Institute of Computer Graphics

Technische Universität Braunschweig

Master's Thesis

Visualization of scientific data in multi-user augmented reality

Jan Wulkop

February 9, 2022

Institute of Computer Graphics
Prof. Dr.-Ing. Martin Eisemann

Supervisors:
Prof. Dr.-Ing. Martin Eisemann
Dr.-Ing. Georgia Albuquerque

Statement of Originality

This thesis has been performed independently with the support of my supervisor/s. To the best of the author's knowledge, this thesis contains no material previously published or written by another person except where due reference is made in the text.

Braunschweig, February 9, 2022

Abstract

Humanity has always strived to learn more about the origins of our neighboring celestial bodies. With the help of modern rover systems, unknown areas are explored through scientific measurements. With increasingly better sensors, this data becomes more extensive and complex, creating an evident need for new and improved tools. These tools should support the scientists in the collaborative analysis of the recorded measurements. Scientists from different disciplinary backgrounds work together on this analysis. Exploring the data can be made more efficient with the help of intuitive visualization, interaction, and collaborative tools. At the same time, misunderstandings among the experts can be minimized.

This thesis investigates how modern augmented reality approaches can support the process of collaborative rover data analysis. Three main aspects are considered: the three-dimensional visualization of high-resolution terrain data, the visualization and interaction with rover data, and the integration of multi-user collaboration tools for the collaborative discussion. A mobile augmented reality device, the Microsft HoloLens 2, is used to input, output, and process the data.

In order to evaluate the implemented visualization and interaction concepts, an expert interview and several experiments for a user study are prepared in this work. Due to the current COVID-19 pandemic restrictions, both interview and user study could not be conducted. Based on promising informal preliminary user tests, potential improvements of the presented concepts are discussed.

Kurzfassung

Seit jeher strebt die Menschheit danach, die Entstehung unserer benachbarten Himmelskörper zu untersuchen. Mithilfe von modernen Roversystemen werden unbekannte Gebiete durch wissenschaftliche Messungen erforscht. Durch zunehmend bessere Messsysteme werden diese Daten immer umfangreicher und komplexer, wodurch neue, verbesserte Analysewerkzeuge nötig werden. Wissenschaftler mit unterschiedlichen fachlichen Hintergründen analysieren gemeinsam die Daten. Mithilfe intuitiver Visualisierung, Interaktion und auch kollaborativer Werkzeuge zur Erkundung dieser Daten, kann der Prozess effizienter gestaltet werden. Gleichzeitig können Missverständnisse unter den Experten vermindert werden.
Diese Arbeit untersucht, inwiefern moderne Augmented Reality den Prozess der kollaborativen Rover-Daten Analyse unterstützen kann. Drei Schwerpunkte werden in der Arbeit näher betrachtet: die dreidimensionale Visualisierung von hochaufgelösten Terraindaten, die Darstellung und Interaktion mit den Rover-Daten, sowie die Integration von Kollaborationswerkzeugen für die gemeinsame Diskussion. Ein mobiles Augmented Reality Gerät, die Microsoft HoloLens 2, soll dabei sowohl die Ein- und Ausgabe, als auch die Verarbeitung der Daten übernehmen.
Um die implementierten Visualisierungs- und Interaktionskonzepte zu evaluieren, wird in dieser Arbeit ein Experteninterview, sowie mehrere Experimente für eine Nutzerstudie vorbereitet. Aufgrund der Einschränkungen durch die aktuelle COVID-19 Pandemie, konnten sowohl das Interview, als auch die Nutzerstudie nicht durchgeführt werden. Mögliche Verbesserungspotenziale der vorgestellten Konzepte werden aufbauend auf den Erkenntnissen vorläufiger, vielversprechender Nutzertests diskutiert.

Contents

List of Figures

Nomenclature

AR Augmented Reality

ARCHES Autonomous Robotic Networks to Help Modern Societies

CAVE Cave Automatic Virtual Environment

CPU Central Processing Unit

DEM Digital Elevation Model

DLR German Aerospace Center

GIS Geographic Information Systems

GPS Global Positioning System

GPU Graphics Processing Unit

HIT Horizontal Image Translation

HLSL High Level Shading Language

HMD Head-mounted display

IMU Inertial Measurement Unit

ISS International Space Station

JPEG Joint Photographic Experts Group

JSON Java Script Online Notation

LIBS Laser-Induced Breakdown Spectroscopy

LRU Lightweight Rover Unit

PNG Portable Network Graphics

QUESI Questionnaire for the subjective consequences of intuitive use

REST Representational State Transfer

SAGAT Situation Awareness Global Assessment Technique

SUS System Usability Scale

SVT Streaming Virtual Textures

TIFF	Tagged Image file Format
TLX	NASA TLX
TRS	Terrestrial Reference System
UAV	Unmanned Aerial Vehicle
UE	Unreal Engine
UI	User Interface
UTM	Universal Transverse Mercator
UWP	Universal Windows Platform
VR	Virtual Reality
WIM	World in miniature

1. Introduction

"Human beings face ever more complex and urgent problems, and their effectiveness in dealing with these problems is a matter that is critical to the stability and continued progress of society. A human is effective not just because he applies to a problem, a high degree of native intelligence or physical strength (with a full measure of motivation and purposefulness), but also because he makes use of efficient tools, methods, and strategies."

(Engelbart, 1962 [26])

Douglas Engelbart, one of the pioneers of the modern computer industry, envisioned the computer as a tool to augment the human intellect and its abilities. In his vision, Engelbart considered an information space where experts could access interactive *working stations* to solve problems together [25]. Increasingly complex challenges could be solved using the best tools assisted by computers augmentation and guiding.

Although Engelbart introduced his concepts 60 years ago, the vision of augmented humans and computers as tools is still relevant during the development of modern applications. Automation, interaction, visualization, and communication are more relevant than ever as key aspects of human-machine interaction to assist humans in their tasks. Data is becoming increasingly multidimensional and complex. This extensive data needs to be visually processed by modern tools to provide experts with an overview for critical data-based decisions and thus assist them in solving their challenges.

A major field of today's challenges is the exploration of space. In the field of aerospace, technology research and development are an international and distributed process. Engineers from different backgrounds and research areas are working together on interplanetary missions. In the future, missions to other celestial bodies will have to be planned. Scientific results have to be analyzed in order to gain detailed insight for further mission planning in an unknown environment. In future missions, rovers will explore and collect scientific data in these areas.

The autonomous exploration of unknown environments with a group of robotic systems is a core aspect of the ARCHES (Autonomous Robotic Networks to Help Modern Societies) project [130]. As part of this project, the German Aerospace Center (DLR) is developing robotic systems capable of conducting scientific measurements autonomously. Multispectral images, soil sample material spectroscopy, as well as three-dimensional reconstructions of collected samples and the environment surrounding the rover are the primary results.

The analysis process of these scientific measurements is a complex task and is still done by humans. Planet scientists have to evaluate these scientific data to get further insights into the investigated areas. Due to the different scientific backgrounds of the experts, misunderstandings can not be avoided. An open discussion among the experts helps to clarify this. To support a successful discussion between the experts, visualization of the collected data and measurements is crucial. A good visualization can help avoid misunderstandings between the experts and enrich the discussions.

During the analysis process, not only the collected data, but also the specific location on the terrain and the spatial relationship to other scientific activities are relevant. Experts might want to compare new measurements, acquired by the rover, to previous measurements in the vicinity. Furthermore, the topological background of a finding is significant. The chemical composition of a soil sample, collected in a valley, might differ from a sample collected on top of a mountain. Therefore the spatial relationship of a measurement to the landscape but also to previous recordings is an additional aspect in the analysis discussion that needs to be visualized.

Past studies showed that linking data to their precise position on a map can emphasize the spatial context in an analysis process [65, 134, 114, 141]. In this project, an augmented reality (AR) demonstrator is developed based on these findings to investigate how AR can support the analysis process using immersive three-dimensional visualization and interaction. Specifically, a *co-located* collaboration scenario is considered in which several experts are together in a room and analyze data recorded by the rover. Both the data collected by the rover and the surrounding landscape in the vicinity of the rover's deployment area will be available to the experts as interactive virtual holograms in a discussion. A user interface will enable the experts to jointly analyze the data and to spatially classify them based on a realistic landscape representation.

1.1. Objective

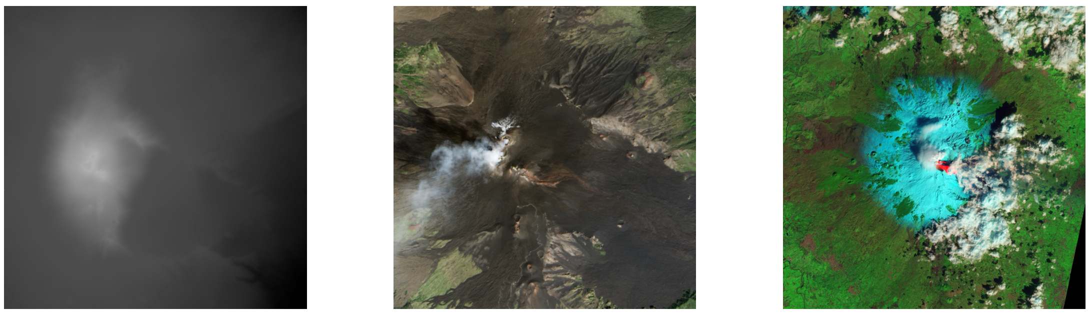
This work aims to examine whether AR can enrich the collaborative analysis of rover data. It will investigate if AR systems' benefits can be applied to make the data analysis more intuitive, time-saving, and user-friendly. This thesis deals with three core aspects: large-scale terrain rendering, rover data visualization, and multi-user collaboration.

Immersive AR has the potential to revolutionize the way virtual data is presented. With devices like Microsoft HoloLens, virtual content can be projected on a specific location into the real environment. Data can be inspected three-dimensionally as a hologram directly in front of the user. By walking in-between these virtual elements, users can look at holograms from different perspectives while still being able to perceive their real environment.

Related to the use case of a collaborative co-located analysis session, augmented reality offers many advantages that could further enrich the discussion among experts. The data to be discussed can be displayed immersively and three-dimensionally. Each expert can individually change his or her position to gain a different perspective on the data. In contrast to virtual reality approaches, the experts can perceive both the virtual content and the other experts in the discussion. Through the direct contact of the experts, non-verbal communication can be intuitively incorporated into the discussion.

These advantages of augmented reality should be used to optimize the discussion and the analysis process of scientific rover data. In the context of this work, the ARCHES project is specifically investigated. The DLR is involved in the development of two rovers and a drone. The results of the scientific measurements of these robotic systems should be presented to the experts in augmented reality.

These data can be classified into four categories: multispectral images, chemical material analysis of soil samples, three-dimensional reconstructions, and aerial images taken by the drone. In addition, digital elevation models, satellite imagery, and other scientific analyses describe the rover's area of operation (see figure 1.1). These data are also to be displayed in AR.


(a) Digital elevation model of the region around Mount Etna. Brighter values represents higher elevation.

(b) Landsat 7 satellite images of Mount Etna.

(c) Etna outburst Feb. 2013. Shortwave infrared, near-infrared, and green light combined in false color image.

Figure 1.1.: Terrain data to be visualized.

For this purpose, a software concept is to be developed, implemented, and finally evaluated, enabling a collaborative exploration, visualization, and interaction. The goal is to stimulate the discussion of the experts through a visualization, and interaction concept that is appropriate for the type of data.

In this visualization, the presentation of the measurements and their spatial context to the surrounding landscape is an important aspect. For this purpose, the mission-relevant area should be displayed as a realistic three-dimensional model. Image datasets such as satellite images should be dynamically shown on the landscape surface. Using this model should allow the experts to navigate even beyond the border of this area to explore the large-scale surroundings.

This model should serve as a central visual element for browsing the scientific activities of the rovers. Each location where the rover has conducted a measurement should be highlighted as an interactive marker on the landscape model. The aim is to provide a spatial context linking the rover data with the surrounding landscape.

Overall, three different focal points of this thesis can be identified. Firstly, large-scale high-resolution landscape data shall be presented on the limited computational capacity of a mobile AR device. Secondly, the data recorded by the rover should be presented to the experts in a meaningful format during the analysis. The third focus of this work is the collaborative exploration of these data. Tools to support collaborative analysis are to be implemented.

Subsequently, both the presented visualization and interaction concepts have to be evaluated. An expert interview should be conducted to qualitatively test whether the implemented approaches could help in the analysis process in a realistic scenario. A user study should qualitatively and quantitatively evaluate the user interfaces for rover and landscape data from a non-expert perspective. The usability and workload of the presented tools and visualizations shall be investigated.

1.2. Research Questions

The following research questions are discussed in this thesis:

- **RQ1:** How can large-scale landscapes be rendered on the limited computing capabilities of mobile AR devices? (section 5.4.1)
- **RQ2:** How can the visualization of the rover data together with the surrounding landscape in immersive AR improve the analysis process? (chapter 6)
- **RQ3:** How can the rover data be presented in a meaningful way through 3D visualization? (section 5.4.3 chapter 6)
- **RQ4:** Which 3D interaction metaphors are beneficial for browsing rover data? (chapter 6)
- **RQ5:** Which collaboration tools can be used for discussing results in a co-located scenario? (section 5.4.5 and chapter 6)

1.3. Overview

This thesis is structured in seven chapters. This first chapter presents the problem, motivation and research questions. Subsequently, in Chapter 2 a necessary knowledge about the used concepts and hardware and software components is given. In chapter 3, related work is presented and discussed with regards to the use cases considered in this thesis. Subsequently, in Chapter 4, the individual challenges of the tasks are differentiated from each other. Based on the related work, the feasibility of the different approaches is discussed in this chapter. The implementation of these approaches are demonstrated in Chapter 5. The following Chapter 6 evaluates the implemented prototype. Here the planned expert interview as well as a user study is presented. Additionally the approach for the qualitative and quantitative evaluation is explained. The thesis concludes with a summary and discussion about limitations of the current implementation and future improvements in Chapter 7.

2. Basics

The following chapter presents the fundamental knowledge about the concepts and hardware and software components used in this project.

2.1. Augmented reality

In the past, many scientists attempted to define and separate the term *Augmented Reality* from comparable terminologies, such as *Virtual Reality* and *Mixed Reality*. One of the first and most widely accepted definitions dates back to the work of Milgram et al. in the year 1995. To distinguish between virtual and real content, this work defines the *virtuality continuum* [110], which is a continuum between real and virtual environments. Figure 2.1 depicts a diagram of this continuum.

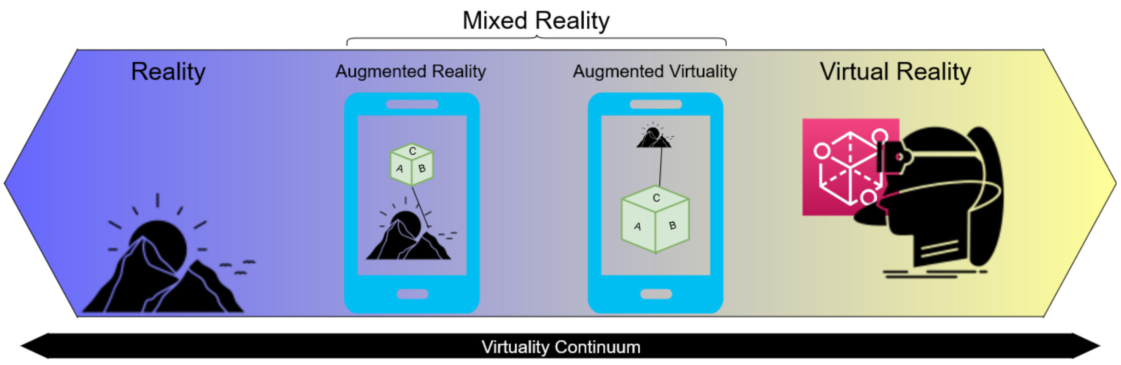


Figure 2.1.: Virtuality Continuum according to Milgram et al.

There are two extremes in this continuum: On the left side are scenarios where the user sees and interacts with real-world objects only. The environment is depicted exactly as it is. On the right side of the spectrum is the concept of virtual reality. Everything a user sees, feels and hears in this environment is purely virtual and does not exist in reality. The user's entire perception is artificially computer generated. The transition between these two extremes is guided by augmented reality (AR) and augmented virtuality.

According to Milgram et al., augmented reality uses real-world objects and annotates them with virtual content. The user can still perceive the real environment, but additional virtual information about these objects is displayed [110]. Augmented virtuality is characterized by the fact that the virtual environment is clearly in focus, but it is enhanced by individual real components.

The transition between augmented reality and augmented virtuality is seamless. According to Milgram et al., this transition is called *Mixed Reality*. Although Milgram et al.

focuses on the perception of visible virtual content, the ideas of the virtuality continuum can also be applied to the other human senses, including aural, tactile, and vestibular senses.

Azuma et al. published an expanded definition for augmented reality in 2001, based on the work of Milgram et al. [9]. According to Azuma et al., the purpose of augmented reality applications is to support real-world tasks through the targeted integration of virtual content into the real world. They derive three different criteria for classifying an application as an augmented reality application:

1. Real and virtual objects are combined in a real environment.
2. The application is interactive and runs in real-time.
3. Virtual objects are registered in the real environment and are displayed with real objects in spatial relation to each other.

Azuma et al. do not limit these criteria only to the visual display or a specific output device. These criteria apply regardless of the output method or specific perceptual senses. Various types of devices can be used to perceive augmented reality. One of these devices, the HoloLens, is introduced in the following section.

2.2. HoloLens 2

The HoloLens is a wireless mobile computer developed by Microsoft. HoloLens 1, the first generation of this device, was released in 2016 [85]. This work uses the successor generation, the HoloLens 2, made available in November 2019 [105]. In the following, the HoloLens of the second generation will just be called HoloLens.

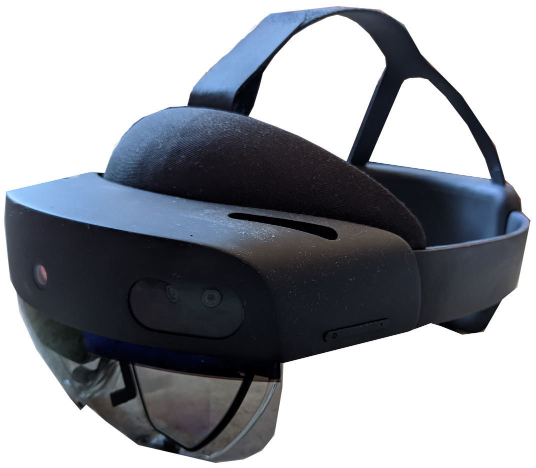


Figure 2.2.: Microsoft HoloLens 2

The HoloLens, as shown in Figure 2.2, is unlike any other mobile computer in terms of both appearance and functionality. As a mobile mixed reality device, the HoloLens combines computer and integrated input and output device. Virtual content is displayed on two transparent screens in front of the user's eyes. This content is controlled by a graphical application, running on the integrated computer. The image content can differ on the two screens. This enables applications to trick the user into perceiving three-dimensional virtual content by precisely shifting the perspective of the two displayed images. This virtual content can augment the wearer's real-world surroundings [97].

In contrast to *digital see-through* AR head-mounted displays (HMDs), in which the real environment is captured by a camera and later blended with the virtual content, HoloLens augments virtual content and the real environment in the wearer's eye. Since the user perceives the real environment directly through the transparent glasses, the HoloLens is considered an *optical see-through* augmented reality HMD [126].

HoloLens enables applications to place virtual content in the real environment on a fixed location. The user can move freely around the virtual object and inspect it from all perspectives [97]. The stability of these fixed holograms depends on the accuracy of the devices' position and orientation tracking. The HoloLens uses a combination of sensors to determine the user's movement in real-time.

2.2.1. Hardware

The hardware components of the HoloLens are described below. First, the typical computer hardware specifications of the HoloLens are characterized. Then, the integrated sensors, the available human-machine and communication interfaces of the device are explained.

2.2.1.1. Data processing on HoloLens

As a mobile AR headset, the HoloLens is powered by an integrated computer. All the information processing and rendering for AR applications is executed on that computer. The device is equipped with a central processing unit (CPU), an integrated graphics chip, main memory, and persistent flash storage.

A Qualcomm Snapdragon 850 compute module serves as the 64-bit processor [87]. Eight individual cores with a base frequency of 2.96 GHz are used for data processing [121]. The CPU is supported by a co-processor developed by Microsoft specifically for the HoloLens, the *Holographic Processing Unit* [87]. Among other things, this unit processes the sensor and camera streams and thus relieves the CPU.

Furthermore, the device is equipped with 4 GB of main memory. On average, 2 GB are reserved for the operating system, and only 2 GB of memory is available for an application [87]. If an application allocates more than 2 GB at any time, the process is automatically terminated.

The hardware access of applications is managed by the *Windows Holographic* operating system, developed specifically for the HoloLens by Microsoft. This operating system is based on Windows 10 but is adapted for the processing of AR applications. The integrated 64 GB flash memory stores the operating system and the installed applications [87].

As an AR input and output device, graphical applications can visualize virtual content through the HoloLens. The graphics calculations of these applications are conducted on the *Graphical Processing Unit* (GPU) of the Qualcomm compute module. This module includes a Qualcomm Adreno 630 graphics chip without dedicated graphics memory [121]. This means that the CPU and the graphics chip share the 2 GB of memory available to the application. The graphics chip supports the DirectX 12, Vulkan 1.1 and OpenGL ES 3.2 graphics interfaces [121].

2.2.1.2. Sensors

A core aspect of augmented and mixed reality is the interaction of virtual content with the real world. The HoloLens makes it possible to place virtual objects in a fixed location in the physical world. The wearer can then move freely around this content to view it from all angles. This requires that the glasses determine the users' position and orientation in three-dimensional space in real-time.

The HoloLens uses a combination of different sensors and cameras to detect the environment and the exact position and orientation of the glasses in it. The *Inertial Measurement Unit* (IMU) measures the current attitude and acceleration. Tilt and acceleration of the user's head are derived from these sensor values. In addition to the IMU, HoloLens is equipped with four cameras and a 1-megapixel time-of-flight depth sensor [87]. The four color cameras capture distinctive high-contrast feature points in the user's environment, trying to recognize these points as often as possible. HoloLens estimates the user's movement and orientation by tracking these feature points over time.

The built-in depth sensor operates in two modes. In the first mode, the sensor detects the immediate environment with a refresh rate of 45 Hertz at a distance of up to one meter. This mode is used to locate fingers and hands with a high frequency. The second mode detects the broader environment of the user with a frequency of 1-5 Hertz. A three-dimensional environment reconstruction is created and continuously updated using this depth information [95]. This data stream is used to recognize prominent features in the reconstruction and thus estimate the user's movement.

Two infrared cameras, pointing at the user's eyes, allow the direction of the gaze to be determined with an accuracy of up to 1.5 degrees. With the help of an additional front-facing color camera, 8-megapixel photos and full HD videos can be recorded from the user's point of view within applications [87].

2.2.2. Interaction models

Four different input methods are available to interact with the virtual content displayed on the HoloLens. These are gaze direction control, voice control, finger tracking and eye-tracking [87]. The sensors of the device continuously detect the user's surroundings. This facilitates an exact determination of where the user is in relation to his surroundings and in which direction he has turned his head. This head direction can be used as an input method to point to a specific virtual object. With the development of the HoloLens 2, an eye-tracking sensor was added to the headset. This sensor detects what the user's pupils focus on in the virtual scene. With the integrated depth sensor, the HoloLens can detect the position and orientation of individual fingers when the hand is in the sensor's detection range.

Microsoft derives five distinct interaction models for selecting virtual objects from the available input interfaces [96]:

1. **Direct hand manipulation:** Direct hand manipulation is a primary input method. Using the built-in hand tracking system, the articulated hands of the user can be tracked. Microsoft's principle of direct hand manipulation is based on the premise that the user can touch and manipulate the virtual holograms directly with his or her own hands. These virtual objects behave physically like objects from the real world, if they were touched by the user. Virtual objects can be picked up by grasping them, buttons can be activated by pressing them. Figure 2.3 shows a user interacting with a hologram using direct hand manipulation. How a hologram interacts with the articulated hand can be defined in the application. [90].

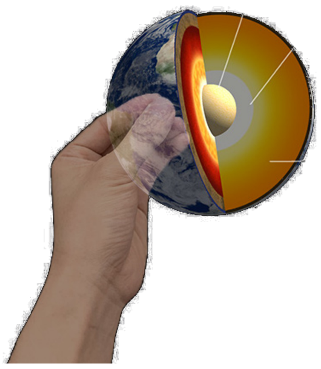
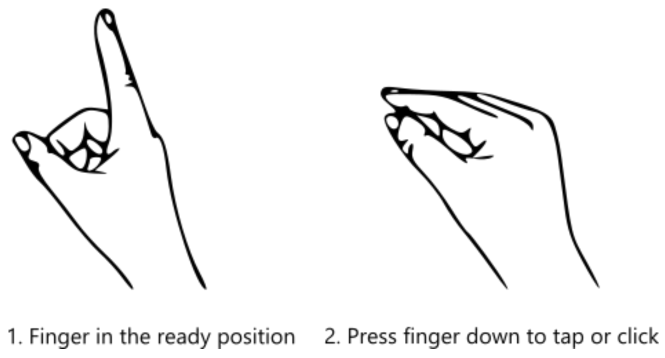


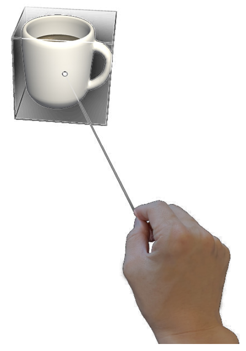
Figure 2.3.: User interacting with virtual content using direct hand manipulation [90].

2. **Point and commit with hands:** The point and commit interaction is commonly used to select virtual content outside the user's reach. This mechanism is divided into the pointing phase and the commit phase. By default, the user is in the pointing phase.

   During this phase, the articulated hand is used as a pointer by projecting a beam from the palm towards the tip of the index finger. If this beam collides with a virtual object, a cursor appears on the intersection point. By making a gesture, the *Air Tap*, the user switches to the commit phase. This gesture is shown in figure 2.4a. It is formed by pressing the index finger and the tip of the thumb together. Each time the user switches between the pointing and commit phase, this change can be processed and the targeted object is selected. [98]. Figure 2.4b shows how a user selects a far object using this point and commit method.


1. Finger in the ready position
2. Press finger down to tap or click

(a) Using the Air Tap gesture



(b) Selecting an object with point and commit

Figure 2.4.: Point and commit with hands.

3. **Voice Input:** Voice commands can be used to execute actions defined in the application. This input method makes it possible to control the flow of the application without using hands. These can be working on other activities at the same time. In noisy environments, speech recognition cannot be used reliably [107].
4. **Gaze and dwell:** The gaze and dwell interaction method is another hands-free interaction. This input model can be considered as a fallback solution when voice commands are not feasible due to the user's environment. Either the head direction (head-gaze and dwell) or the pupil direction (eye-gaze and dwell) controls the cursor. To select an object, the cursor must stay on the object for a predefined time [92].
5. **Gaze and commit:** The principle of the gaze and commit input model is derived from the point and click input model with the mouse on a regular computer. Either the head direction (head-gaze and commit) or the pupil direction (eye-gaze and commit) controls the cursor. If the cursor hits an object, it can be selected by pressing the Air Tap gesture (see figure 2.4a). This input method is similar to the point and commit method with the difference that the cursor is not controlled by hands, but either with the head movement or the gaze direction of the eyes [91].

2.2.3. Application development

HoloLens-compatible applications can be developed in various ways. Microsoft explains two approaches in more detail [89]:

On the one hand, mixed reality applications can be developed from scratch based on the open-source *OpenXR* specification. OpenXR acts as an abstraction layer between mixed reality input/output devices and applications that communicate with this hardware [73]. The standard enables interoperability between a mixed reality application and mixed reality devices from different vendors. Typical data, such as the user's position and viewing direction, control inputs or the stereo image to be displayed, are exchanged between hardware and software via this unified interface [73].

On the other hand, Microsoft recommends the use of *game engines*. Such an engine provides developers with various tools for developing interactive applications. Essential functions of the image rendering and physics simulation, but also the communication of applications between each other, are abstracted and already implemented by the engine [37]. These pre-built fundamental blocks allow developers to fully concentrate on the implementation logic without re-implementing these basic concepts on a hardware level.

As can be noticed from the term *game engine*, the primary use case for these engines is the development of interactive three-dimensional video games. However, these engines are increasingly used for video production, medicine, and research applications [1, 48]. Through continuous development and integration of AR software components and hardware support, these engines are used as a basis for research in mixed reality, human-machine interaction, telepresence, and collaboration.

While only the *Unity 3D* game engine is supported for HoloLens 1, software developers can also use the *Unreal Engine 4* game engine for HoloLens 2 applications [102].

2.3. Unreal Engine

Unreal Engine (UE) is a cross-platform game engine released in its first version in 1998 [36]. Such an engine provides software developers with pre-built tools for developing interactive applications. Essential components for image rendering, physics simulation, user interface and network communication are provided by Unreal Engine. The development of interactive two- and three-dimensional applications are supported.

Game engines are generally considered an abstraction layer between application logic and hardware-specific code. This allows the development of applications for different hardware configurations. The application is programmed against the interfaces of the engine. From the engine, the application can be ported to run on different target platforms.

The current stable version Unreal Engine 4.27 supports the export and execution of applications on over 20 target platforms [49]. Since the release of Unreal Engine 4.25, HoloLens 2 has been one of these supported platforms [103].

2.3.1. Application structure

The Unreal Engine is a collection of tools for developing interactive applications.

On the one hand, Unreal Engine can be considered a software library. Developers access the interfaces of this library in their application to use the pre-implemented functionality. On the other hand, Unreal Engine can be considered as a complete development environment. This environment includes its own programming language and a graphical editor [41, 38]. Basic project properties and the general structure of an application are defined via this integrated editor. The developers can preview and debug the application directly within it. Changes to the program code are thus visible and can be tested with short iteration times.

Fundamentally, any application built on Unreal Engine is based on the object-oriented approach to software development. Individual components of an application are divided into entities according to their task.

An application consists of three clearly differentiable entities: *levels*, *actors* and *components*. A level describes a virtual scene. This scene is built up from several hierarchically structured actors [42]. Consequently, an actor is a virtual object in a scene. Every actor has basic properties, like position, rotation and scaling, which describe how this actor is located in the level. Every object visible in the application, such as a three-dimensional geometry, is bound to an actor. They can also be invisible but influence other actors, such as light sources. The behavior of an actor is controlled by its components [35].

The integrated editor is used to configure the levels by placing and configuring the hierarchy of actors. The application logic is controlled by writing custom actor and component classes in the C++ programming language or the graphical scripting language called *Blueprints* [40].

2.3.2. Rendering

The term rendering generally refers to image synthesis of a virtual scene. The process of synthesizing such an image is called the *render pipeline*. The following section will explain the basic ideas and core components of this pipeline.

2.3.2.1. Camera

The image synthesis process is derived from the idea of taking a photo in the real world. In a photograph, an image of the real three-dimensional environment is captured on a two-dimensional medium. For this purpose, the photographer uses a camera that projects the 3D content onto a 2D sensor. The content of a photo depends on the scenery in front of the camera, lighting conditions, but also on camera-specific parameters.

These parameters can be divided into two different categories: extrinsic and intrinsic camera parameters [149]. The *extrinsic parameters* describe the camera's perspective on the scene. The position and orientation of the camera relative to the environment affect which objects are visible on the image.

The *intrinsic camera parameters* describe the projection of 3D coordinates onto the corresponding 2D coordinates on the sensor. This projection is influenced by many camera-

and lens-specific factors in real cameras. These factors include the lens's focal length, distortion properties of the whole optical apparatus, and the center of the sensor [149].

If intrinsic and extrinsic camera parameters are known, it can be determined precisely on which 2D coordinate on the camera sensor a 3D point is projected. Depending on these parameters, the viewing frustum can be calculated. This frustum defines the visible volume in the scene that can be captured with the current camera setting.

In game engines, the metaphor of the camera is used for image synthesis. The camera is the imaginary component that records the virtual three-dimensional scene and then projects it into a two-dimensional image from its perspective. Similar to a real camera, extrinsic and intrinsic parameters describe the projection properties of a virtual camera. Unreal Engine, however, uses a simpler camera model than the real cameras by default. The *pinhole camera model* is used for projection. This model ignores lens distortion effects that occur in a real camera and uses an idealized pinhole camera with an infinitely small hole [142]. Figure 2.5 shows a cross-section of such a pinhole camera.

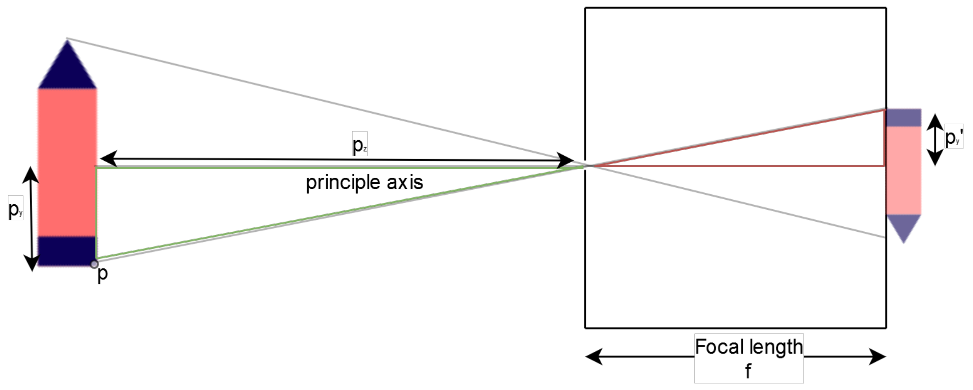


Figure 2.5.: Two-dimensional pinhole camera model.

Due to the infinitely small hole, a light beam reflected from a three-dimensional point on an object's surface can only be projected onto exactly one two-dimensional point on the sensor. Geometrically, this results in a dependence of the two color-coded triangles. This dependence is described in the following equation, also called the *intercept theorem* [147, p. 9].

$$\frac{p'_y}{f} = \frac{p_y}{p_z} \implies p'_y = \frac{f * p_y}{p_z} \tag{2.1}$$

As can be seen in equation 2.1, the projected coordinate on the sensor can be calculated if the focal length and the center coordinate of the optical axis are known. These intrinsic camera parameters can be set for the virtual camera in Unreal Engine with the *projection matrix*.

Depending on these parameters, the frustum of the virtual camera also changes. Figure 2.6 shows an example of the frustum of a virtual camera in Unreal Engine represented by the four pink stripes. Surface points, which are located within this volume and are not occluded, are visible on the rendered image.

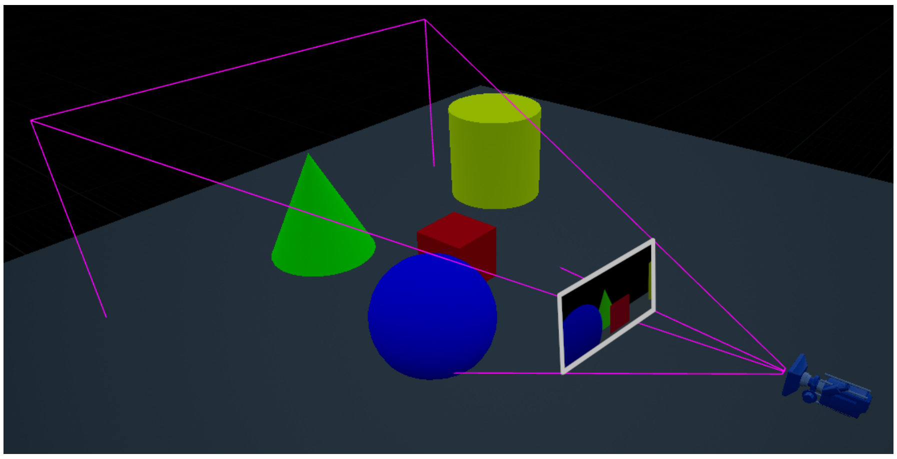


Figure 2.6.: Frustum of a virtual camera, bounded by pink lines.

2.3.2.2. Default Render Pipeline

A *shader* is a program that is executable on the particular hardware architecture of a graphics chip. Shaders are used to accelerate the image synthesis process.

For this purpose, the concepts of perspective projection described above are defined programmatically in shader programs. During rendering, shaders process the complex data of a virtual environment so that the scene's geometries are projected and colored on a two-dimensional image. The result of this calculation is displayed as pixels on an output device or stored in an image.

Many computational operations in the field of 3D image syntheses, such as projection or lighting calculation, can benefit from the highly efficient architecture of a graphics chip. This chip is composed of many *shader units*. With each having multiple processing cores, parallelizable tasks can be executed efficiently by distributing the computing load. Shader programs on modern graphics chip architectures can control which mathematical operation is calculated in these shader units.

Shader programs are defined in Unreal Engine either in the editor using a graphical scripting language or in the *High Level Shading Language* (HLSL) [39]. The shader source files are compiled automatically by Unreal Engine for different graphics programming interfaces depending on the selected target platform of the application.

The following section explains the individual logical stages of a shader program relevant for this work in more detail. The chronological sequence of the presented levels is called *render pipeline* (see figure 2.7).

Vertex Shader The vertex shader stage is called once for each vertex [106]. It is responsible for processing and projecting vertices of the geometries loaded in the scene. This stage executes per-vertex operations and projects the vertex coordinates into normalized device coordinates, using the projection matrix and the camera transformation. In each pass, exactly one vertex is processed, and then the projected two-dimensional coordinates are passed along the render pipeline together with other arguments [106].

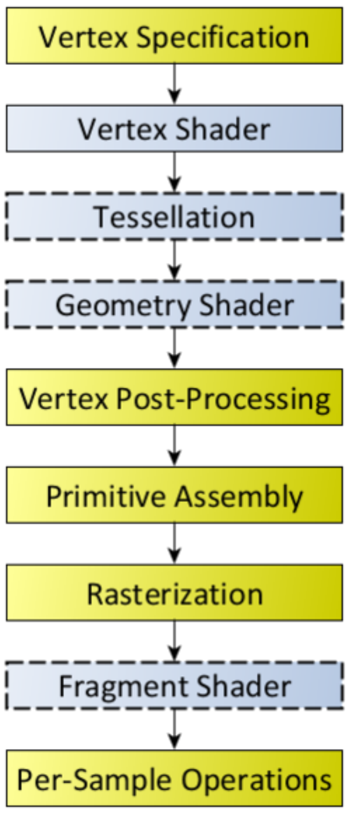


Figure 2.7.: Flow diagram of the render pipeline. The blue boxes are programmable shader stages [74].

Tessellation Shader The tessellation shader stage is optional in the render pipeline. This shader stage consists of three substages: the Hull Shader stage, a fixed-function tessellator stage and a Domain Shader stage [101]. These substages work together to subdivide a geometrically simple mesh into a finer-grained mesh. This subdivision is controlled by a mathematical function and takes place in real-time. To achieve higher geometry detail, primitives of low-resolution geometries are subdivided into several primitives. Tessellation allows for rendering high-resolution surfaces despite the corresponding geometry being of significantly lower resolution. As a result, the geometry size in main memory and the bandwidth for transferring geometry information to the graphics card can be optimized. By dividing the geometry in real-time, the application can dynamically control a geometries level of detail. [101].

Geometry Shader In the geometry shader stage, complete geometric primitives are processed. These primitives can be triangles, lines or single vertices. Like the tessellation stage, the geometry stage is optional in the render pipeline [93]. This shader stage receives the vertex information of an entire primitive, along with its identification number. This information can be used to perform per primitive operations.

If no geometry shader is defined in the render pipeline, the primitives are passed unchanged to the next stage of the render pipeline. In the geometry shader, vertex parameters of existing primitives can be overwritten, primitives can be discarded, or new primitives created to a limited extent [93].

Fragment Shader The fragment shader stage is executed after a process called *rasterization*. The vertices projected into the image plane of the camera form a composed surface. This surface is subdivided during rasterization into individual discrete pixel sized elements. These elements are called *fragments* [30]. The process of rasterization is shown in figure 2.8. The red boxes are the generated fragments. For each pixel covered by a primitive, a new fragment is generated and passed to this shader stage.

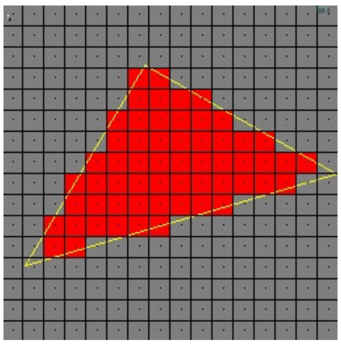


Figure 2.8.: Process of rasterizing a triangle into single fragments (red) [116].

In the fragment shader stage, the fragment's color and depth are determined based on the parameters calculated in the previous stages. [72]. The fragment shader is called for each fragment. This stage is typically used to read values from textures, such as material properties like surface color or roughness. These properties and other parameters about the scene's illumination are used to calculate the color of the surface point covered by the fragment [72].

2.3.2.3. Mesh geometry

Unreal Engine offers different components for the representation of geometries. Usually, a 3D model is added to a scene either as a *Static Mesh Component* or as a *Procedural Mesh Component*. According to its name, a Static Mesh Component is characterized by the fact that the basic properties of the geometry, like vertex buffer, index buffer and UV layout, are already known at compile time and do not change during the runtime of the application [45].

Procedural Mesh Components are used when the topology of the geometry changes while executing the application. This happens when vertices are moved, added or deleted [51].

2.4. Geospatial coordinate systems

The visualization of georeferenced information is a central aspect of this work. There are many different approaches to describe a location on a celestial body like our Earth. Most people think of geocoordinates as coordinates in latitude and longitude format. But these latitude and longitude data are not useful without further parameters, which describe the geocoordinate system. Latitude and longitude coordinates are always relative to a particular geocoordinate system, which has to be known, to process the position information. There are numerous coordinate systems in use worldwide, each with a slightly different orientation and origin. As a result, the same coordinates in different coordinate systems points to distinct real-world locations. Crossley discusses this aspect in his work and illustrates in figure 2.9 how the same coordinates can be located with different coordinate systems. [22, p. 6].

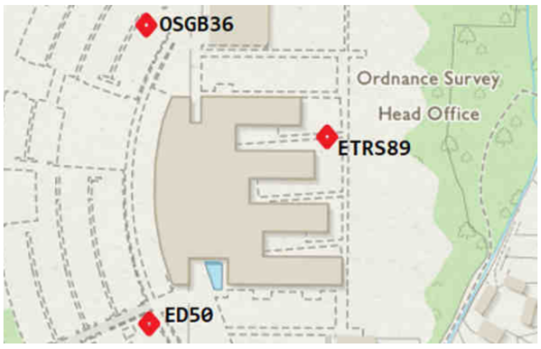


Figure 2.9.: Same latitude and longitude location in three different coordinate systems [22, p. 6].

Depending on the coordinate system, the points deviated up to 200 meters [22, p. 6]. In order to locate a point on the surface as accurately as possible, the used coordinate system needs to be specified precisely for further processing.

The following section explains, what types of coordinate systems are commonly used and how a geocoordinate is unambiguously defined.

2.4.1. Types of coordinate systems

A geocoordinate, together with a defined geocoordinate system, should uniquely and numerically describe a point on the surface. This coordinate system must be defined so that each topographic feature of a surface can be described unambiguously and precisely by a set of numbers. Below three different geocoordinate systems are briefly introduced and explained.

2.4.1.1. Latitude, longitude and ellipsoid height

The most common way to define a position on the earth is by latitude angle ϕ , longitude angle λ and ellipsoidal height H. These three parameters define a point on the reference surface of an ellipsoid. An ellipsoid is a mathematical reference model that approximates the shape of a celestial body. Depending on the use case and its accuracy requirements different ellipsoids either globally best-fitting or locally-best fitting ellipsoids are applied. In order to process geocoordinates precisely, the used ellipsoid needs to be known in advance.

Lines that run along the ellipsoid from north to south pole having the same longitude are called *meridians*. Lines from west to east with the same latitude are called *parallels*. All points along the equator of the ellipsoid have a longitude of 0 degrees. Depending on the geocoordinate system, one meridian is defined as a prime meridian. The longitude of each point on the ellipsoid surface is the angle between the prime meridian and the meridian passing through the point [22, pp. 11 sq.]. The latitude is the angle between the equatorial plane and the normal of the point on the ellipsoid surface.

Using latitude and longitude, the direction vector between the center of the ellipsoid and the point to be determined can be calculated. The third parameter, the height, describes the elevation difference between the point and the ellipsoid's surface. Figure 2.10 shows a point P described in different coordinate systems.

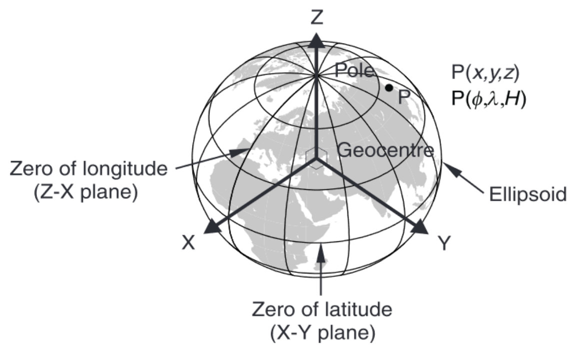


Figure 2.10.: Point p described in Cartesian and latitude, longitude height coordinates [22, p. 12].

2.4.1.2. Cartesian coordinates

Another way to numerically describe a location is to use Cartesian coordinates. A three-dimensional coordinate defines a point relative to the center of an ellipsoid in this type of coordinate system. The origin of this system is equal to the center of the specified ellipsoid. The x-axis intersects the prime meridian and the equatorial plane. The y-axis intersects the equatorial plane and the 90° longitude in the west. The z-axis corresponds to the polar axis, intersecting the south and north poles [22, p. 13]. Figure 2.10 depicts the relationship between a Cartesian and latitude, longitude, height coordinate system.

If the used ellipsoid, as well as the prime meridian are equal, coordinates can be converted from the latitude, longitude, height coordinate system to the Cartesian coordinate system and vice versa. If a different ellipsoid or prime meridian is used, the coordinates have to be transformed.

2.4.1.3. Eastings and northings

Another type of geocoordinates are *grid coordinates* or *map coordinates*. These coordinates are used to locate a position on a two-dimensional map. The main axes of such a map are called *eastings* and *northings*. A three-dimensional curved surface is projected onto a two-dimensional plane. This type of projection is called *map projection* [22, p. 17]. Depending on the map projection, the same point is transformed into different grid coordinates. A well-known map projection is the Universal Transverse Mercator (UTM) projection. Since the projection inevitably loses information, it is mainly used in geospatial information systems for visualization purposes and not for further data processing [22, p. 17].

2.4.2. Geodetic datum

The previous section showed that the definition of a coordinate system depends on different parameters.

If one of these parameters is changed, the numerical representation of a location is also altered. Further, the modified and original geocoordinate systems are no longer compatible. To effectively compare or process coordinates from two distinct coordinate systems, they have to be transformed into a common coordinate system [22, p. 18].

The *Terrestrial Reference System* (TRS) defines a coordinate system uniquely. This standard is also known as the *geodetic datum*. Crossley explains in his work:

> "The datum definition consists of eight parameters: the 3-D location of the origin (three parameters), the 3-D orientation of the axes (three parameters), the size of the ellipsoid (one parameter) and the shape of the ellipsoid (one parameter)" [22, p. 18].

The most widely known geodetic datum is WGS 84, which is used for the *Global Positioning System* (GPS). Other geodetic datums are required for the precise location of points on other celestial bodies.

2.5. GeoTIFF

GeoTIFF is an image format standardized by the Geospatial Consortium [19]. As an extension to the TIFF format, it can store 32-bit information per color channel. Due to the lossless compression and the high available color depth, this format is used for storing scientific image information [112]. The GeoTIFF standard was primarily developed as a distribution format for aerial photography and digital terrain data [112]. GeoTIFF is distinguished from the TIFF image format because additional meta-information for georeferencing the image content is attached [19]. Using georeferenced images allows the image content to be mapped precisely to a geocoordinate system. Each pixel in the image describes a property of a specific location within an area. To relate the image to a geographic coordinate system, meta-information describing the corner coordinates of the area covered by the image and the geocoordinate of the center of the image area are attached [19]. In addition, the spatial resolution of the image contents is described. This spatial resolution is the length in meters, covered by a pixel length in the image. In order to process the geocoordinates correctly, the used geocoordinate system must be specified. Therefore the GeoTIFF standard requires information about the map projection used and the geodetic datum [19].

2.6. ARCHES Project

ARCHES (Autonomous Robotic Networks to Help Modern Societies) is a Helmholtz-future project. The following section briefly introduces the ARCHES project. The technical capabilities of the rovers developed in the ARCHES project provide the restrictions for the visualization of rover data discussed in this thesis. First, a brief explanation of the project, its goals and duration is given. Then, the rovers and their sensors developed in the ARCHES project are described.

2.6.1. Outline

The vision of this project is to develop heterogeneous autonomous and interconnected robotic systems. The goal is to unite research on robotic systems across disciplines and domains. A diverse team of rovers, each with their own capabilities and scientific measurement tools, explore an unknown environment autonomously and collaboratively [82]. The project's focus is on investigating the basic requirements for the autonomous operation of the robot network. This research should also be transferable to several research areas, like autonomous ocean environmental monitoring, crisis intervention scenarios, and solar system exploration. The Alfred Wegener Institute, the Helmholtz Centre for Polar and Marine Research, the GEOMAR Helmholtz Centre for Ocean Research Kiel, KIT - Institute of Technology, and the DLR are all partners in the ARCHES project [82].

The joint exploration will be tested in a demo mission in summer 2022, initially with only a small group of rovers at Mount Etna. This demonstration mission is divided into three stages [83].

During the first phase, two rovers and a drone will autonomously explore a defined area and collect targeted scientific data at relevant locations. During the second phase, the teleoperation and remote control of the rovers will be investigated in greater depth. The third scenario will be used to see if the rovers' position accuracy can be improved by using low-frequency radio antenna arrays [83]. Only the first scenario is relevant, because this work focuses on evaluating scientific measurement data.

2.6.2. Rover Hardware

As part of the ARCHES project, DLR is developing various heterogeneous robots. Two *lightweight rover units* (LRU 1 and LRU 2) and an aerial vehicle (drone) called *ARDEA*, are being developed. These three separate robots will work together to explore a defined territory and conduct scientific experiments. The rovers differ in the scientific instruments, they carry on board.

LRU 1: The LRU 1 rover is an individually controllable vehicle. The rover can travel at a speed of 1.1 m/s. The main tasks of this rover are autonomous terrain mapping and the acquisition of multispectral images of the area and soil samples [130].

The rover is equipped with a pan/tilt unit. This unit consists of a collection of camera systems. Two stereo navigation cameras are used to capture the landscape and rocks. The camera system also includes an infrared camera and a narrow-angle color camera. A greyscale camera is attached to each end of the pan/tilt unit. These two cameras are used together as an additional stereo camera.

In front of each sensor in this stereo camera, a filter wheel with nine individually selectable filters is located. Each filter wheel contains three color filters with a bandwidth of 100nm and six narrow band filters with a bandwidth of 10nm. The filter wheels on the left and right have the same three color filters but different narrow band filters. Light wavelengths ranging from 440nm to 660nm, as well as 720nm to 1000nm, can be filtered by these filter wheels [130].

Different materials absorb light at different wavelengths. These multispectral images are used to derive information about the material properties of the objects visible in the image. Even materials that look the same in visible light can have different material properties. These multispectral images are used to distinguish these material properties from one another [130].

LRU 2: The LRU 2 rover is equipped with a pan/tilt unit like the LRU 1 rover, but instead of the science cam, this rover uses just a stereo navigation camera to orient itself in its environment. A built-in manipulator arm allows the rover to collect soil and rock samples from the ground. The primary purpose of this rover is to be used as a vehicle for transporting payloads, such as collected rocks. These samples can then be analyzed for chemical composition directly onboard using a technique called *laser-induced breakdown spectroscopy*. The spectrometer covers a wavelength range from 550nm to 770nm. Hence common rock-forming elements such as calcium, sodium or potassium can be chemically detected in the samples [130].

ARDEA The ARDEA drone outperforms the rovers in terms of maneuverability. Flying through the atmosphere allows the drone to move quicker than the rovers. It is used to quickly and roughly survey new unknown areas from the air and identify points of interest where scientific activities can take place. For aerial photography, a wide-angle stereo camera is used [130].

3. Related work

This section aims to summarize the most relevant literature and recent advancements for this thesis briefly.

3.1. Augmented reality for space exploration

From the early development phases of mobile AR devices on, NASA explored the use of these devices in space missions and astronaut programs. An early device was developed collaboratively in 1998 between the University of California Los Angeles, the Jet Propulsion Laboratory and McDonnell Douglas called the *Wireless Augmented Reality Prototype* [3]. The goal of this device was to display information wirelessly from the astronauts' biometric sensors, as well as images and text in front of the astronaut's eyes. The data to be displayed is exchanged via a wireless connection between the head-mounted display and an external computer. This prototype represents one of the first wireless HMDs designed for space exploration.

Today, NASA is still working on integrating augmented reality approaches to assist astronauts. As part of the project *Sidekick*, several Microsoft HoloLens 1 devices were transported to the International Space Station (ISS) on December 6th, 2015 [129]. The intention was to test crew assistance scenarios using the HoloLens. Two distinct use cases were examined in depth.

The first one addresses the scenario of remote collaboration. An expert on Earth can see the HoloLens wearer's first-person perspective in real-time and assist the crew member with additional information and three-dimensional annotations.

The second use case investigated how checklists and illustrations can help crew members work through fixed procedures. Animated virtual assets are displayed next to the objects to be interacted with, according to the current work step. The project ended in March 2016. Unfortunately, the results of the studies conducted at the ISS are not available [115].

A similar application scenario was also explored by Helin et al. in 2018 [63]. They investigated how manual activities are performed when assisted with augmented reality based on HoloLens 1. Two different scenarios were examined using an ISS service module replica in ESA's European Astronaut Centre [63].

In the first scenario, Helin et al. investigate the use case of ad-hoc remote support by the control center on Earth. For this purpose, the voice and camera feed of the HoloLens is sent to experts in the control center. They guide the crew member by annotating the actions to be taken.

In the second scenario, the test person has to replace a component in a standardized procedure. During the procedure, the proband is supported by virtual arrows pointing to the required tools, as well as video, photo and audio overlays for correct assembly. It was shown that such an AR system is generally perceived as positive and helpful by the user. The system was evaluated using the System Usability Scale (SUS) and achieved an above average score of 68 [63].

Augmented reality, based on Microsoft HoloLens, is already being used for assembly work on future space missions. One example is the planning and assembly of the Orion space capsule, where the individual assembly steps are visualized to the engineers [86].

Further AR solutions are being developed in the field of manned space exploration outside the ISS, especially for the exploration of other planets. With LunAR, Radway et al. present an augmented reality system developed for deep-space exploration missions. This system is designed to be operated even under the motor restrictions of an astronaut in a space suit [122]. A user interface concept based on speech recognition is presented. A camera integrated with the AR system allows astronauts to document their actions as a photo and to annotate these photos using voice recognition. During this, the astronauts are supported by instructions from mission control. With the help of object recognition and pose estimation, tools are recognized in front of the astronaut. The recognized pose is continuously checked, and the system alerts the astronaut in case of errors [122].

A similar application scenario is also being investigated in the Holo-SEXTANT project. Anandapadmanaban et al. are researching the use case of astronaut guidance while walking on another celestial body [5]. A HoloLens-based AR system guides the astronaut with navigation instructions in the user's field of view. For this purpose, the HoloLens is equipped with an external GPS sensor. Via network, the device receives the following waypoints and shows them to the user. In a field test, the researchers showed that the overlay of additional information relieves the user's cognitive load in contrast to classic paper map navigation.

3.2. Mission planning in augmented reality

In addition to space exploration, AR is applied in the field of mission planning.

Su et al. investigate how augmented reality can assist in planning extraction scenarios of *high value targets* from a building [139]. A static landscape model and a three-dimensional layout of a building are displayed in miniature format. The extraction path is planned via a user interface by setting markers. Voice commands control the perspective on the individual floors of the building. Several AR devices are connected in a co-located collaboration scenario. An external tracking system synchronizes the different coordinate systems of the HoloLenses.

Ruano et al. present an approach to display georeferenced points on the real landscape in AR [127]. The presented AR system is used to assist a remote pilot while controlling an unmanned aerial vehicle (UAV). The project's goal was to increase the operator's situational awareness by superimposing additional georeferenced information on a live camera feed of the aircraft. The AR system receives the geocoordinates of the following waypoints to be flown. These geocoordinates are first transformed into a common geocoordinate system. Using the UAV camera's extrinsic and intrinsic parameters, the geocoordinate is displayed to the operator as a three-dimensional marker on the camera feed. The augmented reality system avoids the pilot having to look at several screens simultaneously. Ruano et al. were able to show that augmenting the virtual markers on the real environment increases situational awareness [127].

Especially in the field of virtual reality, solutions for collaborative mission planning already exist. An example of collaborative planetary research is the EU project CROSSDRIVE [53, 54, 27]. Garcia et al. are working on infrastructures for collaborative mission planning scenarios in virtual reality [53]. The goal is to integrate remote scientists and engineers into the data analysis process through virtual reality visualization and interaction.

CROSSDRIVE is intended to run in a virtual environment similar to CAVE[1] (see figure 3.1). With a virtual cursor attached to a flystick, two-dimensional menus can be selected.

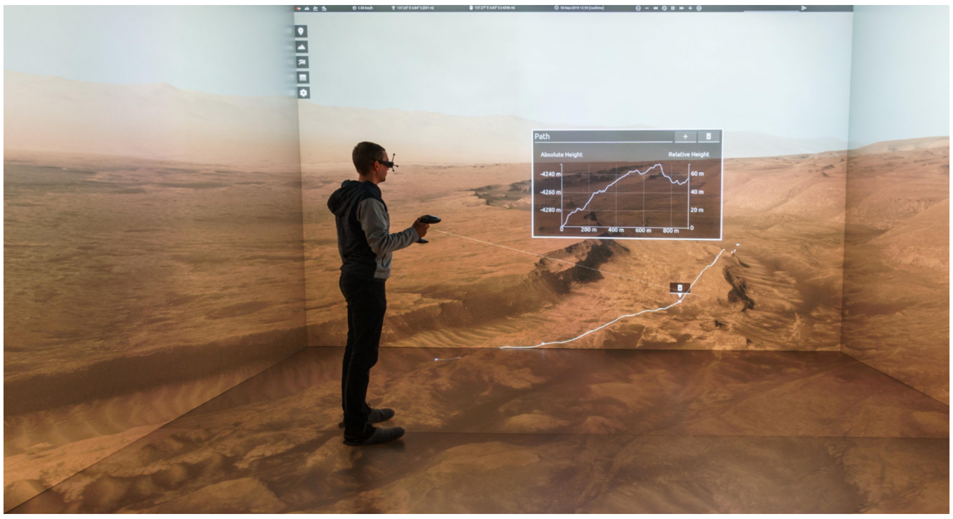


Figure 3.1.: CosmoscoutVR running in CAVE [55].

In contrast to an AR environment, where the user can move freely in the real world, the freedom of movement is restricted, especially in CAVE systems. The user depends on fly-navigation and teleportation approaches to move in the virtual environment.

Garcia et al. argue that non-verbal communication between partners is essential for successful collaboration [53]. In contrast to a co-located collaboration scenario, Garcia et al. mainly consider remote collaboration, where each participant is in his or her own virtual CAVE environment. To enable this non-verbal collaboration, the scientists reconstruct the participants as virtual avatars based on Shape-from-Silhouette algorithms [53].

Besides the research in the field of telepresence, the project CROSSDRIVE also borders on the research field of real-time visualization of large landscapes [53]. A stable frame rate

[1]CAVE: Cave Automatic Virtual Environment

is even more important in virtual reality than in augmented reality. If the frame rate is too low, there is an increased risk that the user will experience discomfort and dizziness. In the CROSSDRIVE project, a planet-scale landscape is displayed with a minimum refresh rate of at least 30 Hertz per eye. To archive this goal, a level of detail approach is used to minimize the amount of data loaded into memory at any time. That way, the geometry to be processed by the graphics cards is kept as simple as possible [27]. In a quad-tree like structure, the different levels of detail of the surface information are organized. This geometry is tessellated in a *HEALPix* structure [58, 144]. This tessellation is shown in figure 3.2. A sphere is subdivided hierarchically into curvilinear quadrilaterals. Each pixel of this tessellation with the same resolution covers an area of the same size.

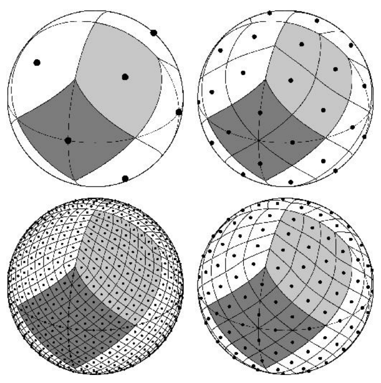


Figure 3.2.: HEALPix tesselation of a spherical surface [58].

These features are beneficial to organizing the terrain data in a quad-tree structure efficiently. The individual detail levels of the surface data are generated in a preprocessing step on a GPU cluster [27]. Using a web map service, the renderer can request the data of a specific landscape segment at a desired quality level over the network. The requested textures and models are then streamed into the visualization.

The CROSSDRIVE project shows many parallels with the use case considered in this thesis. In both scenarios, the detailed representation of the landscape surface at interactive frame rates is a central aspect. The focus on collaboration is considered in the CROSSDRIVE project as well as in our use case. However, in the CROSSDRIVE project each expert is located in their CAVE-based virtual reality system at a different place. In our use case, co-located collaboration in AR is explored. All experts are in the same real environment.

Garcia et al. also discuss the use of HMDs for future scenarios. At the moment, only a normal mono or stereo screen or a configurable CAVE projector system is supported as output medium [54]. High-performance desktop computers or computer clusters with desktop graphics cards are responsible for the image synthesis. This is in direct contrast to our use case, where the hardware with HoloLens is very specific and limited in computing power.

A further development of the concepts for landscape visualization developed in CROSS-

DRIVE is the open-source CosmoscoutVR project [55]. Similar to CROSSDRIVE, it uses a combination of map server, HEALPix tessellation and preprocessed data to visualize large landscapes in real-time. Using an HMD, scenarios for landing site investigation, weather data visualization and on-orbit servicing can be explored in virtual reality.

With Virtual Mission Control (vMCC), Anandapadmanaban introduces another virtual reality-based concept for mission planning [4]. In a remote collaboration scenario, several experts are located in a virtual room. Avatars of the participants are visible to the other participants. In a virtual planning room, a shared tabletop visualization for three-dimensional models is located in the center. At the same time, the walls of the virtual environment serve as a projection surface for two-dimensional images. Any image from the computer's hard drive or a virtual desktop can be displayed on this projection surface. All participants can see and interact with these virtual elements. The experts discuss the displayed data with virtual laser pointers and voice transmission. The 3D models displayed in the tabletop view and the walls can be annotated with freehand drawings using a 3D drawing tool. Although static three-dimensional terrain models can also be displayed in tabletop view, this work focuses more on the user interface and the developed collaboration tools than on a geographically precise representation of the terrain. The concept presented was developed for a virtual reality headset connected to a desktop computer. The author stated that the concepts presented could also be used for mobile head-mounted displays. However, a great deal of effort would be required to optimize the performance and memory utilization since the capabilities of mobile devices are more limited [4].

3.3. Augmented reality applications in geospatial information systems

Temporal changes in the terrain are research areas of geoobservers. You and Thompson present a system that uses mobile AR devices in the form of smartphones and tablets [148]. The presented scenario consists of an in-situ scientist at the landscape section to be observed in person and a scientist in the office, both equipped with a mobile AR device. With the help of optical tracking methods and the use of GPS and compass data from the mobile device, the pose and their global geocoordinate and orientation are determined in real-time. The person in the office sees the current position of the in-situ person on a digital model of the landscape in either a miniature format or at a 1:1 scale. The scientist in the office can annotate the landscape model with touch gestures. The other scientist sees these annotations superimposed on the real landscape. The management of the landscape data and the rendering of the virtual content, is performed in real-time on a server architecture.

Hedley et al. compare user interfaces for displaying geospatial information system (GIS) data [62]. With the AR PRISM interface, several experts can view a defined landscape section in a co-located collaboration scenario using digital see-through head-mounted displays in a tabletop setting. A two-dimensional map is augmented with the corresponding three-dimensional elevation model. An external tracking system is used to determine the head poses of the participants. By moving a physical marker on the two-dimensional map, the experts determine which part of the landscape is augmented. This marker is used to focus the attention of the other participants on relevant parts of the landscape and reduce the computational and memory load of the rendering machines.

One of the best known geospatial information systems for the general public is Google Earth. With the release of its virtual reality extension (Google Earth VR) in November 2016, users can explore the globe in immersive virtual reality [57]. The landscape is presented either in miniature format or as a two-dimensional horizontal surface for quick navigation in front of the user. Panning, zooming, and rotating are implemented using six degrees of freedom controllers to perform one or two-handed gestures. Google Earth VR uses a desktop graphics card connected to the head-mounted display. Using Google Earth VR on a mobile HMD is not supported.

During this thesis, Epic Games presented an AR demo project called *Project Anywhere XR* [43]. This project uses publicly available geodata sources. These data include point clouds, photogrammetry, and typical raster data, like digital elevation models and high-resolution satellite imagery. Using a standardized format called *3D Tiles*, this heterogeneous data collection is streamed over the network. The landscape is displayed in miniature format. While the orientation of the landscape can not be changed, navigation gestures for panning and zooming are supported.

Although Project Anywhere XR was clearly developed to operate on HoloLens 2, the application itself is not executed on HoloLens, but uses *holographic remoting* to outsource the graphics- and memory-intensive computations to an external high-performance computer. The project relies on a commercial cloud infrastructure to preprocess and filter the geospatial datasets. The HoloLens acts exclusively as an input and output device. The virtual content is rendered on an external computer and sent over the network as a compressed video stream with a delay. In contrast to their approach, our prototype should be executed directly on the HoloLens device, to avoid compression artifacts and transmission delay.

3.3.1. Large scale terrain rendering in game engines

Both Unreal Engine and Unity3D are two of the most widely used public game engines. A core aspect of many of today's video games is a large game world. The representation of realistic landscapes, some of which are several hundred square kilometers in size [145], is a significant challenge of current video games. In order to solve this problem, the game engines are working on ready-made solutions for the realistic representation of large-scale landscapes.

Both engines offer already implemented components for landscape visualization. Importing of heightmaps and the conversion into terrain geometry are standard features provided by the engines. However, the landscapes must be converted to a suitable format in both engines before building and packaging the application. It is not possible to dynamically load new landscapes at application runtime using the tools provided.

Unreal Engine has a tile-based approach to manage and render landscapes [44]. Instead of the whole landscape being one coherent model, the landscape is divided into several components. Each component covers the same area size and has a square shape. Basic settings for collision detection, visibility and material options are defined at the component level. Each component is further divided into four sections. These sections serve as a logical unit for controlling the landscape geometry level of detail [44]. The section contains the actual geometry data, represented by a grid of quads. Components outside the user's field of view are automatically unloaded from the main memory and dynamically reloaded when needed. Sections far from the user are rendered in a lower quality level to save computing time [44]. To make the landscape surface as detailed as possible, the geometry in each section is subdivided using hardware-accelerated tessellation. In order to use the landscape system provided by Unreal Engine, the GPU must support tessellation shaders.

Lütjens et al. display bathymetry data with the Unreal Engine landscape system [84]. A total landscape area of 160 km x 80 km is explored in lifesize scale virtual reality. Lütjens et al. utilize engine-provided level of detail approaches to visualize distant landscape segments at lower complexity. The parts of the landscape are dynamically swapped depending on the user's location and viewing direction. Since the user sees the landscape from a first-person perspective in this scenario, the computational load can be significantly reduced by varying the level of detail. Figure 3.3 shows the different levels of detail of the loaded sections. Depicted is a top-down perspective on the landscape. The user's virtual camera is located in the center of the white circle.

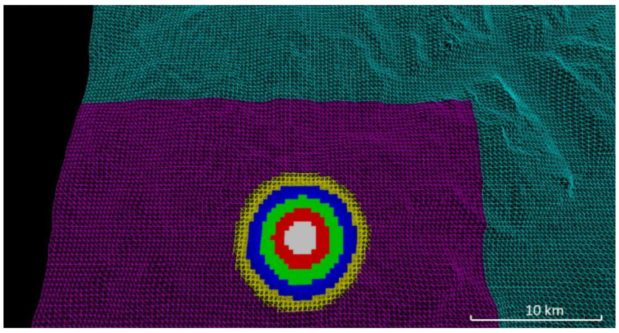


Figure 3.3.: Landscape sections in different level of details: White sections are loaded with full spatial resolution. In the sequence of red, green, blue, yellow and pink, the level of detail is reduced. Cyan sections are not loaded in main memory [84].

The landscape sections close and centered around the user are rendered in the original quality level. Sections further away are displayed in a lower quality. A user study showed that the representation of large landscapes in virtual reality improves the perception of the topography compared to the classic representation as a two-dimensional map [84]. Furthermore, it was demonstrated that real-time rendering of such large landscapes is a challenge even with stationary desktop computers[2]. The minimum refresh rate of 60 Hertz, which is usually targeted for virtual reality applications [34], could not be met even using the optimizations for landscape rendering provided by Unreal Engine [84].

3.3.2. Terrain navigation in augmented reality

The work of Kato et al. is one of the first to transfer existing design principles of tangible user interfaces into tabletop AR applications [70]. Kato et al. introduce the term *Tangible Augmented Reality* and define it as an AR system that takes advantage of a tangible user interface and meets the following five criteria:

- Object affordances should match the physical constraints of the object to the requirements of the task.
- The ability to support parallel activity where multiple objects or interface elements are being manipulated at once.
- Support for physically based interaction techniques (such as using object proximity or spatial relations).
- The form of objects should encourage and support spatial manipulation.
- Support for multi-handed interaction. [70]

Human-machine interaction with two-dimensional digital maps has been explored in the past from various aspects. Interaction with mouse and keyboard, as well as other input devices, such as multi-touch gestures on mobile devices [71, 6] or large screens [146, 7, 78, 59], is subject of research. Further studies focus on two-dimensional map interactions using input methods such as speech recognition or three-dimensional gestures [21].

Austin et al. investigate various hand and foot gestures for navigating an AR tabletop map visualization [8]. Using a digital-see-through HMD, a map of New York City is presented with three-dimensional buildings in a miniaturized format. Hand, as well as foot position, are tracked in real-time three-dimensionally. Different hand and foot gestures for navigation are tested for their intuitiveness in a user study. In particular, gestures for panning, rotating, and zooming, as well as changing the ground level height of the map, are tested for their acceptance [8].

[2]PC was equipped with Intel Core i7-6700HQ, NVIDIA GeForce GTX 1060, and 16 GB RAM

Pan: Most users either swipe with an open hand or use a grab and drag gesture to move the map. Both gestures are shown in the first column of figure 3.4.

Rotate: Most subjects preferred an input action analogous to turning a wheel in the real world to rotate a landscape. This action is shown in the second column of figure 3.4.

Zoom: The most intuitive gesture for zooming into the map is to grasp the landscape with both hands and then control the zoom level by changing the distance between the two hands. The third column of figure 3.4 shows the hand movement.

Change height: Most probands prefer a two-handed gesture to raise or lower the entire map. The height of the map is adjusted by lifting the forearms. This gesture is shown in figure 3.4 in the fourth column.

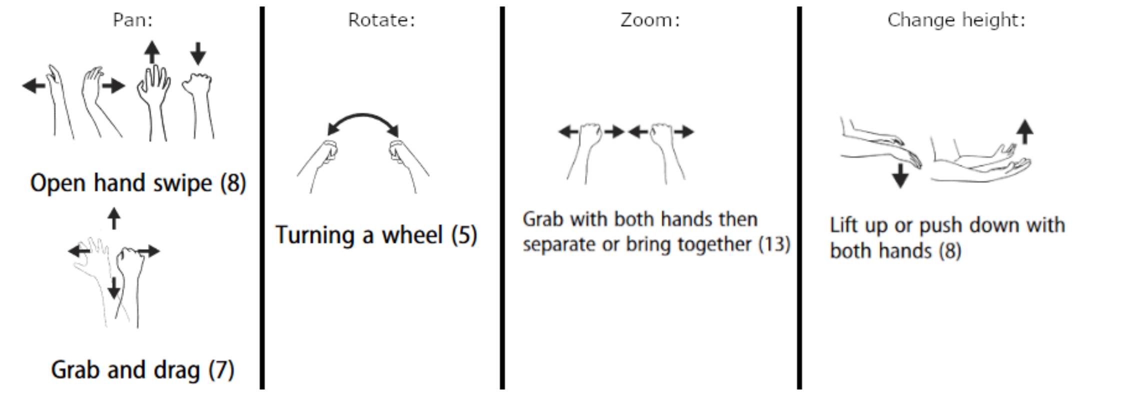

Pan:
Open hand swipe (8)
Grab and drag (7)
Rotate:
Turning a wheel (5)
Zoom:
Grab with both hands then separate or bring together (13)
Change height:
Lift up or push down with both hands (8)

Figure 3.4.: Most common map navigation hand gestures in augmented reality [8].

3.4. Scientific visualization in augmented reality

3.4.1. 3D Photobrowsing

The idea of linking photographs with a map to create a spatial connection between photographs and their surroundings is not new. Takata, Kanaya, and Sato proved that the spatial sorting of several photos emphasizes the connection between them [141]. In the context of an archaeological site, a photo browsing system was presented, allowing a digital display device to be placed at different positions on a two-dimensional map of the excavation site. The position and orientation of this display device relative to the map is automatically determined, and the photo taken at this location is shown on the output device. A user study found that the presented photo browsing tool can be effectively used to improve the spatial relationship between the image and the map. Under the condition that the position of a searched photo is known, users could find this photo faster than without spatial sorting [141].

Their work focuses exclusively on the spatial localization of photos on a two-dimensional map. Other approaches that automatically sort a collection of photos in relation to each other in three-dimensional space are also being investigated. Two prominent examples of such approaches are *Photo Tourism* and *Photo Navigator* [134, 65]. In both works, feature matching is used to reconstruct the spatial relationship between the photographs. Photo Tourism follows the premise that many photographs are available for reconstructing the environment and the spatial relationships between the images. Thus, a dense photo data collection is required. Photo Tourism uses public photographs of famous landmarks from the internet as a data source. The estimated camera frustums are visualized three-dimensionally in the reconstructed environment in the user interface. This interface allows the user to navigate the reconstruction and move in-between images using image-based morphing techniques.

Photo Navigator has a slightly different focus and tries to reconstruct the spatial relationship between photos even with a very sparse photo collection [65]. In order to achieve this, a spatial slideshow of photos is used to smooth the transition in-between images based on the reconstructed environment. A user study showed that with the additional spatial reference of the photos to each other, the viewer retained spatial details of the depicted environment significantly stronger in memory.

These studies show that photos with a spatial context can be explored more intuitively. By emphasizing the spatial connection between the pictures, visual content elements are recalled more easily.

3.4.2. Photospheres in augmented and virtual reality

The visualization of 360 degree panoramic images is an active research topic in virtual and augmented reality applications. The projection of those pictures in HMDs enables the exploration of a remote environment.

Robertson et al. [125] compare the perception of 360-degree panoramic photographs shown as an immersive photosphere surrounding the user in VR to a two-dimensional visualization of the panorama. It was shown that the subjects in a photosphere could better perceive the details of the panorama image.

In another study, Ball demonstrates that photospheres can be used to accustom medical professionals to a new environment [10]. In the study, a photosphere panorama image of clinical environments was presented to probands. Two different conditions are evaluated. In one condition, the subjects perceived the clinical environment exclusively as two-dimensional photographs. In the second condition, the subjects perceived the same environment as a photosphere in augmented reality. Afterwards, all subjects visited the previously viewed clinical environment in person. It could be shown that the subjects, who had explored the environment in an immersive photosphere before, perceived the environment as more familiar, and the stress level was significantly lower.

3.4.3. Stereo image visualization in augmented reality

An important aspect of human depth perception is the processing of stereo depth cues. The human being perceives these stereo depth cues through his two eyes, which are slightly shifted in perspective, and processes the perspective disparity in the brain. This disparity allows us to estimate the distance to an object in front of our eyes more precisely.

In the context of robotic systems, optical depth perception based on stereoscopic images is essential to estimate the robot's distance to other objects. Many robotic systems are still remote controlled by humans. Therefore, how humans perceive depth provided by stereoscopic data is an active area of research. The most common representations of these stereoscopic images are [76]:

- Active 3D screens
- Passive 3D screens
- Head-mounted displays

Kot, Novak, and Bajak compare the advantages and disadvantages of these display types [76]. In addition to virtual reality HMDs, they examine the HoloLens as a possible output device for stereoscopic images. A user study compares the display in VR, AR, and on a 3D monitor. In both VR and AR, the stereoscopic image is placed on a fixed position as a two-dimensional virtual screen. A *Horizontal Image Translation* technique is shifting the two images on the virtual monitor to correct for the varying perspective to the screen [76].

It was shown that the perception of spatial distance is significantly better with both VR and AR displays than with a 3D monitor. Even though the depth perception in VR is slightly better, HoloLens outperformed the VR system in the described tests, due to its better surrounding awareness and comfort [76].

3.5. Co-located multi-user collaboration for data analysis

One of the first publications of a co-located collaboration AR interface is the project *Studier-Stube* by Szalavari et al. from 1998 [140]. HMDs are used to explore three-dimensional virtual models superimposed on the real world collaboratively. A user study showed that users prefer collaboration in person compared to remote collaboration because the proximity of the experts to each other enables better verbal and non-verbal communication. Szalavari et al. [140] identify five key aspects for collaborative AR environments:

- Virtuality: Objects that don't exist in the real world can be viewed and examined.
- Augmentation: Real objects can be augmented by virtual annotations.
- Cooperation: Multiple users can see each other and cooperate in a natural way.
- Independence: Each user controls his own independent viewpoint.

- Individuality: Displayed data can be different for each viewer.

Lee and Hua investigate the impact of personalized tabletop visualizations in different co-located collaboration scenarios [80]. In one scenario, two experts work on the same task, which requires each expert to refer to the virtual objects of the other expert. In a second scenario, the experts work on different tasks less dependently on each other. A user study examined how fast and frustrating the experts in both tasks are, if the virtual objects are arranged differently for each expert. It was demonstrated that a personalized view for the first task, in which the objects have to be strongly referenced, negatively influences the task completion time. In the second task, the experts were less dependent on each other, and an individualized presentation provides a more focused view and thus reduces the task completion time.

Lee et al. demonstrate that in a collaboration scenario, even in virtual reality, users instinctively distinguish between a shared workspace and the workspaces of other experts [79]. Subjects respected the limits of their workspaces and did not interact with virtual objects from others. Discussions about virtual objects could be initiated using the shared virtual space. They stated that collaboration tools such as a virtual laser pointer and avatars are necessary features for a successful discussion in VR.

4. Methods

This chapter describes the theoretical and methodical considerations for the implementation of the AR demonstrator. The prototype developed in this thesis will be evaluated to see how AR may be integrated into the rover data analysis process. As mentioned before this thesis deals with three core aspects: large-scale terrain rendering, rover data visualization, and multi-user collaboration. Each of these core topics is first considered theoretically in this chapter.

4.1. Landscape visualization

The landscape visualization aims to give the experts an overview of the rover's surroundings and provide spatial context to the collected scientific results. Therefore the terrain has to be presented as detailed as possible. This section explains the concepts used to render large-scale and high-resolution landscape datasets in interactive frame rates, even with the limited computational power of the HoloLens.

4.1.1. Visualization format

Researchers analyze the raw results of the rover's scientific measurements. To contextualize these results, they consider the exact location, and if other measurements have already been collected in the vicinity. One scenario might be the analysis process of a soil sample's chemical composition. Researchers might want to compare the composition of one sample with the composition of other samples in the vicinity. Sometimes, the shape of the terrain or the topological environment of a finding is necessary to interpret scientific measurements. The goal is to use an immersive AR landscape representation to integrate the scientific results in a spatial context with the surrounding terrain.

Two different presentation modes can be distinguished for the visualization of landscapes [137].

The first approach lets user explore the landscape on a 1:1 life-size scale. This approach offers the opportunity to move on the surface like in the real world. Provided a detailed terrain data, the user can explore the virtual terrain realistically. At the same time, however, this approach has the disadvantage that the 1:1 scale limits the user's range. To move one meter in the virtual landscape, the user needs to move one meter in the real world or apply virtual teleportation techniques. The exploration of large landscapes is thus cumbersome.

The second visualization format is called *world in miniature* (WIM). Using this approach, a scaled-down landscape is rendered in front of the user. A landscape with a length of l is shown in miniature on a model with a length t. The scaling factor s results accordingly from the ratio of t to l.

$$s = \frac{l}{t} \tag{4.1}$$

Large landscapes are mapped onto a smaller model with this scale. The user can get an overview by looking at the whole scene from a top-down perspective.

For the use case discussed in this thesis, the landscape visualization should emphasize the spatial context of each scientific measurement. In the ARCHES demo mission, a total area of about 15 x 15 kilometers is considered. Navigating a landscape section of this size in life-size scale AR is difficult, confusing and does not provide a situational overview. Therefore, the landscape is presented to the experts in a scalable world in miniature format (see figure 4.1). In our use case, several experts in the same room should inspect the landscape section and interact with it. The model is visualized in a tabletop format in the center of the room. It was decided to display the virtual landscape in a cylindrical shape to make it easier for the experts to view the terrain from all sides.

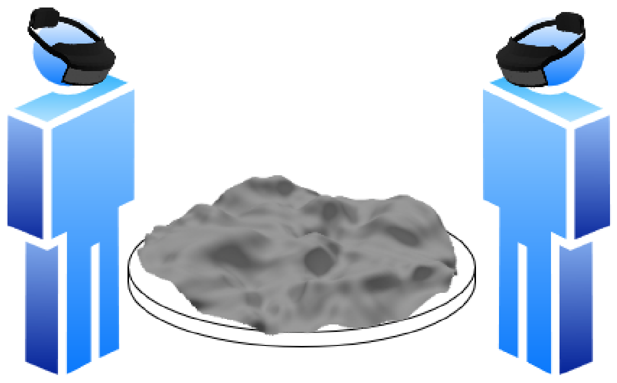


Figure 4.1.: Two experts discussing the landscape in world in miniature format.

4.1.2. 3D terrain rendering

In order to show the terrain three-dimensionally, information about the elevation profile of the mission-relevant areas is required. This elevation data (also called *digital elevation model* or DEM) is a texture encoding the height of a defined landscape section in each pixel. It is assumed that the elevation data is given in a GeoTIFF image format. As discussed in section 2.5, one distinguishing feature of this format is that metadata about the georeference is attached. This can be used to map precisely one geocoordinate to a pixel coordinate. Conversely, it is also possible to read the height of a specific geocoordinate from this GeoTIFF image.

To display the DEM as a three-dimensional landscape model, the straightforward solution could be using the pre-implemented components for landscape visualization of the Unreal Engine. Those components include optimizations in terms of memory utilization and visual quality (see section 3.3.1). Several computational operations are outsourced from the CPU to the GPU. One of these operations on the GPU is a hardware-accelerated tessellation of the landscape surface. The Engine utilizes this feature to dynamically adjust the granularity of the landscape without changing the geometry stored in main memory. However, HoloLens 2 hardware does not support hardware-accelerated tessellation in the Unreal Engine graphics pipeline. Trying to implement this approach, it showed that the landscape could not be rendered at all when running on HoloLens. Furthermore, UE landscape visualization does not support the dynamic integration of new landscapes at runtime but requires the landscape to be defined statically already at compile-time. Therefore this tool can not be used for our use case.

Another approach of landscape visualization is based on the idea that the surface can be approximated by a two-dimensional uniform vertex grid. A 3D-coordinate for each given geocoordinate can be determined by sampling the height information of the DEM. The X and Y-component can be computed using geocoordinate transformations (later explained in section 4.4.2), and the height (Z-component) is read from the elevation model. Figure 4.2 shows such a landscape model. The geometry shown in figure 4.2a is a uniform vertex grid evenly distributed in the XY plane. Each vertex is moved up or down according to the value encoded in the elevation texture in figure 4.2b.

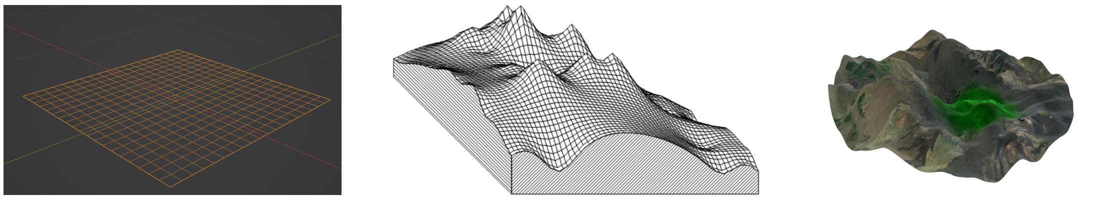

(a) Uniform vertex grid (b) Height displaced vertex grid [69] (c) Landscape visualization with satellite image and false color image

Figure 4.2.: Three-dimensional representation of a digital elevation model.

Additional textures such as satellite images can be visualized on the surface of the landscape model. Further scientific evaluations as false color images can be displayed on the surface. Figure 4.2c indicates in green the rover's previous explorations. Multiple color textures can be blended together and are visible on the landscape. Using a height displaced uniform vertex grid could be a suitable approach to render the landscape on the HoloLens.

4.2. Landscape loading

A realistic landscape model is rendered by combining color and height data. To do this, the application has to interpret the different image formats of the datasets and align the textures geographically. Without having to rebuild the application, experts should be able to add new datasets to the visualization. Although the mission area's geographical extent is known in advance, the AR application should allow experts to explore nearby areas, given the datasets are available. So the visualized part of a dataset is a subject of change.

4.2.1. Parameterization of terrain datasets

File format: The datasets describing the visualized area must be loaded onto the HoloLens to render the landscape. The image format of these texture data varies. The digital elevation model is saved as a GeoTIFF image, with each pixel describing a 32-bit elevation value. Satellite images, for example, can be saved in other image formats like Portable Network Graphics (PNG) or TIFF.

Covered area and spatial resolution: In our scenario, we consider an area of 15 x 15 kilometers to be observed. Digital elevation models, as well as aerial images, normally cover much larger areas, especially if the data comes from satellite measurements. An example is the *Shuttle Radar Topography Mission* carried out in February 2000. Almost 100% of the Earth's surface was surveyed during this operation with a spatial resolution of about 30 meters [28]. During this mission alone, the resulting digital elevation models were composed of more than ten terabytes of data [29]. At the same time, some datasets only cover a small area, but represent it in a high resolution [81]. The higher the resolution and covered area of a dataset, the larger the file size. As a result, the amount of data in each dataset varies depending on the covered area and the spatial resolution.

4.2.2. Dataset storage

As previously mentioned, dataset files can be several gigabytes in size. In order to render the landscape on the HoloLens, parts of the datasets describing the currently visualized area must be loaded in the HoloLens memory. Therefore, it is necessary to think about where to store these files and how these relevant parts are loaded to the main memory.

4.2.2.1. HoloLens' limited memory capacity

As explained in section 2.2.1.1, the HoloLens only has a comparatively small main memory capacity of 4 gigabytes in total, of which an application can allocate approximately 2 gigabytes. Often the size of one dataset already exceeds the memory limit several times [81].

The digital elevation model is a dataset that always has to be at least partially loaded in memory to display the landscape three-dimensionally. If satellite images or false color images should be projected onto the surface, parts of these datasets must also be loaded. Hence, memory management is a challenge while rendering large-scale landscapes on the HoloLens. Three different approaches of terrain data management are presented below.

4.2.2.2. Storing the datasets statically on the device

A simple approach would be to store the elevation and color texture directly on the HoloLens and load these at application runtime in main memory.

Since HoloLens applications can only allocate 2GB of memory simultaneously, the total size of the loaded datasets would have to be below this technical limit. A reduction of the dataset's spatial resolution reduces it's size, but also decreases the quality of the visualized model. In order to provide the best possible spatial context for the scientific analysis, the landscape model should be as detailed as possible.

Another way to reduce the dataset's size is to reduce the coverage area. However, the dataset's landscape region can only be condensed to the extent of the mission-relevant area. This region might still cover hundreds of square kilometers, depending on the mission. Furthermore, a downsized covered area limits the experts' ability to navigate freely on the virtual landscape surface, as neighboring areas are no longer accessible.

Storing the terrain data directly on the HoloLens harbors further disadvantages. Assuming experts want to add a new dataset, modify an existing dataset, or load other datasets to explore a different geographic region. In this case, the experts would need to replace the dataset files on each HoloLens.

Both the reduction in size and the maintenance of the stored datasets would require a lot of manual effort. Experts would be restricted in their application use due to low visualization quality or a limited navigation range. Although this approach is simple to apply, the disadvantages mentioned above outweigh the benefits . Therefore it is not suitable for our use case.

4.2.2.3. Storing the datasets on a network drive

In a second approach, the dataset files are stored on a network drive that the HoloLenses can access. This allows datasets to be added dynamically and modified much more easily than the previous approach, although with a latency due to transferring the data over network. However, the whole dataset files must be transferred into the main memory of the HoloLens. This means that even with this approach, the total size of the datasets is constrained by the limited memory capacity of the HoloLens. Thus storing the files on a network drive is also not a suitable solution.

4.2.2.4. Streaming the terrain data using a web map service

Accessing a web map service is another option. This service provides a standardized interface for streaming georeferenced terrain data [20]. A web map service is responsible for managing, storing, and pre-processing large georeferenced datasets. Such a service is a central component of many geographic information systems. The coupling between datasets and the application visualizing that data is loosened by using such a service.

A server acts as a data source, where applications can request the necessary terrain data in a standardized way and accordingly receive the data in a specified format. In a request-reply pattern, GIS applications communicate with web map services. A geospatial information system makes a request to the server using the *Representational State Transfer* (REST) interface, and the server then returns the requested landscape data [20]. This communication pattern is illustrated in figure 4.3.

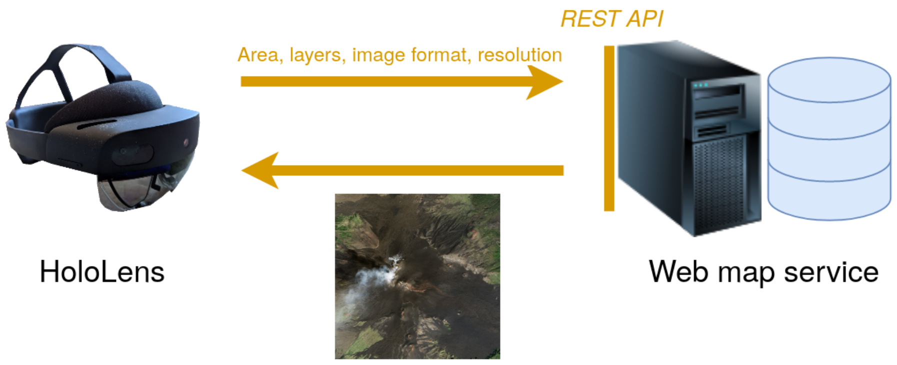


Figure 4.3.: GIS application running on HoloLens, requesting landscape data of a specified area.

Using a web map service may be a suitable approach for the use case in this work. Datasets can be managed dynamically and independently by the web map service. New datasets can be added, and existing ones can be modified or extended without rebuilding the HoloLens application. Instead of being stored on the HoloLens' limited flash storage space, the datasets are located on an external computer or server. Unlike the HoloLens' closed system these machines can be expanded in terms of both computing and storage capacity. This allows the integration of datasets beyond the HoloLens' storage limit of 64 gigabytes. Large datasets with detailed spatial resolution and a large covered area can be processed.

Web map services are not only a simple data source, but they enable pre-processing and aggregation of large datasets [20]. Depending on the parameters specified in the request, the server processes information from one or even several different datasets to consolidate them in the response [20]. A possible use case for this pre-processing step can be to merge different satellite images.

This aggregation of multiple datasets is depicted in figure 4.4. As explained previously, the datasets may be available in various image formats. The output image format is specified in the request to a web map service, and the service is responsible for converting the data. That way, additional computational effort can be avoided on the HoloLens' limited processing power.

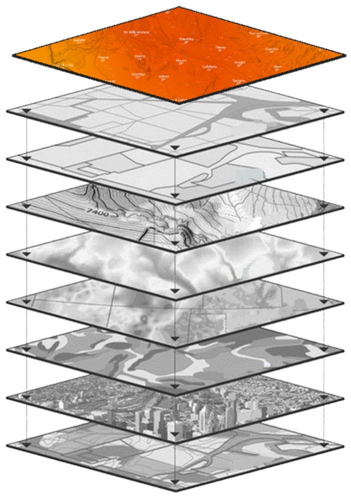


Figure 4.4.: Multiple datasets are aggregated by the web map service [136].

The desired terrain data resolution also needs to be specified in a web map service request. This allows applications to decide whether the full quality of the dataset or a reduced quality version is sent to the application [20]. That way, the application can balance dynamically between memory utilization and the quality of the received data.

Finally, each request has to define the area's dimensions to be covered. Using this spatial filtering allows applications to define an area of interest and thus only receive and process this area's landscape data. For this purpose, two geocoordinates spanning a bounding box of this area are attached as parameters to the request [20]. Only information about the landscape within this bounding box is sent back to the application by the server. This spatial filtering allows applications to reduce network load and memory consumption.

In particular, the possibility to request landscape data in variable resolutions and the option to filter large datasets to smaller areas are beneficial for memory management and navigation flexibility. The use of a web map service proves to be a suitable solution in our use case.

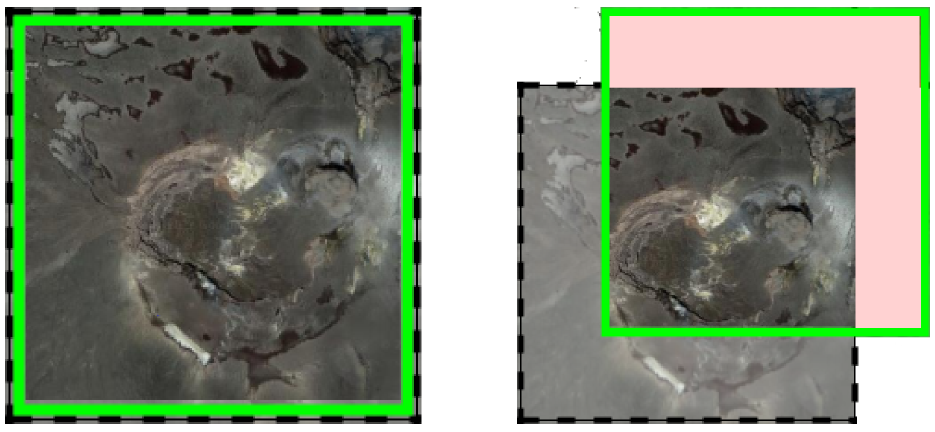
4.2.3. Terrain loading strategies

In order to allow the experts to navigate freely, it must be ensured that the relevant landscape data is loaded from the web map service into the main memory of the HoloLens at the right time. Both the amount of main memory used and the timing of the requested landscape data are relevant to enable seamless exploration of the landscape. Following, three different loading strategies are discussed.

4.2.3.1. Lazy loading

A *lazy loading* approach could be implemented. Following this approach, only the terrain data needed to display the currently relevant area is requested. Thus, at the start of the application, an initial request covering only the landscape area near the rovers is sent to the web map service. The received landscape data is loaded in main memory and is visualized. In figure 4.5a the data in memory is limited to the dotted rectangle. The currently visualized area is shown with a green border.

If an expert wants to explore the neighboring landscape during the analysis, the currently visualized area is shifted (see figure 4.5b). New areas of the landscape are now relevant for the visualization. These parts are shaded red in figure 4.5b. Since these areas are not covered in the data available on the HoloLens, the required data is requested from the web map server and then loaded into the HoloLens main memory for visualization. The new landscape information overrides the previous landscape data in the HoloLens memory. This approach makes it possible to dynamically change the displayed landscape section and its resolution.



(a) Only the currently visualized area (green border) is loaded as texture in memory.

(b) The currently visualized area (green border) is shifted. Landscape data shaded in red is not loaded in main memory and needs to be requested from the server.

Figure 4.5.: Lazy loading approach. The green border represents the currently visualized part of the landscape. The dotted rectangle represents the currently loaded landscape data in main memory.

A typical problem of such lazy loading approaches is delay in receiving the data over the network. The server needs time to process the request and to send the results back. Furthermore, there is also the risk that many requests are repeatedly sent to the server within a short time period. In our use case, this problem could occur, if the experts continuously shift the landscape section. Without considering this problem, the application would send a new request to the server every frame, if the currently visible landscape section has changed. This unnecessarily loads the network, the server and the HoloLens application.

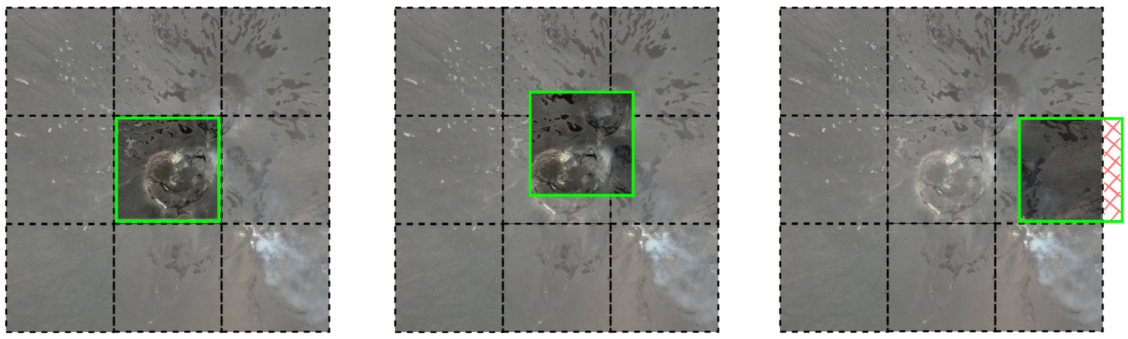
4.2.3.2. Eager loading

A contrasting approach to the lazy loading solution is the *eager loading*. The principle of this approach is to load as much data as possible in advance, even before it becomes relevant for the application. In our use case, the entire mission area and the neighboring areas would be queried at once from the server and sent to the application. This has the advantage that each HoloLens issues only one request to the server and receives one response from the server. However, this results in either increased main memory usage due to larger sections being loaded or reduced spatial resolution. This approach does not take full advantage of a web map service, which is providing terrain data of the exact area and resolution needed. Thus a pure eager-loading approach is not optimal as well.

4.2.3.3. Tiled loading

In this work, a compromise between the two opposing approaches of lazy and eager loading is used - the *tiled loading*. The basic idea of this approach is to load the currently relevant area together with its immediate surroundings (see figure 4.6a). The whole pre-loaded section is subdivided into uniformly sized rectangles. Each of the displayed rectangles is called a *tile*.

In contrast to the lazy loading, this approach does not issue a new request when the visible landscape section is slightly shifted (see figure 4.6b). The neighboring tiles are already loaded. In contrast to the eager loading not the whole mission-relevant area, but only the adjacent tiles are requested. This makes the tiled loading a lightweight and suitable approach.



(a) Reading landscape data of the rover's immediate vicinity

(b) Visible landscape area shifted by a quarter tile length

(c) Visible landscape area shifted by more than one tile length

Figure 4.6.: Tiled loading approach.

4.2.4. Browsing landscape data

Applying the tiled loading approach experts can explore the neighboring landscape without repeated reloading of landscape data. For this, an area of 3 x 3 tiles is requested from the server and held in main memory. Figure 4.6b shows how the center of the currently relevant terrain data is shifted when the landscape has been moved by a quarter tile length in any direction. It is not necessary to reload further landscape data from the server. The 3 x 3 tile grid allows the currently visualized landscape section to be moved by one tile length in all directions, with the terrain still being rendered correctly.

However, if the visible area is moved further than one tile length, parts of the landscape can not be displayed, because the terrain data in this area is not available. This scenario is shown in figure 4.6c. The area in which the data can not be sampled is shaded red. In order to display the shifted landscape correctly, the terrain data needs to be reloaded from the server. To do this, the tile that contains the center of the displayed landscape acts as the new central tile. The 3 x 3 tile grid surrounding that new center tile is requested from the web map service. Thus the initial situation, shown in figure 4.6a, is then restored.

Ideally, the server processes this request immediately and transmits the new landscape data while the expert is still navigating. Since the data from and to the HoloLens is transmitted via wireless LAN, the server may not be able to handle the request immediately, and the processing of the received data in the application also takes time. In practice, there is always a delay between the request and the availability of the new landscape data. This delay means that the new data may not be received and processed in time.

If tiles are not loaded in time, the landscape can not be rendered correctly (see figure 4.6c). This is particularly noticeable, if the expert moves the landscape section at a high velocity. Therefore, new landscape data is requested from the server already after a shift of half a tile length. Although this increases the network traffic, the new tiles can be processed in time.

4.2.5. Zooming into the landscape

Because the application requests a 3 x 3 tile area, the dimensions of the textures received from the web map service are always three tiles long. However, since the application is intended to allow experts to change the scale of the landscape in order to zoom in or out, this tile length may change. When an expert zooms into the landscape, the size of the area to be displayed decreases. When zooming out of the landscape, the size increases. At the same time, the spatial resolution should be improved with increasing zoom level to present the landscape in more detail.

Similar to the reloading of the neighboring tiles, it is important to consider at which zoom level new data should be requested from the server. This new data is provided with the according spatial resolution.

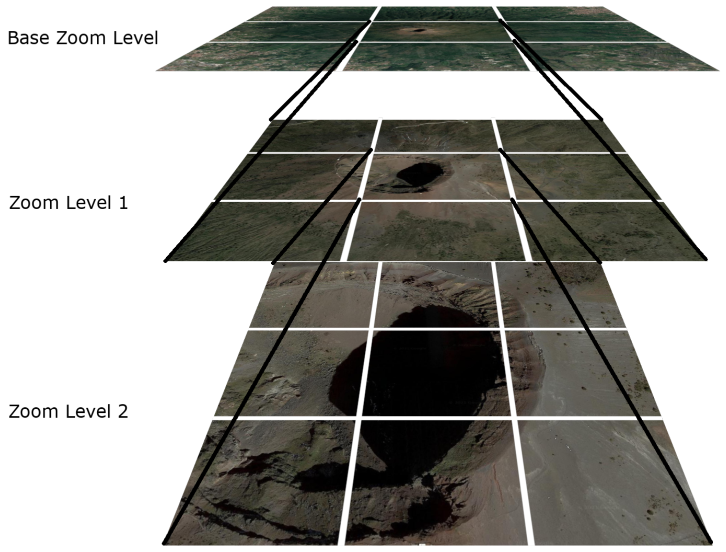


Figure 4.7.: Hierarchical tiled zoom level pyramid.

The idea of zoom levels is shown in figure 4.7 and is based on the concept of the image pyramid. Each layer in this pyramid represents a discrete zoom level. In contrast to the classical image pyramid, the tile length is reduced to one third from the upper to the lower zoom level.

The following equation calculates the zoom level z for a given zoom factor f:

$$z = \lceil \frac{\ln f}{\ln 3} \rceil \tag{4.2}$$

A new request is sent to the web map server when an expert zooms to the next level. In-between zoom levels, no additional data is requested from the server. However, the texture coordinates c used to sample the landscape textures are scaled relative to the texture center coordinate $(0.5, 0.5)$, according to the current zoom factor.

$$c' = ((c + 0.5) * z) - 0.5 \tag{4.3}$$

4.3. User-Landscape interaction

In order to enable the experts to navigate the displayed landscape intuitively, a user interface suitable for the use case has to be developed. As mentioned in section 3.3.2, the evaluation of user interfaces in AR is an active research area. For our use case, we decided to develop a tangible user interface [70] with direct manipulation [133] following Microsoft's guidelines for direct manipulation with hands [90]. The user should interact with virtual content directly by grabbing it with his or her hands. According to Satriadi et al., different design factors have to be considered when designing freehand interactions [128]. Two of these factors are discussed below.

Input mapping: Satriadi et al. define input mapping as the function that maps the user's hand movements to interactions with virtual objects [128]. Two main categories of such functions are distinguished: position-based mapping and rate-based mapping. In position-based mapping, the change of the hand position is directly proportional to the action performed on the virtual element. An example is the direct hand manipulation seen in figure 2.3. The second type of mapping function is rate-based mapping. Here, not the movement but the hand position influences the virtual object. An example of such a mapping is the joystick on a computer.

For our application, a position-based mapping is used. Since it is to be expected that the experts interact with the landscape representation and other virtual content rather over closer distances.

Fatigue: When implementing an interaction concept with mid-air gestures and motions, the user's fatigue plays a central role. This occurs especially when the interaction concept requires the user to keep his arms stretched in the air for an interaction gesture over an extended time period. This effect is referred to as the *gorilla arm effect* [13].

To maximize the time to fatigue, the landscape model is shown at approximately the elbow height of the experts. Studies have shown that hand interactions at this height significantly reduce the gorilla arm effect [64].

The input actions used in this work are guided by the findings of Austin et al., which are explained in section 3.3.2. With the help of three different interactions, experts can change their view on the landscape and navigate over it. These three actions are described in the following.

4.3.1. Rotating

The rotation gesture is used to rotate the entire representation of the landscape around its own axis. Austin et al. showed that most users prefer a wheel metaphor to control the rotation angle when navigating map data [8]. Therefore, such a metaphor is also used in this work. The principle of this input action can be seen in figure 4.8.

Shown is the circular rim of the landscape representation from the top-down perspective. To rotate the landscape, an expert grasps this rim at point A and moves the hand to point B. To determine the rotation angle from this movement, the hand positions are first transformed from the world coordinate system into the coordinate system of the landscape model. The landscape is to be rotated around the z-axis (up-vector). Therefore, only the X and Y components of the points A and B are considered. To determine the rotation angle α, the difference of the angles of the starting point A and the current hand position B relative to the x-axis (highlighted in red) is calculated:

$$\alpha = (\text{atan2}(a_y, a_x) - \text{atan2}(b_y, b_x)) * \frac{180}{\pi} \tag{4.4}$$

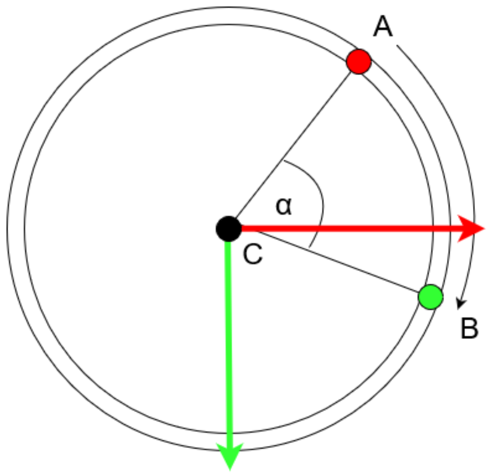


Figure 4.8.: Rotation input.

4.3.2. Panning

To explore the neighboring landscape, the panning gesture illustrated in figure 4.9 is used. Again, the decision to use this gesture is based on Austin et al. observations of navigation in augmented reality maps [8]. The expert grasps the landscape at a point A with his hand and moves it parallel to the model's surface to point B. The landscape behaves according to the principles of tangible user interfaces, as a real object would behave, if it was grasped and dragged.

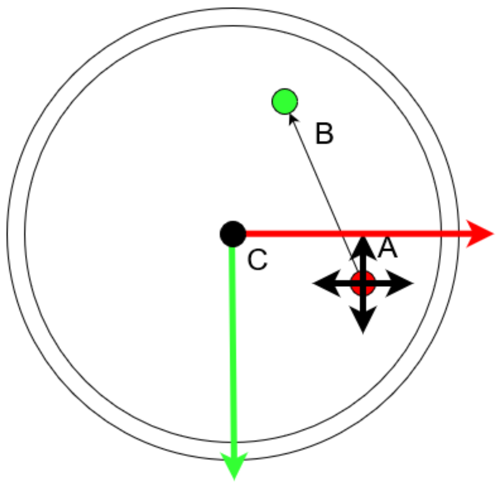


Figure 4.9.: Panning input.

As explained in the previous sections, the landscape rendering is based on the idea that the landscape model is a uniform vertex grid. Each vertex in the uniform grid represents a specific geographic location. By shifting the texture coordinate of a vertex, the mapping between the geographic location and the vertex is changed. If a vertex texture coordinate is altered, the vertex represents a different geographical point on the surface. This effect is visible in figure 4.10.

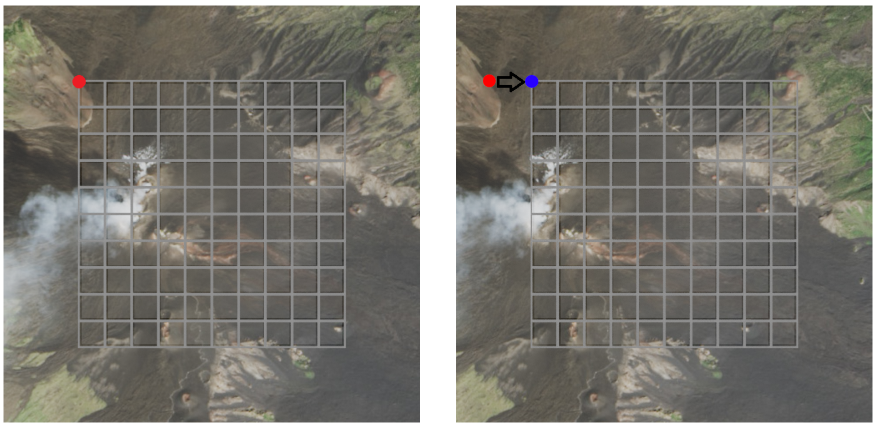

(a) Without texture coordinate offset (b) With a texture coordinate offset causing a pan to the right

Figure 4.10.: Mapping between vertex grid and terrain texture.

In figure 4.10a, a mapping between the vertex grid and the landscape data is illustrated by superimposing the 2D grid on a satellite image. Each vertex is mapped to a specific texture coordinate. In figure 4.10b, the same vertex grid with the same vertex positions is shown, but all vertices' texture coordinates are shifted to the right. Each vertex is mapped to a different geographical location.

Using a texture coordinate offset, the visible part of the landscape can be shifted. To determine this offset in the texture coordinate space, first the starting point A and the current hand position B are transformed from the world coordinate system into the local coordinate system of the landscape. Then the displacement vector $\vec{d}$ in X and Y direction from point A to point B is calculated. This corresponds to the displacement of the two hand positions in Unreal Engine units.

$$\vec{d} = (B_x - A_x, B_y - A_y) \tag{4.5}$$

To calculate the real displacement $\vec{d}'$ of the landscape in meters, this vector is divided by the current scaling factor s (see section 4.1.1).

$$\vec{d}' = \frac{\vec{d}}{s} \tag{4.6}$$

Since the tile length t in meters is known, the displacement vector can be expressed relative to the tile length.

$$\vec{d''} = \frac{\vec{d'}}{t} \tag{4.7}$$

As explained in section 4.2.4, the textures received from the map server are three tile lengths long in both dimensions. To calculate the displacement of the texture coordinate relative to the total size of the landscape textures, the displacement vector is multiplied by a factor of 3.

If the landscape has already been shifted, the previous displacement vector $\vec{d_{prev}}$ is added to the currently calculated offset.

$$\vec{d'''} = \vec{d''} * 3 + \vec{d_{prev}} \tag{4.8}$$

Overall, the texture coordinate offset can be calculated as follows:

$$\vec{d'''} = \frac{(B_x - A_x, B_y - A_y)}{t * s} * 3 + \vec{d_{prev}} \tag{4.9}$$

As explained in section 4.2.4, new landscape data is requested, if the landscape is moved by more than half a tile length. Once this new data is received and processed, the displacement vector is recalculated relative to the new center tile.

4.3.3. Zooming

According to Austin et al., a *pinch to zoom* metaphor is preferred for zooming in or out of an AR map visualization [8]. The user grips the landscape and can control the zoom level by changing the distance between the hands. The model is then scaled around the point midway between the user's two hands. While it was initially planned to use this input metaphor for zooming, an easier to implement user input is used, due to the limited time frame of this thesis. Instead the user can select the current zoom factor relative to the base zoom level in a range from 0.1 to 10 using a slider.

4.4. Rover data processing

This section explains the general ideas of the processing and visualization of rover data.

4.4.1. Retrieving metadata

One of the goals of this work is to provide the experts with a spatial overview of the scientific measurements collected by rovers. For this purpose, the landscape representation described in the previous section is used to present the measurement locations to the experts interactively.

Before explaining the details of the data representation, we first consider the provenance of the data. In a realistic scenario, the rover's scientific instruments collect the data. The rover then sends this data to a receiving station, which catalogues the data, adds metadata, and then stores it.

To visualize the scientific results on the HoloLens, the corresponding data needs to be loaded into its memory. Similar to the terrain data, the rover data should be streamed at runtime from a server to the HoloLens. That way, new rover data can be added without having to deploy the application again.

Throughout an entire mission, a rover can collect a total of several terabytes of data. Managing those large amounts of data would be too demanding for a desktop computer, let alone HoloLens. In order to use the limited memory of the HMD efficiently, the rover data is loaded on demand. That way, only the scientific data currently inspected by the experts is streamed over network into the memory of each HoloLens.

To highlight the locations of the scientific activities on the landscape model, the metadata of these measurements are queried by the HoloLens from a web server at the application startup. This metadata includes information about the category, time of acquisition and precise description of the position at which the rover conducted the scientific measurement. An overview of the various types of metadata transmitted is shown in figure 4.11.

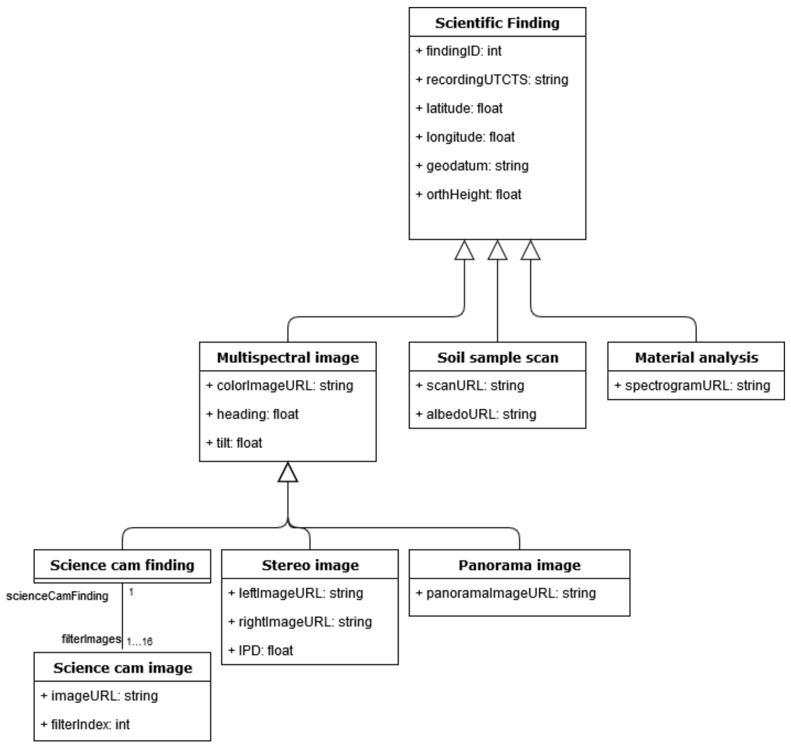


Figure 4.11.: Scientific finding categories and metadata.

Each scientific activity is modeled as a *Scientific Finding*. In addition to an identification number, recording time, the geographical position and the geodetic datum used are specified for each finding. Then, depending on the type, a distinction is made between a multispectral image, 3D scan and material analysis. The actual scientific data of a finding can be queried from a server using the links provided in the metadata. The individual type-specific details are discussed in depth in section 4.4.4.

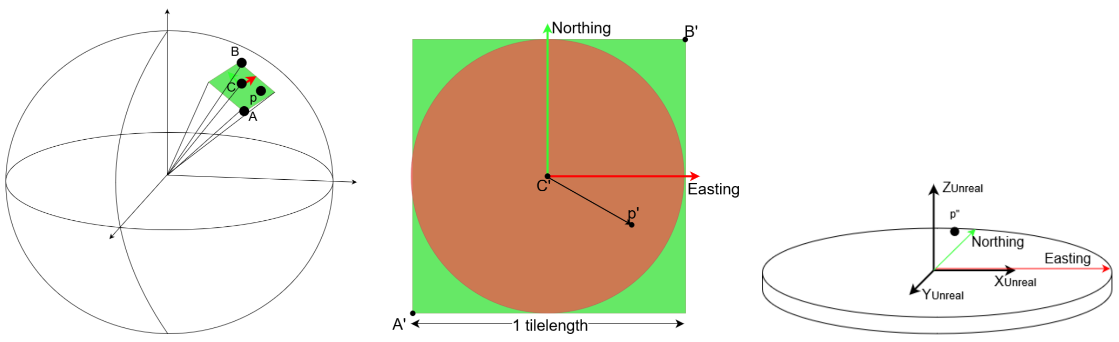
4.4.2. Geocoordinate transformations

To highlight the locations of the scientific activities with virtual markers on the landscape model, the geocoordinate specified in the metadata needs to be transformed into the Unreal Engine coordinate system. The goal is to determine at which 3D coordinate relative to the center of the landscape model the marker is to be placed. As shown in figure 4.11, the position of each finding is given in a latitude, longitude and height format, and the geocoordinate system is defined with the geodetic datum.

This geocoordinate has to be transformed into the Cartesian Unreal Engine coordinate system to show the markers at the correct position. For this purpose, the following parameters of the currently displayed landscape area S are known in advance:

- The geocoordinate C, the center point of S.
- The two corner geocoordinates A and B spanning this section S.
- The world in miniature scaling factor s describing the relation between Unreal Engine units and meters on the landscape.

In our case, it is assumed that the dimensions of this area S are small compared to the length of the ellipsoid's semi-axis. This means that the curvature of the surface can be ignored for coordinate transformation and thus the landscape area can be considered as a tangential rectangle on the surface. This rectangle is highlighted in green in figure 4.12a.



(a) Landscape section in lat/lon format (b) Map projection of the visible landscape tile (c) 3D Unreal Engine coordinate system

Figure 4.12.: Geocoordinate transformation.

A geocoordinate $\vec{p}$ within this area is projected into grid coordinates p' using a map projection (see section 2.4.1.3). The projection is designed that the center C is the origin and point A and B are projected half a tile length away from the center in northings and eastings, respectively.

$$\vec{p'} = project(A, B, C, \vec{p}) \tag{4.10}$$

These grid coordinates describe the displacement in meters to the north and east relative to the center C. As shown in figure 4.12b, the grid coordinates are specified in a right-handed coordinate system. Unreal Engine internally uses a left-handed coordinate system. Therefore, the Y-component of the grid coordinate is inverted.

The Z component, i.e. the *height*, is specified directly in meters in the metadata of the finding. Since the landscape is displayed in the world in miniature format in front of the users, the calculated 3D coordinate is scaled by the scaling factor s.

$$\vec{p''} = \left(\vec{p'} * \begin{bmatrix} 1 \\ -1 \\ 1 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ height \end{bmatrix} \right) * s \tag{4.11}$$

With this transformation every geocoordinate can be highlighted with a marker at the correct position $\vec{p''}$ on the landscape model. If the current landscape section changes, either by an expert moving the landscape or by changing the zoom level, the position of the marker is recalculated and adjusted accordingly. If the grid coordinate is projected outside the currently rendered area (highlighted in red in figure 4.12b), this marker will not be rendered.

4.4.3. Marker interaction

The goal is to use the landscape representation as a central visual component to spatially contextualize the data acquired by the rover. The locations where a rover collected scientific data are highlighted.

In the context of this work, a pin metaphor is used to represent such a marker. This metaphor is a known concept to users of classic two-dimensional digital maps, such as Bing Maps or Google Earth [88, 57]. It is clear to the user that such a pin marker represents a specific location on a map. By selecting the marker, further information is displayed to the user. The same ideas behind these actions are applied to AR in this prototype. Instead of a two-dimensional pin, a simple three-dimensional model is displayed. Depending on the category of the finding, the marker is colored differently. The distinct colors of these pins are shown in figure 4.13.

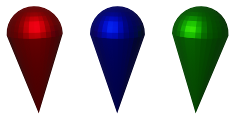


Figure 4.13.: 3D marker model in different colors: multispectral image (red), 3D scan (blue), material analysis (green).

The tangible user interface paradigm is a primary principle for interacting with virtual elements in the presented prototype. This principle is also applied to the user's interaction with the marker. According to Microsoft, there are five different approaches to selecting virtual elements [96]. These approaches have already been described in section 2.2.2.

In a practical test, the eye-tracking sensor turned out to be too inaccurate to reliably select a specific marker. It also indicated that targeting a marker with the head gaze requires attention and a calm head movement. Therefore, the gaze and dwell and gaze and commit selection methods were excluded. Also, selection via voice commands can not be used in our application. Several experts are in the same room. If an expert wanted to control actions on his HoloLens via voice commands, these commands would also be recognized by the other HoloLenses. Such a user interface would lead to confusion, unintended actions, and interference with the discussion in practical use.

The qualifying interaction methods therefore are direct manipulation with hands and point and commit with hands. These are used in the prototype to select the markers. An expert can either grab a virtual marker with his hands or, if it is not within reach, select it with point and commit. To avoid accidentally selecting markers, the user has to lift the marker over a defined distance to select it. The gesture to select a marker is illustrated in figure 4.14.

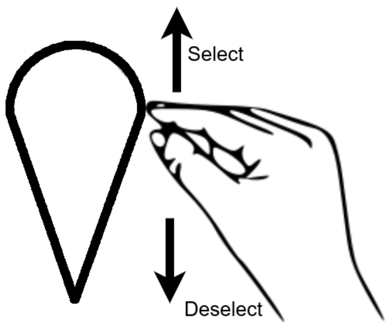


Figure 4.14.: Selecting and deselecting a marker using direct hand manipulation.

According to the basic principles of direct manipulation [133], every action must be reversible. This means that a selected marker should be deselected again with an opposite interaction gesture. Following this principle, the user has to grab a marker as a reverse action and move it down by a defined distance to deselect it (see figure 4.14).

4.4.4. Visualization

One goal of this work is to display the data acquired by the rover in AR. As already explained in section 2.6.2, the collected data of the LRUs of the ARCHES project can be roughly divided into three different categories: multispectral images from the science cam unit, 3D reconstructions and material analyses of the LIBS spectroscopy. The basic ideas to visualize these different data types are explained below.

4.4.4.1. Multispectral images

In the context of the ARCHES project, a multispectral image is a series of photographs taken with the camera pointed at the same subject but with different wavelength filters. Only a distinct part of the light spectrum can reach the sensor, depending on the selected filter. Thus, a multispectral image can be considered a three-dimensional texture array. The first and second parameters are width and height, and the third coordinate is the filter index.

These multispectral images are represented in different formats for visualization. Several images from the same position but with a different tilt and orientation of the camera can be combined into a panoramic image. Using the advantages of AR, these panoramic images should be visualized within the demonstrator.

As already explained in section 2.6.2, the multispectral images are captured using two cameras, each equipped with a filter wheel containing nine filters. Three of these nine filters cover the same wavelength range. Selecting one of these three filters on both cameras allows the rover to take stereoscopic images using the science cameras.

In total, three subcategories of these multispectral images are to be visualized in the application: mono, stereo and panoramic images.

Mono multispectral image: This thesis assumes that a mono multispectral image is a set of 18 images, each taken with a different wavelength filter.

If an expert selects the marker of a multispectral image, the first image with the filter index 0 is shown to the expert on a projection surface. This projection surface acts as a virtual monitor and is freely placeable in the air. The expert can then select a filter via a slider, and the corresponding image taken with this filter is loaded from the texture array and displayed on the projection surface.

Simply displaying the photo as a two-dimensional image does not take full advantage of the possibilities offered by AR. Several approaches are used to emphasize the spatial context of the image, which are presented below.

A line connecting the projection surface and the marker on the landscape model indicates to the user where the multispectral photo was taken.

The extrinsic camera parameters can be derived by processing the exact position, heading, and tilt of the camera at acquisition time from the finding metadata. The intrinsic parameters of the camera are static and known to the application. Since both the extrinsic and intrinsic camera parameters of the real rover camera can be determined in the application, the camera can be modeled as a virtual camera with sufficient accuracy. Displaying the virtual camera's frustum, the orientation and it's field of view is presented to the experts. The virtual camera makes it possible to project virtual scene objects into the existing captured image with the correct perspective.

Further information is augmented into the multispectral image. The idea of this augmentation is shown in figure 4.15.

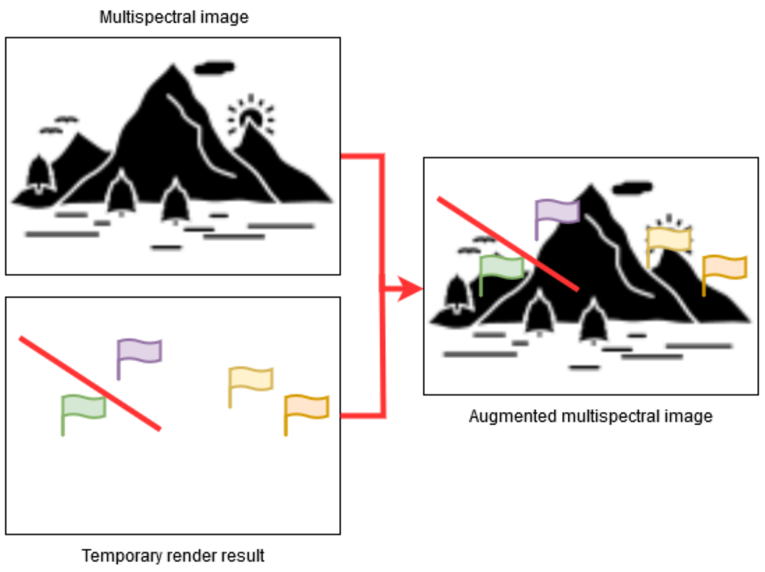
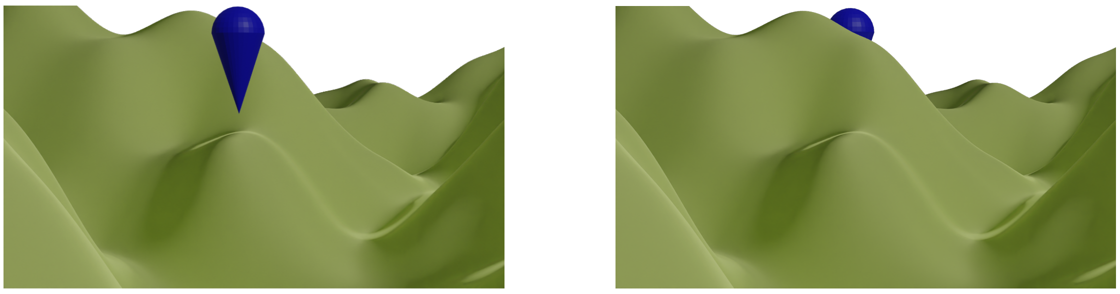


Figure 4.15.: Multispectral image blended together with virtual annotations.

Specifically, the markers of other scientific activities visible in the photo, virtual laser pointers and annotations placed by the experts on the landscape are projected into the image. The virtual camera's image is written into a temporary render texture and merged with the photo content.

While rendering annotations from the virtual camera's perspective, objects obscured by the landscape should not be visible in the augmented image. This problem is illustrated in figure 4.16.



(a) Marker visible in front of the landscape.

(b) Marker is correctly occluded by the landscape.

Figure 4.16.: Rendering of occluded virtual content.

The blue marker is behind the landscape from the perspective of the recording camera. If the occluded markers were also displayed, then it would not be possible to determine in the two-dimensional image whether the marker is in front of or behind the landscape (see figure 4.16a). Therefore, only the annotations not occluded by the landscape model are augmented in the image (see figure 4.16b).

The overlay of these additional annotations can be activated and deactivated via the user interface, if preferred by the experts. Thus, a link is established between the two-dimensional multispectral image and the three-dimensional representation of the landscape and its associated annotations.

Another approach to increase spatial awareness is the idea of linking between the image content and the landscape model. Suppose an expert finds an interesting feature in a multispectral image, he wants to locate that feature on the landscape model. This scenario is depicted in figure 4.17. A certain area of the multispectral image is marked in red by an expert (figure 4.17a). Starting from the position of the virtual camera, a ray is projected into the scene through this pixel (figure 4.17b).

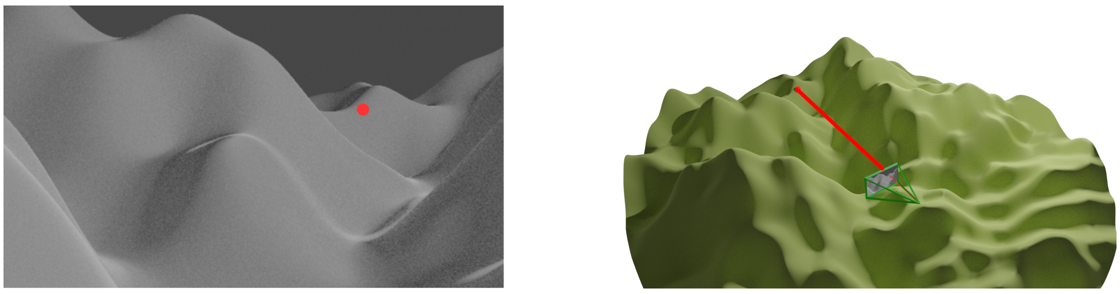

(a) Annotated spectral image (b) Annotation projected onto landscape model

Figure 4.17.: Projection of marked pixels into 3D landscape.

Stereo multispectral image: Another type of multispectral recordings are stereo images. Both LRU 1 and LRU 2 are equipped with a stereo navigation camera. In addition, LRU 1 can capture stereo images in the multispectral range using its science cam unit. These stereo images, similar to the mono multispectral images, will be analyzed in AR by the experts. The stereo image is composed of two images taken by cameras that have a defined spatial distance and parallel optical axes to capture the same motive, such as a landscape or a particular soil sample.

Similar to the display of a mono multispectral image, the stereo images are presented on a freely placeable virtual monitor.

In stereo image, extrinsic and intrinsic camera parameters of both cameras can also be determined in the application. This makes it possible to apply the same concepts as for mono multispectral images to augment stereo multispectral images with annotations and markers. Depending on whether the right or left perspective is currently displayed on the virtual monitor, the virtual camera is slightly shifted according to the distance of the rover camera sensors.

Previous research has shown that depth cues are perceived stronger on stereo images displayed three-dimensionally in AR than on 3D capable monitors [76]. It therefore makes sense to give the experts the option of viewing the stereo images captured by the rover in stereo. The experts can switch between monoscopic display of the left and right images individually and stereoscopic display of the two images combined.

To perceive the multispectral image in stereo, two different images are displayed on the virtual monitor depending on the eye. While rendering the scene for the left eye, the left multispectral image is displayed; for the right eye, the right image is visualized. This setup is shown in figure 4.18. On a projection surface (black border), two images are displayed, one red and one blue.

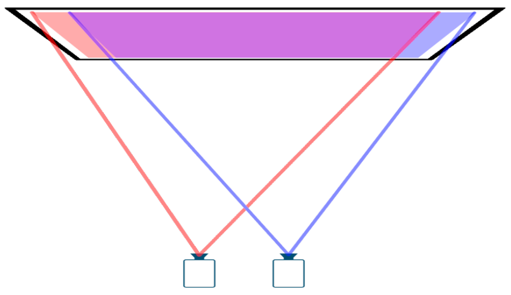


Figure 4.18.: A stereo image composed of two images (red and blue) is visible on a virtual screen. The left eye only sees the red image and the right eye only sees the blue image. Both images are shifted using *horizontal image translation*.

These two colors represent the left and right perspectives of the stereo image. Both images are slightly shifted horizontally and do not overlap exactly. This shift is called *horizontal image translation* (HIT) [15]. By varying the size of this displacement, depth ques in the image can be made less or more pronounced. If the depth differences are emphasized, they are perceived more clearly.

Multispectral panorama: The third category of multispectral images are panorama photos. The rover acquires these panoramic images by taking mono multispectral images in all directions from a fixed position. The camera system's direction and tilt are adjusted using the pan/tilt unit in each shot. Such a 360 degree panorama is composed of a mosaic of 20 x 3 images in the case of the LRU 1 [130]. This work assumes that the individual images have already been stitched together in a pre-processing step. This panoramic image describes the environment spherically around the rover. From the rover's perspective, the environment appears like a sphere with the camera in it's center. To describe surface properties of a spherical object on a two-dimensional image, the sphere needs to be projected. The *equirectangular projection* [135], a type of map projection, (see section 2.4.1.3), is frequently employed. Figure 4.19 shows an example of such a panoramic photograph in this format.

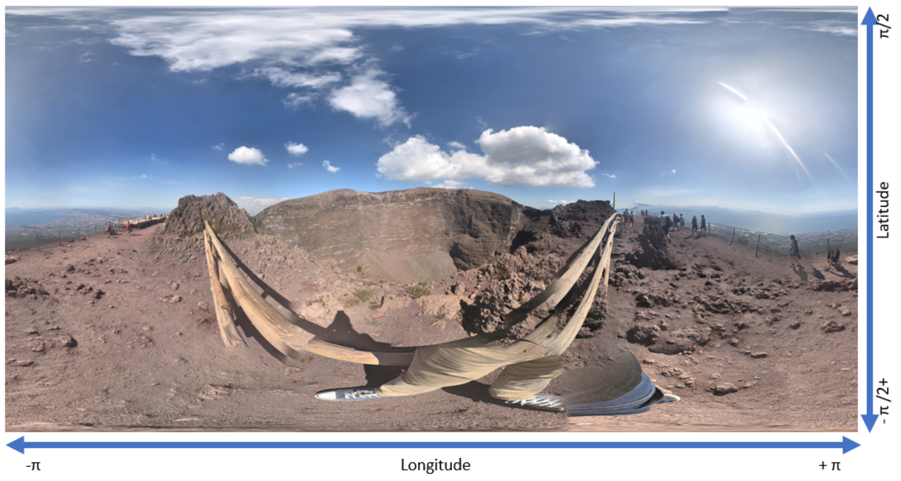


Figure 4.19.: Equirectangular panorama describing a spherical environment. Pixels on the same row are equal in their latitude. Pixel in the same column are equal in their longitude.

The surface of a spherical object is described as a flat two-dimensional image in this format. Meridians are mapped to vertical lines with constant spacing and parallels to horizontal lines. As a result, all pixels in a row have the same latitude and all pixels in a column have the same longitude.

The panoramic images are assumed to be available to the application in equirectangular format. These panoramic images have to be rendered in AR and can be presented using two distinct visualization approaches.

Full photosphere visualization: The photosphere is a standard method for visualizing a panoramic image immersively [67]. For this purpose, a sphere is placed and kept centrally around the user. Then the panorama image is displayed on the inside of the sphere's surface so that the viewer can see it.

To display the image content at the correct position on the sphere surface, the texture coordinates for the panoramic image needs to be calculated. The Cartesian coordinates of a point p on the sphere surface are converted into latitude and longitude format (see section 2.4.1.2). As shown in figure 4.19, the width of an equirectangular image maps longitude values from $-\pi$ to $+\pi$. The height describes the latitude between $-\frac{\pi}{2}$ and $+\frac{\pi}{2}$

By dividing the calculated latitude value by π and the longitude by $2 * \pi$, the texture coordinate t belonging to the point p is determined, and the color value can be sampled.

$$t = (\frac{lon}{2 * \pi}, \frac{lat}{\pi}) \tag{4.12}$$

Flashlight panorama photosphere visualization: Visualizing the panoramic image as a photosphere causes the user's real surroundings to be blended over by the panoramic images. This effect is sometimes so strong on the HoloLens that the real environment is no longer visible to the user. On the one hand, this has the advantage that the image content of the panoramic image is clearly visible. On the other hand, especially in a co-located collaborative session, this can lead to a loss of reference to the user's real surroundings and thus impair the discussion among the experts. In particular, the integration of the real environment is a unique feature of AR applications in contrast to VR. This aspect is lost through the full superimposition of a virtual environment in the form of a photosphere.

Therefore, an alternative approach to the classic photosphere is presented in this application. This approach is based on the findings of Ridel et al. [123]. With a flashlight metaphor the behavior of a virtual flashlight controlled by the user is simulated. Just like a real flashlight the virtual flashlight is used to explore specific areas of the panoramic image. Following the principles of tangible user interfaces, the user controls a virtual flashlight's position and beam direction with his hand.

Only the parts of the photosphere within the light cone are visible. The rest of the photosphere is black, which is rendered transparent on the HoloLens displays. This allows the virtual environment to be explored in the photosphere while still perceiving the real environment during a collaborative session. Each expert controls his or her flashlight and can illuminate different areas to draw the attention of the other experts to something specific.

Depending on the use case and the expert, there are different requirements for the virtual flashlight. Sometimes a broader light cone is more efficient, for example when searching for a specific feature in a panoramic image. In other cases, a shallower light cone may be more beneficial, for example, to point at something specific or when the user needs to focus on the real environment. Therefore, experts should be able to change the focus of the virtual flashlight dynamically.

Many physical flashlights have a lens on the front that can be turned to vary the focus. This input metaphor is transferred to the virtual flashlight and is shown in figure 4.20. By rotating the light along its own axis, the light cone angle is adjusted.

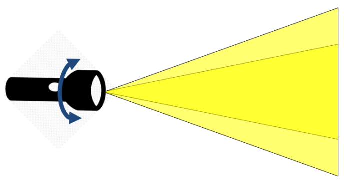


Figure 4.20.: Rotating the virtual flashlight allows to change the focus.

Other virtual elements, like markers and annotations of the experts, should also be blended into the panoramic image at the correct position. The virtual scene is rendered and merged with the recorded panorama image using a virtual 360 degree camera.

4.4.4.2. 3D Scans

The 3D scans of objects recorded by the rover are predestined to be visualized immersively in three-dimensional AR.

The model is loaded from a server into the HoloLens main memory. This model is then displayed floating in the air above the landscape model in front of the experts. If a color texture (albedo texture) is specified in the metadata of this finding, the texture is downloaded from a server and then used to color the scan model. It is assumed that a suitable UV layout has already been created for the 3D scan in a pre-processing step, so that the model can be textured correctly. Further, it is assumed that the coordinate system of this 3D scan is aligned with a geocoordinate system.

This model is displayed in front of the experts and can be inspected from all sides. By grabbing the model with the hands, the position and orientation can be changed in six degrees of freedom. The principle of tangible user interfaces is also applied here for interaction with the model. The virtual model behaves like a real physical object if picked up and moved, except that the model remains suspended in the air after the interaction is complete. By grasping the model with two hands and then stretching the hands apart or bringing them together, the scan can be enlarged or scaled down. Depending on the provided color texture, either the actual material color reconstructed from photos, or other scientific evaluations in the form of false color images can be displayed on the reconstruction's surface.

4.4.4.3. Material analysis

A material analysis is performed to investigate the chemical composition of a soil sample taken by the rover. The LRU 2 rover is equipped with a LIBS spectrometer. The data collected by the spectrometer is displayed as a spectrum in a diagram. Figure 4.21 shows an example of such a spectrum.

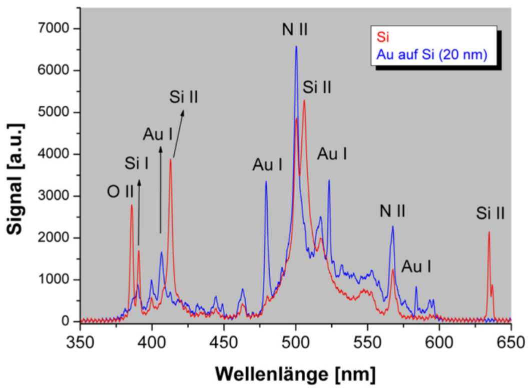


Figure 4.21.: Typical LIBS spectrum [124].

The plotted graph is assumed to be available to the application as a high-resolution image. This diagram is displayed similar to the multispectral images on a virtual screen that can be placed freely in space. Experts can analyze the characteristic atomic emission lines located at different wavelengths. That way the elementary composition can be determined from the peaks in the spectrum [124]

Given complete raw data, the spectrum could additionally be plotted interactively in the AR application.

4.5. Multi-user collaboration

In this thesis, we consider a scenario in which multiple experts work together collaboratively. All experts are located in the same real and the same virtual environment.

Since the representation of the landscape as a tabletop model is a central part of the discussion, it has to be ensured that all parameters of this model are the same for all experts. If this is not the case, some experts see the landscape represented differently, for example, shifted or zoomed in. This leads to ambiguity and confusion in the discussion. To keep the model synchronized, all landscape-related interactions including panning, rotating, and zooming are replicated and displayed to all participants.

Interactions with the rover data or marker are also synchronized in a similar fashion. All synchronized data-related interactions are listed in table A.1.

4.5.1. Virtual-real world registration

Interpreting table A.1 shows that the most frequently synchronized property is the 3D transformation of the virtual elements. During a discussion, experts should share a common unified view on the presented data. Every expert should see these actors in the same physical location in the real environment. Hence, the 3D transformation of these actors needs to be synchronized across all participants of a virtual session.

However, it is not possible to directly apply the transformation of a virtual element from one HoloLens to another. Each HoloLens, even if they are in the same room, has a different coordinate system. This coordinate system is reinitialized every time a HoloLens application is restarted. Therefore, the global coordinate systems of the participants are always different. This issue is visualized in figure 4.22. A transformation matrix T_{AB} is needed to transfer a point $\vec{p_A}$ relative to a coordinate system A into another coordinate system B.

$$\vec{p_B} = T_{AB} * \vec{p_A} \tag{4.13}$$

This transformation matrix describes the transition between the coordinate system on HoloLens A and the coordinate system on HoloLens B. Since HoloLens A does not know the coordinate system of HoloLens B, T_{AB} can not be determined directly.

One way to solve this problem is to define a common point and direction in the real environment. When both HoloLens applications recognize these features, an intermedi-

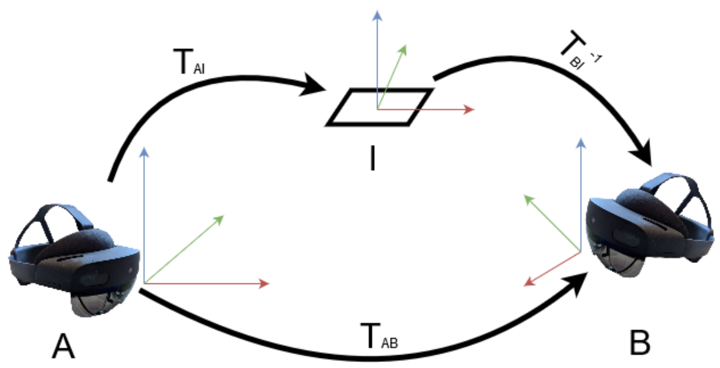


Figure 4.22.: Multiple HoloLens are initialized with different coordinate systems. In order to transform a point from one HoloLens coordinate system to another an intermediate coordinate system is used

ate coordinate system I can be constructed. Since I is located in the same place relative to the real environment for all participants, transformations to be exchanged can be expressed relative to this common coordinate system. The transformation between the coordinate systems of two HoloLenses can be divided into two single transformations using the marker coordinate system:

$$T_{AB} = T_{AI} * T_{BI}^{-1} \rightarrow \vec{p_B} = T_{AI} * T_{BI}^{-1} * \vec{p_A} \qquad (4.14)$$

There are several ways to register this common point in the real environment. One option is to manually mark prominent points of the environment in the application, like a pattern in a carpet or a table corner. However, this type of registration can be cumbersome, depending on the accuracy requirement.

In addition to manual approaches, automated approaches also exist. *Spatial anchors* are an example of such an automated approach [100]. A spatial anchor is a descriptor of a prominent feature in the environment detected by the HoloLens. To describe this feature, the HMDs uses color images from the cameras as well as the reconstructed environment of the user. The concept of spatial anchors is based on the idea that other HoloLenses can recognize the color and shape pattern in the same environment [100]. Depending on the spatial anchor descriptor, larger parts of the spatial reconstruction need to be sent over network to all HoloLenses in the session, which increases the preparation time. Concluding, the recognition of the features using spatial anchors was not reliable and not fast enough for our use case.

An additional approach could be the, automatic registration by QR code. A printed copy is mounted in the room on a fixed location. The QR code is then recognized using the HoloLens front camera. Since a QR code is uniquely oriented, the marker coordinate system can be derived directly using its orientation and position. To send a position $\vec{p_A}$ from HoloLens A to other participants of the virtual session, $\vec{p_A}$ is transformed into the marker coordinate system using T_{AI} and then sent over the network.

$$T_{BI} * \vec{p_B} = T_{AI} * \vec{p_A} \tag{4.15}$$

The other participants in the session, in this case HoloLens B, transform the received position $\vec{p_I}$ into their global coordinate system using the inverse transformation T_{BI}^{-1}. This ensures that all participants see these virtual elements with the same position and orientation in the real environment.

Using a QR code to determine a common coordinate system turned out to be a simple, robust and reliable approach.

4.5.2. Shared and local workspaces

The implementation of multi-user collaboration is considered a vital aspect of this work. The goal is that all experts can view and discuss the data recorded by the rover. However, based on the findings of Lee et al. and Lee and Hua, it has already been observed in the past that conflicts can arise between the participants, if they pursue different goals [80, 79]. Particularly in the context of space missions, the experts are interdisciplinary and have different approaches to analyzing the data. One participant's actions interfering with those of the other participants, leads to unnecessary delay and frustration. Interference could occur, when several experts analyze a dataset together, but one expert, in particular, wants to examine a different part of the landscape.

A trivial way for the expert to solve this problem is to disconnect from the virtual session and open a separate session just for himself. However, this has the disadvantage that the user completely excludes himself from the discussion. Furthermore, it is also inconvenient from the system's point of view, since both rover and terrain data have to be requested again from the server.

Another option is to limit each session to only one participant interacting with the data and landscape model. All the other participants are exclusively passive spectators. With the option for the active participant to give control to passive participants, this is similar to an approach often used in video conferences with screen sharing [99]. However, this approach leads to unequal permissions between the experts. They no longer have equal rights. With the demonstrator presented in this work, it is thrived to support an ad-hoc "brainstorm" like discussion between the experts as equal peers.

A promising approach to shared and separated workspaces is based on the work of Lee et al. [79]. All participants interact by default with a shared view on the data. All experts have equal rights and see the changes to the visualization caused by other participants. This is the shared workspace. As soon as a participant wants to focus on a specific aspect of the presented data independently from the actions of the other experts, he can create his own workspace.

The expert instantiates his own local copy of the current data representation. This local copy is not visible to the other participants. Thus, they can not interrupt an expert in his own workspace (see figure 4.23).

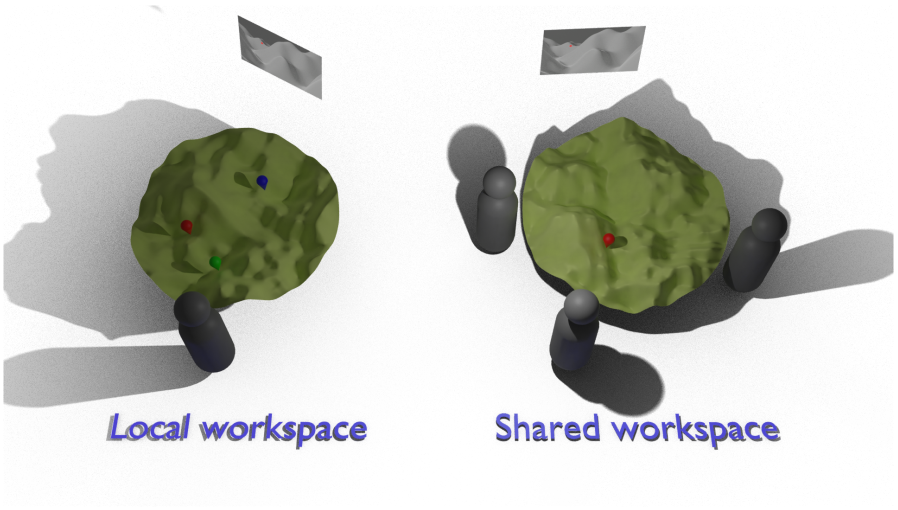


Figure 4.23.: Shared and local workspaces next to each other. Three experts are working collaboratively together on the shared workspace (right) and one expert working on his local copy (left). The expert can dynamically join or leave the discussion with the others.

If a participant in our use case creates a local copy, a clone of the shared landscape model is instantiated at another position in the room. This is the user's local copy. This landscape representation can then be placed freely in the room. From the expert's point of view, two tabletop models of the landscape are now visible: the shared landscape and the local copy. The actions of other experts on the shared landscape are still visible to the one expert. He is therefore not excluded from a discussion and can dynamically join or leave it by moving to the corresponding tabletop model.

This approach combines the individual interests of the experts with a collaborative discussion and is therefore selected for this prototype.

5. Implementation

This chapter describes the technical implementation of the theoretical concepts presented in the previous chapter. It starts with an overview of the tools and libraries used. Next, the application workflow is described on a high level. Afterwards, it is divided into two phases - preparation and analysis phase.

5.1. Used tools

The application developed in this work is based on the *Unreal Engine* game engine. Unreal Engine supports the development of AR applications for the HoloLens 2. It offers ready-made basic functions for creating, managing and displaying three-dimensional scenes in AR. Using the UE Editor, complex scenes can be constructed and previewed. With the help of the integrated profiler *Unreal Insights*, statistics about the utilization of GPU, CPU and main memory are collected and listed [50]. A library provided by Microsoft called *UX Tools* is integrated [108]. This library enables an application to access the different input methods of the HoloLens, such as hand tracking, eye tracking and speech recognition. It provides essential pre-built three-dimensional user interface (UI) components such as sliders and buttons that can be interacted with using hand tracking. *Graphics Tools* is another library that provides pre-made shaders and materials that have been optimized for HoloLens' limited processing capabilities [109]. Several of the basic shaders in Unreal Engine can be replaced with optimized versions from this library to improve performance during execution. For processing the digital elevation models, the library *libtiff* is used [33]. Since UE does not support the TIFF image format by default, this library loads and converts the elevation textures into a supported format.

A library named *proj* provides generic geocoordinatesystem transformation functions [120]. All functionalities of this application have been programmed in C++ classes based on Unreal Engine 4.26. To compile the AR application for HoloLens, *Windows 10* and *Visual Studio 2019* are required.

In addition to the libraries used for the AR application, this thesis also depends on services to store and manage terrain data on an external computer. Due to the open-source structure, maturity and detailed documentation, *mapserver* was chosen for managing terrain datasets [32]. This service, combined with an *Apache web server* [31], is used to stream map data over the network to the HoloLens. Both services are orchestrated using *Docker* [23].

5.2. Application workflow

Before looking at the individual application features, this section describes the basic application workflow from the user's perspective at an abstract level. This workflow is divided into two phases, the preparation phase and the analysis phase. Figure 5.1 shows the transitions between the two phases. As soon as the user opens the application, he starts in the preparation phase. To transition from the preparation phase to the analysis phase, the user either hosts a new virtual collaboration session or joins an existing session. The application automatically switches from the analysis phase back to the preparation phase when either the session has been terminated by the host or the client has disconnected from the session on its own. Each phase is modeled in a separate Unreal Engine level.

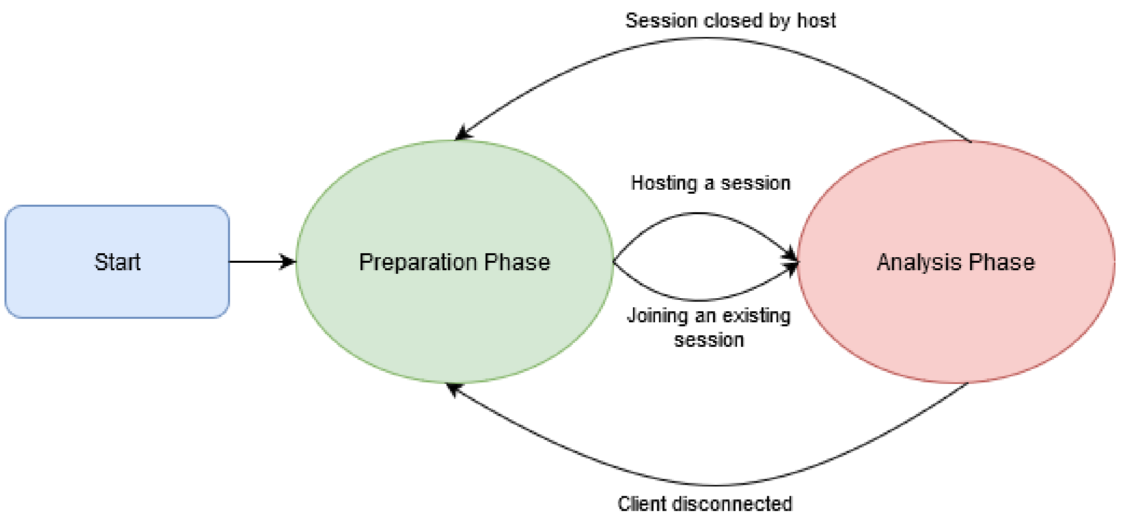


Figure 5.1.: Transitions between the two major application phases.

5.3. Preparation phase

The application automatically loads in the preparation phase level. This phase is used to familiarize the user with the concept of a hand-attached menu and set the basic requirements for multi-user collaboration. This menu is shown in figure 5.2. The representation of such a hand-attached menu is used in many parts of the application. It was decided to use such a uniform representation of the menu elements because it is always quickly accessible in the three-dimensional space. A danger of three-dimensional user interfaces is the occlusion of other virtual objects behind the menu. This type of menu avoids this problem. It is only visible when the user holds his palm upright in front of his head in the viewing direction.

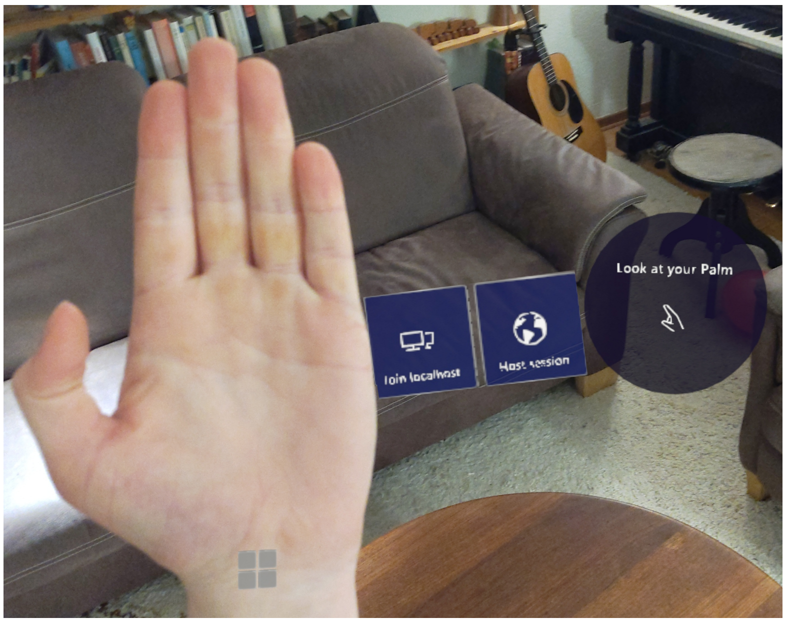


Figure 5.2.: Using the hand menu, the user can join an existing virtual session or host a new one.

This menu is used to manage the virtual collaboration sessions. In figure 5.2, the hand menu consists of two buttons: the right button displays the word "Host session". A new virtual collaboration session is created when the user presses this button. Using the left button, the user joins an existing virtual analysis session. While the application is in the preparation phase, it periodically checks whether other experts in the same network have already created sessions. For each session in the network, another join button is displayed with the session's name (see figure 5.2).

Before the user can join a session or open a new session, the user is notified by a floating hint that a QR code needs to be scanned first. This QR code is used to synchronize the different coordinate systems of the HoloLenses. This application recognizes the QR code from the HoloLens front-camera feed.

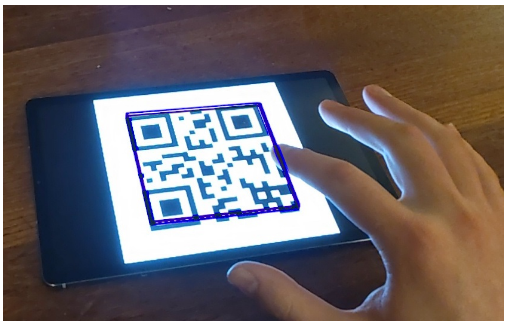


Figure 5.3.: The QR code acts as a common fixed coordinate system between all HoloLenses in a session.

As shown in figure 5.3, the 3D position and orientation of the QR code are determined from the pixel coordinates of the recognized corner points and the known dimensions of the printed marker. Based on position and orientation, the marker coordinate system is initialized.

5.4. Analysis phase

The analysis phase is the main phase of the application. In this phase, the virtual collaboration session is established, and the experts can explore the landscape and the scientific activities of the rovers.

The following section will first explain how large high-resolution terrain datasets and rover data is displayed in AR. The focus is to visualize this data on the limited computing capacity of the HoloLens and simplify adding new and managing existing datasets. Several tools that the experts can use to support the discussion and highlight the spatial relationship of the rover data are presented.

5.4.1. Landscape visualization

As discussed in section 4.1.1, the rovers' surrounding landscape should be displayed horizontally in front of the experts. One goal of this landscape tabletop model is to represent the terrain features as detailed as possible. The following section will examine how a given digital elevation model of the landscape can be displayed on the HoloLens with special consideration of its performance capacities. For this purpose, different technical implementations are tested, and their influence on the application's frame rate and memory utilization is investigated. Finally, the technical implementation of the landscape navigation is described.

5.4.1.1. Terrain visualization on HoloLens

The DEM, satellite and aerial imagery are to be used to render an area of a landscape three-dimensionally in front of the user in the world in miniature format. A circular section of the landscape is to be displayed horizontally at a defined position in space, visible for all experts. To visualize the landscape model on the circular section, we decided to follow the vertex displacement approach.

The basic idea of this approach has already been introduced in section 4.1.2. A height displaced uniform vertex grid represents the landscape surface. The points in this grid are shifted according to the height value in the elevation texture. The quality of this approximation depends on the density of the two-dimensional point grid.

As explained in section 2.5, the entries of the GeoTIFF digital elevation model represent the elevation in meters above the surface of the associated georeference system. Since the landscape in front of the experts is displayed in world in miniature format, the height of each point has to be scaled according to the scale factor between Unreal Engine units and meters on the landscape (see section 4.1.1).

The vertex displacement approach was implemented in Unreal Engine in two different ways to determine the best fitting solution.

Geometry vertex displacement: One approach updates the actual vertices in the grid. The application iterates over each vertex in the vertex buffer, samples the height from the elevation texture and moves the vertex up or down according to the sampled value. For each vertex this operation is performed on the CPU. The updated vertex buffer overwrites the old buffer in main memory. This process can only be performed serially and not in parallel on multiple cores. If an expert shifts the landscape model or changes the zoom level, the geometry of the model needs to be updated to represent the new section. The mesh needs to be updated by resampling the heights for each vertex and displacing the vertices. This results in the CPU updating the geometry every time the landscape section is moved.

Shader vertex displacement: An alternative approach is not to change the model's geometry in the main memory but to change the visual representation by programming the vertex shader of the model. As explained in section 2.3.2, the vertex shader is used, among other things, to project the three-dimensional vertex coordinates from the world coordinate system into the screen space. Unreal Engine allows the integration of custom shader code into the render pipeline. This makes it possible to change the vertex coordinates during rendering. The height texture stored in the graphics memory can be accessed from the vertex shader. In the shader, the corresponding height of the point is read from the elevation texture based on the previously calculated texture coordinates. During the rendering process, the vertex shader is called once for each vertex in the mesh. This means that in each frame, the height is resampled from the texture and the vertex is moved, regardless of whether an expert navigates the landscape section or not. However, unlike vertex displacement on the CPU, this operation is performed in parallel on the graphics card. The CPU can take over other tasks during this time. Since the mesh's geometry is not changed, a static mesh component can be used for this mesh in the Unreal Engine scene. Since the topology of such a static mesh does not change during runtime, according to Unreal Engine, calculations during image synthesis can be accelerated in contrast to procedural mesh components.

Performance measurements A stable frame rate is essential for AR applications to avoid discomfort and nausea. According to Microsoft's claims, a HoloLens application should be able to achieve a refresh rate of at least 30 Hertz permanently [94]. To check which of the two approaches described above, achieves a better frame rate, both approaches were implemented and tested in two different scenarios.

Scenario 1: In the first scenario, it is assumed that the landscape section does not change. The height texture of Mount Etna derived from Ganci et al. [52] is used as the digital elevation model. A rectangular uniform grid consisting of 250 x 250 quads is instantiated as a static and procedural mesh component for both approaches. Each mesh component is placed in a different Unreal Engine level so that when loading into the level, the whole mesh is rendered each frame. The procedural mesh component uses the approach described first and displaces the vertices in the geometry (geometry vertex displacement). The static mesh component displaces the vertices each frame in the vertex shader (shader vertex displacement). In both combinations the mesh has a model length of one meter and is rendered with *Phong shading* [118] and a directional light without shadows for two minutes from the same perspective.

Scenario 2: The second scenario differs from the first scenario because the visible landscape section changes. For this purpose, the landscape section is continuously shifted in one direction. This is intended to simulate the manual shifting of the section by an expert. Besides that, the same test setup is taken as described in scenario 1.

For two minutes, the application's performance was measured by the Unreal Insight profiler. The application ran on the HoloLens, and the HoloLens was placed at a fixed position in the room to exclude possible side effects caused by user movement.

Diagram 5.4 shows the measured frame rates for both approaches and both scenarios. The average and first percentile minimum frame rates are shown in the diagram.

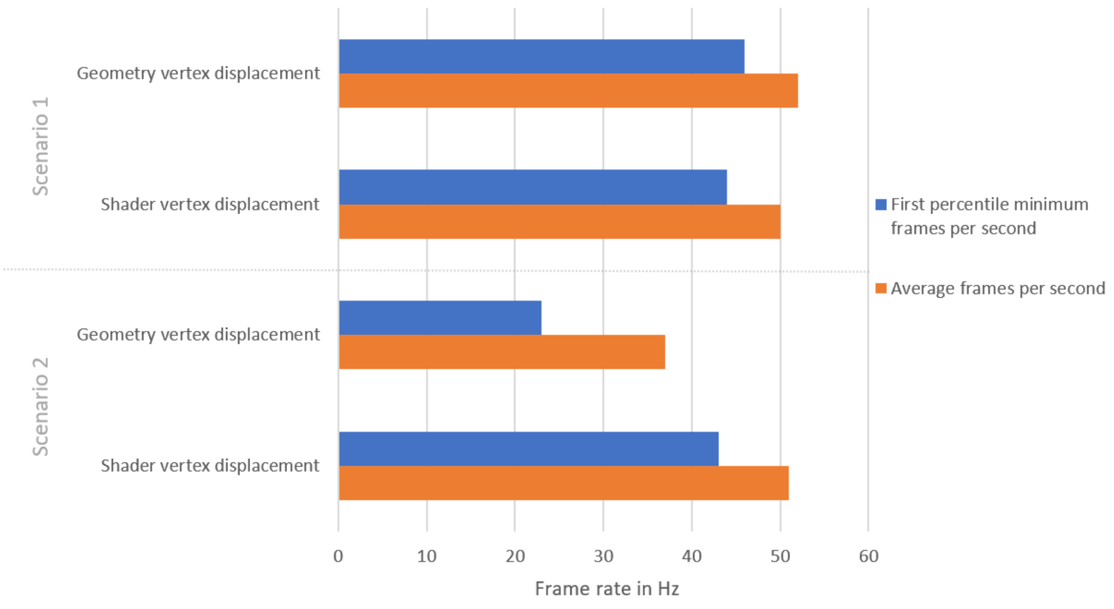


Figure 5.4.: Average and first percentile minimum frames per seconds using geometry vertex displacement and shader vertex displacement on stationary and shifted landscapes.

The first percentile frame rate is the lowest frame rate of the 99% fastest rendered images. This metric is often preferred to the absolute minimum frame rate, since this is strongly influenced by individual measurement outliers. Especially at the start of the application, where memory may have to be reallocated, the rendering of a single frame may take significantly longer and thus strongly influence the minimum frame rate. In order to disregard these individual outliers, it was decided to examine the first percentile minimum frame rate.

Based on these measurements, it can be seen that the frame rate with the shader vertex displacement, is statistically the same in both scenarios. Each time a texture lookup and calculation of the new displaced vertex position must be performed in the vertex shader in every frame.

With geometry vertex displacement the application's performance differs significantly between scenarios 1 and 2. In scenario 1, a slightly better frame rate can be achieved on average using geometry vertex displacement compared to shader vertex displacement. This can be explained by the fact that in scenario 1, the visible section of the landscape is not changed and therefore the vertex buffer is updated only once. With the shader vertex displacement, the height texture is sampled every frame causing a performance drawback.

In scenario 2, the average frame rate of the geometry vertex displacement approach drops by 50% since the vertex position is updated serially by the CPU in every frame. The application is severely impacted by the HoloLens' limited CPU performance.

The navigation of the landscape section to be displayed - tested in scenario 2 - represents a central aspect in the described use case. Especially during the interaction and navigation of the landscape, the application must achieve a consistent frame rate to avoid user discomfort. Therefore, it was decided to visualize the three-dimensional terrain based on displacement in the vertex shader.

The visual quality of the landscape projection depends, among other things, on the selected resolution of the grid. The impact of different grid resolutions on the visualization of the landscape is shown in figure 5.5. From left to right, the resolution of the grid increases from 16 x 16 quads to 128 x 128 quads. The higher the resolution, the better the approximation of the model to the actual shape of the landscape.

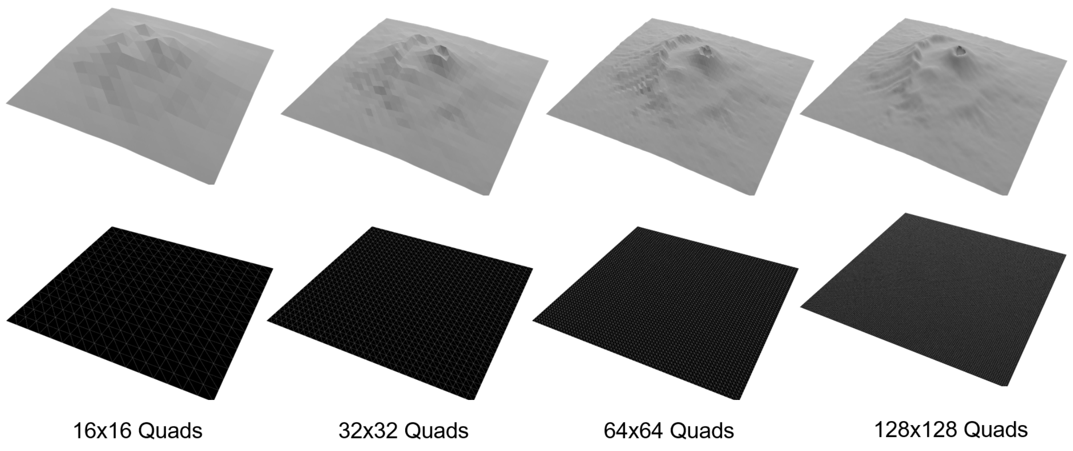


Figure 5.5.: Influence of grid resolution on visual representation of the landscape.

It is investigated how the resolution of the grid influences the frame rate of the application. Based on the previous test scenario, an Unreal Engine scene was designed to render different grid resolutions and measure the frame rate. The grid was placed in front of the virtual camera in the scene to make the entire grid visible. The application was run on the HoloLens fixed in the same room for 2 minutes, and the frame rate was measured using the Unreal Insight profiler.

The result is shown in diagram 5.6. Generally, it can be seen that the higher the number of triangles to be rendered simultaneously, the lower is the frame rate. Based on these test results, it was decided to use a resolution of 250 x 250 quads (125000 triangles), as this offered the best compromise between frame rate and terrain quality.

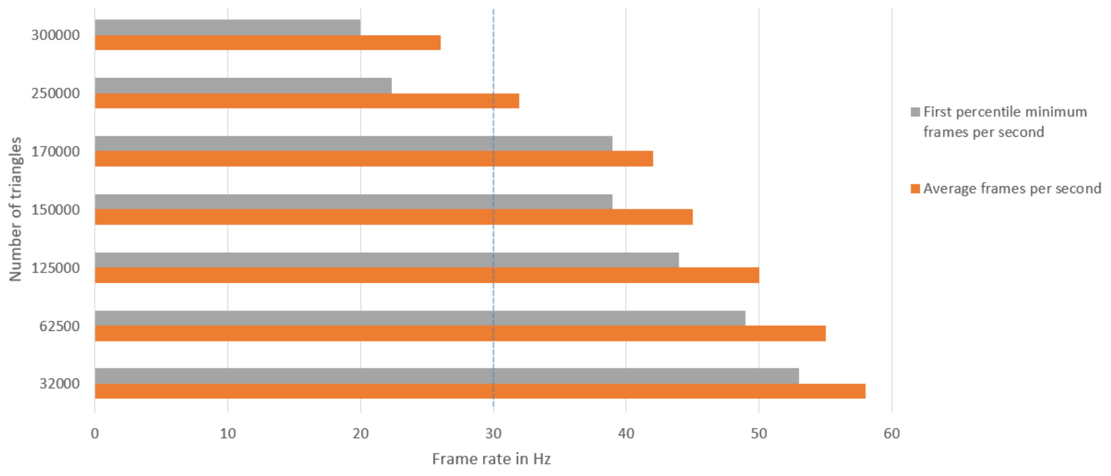


Figure 5.6.: Influence of number of triangles in terrain grid on average and first percentile frame rate.

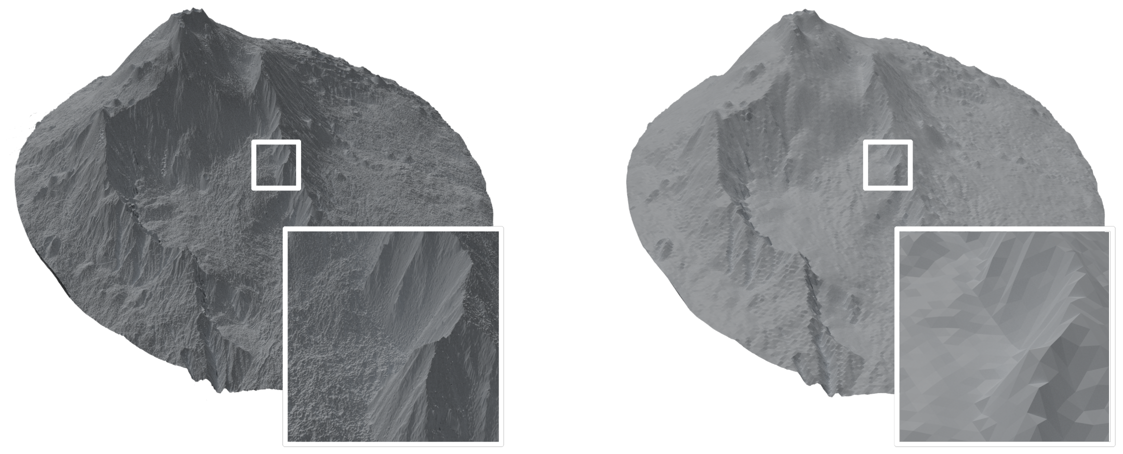
5.4.1.2. Visual enhancements

The visual quality of the model rendered on HoloLens in real-time is compared to a high-quality offline render of the same region.

With a given digital elevation model, figure 5.7 shows a good approximation of the actual measured surface profile on the left side. For a visual comparison, a digital elevation model of the area near Mount Etna is used [52].

The spatial resolution of this elevation texture is four meters, and the absolute is 3780 x 3780 pixels. In figure 5.7a the landscape model is rendered using a uniform grid with a resolution of 7560 x 7560 quads. So the elevation texture is oversampled twice. It thus represents an accurate approximation of the terrain shape according to the *Nyquist Shannon theorem* [131]. However, since this grid comprises more than 114 million triangles, this model can not be rendered in real-time on the HoloLens in this quality. Therefore the model on the left was rendered offline.

The landscape in figure 5.7b can be rendered in real-time on the HoloLens since the grid only approximates the landscape surface with a resolution of 250 x 250 quads, which is a


(a) Landscape model rendered offline using 7560 x 7560 grid

(b) Landscape model rendered in real-time using 250 x 250 grid

Figure 5.7.: Visual comparison between high-quality offline rendering and real-time capable landscape rendering.

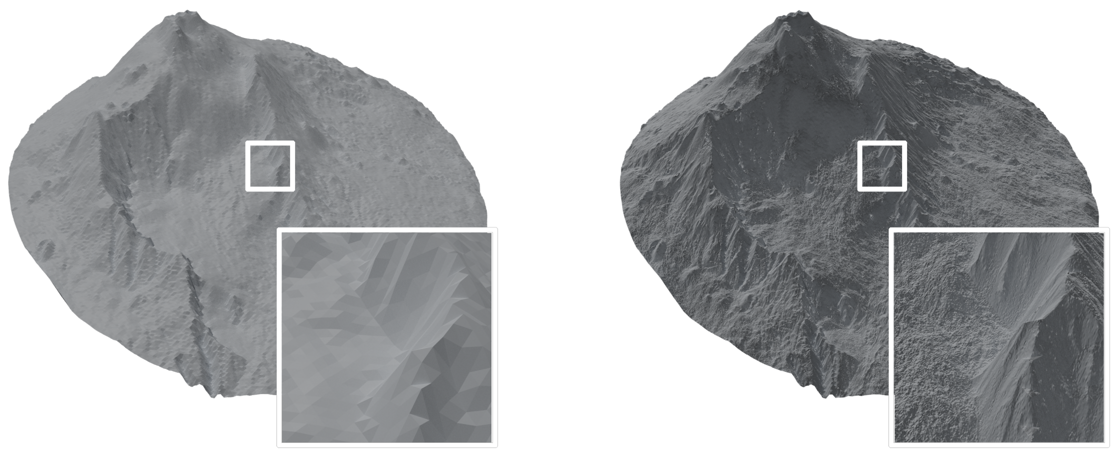
real-time capable resolution as previously shown. Comparing the resolution that can be displayed on the HoloLens in real-time (figure 5.7b) to offline rendered landscape (figure 5.7a), the difference in quality is clearly noticeable, especially in the details of the furrows in the mountain. In the following section, three different computer graphics approaches to improve the visual quality without increasing the geometric complexity have been implemented.

Normal mapping: The basic idea of normal mapping was first formulated by Blinn in 1978 [12] and subsequently generalized by Cohen, Olano, and Manocha in 1998 [17]. The principle of normal mapping is based on the idea of storing surface details of the geometry to be visualized in a texture, the *normal map*. Storing these details makes it possible to enhance the visual quality without increasing the geometry's complexity (meaning the number of triangles).

By default, Unreal Engine uses a physically-based rendering approach to calculate how incident light interacts with the surface and its parameters and thus how the corresponding fragments are subsequently colored. This calculation mainly takes place in the fragment shader of the render pipeline [68]. An essential factor during this calculation is the surface normal of the fragment to be rendered. This normal is implicitly given by the geometry of the surrounding vertices of the current primitive. Depending on the normal direction, incident light interacts differently with the surface material. Details in the surface, such as small indentations, causes changes in the surface normal in an otherwise homogeneous region.

In normal mapping, these changes in the normal direction are stored in a separate texture, the normal map. Then, during image calculation, this change in the normal direction is read from the texture for each pixel and the surface normal is adjusted for light calculation.

This technique is used to visually enhance the landscape representation in this work. The height information of the landscape implicitly describes the changes of the surface normals. These normals are reconstructed from the partial derivatives in X and Y direction of the height texture. In the fragment shader these changes in the normal direction are sampled from the normal map and then applied for the lighting calculations. The visual difference in the landscape rendering is shown in figure 5.8. On the left is the landscape without applied normal mapping technique, on the right side the shading is influenced by normal mapping. Although the mesh resolution and height texture are equal in both models, the model on the right is more detailed.


(a) Landscape model rendered in real-time using 250 x 250 grid.

(b) Landscape model rendered in real-time using 250 x 250 grid and normal mapping.

Figure 5.8.: Visual comparison between real-time landscape rendering with and without normal mapping.

Level of detail: The use of level of detail approaches was also tested for landscape rendering. As explained in section 3.3.1, the Unreal Engine's landscape renderer divides the landscape into sections. Per section, a different level of detail can be loaded, depending on how far away this section is from the camera.

Level of detail optimizations are particularly effective when the objects to be rendered vary greatly in distance from the virtual camera. In our use case, we consider the world in miniature format of the landscape, where we expect users to view the model from an approximate top-down perspective most of the time (see figure 4.1). In this format, each section of the landscape would have approximately the same level of detail distance to the camera. Only in situations where the user views the landscape parallel to its surface could such an approach be beneficial. Due to the increased complexity of splitting the landscape surface, as well as the low expected performance advantage, it was decided to not use different levels of detail.

Relief mapping: Another implemented approach to represent surface details without increasing geometry complexity is *Relief Mapping* [119].

This mapping allows geometrically simple meshes to be enhanced with information from an elevation texture. Figure 5.9 shows an example of this mapping. A single quad is rendered. Figure 5.9a renders the quad with normal mapping and a color texture. In figure 5.9b, relief mapping is used for rendering.

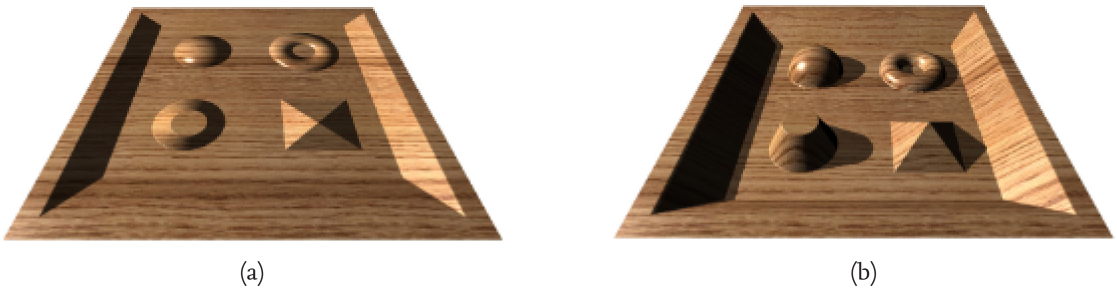


Figure 5.9.: A single quad rendered from the same perspective with: (a) Normal mapping, (b) Relief mapping [119].

Although both figures show the same geometry with the same heightmap, the image rendered with relief mapping looks much more realistic. Effects such as self-occlusions, interpenetrations, shadows and per-pixel lighting effects are rendered with the correct perspective.

The relief mapping uses a screen-space raytracing algorithm to determine the surface ray intersections. The principle of this algorithm is shown in figure 5.10. Assume that the red colored edge is the surface of the primitive to be rendered and the green curve are the entries in the height texture. Further, let position B be the current fragment to be rendered. The viewing direction is defined as the vector from the viewer to the 3D position of the considered fragment. The goal is to determine precisely where the viewing direction intersects the height field (point 3). If this intersection point is known, the fragment attributes, such as normal, depth or color of this point are used for shading, instead of those of fragment B. Using a combination of linear and binary search, this intersection point is calculated for each fragment in each frame [117].

Figure 5.9 shows that relief mapping allows very simple geometries to be rendered realistically and according to a given heightmap with the correct perspective. It is hypothesized that relief mapping can add visual detail to the existing landscape rendering without increasing the geometric complexity. To verify this, the algorithm described above is implemented in an Unreal Engine shader. Three different parameters determine the accuracy and performance of this shader: linear search steps, binary search steps and best

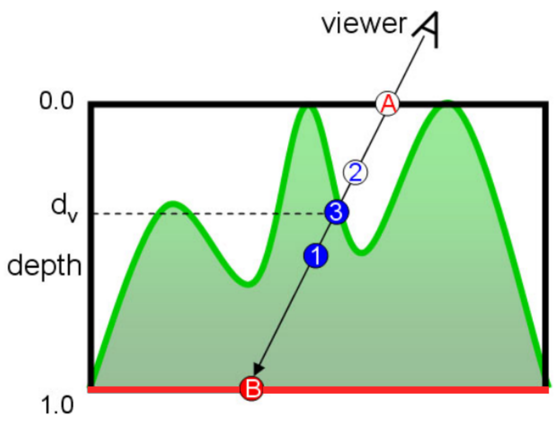


Figure 5.10.: Ray intersection with a height-field surface [119].

depth threshold. The first two parameters determine the maximum number of iterations for the linear and binary search for the ray-surface intersection. The higher these values are, the more often the height field is sampled along the ray and the slower the shader is accordingly. The third parameter determines the threshold from which a surface is considered to be intersected.

With the test setup described in the previous section 5.4.1.1, the influence of the additional calculations in the shader on the frame rate was determined. The relief mapping is compared to the 125000 triangle grid. Based on recommended values, the following relief mapping parameters were tested: linear search steps: 15, binary search steps 7, depth threshold: 0.996 [119].

Diagram 5.11 clearly shows that due to the additional texture lookups and calculations for each fragment, the relief mapping approach is on average 9 frames per second slower.

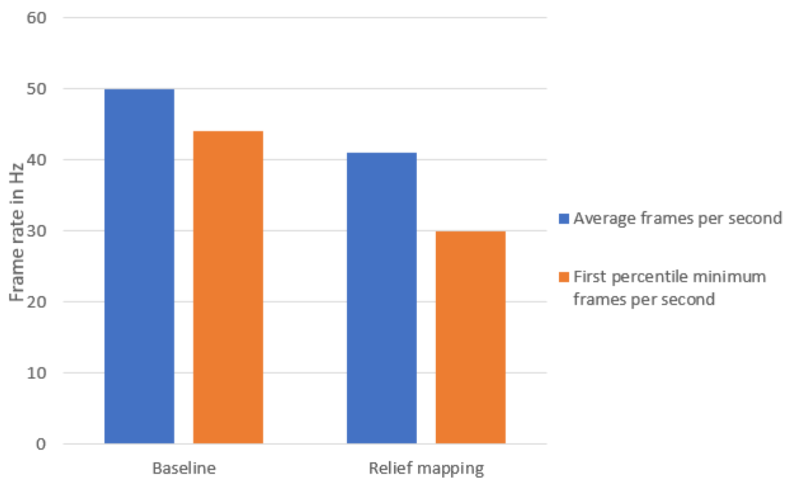


Figure 5.11.: Average and first percentile minimum frames per seconds using normal mapping and relief mapping shading.

The effect of relief mapping compared to the landscape model rendered with normal mapping can be seen in figure 5.12. In particular, when viewing the ridge in magnification, it is noticeable that relief mapping makes some details of the landscape more prominent with the same geometric complexity. However, this difference in quality is not noticeable enough to justify the performance drawbacks. Therefore the prototype includes relief mapping but it is not enabled by default. The user has the option to toggle the mapping on and off dynamically.

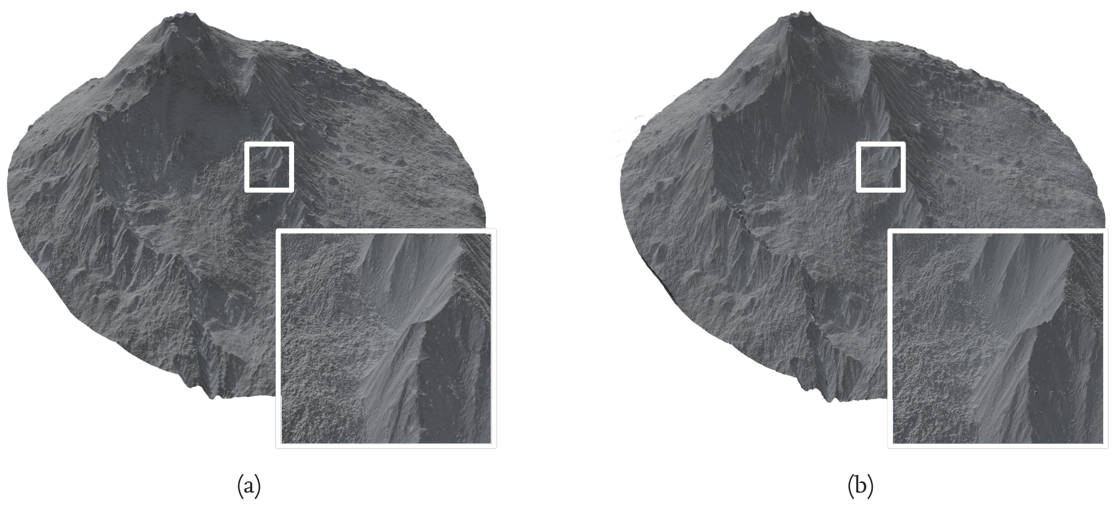


Figure 5.12.: A 250 x 250 grid landscape model rendered with: (a) Normal mapping, (b) Relief mapping [119].

5.4.1.3. Landscape loading

This section describes the setup of the web map service, the available terrain datasets and how the application accesses this data through the service.

Setting up web map server It was decided to use a web map service in section 4.2.2.4. This service is setup on a external computer and connected to the network. For easier management it is run in a docker container on this computer, and the datasets listed in table 5.1 are set up.

All datasets used are publicly available and can be downloaded free of charge. It has been deliberately ensured that both global datasets - *Bluemarble* and *Etopo* - and local datasets are available. With the help of the global datasets, a landscape model of almost any place on Earth can be displayed while the local datasets provide specific landscape areas in higher resolution to the application. The datasets used are stored on the server either in GeoTIFF format or as PNG images. Configuration files are used to assign layer names to the

Dataset name	Type	Area	Coord. system	Spatial resolution
Bluemarble	Color	Global	EPSG:4326	500m
Etna Elevation	DEM	16km x 16km Etna	EPSG:32633	4m
Etna Landsat	Color	17km x 17km Etna	EPSG:32633	3m
Etopo	DEM	Global	EPSG:4326	1 arc-minute
Natural Earth	Bathymetry	Global	EPSG:4326	10m
Vesuv Elevation	DEM	10km x 10km Vesuv	EPSG:4326	10m
Vesuv Landsat	Color	10km x 10km Vesuv	EPSG:4326	3m

Table 5.1.: Configured datasets on web map service.

datasets. In the case of PNG images, the missing georeference information is defined in the configuration.

Selecting the datasets in the application The experts should be able to dynamically select a different elevation model and one or more color datasets during the analysis phase. Before a landscape model is displayed in front of the experts, the application queries the web map service over the network to determine which datasets are available. Then the application distinguishes between digital elevation model and color datasets. Via a user interface, shown in figure 5.13, the experts can dynamically select a digital elevation model and one or more color datasets.

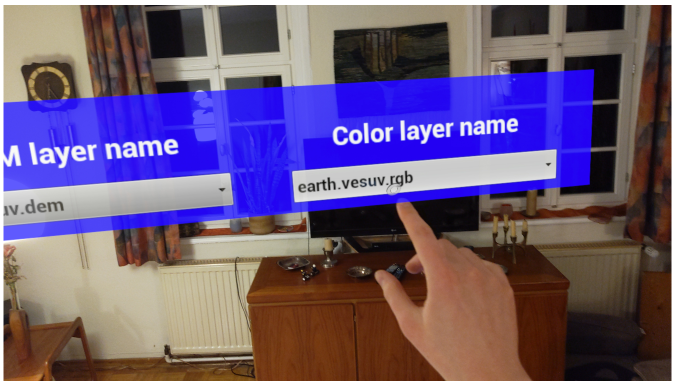


Figure 5.13.: User is selecting the DEM layer (left) and the color layers (right) to be loaded from two drop-down menus.

Fetching terrain tiles over network As soon as the user selects a combination of datasets, a request to the web map service is prepared. The basic idea of tiled landscape loading was discussed in section 4.2.4. A 3 x 3 tile environment around the current landscape area is queried from the web map service via REST requests. The digital elevation model and the color textures are requested one after the other. The server responds by sending the 3 x 3 tile section of the DEM model in GeoTIFF format and the color texture as PNG with additional georeference information. By default, it is assumed that a tile length, which means the area currently shown on the model, is five kilometers long. This value resulted from a discussion under consideration of characteristics of the ARCHES demo mission. Since the area around Mount Etna is in focus in this demo mission, the 15km x 15km neighborhood of Mount Etna is requested from the server. Both the tile length and the geocoordinate of the landscape center can be adjusted in a configuration file in the current implementation.

Caching of tiles When an expert navigates the landscape, new tiles become relevant for the visualization. To avoid requesting the same tiles repeatedly a simple caching approach is employed. Each received landscape tile is tagged with an identification string composed of the dataset name and the landscape area described in the tile. Each tile can be uniquely identified with this string. Using this identification, a simple first-in-first-out caching of the landscape data is implemented on HoloLens' flash memory. Tiles that are still stored in the HoloLens cache do not have to be requested again, but can be loaded directly from the flash memory into the main memory.

This caching reduces the network load and latency. Each time the application is restarted, the cache is cleared to receive the most recent landscape data.

5.4.1.4. User-landscape navigation

To allow the experts to freely explore the landscape and change the zoom level of the model, interactions with the landscape are implemented. In section 4.3 the possible landscape interaction gestures have been discussed. This section shows how these gestures (rotating, panning, zooming) are implemented in the application.

Rotating The rotation of the landscape model is implemented based on the wheel interaction discussed in section 4.3.1. Figure 5.14 shows a landscape model rendered from the HoloLens perspective. A rim runs around this model in a radial pattern. This rim is the wheel, which the user turns to rotate the landscape.

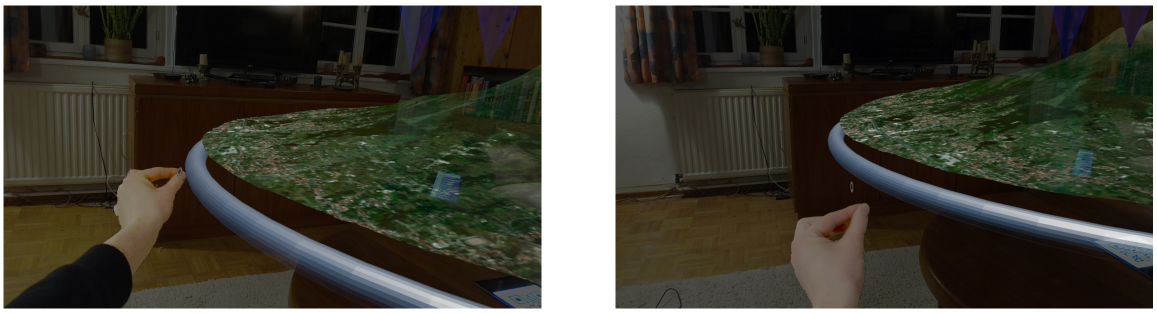


Figure 5.14.: User is rotating the landscape by gripping and dragging the outer rim.

Unreal Engine components for physics simulation are used to implement this in the application. These components allow an application to check, if two actors intersect spatially. When the hand of an expert is detected in the field of view of the HoloLens, a digital replica is added as an actor to the scene. If an expert grasps the rim, the hand and the rim actor overlap. Using functions of the UXTools library, the position and orientation of the fingers are evaluated and it is checked whether the user is grasping the rim. If this is the case, the rotation angle is determined according to equation 4.4 and then applied to the actor of the landscape model.

Panning As discussed in section 4.3.2, an expert should able be to shift the landscape section by grabbing and moving the landscape surface. When the user's hands intersect the landscape model and the hand forms a grasping gesture, the application begins the panning process (see figure 5.15). In each subsequent frame, it is checked whether the user is still grasping the landscape.

According to the equation 4.5, the texture coordinate displacement is calculated and updated in the material of the landscape model.

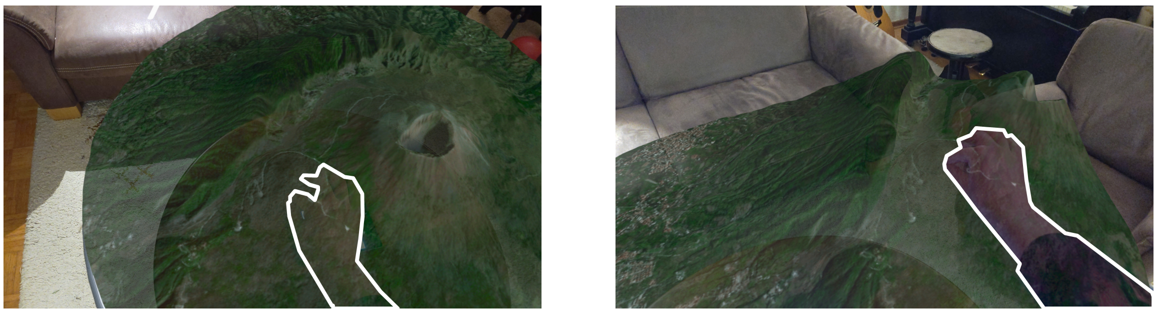
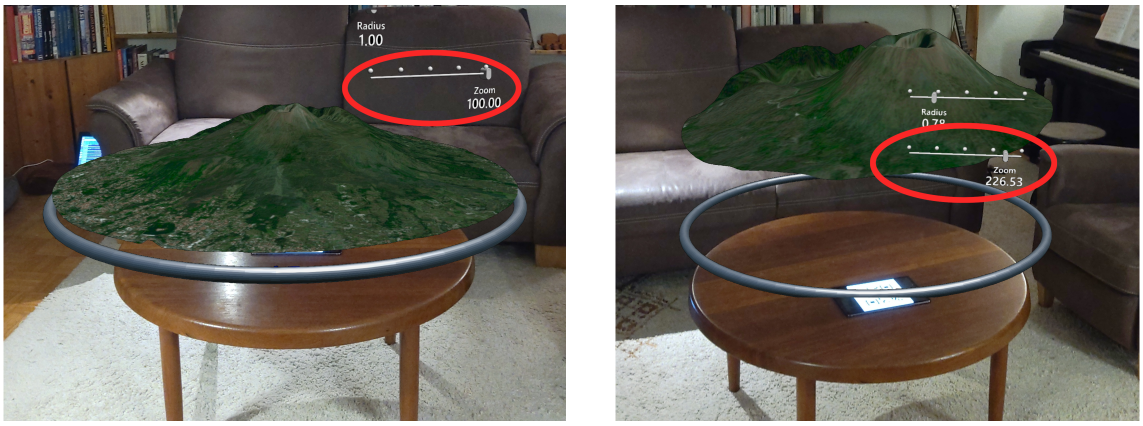


Figure 5.15.: User is panning the landscape by gripping and dragging the surface. The outline of the users hand is highlighted in white in order to increase contrast between hand and virtual content for visualization purposes.

Zooming An expert is able to change the zoom factor of the landscape using a slider. This slider can be seen in figure 5.16. By shifting the knob on the slider, the current scaling factor can be varied within a range from 0.1 to 10. This value range is mapped to a range between 10 and 1000 on a logarithmic scale in the user interface for visualization purposes.

Starting from a tile length of 5 kilometers on the base zoom level, the user can zoom in so far that the entire landscape model covers an area of 500m x 500m. With a scaling factor of 10, an area of 50km x 50km is mapped onto the model. While it would be possible to increase the range of values on the slider to zoom in or out further, it was demonstrated that this significantly complicates the precise adjustment of the desired zoom level.



(a) (b)

Figure 5.16.: User is zooming the landscape by adjusting the zoom factor slider (highlighted in red): (a) landscape zoomed out, (b) landscape zoomed in.

As soon as the expert changes the zooming factor by moving the slider, this value is updated as a material parameter of the landscape model. Following equation 4.3 the texture coordinates are scaled in the vertex shader. If the zoom factor is not changed, the base zoom level of the landscape data is loaded by default. Six different zoom levels can be explored with a zoom factor ranging from 0.1 to 10. According to equation 4.2 the corresponding zoom level to the zoom factor is calculated, and, if necessary, new terrain data is requested.

5.4.2. Texture compression

Different types of images are to be visualized: elevation textures, satellite images, multispectral images and spectrograms. Each of these data types is a texture, which, when being rendered, needs to be loaded in the HoloLens graphics memory. As part of the HoloLens' main memory, the graphics memory is very limited with its 2GB of available combined capacity. By compressing textures, this memory can be used more efficiently. Well-known image compression algorithms, such as JPEG, can reduce the size of an image file on a hard disk. Graphics hardware-accelerated compression algorithms allow a compressed storage of a texture in the graphics main memory and the subsequent sampling of the image contents almost without performance drawbacks [47].

By default, Unreal Engine does not compress image content received at runtime. This is because the compression is not real-time capable. Only the decompression is hardware-accelerated. Depending on the chosen algorithm, it can take several seconds to compress an image. Since this process can technically only take place in the main thread in Unreal Engine, compression at runtime pauses the running application during this time. Nevertheless, we investigated how a compression of the image contents could be used in our concrete use case:

Unreal Engine supports three relevant real-time-capable image compressions for the HoloLens graphics hardware: DXT1, DXT5 and BC7 [47]. All three compression algorithms are lossy. This means that visual quality is lost in favor of a smaller memory footprint. While DXT1 reduces a 32-bit image with a compression ratio of 8:1, DXT5 and BC7 can only save memory with a compression ratio of 4:1. But DXT5 and especially BC7 compressed images have significantly fewer artifacts than DXT1 compressed images [143].

To test the applicability of these algorithms for the HoloLens, a 32-bit RGBA image with a resolution of 1292 x 964 is compressed 10 times and then the average time is calculated. The result is shown in table 5.2.

Compression algorithm	DXT1	DXT5	BC7
Compression time in seconds	0.45	0.83	1.27

Table 5.2.: Average compression times when compressing an RGBA 32-Bit 1292 x 964 image on HoloLens hardware.

Based on the measured times, it is clear that textures can not be compressed with any of the three algorithms in real-time on the HoloLens hardware. During this time, the immersive visualization and the expert's discussion would be impaired. Furthermore, it can be argued that artifacts in the image caused by memory optimization should be avoided when analyzing scientific measurement data. Therefore, we decided to store image content uncompressed in the main memory, but with the option to enable texture compression in a configuration file.

5.4.3. Rover data visualization

This section explains how the visualization of the rover's data and their sites is implemented. First, it is explained how the application processes the metadata of the locations. Then it is shown how the different data categories are distinguished from each other as markers on the landscape. Finally, the implementation of the visualization of multispectral images,3D scans and material analyses are explained.

5.4.3.1. Streaming rover data

For each scientific activity, a virtual marker shall be displayed on the landscape surface at the corresponding position. The rover data is managed by a server and stored in a database. To display the virtual markers at the correct location, the application requests the metadata of all scientific findings in the currently loaded 3 x 3 tile neighborhood. That way, the amount of metadata to be processed simultaneously can be reduced. For each scientific activity, a virtual marker object is instantiated. Data fields necessary for visualization, like location and category are stored as properties on the virtual marker object. When a new 3 x 3 tile grid is loaded from the server, the metadata for this region is requested.

When instantiating a marker, it is necessary to determine where it should be displayed in the Unreal Engine coordinate system. Each measurement taken by the rover is precisely located using geocoordinates. Following the geocoordinate transformation explained in section 4.12, each marker's coordinates is transformed relative to the currently displayed landscape into the Unreal Engine coordinate system. The marker is positioned at the correct location above the landscape surface as a 3D model.

In figure 5.17, several markers are distributed over the surrounding landscape of Mount Vesuvius. The different colors of the markers represent the different data categories. A red marker indicates a multispectral analysis at that location, a blue marker shows the acquisition of a 3D reconstruction, and a green marker represents a material analysis.

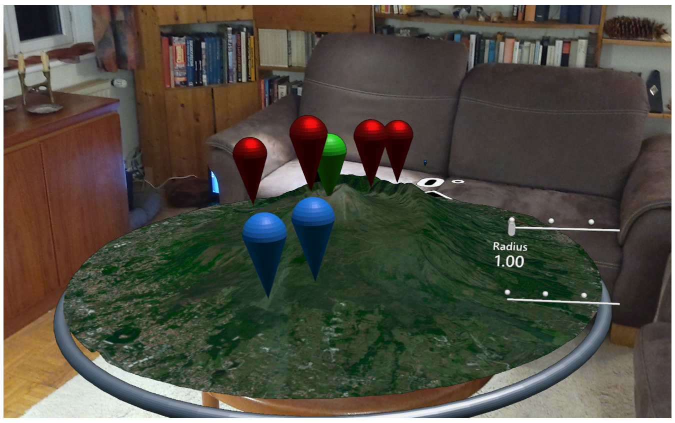


Figure 5.17.: HoloLens view on Mount Vesuv with virtual markers.

Each marker is 6 cm in height when unselected and 10 centimeters when selected. In practical tests, this size proved to be a good compromise. If the marker is too small, it is difficult to grab it by hand, and a marker that is too large tends to overlap with other markers.

As introduced in section 4.4.3, experts are supposed to select or deselect a marker with their hands. A user should only move a marker up or down. The movement is therefore limited to one degree of freedom. The user has two different options to grab a marker:

Firstly, the user can directly grasp the virtual object with his hand and drag it up or down. The interaction is shown in figure 5.18. The user grabs the marker (see a) and moves it up (see b).

Secondly, the user can interact with the marker using the point and commit input method. This method is especially useful for markers that are not within the user's reach. Figure 5.18c shows how a user selects and lifts the marker using his pointer beam.

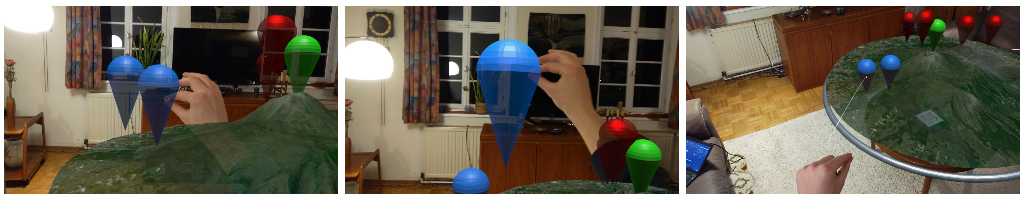

(a) (b) (c)

Figure 5.18.: User is selecting a marker.

To select or deselect a marker, the user must raise or lower it 5 centimeters. This threshold is to avoid unintended interactions. While the user grabs a marker, the selection progress is shown as a slight change in the color brightness as feedback.

5.4.3.2. Multispectral images

The prototype of this work visualizes three different types of multispectral images in AR: mono, stereo and panoramic images. Each multispectral image is symbolized as a red marker on the landscape model.

As decided in section 4.4.4.1, the two-dimensional multispectral images are to be augmented with virtual three-dimensional annotations from the discussion of the experts. Other markers in the image's field of view, together with the experts' drawings and virtual laser pointers, should be augmented to strengthen the spatial context between the image and the surrounding landscape. With this improved spatial context, the experts' discussion should be supported, and possible connections and anomalies in the measurement results should be better understood.

In order to project perspective-correct virtual elements into the multispectral image, the scene is rendered from an additional perspective. This is the perspective that the rover camera had on the surrounding landscape at the time the photo was taken. This additional rendering of the scene is called *point of view rendering*. Figure 5.19 shows neighboring markers and annotations of an expert augmented into the image.

To enable this rendering, a virtual camera is modeled based on the actual rover camera's extrinsic and intrinsic parameters.

For point of view rendering, the scene is rendered from this virtual camera's perspective. This means that the complete render pipeline has to be passed one additional time, which leads to a reduced frame rate in the application. These performance losses increase with each additional multispectral image that is opened simultaneously.

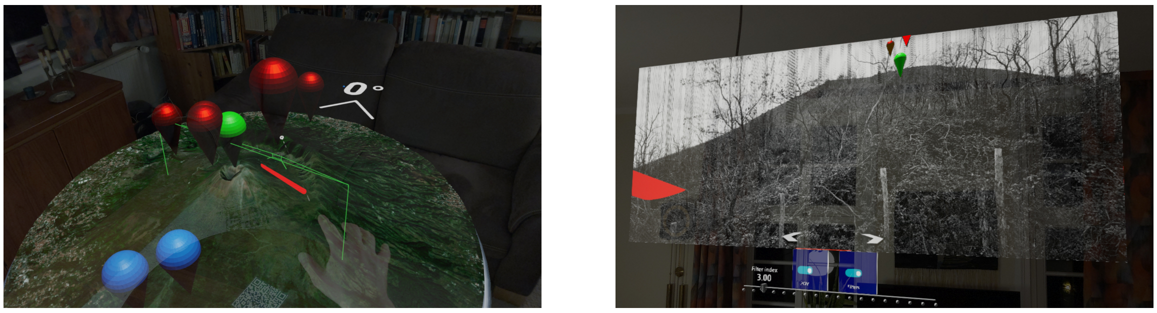


Figure 5.19.: Point of view rendering: The multispectral image is augmented with markers and annotations.

Two implemented approaches are tested to reduce these performance impacts:

First, the point of view image is only re-rendered when the actors to be augmented in the scene change. This is the case when an expert adds annotations to the landscape, actively uses a virtual laser pointer, or opens or closes a marker in the image's field of view. Therefore, in most frames the scene does not need to be re-rendered for point of view rendering.

Further performance impacts can be reduced by avoiding rendering particularly complex geometries in this additional render pass. The most complex geometry in the scene is the landscape model. Since only a few actors in the scene are blended over the multispectral image, the landscape model doesn't need to be rendered. With this, only simple geometries like splines, cylinders and the marker models have to be processed.

At the same time, this leads to the problem of occlusion (see section 4.4.4.1). If the landscape model is not rendered in the pass, otherwise occluded fragments will become visible. To avoid this, the complex landscape model is rendered once after starting the point of view rendering. During this rendering, not the fragment's color but the depth relative to the virtual camera is stored in the texture. If for example a fragment of a marker, is obscured by the landscape, this fragment has a greater depth than the fragment of the rendered landscape. Thus, the single-rendered image of the landscape is used as an occlusion mask.

Even though the landscape model is rendered only once for each multispectral image, this additional rendering process still lead to a noticeable stutter in the application in practical tests. To remove this stuttering, the rendering time of the occlusion mask needs to be reduced. Therefore, the landscape model is not rendered in full resolution of the multispectral image but in a quarter of that resolution. This results in a shorter rendering time and a lower memory consumption of the extra required texture. After implementation, it could be shown that even with the reduced resolution, the occlusion can be determined sufficiently precisely.

For each multispectral image, memory space is allocated. The following textures are instantiated:

- Multispectral image (full resolution)
- Point of view rendering render texture (full resolution, only allocated when point of view rendering is active)
- Occlusion mask ($\frac{1}{4}$ resolution, only allocated when point of view rendering is active)

To prevent the application from reserving more than the two available gigabytes of memory and thus terminating automatically, a maximum of 20 multispectral images may be opened simultaneously. If the received textures are compressed (see section 5.4.2), up to 40 images can be displayed at the same time.

Mono images As soon as a user opens a mono multispectral image marker, a virtual monitor is instantiated as a free-floating projection surface next to the landscape model. In contrast to a physical monitor, the size of the virtual monitor can be scaled by using the virtual handles (see figure 5.20).

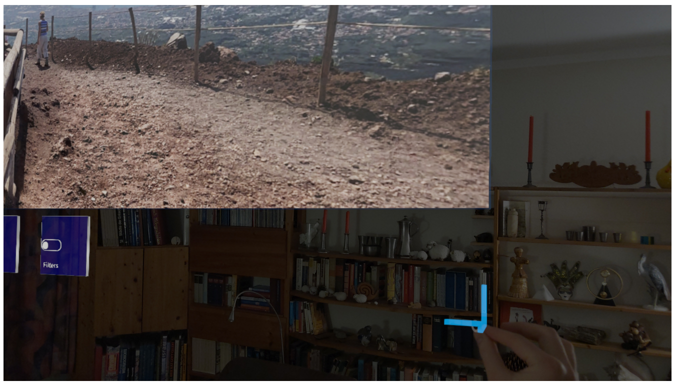
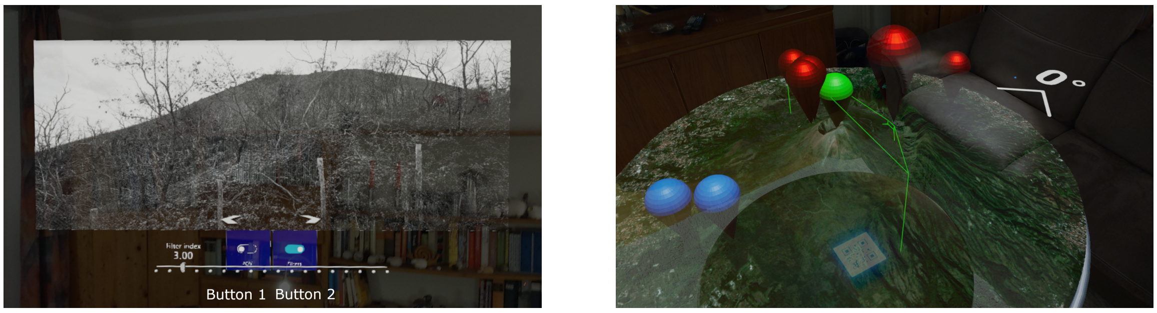


Figure 5.20.: A user scaling the size of the virtual monitor.

The 3D transformation of each virtual monitor is synchronized with all other experts. This guarantees that all participants in a session are referring to the same virtual monitors in the same physical places.

Figure 5.21a shows the virtual projection screen. After opening the marker, the first spectral image with the filter index 0 is displayed on this virtual monitor by default. Two buttons from the UXTools library are displayed below the projection surface.

One button allows experts to activate and deactivate the point of view rendering. The other button controls the visibility of the filter slider. When this button is selected, a slider is displayed below the buttons. This slider is used to select an image with another multispectral filter. Each time a user selects a new filter, it is checked whether the corresponding image is already stored in the application's cache. If not, it is loaded from a server to the HoloLens. The loaded texture is then displayed on the projection screen.



(a) (b)

Figure 5.21.: A virtual monitor showing a mono multispectral image (a) and its associated frustum on the landscape (b).

Not all 18 filter images are automatically loaded into the main memory. When a multispectral image is opened, memory is allocated for this one spectral image, and the previous image contents are overwritten. Loading the new image data leads to a delay of about half a second between changing the filter and displaying the image. However, this allows the application to use the scarce main memory capacity of the HoloLens more efficiently.

A visualization component provided by Unreal Engine is used to display the frustum on the model surface. This component receives the extrinsic and intrinsic camera parameters as 3D transformation matrix and projection matrix. This frustum visualization is depicted in figure 5.21b.

Having multiple images open, it proved to be difficult to keep track, which virtual screen belonged to which marker on the model. Additional to the connection line discussed in section 4.4.4.1 another method to emphasize the linking was implemented. Both the respective frustum and the marker itself are now highlighted in color, when a user interacts with a virtual monitor.

Stereo images As soon as the marker of a stereo multispectral image is opened, a virtual monitor is instantiated similar to the mono multispectral image. Such a virtual monitor is shown in figure 5.22.

By default, the left image of the stereo image is displayed on the projection surface. If necessary, this is loaded from the server or from the cache into the main memory.

In addition to the already known buttons (marked 1 and 2 in figure 5.22), experts can control the display of the stereo image via two other buttons. Button 4 is used to switch between the left and right image. When switching between the images, the image contents in the main memory is overwritten. If point of view rendering is activated, the virtual camera is displaced to the position of the correspondingly selected left or right sensor. Button 3 switches between mono and stereo view of the image. Since both images have to be displayed simultaneously during the stereo view, twice as much memory is allocated. A custom shader displays the image contents stereoscopically on the virtual monitor, by

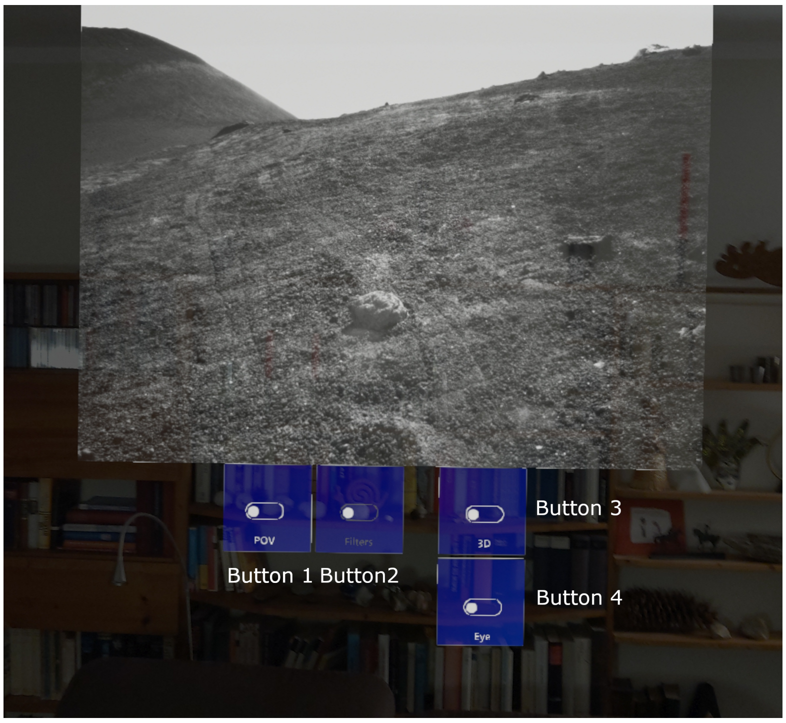


Figure 5.22.: The virtual monitor of a stereo multispectral image.

differentiating between left and right eye rendering in the fragment shader.

If stereo display is active, button 4 is deactivated. Instead, the experts can adjust the strength of the horizontal image translation via a slider. Depending on the value set, the texture coordinates for the sampling of the stereo image are shifted horizontally in the vertex shader.

As soon as the user switches back to mono display, the memory of the second texture is released, the horizontal image translation slider is hidden, and button 4 is activated again, so that the user can switch between left and right eye.

Panorama images If a marker of a multispectral panoramic image is selected, a photosphere is instantiated for each expert in the virtual session.

A large radius of 20 meters was chosen for the representation of the photosphere. There are two reasons for choosing such a large radius. On the one hand, it prevents other virtual content from being occluded. On the other hand, it is more comfortable for the user to focus his eyes on the photosphere at a greater distance.

The experts can choose between two different representations of the photosphere: A complete immersive visualization and a filtered visualization using a virtual flashlight. With a hand menu, each expert can select his or her individual visualization mode.

Regardless of the representation, the equirectangular panoramic image is loaded from the server. To avoid perspective distortion of the panoramic content, the sphere is centered on the user's current head position in each new frame. For both representations, the Cartesian coordinates of the sphere vertices are converted into equirectangular texture coordinates in the vertex shader according to equation 4.12. For a complete immersive photosphere representation, the fragment shader samples the color information from the panoramic image at the calculated texture coordinate for each fragment of the sphere. The result is rendered on the inside of the sphere.

If the user selects the filtered flashlight visualization, it is checked whether the HoloLens detect the user's right hand in each frame. This hand controls the virtual flashlight. To pick up the light, the user is asked to form and hold the Air Tap gesture as long as he or she wants to control the flashlight. The hand position is interpreted as the flashlight position, and the direction vector between the wrist and palm controls the beam direction. Furthermore, the focus is controlled by the user. The rotation angle of the right wrist is mapped to an angle between 180 degrees (fully turned to the right) and 5 degrees (fully turned to the left). Figure 5.23 shows the panoramic photosphere when an exert interacts with his virtual flashlight.

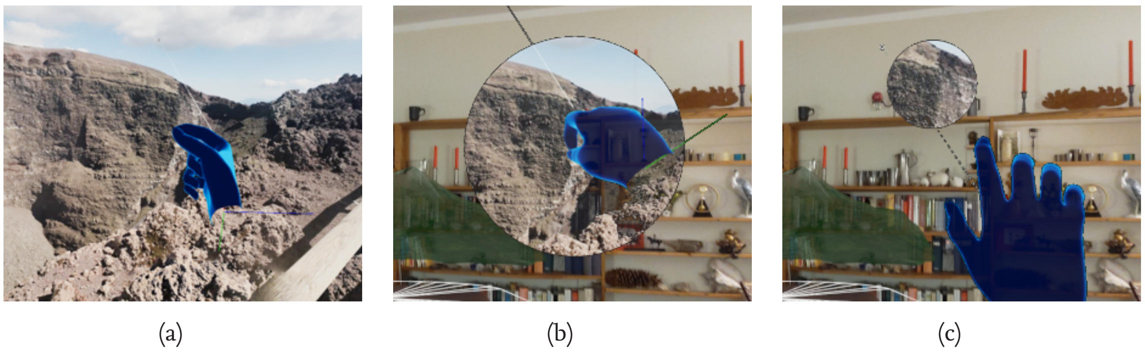


Figure 5.23.: Panorama visualizations: (a) immersive full photosphere, virtual flashlight with medium (b) and small focus (c).

The hand position, beam direction and focus are recorded in every frame as long as the user holds the Air Tap gesture. These parameters are passed to a custom panorama shader as a parameter. The fragment shader then checks whether the fragment is inside or outside the light cone.

For this purpose, the shader processes the current flashlight position $\vec{l}$, the current light direction $\vec{d}$, the focus angle α and the 3D position of the fragment $\vec{z}$.

First, the difference vector $\vec{p}$ pointing from the fragment to the light source is calculated:

$$\vec{p} = \vec{l} - \vec{z} \tag{5.1}$$

To check whether the fragment is inside the light cone, the angle θ between the light direction $\vec{d}$ and the difference vector $\vec{p}$ is calculated using the dot product.

$$\theta = \vec{p} \cdot -\vec{l} \tag{5.2}$$

A fragment is inside the light cone if θ is smaller than the focus angle α. The fragment's color information is sampled from the panoramic image and displayed on the inside of the sphere. If θ is larger than the set focus α, the fragment is outside the light cone and is rendered transparent on the HoloLens display.

Regardless of how the panoramic image is shown to the user, additional virtual annotations are augmented on top of it when using point of view rendering. In contrast to the mono and stereo rendering of the multispectral image, the panoramic image does not have a limited field of view, but covers the entire 360 degrees environment centred around the camera. This means that the entire virtual scene is captured as a 360 degree image by the virual camera.

5.4.3.3. 3D soil sample scans

When an expert selects a 3D scan marker, the system first checks whether this model is already stored in the application's cache. This is the case, if this marker has been selected before in the same session. If the model is not cached, it is loaded as an STL [1] file from the server. If the metadata of the 3D scan contains a path to a texture that is to be displayed on the scan, this texture is also loaded from the server.

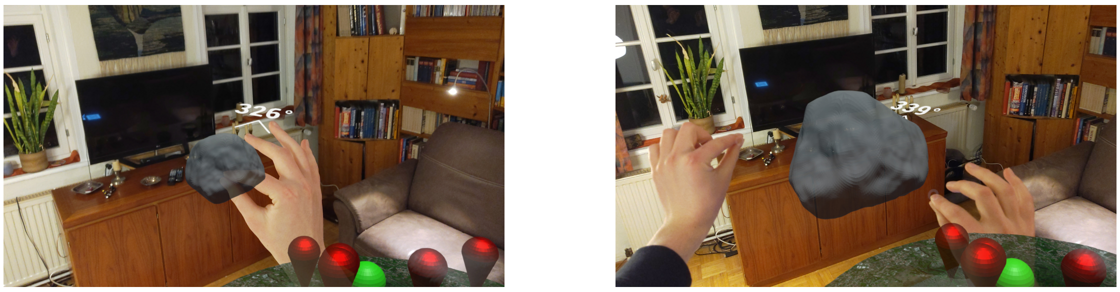
Since Unreal Engine does not support the direct rendering of a STL models imported at runtime, a procedural mesh component is instantiated. The geometry information (vertex buffer and index buffer) is copied from the model. The procedural mesh component, including the optional color texture, is visualized as a 3D model above the marker. As previously determined experimentally, the application's frame rate on the HoloLens decreases significantly with an increasing number of triangles to be displayed. When a scan model is opened, additional triangles need to be processed by the graphics chip. Therefore, it should be noted that the rendering of a geometrically complex scan directly impacts the graphics performance of the application.

A collision component is attached to the model to enable the user to interact with the scan using his hands. This component describes the geometry of a virtual object, which is used for its physics calculations. This collision geometry is often only an approximation of the rendered geometry for performance reasons.

To detect when a user grabs a scan, it is checked each frame if the collision geometries of the hand and the scan overlap. This check is accelerated by using the convex hull of the scan as a simplified collision geometry.

[1] Standardized file format to describe a triangulated mesh

A user's interaction with an opened 3D scan is shown in figure 5.24. A user grabs the 3D scan and changes its position and orientation in 6 degrees of freedom (see figure 5.24a). With the help of the second hand, the scaling of the scan can also be changed freely. Depending on whether the user brings his hands closer together or spreads them apart, the scaling decreases and increases (see figure 5.24b). A scan can be scaled to 20 times its actual size, to better recognize details.



(a) (b)

Figure 5.24.: Expert is investigating a 3D reconstruction of a soil sample with one-handed (a) and two-handed interaction (b).

An arrow points horizontally to the north side of the scan to indicate its original orientation. After opening a 3D scan, the model is automatically oriented so that the north of the scan points north on the landscape model. If the scan's orientation changes, the relative angle to the north of the landscape is shown.

5.4.3.4. Material analysis

When an expert opens a marker of a material analysis, a virtual monitor is instantiated and positioned next to the landscape model. This monitor can also be placed freely in the room for the experts. First, it is checked if the analysis data is already available in the application's cache. If this is not the case, this spectrum is loaded from a server as a PNG image file and displayed on the virtual monitor.

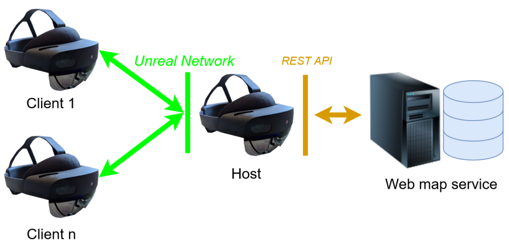
5.4.4. Network communication

To enable a unified view and shared interactions, the applications on the different HoloLenses must communicate with each other. The internal state of the shared actors is synchronized between all participants. An overview of those actors and their properties are listed in table A.1.

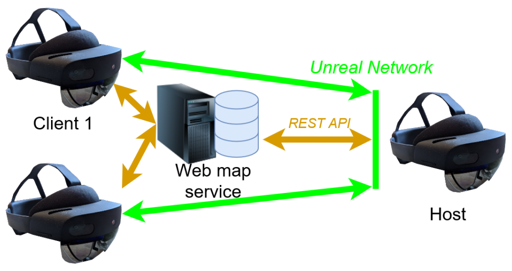
For communication and synchronization of the actors, the replication system provided by Unreal Engine is utilized. UE uses a client-server model to coordinate this communication and ensures that all participants get the correct data. One participant acts as host and opens a virtual session. All other participants connect to the host as a client. The host is then responsible for synchronizing the current state of the application. The synchronized properties have to be exchanged via the host. Client applications should not communicate directly with each other.

In our implementation, not only the HoloLenses communicate with each other, but also receive external data from the Web Map Service or the rover database. Similarly, this data should be received synchronously by all HoloLenses.

One way to distribute the external data to all participants is shown in figure 5.25a. Only the host communicates with the external server and then distributes the received data to all other participants via the Unreal Engine network. This has the advantage that the host knows precisely when all HoloLenses have received the data completely. However, the host's computational and memory overhead increases with the number of participants in his session. In addition, the duration until all participants have received the data increases, due to the limited network bandwidth of the host HoloLens. These disadvantages disqualified this approach for our use case.



(a)



(b)

Figure 5.25.: Integrating external data into Unreal Engine network. Only the host communicates with external servers and forwards the data to all clients (a). Each application is communicating with the server individually (b)).

An alternative approach is used in the current prototype. This approach is shown in figure 5.25b. Again, the internal state of the actors is synchronized through the UE replication system via the host. However, in this case, each HoloLens communicates directly with the external servers. This approach has the disadvantage that the host does not know when all participants in the session have successfully received the data from the server. Therefore, the prototype assumes that all participants in the session receive the terrain or rover data needed for the visualization with low latency and approximate timing. This approach avoids the additional memory and computational overhead of the host. The transfer of data between the server and HoloLens is thus only limited by the network infrastructure, even with many participants in the session.

5.4.5. Tools

Within the scope of this work, different tools were developed. They can be classified into three categories: tools for filtering the markers on the surface, tools for collaboration and discussion among the experts, and tools for the targeted investigation of specific landscape areas. Each of these tools are accessed by the experts with a hand menu shown in figure 5.26.

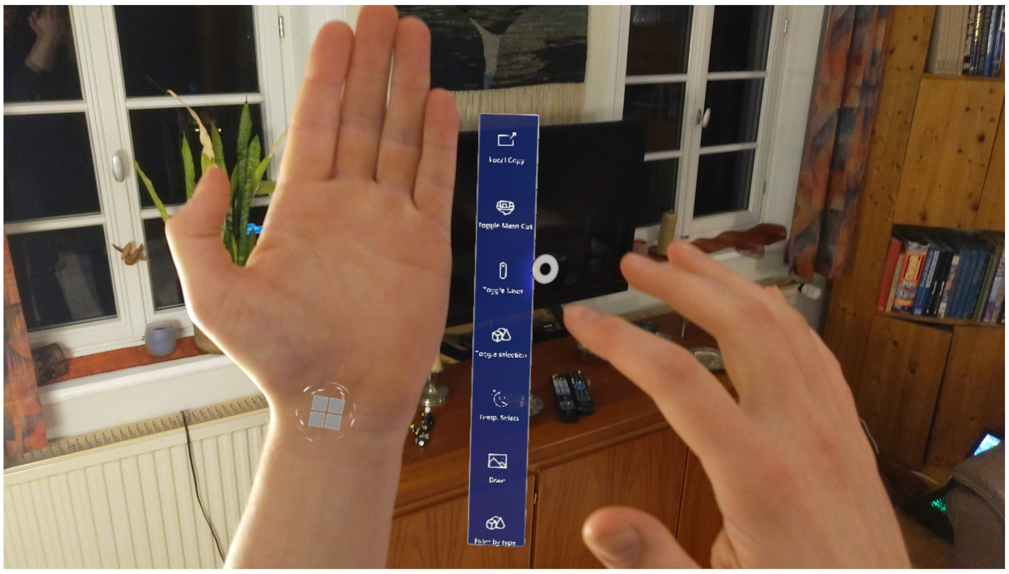


Figure 5.26.: The user is selecting his tool from the hand menu.

5.4.5.1. Filtering tools

Depending on the complexity and time span of the rover mission, the number of the scientific activities varies. The presented AR application should be able to display the scientific activity history of complex rover missions, even if a lot of measurements were performed during the mission. In the approach described so far, a marker is placed on the landscape surface for each scientific activity. If this approach is applied to a mission where several hundred measurements have been performed, this results in a large number of markers being displayed on the landscape model simultaneously. This affects both the application performance and the expert's overview by visualizing too much information at the same time. Different tools for filtering scientific activities are integrated into the application.

Spatial filtering The spatial filtering tool allows the expert to obtain only information on scientific measurements within a predefined area. Markers of scientific results outside this area are hidden. This tool is based on the freeform lasso selection tool, which is a standard for image processing programs [2, 56]. Its concept is transferred to virtual and augmented reality to select in 3D space.

Using the spatial filtering tool, an expert marks the area to be selected directly with his hands. The integrated hand and finger tracking of the HoloLens is used for this purpose.

This tool was implemented in the following steps:

- Once an expert selects this tool, the shared landscape model is locked in its translation. This means that no expert can move the landscape while another expert is in the process of selecting an area. This is to prevent the selection process from being influenced by the actions of the other experts.

- In each frame, the current hand transformation of the user is queried. A blue beam - the virtual hand ray - starting from the right hand in the direction of the index finger is displayed to all experts in the virtual session. This ray serves as an extension of the index finger to indicate precisely where he or she is drawing on the landscape.

- Each frame in which the ray is intersecting the landscape model while the user is holding the Air Tap gesture, the 3D coordinate of the intersection point is calculated and stored in a list.

- The entries in the list are then used to render a spline curve to show the drawn path during the selection process.

- The user confirms the selection process by deselecting the tool in his hand menu. The intersection points are processed to check which markers are located within the drawn area. This check is called *point-in-polygon* test. Depending on the total number of markers, this process can decrease the application performance. To reduce the computational complexity of these checks, a more efficient *point-in-convex-hull* test is implemented.

  The shape of the area selected by the expert is approximated by a convex hull of the intersection points. This offers two advantages for the performance of the application. Firstly, the construction of the convex hull reduces the number of intersection points that span the area.

  Secondly, point-in-polygon tests have a higher temporal complexity than point-in-convex-hull tests [14]. This means that checking if a marker is inside a convex hull requires less computational effort then checking if it is inside a polygon.

  By deactivating the tool, all experts can move the landscape section again, and the expert's hand ray disappears.

Temporal filtering The experts use the temporal filtering tool to visualize only activities recorded within a certain time period.

Next to the landscape visualization, a 3D slider is displayed for all experts. This slider is shown in figure 5.27. The timespan between the earliest and the latest scientific activity

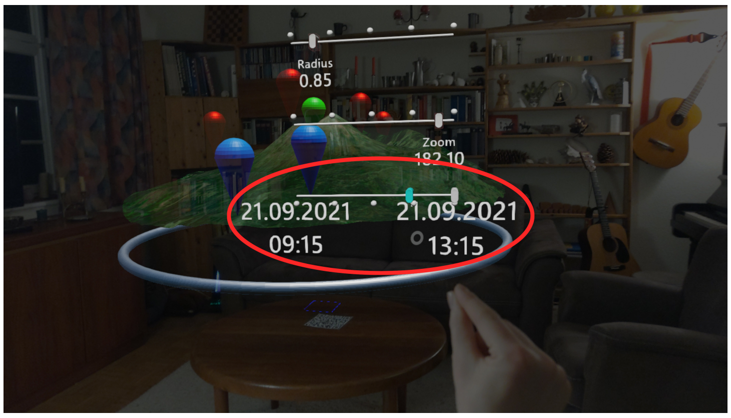


Figure 5.27.: A user selecting a timespan for temporal filtering.

is mapped on this slider. The two handles can be grabbed and moved to select a specific timespan. The timestamp of each activity is checked and the marker on the landscape is shown or hidden.

Category filtering The category filter tool allows experts to filter the scientific activities by category. For this purpose, a menu was designed using the Microsoft UX Tools components to filter first by main category (multispectral images, 3D scans and material analysis) and then by subcategory (panoramic, mono or stereo images). This menu is shown in figure 5.28. It is designed as a hand menu and is always accessible to each expert. The individual categories can be activated or deactivated by touching the checkboxes with a finger.

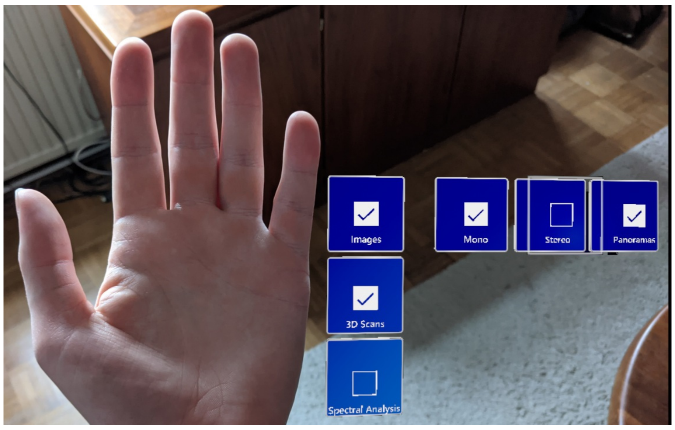


Figure 5.28.: A user filtering the markers by category.

5.4.5.2. Collaboration tools

In the following section, three further tools are implemented to stimulate and support the discussion of experts in a co-located collaboration scenario.

Laser pointer The laser tool is based on the metaphor of a laser pointer and is used to draw other people's attention to specific details. This tool is shown as a virtual hand ray, highlighted red in figure 5.29. Although all experts are in the same environment, the use of a virtual hand ray is beneficial, even if all experts can directly see the pointing person [16, 75]. The 3D transformation of the virtual hand ray is synchronized for all participants.

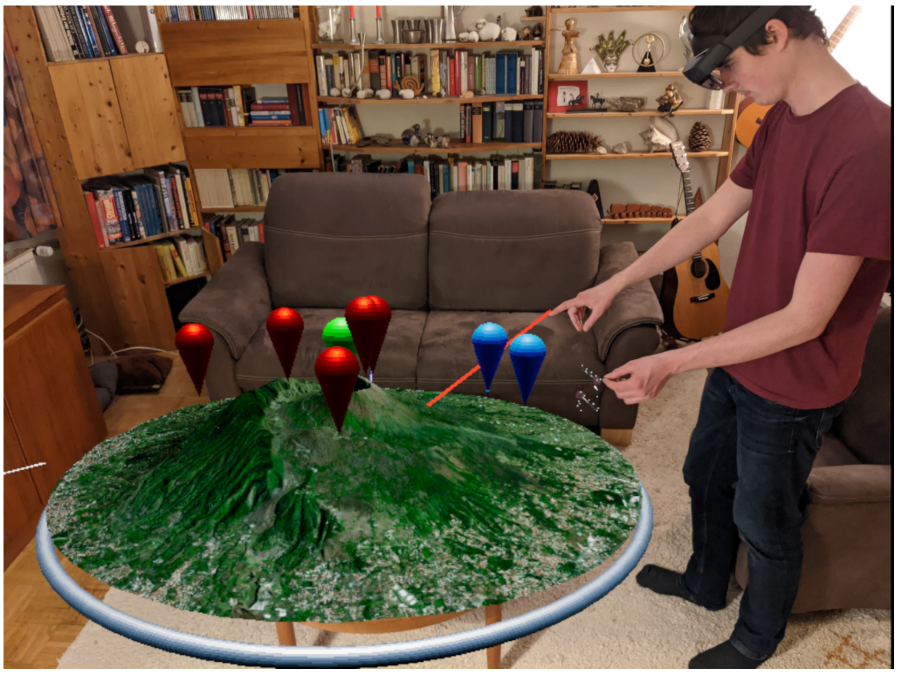


Figure 5.29.: An expert highlights a specific landscape feature to other participants in the session using the laser tool.

The laser tool should enhance the linking between the scientific measurements and their surrounding landscape. It is intended to connect the multispectral images and the landscape model. Using point of view rendering, the lasers of all experts are augmented into the image. That way, a marked point on the landscape model is highlighted in the multispectral images. Conversely, if an experts points to a specific location in the multi-spectral photo, this location is highlighted on the landscape model. This principle can be seen in figure 5.30.

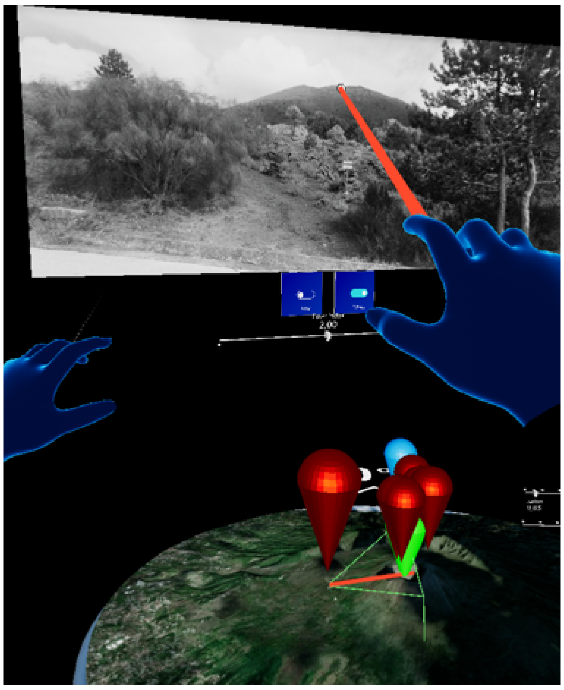


Figure 5.30.: The virtual hand ray is projected onto the landscape model.

The laser pointer tool was implemented as follows:

- The current hand position of the user is queried in each frame. The virtual hand ray is displayed in red color to all experts.
- If a user directs his laser beam onto the screen of a multispectral image, the texture coordinate of the intersection point is accurately determined.
- The projection matrix of a virtual camera describes how three-dimensional coordinates are projected onto a two-dimensional surface. The goal now is to determine the opposite direction: It is calculated how the texture coordinate on the image propagates as a ray in three-dimensional space.

  For each multispectral image the intrinsic parameters and consequently it's projection matrix are known. A given texture coordinate can be transformed into a three-dimensional ray using the inverse projection matrix. Since the depth of the image content is not known, one image coordinate can not be assigned to exactly one 3D coordinate, but to an infinite number of coordinates along a ray. The linear equation of this ray is determined in the camera coordinate system for the given texture coordinate.
- Subsequently the ray is transformed into the world coordinate system using the known extrinsic camera parameters. Starting from the point on the landscape where the photo was taken, the laser beam is displayed in the calculated direction.

Pen The pen tool allows the experts to highlight certain areas on the landscape model. In contrast to the laser tool where only the current point is shown, the pen tool allows to draw directly with the hands on the three-dimensional surface. These drawings remain visible until the expert deactivates the tool. Figure 5.31 shows annotations drawn by an expert on the landscape model.

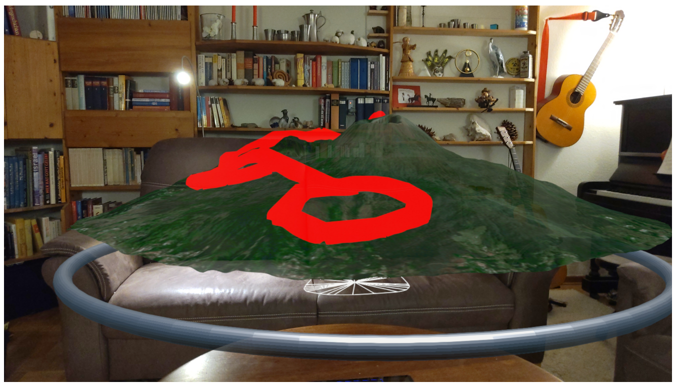


Figure 5.31.: Expert annotating specific areas on the landscape surface.

The tool is implemented in the following steps:

- Once an expert activates this tool from the tool menu, the virtual hand ray is automatically displayed on the expert's right hand.
- In each frame, the expert's hand position is queried, and the virtual hand ray transformation is updated.
- If the user points his virtual hand ray onto the landscape surface, a cursor appears at the intersection point with the landscape.
- As long as the user holds the Air Tap gesture and the hand ray intersects a landscape surface, the current intersection position is added to a spline in each frame. This spline is visible on the surface for all experts.

Local copy tool The implemented AR application is intended to support discussion through the shared and local workspace approach (see section 4.5.2). The actors in each workspace are organized hierarchically. Figure 5.32 shows an overview of the structure of actors within the workspace.

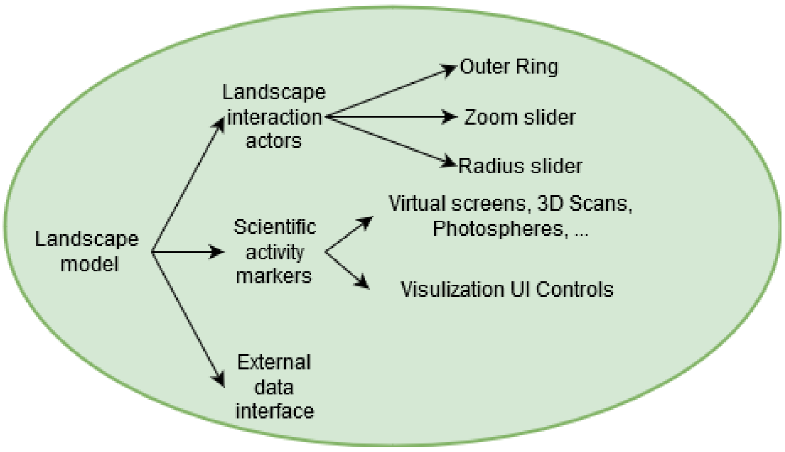

Landscape model
Landscape interaction actors
Outer Ring
Zoom slider
Radius slider
Scientific activity markers
Virtual screens, 3D Scans, Photospheres, ...
Visulization UI Controls
External data interface

Figure 5.32.: Hierarchical structure of actors in a workspace.

The top level of the workspace is the landscape model as the central interactive element of the discussion. The components experts use to interact with the landscape are associated with this model: the outer ring and the zoom and radius slider. Those are also directly part of the workspace.

The scientific activity markers on the landscape's surface and the components used to reload rover and landscape data form the next layer of the workspace. Actors that visualize the recorded rover data, such as the virtual monitors or the 3D models and their UI elements, are connected with their respective marker object and are thus also part of the workspace.

To create a local copy of the shared workspace, the internal state of its actors is copied. Components provided by Unreal Engine are used to create a recursive copy of the hierarchically arranged actors. The newly created actors are explicitly excluded from the UE replication system, so they are not instantiated by the other participants in the session.

As an independent copy, the local landscape model also accesses its own separate DEM and color textures. This allows different landscape areas and even different datasets to be displayed on the two models simultaneously. At the same time, however, double the memory space is allocated. Furthermore, in the worst case, when both models are in the user's viewing area, twice as many triangles are processed by the graphics card. Instead of 125000 triangles, 250000 triangles are rendered only for the landscape models. This leads to a significant impact on the frame rate. However, according to the previous performance measurements (see figure 5.6), it shows in practice that even in this worst case, the average frame rate does not fall below 30 frames per second.

5.4.5.3. Extract tool

This tool is designed to facilitate the analysis and precise investigation of a specific landscape area by visualizing this area separately as a three-dimensional cut-out model.

An expert selects the area of interest on the landscape model with his virtual hand ray (see figure 5.33a). After confirming the selection, this area is visualized as an independent free-floating model in addition to the main landscape model. Therefore, it can be freely placed in space to a convenient location by any expert (figure 5.33b). The cut-out model can be scaled independently to see fine details in the area (figure 5.33c). Together with the central landscape model, the experts can analyze two different areas of interest at the same time.

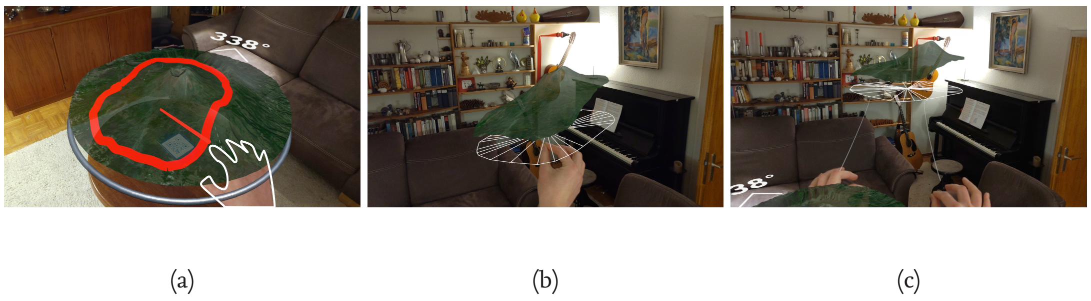


Figure 5.33.: User is extracting an area of interest (a) and inspecting it with one-hand (b) and two-hand (c) manipulation gestures.

The extract tool extends the implementation of the previously described pen tool:

- In each frame where the virtual hand ray intersects the landscape model surface while the user holds the Air Tap gesture, the texture coordinate of the intersection is stored in a list.
- This list is processed as soon as the user confirms his selection by deactivating the tool. The convex hull and bounding box of the texture coordinates in this list are calculated to simplify the cutting process in reducing number of relevant intersections points.
- The resulting bounding box describes the relevant image patch of the DEM and color texture to represent the extracted area. Since the image content of those textures changes, if a different area is displayed, a new DEM and color texture with the size of the bounding box is created. The image content within this bounding box is copied into the new textures.
- A new uniform vertex grid with the shape of the previously calculated convex hull is instantiated as a procedural mesh component and added to the scene.
- The section of the DEM and color texture is passed to this procedural mesh component. With this height and color information the uniform vertex grid is visualized as a three dimensional free floating landscape model.

- Similar to the interaction with 3D scans, the landscape section can be manipulated and scaled with one- and two-handed gestures (see figure 5.33b and c). The components described in section 5.4.3.3 are utilized for this interaction.
- The same cut-out of the landscape should be visible and intractable for the other experts at exactly the same physical location. To achieve this, the points of the convex hull are synchronized to the other participants. As soon as an expert moves the cut-out, its new transformation is synchronized to all others. This ensures that the section is aligned and located at the same physical position for everybody.

5.5. Application performance in demanding multi-user scenarios

This section investigates if the prototype can guarantee a stable interactive frame rate and a memory usage below the design limit of 2 GB, even in very complex scenarios. In the scenario to be tested, six participants are connected in a virtual session. Since only one HoloLens was available at the time of the measurement, only one participant is real. The other participants are controlled by five computers. The application performance is measured on the HoloLens.

In this challenging scenario, the 10 x 10 kilometer environment around Mount Etna is shown. This landscape is displayed using the standard prototype settings described above[2]. Two landscape models, one as a shared and one as a local workspace, are visualized simultaneously. To test the performance of the application when reloading terrain data, new landscape data is requested and processed every 10 seconds. A total of 50 markers are placed on the shared landscape model, 23 of which are opened simultaneously:

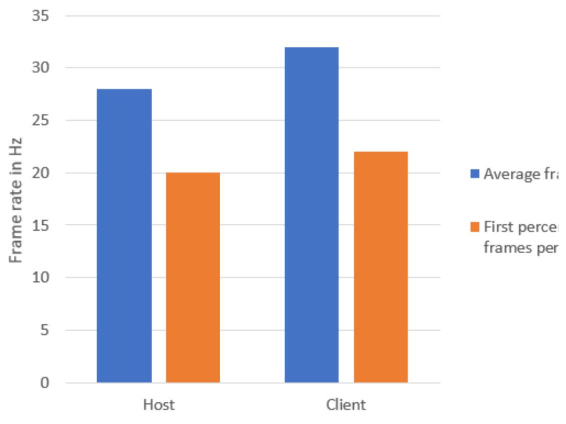
- 20 multispectral images (17 mono multispectral images, 2 stereo multispectral images, 1 panoramic image) [3]
- 3 soil sample scans (1211 triangles, 828 triangles, 930 triangles)

The five computer-controlled participants move in each frame, so their transformation is always synchronized via network. At the same time, a new filter is selected every five seconds for all mono multispectral images. This scenario is used to measure the memory consumption and the frame rate over a period of five minutes. In one condition the HoloLens is the host of the session, in the other it is the client.

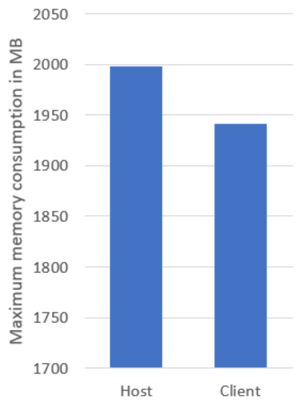
[2]250 x 250 vertex grid, normal mapping, 1000 x 1000 pixel DEM model and 3000 x 3000 pixel satellite color texture.

[3]Each multispectral image had a resolution of 1388 x 1038 pixels and the panoramic image had a resolution of 4000 x 447 pixels.

As shown in diagrams 5.34a and 5.34b, the additional management and synchronization overhead on the host affects the performance of the application. Further, we can see that the maximum memory consumption in both scenarios is less than two GB. Finally, it can be seen that both the HoloLens as host and as client achieve interactive frame rates on average even in this demanding scenario.



(a) Average and first percentile minimum frames per second



(b) Maximum memory consumption in megabyte

Figure 5.34.: Application performance on HoloLens as host and client in a demanding multi-user scenario.

6. Evaluation

This chapter evaluates the implemented concepts presented in the previous chapters. An expert interview and user study were planned to evaluate the concepts' relevance for a real-world scenario and the system's usability. Unfortunately, neither the expert interview nor the user study could be conducted due to the current COVID-19 pandemic. Therefore, no qualifiable data on the usability of the demonstrator could be collected. However, preliminary user feedback on the presented interaction gestures and visualization approaches are promising.

Even though the expert interviews and user study could not take place before submitting this thesis, the interview and the study's planned procedure will be explained in the following.

6.1. Expert interview

The expert interview was designed to evaluate the usability of the implemented demonstrator in a realistic scenario. For this purpose, several experts would have been invited to an expert interview, to show them the prototype and demonstrate it with a concrete use case. Afterwards, the experts' opinions on the presented concepts were to be evaluated qualitatively in an open discussion.

An expert interview with several researchers of the ARCHES project was planned. Especially the team of planetary scientists should have been involved. The members of this team are responsible for the rover data analysis. To evaluate the usability of the implemented concepts, we wanted to interview these experts who work in this field of research every day. Because the experts involved in planetary research work at the German Aerospace Center (DLR) in Berlin, the expert interview was scheduled to take place there.

After a brief round of introductions, the demonstrator's basic concept and the HoloLens as an AR device would have been explained first. Using Microsoft's standard showcase applications the device's capabilities would have been demonstrated. The available interaction methods between these showcase applications and the user would be demonstrated to the experts playfully. This introduction would have given the experts an overview of the possibilities the HoloLens offers. It would have enabled them to assess strengths and weaknesses better when testing the actual prototype.

With three available HoloLenses, an expert group of three persons could have tested the prototype simultaneously. The mixed reality capture feature of the HoloLens would convey to the other experts what is currently visible on the HoloLenses. The other experts would see a live feed of one HoloLens' front camera, blended with the currently displayed virtual content. Thus enabling them to share the wearers experiences. All experts would have had the chance to try out the HoloLens.

After that introduction, the prototype would have been launched on the three HoloLenses. The applications would have been connected in a joint virtual session. One of the three HoloLenses would have been worn by the interviewer, the others by two experts. Then, the interviewer would have explained the landscape navigation gestures and shown the visualization of the multispectral images, 3D scans, and material analyses. After this explanation, the implemented tools would have been presented to the experts. The principle of the local and shared landscape models would have been shown. Finally, the experts would have been given the opportunity to test the prototype on their own using a prepared demo scenario.

In this demo scenario, the scientists would explore the landscape and measurements at the ARCHES demo mission location, near Mount Etna. A high-resolution digital elevation model of this area would have been visualized in front of the experts. On the landscape model, a total of 15 markers would have been selectable. These markers are a collection of mono/stereo/panorama multispectral images, 3D reconstructions and material analysis.

The experts would have been encouraged to explore the collected data behind the markers using the provided tools. Interactions with the three-dimensional landscape model and discussions among the participants would have been supported.

This procedure would have been repeated until all experts had a chance to test the prototype. Afterwards, the experts' opinions on the demonstrator would have been evaluated in an open discussion. The purpose of this discussion is to assess the usability of the presented tools in a realistic application scenario.

Unfortunately, due to the current Corona restrictions, the expert interview could not take place in person. It was considered to conduct this interview using a telephone conference with a video transmission from the perspective of the HoloLens. The focus of this work is to investigate how the benefits of AR can be applied to the rover data analysis. One of the major advantages of AR is the immersive visualization of three-dimensional virtual content in the real environment. The advantages of the AR prototype would have been difficult to show, only through a video transmission of someone else's point of view, especially to users who have not yet used AR devices. The experts would have neither been able to perceive the virtual content immersively in their own environment. Nor they would have been able to inspect the 3D representation of landscape and rover data on their own. With a remote expert interview, the concepts implemented in this thesis could not have been evaluated in a meaningful way. Therefore, we decided against a telephone expert interview.

6.2. User study

The following user study is intended to determine the usability and intuitiveness of the implemented tools, gestures and visualizations. Four different experiments are planned to be conducted. During these experiments qualitative data will be collected with questionnaires and quantitative data will be measured.

The qualitative evaluation methods are explained first, and their use for evaluating the prototype is discussed. Afterwards, the planned user study is described. Then each of the four experiments including their conditions, null hypotheses and task descriptions are explained. Following that, the statistical evaluation methods employed are outlined, and the user study's possible outcomes are discussed.

6.2.1. Qualitative evaluation methods

The following section describes the qualitative evaluation methods used in this thesis.

6.2.1.1. NASA Task Load Index

The NASA Task Load Index (NASA TLX) is a standardized questionnaire to measure the total workload required to complete a given task [61]. Six different factors are evaluated in six questions (see appendix A.2):

- Mental demand
- Physical demand
- Temporal demand
- Task performance
- Effort
- Frustration

The first three factors evaluate the requirements for completing the tasks; the last three factors explain how the probands deal with the task and how satisfied they are with their execution. Each factor is rated on a scale from 0 to 100 in increments of 5. A variant called Raw NASA TLX is applied to make a statement about the total workload with a smaller number of probands. This variant does not require a pairwise comparison of the importance of the individual factors for each proband [60].

6.2.1.2. System Usability Scale

The System Usability Scale (SUS) is a standardized scale that evaluates the subjective usability of a system for a given task [11]. The user is asked to describe his or her position on a Likert scale on ten statements (see appendix A.3). The answer options vary from *"strongly* *disagree"* (0) to *"strongly agree"* (4). The selected answers are added together and then multiplied by 2.5 to calculate a value between 0 and 100. In general, a system with a usability score of over 68 is considered to have above-average usability. The system usability scale has the advantage that the calculated score is a simple metric for the usability of a system even with small numbers of probands.

6.2.1.3. Questionnaire for the subjective consequences of intuitive use

The QUESI (Questionnaire for the subjective consequences of intuitive use) is a standard questionnaire to evaluate how intuitively a software can be used. It is assumed that a system can be used intuitively, if previously learned methods are unconsciously processed by the system as expected. The questionnaire assumes that an intuitive system leads to a reduced cognitive load, a lower learning effort, familiarity, and subjective confidence in error-free operation [113, 66]. The factors evaluated in the QUESI therefore include: *subjective mental load, perceived achievement of goals, perceived effort of learning, familiarity* and *perceived error rate.*

Like the NASA TLX questionnaire, the QUESI is used to evaluate the subjective maturity of a system during a task. The QUESI, however, looks at the system from a different perspective. The questionnaire is attached in the appendix A.4.

6.2.1.4. Situation Awareness Global Assessment Technique

In one of the experiments explained in the following section, we investigate to what extent the presentation of the panoramic images in AR influences the subject's awareness of the real environment.

According to Endsley, situational awareness is the awareness of information and knowledge associated with each situation [24]. This includes the perception of all relevant variables of a situation, the understanding of these variables and the projection of them into the future. One way to measure this situational awareness is a SAGAT test (Situation Awareness Global Assessment Technique). A SAGAT's basic concept is that the subject is given a task while also being asked to recall various parameters of the current situation. The experiment is then interrupted in the middle of the task (simulation freeze), and all situation-dependent parameters are hidden. The proband's situational awareness is assessed by asking him to recall these hidden parameters.

6.2.2. Planned procedure

The planned user study would have been taken place at the premises of the DLR in Braunschweig. To ensure similar conditions for each test run, all experiments would have been conducted in the same large room. Since the sensors and the tracking of the HoloLens' position and orientation is influenced by light, a room without windows was chosen. The contrast of the virtual content on the HoloLens displays also depends on the lighting in the room. These external interferences can be avoided by using constant and controlled lighting. It would have been ensured that these conditions were the same for all subjects in each run.

In all experiments, we rely on a web map service streaming the landscape data to the HoloLenses over the network. It is ensured that the transfer rate between HoloLens and the server is as constant as possible. The computer hosting the web map service is directly connected to a router via a Gigabit Ethernet connection. This router broadcasts the Wi-Fi network to which the HoloLenses are connected. No other devices are connected to this network to prevent any external device from interfering with the connection.

At the beginning of the user study, each participant is first led into a separate room. The participant will then be asked to perform a SARS-CoV2 test following DLR's hygiene concept. If this test is negative, the subject is allowed to enter the room where the user study is being conducted. A note to be signed by the subject informs him or her that basic personal data may be used anonymously for this work and that the subject may stop the user study at any time (see appendix A.5). Afterwards, basic information about the person is recorded, including age and gender. Since the need for glasses can influence the perception of virtual content, this is also noted. In addition, the technical affinity of the test persons is assessed with the help of a modified TA-EG questionnaire (see appendix A.1). This questionnaire determines the factors *enthusiasm* and *expertise* with new technologies. Furthermore, the probands are asked about their previous experience with AR and VR applications. With a Likert scale the expertise in these technologies is determined. The demographic questionnaire can be found in the appendix A.6.

Since the distance between the eyes influences the 3D perception of virtual content, the HoloLens is calibrated for each participant before starting the experiments.

Each proband performs all four experiments. Both the order of the experiments and the conditions within an experiment are randomly selected to avoid habituation effects. The subject performs the same task with two different conditions in each experiment, except for experiment three, where three different conditions are examined. After each condition, the perceived workload during the task and the usability of the presented feature are recorded using the NASA TLX, SUS and QUESI questionnaire. In addition, the proband has the opportunity to give feedback on the presented feature in a comment field. A break of 5 minutes is scheduled between each experiment to avoid possible eye strain.

6.2.3. Experiment 1: Spatial relationship in 3D augmented reality

The first experiment of the user study aims to investigate how the immersive three-dimensional landscape visualization influences spatial perception. For this purpose, two different conditions are tested and compared:

- **C1:** The user is shown the landscape, including markers in immersive AR on HoloLens. In this condition, the proband uses the HoloLens to inspect the landscape. He has the option to rotate the landscape using the gesture described in section 5.4.1.4. Panning and zooming is not needed in this task and is therefore not relevant. A marker is selected by grasping it with the hands.

- **C2:** The user is shown the landscape model and markers two-dimensionally on a computer screen. The proband inspects the landscape on a single 24-inch full HD monitor. The user can move in the virtual scene using keys on the keyboard, and the viewing direction is controlled using the mouse. A marker can be selected by clicking on it.

The following null hypotheses are investigated in this experiment:

- $H_{1.1}$**:** Visualizing a landscape section in immersive AR does not affect the spatial perception.
- $H_{1.2}$**:** Visualizing a landscape section in immersive AR does not affect the total workload.
- $H_{1.3}$**:** Visualizing a landscape section in immersive AR does not affect the usability.

For both conditions, the application is prepared so it is already in the analysis phase and the landscape is visible 1.5 meters in front of the user. The landscape model has a fixed radius of 0.75 meters. The digital elevation model used to represent the landscape is the DEM model published by Ganci et al. [52]. Since the color texture of the landscape surface is not relevant for this experiment, the landscape surface is represented in white.

This experiment aims to evaluate how the spatial perception of markers on the landscape model varies with the presentation in AR or two-dimensional on a screen. The proband is given the task of finding and selecting the nearest three markers to a chosen reference marker. Before the experiment, the user can make himself familiar with the landscape interactions.

In total, ten consecutive runs are performed for each candidate. The user confirms his selection and start the next run or end the run with a button press. To avoid habituation effects, the distribution of the markers and the reference marker is randomly reselected after each run. In each run, a total of 20 markers are shown on the landscape model simultaneously.

The time needed from the start to the end of the 10th run is measured for each participant. This time is interpreted as an indicator for the spatial perception of the markers on the landscape models. After filling out the questionnaires at the end of the experiment, the subject is asked to continue with the other condition and familiarize himself with it.

6.2.4. Experiment 2: Laser pointer linking

The second experiment of the user study investigates how the laser pointer tool can help providing spatial context and understanding. This tool is used to link a multispectral image with the landscape to help locating the site spatially. In this experiment, two different conditions are considered:

- **C1:** The subject is allowed to use all implemented tools, including the laser pointer tool.
- **C2:** The subject is only allowed to use the drawing tool.

The following null hypotheses are examined in this experiment:

- $H_{2.1}$: Using the linking tool does not affect the time required to locate features.
- $H_{2.2}$: Using the linking tool does not affect the accuracy while locating features.
- $H_{2.3}$: Using the linking tool does not affect the total workload required to locate features.
- $H_{2.4}$: Using the linking tool does not affect the usability while locating features.

This experiment consists of ten rounds, in each of which the subject has the same task. Every round a randomly selected multispectral image on the landscape is presented to the subject. In this image, the subject is shown a randomly selected point from a collection of possible points. The subject has the task of locating this highlighted point on the 3D landscape model as accurately as possible. The landscape model can be zoomed, rotated or moved. The subject marks the chosen point on the landscape with the drawing tool. When the proband is satisfied with his selection, he can continue by pressing a button. Before showing the next feature to be located, a 10 second delay is added to allow the proband to rest his arm.

After each round, the time used and the accuracy of the selection are measured and saved. When the proband has completed all ten rounds, the workload and usability are measured using the NASA TLX, the SUS and the QUESI questionnaire. Subsequently, the user is asked to start with the other condition.

6.2.5. Experiment 3: Panorama exploration

The third experiment deals with the hypothesis explained in section 4.4.4.1. This experiment aims to determine, if different photosphere visualizations influence the user's situational awareness. Three different conditions are tested:

- **C1:** The subject is surrounded by a panoramic image projected on a photosphere. The image content is fully visible.
- **C2:** In the second condition, the subject can control the visible part of the panoramic image using the flashlight tool. The focus of this flashlight is set to 25 degrees.
- **C3:** In the third condition, the subject can fully control the flashlight's orientation and focus.

The following null hypotheses are investigated in this experiment:

- $H_{3.1}$: Using the different panorama display modes does not affect the time required to find features.
- $H_{3.2}$: Using the different panorama display modes does not affect the workload required to find features.
- $H_{3.3}$: Using the different panorama display modes does not affect the usability while locating features.
- $H_{3.4}$: Using the different panorama display modes does not affect the situational awareness while locating features.

The task setup for this experiment is illustrated in figure 6.1.

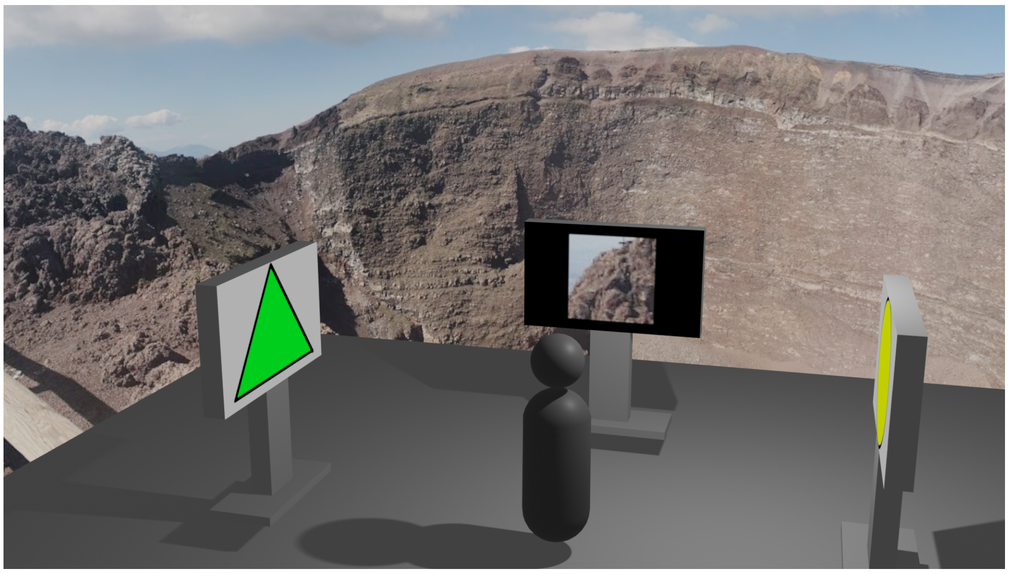


Figure 6.1.: Setup for user study experiment 3: The subject is standing in-between three monitors. Surrounding the user the photosphere is visible (Condition C1).

In this task the user is given the HoloLens with an application showing a multispectral panorama. He has now the task of finding specific features in the panorama image. These features are shown on a 30-inch monitor located about 3 meters in front of the subject. Each feature is a patch of 50 x 50 pixels of the panorama image. The task is to find this feature in the photosphere and mark it with the drawing tool. As soon as the proband confirms his choice by pressing a submit button, the next feature is shown on the screen.

According to Endsley, a key principle of testing the situational awareness is the simulation freeze in the middle of the task [24]. During this interruption, the user can not see task-relevant parameters. Instead, the subject is asked for the last parameters he remembers.

To measure a user's situational awareness in his real environment, a second task is given is given in parallel. Two additional monitors, one to the left and one to the right, are placed in the room. On each of these two monitors, a different colored geometric shape is shown. Every 10 seconds, a new shape and color is randomly selected from a collection of six distinct geometric shapes and four colors (see appendix A.7). The user has the task to remember the currently displayed colors and shapes.

During the simulation freeze, the HoloLens display and the screens are turned off at a certain time. The user is now asked about the feature and the geometries' color and shape. Six different features are shown on the screen. The subject has to select the feature he is currently searching and tell the right combination of shape and color. After each freeze, a new panorama image is shown to the subject. The number of correct answers and the time taken for locating each feature is measured and logged. After ten freezes, the experiment is finished, and the subject is asked to fill out the NASA TLX, SUS and QUESI questionnaires.

Afterwards, the user is asked to start the experiment again with one of the two remaining conditions.

6.2.6. Experiment 4: Multi-user collaboration

The fourth experiment of the user study investigates, if the implemented collaboration tools affect the time, workload or usability.

Two different conditions are considered in this experiment:

- **C1:** All probands are connected in a virtual session and are allowed to use the implemented collaboration tools.
- **C2:** All probands are in different sessions.

The following null hypotheses are examined in this experiment:

- $H_{4.1}$: Using the multi-user collaboration tools does not affect the time required to locate features collaboratively.
- $H_{4.2}$: Using the multi-user collaboration tools does not affect the total workload required to locate features collaboratively.
- $H_{4.3}$: Using the multi-user collaboration tools does not affect the usability when locating features collaboratively.

In condition C1, three probands are in the same virtual session. The shared landscape is placed in the same physical location for all participants. For condition C2, the probands, each in separate sessions with their own landscape model, are placed next to each other with a distance of four meters. The application is then started for each user and set up so that the landscape appears 1.5 meters in front of each subject.

The same landscape section of the environment around Mount Etna [52] is used in both conditions. This landscape model is colored using images from the Landsat 7 satellite of that region. In total, an area of 10 x 10 kilometers is covered. At the start of the experiment, a landscape area with a radius of 2.5 kilometers is shown on the model. The probands can navigate the landscape by rotating, panning and zooming, using the implemented interactions. Over the entire area of 10 x 10 kilometers, 15 different markers are placed for this experiment:

- 5 mono multispectral images
- 3 stereo multispectral images
- 3 panorama multispectral images
- 4 3D scans

Because each subject tests both conditions, different images and 3D scans are used for each condition.

Before the experiments starts, each user can test the interactions to familiarize themselves with the interaction gestures. In this test run, the subject is shown the same landscape area with one 3D scan, one panoramic image, one mono and one stereo multispectral image. The probands are taught how to navigate the landscape and select markers. Also, the presented filter, laser and drawing tools are introduced. Each proband can try these interactions until he feels confident using them.

Before starting the experiment, the subjects are given a list of six items to find. An exemplary list can be found in appendix A.8. Multispectral images are printed on the list. 3D scans are depicted with a uniquely identifiable rendered image.

The subjects' task in both conditions is to find and open the markers containing the elements shown on the list as quickly as possible. If the probands are satisfied with their selection, they can use a button to end their run.

Afterwards, each participant is asked to fill in the NASA TLX, QUESI and SUS questionnaire individually. Subsequently, the probands complete the same task in the other condition.

6.2.7. Results and Discussion

This section explains how the results collected in the user study would have been interpreted, if it had been conducted successfully.

The participants in the planned user study are expected to consist mainly of students and employees of the DLR in Braunschweig. It can be assumed that even, if the restrictions imposed by Corona had not made the study impossible, the number of participants would have been in the range of 10 to 20 people. Due to the relatively small number of subjects and the neglect of individual factors (age, gender, visual acuity), the study would have been considered a *within-group study*.

For the analysis of the results, a series of standard significance tests are used to prove or disprove the null hypotheses. The significance of the measurements is evaluated with the t-test and the Wilcoxon test. In addition, the Shapiro-Wilk test is used to prove the normality of the individual conditions for each experiment.

6.2.7.1. Statistical evaluation methods

In the following, these tests are briefly explained and discussed why they are relevant for the presented study.

T-Test: The paired t-test is a standardized test in mathematical statistics [138]. It is used to make a statement about the differences between two datasets. The means of two datasets are compared to each other and checked, if their difference is statistically significant. The differences in the same person's data but recorded under distinct conditions are compared. Two fundamental assumptions are made about the recorded data:

- The datasets were recorded by a randomly selected group of persons.
- The datasets are normally distributed.

The *two-sided* variant of the t-test, not only proves or disproves the null hypothesis. The test also checks the validity of the alternative hypotheses.

As a well-known statistical test, this t-test is implemented in Microsoft Excel. The inputs are the datasets D_a and D_b, recorded in the test run with conditions A and B. The t-statistic and the p-value are returned as a result.

The t-statistic provides information about the significance of the null hypothesis. The p-value indicates whether there is a significant difference between the two datasets. This p-value is compared with a fixed significance level α. If the p-value is less than α, then it can be assumed that there is a significant difference. This indicates that the null hypothesis can be rejected. Typically, a significance level of 0.05 is set as α.

Wilcoxon Test: The Wilcoxon test is another method for determining statistical significance [18, p. 350]. It is assumed that the two datasets to be compared were collected from the same group of participants. In contrast to the t-test, the data is not required to be normally distributed for this test. Therefore, it is often used to evaluate studies with fewer users or expected significant outliers. The function provided by Microsoft Excel is also used for the calculation. The t-statistic and the p-value are returned as a result and compared to a significance level as described above.

Shapiro-Wilk Test: The Shapiro-Wilk test is considered in this evaluation as a pre-evaluation to the t-test [132]. It is used to ensure that the data sets to be processed come from a normally distributed population. Since this is one of the basic assumptions of the t-test, the Shapiro test is checked before the t-test. This ensures that the t-test results are valid.

In addition to the statistical tests, each dataset's mean and standard deviation is calculated and compared.

6.2.7.2. Discussion

Experiment 1: During the first experiment, the user has to select the nearest neighbors of a given marker as quickly as possible. For the quantitative evaluation, the time needed for all ten runs is measured in this experiment. The measurement starts when the first marker is highlighted and ends when the user finishes the last run. This time is written to a file, which is then statistically analyzed in Excel using the tests described above.

Since the study could not be conducted, no real data has been collected. Thus, the null hypotheses of this experiment can not be refuted.

However, preliminary tests with a few people showed that especially the height differences of markers in AR, are significantly better noticeable due to the 3D visualization. Further, the perspective on the landscape in AR can be changed more intuitively by head movement. A user can change his view from the top-down perspective to a lateral view to recognize height differences more clearly simply by moving his head.

At the same time, the benefit of condition C2 is, that many users are already familiar with the concept of manipulating a virtual camera with a mouse and keyboard from other 3D computer programs and video games. Even if this can not be confirmed without user study, it is assumed that with the 3D immersive representation of the landscape, spatial relationships and height differences can be perceived significantly faster than two-dimensionally displayed on the monitor.

Regarding hypothesis $H_{1.2}$, it can be argued, that exploring and selecting of markers with hands leads to significantly more physical strain in contrast to mouse and keyboard. In AR the user has to move his head or even his whole body to change his perspective on the landscape. At the computer he only moves his hands in minimal motions to control the application.

Regarding hypothesis $H_{1.3}$, it can be argued, that the system's usability depends on the user's experience with the input and output methods. It can be expected that most probands are familiar with the visualization of three-dimensional applications on a two-dimensional monitor. Consequently, using mouse and keyboard to control the perspective on the virtual scene is expected to be familiar to most probands. But for many subjects using AR offers a new and unknown way of inspecting three-dimensional virtual content. Each participant is encouraged to test the system before the actual run. Although these test runs reduce the effect of prior experience with the input and output methods, it is expected that the users experience still influences the subjective usability of the system.

Even though this could not be confirmed experimentally, it is assumed that, the user can interact more intuitively with the landscape model in AR after sufficient prior training. A user can change his perspective on the virtual content in AR in the same way he does in the real world just by moving the head. Furthermore, the interactions with landscape and markers are designed so that the virtual objects behave like real physical objects. Because of these parallels to interactions with real-world objects, it can be assumed that direct interaction is more intuitive than using mouse and keyboard as conventional input devices after a certain amount of training.

Experiment 2: The second experiment investigates, if the laser tool can be used to locate features in multispectral images on the landscape model faster and more accurate. The probands are given ten features on the image, which they have to locate on the landscape model as accurately and as quickly as possible.

During this experiment, the task completion time and the accuracy of the markings are measured. The timer starts, when the first feature of a multispectral image is highlighted and ends when the user confirms his selection by pressing the button in the last run. The accuracy is determined by the euclidean distance between the target position and the user's marking.

Due to Corona pandemic restrictions, no study could be conducted for this experiment, and thus no evaluable data could be collected. Therefore, none of the null hypotheses can be disproved or confirmed.

Based on preliminary user feedback, a positive trend to use the linking tool was found. By back-projecting the laser beam onto the three-dimensional landscape and highlighting the camera frustum on the model, the search area for the feature on the landscape could be reduced. Thus, it can be assumed that the user could find the feature on the landscape model faster (null hypothesis $H_{2.1}$).

However, the feedback also showed that using the laser pointer for several minutes can lead to arm fatigue. In order to point at something specific, the user must lift and hold the arm, which is physically demanding. As long as the laser tool is actively used only for a few seconds at a time, this effect does not occur. It is assumed that in the practical use of such a tool, the user will rarely point to something specific for a more extended time period but only for a few seconds at a time. Between the individual runs, a break of 10 seconds is integrated into the experiment procedure to allow the probands to rest their arm.

To determine the usability and the total workload, the subjects in this experiment would have filled out the NASA TLX, SUS and QUESI questionnaires. Due to the additional hand movement with the laser tools (condition C1), it can be assumed that the physical workload is larger than in condition C2. Further, it can be argued that the mental workload might be lower in condition C1 than in condition C2 due to the narrowing of the search area.

Experiment 3: The third experiment investigates how the three presented approaches of visualizing panoramic images influence the situational awareness in the real and virtual environment. Three different conditions, C1, C2 and C3, are tested. With the help of a SAGAT procedure, the situational awareness of the proband in his real environment is evaluated. For the quantitative evaluation, the number of correctly answered questions in the SAGAT and the time required for the experiment is recorded. These recorded values are then transferred from the HoloLens and evaluated in Excel using the significance tests described above.

The timer starts when the first feature is shown to the subject and ends when he confirms his selection of the last feature with the button. The SAGAT score is the number of correct answers during the freeze. To answer correctly, the proband needs to recall the shape and color of both geometries and the correct feature.

Due to Corona, none of the hypotheses has been tested and therefore could not be confirmed or refuted.

Informal feedback, showed that with condition C1, the recognition of objects in the real environment around the user was difficult while inspecting a bright panoramic image. In dark areas, objects in the real environment were clearly visible. The real environment is perceived much more consciously when only specific areas of the photosphere are visible (condition C2 and C3). It was remarked that exploring the panoramic image in these conditions is physically more difficult since the user has to move his hands to explore other parts of the image. The feedback indicates that condition C1 is more beneficial when the primary task is to explore the panoramic image. When the user's focus should be on both the panoramic image and his real-world surroundings, like in a collaborative scenario, either condition C2 or C3 is desirable.

In preliminary tests, no difference in the time required can be identified when conditions C2 and C3 are directly compared. According to the feedback, the default flashlight angle of 25 degrees (condition C2) was perceived as too narrow. When investigating condition C3, the flashlight's focus was often set once at the start of the experiment, and then the users attempted to maintain the same focus. One user showed frustration when testing condition C3 because his focus changed due to unintentional hand movement. The informal feedback showed that condition C2 is preferred over condition C3.

Experiment 4: The fourth experiment investigates a co-located collaboration scenario. Two different conditions C1 and C2 are tested. The probands have the task to locate a set of given scientific findings. This experiment aims to find out how the task completion time, the workload, and the system's usability differ between the two conditions.

The quantitative value measured during the experiment is the task completion time. For this purpose, time measurement is started from the moment the users are given the list. As soon as the users select all the items from the list, the timer is stopped. Qualitative and quantitative data are analyzed in Excel using the described statistical tests.

Due to Corona restrictions, the null hypotheses of this experiment could neither be confirmed nor disproved.

This test was conducted informally with two people. Since only one HoloLens was available at this time, the test procedure was slightly modified. Instead of both persons wearing a HoloLens during the experiment, only one person uses the HoloLens. The second person participates in the virtual session with the prototype running on an ordinary computer using mouse and keyboard as input and a monitor as an output device.

In total, condition C1 was tested once and then condition C2 was tested twice with these two individuals. In all conditions and test runs a new list of items and markers has been used to prevent learning effects. In the two runs with condition C2, two different strategies emerged that the subjects employed. In the first strategy, the two subjects split the items on the list. The first subject tried to find the items 1 to 3, and the second subject searched for items 4 to 6. In the second strategy, the list was worked through synchronously, item by item. Both experts searched for the same finding at the same time.

In condition C1, there was no talking between the two persons at all. Both subjects were focused on their task and found the markers in 2 minutes and 54 seconds and 2 minutes and 43 seconds. In condition C2, the two users found the six markers within 2 minutes 33 seconds using the first strategy and in the second run within 1 minute and 46 seconds with the second strategy.

When comparing the two strategies of condition C2, the subjects communicated less with each other in the first strategy than in the second strategy. Neither in condition C1 nor in condition C2 did the two subjects use the function of a local working copy.

Preliminary feedback from these informal tests suggests that working together on a task can reduce the time required. Splitting the tasks in a co-located collaboration scenario (first strategy condition C2) shows only a slightly reduced task completion time compared to solving the task alone (condition C1). This could be because no local working copy of the landscape was created in the test. Both subjects pursued different goals and interfere with each other's actions on the shared landscape model. Working synchronously (second strategy condition C2) the two users were able to complete the task as quickest.

7. Summary and future work

This thesis considers methods and their implementation to integrate AR visualizations in the collaborative scientific analysis of georeferenced rover data. 3D interaction and visualization approaches for the rover data itself, as well as the surrounding landscape, are discussed.

It is proved that large-scale and high-resolution terrain datasets can be rendered in interactive frame rates, even on the limited memory and computational capacities of the Microsoft HoloLens.

Using on-demand terrain data streaming allows seamless exploration of large landscapes. With efficient caching and swapping algorithms, individual parts of terrain data are dynamically loaded and unloaded. That way, even complex datasets with hundreds of gigabytes in size can be explored on the HoloLens. Visual quality is enhanced using computer graphics approaches without increasing geometric complexity. An interaction concept based on tangible user interfaces and direct manipulation allows a collaborative exploration of the landscape.

The landscape model is used as a central tool for browsing the rover data. Markers on the landscape model represent the scientific measurements of the rovers. Three main categories are considered: multispectral imagery, 3D scans of soil samples and rocks, and chemical material analysis. For each of these data types, a visualization method in AR is presented, implemented, and discussed in the context of an efficient collaborative analysis session. The implemented 3D interaction and visualization concepts in AR aim to emphasize the spatial context between the rover's measurements and their surrounding landscape.

Introducing various tools for selecting and filtering rover data by type, acquisition time, and position enable efficient exploration of large rover datasets. Additional tools for collaborative analysis of the available data are intended to stimulate discussion among experts. Approaches of shared and local workspaces have been continued. They can be used to obtain both an individual and a shared view on the landscape and the scientific results.

It could be shown experimentally that even in a demanding scenario with parallel rendering of two different landscape sections and the simultaneous analysis of several scientific findings, the average frame rate is above 30 Hertz. Therefore it can be used interactively.

The concepts implemented in this work were to be evaluated in an expert interview and a user study. Due to the COVID-19 restrictions during the development of this thesis, neither the interview nor the study could be conducted. Therefore, it was not possible to qualitatively evaluate the presented concepts for their applicability, usability and intuitiveness.

A preliminary informal user feedback shows a generally positive trend for the use of AR in a collaborative analysis process. It shows that the immersive 3D representation of the landscape provides a better spatial impression of the rover's measurement locations compared to a 2D presentation.

Potential for future optimizations has also been identified. To improve the concepts developed in this prototype, they must first be tested in a realistic application scenario. Although the preliminary informal user feedback shows promising results, it is important to evaluate the applicability of the prototype in expert interviews. Planetary scientists analyse rover and landscape data on a daily basis. They can best assess the opportunities but also the challenges of integrating such an AR prototype into their daily work. As soon as the current pandemic situation allows, the planned expert interview, as well as the user study, will be conducted and evaluated.

An important milestone will be the ARCHES demo mission in summer of 2022. This will be the first time that real data collected by the rovers will be available to the application. The data used to test the prototype were georeferenced images and 3D scans, but they were not collected by the actual rover hardware.

Using the implemented approaches for dynamic swapping and reloading of landscape data, landscape areas of unlimited size can be visualized interactively on the HoloLens. The quality of the visual representation is still limited by its computational and main memory capacity. To continue to improve the quality in the future, the main memory needs to be used even more efficiently. A promising approach to do this is texture compression discussed in section 5.4.2. While BC7 compression can avoid most artifacts in the textures, the compression of image content on the HoloLens is not real-time capable. This problem could be solved by compressing the image data into BC7 format on an external computer and then sending it to the application. Other algorithms could be used to further compress the texture files. A promising group of compression algorithms are *GPU-decodable Supercompressed Textures* [77]. These can reduce the memory size of already compressed textures by up to 50% while maintaining the same quality. They achieve comparable compression rates to JPEG, while being decodable in real time on the graphics card.

Another promising approach to memory optimization is the use of *Streaming Virtual Textures* (SVT) [46, 111]. In the current implementation, the terrain data is stored in a 3 x 3 tile neighborhood as a texture. Only in the rarest of cases is the entire texture required for rendering. By visualizing the graphics memory, SVT enables individual blocks of a texture to be swapped in and out of the graphics memory on the persistent storage dynamically and hardware-accelerated. While Unreal Engine already supports this feature for desktop systems, the implementation for mobile devices is still experimental and not yet compatible with HoloLens 2. In the future, this approach could enable significantly better memory management and thus the rendering of higher-resolution landscape data.

Currently, the landscape surface is modeled as a height displaced plane. The curvature of the celestial body is not considered in the visualization. This is a good approximation for the local investigation of certain areas here on Earth. However, when looking at much smaller celestial bodies, this curvature becomes increasingly relevant. Appropriate tessellations of the landscape surface, such as Healpix tessellation [144, 58], could be used to save memory and computational power by providing a higher resolution for central areas than for areas near the edge.

Another limitation of the current prototype is the visualization of the landscape surface as a DEM based surface model. While this approach allows using a performant shader-based vertex displacement, it has its limitations. Each pixel in the DEM maps a geocoordinate to exactly one height value. However, in the real terrain on Earth caves, overhangs or trees exist. Those can not be represented with this surface model. To visualize these terrain features, the import, storage and efficient rendering of the terrain as a complete 3D model or even as a point cloud reconstruction instead of a DEM is a challenge for the future.

The thesis promotes collaborative discussion and analysis in a co-located scenario. This means that all experts are in the same real-world environment. However the current corona pandemic has shown that it is not always possible to work all together in person. Topics such as home office, telepresence and remote collaboration are more relevant than ever. The ideas of the current application's interaction methods are based on the premise that the experts can communicate directly with each other, verbally and non-verbally. For future remote collaboration scenarios, it would be necessary to investigate how effectively the developed approaches - like avatars and laser pointers - can be used, even if each expert is in a different real environment.

The two main aspects of this thesis - rendering of large-scale, high-resolution terrain data and the integration and visualization of georeferenced data in AR - could be transferred to other application and research areas in the future. In addition to the analysis of collected rover data, it would be feasible to extend the prototype to collaborative mission planning. The next steps of the rover and possible landing sites for future rover missions, could be discussed collaboratively in AR with a high-resolution landscape representation and the appropriate tools. Even outside the application area of space exploration, such an immersive representation of georeferenced data could be effectively applied in the fields of urban planning, weather simulation and archaeology.

Bibliography

[1] Dan Adams. *No Limits - Unity in Cross Industry Development*. June 2014. URL: `https://blog.unity.com/community/no-limits-unity-in-cross-industry-development` (visited on 02/08/2022).

[2] Adobe. *Select with the Lasso tool*. Aug. 2020. URL: `https://helpx.adobe.com/photoshop/using/selecting-lasso-tools.html` (visited on 02/08/2022).

[3] Martin Agan, LeeAnn Voisinet, and Ann Devereaux. "NASA's Wireless Augmented Reality Prototype (WARP)". In: *AIP Conference Proceedings*. Vol. 420. 1. American Institute of Physics. 1998, pp. 236–242.

[4] Eswar Anandapadmanaban. "VMCC: a virtual reality framework for augmenting mission control operations". PhD thesis. Massachusetts Institute of Technology, 2020.

[5] Eswar Anandapadmanaban et al. "Holo-SEXTANT: an augmented reality planetary EVA navigation interface". In: 48th International Conference on Environmental Systems. 2018.

[6] Eva Artinger, Martin Schanzenbach, and Florian Echtler. "Alternative multitouch gestures for map interaction". In: *ACM International Conference on Interactive Tabletops and Surfaces*. 2010, pp. 297–297.

[7] Eva Artinger et al. "Exploring multi-touch gestures for map interaction in mass casualty incidents." In: *GI-Jahrestagung*. 2011, p. 274.

[8] Christopher R Austin et al. "Elicitation study investigating hand and foot gesture interaction for immersive maps in augmented reality". In: *Cartography and Geographic Information Science* 47.3 (2020), pp. 214–228.

[9] Ronald Azuma et al. "Recent advances in augmented reality". In: *IEEE computer graphics and applications* 21.6 (2001), pp. 34–47.

[10] Sarah J Ball. "Effect of Augmented Reality on Anxiety in Prelicensure Nursing Students". PhD thesis. Walden University, 2018.

[11] Aaron Bangor, Philip T Kortum, and James T Miller. "An empirical evaluation of the system usability scale". In: *Intl. Journal of Human–Computer Interaction* 24.6 (2008), pp. 574–594.

[12] James F Blinn. "Simulation of wrinkled surfaces". In: *ACM SIGGRAPH computer graphics* 12.3 (1978), pp. 286–292.

[13] Sebastian Boring, Marko Jurmu, and Andreas Butz. “Scroll, tilt or move it: using mobile phones to continuously control pointers on large public displays”. In: *Proceedings of the 21st Annual Conference of the Australian Computer-Human Interaction Special Interest Group: Design: Open 24/7*. 2009, pp. 161–168.

[14] Thomas Brinkhoff et al. “Measuring the Complexity of Polygonal Objects.” In: *ACM-GIS*. Vol. 109. Citeseer. 1995.

[15] David K Broberg. “Guidance for horizontal image translation (HIT) of high definition stereoscopic video production”. In: *Stereoscopic Displays and Applications XXII*. Vol. 7863. International Society for Optics and Photonics. 2011, 78632F.

[16] Lei Chen et al. “Effect of visual cues on pointing tasks in co-located augmented reality collaboration”. In: *Symposium on Spatial User Interaction*. 2021, pp. 1–12.

[17] Jonathan Cohen, Marc Olano, and Dinesh Manocha. “Appearance-preserving simplification”. In: *Proceedings of the 25th annual conference on Computer graphics and interactive techniques*. 1998, pp. 115–122.

[18] William Jay Conover. *Practical nonparametric statistics*. Vol. 350. john wiley & sons, 1999.

[19] Open Geospatial Consortium. *OGC GeoTIFF standard*. Sept. 2019. URL: `http://docs.opengeospatial.org/is/19-008r4/19-008r4.html` (visited on 02/08/2022).

[20] Open Geospatial Consortium. *Web Map Service*. URL: `https://www.ogc.org/standards/wms` (visited on 02/08/2022).

[21] Andrea Corradini, Richard M Wesson, and Philip R Cohen. “A map-based system using speech and 3d gestures for pervasive computing”. In: *Proceedings. Fourth IEEE International Conference on Multimodal Interfaces*. IEEE. 2002, pp. 191–196.

[22] Martin D. Crossley. “A guide to coordinate systems in Great Britain”. In: 1999.

[23] Docker. *Empowering App Development for Developers | Docker*. URL: `https://www.docker.com/` (visited on 02/08/2022).

[24] Mica R Endsley. “Situation awareness global assessment technique (SAGAT)”. In: *Proceedings of the IEEE 1988 national aerospace and electronics conference*. IEEE. 1988, pp. 789–795.

[25] Douglas C Engelbart. “Augmenting human intellect: A conceptual framework”. In: *Menlo Park, CA* (1962).

[26] Douglas C Engelbart. “Program on human effectiveness”. In: *Reprint in: Nyce, James M./Kahn, Paul (1991)(Hg.)(1962)* (1962), pp. 237–244. URL: `https://archive.org/details/1962-program-on-human-effectiveness`.

[27] Wito Engelke et al. “Scientific Visualization for Space Science Data Analysis in Collaborative Virtual Environments”. In: *IEEE Visualization 2015* (2015).

[28] Jet Propulsion Laboratory Eric Ramirez. *Shuttle Radar Topography Mission - The Mission to Map the World.* URL: `https://www2.jpl.nasa.gov/srtm/datacoverage.html` (visited on 02/08/2022).

[29] Tom G Farr and Mike Kobrick. "Shuttle Radar Topography Mission produces a wealth of data". In: *Eos, Transactions American Geophysical Union* 81.48 (2000), pp. 583–585.

[30] Randima Fernando and Mark J. Kilgard. *The Cg Tutorial: The Definitive Guide to Programmable Real-Time Graphics.* USA: Addison-Wesley Longman Publishing Co., Inc., 2003. ISBN: 0321194969.

[31] Apache Software Foundation. *The Apache HTTP Project.* Dec. 21. URL: `https://httpd.apache.org/` (visited on 02/08/2022).

[32] Open Source Geospatial Foundation. *Welcome to MapServer.* Jan. 2022. URL: `https://mapserver.org/` (visited on 02/08/2022).

[33] Mike Welles Frank Warmerdam Andrey Kiselev and Dwight Kelly. *LibTIFF - TIFF Library and Utilities.* URL: `http://www.libtiff.org/` (visited on 02/08/2022).

[34] Philippe Fuchs. *Virtual reality headsets-a theoretical and pragmatic approach.* CRC Press, 2017.

[35] Epic Games. *Actors.* URL: `https://docs.unrealengine.com/4.27/en-US/ProgrammingAndScripting/ProgrammingWithCPP/UnrealArchitecture/Actors/` (visited on 02/08/2022).

[36] Epic Games. *An Overview of the Unreal Engine product.* URL: `https://www.unrealengine.com/en-US/unreal` (visited on 02/08/2022).

[37] Epic Games. *An overview of the Unreal Engine product.* URL: `https://www.unrealengine.com/en-US/unreal` (visited on 02/08/2022).

[38] Epic Games. *Blueprint Visual Scripting.* URL: `https://docs.unrealengine.com/4.27/en-US/ProgrammingAndScripting/Blueprints/` (visited on 02/08/2022).

[39] Epic Games. *Custom Expressions.* URL: `https://docs.unrealengine.com/4.27/en-US/RenderingAndGraphics/Materials/ExpressionReference/Custom/` (visited on 02/08/2022).

[40] Epic Games. *Introduction to C++ Programming in UE4.* URL: `https://docs.unrealengine.com/4.27/en-US/ProgrammingAndScripting/ProgrammingWithCPP/IntroductionToCPP/` (visited on 02/08/2022).

[41] Epic Games. *Level Editor.* URL: `https://docs.unrealengine.com/4.27/en-US/BuildingWorlds/LevelEditor/` (visited on 02/08/2022).

[42] Epic Games. *Levels.* URL: `https://docs.unrealengine.com/4.27/en-US/Basics/Levels/` (visited on 02/08/2022).

[43] Epic Games. *Project Anywhere XR.* URL: `https://docs.unrealengine.com/4.27/en-US/Resources/Showcases/ProjectAnywhereXR/` (visited on 02/08/2022).

[44] Epic Games. *Project Anywhere XR*. URL: https://docs.unrealengine.com/4.27/en-US/BuildingWorlds/Landscape/TechnicalGuide/ (visited on 02/08/2022).

[45] Epic Games. *Static Mesh Component*. URL: https://docs.unrealengine.com/4.27/en-US/Basics/Components/StaticMesh/ (visited on 02/08/2022).

[46] Epic Games. *Streaming Virtual Texturing*. URL: https://docs.unrealengine.com/4.27/en-US/RenderingAndGraphics/VirtualTexturing/Streaming/ (visited on 02/08/2022).

[47] Epic Games. *Texture Compression Settings*. URL: https://docs.unrealengine.com/4.27/en-US/RenderingAndGraphics/Textures/TextureCompressionSettings/ (visited on 02/08/2022).

[48] Epic Games. *The Many Uses of Unreal Engine For All Industries*. URL: https://www.unrealengine.com/en-US/solutions/more-uses (visited on 02/08/2022).

[49] Epic Games. *Unreal Engine 4.27 Release Notes*. URL: https://docs.unrealengine.com/4.27/en-US/WhatsNew/Builds/ReleaseNotes/4_27/ (visited on 02/08/2022).

[50] Epic Games. *Unreal Insights*. URL: https://docs.unrealengine.com/4.27/en-US/TestingAndOptimization/PerformanceAndProfiling/UnrealInsights/ (visited on 02/08/2022).

[51] Epic Games. *UProceduralMeshComponent*. URL: https://docs.unrealengine.com/4.27/en-US/API/Plugins/ProceduralMeshComponent/UProceduralMeshComponent/ (visited on 02/08/2022).

[52] Gaetana Ganci et al. *Digital Elevation Model of Mt Etna updated to 18 December 2015*. data set. Supplement to: Ganci, G et al. (2018): Mapping Volcanic Deposits of the 2011–2015 Etna Eruptive Events Using Satellite Remote Sensing. Frontiers in Earth Science, 6:83, https://doi.org/10.3389/feart.2018.00083. 2019. DOI: 10.1594/PANGAEA.899140. URL: https://doi.org/10.1594/PANGAEA.899140.

[53] Arturo S Garcia et al. "A collaborative workspace architecture for strengthening collaboration among space scientists". In: *2015 IEEE Aerospace Conference*. IEEE. 2015, pp. 1–12.

[54] Arturo S Garcia et al. "Collaborative virtual reality platform for visualizing space data and mission planning". In: *Multimedia Tools and Applications* 78.23 (2019), pp. 33191–33220.

[55] Andreas Gerndt. "CosmoScout VR: Interactivity and Immersion for Space Data Exploration and Mission Planning". In: *AR/VR for European Space Programmes*. 2019. URL: https://elib.dlr.de/146855/.

[56] Gimp. 2.4. *Free Selection (Lasso)*. URL: https://helpx.adobe.com/photoshop/using/selecting-lasso-tools.html (visited on 02/08/2022).

[57] Google. *Google Earth VR*. URL: https://arvr.google.com/earth/ (visited on 02/08/2022).

[58] Krzysztof M Gorski et al. “HEALPix: A framework for high-resolution discretization and fast analysis of data distributed on the sphere”. In: *The Astrophysical Journal* 622.2 (2005), p. 759.

[59] Mark S Hancock et al. “Rotation and translation mechanisms for tabletop interaction”. In: *First IEEE International Workshop on Horizontal Interactive Human-Computer Systems (TABLETOP’06)*. IEEE. 2006, 8–pp.

[60] Sandra G Hart. “NASA-task load index (NASA-TLX); 20 years later”. In: *Proceedings of the human factors and ergonomics society annual meeting*. Vol. 50. 9. Sage publications Sage CA: Los Angeles, CA. 2006, pp. 904–908.

[61] Sandra G Hart and Lowell E Staveland. “Development of NASA-TLX (Task Load Index): Results of empirical and theoretical research”. In: *Advances in psychology*. Vol. 52. Elsevier, 1988, pp. 139–183.

[62] Nicholas R Hedley et al. “Explorations in the use of augmented reality for geographic visualization”. In: *Presence* 11.2 (2002), pp. 119–133.

[63] Kaj Helin et al. “User experience of augmented reality system for astronaut’s manual work support”. In: *Frontiers in Robotics and AI* 5 (2018), p. 106.

[64] Juan David Hincapie-Ramos et al. “Consumed endurance: a metric to quantify arm fatigue of mid-air interactions”. In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. 2014, pp. 1063–1072.

[65] Chi-Chang Hsieh et al. “Photo navigator”. In: *Proceedings of the 16th ACM international conference on Multimedia*. 2008, pp. 419–428.

[66] Jörn Hurtienne and Anja Naumann. “QUESI—A questionnaire for measuring the subjective consequences of intuitive use”. In: *Interdisciplinary College* 536 (2010).

[67] Corinna Jacobs. *Interactive panoramas: techniques for digital panoramic photography*. Vol. 1. Springer Science & Business Media, 2004.

[68] Brian Karis. “Real Shading in Unreal Engine 4 by”. In: 2013.

[69] Główny Urząd Geodezji i Kartografii. *Digital Elevation Model (DEM)*. URL: `https://www.geoportal.gov.pl/en/dane/numeryczny-model-terenu` (visited on 02/08/2022).

[70] Hirokazu Kato et al. “Virtual object manipulation on a table-top AR environment”. In: *Proceedings IEEE and ACM International Symposium on Augmented Reality (ISAR 2000)*. Ieee. 2000, pp. 111–119.

[71] Frederic Kerber, Antonio Krüger, and Markus Löchtefeld. “Investigating the effectiveness of peephole interaction for smartwatches in a map navigation task”. In: *Proceedings of the 16th international conference on Human-computer interaction with mobile devices & services*. 2014, pp. 291–294.

[72] Khronos. *Fragment Shader*. Nov. 2020. URL: `https://www.khronos.org/opengl/wiki/Fragment_Shader` (visited on 02/08/2022).

[73] Khronos. *OpenXR*. URL: https://www.khronos.org/api/index_2017/openxr (visited on 02/08/2022).

[74] Khronos. *Rendering Pipeline Overview*. Feb. 2021. URL: https://www.khronos.org/opengl/wiki/Rendering_Pipeline_Overview (visited on 02/08/2022).

[75] Kiyoshi Kiyokawa et al. "Communication behaviors of co-located users in collaborative AR interfaces". In: *Proceedings. International Symposium on Mixed and Augmented Reality*. IEEE. 2002, pp. 139–148.

[76] Tomas Kot, Petr Novak, and Jan Bajak. "Using HoloLens to create a virtual operator station for mobile robots". In: *2018 19th International Carpathian Control Conference (ICCC)*. IEEE. 2018, pp. 422–427.

[77] Pavel Krajcevski, Srihari Pratapa, and Dinesh Manocha. "GST: GPU-decodable supercompressed textures". In: *ACM Transactions on Graphics (TOG)* 35.6 (2016), pp. 1–10.

[78] Songyang Lao et al. "A gestural interaction design model for multi-touch displays". In: *People and Computers XXIII Celebrating People and Technology* (2009), pp. 440–446.

[79] Benjamin Lee et al. "Shared Surfaces and Spaces: Collaborative Data Visualisation in a Co-located Immersive Environment". In: *IEEE Transactions on Visualization and Computer Graphics* 27 (2 2021), pp. 1171–1181. ISSN: 2160-9306. DOI: 10.1109/TVCG.2020.3030450.

[80] Sangyoon Lee and Hong Hua. "Effects of viewing conditions and rotation methods in a collaborative tabletop AR environment". In: *IEEE transactions on visualization and computer graphics* 17.9 (2011), pp. 1245–1258.

[81] Jon Derek Loftis et al. "Using lidar elevation data to develop a topobathymetric digital elevation model for sub-grid inundation modeling at langley research center". In: *Journal of coastal Research* 76 (10076) (2016), pp. 134–148.

[82] Deutsches Zentrum für Luft- und Raumfahrt. *Arches Projekt Details*. URL: https://www.arches-projekt.de/projekt-arches/arches-projekt-details/ (visited on 02/08/2022).

[83] Deutsches Zentrum für Luft- und Raumfahrt. *Demomission Weltraum*. URL: https://www.arches-projekt.de/demomission/weltraum/ (visited on 02/08/2022).

[84] Mona Lütjens et al. "Virtual reality in cartography: Immersive 3D visualization of the Arctic Clyde Inlet (Canada) using digital elevation models and bathymetric data". In: *Multimodal Technologies and Interaction* 3.1 (2019), p. 9.

[85] Pina Meisel. *Microsoft HoloLens kommt nach Europa*. Oct. 2016. URL: https://news.microsoft.com/de-de/microsoft-hololens-kommt-nach-europa/ (visited on 02/08/2022).

[86] Lars Knoke Michael Zawrel. *Zum Mond und zurück: Wie Microsoft HoloLens 2 beim Bau der NASA-Raumkapsel Orion hilft.* Sept. 2020. URL: https://news.microsoft.com/de-de/microsoft-hololens-2-hilft-beim-bau-der-nasa-raumkapsel-orion/ (visited on 02/08/2022).

[87] Microsoft. *About HoloLens 2.* Nov. 2021. URL: https://docs.microsoft.com/en-us/hololens/hololens2-hardware (visited on 02/08/2022).

[88] Microsoft. *Bing Maps.* URL: https://www.bing.com/maps/ (visited on 02/08/2022).

[89] Microsoft. *Choosing your engine.* Nov. 2021. URL: https://docs.microsoft.com/en-us/windows/mixed-reality/develop/choosing-an-engine?tabs=unity (visited on 02/08/2022).

[90] Microsoft. *Direct manipulation with hands.* Nov. 2021. URL: https://docs.microsoft.com/en-us/windows/mixed-reality/design/direct-manipulation (visited on 02/08/2022).

[91] Microsoft. *Gaze and commit.* Nov. 2021. URL: https://docs.microsoft.com/en-us/windows/mixed-reality/design/gaze-and-commit (visited on 02/08/2022).

[92] Microsoft. *Gaze and dwell.* Jan. 2021. URL: https://docs.microsoft.com/en-us/windows/mixed-reality/design/gaze-and-dwell (visited on 02/08/2022).

[93] Microsoft. *Geometry Shader Stage.* May 2021. URL: https://docs.microsoft.com/en-us/windows/win32/direct3d11/geometry-shader-stage (visited on 02/08/2022).

[94] Microsoft. *Hologram stability.* Oct. 2021. URL: https://docs.microsoft.com/en-us/windows/mixed-reality/develop/advanced-concepts/hologram-stability (visited on 02/08/2022).

[95] Microsoft. *HoloLens Research Mode.* Oct. 2021. URL: https://docs.microsoft.com/en-us/windows/mixed-reality/develop/advanced-concepts/research-mode (visited on 02/08/2022).

[96] Microsoft. *Introducing instinctual interactions.* Nov. 2021. URL: https://docs.microsoft.com/en-us/windows/mixed-reality/design/interaction-fundamentals (visited on 02/08/2022).

[97] Microsoft. *Microsoft HoloLens 2.* URL: https://www.microsoft.com/en-gb/hololens (visited on 02/08/2022).

[98] Microsoft. *Point and commit with hands.* May 2021. URL: https://docs.microsoft.com/en-us/windows/mixed-reality/design/point-and-commit (visited on 02/08/2022).

[99] Microsoft. *Share and present content from Skype Meetings App (Skype for Business Web App).* URL: https://support.microsoft.com/en-gb/office/share-and-present-content-from-skype-meetings-app-skype-for-business-web-app-234b0c06-a88d-4707-904c-4fd6c571fc01#ID0EBBD=Skype_Meetings_App (visited on 02/08/2022).

[100] Microsoft. *Spatial anchors.* Oct. 2021. URL: https://docs.microsoft.com/en-us/windows/mixed-reality/design/spatial-anchors (visited on 02/08/2022).

[101] Microsoft. *Tesselation Stages.* Sept. 2020. URL: https://docs.microsoft.com/en-us/windows/win32/direct3d11/direct3d-11-advanced-stages-tessellation (visited on 02/08/2022).

[102] Microsoft. *Unreal Development Overview.* Sept. 2021. URL: https://docs.microsoft.com/en-us/windows/mixed-reality/develop/unreal/unreal-development-overview?tabs=ue426%2Cmrtk%2Casa%2CD365 (visited on 02/08/2022).

[103] Microsoft. *Unreal Development Overview.* Sept. 2021. URL: https://docs.microsoft.com/en-us/windows/mixed-reality/develop/unreal/unreal-development-overview (visited on 02/08/2022).

[104] Microsoft. *Use HoloLens (1st gen) devices with Dynamics 365 Guides.* Nov. 2021. URL: https://docs.microsoft.com/en-us/dynamics365/mixed-reality/guides/hl1 (visited on 02/08/2022).

[105] Microsoft. *Versionshinweise für HoloLens 2.* Dec. 2021. URL: https://docs.microsoft.com/de-de/hololens/hololens-release-notes (visited on 02/08/2022).

[106] Microsoft. *Vertex Shader Stage.* Aug. 2020. URL: https://docs.microsoft.com/en-us/windows/win32/direct3d11/vertex-shader-stage (visited on 02/08/2022).

[107] Microsoft. *Voice input.* May 2021. URL: https://docs.microsoft.com/en-us/windows/mixed-reality/design/voice-input (visited on 02/08/2022).

[108] Microsoft. *What are the UX Tools?* Oct. 2021. URL: https://github.com/microsoft/MixedReality-UXTools-Unreal (visited on 02/08/2022).

[109] Microsoft. *What is Graphics Tools?* Aug. 2021. URL: https://github.com/microsoft/MixedReality-GraphicsTools-Unreal (visited on 02/08/2022).

[110] Paul Milgram et al. "Augmented reality: A class of displays on the reality-virtuality continuum". In: *Telemanipulator and telepresence technologies.* Vol. 2351. International Society for Optics and Photonics. 1995, pp. 282–292.

[111] Martin Mittring and Crytek GmbH. "Advanced virtual texture topics". In: *ACM SIGGRAPH 2008 Games.* 2008, pp. 23–51.

[112] NASA. *GeoTIFF.* Mar. 2021. URL: https://earthdata.nasa.gov/esdis/eso/standards-and-references/geotiff (visited on 02/08/2022).

[113] Anja Naumann and Jörn Hurtienne. “Benchmarks for intuitive interaction with mobile devices”. In: *Proceedings of the 12th international conference on Human computer interaction with mobile devices and services*. 2010, pp. 401–402.

[114] Florian Niebling and Marc E. Latoschik. “Browsing Spatial Photography Using Augmented Models”. In: Munich, Germany. Munich, Germany: IEEE, 2018, pp. 47–48. ISBN: 978-1-5386-7593-9. DOI: `10.1109/ISMAR-Adjunct.2018.00031`.

[115] Jeffrey Norris. *Sidekick: Investigating Immersive Visualization Capabilities*. URL: `https://www.nasa.gov/mission_pages/station/research/experiments/explorer/Investigation.html#id=2018` (visited on 02/08/2022).

[116] NVIDIA. *Don’t be conservative with Conservative Rasterization*. Nov. 2014. URL: `https://developer.nvidia.com/content/dont-be-conservative-conservative-rasterization` (visited on 02/08/2022).

[117] Manuel M Oliveira and Fabio Policarpo. “An efficient representation for surface details”. In: *Instituto de Informatica UFRGS* (2005).

[118] Bui Tuong Phong. “Illumination for computer generated pictures”. In: *Communications of the ACM* 18.6 (1975), pp. 311–317.

[119] Fábio Policarpo, Manuel M Oliveira, and Joao LD Comba. “Real-time relief mapping on arbitrary polygonal surfaces”. In: *Proceedings of the 2005 symposium on Interactive 3D graphics and games*. 2005, pp. 155–162.

[120] PROJ. *PROJ 8.2.1 documentation*. URL: `https://proj.org/` (visited on 02/08/2022).

[121] Qualcomm. *Snapdragon 850 Mobile Compute Platform*. URL: `https://www.qualcomm.com/products/snapdragon-850-mobile-compute-platform` (visited on 02/08/2022).

[122] Sarah Radway et al. “Beyond LunAR: An augmented reality UI for deep-space exploration missions”. In: *arXiv preprint arXiv:2011.14535* (2020).

[123] Brett Ridel et al. “The Revealing Flashlight: Interactive Spatial Augmented Reality for Detail Exploration of Cultural Heritage Artifacts”. In: *ACM Journal on Computing and Cultural Heritage* 7.2 (2014), 6:1–6:18. DOI: `10.1145/2611376`.

[124] Wolfgang Riede. *Laser-based Stand-off Detection*. URL: `https://www.dlr.de/tp/Portaldata/39/Resources/Handout_LIBS.pdf` (visited on 02/08/2022).

[125] Caroline E Robertson et al. “Neural representations integrate the current field of view with the remembered 360 panorama in scene-selective cortex”. In: *Current Biology* 26.18 (2016), pp. 2463–2468.

[126] Jannick P Rolland and Henry Fuchs. “Optical versus video see-through head-mounted displays in medical visualization”. In: *Presence* 9.3 (2000), pp. 287–309.

[127] Susana Ruano et al. “Augmented reality tool for the situational awareness improvement of UAV operators”. In: *Sensors* 17.2 (2017), p. 297.

[128] Kadek Ananta Satriadi et al. "Augmented Reality Map Navigation with Freehand Gestures". In: *2019 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)*. 2019, pp. 593–603. DOI: `10.1109/VR.2019.8798340`.

[129] Stephanie Schierholz. *NASA, Microsoft Collaborate to Bring Science Fiction to Science Fact*. June 2015. URL: `https://www.nasa.gov/press-release/nasa-microsoft-collaborate-to-bring-science-fiction-to-science-fact` (visited on 02/08/2022).

[130] Martin J. Schuster et al. "The ARCHES Space-Analogue Demonstration Mission: Towards Heterogeneous Teams of Autonomous Robots for Collaborative Scientific Sampling in Planetary Exploration". In: *IEEE Robotics and Automation Letters* 5.4 (Oct. 2020). Ed. by Jonathan Roberts, pp. 5315–5322. URL: `https://elib.dlr.de/135622/`.

[131] C.E. Shannon. "Communication in the Presence of Noise". In: *Proceedings of the IRE* 37.1 (Jan. 1949), pp. 10–21. DOI: `10.1109/jrproc.1949.232969`. URL: `https://doi.org/10.1109/jrproc.1949.232969`.

[132] Samuel Sanford Shapiro and Martin B Wilk. "An analysis of variance test for normality (complete samples)". In: *Biometrika* 52.3/4 (1965), pp. 591–611.

[133] Ben Shneiderman. "The future of interactive systems and the emergence of direct manipulation". In: *Behaviour & Information Technology* 1.3 (1982), pp. 237–256.

[134] Noah Snavely, Steven M Seitz, and Richard Szeliski. "Photo tourism: exploring photo collections in 3D". In: *ACM siggraph 2006 papers*. 2006, pp. 835–846.

[135] John P Snyder. *Flattening the earth: two thousand years of map projections*. University of Chicago Press, 1997.

[136] Spire. *What is WMS?* URL: `https://spire.com/wiki/spire-weather-web-map-service/` (visited on 02/08/2022).

[137] Richard Stoakley, Matthew J Conway, and Randy Pausch. "Virtual reality on a WIM: interactive worlds in miniature". In: *Proceedings of the SIGCHI conference on Human factors in computing systems*. 1995, pp. 265–272.

[138] Student. "The probable error of a mean". In: *Biometrika* (1908), pp. 1–25.

[139] Simon Su et al. "Sensor data fusion framework to improve holographic object registration accuracy for a shared augmented reality mission planning scenario". In: *International Conference on Virtual, Augmented and Mixed Reality*. Springer. 2018, pp. 202–214.

[140] Zsolt Szalavari et al. ""Studierstube": An environment for collaboration in augmented reality". In: *Virtual Reality* 3.1 (1998), pp. 37–48.

[141] Yosuke Takata, Ichi Kanaya, and Kosuke Sato. "Photo Browsing System for Sharing Information in Archaeological Research". In: Jan. 2008.

[142] Carlo Tomasi. "A Simple Camera Model". In: (2017).

[143] JMP Van Waveren and Ignacio Castaño. “Real-time YCoCg-DXT compression”. In: *Tech. Rep., id Software, Inc. and NVIDIA Corp., 2007* 2 (2007), p. 3.

[144] Rolf Westerteiger, Andreas Gerndt, and Bernd Hamann. “Spherical Terrain Rendering using the hierarchical HEALPix grid”. In: *Visualization of Large and Unstructured Data Sets: Applications in Geospatial Planning, Modeling and Engineering-Proceedings of IRTG 1131 Workshop 2011*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik. 2012.

[145] Studio Wildcard. *Ragnarok - ARK Expansion Map*. June 2017. URL: `https://store.steampowered.com/app/642250/Ragnarok__ARK_Expansion_Map/` (visited on 02/08/2022).

[146] Don Willems and Louis Vuurpijl. “Pen gestures in online map and photograph annotation tasks”. In: *Behavior Genetics - BEHAV GENET* (Oct. 2006).

[147] Gang Xu and Zhengyou Zhang. *Epipolar geometry in stereo, motion and object recognition: a unified approach*. Vol. 6. Springer Science & Business Media, 1996.

[148] Suya You and Charles. K. Thompson. *Mobile collaborative mixed reality for supporting scientific inquiry and visualization of earth science data*. 2017. DOI: `10.1109/vr.2017.7892266`.

[149] Zhengyou Zhang. “A flexible new technique for camera calibration”. In: *IEEE Transactions on pattern analysis and machine intelligence* 22.11 (2000), pp. 1330–1334.

A. Appendix

	Synchronized actor	Synchronized properties	
Avatar	Head	· 3D Transformation · Connection status	
Avatar	Hand	· 3D Transformation · Gripping status · Avatar ID · Active tool	
	Landscape	· Zoom level, · Centre coordinate · Heading · Selected DEM layer · Selected color layers	
Rover data visualization	Multispectral images	· Selection status · 3D transformation · Currently selected filter · Augmented overlay status	
Rover data visualization	Multispectral images	Stereo multispectral image	· Left/Right eye selection · Stereo/Mono selection
Rover data visualization	Multispectral images	Multispectral panorama image	· Flashlight transformation
Rover data visualization	3D Scan	· Selection status · 3D transformation · Color texture	
Rover data visualization	Material analysis	· Selection status · 3D transformation	

Table A.1.: Synchronized actors and their properties.

TA-EG questionaire

	Strongly disagree				Strongly agree
1. I inform myself about electronic devices, even if I have no intention to buy.					
2. I love to own new electronic devices.					
3. I am excited when a new electronic device comes on the market.					
4. I like to go to the electronic equipment store.					
5. I enjoy trying out an electronic device.					
6. I know most of the features of the electronic devices I own.					
7. I have or would have problems understanding when reading electronics and computer magazines.					
8. I find it easy to learn how to operate an electronic device.					
9. I know the field of electronic devices.					

NASA Task Load Index

Name	Task	Date

Mental Demand — How mentally demanding was the task?

Very Low — Very High

Physical Demand — How physically demanding was the task?

Very Low — Very High

Temporal Demand — How hurried or rushed was the pace of the task?

Very Low — Very High

Performance — How successful were you in accomplishing what you were asked to do?

Perfect — Failure

Effort — How hard did you have to work to accomplish your level of performance?

Very Low — Very High

Frustration — How insecure, discouraged, irritated, stressed, and annoyed wereyou?

Very Low — Very High

System Usability Scale

	Strongly disagree				Strongly agree
1. I think that I would like to use this system frequently					
2. I found the system unnecessary complex					
3. I thought the system was easy to use					
4. I think that I would need the support of a technical person to be able to use this system					
5. I found the various functions in this system were well integrated					
6. I thought there was too much inconsistency in this system					
7. I would imagine that most people would learn to use this system very quickly					
8. I found the system very cumbersome to use					
9. I felt very confident using the system					
10. I needed to learn a lot of things before I could get going with this system					

QUESI

	Fully disagree	Mainly disagree	Neutral	Mainly agree	Fully agree
I could use the system without thinking about it					
I achieved what I wanted to achieve with the system					
The way the system worked was immediately clear to me					
I could interact with the system in a way that seemed familiar to me					
No problems occurred when I used the system					
The system was not complicated to use					
I was able to achieve my goals in the way I had imagined to					
The system was easy to use from the start					
It was always clear to me what I had to do to use the system					
The process of using the system went smoothly					
I barely had to concentrate on using the system					
The system helped me to completely achieve my goals					
How the system is used was clear to me straight away					
I automatically did the right thing to achieve my goals					

User Study consent form

Please read and sign this form.

In this user study:

- You will perform given tasks using the Microsoft HoloLens
- You will be asked to fill out questionnaires after finishing the tasks
- You will be asked to think loud about the actions you are currently doing

I have read the safety briefing and fully understand the risks: yes ☐ no ☐

Participation in this user study is voluntary. All personal information will be anonymized. At no time will your name or other identification be used. You can withdraw your consent to the user study and stop participating in the experiments at any time without giving reasons. The results of this user study will be used in the context of a master thesis.

If you have any questions, please contact Jan Wulkop at: Jan.Wulkop@dlr.de

I have read and understood the information on this form.

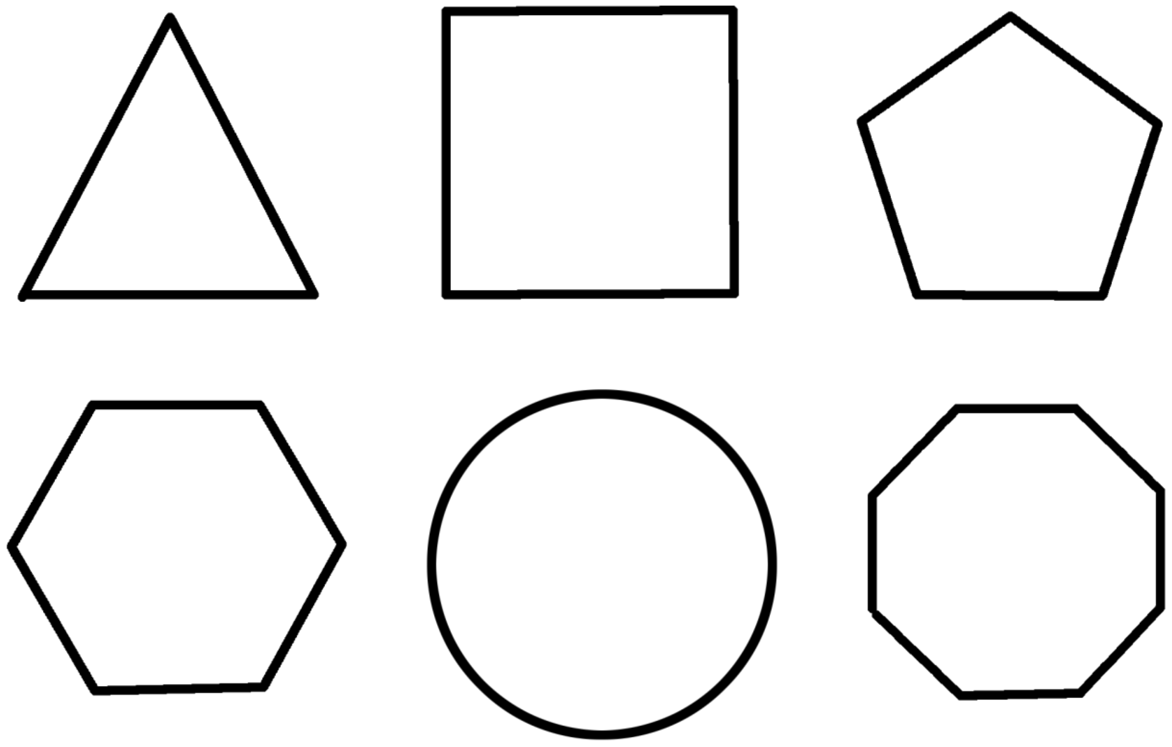
Subject's signature	Date

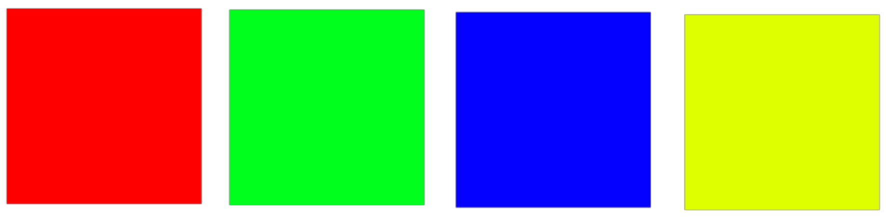
Demographic Questionnaire

1. What is your gender? Male ☐ Female ☐
2. What is your age?
3. Do you wear a visual aid? No ☐ Glasses ☐ Contacts ☐
4. If yes: are you nearsighted or farsighted? Near ☐ Far ☐

			1	2	3	4	5	
5	Do you have experience with VR/AR?	No experience	☐	☐	☐	☐	☐	Much experience
6	Dou you have experience with HoloLens?	No experience	☐	☐	☐	☐	☐	Much experience

Experiment 3: geometric shapes and color

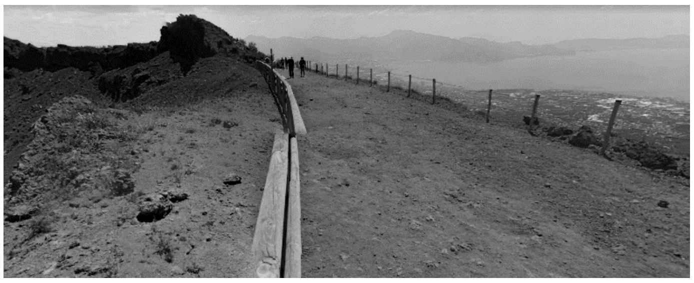




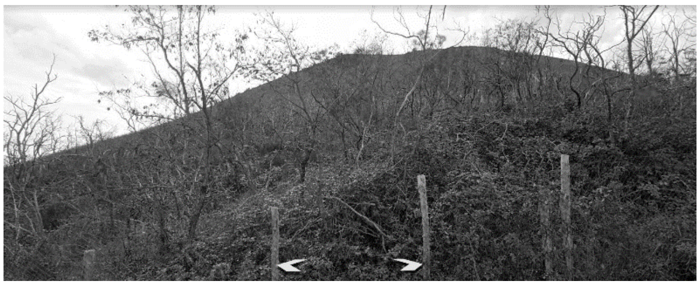
Experiment 4:

Task: Find and open only the markers describing the following six scientific measurements:

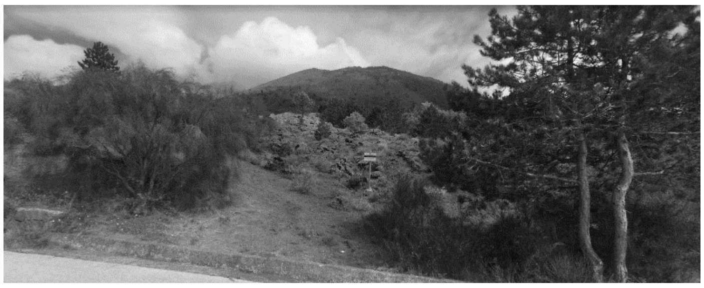
1.



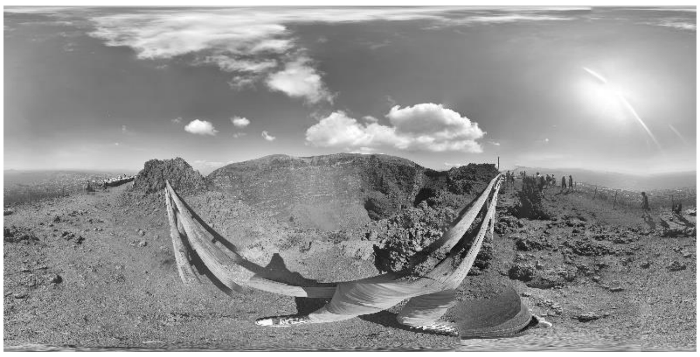
2.

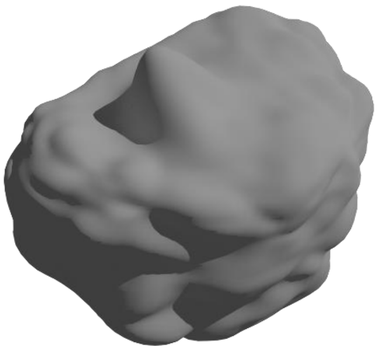


3.

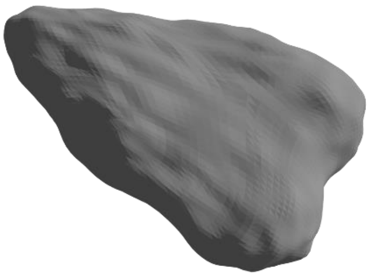


4.





5.



6.

Master Thesis Project

Planjahr	Verantwortliches Institut	Projektleiter/Vorhabensverantwortlicher	Datum
2021	SC	Cesar de Albuquerque Richers, Georgia	15.11.2021

Schwerpunkt	Raumfahrt
Forschungsgebiet	Interactive Visualization
Teilgebiet	3D Interaction in AR and VR

Projekt-/Vorhabenstitel

Visualization of scientific data in multi-user augmented reality	**Dauer**	6 Months

Beteiligte Institute und Einrichtungen

Institut	Verantwortlich
DLR, SC-SRV – Supervisor	Cesar de Albuquerque Richers, Georgia
TU Braunschweig – Supervisor	Martin Eisemann
TU Braunschweig –Student	Jan Wulkop

1 Project description and rationale

In the field of aerospace technology research and development are an international and distributed process. Engineers coming from different backgrounds and research areas are working together on interplanetary missions. In the future, missions to other celestial bodies will have to be planned. Collected scientific results have to be analyzed in order to gain detailed insight for further mission planning in an unknown environment. In future missions rovers will be used to explore and collect scientific data from unknown areas of celestial bodies.

The autonomous exploration of unknown areas with a group of robotic systems is a core aspect of the ARCHES (Autonomous Robotic Networks to Help Modern Societies) project. As part of this project the German aerospace center is developing robotic systems which are capable of conducting scientific measurements autonomously. Multispectral images, soil sample material spectroscopy as well as 3D reconstructions of collected samples and the environment surrounding the rover are the primary results collected by these rovers.

The analysis process of this kind of scientific measurements is a complex task and is still done by humans. Planet scientists coming from different backgrounds have to evaluate these scientific data in order to get further insights about the investigated areas. In order to support a successful discussion between the experts the visualization of these measurement results is a key aspect. A good visualization can help avoiding misunderstandings between the experts and could enrich the discussions between them.

Immersive Augmented Reality (AR) has the potential to revolutionize the way we work in the industry. With devices like Microsoft HoloLens we can visualize 3D models in real world space, interact with them, and overlay them on the top of the real world objects. The benefits of augmented reality could streamline the process of data analysis in many areas. AR allows the landscape immediately surrounding the rover to be blended into the real world as a 3D hologram. The user can freely move around the landscape and perceive the data in real 3D. When analyzing the data, the spatial context of the site to the landscape and other previously recorded data is also relevant. Blending of the site's locations on the 3D landscape in AR could improve the perception of the spatial context.

The interface of the application should be as intuitive as possible, both for the visualization of the landscape and for the scientific results. 3D input methods are useful for the interaction between the user and the 3D hologram. The Microsoft HoloLens glasses enable the precise location of the user's fingers in three-dimensional space. This information can be used in an application to enable direct interaction between the user and virtual content using the user's hands.

Experts from various disciplines are involved in the analysis of the recorded scientific data. Due to the different scientific backgrounds of the experts, misunderstandings cannot be avoided. A direct discussion among the experts helps to clarify these misunderstandings. With the help of augmented reality, experts can work together on site on the same virtual hologram and discuss the results directly.

The goal of this work is to investigate whether the use of augmented reality methods is useful for collaborative analysis of science rover data. It will be investigated whether the data analysis can be made more intuitive, time-saving or user-friendly by taking advantage of augmented reality.

2 Goals and expected outcome, Milestones

The proposed work would include the following steps:

2.a Investigating existing scientific data visualization techniques in Immersive Augmented Reality

In this phase the student has to explore the current object manipulation techniques used in virtual and augmented reality especially for immersive AR devices like Microsoft HoloLens. The student would also explore the HoloLens software development platform and the hand tracking equipment.

2.b Implement and benchmark landscape visualization techniques

In this phase the student will benchmark existing landscape visualization approaches for the HoloLens. In order to give a realistic representation of the surrounding landscape, we would like to visualize the highest possible high level of detail given the limited computing power of the HoloLens.

2.c Investigate existing solutions for managing large terrain data

As a mobile, stand-alone device, the HoloLens device is very limited both in its available computing power and in its memory capacity. At the same time, terrain and satellite data can be very large in terms of data size, depending on the level of detail. Ideally, the application should have loaded the highest meaningful level of detail of the terrain data for the landscape section currently being displayed. This requires that the landscape data is filtered by position and level of detail and can be dynamically loaded and unloaded. The student will investigate existing techniques and approaches for filtering large terrain data.

2.d Investigate User Interface design principles in immersive Augmented reality

In order to not only visualize holograms in augmented reality, but also to interact with them, a user interface must be provided by the application. Similar to classical 2D user interfaces, there are also different design principles and metaphors for 3D user interfaces. In this phase the student will get an overview of the differences between 2D and 3D user interfaces and their principles. Subsequently, a reasoned user interface concept suitable for the use case should be selected and integrated.

2.e Implement viewer for the different types of scientific rover data

One goal of this work is to visualize in augmented reality how scientific data collected by the rover. These data can be divided into 3 main categories: 3D scans, material analysis and images from the rover cameras. The images can be mono or stereo, each covering a limited wavelength range. At the same time, several images will be stitched together to form a 360 degree panoramic image, which is also to be visualized in augmented reality. The student's task is to investigate and then implement existing techniques for rendering these data formats in AR and VR.

2.f Implement tools for an immersive shared collaboration experience

Another focus of this thesis is to improve the collaborative work among experts by using augmented reality techniques as a tool. Several experts will work together on the analysis of the data. In order to enable a common interaction with the virtual content, a tool chain will be developed, which ideally allows to understand the intentions of the other experts faster in a discussion. At the same time, an expert should also be able to focus on a problem alone without being influenced or irritated by the other experts. In this step, the student will explore the already known benefits and challenges of shared virtual collaboration.

2.g Evaluation and expert interview

After the implementation of the points listed in the previous section, the influence of the selected augmented reality approaches will be evaluated with the help of expert interviews and user studies. The focus will be on the question of whether the selected visualization and interaction techniques can make the workflow for analyzing the data more intuitive, faster and more pleasant for the user. For this purpose, the experts will test out a scenario in the application and then express their opinion with the help of a questionnaire and an open discussion. Depending on how the pandemic situation develops at the end of the processing time, a user study can be carried out to find out how intuitive the user interface is for a non-expert.

Milestones:

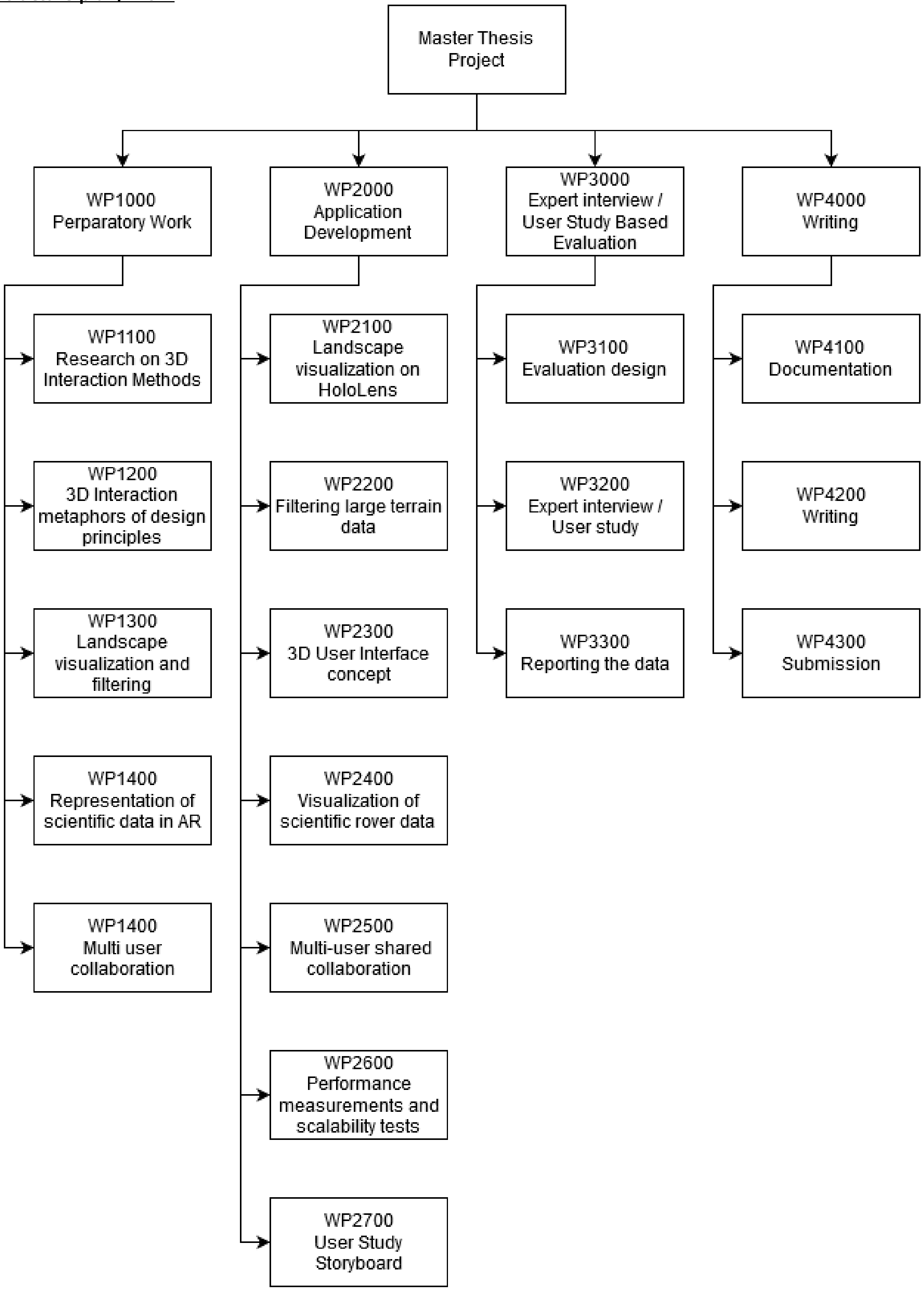
1. Milestone: until 09.08.2021
    - Agreement between the University Supervisor, DLR Supervisor and Student about the topic and expected content of the thesis
    - Official start of the thesis period
2. Milestone: until 09.11.2021

- Literature review of:
    - 3D interaction techniques
    - landscape visualization approaches
    - terrain data filtering
    - scientific data visualization in AR/VR
    - rover mission planning
- Implementation of multi-user landscape and data visualization

3. Milestone: until 30.11.2021
    - Contacted experts for expert interview
    - Sent invitations for non-expert user study (if allowed)
    - Prepared questionnaire and open questions for expert interview
    - Prepared user study questionnaire
    - User study design
4. Milestone: until 30.12.2021
    - Conduct expert interview / user study
    - Compiling the results of the expert interview / user study and performing significance tests
    - Improving the results from feedback
5. Milestone: until 15.01.2022
    - First complete draft of thesis
6. Milestone: until 08.02.2022
    - Submission of thesis
    - Final presentation at DLR

3 Organization

3.a Structure plan, Work


Master Thesis Project
WP1000 Perparatory Work
WP2000 Application Development
WP3000 Expert interview / User Study Based Evaluation
WP4000 Writing
WP1100 Research on 3D Interaction Methods
WP1200 3D Interaction metaphors of design principles
WP1300 Landscape visualization and filtering
WP1400 Representation of scientific data in AR
WP1400 Multi user collaboration
WP2100 Landscape visualization on HoloLens
WP2200 Filtering large terrain data
WP2300 3D User Interface concept
WP2400 Visualization of scientific rover data
WP2500 Multi-user shared collaboration
WP2600 Performance measurements and scalability tests
WP2700 User Study Storyboard
WP3100 Evaluation design
WP3200 Expert interview / User study
WP3300 Reporting the data
WP4100 Documentation
WP4200 Writing
WP4300 Submission

3.b Description of Work Packages

WP1000: Preparatory Work

WP1100: Research on 3D Interaction Methods

- Doing research on 3D interaction in virtual and augmented reality
- Determine core features necessary for any implementation

WP1200: Understanding 3D Interaction metaphors of design principles

- Critically evaluate the existing interaction techniques and metaphors used for interaction in HoloLens
- Examine differences between well known 2D design principles and 3D principles
- Work out advantages, disadvantages and limitations of existing 3D user interfaces for our use-cases

WP1300: Landscape visualization and filtering

- Find related work addressing the topic of high detail landscape visualization for mobile devices
- Benchmark different rendering approaches regarding memory usage and frame rate
- Search for well known solutions for managing large-scale terrain data
- Setup Terrain Filtering Service

WP1400: Representation of scientific data in AR

- Investigate related work and user studies for visualization of:
  - 3D scans
  - Multi-spectral images
  - 360 degree images
  - Stereoscopic images
- Develop approaches for further combination of findings between each other:
  - Group findings together based on position
  - Navigation between two neighboring findings
  - Project finding information onto landscape or other findings

WP 1500: Multi user collaboration

- Evaluate multi-user approaches in related AR-applications
  - All virtual elements are shared
  - No virtual element is shared
  - Hybrid approaches
- Work out strategy based on requirements for our use case
- Work out possible multi-user tools to enhance communication and discussion between experts (e.g. laser pointer tool)

WP2000: Application Development

WP2100: Landscape visualization on HoloLens

- Implement, test and benchmark different terrain rendering approaches compatible with Unreal Engine
- Measure memory overhead and framerate in relation to landscape mesh resolution

WP2200: Filtering large terrain data

- Setup and test different map servers
- Implement lazy loading mechanism for HoloLens to request currently relevant terrain data
- Implement caching mechanism
- Implement dynamic loading and unloading of terrain data

- Request new landscape tiles in order to explore surrounding landscape
- Add option to request different level of detail terrain data

WP2300: 3D User Interface concept

- Evaluate and implement UI strategies and metaphors for:
    - Joining/Hosting a shared virtual session
    - Navigate on landscape (panning, zooming, rotation)
    - Open/Close findings
    - Interact with finding visualization
    - Collaborative toolchain

WP2400: Visualization of scientific rover data

- Implement viewer for 3D soil sample scans
- Implement viewer for 2D material analysis
- Implement viewer for 360 degree multi-spectral images
- Implement viewer for 360 degree stereoscopic images
- Implement viewer for mono color image
- Implement viewer for mono multi-spectral images
- Implement viewer for stereoscopic color-image

WP2500: Multi-user shared collaboration

- Develop a strategy on what should be shared among all experts and what should be only locally
- Implement replication of all shared properties and data using Unreal Engine network library
- Implement collection of tools in order to optimize communication between experts (e.g. laser pointer tools)

WP2600: Performance measurements and scalability tests

- Benchmark the implemented application (memory consumption, framerate, network synchronization overhead, lag)
- Variables:
    - Number of participants in shared session
    - Level of detail of terrain data
    - Number of opened finding visualizations at the same time
    - Number of triangles in 3D scans

WP2700: User Study Storyboard

- Make a storyboard that will help evaluate chosen visualization and interaction methods
- Record the results of the expert interview/user study

WP3000: Expert interview / User Study Based Evaluation

WP3100: Evaluation design

- Design the expert interview questions and perform a pilot test
- Identify the user behavior and make any modifications if necessary
- Design evaluation questionnaire to be filled by each user for each experience

WP3200: Expert interview / User study

- Conduct the expert interview / user study
- Gather the logged data and develop inferences about human behavior
- Gather the questionnaire data and transform it into numerical form

AP3300: Reporting the data

- Generate reports and tables of the statistical significance tests on the data
- Report user evaluation and feedback in comparative and quantifiable manner

WP4000: Work related to the thesis

WP4100: Documentation

- Documentation of the project

WP4200: Writing

- Writing contains capturing and reporting all results of the project in the thesis

W45300: Submission

- The scientific thesis has to be created independently and has to be submitted to the University on due date

4 Signatures

Supervisor University	**Supervisor DLR**	**Student**

B. Storage Device

Put a table of the contents of your attached Storage Device (SD card/USB flash drive) here. Add explanations where needed!