



**UNIVERSITY
OF OULU**

FACULTY OF INFORMATION TECHNOLOGY AND ELECTRICAL ENGINEERING

Şan Güneş

**VIEWPOINT MANIPULATIONS FOR 3D
VISUALIZATIONS OF SMART BUILDINGS**

Master's Thesis
Degree Program in Computer Science and Engineering
May 2019

Güneş Ş. (2019) **Viewpoint Manipulations for 3D Visualizations of Smart Buildings**. University of Oulu, Degree Programme in Computer Science and Engineering, 47 p.

ABSTRACT

This thesis covers the design and implementation of a new single-input viewpoint manipulation technique aimed for a specific use case. The design of the technique is made based on previous literature. The objective of the research is to assess whether a single-input viewpoint manipulation technique can be as efficient as a multi-input viewpoint manipulation technique when used for observing a three dimensional (3D) model of a smart building.

After checking the existing literature about basics of viewpoint manipulation, it was decided to design a single-input viewpoint manipulation technique that can be used on a wide range of hardware including touch screen devices not capable of multi-touch input and personal computers with a regular mouse.

A 3D visualization of a nursing home was implemented to be viewed with the new technique. The nursing home in question is a smart house with sensors deployed in it, and sensor data is visualized in the 3D model. Aside from the new single-touch technique, a commonly used multi-touch technique was also implemented in order to compare the single-touch technique against it.

Participants were recruited and user tests were made to find issues with the system. The yielded results indicate some clear points in the new technique that can be improved for future research.

Keywords: interaction techniques, 3D navigation, 3D user interfaces

TIIVISTELMÄ

Tämä työ kuvaa suunnittelu- ja implementaatioprosessin kuvakulmien manipulointiin tarkoitettulle uudentlaiselle yhden sormen (single-touch) syöttöjärjestelmälle. Suunnittelu pohjautuu aiempaan tutkimukseen. Tutkimuksen tarkoituksena on arvioida, onko yhden sormen järjestelmä yhtä tehokas monen sormen (multi-touch) järjestelmään verrattuna, kun kohteena on kolmiulotteinen (3D) malli älykkäästä rakennuksesta.

Aiempiin tutkimuksiin nojaten yhden sormen järjestelmään päädyttiin, koska se tukisi suurempaa laitekantaa monen sormen järjestelmiin verrattuna. Työssä kehitettiin hoitokotia esittävä älyrakennuksen 3D-malli, jota käytettiin järjestelmän tarkastelemiseen. Kyseinen hoitokoti on anturointia sisältävä älykäs rakennus; 3D-mallia käytettiin anturidatan visualisoimiseen. Koejärjestelyissä käytettiin tavanomaista monen sormen järjestelmää vertailukohtana kehitettyyn järjestelmään. Vertailu tehtiin kohenkilöiden ja käyttäjättestien avulla. Tuloksista paljastui ominaisuuksia, joita tulisi parantaa järjestelmän tulevissa versioissa.

TABLE OF CONTENTS

ABSTRACT	
TIIVISTELMÄ	
TABLE OF CONTENTS	
FOREWORD	
LIST OF ABBREVIATIONS AND SYMBOLS	
1. INTRODUCTION	7
1.1. Research Motivation	8
1.2. Method	9
1.3. Structure of the Thesis	9
2. RELATED WORK	11
2.1. Smart Buildings	12
2.2. Viewpoint Manipulation Techniques	13
2.2.1. Rotation	14
2.2.2. Point of Interest Navigation	15
2.2.3. Orbiting	18
2.2.4. Planned Viewpoint Manipulations	19
3. SYSTEM DESIGN AND IMPLEMENTATION	21
3.1. System Architecture	21
3.2. Building Model	22
3.3. Temperature Visualizations	22
3.4. Single Touch Viewpoint Manipulation	23
3.4.1. Switching Between Floors	23
3.4.2. Switching Between Rooms	24
3.4.3. Orbiting	26
3.5. Multi-Touch Viewpoint Manipulation	26
4. EVALUATION SETUP	30
5. RESULTS	35
5.1. Quantitative Results	35
5.1.1. View Matching Task	35
5.1.2. Post Questionnaire	35
5.2. Qualitative Results	38
5.2.1. User Comments	38
5.2.2. Other Observations	39
6. DISCUSSION	40
6.1. Heuristic Evaluation	40
6.2. Limitations and Future Work	41
7. CONCLUSION	43
8. REFERENCES	44

FOREWORD

This thesis was made in the Center for Ubiquitous Computing at the University of Oulu. It was funded as part of the VIRPA-A project by the former Finnish Funding Agency for Technology and Innovation (Tekes). I would like to thank my supervisor Dr. Matti Pouke for his continuous support. I would also like to express my gratitude to my friends Mahmoud and Mounib for creating an amazing supportive environment during our studies. Most importantly, I would also like to thank my family and all of my friends who give meaning to my life.

Oulu, 13th May, 2019

Şan Güneş

LIST OF ABBREVIATIONS AND SYMBOLS

2D	two dimensional
3D	three dimensional
ANOVA	analysis of variance
BIM	building information modeling
BIMs	building information models
DOF	degrees of freedom
FOV	field of view
FPS	first person shooter
GPU	graphical processing unit
HVAC	heating, ventilation and air conditioning
PC	personal computer
POI	point of interest

1. INTRODUCTION

When viewing three dimensional (3D) virtual environments on digital computers, it is possible to view different parts of the virtual environment and examine it. This process can be done by manipulating the viewpoint using various input methods. Inputs can be given to the computer using different types of hardware. This hardware can be general purpose input hardware such as computer mice and keyboards, or some hardware that is designed for more specific purposes such as joysticks. There are computers that can be interacted with by touching their display monitors and these are usually referred to as touch screen devices. Some of these devices can recognize only a single point of input on their screens, while some can recognize multiple ones simultaneously and are usually referred to as multi-touch devices. Example usages of touch screen devices can be seen in figures 1 and 2.



Figure 1. CC BY-SA 2.0 Tim Bartel. A person using a touch screen to draw a painting. [1]

Degree of freedom is a parameter used in order to define the configuration of an object in a space. Depending on the number of dimensions in the space, different numbers of degrees of freedom (DOF) are needed in order to completely define the motion of an object in space. On a two dimensional (2D) plane, only 2 DOF are needed in order to define the position; the horizontal axis, and the vertical axis. In a 3D space, 6 DOF are needed in order to define the position and rotation of an object; 3 DOF for the position and 3 DOF for the rotation.

Popular input devices such as computer mice or touch screens can detect input on a plane with two axes, and therefore support 2 DOF. In order to fully manipulate the viewpoint position in the virtual 3D space, 6 DOF are needed. Because of this, there are various viewpoint manipulation techniques that enable manipulating 6 parameters of a viewpoint with input devices that support less than 6 DOF. This issue is discussed in more detail later.



Figure 2. CC BY-SA 4.0 Markus Bachofner. A person using an interactive application on a handheld tablet computer with a touch screen. [2]

This thesis covers the design and implementation of a new single-input viewpoint manipulation technique aimed for a specific use case. This viewpoint manipulation technique is intended to work with 2 DOF input hardware such as touch screens or computer mice. Since it is a single-input viewpoint manipulation technique, it means that it does not require simultaneous detection of multiple touches on a touch screen. Therefore, it can be used with touch screen devices that do not support multi-touch. The design of the new technique is made based on previous literature.

1.1. Research Motivation

The thesis is made as part of the VIRPA-A[3] project which was funded by the former Tekes, which is now part of Business Finland[4]. A 3D visualization of a nursing home was implemented as part of this project. The building is a smart house with sensors deployed in it. The sensor data was available in order to be visualized over the 3D model. The temperature values of the rooms in the building were visualized on the 3D model.

A new single-input viewpoint manipulation technique was designed and implemented to be used for interacting with the 3D model in question. The decision to design a new single-input technique rather than a multi-input one was made after looking into the existing literature. By using a single-input technique that accepts input from 2 DOF input hardware, it is possible to support a wider range of devices that can make use of this technique. The objective of the research is to assess whether a single-input viewpoint manipulation technique can be as efficient as a multi-input viewpoint manipulation technique when used for observing a smart building 3D model.

1.2. Method

Existing literature about the basics of viewpoint manipulation was mapped. Afterward, it was decided to design a single-input viewpoint manipulation technique that can be used on a wide range of hardware including touch screen devices not capable of multi-touch and personal computers with a regular mouse. The design is based on existing literature for single-touch techniques, and is designed to be used for examining the 3D visualization of the smart nursing home in question. Aside from the new single-touch technique, a commonly used multi-touch technique was also implemented in order to compare the single-touch technique against it. Participants were recruited and user tests were made to find issues with the system. A handheld tablet computer with a multi-touch screen was used during the evaluation. Making a second iteration in order to correct the issues found from the user tests is not in the scope of this thesis. Even though the new technique can be used with both computer mice and touch screens, the focus is mostly made on touch screens because the evaluation was made in order to compare it to multi-touch techniques.

1.3. Structure of the Thesis

Chapter 2 covers the related research about smart buildings and viewpoint manipulation techniques under sections 2.1 and 2.2 respectively. The section 2.2 is later divided into subsections that cover research regarding various aspects of viewpoint manipulation. The ones which are most important for this thesis are subsections 2.2.1 and 2.2.2 which are about rotation and point of interest (POI) navigation.

Chapter 3 is giving information about how the system was designed and implemented. Background information about the building model is given in section 3.2. The new single-touch technique is later described by dividing it into several parts. Section 3.4.1 covers the functionality for selection of the floors of the building to be viewed. Section 3.4.2 covers the functionality for navigating between rooms, and section 3.4.3 covers the orbiting functionality of the new technique. Finally, the multi-touch technique that the single-touch technique is compared against, is described in section 3.5.

Chapter 4 outlines how the evaluation was made and gives information about how the user tests were conducted.

Results from the evaluation are presented in chapter 5. The chapter is divided into two sections. Section 5.1 presents the quantitative results, while section 5.2 presents the qualitative results.

Results that were presented in chapter 5 are interpreted under chapter 6. A heuristic evaluation is made under section 6.1. Section 6.2 points out some limitations of the research and provides ideas with possible improvements to the system, as well as areas that are open to further research.

Chapter 7 concludes the thesis by summing up the key results that were found. Single-input interaction techniques can be used on both touch screen devices and personal computers with a mouse. The action from the input device can

be a “touch” in case of a touch screen, and a “click” in case of a mouse. Correspondingly, the input location can be referred to as “touch position” when using a touch screen, and “cursor” when using a mouse. Throughout the thesis, both of these concepts might be used interchangeably. However, the author prefers to use “cursor” and “click” when describing the input action even though it might be used on a touch screen device.

2. RELATED WORK

This thesis focuses on viewpoint manipulation for 3D visualizations of smart buildings. Both 3D viewpoint manipulations and smart buildings are large research areas with a lot of specific topics that can be covered within them. Some of the research related to these topics are briefly outlined later in this section.

3D virtual environments are environments that are created using 3D computer graphics. These environments can be used to represent either real world locations or imaginary locations. In this thesis, the virtual environment that is used is a 3D model of an existing nursing home. Viewpoints in these 3D environments can be described as virtual cameras that are used to render the 3D virtual environment from a specific position on a 2D plane such as a computer monitor. By manipulating the position of these viewpoints, a user can look at different parts of the virtual environment. The process is similar to walking inside a house or moving an object in one's hand in real life. In both cases, the relative position of the viewpoint (one's eyes) to the object (the house or the object) changes. In the case of 3D virtual environments, the viewpoint is the virtual camera and the object is the 3D model that is being rendered.

Interacting with 3D virtual environments can be categorized into three tasks: Viewpoint manipulation, object manipulation, and application control[5, 6]. First can also be named as navigation and is about moving in the 3D environment. The second is about selecting an object in the scene and manipulating it by changing its position, rotation, or by scaling it. The last is about the communication between the user and the system which is not part of the virtual world. This thesis is focusing on the viewpoint manipulation task. More information about different types of interaction techniques for 3D environments can be found here [7]. More background information about viewpoint manipulation techniques is available later in this section.

When creating 3D virtual environments from real world locations such as existing buildings, the process mainly consists of creating the 3D computer model of the location in question. In case of smart buildings, there can be dynamic digital data available that is coming from the real building. This kind of data can be represented in the virtual environment in a way that is not possible in the real world. In the smart building that is covered in this thesis, some temperature data was available. The temperature of a location within the building can be represented by floating color coded numbers in the 3D virtual environment. More information about smart buildings is available later in this section.

Since the thesis focuses on viewpoint manipulations for smart building visualizations, first the importance of smart buildings is outlined and different ways of visualizing them are briefly covered. Later, different existing viewpoint manipulation techniques are examined.

2.1. Smart Buildings

Buildings that have automated systems deployed in them in order to control and monitor their various properties such as heating or lighting are often referred to as smart buildings[8].

Weng and Agarwal[9] point out that the heating, ventilation and air conditioning (HVAC) systems which are used in most of the commercial buildings are usually activated in static time schedules. This can result in a lot of energy waste. According to the authors, smart buildings should be able to sense various variables and automate the HVAC operations accordingly. Sensing is a crucial part and can include variables such as temperature, air quality, and occupancy information. With this information, HVAC systems might be deactivated for unoccupied areas of a building in order to save energy.

According to the National Building Information Model Standard Project Committee, building information modeling (BIM) is defined as “*a digital representation of physical and functional characteristics of a facility. A BIM is a shared knowledge resource for information about a facility forming a reliable basis for decisions during its life-cycle; defined as existing from earliest conception to demolition.*”[10]. However, this is not always the case. Usually, these digital representations which are 3D models with additional information about the building, may not be used throughout the whole life cycle of the building. It is possible to create a 3D model of a building after its construction, and for example in a case of a smart building, including the positions of some sensors in that model and bundle it as a BIM model. More information about BIM is available in Eastman et al.’s book[11].

Yan et al.[12] present a game for architectural design and visualization which makes use of BIM. A building can be designed by using BIM, and using the information available there the building can be imported into a 3D computer game environment. The users can explore the building in the game, and have some tasks such as navigating the building on a wheelchair in order to understand the accessibility issues of the building better.

Hagedorn and Döllner[13] present a system for 3D visualization of building information models (BIMs) and city models. In a BIM model, it is possible to distinguish between different parts of the building such as doors, water pipes, etc. by using BIM properties. These BIM properties are used extensively in order to visualize the buildings in Hagedorn and Döllner’s system. Roof and wall objects can be detached or “exploded” from the building in order to have a peek inside the walls. The rooms defined via BIM can be rendered as 3D meshes and can be given materials with colors reflecting certain data such as temperature, or a room containing hazardous materials can be marked with red color. Using the story information, the different stories of the building can be separated or the top floor can be tilted to gain insight of the building model.

In this thesis, using BIM was an option; however, it was decided that it is not necessary. Instead, FBX was used. FBX is a popular proprietary file format for 3D meshes. This is later described in more detail in section 3.2.

2.2. Viewpoint Manipulation Techniques

Manipulating the viewpoint in 3D virtual environments requires adjusting 6 parameters often referred to as DOF; 3 dimensions for the position of the camera and 3 for the orientation of it. The 3 parameters for adjusting the orientation are often described as pitch, yaw, and roll. An illustration of this can be seen on Figure 3. Sometimes zoom level is included as a seventh parameter if it is controlled by adjusting the field of view (FOV) of the camera[14], even though it is an intrinsic camera parameter[15]. Alternatively, zooming can be done by moving the camera along its viewing direction[16] without touching the FOV of the camera. The main challenge lies in using input devices that do not have enough DOF to be mapped to these parameters directly. There is research done about input devices with 6-DOF or more[17, 18, 19, 20] and there are some commercial products existing in the market [21, 22]. However, this kind of devices are not widely used. Most of the input devices that are widely used today such as computer mice and touch surfaces have 2-DOF. When using 2-DOF input hardware, contexts or states have to be used in order to switch between what degrees are affected by the input.

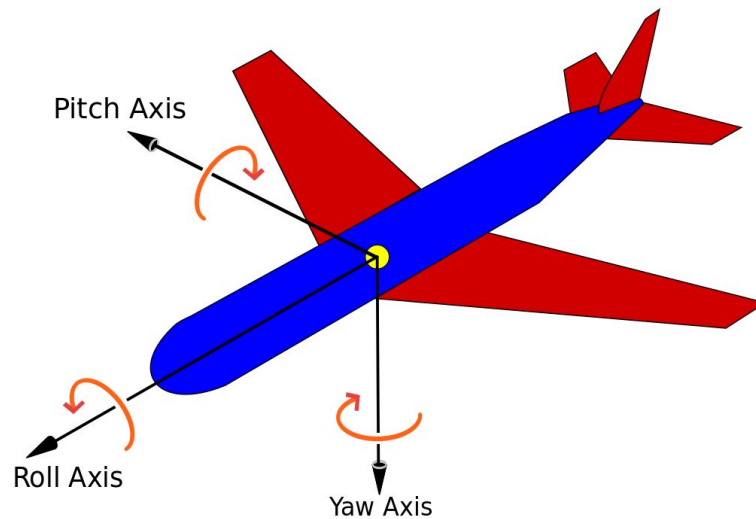


Figure 3. CC BY-SA 3.0 Jrvz and Auawise. A representation of an airplane to illustrate the three rotation axes often referred to as pitch, yaw, and roll[23].

With touch surfaces, multi-touch[24] can be used if there is the possibility to detect more than one input at the same. In that case, each finger or input in a multi-touch system can be mapped to two additional DOF. Edelmann et al.[25] implemented a multi-touch system called “The DabR”. The system uses graphical processing unit (GPU) accelerated image processing to detect the fingers touching the screen. The authors argue that using multi-touch increases the DOF as every additional finger adds 2 DOF to the interaction. The users can pan and tilt the camera by using a single finger. Rolling the camera is not supported in order to avoid confusion. When using two fingers, the users can move the camera.

Similarly, adjusting the FOV is not supported to avoid confusion. This thesis will not be focusing on the multi-touch interaction techniques as the aim is to support a wider range of input devices. Furthermore, in some cases, single-touch interactions could be more feasible on touch screen devices[26].

A common way of manipulating the camera in a 3D environment is by rotating, panning, and zooming[7]. This concept has been outlined by Phillips and Badler[16] and is widely used by 3D modeling software[27, 28]. Each of these three routines require at most 2-DOF and therefore can be used with regular computer mice. The routines can be seen as states or contexts that can be switched in between. During the rotating state, the camera is orbited around a fixed point in the space. Panning moves the camera along its horizontal and vertical axis. Zooming is achieved by moving the camera along its view axis.

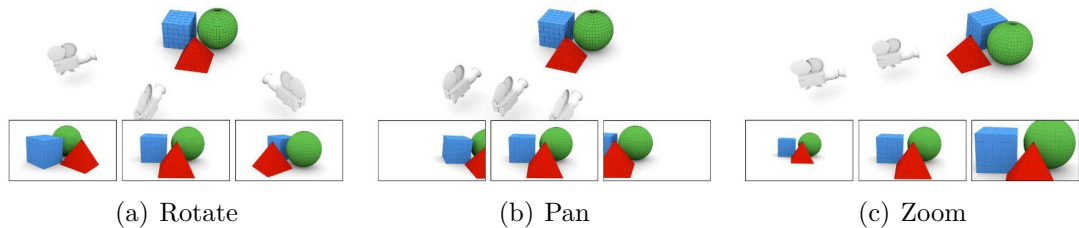


Figure 4. Rotating, panning, and zooming are the primary camera movements used in almost every 3D modeling environment[7].

Another way of navigating 3D environments is by using metaphors. These metaphors include letting the users walk or fly in the given 3D environment[29]. The walking metaphor is commonly used by computer games of the first person shooter (FPS) genre. The type of controls commonly implemented by FPS games is often referred to as “FPS controls”. With these types of controls, the user moves the camera using a keyboard, and rotates it by using the mouse at the same time. More information about this type of controls and how they evolved to come to this can be read here [30].

Houde[31] describes the iterative design process of an interface for manipulating objects in a 3D environment. The interface makes use of a regular mouse with 2 DOF. The implementation and testing of the new interface yielded some general principles that can be applied to a lot of 3D interaction interfaces. Reducing the amount of DOF the user has at a given time makes the user feel more in control, and makes the interaction easier. This can be made with context specific constraints. Making the rotation and translation actions more accessible lets the users to make repetitive actions in a smooth manner. Using a bounding box to control object positions is a consistent way for manipulating objects with different shapes.

2.2.1. *Rotation*

Bade et al.[32] have come up with four general principles for predictable and pleasing interaction techniques for 3D rotation:

1. Similar actions should provoke similar reactions.
2. Direction of rotation should match the direction of 2D pointing device movement.
3. 3D rotation should be transitive.
4. The control-to-display ratio should be customizable.

The first principle is a rather general one and implies that the actions needed to be performed should be predictable. The second one tells that the direction of the user action should correspond to the direction of the computer reaction. The third principle is about enabling the users to return to the initial viewing position; when any movement is made where the end position of the moved pointing device is the same as the initial position, the same should be with the viewing position. The last principle is about customizing the control-to-display ratio for each user; That is, setting the mouse sensitivity.

Hinckley et al.[33] have analyzed different types of 3D rotation techniques with various input hardware. The evaluation process consisted of orientation matching tasks where the users were instructed to rotate an object to match the outcome with a view of the desired result presented to them. The demographic information of the test subjects also included the handedness of the person. The shortest-arc between the resulting orientation and the desired orientation was used to rate the performance of the users. However, the rotated objects which were used in their evaluation were consisting of familiar objects for humans such as a house model. Because of that, Bade et al.[32] points out that such objects might give conclusive hints of the object's orientation, and that this might affect the result for different kinds of objects.

2.2.2. Point of Interest Navigation

Mackinlay et al. defines four different types of viewpoint movement for 3D workspaces depending on the usage task: General movement, targeted movement, specified coordinate movement, and specified trajectory movement[34]. They focus on the targeted movement, which is movement with respect to a specific target or a point of interest. This, for example, can be examining a detail of a large engineering model. They have come up with a new movement technique which they have called "Point of Interest movement". This technique comes in two versions; a basic version where the viewpoint is moved towards or away from a given POI, and the orienting version where in addition to the former, the viewpoint is oriented to face the POI. For the movement of the viewpoint, the speed at which the viewpoint is moving towards the specified POI decreases logarithmically as the viewpoint approaches the location of the POI. This allows the viewpoint to be moved rapidly, while allowing the user to have more precise control as the viewpoint moves close to the POI. Mackinlay et al. conclude that this movement technique is rapid and controlled at the same time. They argue that logarithmic motion is predictable and can help avoid disorientation.

They also point out that although this technique is good for targeted viewpoint movements, it is less effective for general viewpoint movement when there is no POI to anchor the movement to.

Moerman et al.[35] categorize the existing viewpoint manipulation techniques into two: approaches that map the device input to camera parameters, and approaches that make use of gestures. They further made a distinction whether a technique is context sensitive or not. This depends on the technique making use of scene elements. They are proposing a new navigation technique called “Drag’n Go”. The proposed interaction consists of three steps: First, the user clicks the point in the scene where they want to move. Second, the user drags the cursor to the edge of the screen. For the vertical displacement of the cursor, the avatar moves towards the desired point initially clicked. The speed of the movement is similar to the POI concept[34], the movement speed decreases as the cursor is further than the initial clicking point. The viewpoint is rotated horizontally (around the up vector of the camera) for the horizontal displacement of the cursor. The final step is the termination of the interaction when the user ceases clicking. This technique is possible to be used both with a computer mouse, or with touch screen devices. Their evaluation of the proposed technique also included comparing it to the common controls found in FPS games. The test subjects, who are mostly familiar with this kind of games have stated that the FPS controls are less demanding in terms of cognitive load. The authors acknowledge that this might be affected by the experience of the participants, but they also point out that having to constantly move the cursor in the Drag’n Go technique, when comparing to just keeping a button pressed in the FPS controls might also be the reason for this. The authors argue that this technique is simple and easy to implement, and can be used in a wide range of devices. As possible future improvements, they propose the implementation of additional movements for inspection purposes, such as orbiting.

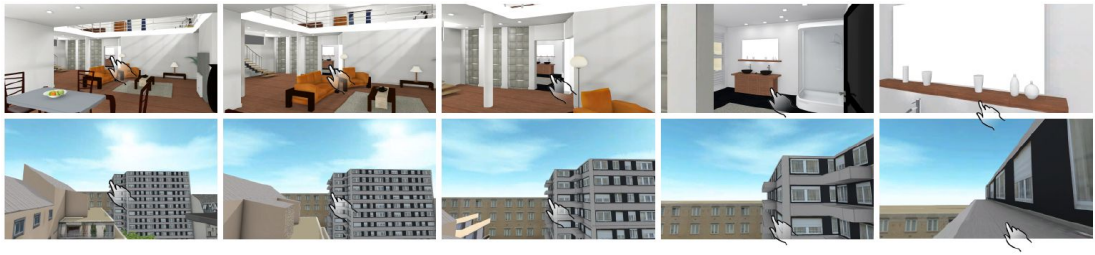


Figure 5. Drag’n Go in various situations[35].

Hachet et al.[36] state that there are 3 types of viewpoint manipulation techniques with 2-DOF devices; orbiting, flying, and point-of-interest. With the point-of-interest techniques, users can only select the target point and later have no direct control over the camera. They propose a new technique called “Navidget” to overcome this drawback. In traditional POI techniques, the camera is moved towards the clicked point in the scene and the user has no control over the distance to the selected point. With Navidget, it is possible to select an area with the circling metaphor and the camera would be moved to fit that area.

When it comes to the viewing direction with the traditional POI techniques, similarly the user has little control over the final direction of the viewpoint. In the proposed technique, a 3D widget is placed in the scene to aid with setting the target orientation of the camera. This technique does not require multi-touch, and therefore can be used with a wide range of input devices. This can be seen on Figure 6.



Figure 6. Navidget on various platforms[36].

Bowman et al.[37] presents a framework for evaluating travel techniques for virtual environments. The framework mainly consists of a taxonomy and quality factors. The taxonomy can be used to split a travel technique to components that can be grouped under certain categories. This kind of categorization can be helpful both when designing a travel technique and when evaluating one. The framework also outlines some quality factors for travel techniques. Some of these factors are speed, accuracy, and spatial awareness of the users. This kind of quality factors can be measured and quantified during the evaluation of travel techniques. Bowman et al. also present some experiments where they demonstrated how to make use of the framework.

Jankowski et al.[38] have conducted a study to compare four different interaction techniques for navigating in a virtual city on touch screen devices. The first technique is the target selection technique which is based on Mackinlay et al.'s POI concept [34], which the authors call as the Go-To technique. The second technique is making use of two virtual joysticks rendered on the user interface. The third technique is called the Slide-Grab technique and it is based on Moreman et al.'s Drag'n Go [35] with the difference that the direction of the movement can be customized. For example when the users move their fingers up, either the world can be pushed away or the camera can be moved forward depending on the preference. In addition to that, the orientation can be adjusted with dual-point input since the experiment is focusing on touch screen devices. The last one is a constrained navigation technique where the scene is navigated with strokes. The areas that the camera can move in (in this case streets) are defined, and the camera can only be moved on these defined routes. A gesture drawn on the screen indicates the direction where the camera should move, and the camera is moved along the defined route. When the user taps the screen, the movement stops. The evaluation was made based on Bowman et al.'s framework which is mentioned above [37]. The final results showed that the Go-To technique had the best task performance overall, and was the easiest to learn. Another outcome was that techniques requiring continuous input (joystick and Slide-Grab) helped

participants to be more accurate when recalling positions of the objects that they saw in the scene.

Mendes et al.[26] state that even on multi-touch capable mobile devices, it might not always be feasible to use multi-touch gestures for interaction. They have come up with a new single-touch technique called “ThumbCam” in order to interact with 3D environments on mobile devices with touch screens. They have designed a state machine to switch between different modes of interaction such as rotation, dragging, and movement. The drag state is based on Moerman et al.’s Drag’n Go technique[35]. The qualitative evaluation of the proposed technique revealed that users enjoyed using the technique, mainly due to the relaxed way of holding the device and being able to interact only with the thumb.

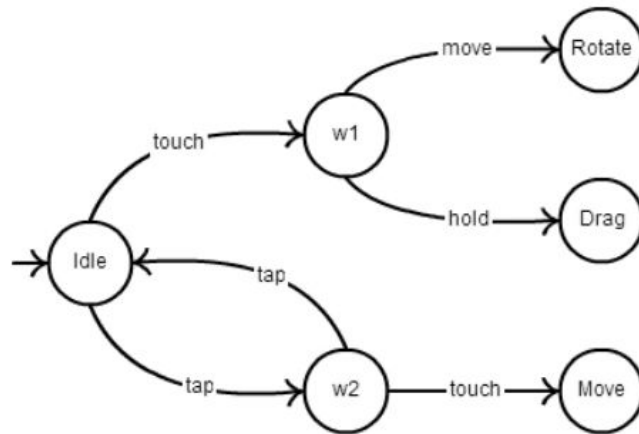


Figure 7. A simplified view of the state machine used by ThumbCam[26].

2.2.3. Orbiting

When it comes to inspecting a 3D model by orbiting around it, simply selecting a pivot point in the center of the model and moving the camera around it can be a simple solution. However, this can be a problem with larger models or models that have too different proportions. As an alternative, the surface of the inspected 3D model can be used to move the camera around it. Various techniques exist to orbit the model based on its surface in a smooth way[39, 40, 41].

Khan et al.’s “HoverCam”[39] technique provides a way to orbit around an object in a smooth manner. The algorithm used in the technique consists mainly of four steps. First, the viewpoint is moved with the user input. At this point, this is just a simple panning of the camera. Secondly, the closest point on the object is found from the new viewpoint location. After that, the viewpoint is rotated to look at the point found in the previous step. Finally, the viewpoint is moved closer or away from the closest point so that the distance between the viewpoint and the object stays the same. In addition to these steps, there’s another step where the distance is adjusted once more to overcome problems with concave surfaces. These problems can occur for example when the viewpoint can end up in multiple

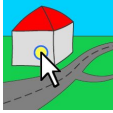
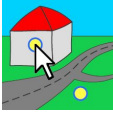
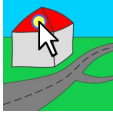
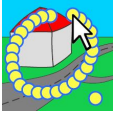
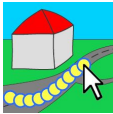
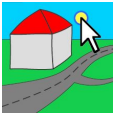
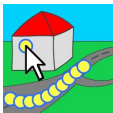
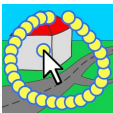
positions when looking at the same point on the object, depending on the previous viewpoint position. One of the limitations of this technique is that it consumes a lot of computing resources when finding the closest point on the object. Another limitation is that the smooth movement cannot be achieved with objects having complex structures for example if the object has a lot of concavities, or if it has disconnected parts. Declé et al.[40] propose a technique called “ScrutiCam” which is supposed to work on mobile devices with less computing power. In contrast to the HoverCam technique, ScrutiCam uses an image-based algorithm instead of relying on an algorithm for finding the closest point on the model in 3D space. A pattern is projected on the model, and based on the deformation of the pattern it is possible to approximate the possible camera movements. Consequently, the complexity of the 3D model does not affect the performance when computing the viewpoint position as much as it does when compared to the HoverCam. Ortega et al.’s “SHOCam”[41] provides a smoother way when orbiting around complex objects when compared to the HoverCam. This is done by computing multiple viewpoint positions around the original viewpoint, and interpolating the new viewpoint position based on this pre computed viewpoint set.

2.2.4. Planned Viewpoint Manipulations

Declé and Hachet[42] have made a study where they compared direct and planned camera manipulations for 3D applications on mobile devices with touch screens. Direct manipulation is where the inputs of the user affect the camera parameters in real time, while in planned manipulation the user first draws a stroke on the screen and the camera is affected by it later. Their study revealed that the stroke based interaction did not offer any significant advantage over the direct interaction. The authors also pointed out that planned camera manipulation might come in handy on mobile devices if there is lag when using direct manipulation.

Hagedorn and Döllner[43] present a sketch based navigation technique for 3D virtual environments. The users can sketch shapes on the screen, and the viewpoint is moved accordingly. The system works with two types of sketches. First are the sketches related to the objects in the scene, and the other are pen based sketches that are not related to the scene. The authors have developed a prototype where the virtual environment is a 3D city model. Most of the navigational commands are defined by object related sketches; for example, if the user clicks on a building, the camera is moved towards the building and if the user sketches a line along a road, the camera is moved on the road. An overview of some of the sketches can be seen in the Table 1.

Table 1. Examples of sketch-based navigation commands. Gestures can be used as modifiers for object-related sketches[43]. Adapted by permission from Springer Nature Customer Service Centre GmbH: Springer Nature *International Symposium on Smart Graphics* Sketch-Based Navigation in 3D Virtual Environments, Hagedorn B. & Döllner J., © Springer-Verlag Berlin Heidelberg 2008

Sketch	Navigation Command & Result	Sketch	Navigation Command & Result
	Point on a building: Finding the shortest path to the building, driving there, and looking at the building.		Point on the ground and point on a building: Flying to the marked ground point and looking at the building finally.
	Point on a building's roof: Flying up to the roof, placing the camera on top and looking around.		Point on the ground and circle-shaped gesture: Flying to the marked ground point and looking around.
	Curve on a street: Driving along the street.		Point on the sky: Soaring above ground for overview.
	Curve on a street and point on a building: Driving along the street and looking at the building finally.		Circle-shaped gesture and point on a building: Soaring above ground and looking at the building finally.

3. SYSTEM DESIGN AND IMPLEMENTATION

The system lets the user examine a smart building in a 3D environment. The building that is used for the system is a four-story nursing home that has temperature and humidity sensors deployed in it. The company that was providing the data of these sensors has only asked for visualizing the temperature values, therefore the humidity values were not visualized. Furthermore, the data is being generated on the client because the real data for each room was not available to be used at the time the system was developed, and the focus of this thesis is investigating viewpoint manipulation techniques. Visualizing this data for a building of this scale in 3D is a challenge as there are several rooms with sensors that provide data, and users might be interested in examining all of them in a rapid way. Having several floors introduce new challenges in navigating the 3D environment as the bottom floors are occluded by the higher ones, and the more generic viewpoint manipulation techniques might not be enough to navigate all of the building rapidly. A new single-touch viewpoint manipulation technique was designed and implemented. An existing multi-touch viewpoint manipulation technique was also integrated into the system to be compared against the new single-touch technique.

This section first gives some brief information about the system architecture, then covers the structure of the building model that was used, the way the temperature values are visualized to the user, and finally the viewpoint manipulation techniques that are used are explained.

3.1. System Architecture

The system is implemented with the Unity[44] game engine. Unity is a popular game engine among independent game developers. Development can be made on a single project which can be exported to various different platforms. The development is made using scenes that have objects in them. These objects belong to a hierarchy, and each object can have multiple components attached. Components can be various things such as a script, a 3D mesh that can be rendered, vectors for transformation information of the object, audio sources, etc.

In the system, the building model was imported having various parts of the building belonging to relevant hierarchies. For example, there were invisible “floor“ objects that had all the 3D models of a building floor attached to them as children. This makes it possible to shift the whole floor of the building by just modifying the position vector of the relevant floor object. The building model is explained in more detail in section 3.2. There were invisible collision meshes filling each room, and these were used when navigating between the rooms. Navigation between the rooms is explained in more detail in section 3.4.2.

3.2. Building Model

When making 3D visualizations of smart buildings, it is quite common to use BIM to maintain the model of the building, and generate a 3D mesh automatically[13, 12, 45]. The smart building that is used as part of this project does not use BIM and only has electrical drawings available. Because of this, it was decided to directly model a 3D mesh of the building based on the building plan that could be seen in the electrical drawings. The implementation would not have used any of the special components defined by BIM. There was no information available that would yield any benefits of using BIM and generating the 3D mesh automatically when compared to maintaining a 3D mesh of the building directly.

The 3D model is maintained in FBX format[46]. This allows everything (model, textures, and object hierarchy in our case) to be embedded in one file. This file can be easily imported into Unity, and the model can be added to the scene. The object hierarchy, along with the object names defined in the FBX file would be preserved. Alternatively, COLLADA[47] could have been used instead of FBX. The reason for choosing FBX was because it has a good reputation among the Unity developers community, and this has made the development process faster and easier.

Five rooms on the top floor of the building were decorated with furniture models. The furnished rooms had some armchairs, a bed, a wardrobe, and a table with a coffee machine on it. The coffee machine was put in the rooms to evaluate how the small objects can be examined in the system. When the scene is loaded, one of the coffee machines is randomly selected and shown to be empty while other the four are shown to have coffee in them.

3.3. Temperature Visualizations

Sensors in the building were maintained by the automation company Gane[48]. They have only asked to have the temperature values visualized over the model, therefore only temperature visualizations were implemented. The temperature values for the rooms are visualized by floating text which is located in the center of the top of the rooms. The orientation of these texts is fixed and does not rotate according to the viewpoint's location.

The color of the text changes depending on the temperature value being shown. If the temperature is lower than 20.5°C, the color is set to light blue to show that it is cold. For temperatures higher than 23.0°C the color is red to indicate that it is hot. For the temperatures between 20.5°C and 23.0°C, the color is green to indicate that it is average room temperature.

The temperature values are randomly generated when the scene is loaded. This is done with a script that selects a random number between two numbers specified. This happens for all of the temperature visualizations in the rooms separately, resulting in every room to have a new temperature value each time the scene is loaded. For most of the rooms, the lowest value is 19.0 and the highest value is 23.5. There is one decimal place. For evaluation purposes, two rooms are selected to have the lowest and the highest temperature in the whole building respectively.

This selection is made manually by setting the random value range to be between 10.0 and 15.0 for the cold room, between 25.0 and 30.0 for the hot room.

3.4. Single Touch Viewpoint Manipulation

It was decided to use a technique that does not require multi-touch and that can be used both in computers with regular mice or touch screens. The existing literature proves that this kind of interactions can be sufficient, and sometimes can be even better than interaction techniques requiring multi-touch or special hardware (such as 6-DOF input devices.)

POI viewpoint manipulation techniques seemed quite promising for navigating large areas rapidly using 2-DOF input devices. However, most of these techniques are not sufficient for navigating between the different floors of the building. Furthermore, most of them make raycasts on the rendered model itself and navigate there. This is good if the navigation technique is to be made compatible with any 3D model, however in this case various optimizations can be made for a 3D model of a house.

The proposed technique is divided into three parts: First, a method for switching between the different floors of the building is explained. Second, navigating between different rooms on the selected floor. Finally, orbiting (rotating around) a selected room is explained. The number of DOF that can be controlled by the user at a given time is tried to be kept as low as possible by dividing the interaction into contexts, as this would make the users feel more in control and result in an easier interaction [31].

3.4.1. *Switching Between Floors*

When the scene is loaded, the building is presented on the top of the screen. In order to continue with the interaction, the user should select a floor of the building to work with. The selection of the floors is made as a drawer metaphor. The user can click and slide the floor towards the middle of the screen, like pulling out a drawer. When the user stops clicking, the floor continues to slide with the speed it was moving with and snaps to its final position. If the movement speed at the time of releasing the click was too low, the floor moves to its snap position with a default speed.

There are two final positions where the floor can snap to. The floor can snap to the active position which is in the middle of the screen, at this point the floor is selected and can be worked with. Alternatively, the floor can snap back to where the building is and become inactive. The snapping position depends on the speed and position of the floor when the click was ended. In other words, if the floor was not pulled out far enough and the movement speed was too low by the time the clicking ended, the floor is slid back to its initial position.

If there was an active floor which was previously slid out and the user slides out a new floor from the building, the previously active floor is moved back automatically as the new floor is slid out of the building.

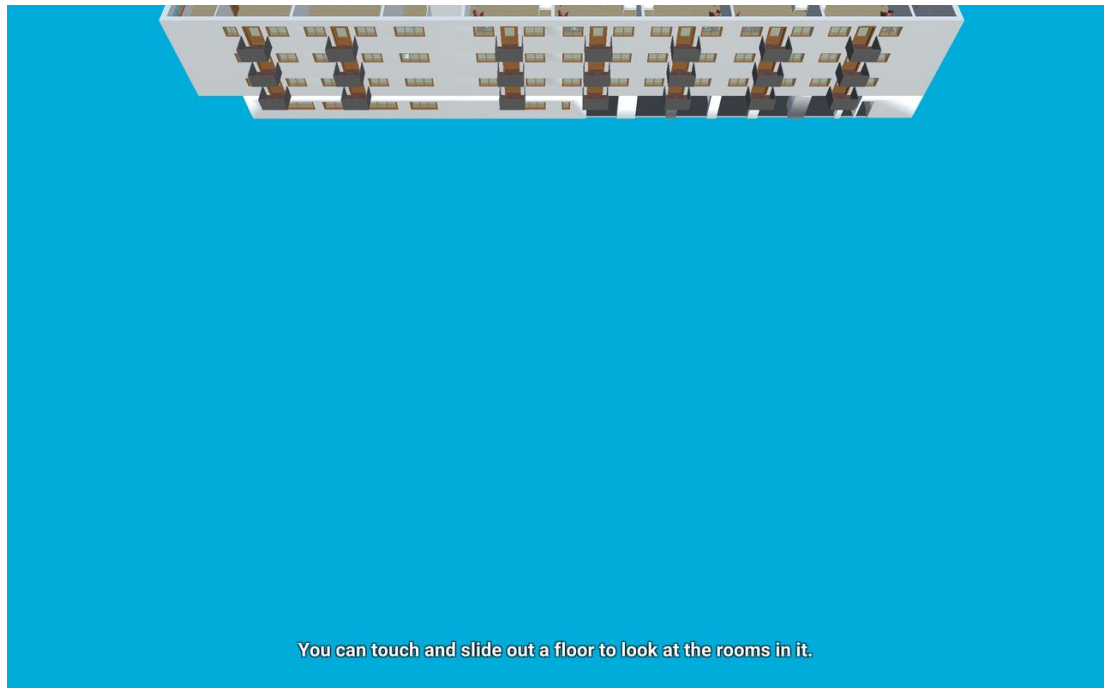


Figure 8. The initial view when the single-touch version is loaded.

The initial implementation did not take into account the movement speed of the floor. When the click ends, the floor was snapping with a default speed to its final position. The snapping position was simply determined if the floor was slid out further than a certain threshold. However, this is not the case in the final implementation. The reason for this is that during the early stages of the development, the interaction was tested on a personal computer with a regular mouse. Once the prototype was deployed on a tablet, the interaction felt uncomfortable for the author and the few people who were asked to try it out. As a result, the floor selection system was modified accordingly. The final version, as stated before is making use of the final movement speed of the floor when the click ends. The snapping movement speed and snapping position are based on that speed.

3.4.2. Switching Between Rooms

The user can navigate closer to the rooms of the selected floor. The user can click a room and move the cursor towards the lower edge of the screen to zoom into a room. This navigation is based on the Drag'n Go[35] technique. However, instead of making a raycast to the building model, invisible collision meshes are used. The collision meshes represent each of the rooms available in the building, and the user is navigated to the center of the desired room. With this approach, the users cannot end up navigating to irrelevant areas such as corners of the rooms. Long corridors or very big rooms can be separated into several virtual rooms.



Figure 9. The single-touch version with the top floor slid out.

The interaction for zooming into a room begins when the user starts clicking a room inside the active floor and ends when the click is released. As the user moves the cursor from the initial click position to the lower edge of the screen, the camera is logarithmically moved towards the clicked room. When the click is released, the distance the camera has traveled with respect to the total distance is checked. The total distance is the distance between the camera's initial position and the final position that it was moving towards. If the camera has traveled further than a certain percentage along the path, it is snapped to its final position. Otherwise, it is snapped back to its initial position. The snapping speed is fixed.

In addition to performing the previously explained action when viewing the floor, it can also be performed when the camera has been already zoomed into a room. The only difference would be that the initial position of the camera is the viewing position of the previous room, instead of the zoomed out view of the whole floor. This lets the user to move between the rooms in a rapid way without zooming out. The user just has to click the neighboring room in the view and move the cursor towards the lower edge of the screen to align the camera with that room. This is demonstrated in the Figure 12.

When zoomed into a room, the user can zoom out to view the whole floor. The process is similar to zooming into a room. In order to zoom out, the user clicks anywhere on the screen and moves the cursor towards the upper edge of the screen.

In contrast to the Drag'n Go, the rotation of the camera is fixed throughout this stage of the interaction. The vertical movements of the cursor have no effect.

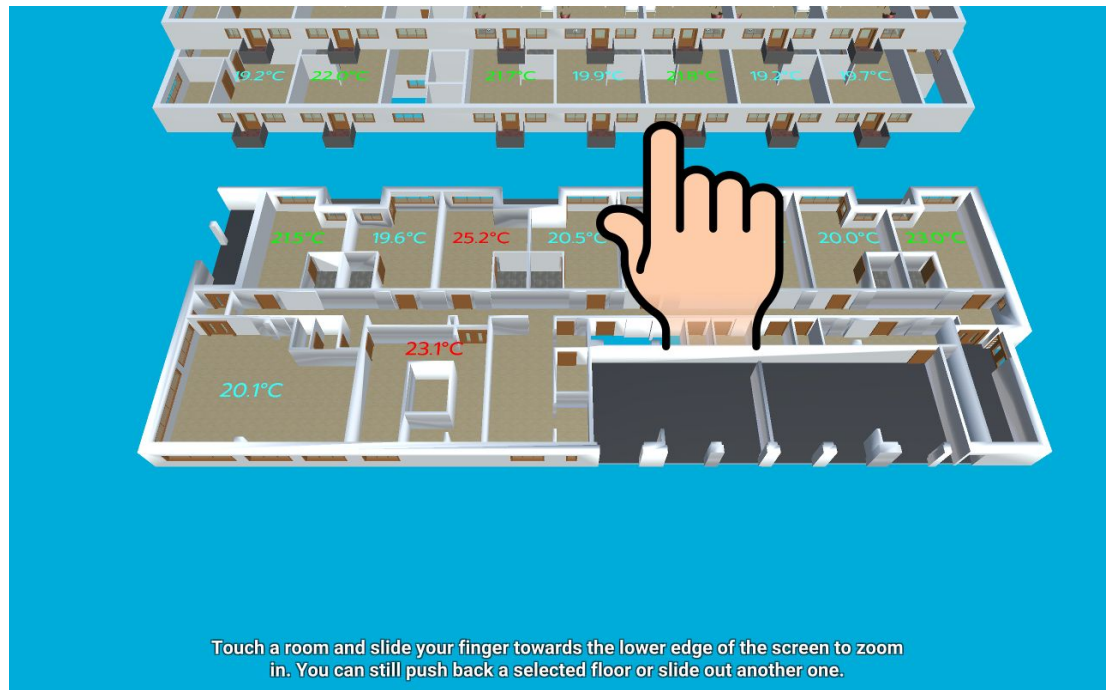


Figure 10. Switching to a different floor in the single-touch version. The position of the cursor is demonstrated with a finger icon and it is moving towards the bottom of the screen. The third floor was clicked, and is being pulled out of the building by the user. The first floor was selected previously and now it is automatically moving back to the building. The second and fourth floors are fixed in their original positions. However, the second floor is not seen in the picture because it is occluded by the third floor which is being pulled out.

3.4.3. *Orbiting*

Orbiting can only be done if the camera is navigated towards a room as a result of the previous stage. The user has to tap the screen or quickly click and release the pointer once to enable orbiting. When the orbiting is enabled, the horizontal input movements would orbit the camera around the center of the room. The vertical input movements would tilt the camera up and down. If the user taps the screen once more at this stage, the orbiting action is ended and the camera is moved back to the initial position of this stage.

3.5. Multi-Touch Viewpoint Manipulation

A commonly used multi-touch technique was also integrated into the system. When using this technique the selection of the floor is made with a dropdown menu in the user interface. When the desired floor is selected, the floors above the selected floor are made invisible while the selected floor and the floors below it are made visible.

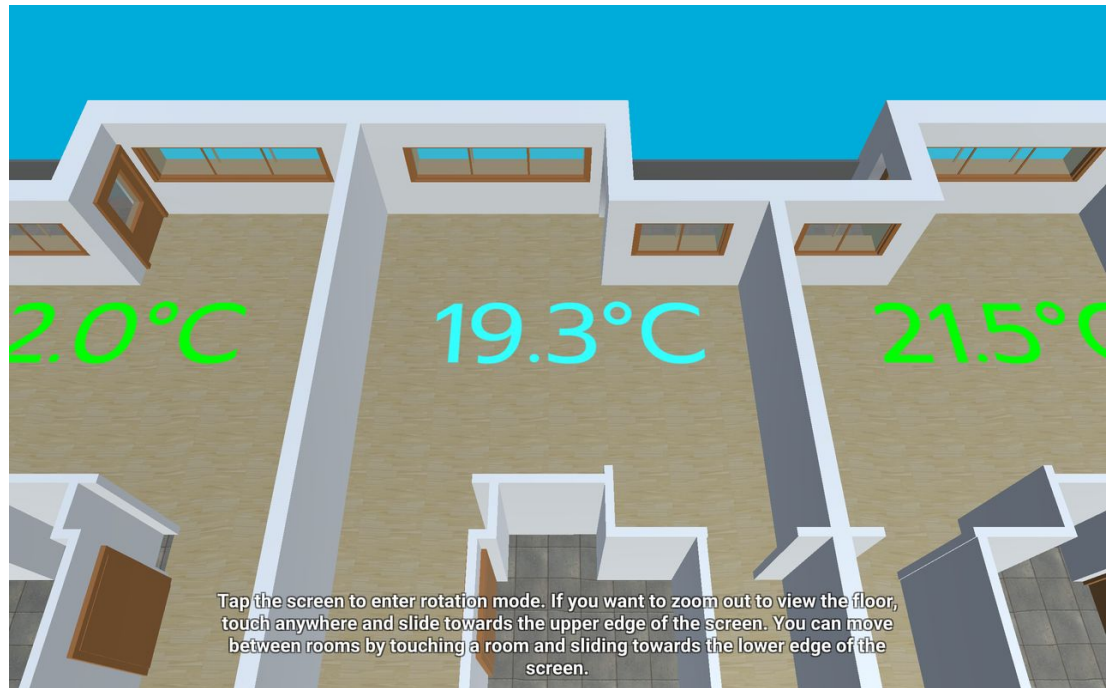


Figure 11. A room view in the single-touch version.

The user can orbit around the model by touching the screen with one finger. Similar to the orbiting stage of the single-touch technique, the horizontal movements would orbit the camera around the model and the vertical movements would tilt the camera up and down.

When the user touches the screen with two fingers, they can either pan the view or adjust the zoom. Zooming is made with the “pinch” gesture; when the two fingers are moved away from each other, the camera moves away and vice versa. Panning is the action of moving the building model. For panning, the middle point between the two fingers is taken into consideration. As this point moves horizontally, the model is moved along the same axis with respect to the



Figure 12. From left to right: Process of moving between the rooms when zoomed in. Click position is shown with a hand icon. The camera is moved to the room on the right as the cursor moves to the lower edge of the screen.

view. The same goes for the vertical movement. The actual direction the model is moving in the virtual world depends on the orientation of the camera. For example, when the two fingers are horizontally moved towards the right edge of the screen, the model is also moved towards the right edge along the screen.

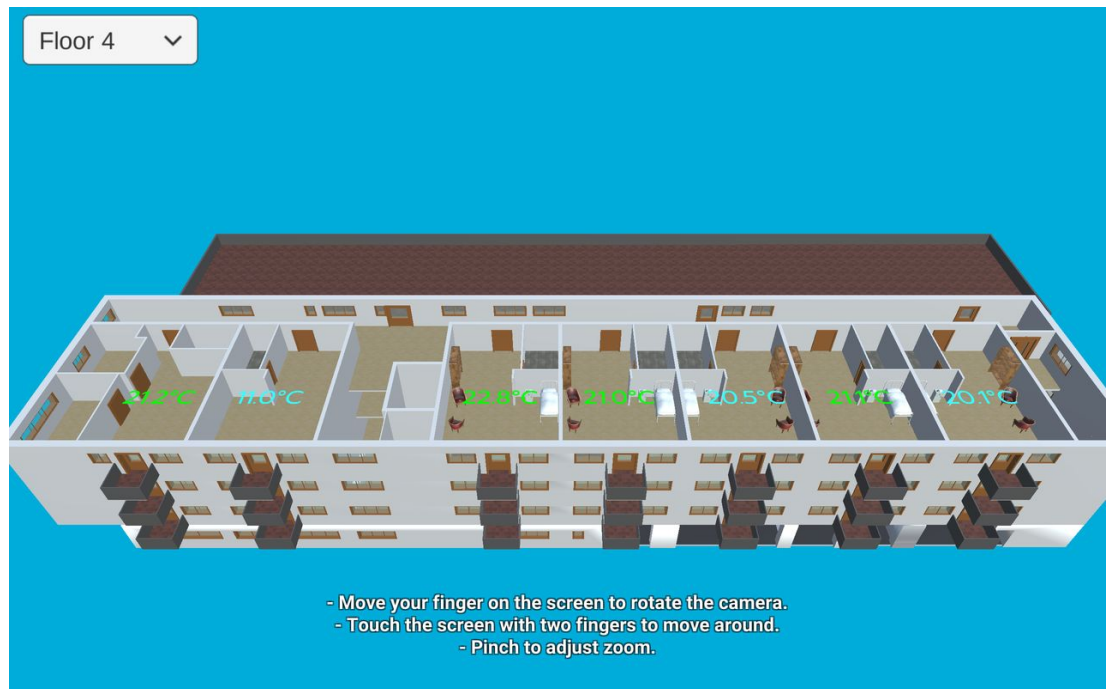


Figure 13. The initial view of the multi-touch version. The fourth floor is selected by default.

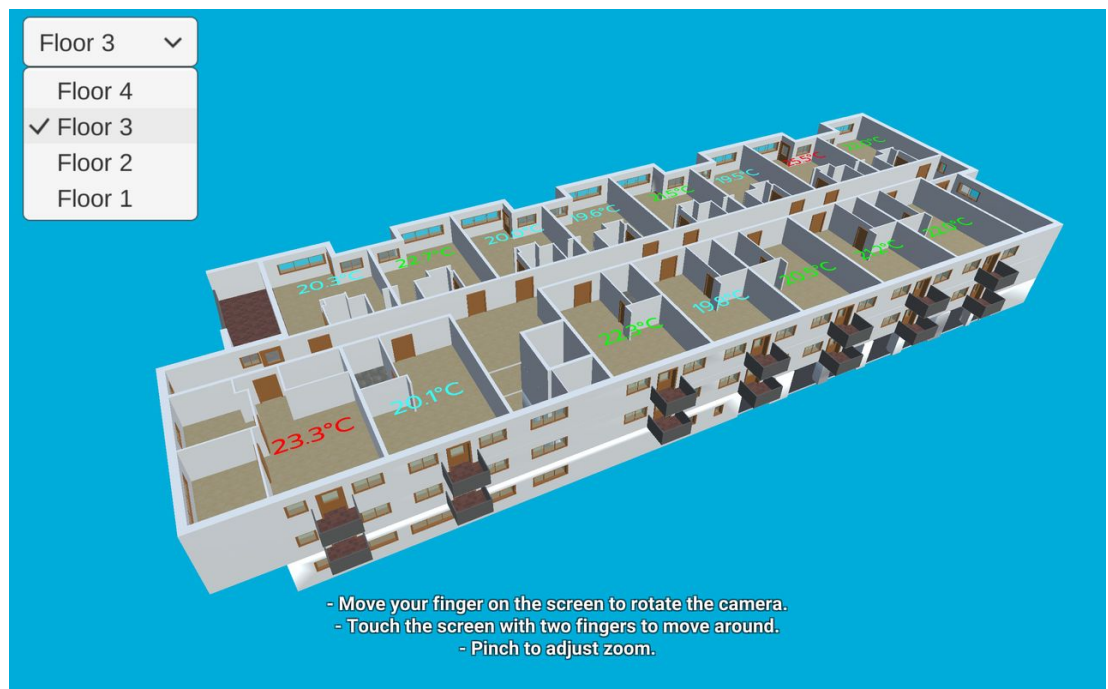


Figure 14. Floor selection on the multi-touch version. The third floor is selected. All of the floors above the selected floor are made invisible.

4. EVALUATION SETUP

In order to evaluate the effectiveness of the new single-touch technique, user tests were made. 10 participants were recruited. Most of the participants had good experience with 3D programs or games. This can be seen on Figure 15 in detail. When asked if they are using a mobile device or a personal computer (PC) more, two of them replied that they are using both the same while the rest replied to be using PCs more. When asked if they prefer a regular computer mouse or a touchpad, only one participant answered that they prefer a touchpad while the rest preferred a mouse.

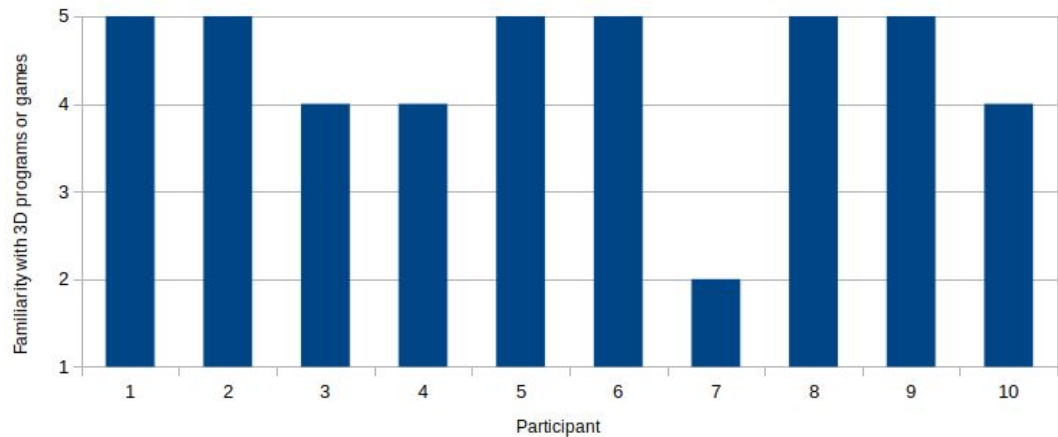


Figure 15. Participants and their familiarity with 3D programs or games. 1 is least familiar, while 5 is the most familiar on the vertical axis.

The participants were required to complete a set of tasks when using one of the viewpoint manipulation techniques available. After completing the tasks with one system, the participants were given a post questionnaire to fill. After the post questionnaire, they had to complete the same tasks with the other technique and fill the post questionnaire again. Half of the participants started with the single-touch technique, and the other half started with the multi-touch technique. A Samsung Galaxy Tab Pro 8.4 (SM-T320) was used when conducting the user tests. The device has an 8.4 inch screen with 2560x1600 pixels of resolution.

The participants were told to think aloud when they are using the systems. They were also told to try to complete the tasks as accurate as possible, rather than completing them as fast as possible. Since accuracy and speed are one of the two main metrics for user interface evaluation, doing this would keep the accuracy as fixed as possible for each participant so that the time it took to complete the tasks can be used as a metric. The screen of the testing device was also recorded using a software to measure how long it took to complete the tasks. The touch points on the screen were also rendered on the video. This also made it possible to analyze the videos to see what kind of mistakes the users did when using the system. When participants were completing the tasks, they were observed and notes were taken. Some of these evaluation practices such

as think aloud protocol, consideration of the speed and accuracy trade off, and summative evaluation where two systems are compared against are based on the book written by Bowman et al.[49 Chapter 11].

Before giving the tasks to the participant, a quick introduction to the selected viewpoint manipulation technique was given and the controls were explained. There were 5 main tasks.

Firstly, the user was asked to find the coldest room in the whole building. The user had to switch between all of the floors and observe the temperature values in each of the rooms and try to remember the coldest one they have seen.

The second task was divided into four sub tasks: Finding the coldest room on the first floor, the hottest room on the second floor, the coldest room on the third floor, and finally finding the hottest room on the fourth floor.

For the third main task, the users were asked to find the hottest room in the whole building and find the picture on the wall inside that room. Finding the hottest room is a similar process to the first task. However, afterward, the user has to rotate the viewpoint in order to view the picture which is hung on a wall which is occluded when looking with the default camera orientation.

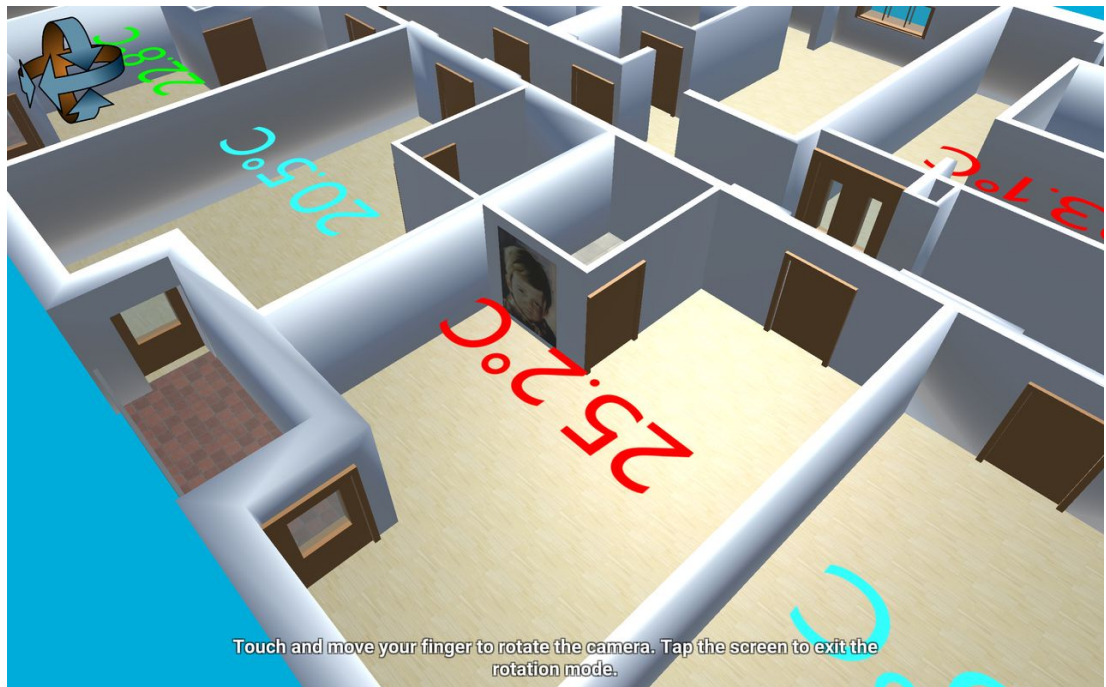


Figure 16. A view of the hot room when using the single-touch version. After zooming to the room the camera was rotated in order to see the picture behind the wall.

The fourth task was a view matching task. The view matching task was started after the user would find the picture on a wall as a result of the previous task. The user was shown a picture on a paper, depending on what system they are using. The picture was a screenshot of the system. They had to match the view on the device to be similar to the one on the picture. The picture that was shown with the single-touch technique can be seen on Figure 18 and the picture that

was shown with the multi-touch technique can be seen on Figure 19. In order to change the viewpoint to the one shown in the pictures, the user had to take roughly the similar steps in both of the systems. First, they have to change the floor that they are viewing, then they have to zoom into a room and rotate the view. There was no implementation made on the software to detect when the user has matched the view. It was up to the observer to decide when the user has matched the view. The timings were calculated after the tests, using the video recordings. Once the participant completes the fourth task, they are viewing the top floor from a zoomed in view.

At this point, the fifth and last task is given and the users are told that there are five furnished rooms on the top floor that they are viewing. One of these rooms has run out of coffee and the participants are asked to find this room which has the empty coffee machine by navigating on this floor without zooming out, in order to evaluate the movement capabilities between the rooms when zoomed in. For the single-touch technique, this means moving the view to the neighboring rooms when already zoomed into a room. A view of the coffee machines inside the furnished rooms can be seen in Figure 17.



Figure 17. A zoomed view of the furnished rooms when using the multi-touch version. The coffee machine on the right has coffee in it while the one on the left does not.

Once these tasks are completed, the post questionnaire is made and then this procedure is repeated with the other viewpoint manipulation technique. The hottest and coldest rooms are always located in the same places for each of the viewpoint manipulation techniques so they are consistent for every participant. However, they are located in different places for the single-touch and for the multi-touch technique so that the participants have to find those rooms again

when the technique is switched. The way the temperature values are set for each room is explained in section 3.3.



Figure 18. The view that was shown to the participant for the view matching task when using the single-touch technique.

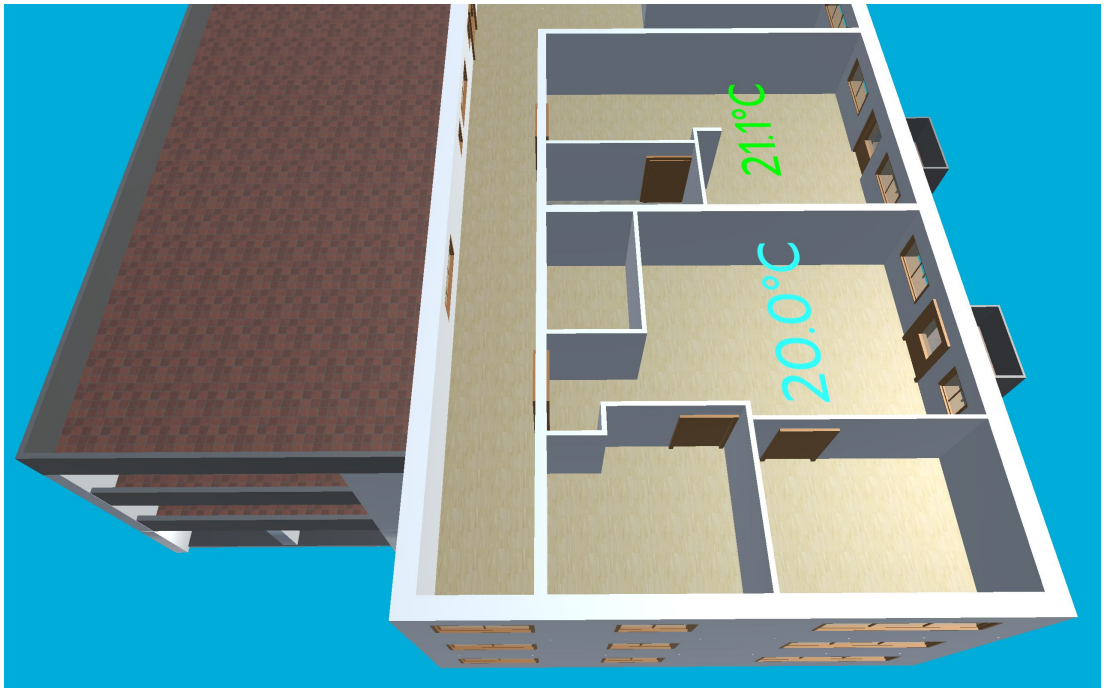


Figure 19. The view that was shown to the participant for the view matching task when using the multi-touch technique.

5. RESULTS

5.1. Quantitative Results

5.1.1. View Matching Task

During one of the tests, the application has crashed during the view matching task. The timings from that task were unreliable and therefore the results of that participant are excluded from the presented results here. Most of the users managed to complete the task faster on the multi-touch system. For a few of the users, it took almost the same time to complete the task using the single-touch technique and the multi-touch technique. The comparison of the timings can be seen on Figure 20. The average time to complete the task was 91 seconds using the single-touch technique, and 64 seconds using the multi-touch technique. The timings were analyzed using one factor analysis of variance (ANOVA) analysis and the p-value is 0.053. The p-value is the probability for the null hypothesis to be true. In this case, it is the probability of the choice of system (single-touch or multi-touch) not having an effect on the time values. A more detailed overview of the task completion times, together with the demographic information of the users can be seen on the Table 2.

Table 2. Completion times for the view matching task in seconds, together with demographic backgrounds of the users

User	First technique that was tested	Familiarity with 3D programs or games (1-5)	PC or mobile device preferred	Mouse or touchpad is preferred	Single-touch time	Multi-touch time
1	single	5	both	mouse	72	73
2	multi	5	pc	mouse	113	117
3	single	4	pc	mouse	50	36
4	multi	4	pc	mouse	100	48
5	single	5	both	mouse	85	45
7	single	2	pc	mouse	129	83
8	multi	5	pc	mouse	57	36
9	single	5	pc	mouse	129	64
10	multi	4	pc	touchpad	89	77

5.1.2. Post Questionnaire

The post questionnaire is based on the Questionnaire for User Interface Satisfaction (version 5) by Chin et al.[50]. Questions that seemed to be irrelevant

View Matching Task Completion Times Per User

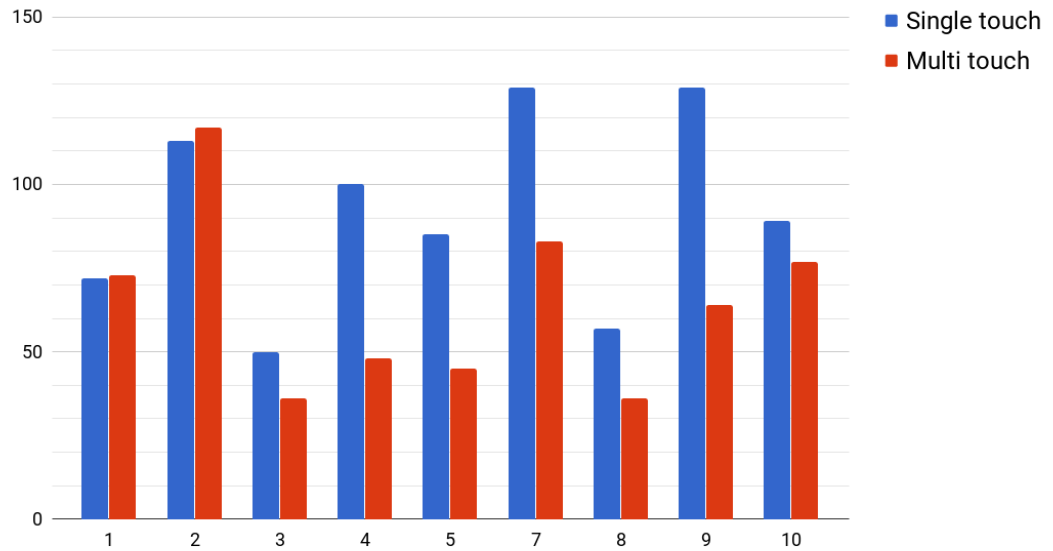


Figure 20. Comparison of completion times for the view matching task in seconds per user.

were left out, and some application specific questions were added. In order to make it more simple to fill the questionnaire, the answers were gathered on a scale of 7 instead of 10. For each of the questions, one factor ANOVA analysis was made in order to compare the answers the users gave for the single-touch technique and the multi-touch technique. The used questionnaire and the results can be seen on Table 3.

Table 3. Results of the post questionnaire. For the answers given for single-touch and multi-touch system, the mean values are presented together with the corresponding standard deviation in parentheses. The higher mean value is marked with bold. The P-value for the ANOVA analysis is also presented. P-values lower than 0.05 are underlined

Title (low (1)/high (7))	Single-touch	Multi-touch	P-value
Overall reactions to the software (terrible/wonderful)	5.7 (0.48)	6.3 (0.67)	<u>0.034</u>
Overall reactions to the software (difficult/easy)	4.8 (1.54)	5.9 (0.73)	0.057
Overall reactions to the software (frustrating/satisfying)	5.4 (1.07)	6.0 (0.47)	0.123
Overall reactions to the software (dull/stimulating)	5.8 (0.78)	5.7 (1.05)	0.813
Overall reactions to the software (rigid/flexible)	5.1 (1.44)	5.4 (1.17)	0.617

Table 3. Results of the post questionnaire (continued)

Title (low (1)/high (7))	Single-touch	Multi-touch	P-value
Characters on the computer screen (hard to read/easy to read)	6.7 (0.48)	6.5 (0.70)	0.469
Organization of information on screen (confusing/very clear)	6.3 (0.48)	6.6 (0.51)	0.196
Use of terms throughout system (In help messages) (inconsistent/consistent)	6.0 (0.94)	5.8 (1.13)	0.673
Position of help messages on screen (inconsistent/consistent)	6.4 (0.69)	5.8 (1.22)	0.196
Position of temperature values of the rooms (inconsistent/consistent)	6.8 (0.42)	6.8 (0.42)	1.000
Learning to operate the system (difficult/easy)	5.0 (1.63)	6.7 (0.48)	0.005
Exploring new features by trial and error (difficult/easy)	5.4 (0.96)	6.0 (0.94)	0.176
Remembering the inputs for controlling the system (difficult/easy)	5.3 (1.49)	6.6 (0.51)	0.018
Tasks can be performed in a straight-forward manner (never/always)	6.0 (0.81)	6.4 (0.51)	0.206
Help messages on the screen (unhelpful/helpful)	6.3 (0.82)	5.5 (1.35)	0.127
System speed (too slow/fast enough)	6.9 (0.31)	6.6 (0.69)	0.232
System reliability (unreliable/reliable)	6.0 (1.15)	6.2 (0.78)	0.656
Correcting your mistakes (difficult/easy)	5.8 (1.31)	6.6 (0.69)	0.106
Experienced and inexperienced users' needs are taken into consideration (never/always)	4.7 (1.49)	5.3 (1.33)	0.356
Navigation between rooms (confusing/very clear)	4.5 (1.64)	6.4 (0.96)	0.005
Navigation between floors (confusing/very clear)	6.7 (0.48)	6.7 (0.67)	1.000

Table 3. Results of the post questionnaire (continued)

Title (low (1)/high (7))	Single-touch	Multi-touch	P-value
Color coding of temperature values (unhelpful/helpful)	6.8 (0.42)	6.7 (0.48)	0.627
Orbiting around a room (difficult/easy)	6.4 (0.84)	6.2 (0.78)	0.590
Finding small objects in the rooms (difficult/easy)	6.7 (0.67)	6.3 (1.56)	0.468

5.2. Qualitative Results

5.2.1. User Comments

There was varying feedback from the users regarding selecting the floors of the building. On the single-touch system, two users have mentioned that it was difficult to select the floors because colliders were too close to each other. It was difficult to understand that the model was a building for one user, and they have suggested that the model could be brought closer a bit to fix this issue. A user has pointed out that it was difficult for them to know which floor number is being viewed. Additionally, there was a user who told that the drawer metaphor was very cool. Another user has commented that the floor selection in the single-touch system was fast and easy to use, making it the most convenient operating mode in the system.

When it came to the floor selection in the multi-touch system, a user has mentioned that navigating floors on the multi-touch system was better than navigating the floors in the single-touch system. Another user has mentioned that they preferred the dropdown box for selecting the floor because it was hard to understand the floor numbers on the single-touch system. One of the users told that the floor selection part has made the multi-touch system stand out.

A user has told that the rotation and navigation actions were too similar on the multi-touch system. Another user has mentioned that using two fingers to rotate the viewpoint was more common in the applications the participant had used before, and that this difference was the only issue regarding the multi touch system. There was a suggestion that zooming by double tapping would be nice in the multi-touch system.

There were several comments regarding the navigation between the rooms in the single-touch system. One of the users has found it a bit confusing, and told that it took a couple of trials for them to learn how to use it. Another user has mentioned that it would be nice to have some highlighting or hints for the rooms that the user can zoom in. A user has told that moving between the rooms in the single-touch system was the most difficult part for them, and that they would prefer one click in order to move between the rooms and double click to enter the rotation state. There was also a user who pointed out the positive findings that made the single-touch system to stand out. For them, during the room viewing

state of the single-touch system it was easy to rotate, find objects, and navigate between rooms.

Multiple users have mentioned that they liked the color coding of the room temperature visualizations, and that it helped them. Additionally, two users told that they would prefer having more colors for the temperature visualizations.

There was also feedback received about the help texts on the screen. A user has mentioned that shorter help texts would be better. Another user has mentioned that they did not need any help messages at all.

One user has told that they prefer the single-touch system because it was easier for them to navigate there. It was also mentioned that overall, the design of the single-touch system looks professional.

5.2.2. Other Observations

The users were observed and notes were taken when they were completing the given tasks.

When trying to select a floor in the beginning of the test with the single-touch technique, a user has tapped a floor in order to select it. After going back to the floor selection mode again following the tasks involving rotation, a user has tried to rotate the view again by tapping the screen in the floor view mode.

There were issues with zooming into rooms in the single-touch system. A user kept pressing a room and have not moved their finger while trying to zoom in. Another user has tapped the room in order to zoom in there.

In the zoomed in phase of the single-touch technique when viewing a room, a user has tapped the neighboring room to move there.

When using the multi-touch technique, two of the users have tried rotating the view with two fingers and moving with one. There was also an issue where the action of zooming and moving was getting mixed. When some users would try to orbit the scene by moving both of their fingers along the screen, the distance between their fingers would change in the process and they would affect the zoom level without having the intention to do so.

It took a while for three of the users to notice that the colors of the temperature value visualizations represent different thresholds of colors.

6. DISCUSSION

By looking at the yielded results, it is possible to see some areas in the single-touch technique that can be improved. This chapter includes the interpretation of the evaluation results and points out some limitations in this research together with improvements that can be made. A heuristic evaluation is also presented here.

The results from the post questionnaire are mostly not reliable. Only some of the questions yielded a P-value lower than 0.05 from the ANOVA analysis. Those questions can be seen in the Table 3 where the P-values lower than 0.05 are marked with bold text and are underlined. The users seem to favor the multi-touch technique over the single-touch technique in all of the questions which have a P-value lower than 0.05. By looking at these results, it is at least possible to point out the areas which the single-touch technique is clearly lagging behind the multi-touch technique.

The most noticeable issue regarding the single-touch technique was the movement between the rooms when the user is zoomed in. This can also be seen in the Table 3 where the users favor the multi-touch technique when answering to the question regarding navigation between the rooms. One possible reason for this might be because users tended to drag the desired room to the middle of the screen, as it was a more natural movement. In the implementation, only the vertical movement of the cursor is taken into account and the horizontal movement is ignored. The technique was designed in this way based on Moerman et al.'s "Drag'n Go" technique[35] and Jankowski et al.'s Slide-Grab technique[38]. Both of those techniques are taking into account only the vertical movement of the cursor when moving the viewpoint.

Drawer metaphor for floor selection was appreciated by some of the users. However, there were cases when it was not even clear for some users in the beginning that they are looking at a building. For some of the users, the building model was too far away and it was difficult for them to select a floor because the floors were too close to each other and they were too small on the screen. In the system that was tested, the building model only had four floors. Furthermore, there were difficulties to see which floor number the user is viewing at the moment. If a building model with more floors is to be used, the floor selection part of the single-touch technique needs more improvements.

There were no negative comments from the users regarding the rotation state in the single-touch technique. There were also no noticeable observations made which might hint any issues.

6.1. Heuristic Evaluation

When designing the single-touch technique, various findings from the previous literature was kept in mind as guidelines. This section makes an overview about how the final design is coherent with the results of existing literature.

Overall, when using the application the user has control of only one degree of freedom at a given time. The only exception is when the user is orbiting around

the room when they have control of two DOF. This is in accordance with the results found by Houde[31] that the user should have control over as few DOF as possible.

The orbiting state fulfills three of the four principles outlined by Bade et al.[32]. The only principle not fulfilled is having the control-to-display ratio customizable. Due to tests being conducted on one type of hardware, it was decided not to include the possibility to have the control-to-display ratio adjustable from the user interface.

The interaction technique for zooming into a room and moving between rooms is based on the POI concept of Mackinlay et al.[34] and Moerman et al.'s Drag'n Go technique[35]. In the Drag'n Go technique, the user can move their finger horizontally to manipulate the viewpoint's orientation horizontally. This was not included in the new technique. The decision was based on Houde's[31] results that showed it is easier for the user if they are controlling fewer DOF at a given time. It was decided to have a separate state solely for orbiting, inspired by the state machine based interaction technique by Mendes et al.[26]. The user can click the screen to switch between movement and rotation contexts.

The new technique was designed and developed for a specific scenario, 3D visualizations of smart buildings. The parts of the 3D model that the user can zoom into were narrowed by using collision meshes. This was made with the intention of making the interaction easier for the user by preventing them from zooming into irrelevant locations in the building.

The same technique is used for zooming into a room from the floor view, and for moving to a neighboring room when already zoomed into a room. When zooming into a room, the movement of the interaction gesture is usually in the same direction as the movement of the viewpoint. However, when moving to a neighboring room this can be different. The user has to click the desired room and move the cursor to the bottom of the screen in a straight vertical line. If the room that the user has clicked to is on the right or left of the previous room, then the movement of the viewpoint is horizontal. This scenario felt unnatural for some users as they tended to move their fingers also along the horizontal axis. They would tap the desired room and drag it towards the center of the screen. One of Bade et al.'s[32] principles about 3D rotation was stating that the direction of the rotation movement should match the direction of the pointing device movement. Seems like this can also be applicable when it comes to 3D movement.

From the technical perspective, the designed single-touch technique was simple to be implemented and it works on a wide range of hardware since it does not require multi-touch screens. Additionally, the feedback from the users indicate that the new technique can be at least a comparable alternative to an existing multi-touch technique.

6.2. Limitations and Future Work

The selected method for measuring the time it took to complete the view matching task was not so efficient. The finishing time for the task was decided by a human and might vary when repeated.

The results indicate that there is a lot of room to make improvements when it comes to moving between the rooms. Perhaps the most apparent would be to modify the behavior based on the observations and also take into account the horizontal movement of the cursor. Instead of dragging the cursor to the bottom of the screen, the user can move the cursor towards the middle of the screen in order to move to the clicked room.

Another area that can be improved is the floor selection. The selected building model consisted of only four floors. If a building with more floors is to be used, the existing implementation would not be enough to cover functionality for selecting all the floors. Furthermore, some users had difficulties selecting the floors because the building model was too far and the area to be clicked was too small. Further research can be made in order to find ways of making it easier for the users to select floors, and make it possible to select floors on a larger building such as a skyscraper.

There was a small number of participants for the evaluation, and a very large proportion of them were very familiar with 3D applications or games. If a more reliable research is to be made, a more diverse set of participants can be selected. Additionally, the way for determining the time it took to complete the view matching task was dependent on the human factor. If a more precise way for measuring the time it takes to complete a task is to be implemented, combined with more view matching tasks, it would be possible to obtain more detailed data in order to measure the quality of the designed interaction technique. A further investigation can be made in order to try to find any correlations between the demographic background of users and task completion times.

7. CONCLUSION

A new tailor-made single-input interaction technique was designed and implemented to be used with a 3D visualization of a smart building. The design was influenced by existing single-input viewpoint manipulation techniques that were described in the literature. User tests were made, and the new technique was compared to an existing multi-touch technique in order to assess the feasibility of using a single-input interaction technique instead of a multi-touch one.

There were not so many participants involved and the demographic backgrounds of the participants were not so diverse. Therefore, it is difficult to make a conclusion about the new technique being as good as an existing multi-touch technique. However, the user tests yielded results that indicate clear areas which are open to improvement when it comes to the new technique. Navigation between the rooms seemed to be the most problematic part.

In the system outlined in this thesis, it was possible to observe a 3D visualization of a smart building using a single-input technique. The feasibility of preferring a single-input interaction technique over a multi-touch one can be decided on a case-by-case basis, and multiple design iterations might have to be made in order to get the most out of it.

8. REFERENCES

- [1] Bartel T. (2011), Touch screen art. Obtained from: <https://www.flickr.com/photos/avatar-1/6035706612> Licensed under CC BY-SA 2.0: <https://creativecommons.org/licenses/by-sa/2.0/legalcode>.
- [2] Bachofner M. (2015). Obtained from: https://commons.wikimedia.org/wiki/File:The_application_%22Reef_Interactive%22_in_use_on_a_tablet..jpg Licensed under CC BY-SA 4.0: <https://creativecommons.org/licenses/by-sa/4.0/legalcode>.
- [3] Virpa a (accessed 12.09.2018). URL: <http://virpa.fi/en/projects/virpa-a-en/>.
- [4] Finpro and tekes uniting as business finland - business finland (accessed 12.09.2018). URL: <https://www.businessfinland.fi/en/whats-new/news/2017/Finpro-and-Tekes-uniting-as-Business-Finland/>.
- [5] Hand C. (1997) A survey of 3d interaction techniques. In: Computer graphics forum, vol. 16, Wiley Online Library, vol. 16, pp. 269–281.
- [6] Bowman D.A., Kruijff E., LaViola Jr J.J. & Poupyrev I. (2001) An introduction to 3-d user interface design. Presence: Teleoperators and virtual environments 10, pp. 96–108.
- [7] Jankowski J. & Hachet M. (2013) A survey of interaction techniques for interactive 3d environments. In: Eurographics 2013-STAR.
- [8] Snoonian D. (2003) Smart buildings. IEEE spectrum 40, pp. 18–23.
- [9] Weng T. & Agarwal Y. (2012) From buildings to smart buildings—sensing and actuation to improve energy efficiency. IEEE Design & Test of Computers 29, pp. 36–44.
- [10] Frequently asked questions about the national bim standard-united states™ | national bim standard - united states (accessed 27.04.2019). URL: <https://www.nationalbimstandard.org/faqs>.
- [11] Eastman C., Teicholz P., Sacks R. & Liston K. (2011) BIM handbook: A guide to building information modeling for owners, managers, designers, engineers and contractors. John Wiley & Sons.
- [12] Yan W., Culp C. & Graf R. (2011) Integrating bim and gaming for real-time interactive architectural visualization. Automation in Construction 20, pp. 446–458.
- [13] Hagedorn B. & Döllner J. (2007) High-level web service for 3d building information visualization and analysis. In: Proceedings of the 15th annual ACM international symposium on Advances in geographic information systems, ACM, p. 8.

- [14] Kashiwakuma J., Kitahara I., Kameda Y. & Ohta Y. (2013) A virtual camera controlling method using multi-touch gestures for capturing free-viewpoint video. In: Proceedings of the 11th european conference on Interactive TV and video, ACM, pp. 67–74.
- [15] Christie M., Olivier P. & Normand J.M. (2008) Camera control in computer graphics. In: Computer Graphics Forum, vol. 27, Wiley Online Library, vol. 27, pp. 2197–2218.
- [16] Phillips C.B. & Badler N.I. (1988) Jack: A toolkit for manipulating articulated figures. In: Proceedings of the 1st annual ACM SIGGRAPH symposium on User Interface Software, ACM, pp. 221–229.
- [17] Zhai S. & Milgram P. (1998) Quantifying coordination in multiple dof movement and its application to evaluating 6 dof input devices. In: Proceedings of the SIGCHI conference on Human factors in computing systems, ACM Press/Addison-Wesley Publishing Co., pp. 320–327.
- [18] Froehlich B., Hochstrate J., Skuk V. & Huckauf A. (2006) The globefish and the globemouse: two new six degree of freedom input devices for graphics applications. In: Proceedings of the SIGCHI conference on Human Factors in computing systems, ACM, pp. 191–199.
- [19] Balakrishnan R., Baudel T., Kurtenbach G. & Fitzmaurice G. (1997) The rockin'mouse: integral 3d manipulation on a plane. In: Proceedings of the ACM SIGCHI Conference on Human factors in computing systems, ACM, pp. 311–318.
- [20] Fröhlich B. & Plate J. (2000) The cubic mouse: a new device for three-dimensional input. In: Proceedings of the SIGCHI conference on Human Factors in Computing Systems, ACM, pp. 526–531.
- [21] Cad mouse / dcc mouse / 3d mouse / spacecontroller. catia, google earth, solid edge, solidworks, autodesk, 3d studio max, ... 3d-maus - spacecontroller | 3d-mouse (accessed 16.03.2017). URL: <http://www.spacecontrol.us/spacecontrol-3d-maus-spacecontroller.html>.
- [22] 3dconnexion : Spacemouse® (accessed 16.03.2017). URL: <http://www.3dconnexion.eu/products/spacemouse.html>.
- [23] Jrvz & Auawise (2010). Obtained from: https://commons.wikimedia.org/wiki/File:Yaw_Axis_Corrected.svg Licensed under CC BY-SA 3.0: <https://creativecommons.org/licenses/by-sa/3.0/legalcode>.
- [24] Lee S., Buxton W. & Smith K. (1985) A multi-touch three dimensional touch-sensitive tablet. In: Acm Sigchi Bulletin, vol. 16, ACM, vol. 16, pp. 21–25.
- [25] Edelmann J., Schilling A. & Fleck S. (2009) The dabr-a multitouch system for intuitive 3d scene navigation. In: 3DTV Conference: The True Vision-Capture, Transmission and Display of 3D Video, 2009, IEEE, pp. 1–4.

- [26] Mendes D., Sousa M., Ferreira A. & Jorge J. (2014) Thumbcam: Returning to single touch interactions to explore 3d virtual environments. In: Proceedings of the Ninth ACM International Conference on Interactive Tabletops and Surfaces, ACM, pp. 403–408.
- [27] Doc:2.4/manual/3d_interaction/navigating - blenderwiki (accessed 20.03.2017). URL: https://wiki.blender.org/index.php/Doc:2.4/Manual/3D_interaction/Navigating.
- [28] Zooming, panning, and orbiting views | 3ds max | autodesk knowledge network (accessed 20.03.2017). URL: <https://knowledge.autodesk.com/support/3ds-max/learn-explore/caas/CloudHelp/cloudhelp/2017/ENU/3DSMax/files/GUID-D65DE5FD-F859-4F66-9E14-F9A5C1016411-htm.html>.
- [29] Ware C. & Osborne S. (1990) Exploration and virtual camera control in virtual three dimensional environments. In: ACM SIGGRAPH Computer Graphics, vol. 24, ACM, vol. 24, pp. 175–183.
- [30] Gkikas K., Nathanael D. & Marmaras N. (2007) The evolution of fps games controllers: how use progressively shaped their present design. In: Panhellenic Conference on Informatics (PCI).
- [31] Houde S. (1992) Iterative design of an interface for easy 3-d direct manipulation. In: Proceedings of the SIGCHI conference on Human factors in computing systems, ACM, pp. 135–142.
- [32] Bade R., Ritter F. & Preim B. (2005) Usability comparison of mouse-based interaction techniques for predictable 3d rotation. In: International Symposium on Smart Graphics, Springer, pp. 138–150.
- [33] Hinckley K., Tullio J., Pausch R., Proffitt D. & Kassell N. (1997) Usability analysis of 3d rotation techniques. In: Proceedings of the 10th annual ACM symposium on User interface software and technology, ACM, pp. 1–10.
- [34] Mackinlay J.D., Card S.K. & Robertson G.G. (1990) Rapid controlled movement through a virtual 3d workspace. In: ACM SIGGRAPH computer graphics, vol. 24, ACM, vol. 24, pp. 171–176.
- [35] Moerman C., Marchal D. & Grisoni L. (2012) Drag'n go: Simple and fast navigation in virtual environment. In: 3D User Interfaces (3DUI), 2012 IEEE Symposium on, IEEE, pp. 15–18.
- [36] Hachet M., Decle F., Knodel S. & Guitton P. (2008) Navidget for easy 3d camera positioning from 2d inputs. In: 3D User Interfaces, 2008. 3DUI 2008. IEEE Symposium on, Ieee, pp. 83–89.
- [37] Bowman D.A., Koller D. & Hodges L.F. (1998) A methodology for the evaluation of travel techniques for immersive virtual environments. Virtual reality 3, pp. 120–131.

- [38] Jankowski J., Hulin T. & Hachet M. (2014) A study of street-level navigation techniques in 3d digital cities on mobile touch devices. In: 3D User Interfaces (3DUI), 2014 IEEE Symposium on, IEEE, pp. 35–38.
- [39] Khan A., Komalo B., Stam J., Fitzmaurice G. & Kurtenbach G. (2005) Hovercam: interactive 3d navigation for proximal object inspection. In: Proceedings of the 2005 symposium on Interactive 3D graphics and games, ACM, pp. 73–80.
- [40] Declec F., Hachety M. & Guitton P. (2009) Tech-note: Scruticam: Camera manipulation technique for 3d objects inspection. In: 3D User Interfaces, 2009. 3DUI 2009. IEEE Symposium on, IEEE, pp. 19–22.
- [41] Ortega M., Stuerzlinger W. & Scheurich D. (2015) Shocam: A 3d orbiting algorithm. In: Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology, ACM, pp. 119–128.
- [42] Declé F. & Hachet M. (2009) A study of direct versus planned 3d camera manipulation on touch-based mobile phones. In: Proceedings of the 11th International Conference on Human-Computer Interaction with Mobile Devices and Services, ACM, p. 32.
- [43] Hagedorn B. & Döllner J. (2008) Sketch-based navigation in 3d virtual environments. In: International Symposium on Smart Graphics, Springer, pp. 239–246.
- [44] Unity - game engine (accessed 20.03.2017). URL: <https://unity3d.com/>.
- [45] Shen Z., Jiang L., Grosskopf K. & Berryman C. (2012) Creating 3d web-based game environment using bim models for virtual on-site visiting of building hvac systems. In: Construction Research Congress 2012: Construction Challenges in a Flat World, pp. 1212–1221.
- [46] Fbx sdk help: What is autodesk fbx technology (accessed 20.03.2017). URL: http://download.autodesk.com/us/fbx/2010/fbx_sdk_help/index.html?url=WS1a9193826455f5ff-150b16da11960d83164-6c6f.htm,topicNumber=d0e127.
- [47] Collada - 3d asset exchange schema (accessed 21.03.2017). URL: <https://www.khronos.org/collada/>.
- [48] Gane oy, turva, sähkö, automaatio (accessed 07.04.2019). URL: <https://www.gane.fi/>.
- [49] Bowman D., Kruijff E., LaViola Jr J.J. & Poupyrev I.P. (2004) 3D User Interfaces: Theory and Practice, CourseSmart eTextbook. Addison-Wesley.
- [50] Chin J.P., Diehl V.A. & Norman K.L. (1988) Development of an instrument measuring user satisfaction of the human-computer interface. In: Proceedings of the SIGCHI conference on Human factors in computing systems, ACM, pp. 213–218.