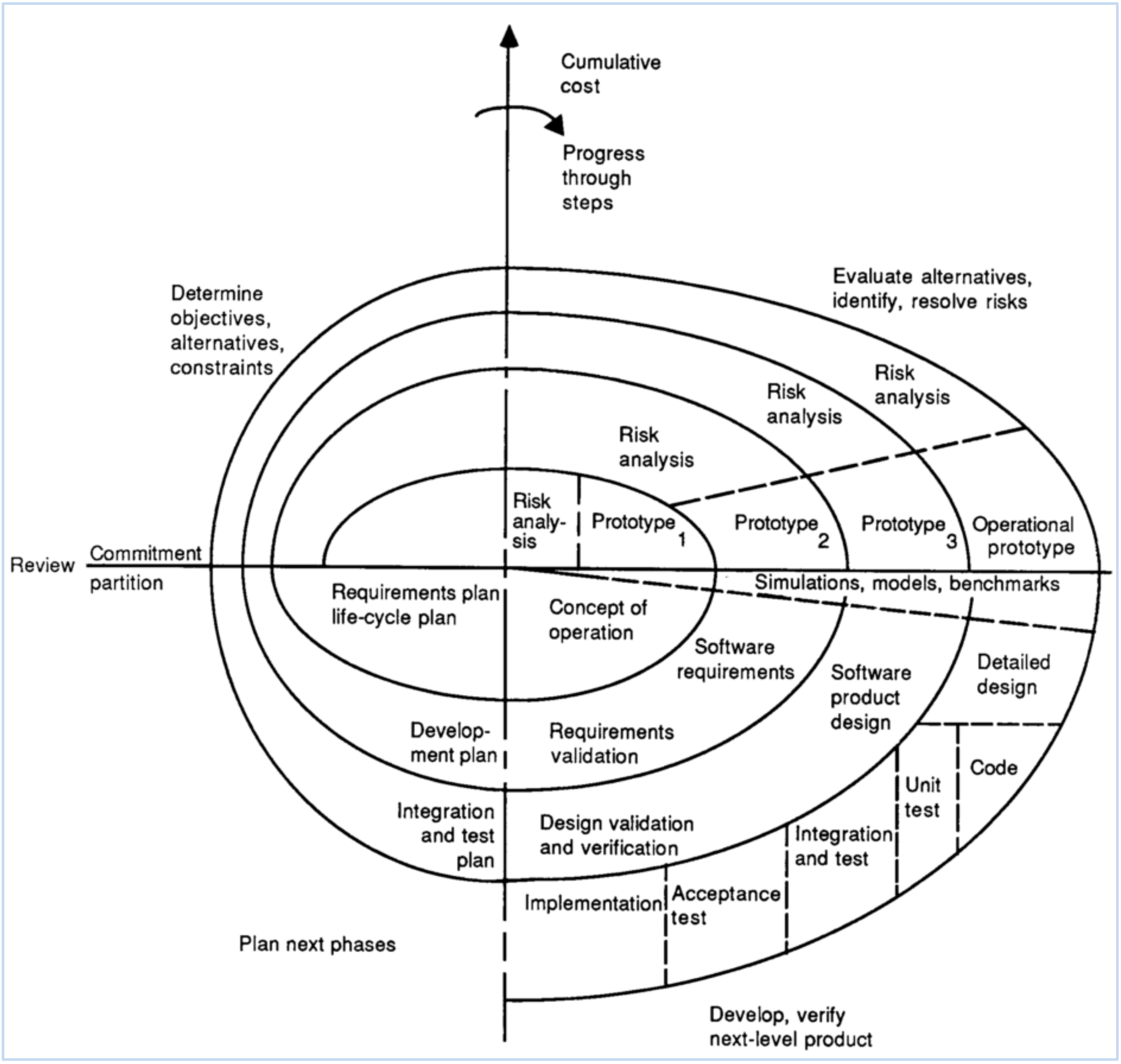


Figure 4. Spiral model of the software process (Bohem, 1988. Figure 2)

Then the project is taken to the second loop, which begins with determining new objectives for the first prototype according to the results of the previous loop. If the results satisfied the objectives of the first loop, the objectives and constraints should be formed for the first prototype (usually a simple program with main functions). Then, again, form a risk analysis for the outcome, create the prototype and form tests to see if it satisfies the goals of the second cycle.

Each next cycle marks the next step for the development and bases on the previous' outcome until the program is complete and ready for the release.

2.3 MoSCoW method

The MoSCoW method is a prioritization technique (Clegg & Barker, 1994), which was used to set standards for the prototype. The MoSCoW acronym's capitalized letters stand for prioritization categories: Must have, Should have, Could have and Won't have.

2.4 Tools

Throughout the research the following tools were used to keep track of the progress, conduct interviews, create the software and gather data for it, and to solve occurring problems.

2.4.1 Diary

To keep track of all the processes, hardships, successes and results, a diary was used to record all progress in each phase of the research, which became a useful tool to quickly find references in case of a need to provide gathered data during meetings.

2.4.2 Researchers

Because of the narrow timescale for the whole project a research team was formed (Siiri Häidma and Elise-Marit Kippar) to find out the interests of the stakeholders. Since the target group was considered to be pupils who start learning geography, and high school students, who pick National Defense as the elective subject, the initial distribution strategy was to sell the game for schools so that teachers can use it during their lessons. But since the teachers have the authority of picking the tools to give lectures with, they represented the group of stakeholders. The responsibility of the research team was to conduct interviews with the teachers on those subjects and find out if the "Topographics" game concept was suitable for them to use in lectures.

The results of the research are not included to this paper, because it was part of separate master thesis'.

2.4.3 Game building software

There was a choice between gaming platforms Unity and Unreal Engine to build the prototype with, and Unity was chosen for personal reasons: the developer (the author of this paper) wanted to have full control over the functions, because there would be a need to write custom code in order to get the necessary results. Although both platforms provide custom scripting, Unreal Engine with C++ and Unity with C# and JavaScript languages, the author chose Unity, because he has an intermediate skill in JavaScript. But the prototype was chosen to be written in C#, because the assets provided by the program are already written in this language. Also in case of problems, the author can switch to the more familiar JavaScript in order to finish the prototype.

2.4.4 Usability scenario

To explain how the software could be used in schools or individually to learn topographical information, a usability scenario was formed (see Annex 1). A usability scenario tells a concrete story of user interaction and conveys a vivid image of what the system will do. It stimulates imagination and encourages "what-if" reasoning about alternatives (Rosson & Carroll, 2002).

2.4.5 Meetings

In theory, there existed digital map data which was used to make topographical maps of the regions of Estonia. Thus an appointment was set with the representatives of the company "Regio", the producers of maps and collectors, editors and visualizers of aerial data in Estonia (Regio, 2018). The reason was to get authentic and widely used map data, which can be used for reverse engineering – a process of taking apart an item to reveal the designs, architecture or to extract knowledge from it (Eilam, 2005) – and possibly use the information to advance development.

2.4.6 Problem solving tool: the Shewhart or PDSA cycle

The PDSA concept is a problem solving method, and the acronym stands for Plan, Do, Study and Act. The idea is that as the first step is needed to plan a change or a test aimed at the improvement, then carry out the change or test (Do), proceed to study the results to see if there is something to learn from or to examine what went wrong. Finally act on the results: adopt the change, abandon it, or run through the cycle again (Deming, 2000).

3. Results

From defining the concept of the game to developing the first prototype, the Spiral Method (Bohem, 1988) was used, forming the following 5 important cycles.

3.1 The first cycle: forming the concept and first sketches of the prototype

The primary goal for the first cycle was to form a unique concept for a digital game for learning symbols of topographical maps, its scaling and how to orientate using a map. The constraints were lack of information about the existing interactive learning materials and need to define the concept as soon as possible. A quick research did prove useful – the orienteering simulators “VirtualO” and “Suunnistussimulaatori” were found during this phase, but no interactive material on drawing a map. In most cases, the basics of topographical maps are taught by audiovisual explanations and animations.

It was decided that the learning game “Topographics” teaches topographical map symbols, scaling and orienteering when the user creates the map by drawing lines and drag-and-dropping symbols on it, while seeing the 3D world being created simultaneously according to the drawn map – all the functions would be put together into one session. Thus the gameplay has two game modes. The first is the drawing mode (see Figure 5), where the player draws lines and drags and drops symbols on an empty map canvas. This mode has a preview window of the terrain, which changes according to drawn map.

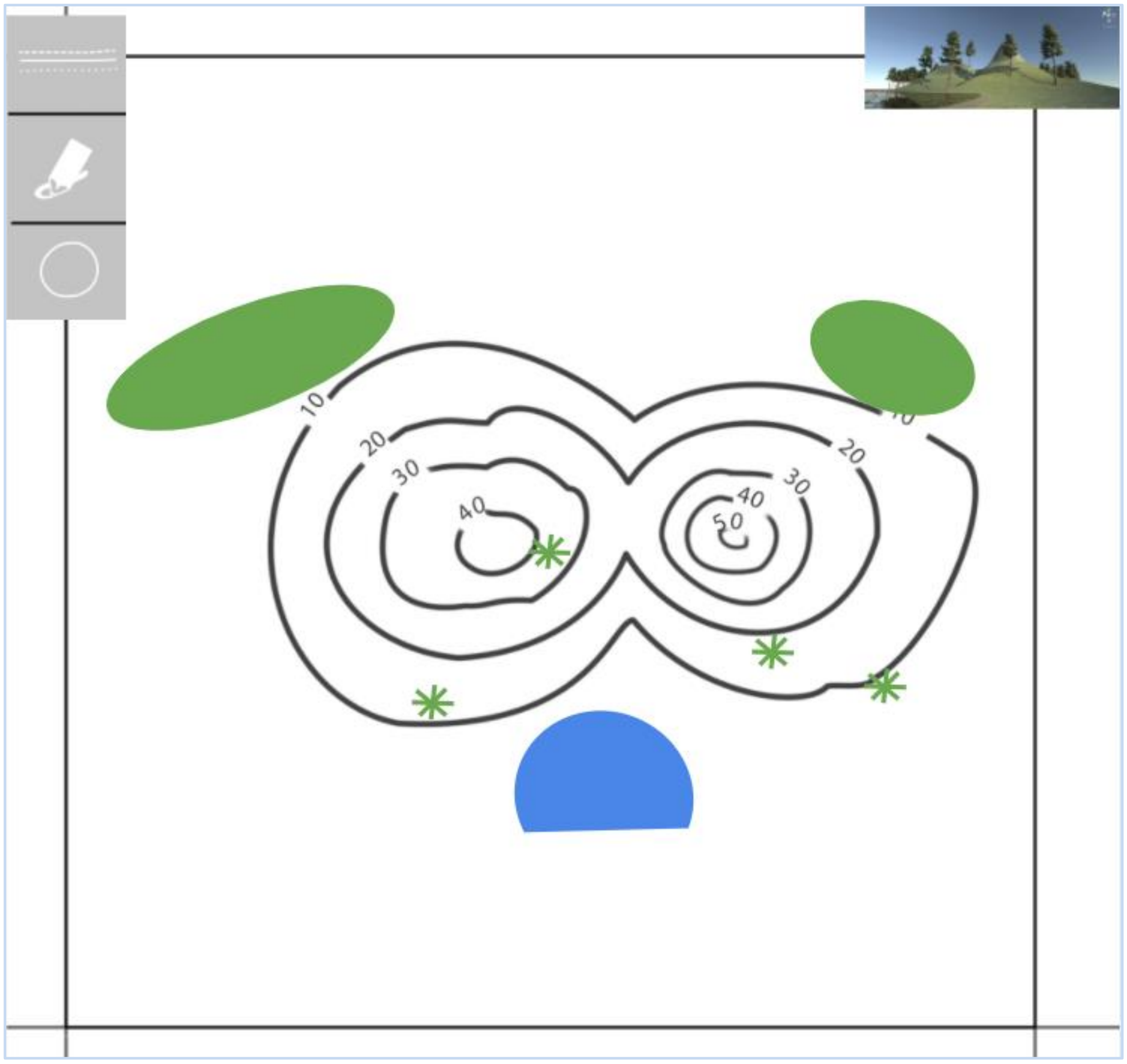


Figure 5. Drawing mode visualized for "Topographics".

The second mode is exploration (see Figure 6. *Exploration mode visualized for "Topographics".*) where the player can explore his new modifications to the terrain. These functions act as the core mechanics, which is "the essential play activity players perform again and again in a game" (Salen & Zimmerman, 2004). Thanks to this, the player can repeatedly add landmarks to the map, but to do so, the player needs to memorize the meanings of the symbols. And by doing that, while making associations knowingly, repeated memorizing records things into memory. (Lorayne, 2001)

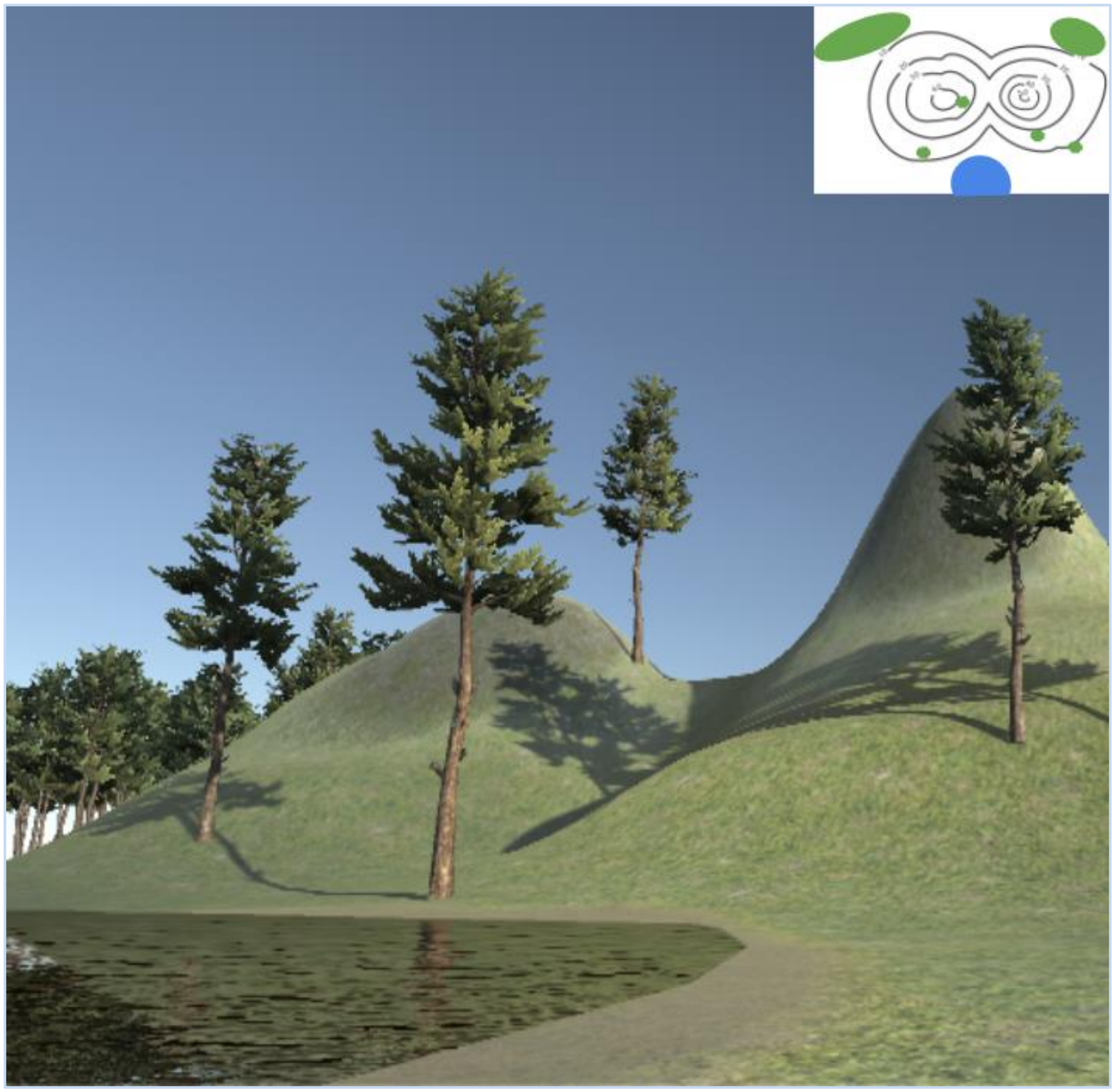


Figure 6. Exploration mode visualized for “Topographics”.

So the game is about drawing the contours and adding symbols to an empty canvas in order to create a map and a 3D world along it. The player’s role is to become the creator of the world by creating the map in the drawing mode, and using the exploration mode to see if the changes to the 3D world are applied as he planned. The gameplay mode is first-person in the exploration mode, and top view for the map drawing mode. The genre would be exploration and creation, and it would also be a learning or serious game. The target audience is pupils who study topographical maps in Geography (6th or 7th grade) or in the elective course National Defense (10th to 12th grade).

The game should run on smartphones and tablets so that the target group can play it anywhere and anytime using their own smartphones for it. Also it should support PC-s and Macs so that the game would reach the target group even in schools which do not have smart-devices, but have a computer lab. The distribution could be possible by schools, or application stores for android and iOS devices.

The gameplay takes place in single player mode in both, drawing and exploration modes. Although there is no competition mode planned for the game, the functionality to share the created world to social media, might create a motivation for the players to create more individual worlds.

To better describe the functionalities of the game, a usability scenario was developed. It also provides the information what the user can do using the game and how it could be used in lectures. At first it was in the format of presentation, which was later written down in detail (see Annex 1).

Although the concept was formed, there was still a risk that there exists an interactive topographical learning material which the research team could not find. This acted as a risk during some of the following cycles.

3.2 The second cycle: interest of the stakeholders

The goal for the next cycle was to find out the interest of the stakeholders and use the information to accordingly adjust the concept of the game. As the deadline of starting the development phase approached, the timescale became a constraint. Therefore, to narrow the risks of losing time, a research team was formed to conduct the interviews with the stakeholders and find out if the game concept was suitable for them. At the same time, the author took the time to reduce the risk that remained from the first cycle – find more information about any interactive learning material which used map drawing to teach topographical markings.

The research team found out that the teachers of Geography probably would not take time to use this game in their classes, because topographical information is too small part of the material needed to teach during the year. They suggested that if the game additionally contained regional, cultural and other types of maps with some tasks to solve, they would use it more willingly (Häidma, 2018; Kippar, 2018).

However the few National Defense teachers stated that they might use the game as the concept described it, but would prefer more an orienteering simulation which has close to real-life aesthetics and maps of real places to prepare pupils for a practical orienteering task (Häidma, 2018; Kippar, 2018). Thus the future versions of the game could have a library of premade maps that match the real-life places.

The author's findings proved useful as well: the free software "MOBO" for orienteering with a mobile application was found, and also the commercial software "OCAD", which focuses on drawing different kinds of maps, topographical and orienteering maps included. But none of them focuses to teach the symbols and simulate the existing or drawn maps in 3D.

According to the gathered information the initial distribution strategy became unstable because of the stakeholders' wish for a more complex game and the lack of need for the current concept. There was an urgency to decide if the current concept was worth of production with a different distribution strategy, or to switch to producing a more sophisticated game.

3.3 The third cycle: data collection for development

The major constraint was the deadline for the development phase being passed, but there was no decision on how to continue with the game. Due to this, the primary goal was to resolve the direction for the development. The secondary objective was to gather existing data to maximize the speed of the development phase.

The path decision for the development phase was affected by the risks: if deciding to switch to developing the more complicated game, there would be no time to build a prototype that had all the core features. So the second choice was accepted: to change the distribution strategy and start developing the prototype based on the original concept.

Another affecting constraint was the uncertainty of the author's programming skills, which raised the risk of not being able to deliver the prototype on time. So there was a need to prioritize, find existing data and code examples that can be used to help or speed up the development, and create an alternative plan for the scenario when the programming proves to be too difficult.

The decisions were strongly dependent of the data that was available – if there existed digital areal data that the game development software Unity could recognize, the development of the prototype for "Topographics" could start by visualizing the data as a topographical map and then proceed to creating the functionalities to alter the data according to the core mechanisms. But if Unity was not capable of reading the data, an alternative plan "Map Converter" would be considered to be conducted. As the title suggests, the idea was to first create a script that could read the areal data, and accordingly creates a visualized map and 3D environment in Unity, then proceed to adding the "Topographics" core mechanisms. Overall the "Map Converter" itself would prove to be an interest for companies which produce maps.

To get the existing areal data, an appointment was set up with "Regio", a company in Estonia that produces maps and visualizes, edits and gathers areal information (Regio, 2018). As a preparation, a usability scenario for the "Topographics" game was formed (see Annex 1) in order to explain how the software could be used in schools or individually to learn topographical maps. During the presentation, both "Map Converter" and "Topographics" concepts were introduced and the feedback was positive for both of those – the company was interested in testing the program and the game if either of the concepts were to be developed into the test-worthy prototypes. Also they directed the author to the website of National Land Board (Maa-amet) (the government agency which is responsible for realization of country land policy, maintaining the land cadaster, geoinformatics and management of geological and topographical data (Maa-amet, 2018)), where the samples of authentic topographical and areal data could be downloaded. Also they provided contacts of developers who might have similar experience and could help the project.

With the sample data from National Land Board and the contacts it was time to proceed to the next step: test the data for Unity and start programming the prototype.

3.4 The forth cycle: Data testing and start of the development.

The goal for this cycle was to test the sample areal data in Unity and begin the development. At first the sample data was analyzed and found out that Unity does not recognize the file formats and failed to read the data. Although it is possible to get the vector data in Unity, it first needs to be converted to a file format that Unity accepts (Lange, 2014). Because of this, it was decided to drop using the existing data because the "Map Converter" concept needs the functionality of being able to read the original data without any interference from other programs. Also when importing the converted file into Unity, it is used as a mesh (3D object that is formed by a wireframe (lines that connect the dots that define the shape of the 3D object) and planes (polygons), that fill the area between the wireframe lines forming the surface of the object), which can only be reshaped by scaling or stretching the whole model. But the "Topographics" concept dictates the need to be able to manipulate small portions of the whole terrain, rendering the sample data unusable at this point.

Without the available data, the "Map Converter" concept was set aside and proceeded to start developing the prototype for the "Topographics" concept.

3.5 The fifth cycle: production of the prototype

The goal was to create the prototype that has the functions defined by the MoSCoW prioritization method. This meant that the prototype:

Must:

- Drawing mode with:
    - a canvas – an "empty" map where symbols can be dragged-and-dropped, and lines drawn to form the contours, roads or rivers;
    - a drawing tool – to draw the contour lines on the canvas.
- Exploration mode with controls to look and move around.
- Basic GUI (graphical user interface) including:
    - a button to activate the line drawing tool;

    - a small screen to see the 3D world change;
    - a functionality to change between the drawing and exploration modes.

Should:

- Map grid in the drawing mode to have a basic idea about the scaling of the drawn objects.
- Authentic textures and colors in both, drawing and exploration modes.

Could:

- Tools for drawing forest, water and some landmarks with according buttons on the GUI.

Won't:

- Advanced aesthetics to provide more engaging experience in the exploration mode:
    - Elevation smoothing – the function that makes the raised terrain more realistic.
    - Sound effects such as footsteps for walking in the exploration mode; background music or ambient sounds.
    - Visually more attractive textures and meshes.
- Better GUI buttons – to provide better user experience and use the topographical symbols as icons so when the user repeatedly use a tool, he could connect the symbol with its meaning.

The development phase was divided to smaller tasks by functionalities:

- First accomplish drawing a basic static line
- Manipulate one point of the terrain
- Manipulate square shaped area
- … circle shaped area
- … custom shaped area
- Line made by mouse movement
- Create GUI button to activate the drawing tool
- Create exploration mode and movement functions
- Add function to GUI to switch between the modes
- Create functions to cancel drawing when in exploration mode
- Apply textures, colors to both modes

- Plan the next version of the prototype (functions to draw trees, water, drag-and-drop symbols, place self, fly-by mode when drawing)

Each smaller functionality was developed by using the PDSA problem solving cycle (Deming, 2000): first, if there was no knowledge on how to build the function, different tutorials and manuals were used to get the knowledge in order to plan the creation of the function. Then execute the plan, and study if the function works exactly how it was needed. If no, then new plans were created for the changes of the function, executed them and checked until the function worked as needed. Acting upon the new standard in this case means that whenever a similar functionality is needed, the code, or part of the code can be used in order to get the needed result.

3.5.1 Drawing a line

To begin with, a tutorial was found on how to make a line using Unity's built-in function "*trace renderer*" (Holistic3d, 2016). This technique was a good introduction to learn how a line is created in a 3D environment. It also provided the function that registers mouse *clicks* or a *touch* of a *touch-screen*, and forms a line according to the movement of the finger or mouse cursor. But the trace renderer's line's starting point is always *rendered* differently – usually thinner than the end point of the line. This meant that later on there was a need for a function which creates a line with unchanged starting and ending points.

Another tutorial was about creating a line on the *terrain* using a *raycasting* method (PushyPixels, 2012). This learning material introduced how to use raycasting to find points on the terrain and create points of a line above it. Thanks to this, it was possible to assume that if a line can be created above the surface of the terrain, the terrain could be manipulated under the line as well.

Both tutorials proved to be very useful for their theoretical information. But still there was no clue how to connect a line and the points of the terrain. Therefor as the next step there was a need to study creating a terrain and how to apply changes to its height.

3.5.2 Manipulating points of the terrain

To change the points of the terrain, a video was found that explains the use of Unity's *class* "terrainData" (Unity Technologies, 2018) and its "setHeight" *function* (UnityChat, 2014). This video introduced how the terrain mesh works and what can be done to it. Its purpose was to show how to manipulate the height value of one or more points in the terrain mesh. Using this knowledge, it was possible to raise not only one point, but a rectangle shaped area of the terrain (see Figure 7).

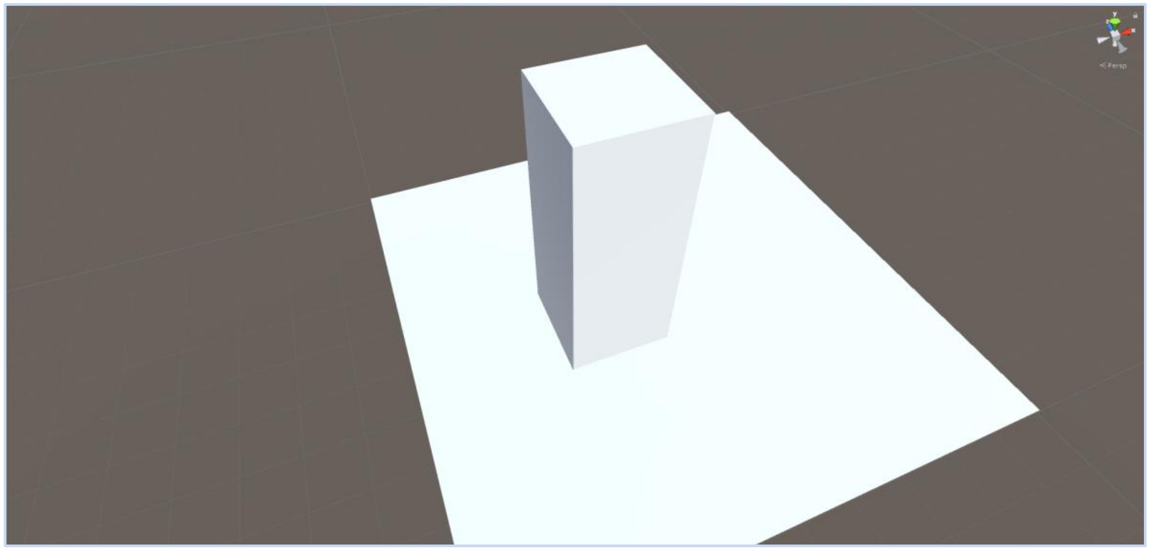


Figure 7. Rectangle shaped area raised in a terrain.

Because the "Topographics" game dictates the possibility of drawing amoeba-like shapes, the next step towards it was to create a circle-shaped elevation to the terrain. This was accomplished by adding a formula of the area of a circle to the function which raises the points inside the rectangle. The result can be seen in Figure 8.

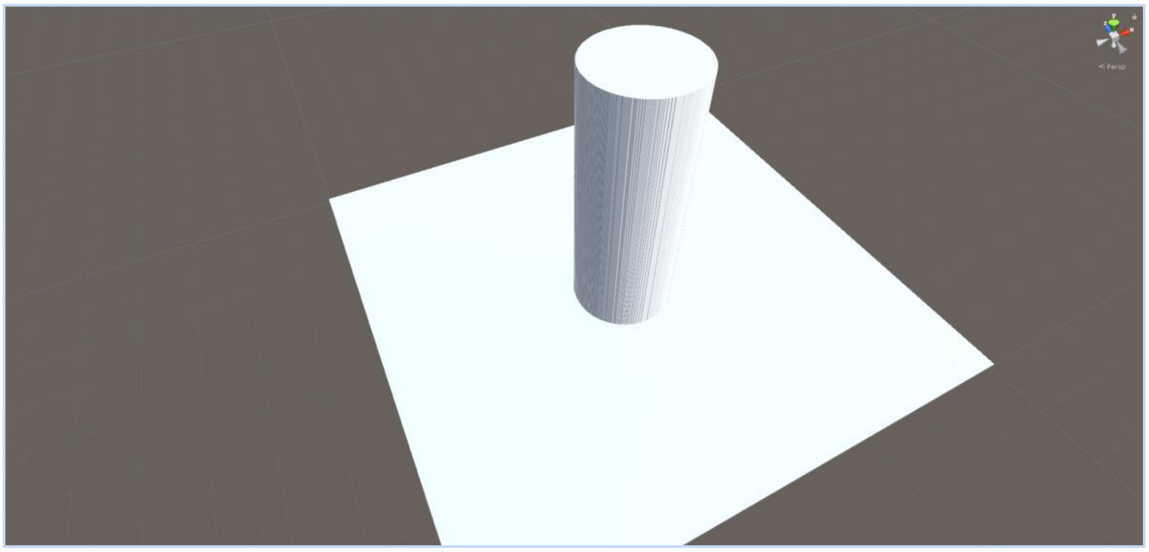


Figure 8. Circle shaped area raised in the terrain

Now it was needed to find a way how to find the *coordinates* of the mouse cursor and find the points in the terrain that match the cursor's coordinates. If the points were found, the values of height could be assigned to them. This could be done similarly to the "Unity Mobile Dev from Scratch: Drawing on the Screen" tutorial – create a *plane* in front of the *camera* and add a raycasting vector to the mouse cursor. According to the mouse movement, the collision between the raycasting vector and the plane was registered (Holistic3d, 2016). The data was used to match and raise the according and surrounding points of the terrain. The result can be seen on Figure 9.

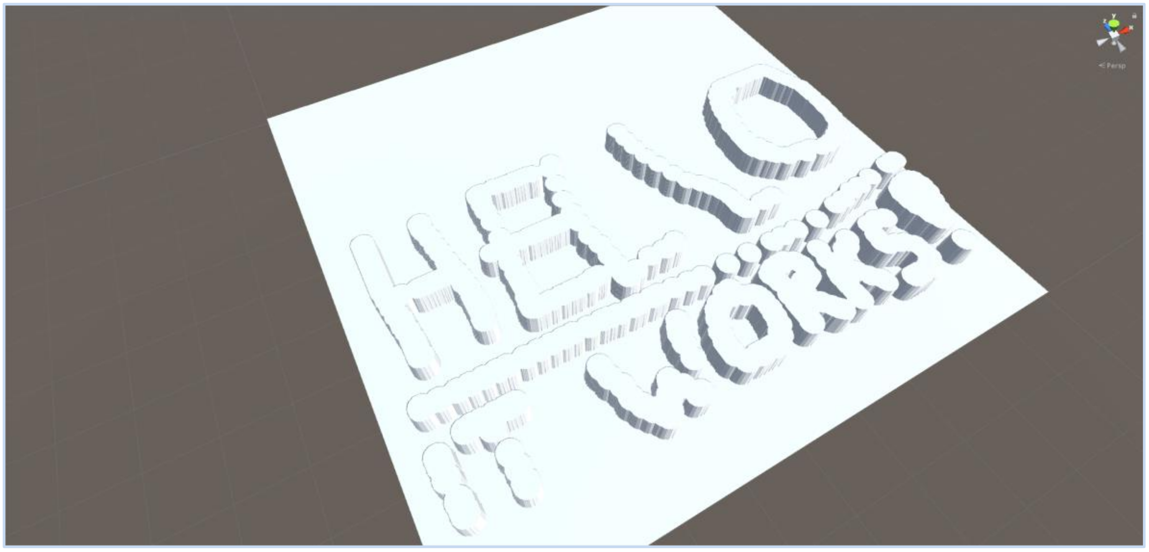


Figure 9. Terrain manipulated by tracing mouse movement.

3.5.3 Manipulate are inside given outline

The next step was to raise the terrain according to a line. The more complex line was created by a free asset "Easy Curved Line Renderer" (Haagndaaz, 2013). This script uses a unity's *Line Renderer* function, which contains an *array* of points and their coordinates, and renders a line between them. Each point's coordinates were used to raise the points in the terrain. The result is displayed in Figure 10.

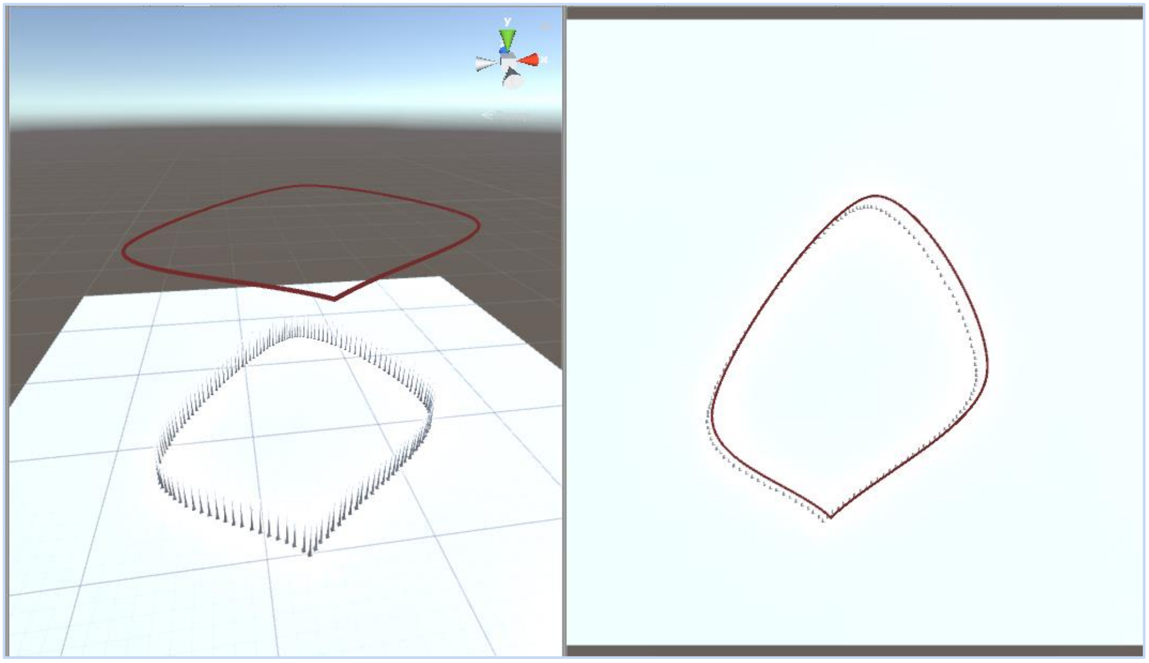


Figure 10. Terrain’s points are raised according to the line.

Since there does not exist a formula to find the area of a complex shape, there was a need to find a method for it. The first findings offered tutorials on how to find the size of the areas of the shape, but nothing on how to determine if a point is inside the outline of the shape.

It was decided to ask help from a math student, Gabriel Kaine, who also has experience in programming. Together a solution was found: first create a rectangle around the shape so that the shape’s most outward points are located inside the rectangle’s sides. Next, start a raycasting method to check row by row and point by point if the point is located inside the outline of the shape or not. If it is inside, then assign the height value and check the next point.

The only problem with it was the quality of controller mechanism – the solution was to create a *variable* that changes its value when the next point’s coordinates match to a point of the shape’s outline. The matched point and each following point will be given the height value until the next point’s coordinates match with the outline’s point (see Figure 11).

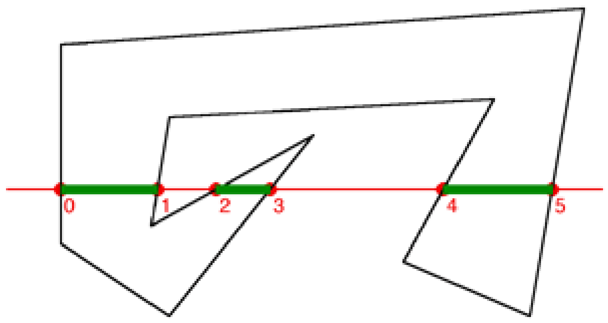


Figure 11. The points between 0-1, 2-3 and 4-5 have to be affected (Finley, 2007. Figure 3)

This controller was not suitable, because the points in the outline had gaps between them. In this case the raycast might slip through the gap without triggering the variable's value change. It could have been solved by increasing the radius for the raycast so it could never slip by the points, but this would have a strong effect on the raised terrain's shape.

Later, an article was found describing the same method: "Efficient Polygon Fill Algorithm With C Code Sample" (Finley, 2007). It even provided code samples in C language, but it seemed too complicated to comprehend what was needed to do to adapt the code to C# language.

It was time to create a post in the "Stack Overflow" forum - the largest, most trusted online community for developers to learn, share their knowledge, and build their careers (Stack Overflow, 2018). The purpose was to explain the problem and get ideas from the community how to solve the problem. The post can be found on address https://stackoverflow.com/questions/48935840/how-to-manipulate-a-shaped-area-of-terrain-in-runtime-unity-3d.

This post was shared with the programmers the author knew, and one of them, Siim Somma, had had a similar problem before. He directed the author to another article in Stack Overflow: "How can I determine whether a 2D Point is within a Polygon?" (Mecki, n.d.). He also provided a code sample that contained the solution for the problem. The principle remained the same – create a rectangle which surrounds the given shape, but this time the sides must not touch the shape. The raycasting method should still be used and the controller mechanism instead of checking a match in the outline points, it checks if the raycasting has passed the side of the shape. The solution is good enough, but might provide errors if the shape and raycast direction are collinear. Still it is enough for the prototype and the result is satisfying (see Figure 12).

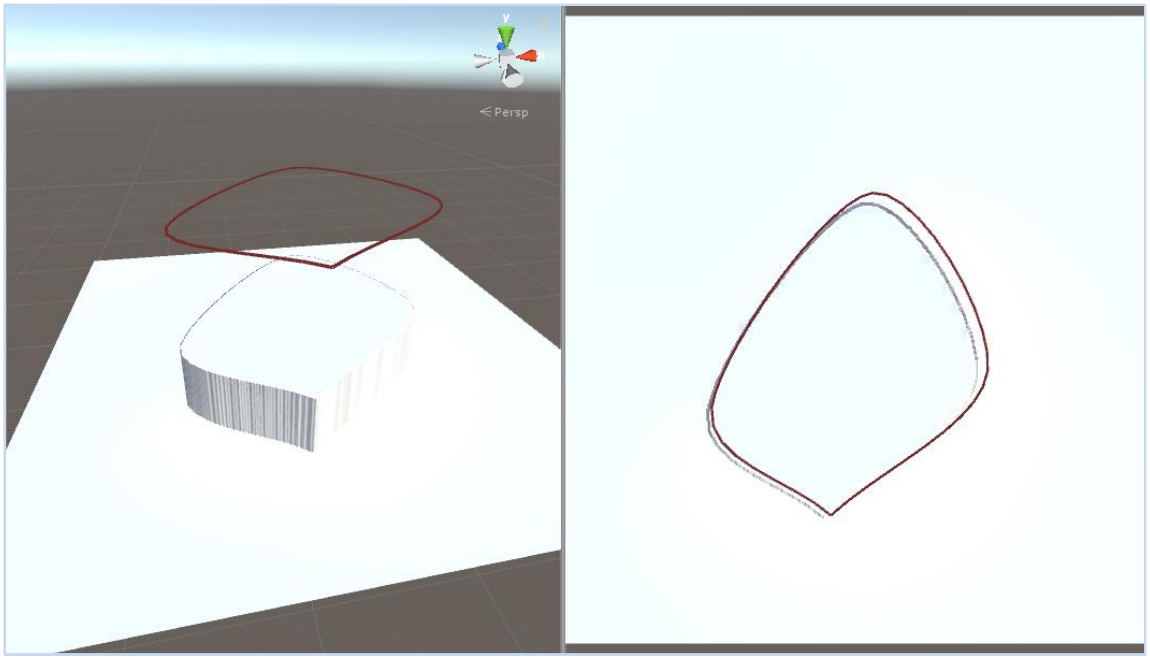


Figure 12. The terrain is raised according to the area of the shape

3.5.4 Lines made by mouse movement

Now it was needed to create the line according to the mouse movement. At first the author wanted to examine if something could be changed in the "Curved Line Renderer" (Haagndaaz, 2013), but that turned out to be too complicated, so it was decided to find more tutorials on how to draw lines in Unity. The most useful turned out to be the tutorial on "How to make a LINE RENDERER Replica in Unity" (Brackeys, 2017). This taught how to make lines according to the mouse movements. And the points in those lines were passed to the function that raises the terrain accordingly. The result is seen in Figure 13.

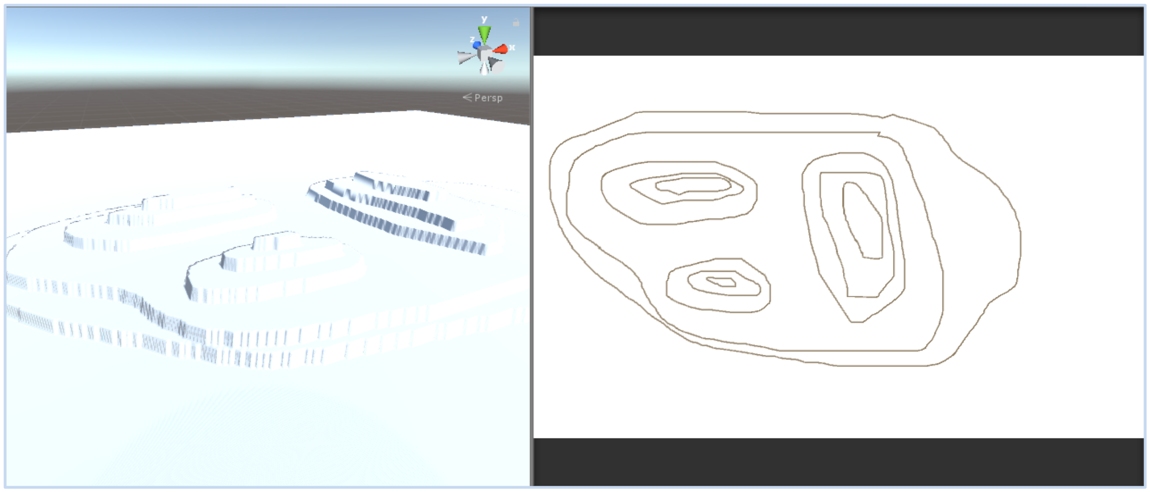


Figure 13. Lines are created by registering mouse movement. According to the shapes the terrain is raised.

3.5.5 Aesthetics, simple GUI and exploration mode

The hardest part was over and it was time to add the second core mechanism – the exploration mode. For this, Unity’s library of assets was used, including player controls for movement and looking around.

To be able to switch between tools and to get the know-how to make one, a simple GUI was created with a button to activate the line drawing function, and a small window to have a quick view what happens in the 3D mode. Also a function was added to switch between the drawing and exploration mode when clicking on the 3D preview screen. This revealed a problem, that if the drawing tool was active, it was possible to accidentally draw new contours while being in exploration mode. To fix this, a new function was added to check which mode is currently active.

Another problem was that when drawing the lines, it was difficult to understand the scale of the created terrain. So a map grid with 200 x 200 meters was created and added to the drawing mode. This helps a little, but in later versions there should be a bar and ratio scale.

Finally, to enhance the aesthetics, a grass texture was applied to the terrain and the “open area” color was chosen as the default setup for the canvas, and the first version of the prototype is complete (see Figure 14 for drawing mode and Figure 15 for exploration mode).

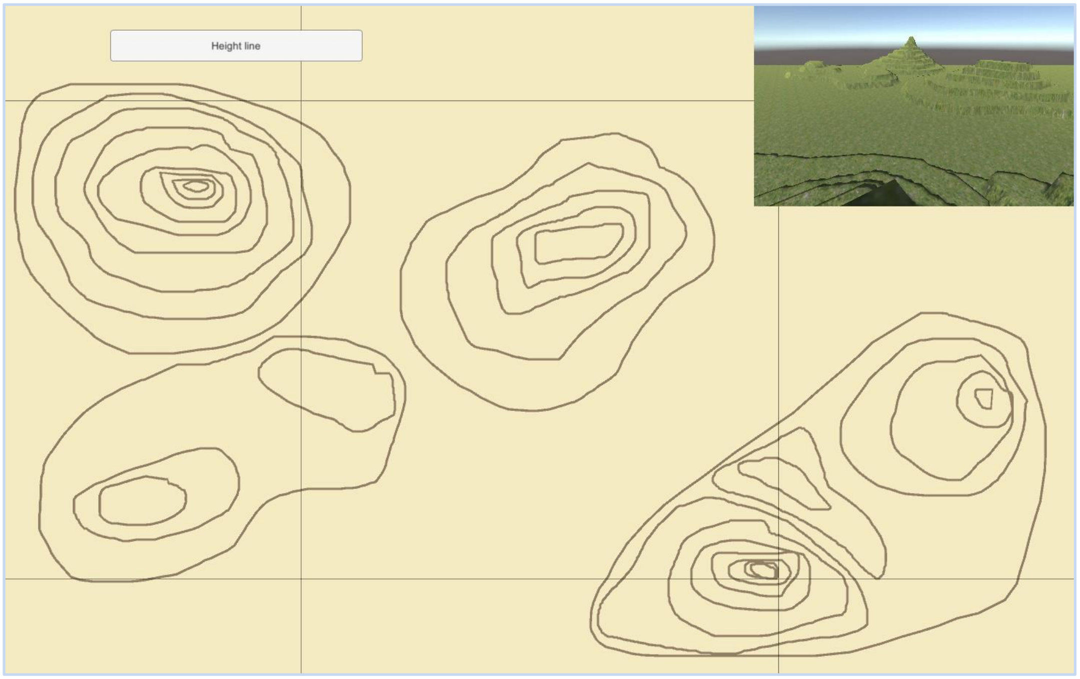


Figure 14. The drawing mode - first version of the prototype

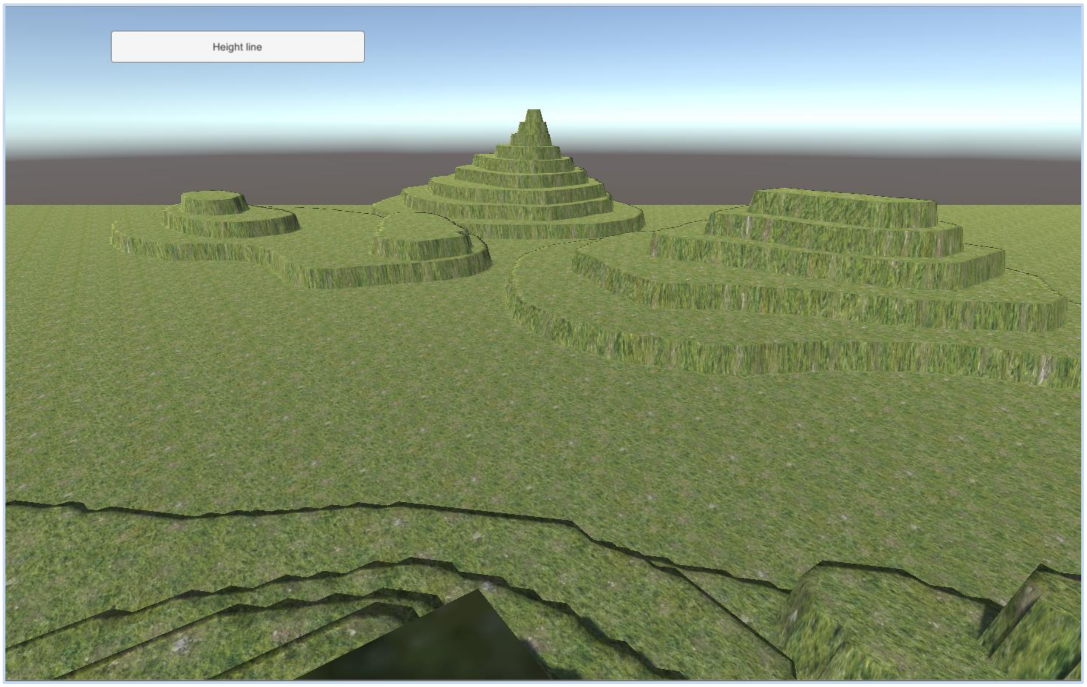


Figure 15. Exploration mode – first version of the prototype

4. Discussion

The results presented the importance of writing down ideas; what were the options when the stakeholders cannot be satisfied; what further research could be conducted using the prototype; what were the outcomes of *alpha testing*; what functionalities the future versions could have; and what was the result of gathering the terms and definitions together.

4.1 Usability scenario

When the game concept was formed, the usability scenario was in the presentation format. It was enough for the author to pass the idea of how the game works, what are the functions and how it could be used by teachers in lessons, but after the research team had conducted their interviews with the teachers of Geography and National Defense, they stated that a written document containing the detailed scenario would have been useful. Furthermore it might even had had an effect of their research results, because when they were asked detailed questions about the game, there were moments where they could not provide answers, because they did not have any physical material to revise the game concept upon. Thus as a suggestion, the written usability scenario should be created right after finishing the formulation of the game concept, so that partners and other people of importance could study better what the concept is about.

4.2 Distribution strategy

The initial distribution strategy expected the teachers, who embodied the stakeholders, to use the game with the core mechanics the concept dictated. It would have been flexible if the stakeholders wanted to have some additional functions while accepting the core mechanics, but as Häidma and Kippar found out, most of the teachers found that the game only supports a small part of the year’s curriculum, which would render the usage of the game an inconvenience. This was a setback for the distribution strategy, because if the teachers would not use it, it would not reach the target group – the pupils who learn topography in Geography and National Defense lessons. Due to this, an alternative distribution strategy was needed.

Even though the stakeholders might not be interested in the game, target group still might be. This way the core mechanics could remain the same, but should definitely be tested on the target group to find out if the game suits them, what they liked or hated about it, and if they would be more motivated to play the game more when they could share their created worlds online. According to the gathered data the game could be modified to be more engaging and motivating, after which the game could be distributed to the Google Play store and App Store for Apple devices. When offering the product to the app stores, the game could be distributed for free, but sell advertisements with it.

4.3 Further possible researches on the prototype

Another research could and should be conducted using the next versions of the prototype to find out if the target group learned the meanings of the symbols and understood the concept of the contours after been playing the game. The results could be used to verify the effectiveness of the teaching methods in the game, and also to add modifications to the game concept in order to improve further the efficiency of teaching the topographical markings with the learning game “Topographics”.

With the future versions of the game, when it had the functionality to print out the map, the functionality could be tested on the 25.3% of the gymnastics teachers who claimed in Kallas’ (2015) research that they need schoolings on how to draw the school orienteering map, to find out if satisfied their needs to create orienteering maps. If the result turned out positive, it might raise the interest of teachers from other subjects as well, such as Geography and National Defense, and might make the first distribution strategy possible.

4.4 Alpha testing

After the core mechanisms were developed and a GUI added, it was needed to conduct an *Alpha testing*, which surfaced the following problems:

5. Hard to understand how the terrain changes because the preview is shown from eye-level perspective;
6. Switching to the exploration mode from the drawing mode, it was possible to accidentally make drawings to the canvas;
7. Hard to understand the scale of the drawn lines;

8. The drawing mode tools are visible in exploration mode;
9. The surface is raised random amount.

Some of those problems were fixed, such as setting the amount of how much the terrain raises, but some proved to be more time consuming and was decided to be taken into plan with the next versions.

4.5 Plan for upcoming versions

The next version should solve the problems emerged in the Alpha testing, such as a fly-by function in the drawing mode for a quick recap of the from a bird's perspective to reach the places where it would take more time to go in the exploration mode. But it also should have new functionalities such as tools to fill map areas with colors and add according objects to the terrain, i.e. green color for forest, blue color for water etc. Basically the needed code already exists – the coloring function could be attached to the mouse tracking script, and add plantation to the affected areas of the terrain.

A new function would be added – dragging and dropping symbols to the map in order to create landmarks on the terrain. This function could be used to place the starting point of exploration as well.

The later versions should have a fully developed GUI, the tutorial and instructions how the game works. Also it should be possible to save and print a created map, or start a new project. And after that, the game should have tasks in the exploration mode as well, such as planning best routes from reaching one point to another, pinpointing his own location on the map, and orientating through the terrain using a compass the drawn map.

4.6 Applying the PDSA cycle produces the functionality

The development phase started with bigger objectives: the need to create the core mechanics. At first there was no understanding of how the goal could be reached, so it was disassembled to smaller objectives, or functions: the drawing function, function to manipulate one point of the terrain, then multiple points, points in a specific shape of an area etc. The PDSA cycle (Deming, 2000) became very useful as it was repeatedly applied to create each function.

4.7 Glossary

As one of the results of this paper, a glossary was formed, containing definitions for topographical, 3D modeling and other terms which describe functions or methods used by the game development program Unity.

Conclusion

The purpose was to find out how to build a digital learning game so that it teaches the given topic effectively. To begin with, a unique concept of the game had to be formed which could be done by searching for similar products and finding out the interests of the stakeholders.

When searching for statements and articles, that describe the current situation about topographical knowledge among pupils in middle and high school of Estonia, nothing relevant was found when using keynotes "topography", "orienteering" and "maps" for the search.

There exist other games, interactive learning materials and softwares which either teach the basics and leaves out the techniques of using a compass and map to orientate, or concentrate only on orienteering while expecting the user to know the basics. Thus a game concept was formed to teach the basics of topographical maps and how to orientate using the map. The game concept was adjusted according to the feedback of the stakeholders, by adding functions to the later versions of the game, such as a printing option for the maps, and adding a library of maps with places that exist in real-life.

When proceeding to production phase, it was needed to know if there exists any authentic digital data that can be used to speed the process of developing the game and use the data in developing process. The sample data was found, but the game developing program Unity was not capable of directly importing it. The data had to be converted first by 3D software, but even if the data could be imported then, the data was converted to a mesh which cannot be manipulated enough to match the concept of the prototype.

During the development phase, when finding solutions to solve bigger problems, the problem was disassembled to smaller tasks and the chosen problemsolving tool was repeatedly and successfully applied until the prototype had the required functionalities.

An alpha testing was conducted after creating the core functionalities in order to find program errors and reveal the unseen problems.

After finishing the first prototype, new objectives were formed for the next versions of the prototype which should have more functions in order to teach better the topographical maps and satisfy the stakeholders’ needs, and also should solve the problems that surfaced during the alpha testing.

Finally a glossary was formed which contains the topographical terms and also specific expressions and definitions used in Unity and 3D modeling.

References

AG, O. (2018). OCAD. OCAD AG. Retrieved from https://www.ocad.com/en/

Argles, T., Minocha, S., & Burden, D. (2015). Virtual field teaching has evolved: benefits of a 3D gaming environment. https://doi.org/https://doi.org/10.1111/gto.12116

Boehm, B. (2007). A Spiral model of software development and enhancement. In *Software Management, Seventh Edition* (pp. 37–48). John Wiley and Sons Inc. https://doi.org/10.1109/9780470049167.ch2

Bohem, B. W. (1988). A spiral model of software development and enhancement. *Computer*, *51*(5), 61–72. https://doi.org/10.1109/9780470049167.ch2

Brackeys. (2017). How to make a LINE RIDER Replica in Unity (Livestream Tutorial). *YouTube*. Retrieved from https://www.youtube.com/watch?v=dmBQQ2XtuhU

Brown, A. L. (1992). Design experiments: Theoretical and methodological challenges in creating complex interventions in classroom settings. *The Journal of the Learning Sciences*, *2*(2), 141–178.

Catching Features. (2008). Retrieved from http://www.catchingfeatures.com

Clegg, D., & Barker, R. (1994). *Case Method Fast-Track: A RAD Approach*. Addison-Wesley Longman Publishing Co., Inc.

Collective, T. D.-B. R. (2002). Design-Based Research: An Emerging Paradigm for Educational Inquiry. *Educational Researcher*, *32*(1), 5–8.

Collins, A. (1992). Toward a design science of education. *New Directions in Educational Technology*, 15–22.

Contours. (2016). Virtual-O. *Contours*. Retrieved from http://store.steampowered.com/app/529020/VirtualO/

Cross Motion. (2016). Cross Motion Tallinn Hackathon And Conference. Retrieved April 26, 2018, from http://www.crossmotion.org/tallinn-conference-and-hackathon/

Deming, W. E. (2000). *The new economics: for industry, government, education. Second edition*. MIT press.

Eilam, E. (2005). *Reversing: secrets of reverse engineering*. John Wiley & Sons.

Finley, D. R. (2007). Efficient Polygon Fill Algorithm With C Code Sample. Retrieved May 1, 2018, from http://alienryderflex.com/polygon_fill/

Haagndaaz. (2013). Easy Curved Line Renderer [Free Utility]. Retrieved May 1, 2018, from https://forum.unity.com/threads/easy-curved-line-renderer-free-utility.391219/

Häidma, S. (2018). *Applying digital game-based learning to topography studies. Manuscript in preparation.*

Holistic3d. (2016). Unity Mobile Dev from Scratch: Drawing on the Screen. *YouTube*. Retrieved from https://www.youtube.com/watch?v=cHVZ0SYIHkI

Kallas, K. (2015). *The Implementation of Orienteering in the National Curriculum of Elementary and Secondary Schools*. University of Tartu.

Kippar, E.-M. (2018). *Analysis of digital learning materials and the content development of a new learning game for geography. Manuscript in preparation.*

Klaar, T. (n.d.). MOBO. Tak-Soft. Retrieved from http://mobo.osport.ee/

Lange, A. (2014). PUTTING REAL MAP DATA IN UNITY GAME ENGINE – PHILLY GAME JAM. Retrieved April 29, 2018, from http://secondtruth.com/2014/11/putting-real-map-data-in-unity-game-engine-philly-game-jam/

Liben, L. S., & Downs, R. M. (1989). Understanding maps as symbols: The development of map concepts in children. *In Advances in Child Development and Behavior*, *22*, 145–201.

Lorayne, H. (2001). *How to Develop a Super Power Memory: Your Absolute, Quintessential, All You Wanted to Know Complete Guide*. Frederick Fell Publishers. Retrieved from http://onlineaccesscenter.com/bmm/library1/How to Develop a Super Power Memory.pdf

Maa-amet. (2018). Maa-amet. Retrieved April 29, 2018, from https://www.maaamet.ee/et/amet-kontakt/tutvustus-ja-struktuur

Mecki. (n.d.). Answer on Scott Evernden. How can I determine whether a 2D Point is within a Polygon? Retrieved May 1, 2018, from https://stackoverflow.com/a/218081/9358680

Pulli, A. (2014). Suunnistussimulaattori. Retrieved from http://www.suunnistussimulaattori.net/portal_en/

PushyPixels. (2012). How to draw lines on terrain. *YouTube*. Retrieved from https://www.youtube.com/watch?v=VrvyJZEMy2g

Regio. (2018). Regio. Retrieved April 28, 2018, from http://regio.ee/

Riley, S. (2008). Map maker. Teacherled.com. Retrieved from http://www.teacherled.com/resources/mapmaker/mapmakerload.html

Rosenblad, Y., Oska, K., Randma, T., Valk, A., Haridus- ja Teadusministeerium, Reps, M., … UE. (2017). Eesti elukestva õppe strateegia 2020. *European Commission*. Retrieved from https://www.hm.ee/sites/default/files/strateegia2020.pdf

Rosson, M. B., & Carroll, J. M. (2002). *Usability engineering: scenario-based development of human-computer interaction*. Morgan Kaufmann.

Salen, K., & Zimmerman, E. (2004). *Rules of play: Game design fundamentals*. MIT press.

Stack Overflow. (2018). Stack Overflow. Retrieved May 1, 2018, from https://stackoverflow.com/company

Unity Technologies. (2018). Unity documentation. Scripting API. Retrieved May 1, 2018, from https://docs.unity3d.com/ScriptReference/TerrainData.SetHeights.html

UnityChat. (2014). Unity - How to Change Terrain Height In-Game (C#). *YouTube*. Retrieved from https://www.youtube.com/watch?v=1j9McAMK4lE

WebRangers. (n.d.). Reading a Map. National Park Service. Retrieved from http://www.nps.gov/webrangers/activities/readingmap/readingmap.swf

Wiegand, P. (2006). *Learning and teaching with maps*. Routledge.

Terms and definitions

Alpha testing	–	a process of software testing to find and fix major mistakes, program errors or unseen problems.
Array	–	a group of data, which can contain numbers, string-type texts, objects etc.
Camera	–	an object in Unity which has a direction and position in the coordinate space. It is used to display objects, meshes, planes and other elements on the screen from the perspective of the camera's location.
Class	–	a constructor function used in object-oriented programming. Used for objects that will have similar functions and attributes but with different values. I.e. names, colors, abilities to make sounds or move.
Click	–	a user action with a mouse which triggers the usage of an element on the screen targeted by the cursor.
Contours	–	(or the contour lines) – lines on a topographical map which connect the points that have the same height.
Coordinate	–	an array of numerical values, the x-, y- and z-position, which describe the location of a point or an object in the coordinate space.
Cursor	–	(or mouse cursor) – usually an arrow-shaped icon to match mouse movements on the screen, and marks the location where mouse actions (such as clicking, double-clicking or drag-and-dropping) are triggered.
Drag-and-drop	–	A user action of moving an element on the screen to another location.

Function	–	an action added to an object in order to get a specific outcome when triggered.
GUI	–	graphical user interface
Line renderer	–	a built-in function in Unity which renders a line between two or more points which' coordinates are passed to it.
Mesh	–	3D object that is formed by a wireframe and polygons.
Mouse	–	an external input device for the computer which controls a cursor on the screen and sends actions to it.
Plane	–	a rectangle shaped surface. Usually used to display a texture, or set to a specific location to create collisions with moving objects and add triggers to the collision event.
Polygon	–	planes that fill the area between the wireframe lines forming the surface of the object.
Raycasting	–	a method of creating a vector with a direction to an object. Usually to measure distance or register collisions with other objects.
Rendering	–	a method for the computer to display a visualized image, 2D or 3D object or other elements on the screen.
Terrain	–	a mesh, which wireframe's points can be given a value of height to simulate a landscape.
Texture	–	an image set to a surface to look more detailed or complex.
Touch	–	a user action of making contact on a touch-screen device which registers usage of the element displayed on the screen where the touch occurred.

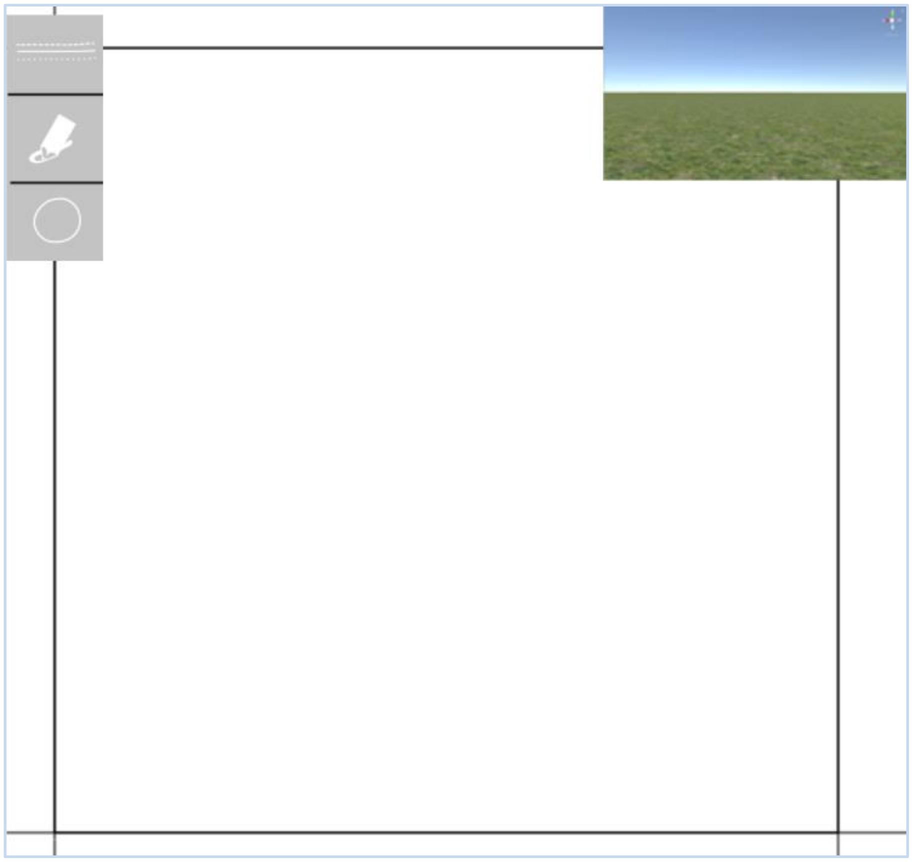
Touch-screen	–	a screen that can register the contact of a user’s finger and do actions with the elements displayed on that location.
Trace renderer	**–**	a built-in function in Unity which creates a line-shaped trace to an object.
Trigger	–	an event which can be used to start functions or other events.
Variable	–	a container for a certain type of value.
Wireframe	–	lines that connect the dots that define the shape of the 3D object.

Appendix

Annex 1

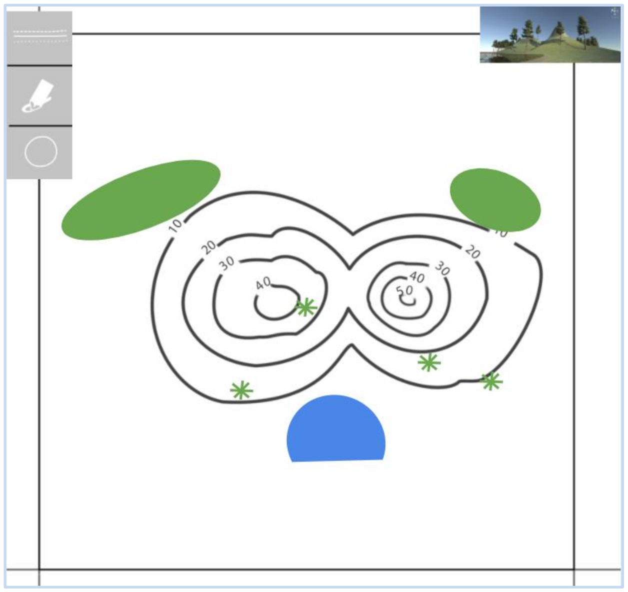
Nate has just started studying geography in 6th grade and the teacher has assigned the first homework for the pupils. The assignment is to download the digital learning game “Topographics” to their smartphones or tablets via Google Play; then start and complete the tutorial in the game, and post results to e-school, or send them via e-mail to the teacher. Unfortunately Nate does not own a smartphone, so the teacher suggests Nate to use the game in the computer class instead.

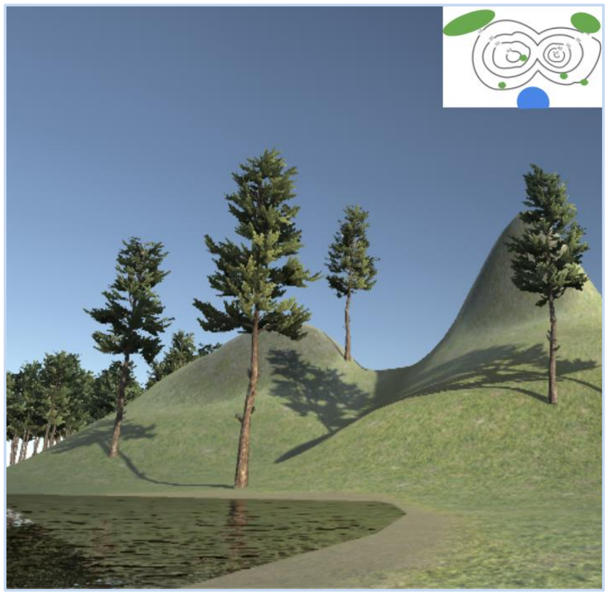
In the computer class, Nate logs in on a PC, and opens the learning game “Topographics” by double-clicking the game icon on the desktop. The game opens and shows options to “start drawing a map”, or to “start the tutorial”. Nate chooses the option for the tutorial, and the screen changes to “the drawing view”. This view contains a drawing canvas, a toolbar on the left of the screen and to the top-right corner of the screen there is a small window showing a flat landscape.



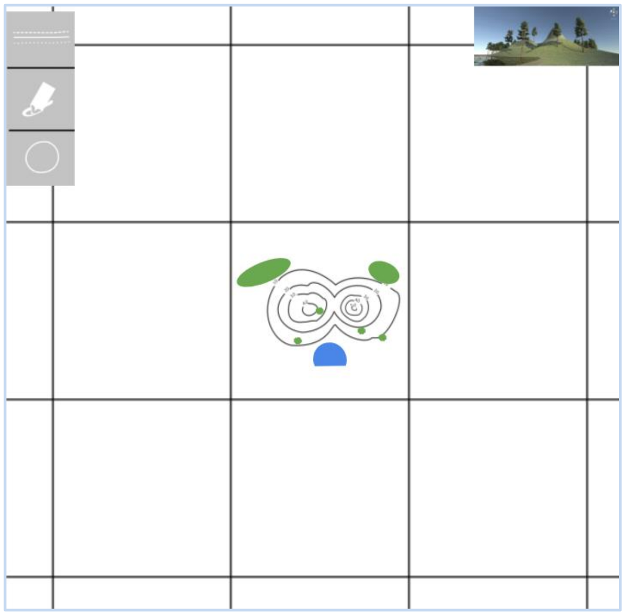
A text “Click here” appears next to the toolbar, with an arrow pointing to a “line” icon in the toolbar. Nate clicks on the icon, and the text changes to “Click & drag to draw a shape”, and an arrow pointing to the canvas. Nate clicks and drags on the canvas and sees a line appearing where the mouse cursor is moving. When he releases the mouse button, he notices the landscape change in the landscape window on the top-right corner: the landscape has raised in the exact shape as Nate drew the line. Next to the landscape window appears a new text “Congratulations! Now you know how to raise the terrain! Click on the terrain window to have a closer look!”

After clicking on the terrain window, the canvas disappears and the “landscape view” is shown in full screen. Nate is introduced how to look and move around in the landscape and also how to switch back to the drawing view. Then the tutorial continues to introduce other tools, such as colors and symbols. After another minute, Nate has created two hills, a lake and some trees which he is eager to start exploring in the landscape view.





Now the tutorial introduces Nate the final task: the canvas zooms out, revealing the grid of the map on the canvas, and ordering Nate to fill all the 9 squares of the grid with any of the tools found in the toolbar.



After finishing the final task, the tutorial shows Nate how to save the work and share it online or by e-mail. Nate sends the work to his teacher, but also chooses to share images of the newly created world on social media, so others can see his creation too.