University of Bergen
Department of informatics

Interactive Visual Exploration and Analysis of Multi-Modal Geospatial Data Collections

Author: Christian Scheie Hein
Supervisor: Helwig Hauser

UNIVERSITETET I BERGEN
Det matematisk-naturvitenskapelige fakultet

October 2022

Abstract

This thesis presents PanoVis, a geovisualization web application for hikers looking for online hiking inspiration. Hikers can add panoramic imagery to the application, and annotations with information about the mountain peaks visible are added to the panoramas using a combination of manual and automatic techniques. A set of visual components automatically update to reflect the image in view, and hikers can explore multiple images through different interaction methods. These visual components include visualizations of an image location's nearby terrain in two and three dimensions. The thesis discusses various aspects concerning the development of PanoVis, including its design, implementation, and how we evaluate its functions.

Acknowledgments

First and foremost, I would like to thank my supervisor, Helwig Hauser, for his guidance and support throughout the development of the PanoVis system and the writing of this thesis. I would also like to thank my friends Andrè, Gerhard, Kristoffer, Mathias, and Sondre for their help and inspiration in this project. Additionally, I would like to thank my friends and fellow students at the ”Glassburet” study hall for their useful feedback and suggestions concerning the programmatic design of the application and, of course, their general camaraderie. Lastly, I would like to thank my family for their moral support throughout my studies, keeping the stress tolerable.

Christian Scheie Hein

October 23, 2022

Contents

List of Figures

Chapter 1

Introduction

People have looked to the mountains for pleasure, relaxation, or to explore nature for centuries. With the rise of the Internet, hikers use digital media to share their explorations and to be inspired by others' mountainous adventures. Moreover, hiking has recently seen an upswing in popularity [94]. Some countries, such as Norway and Sweden, reported that the majority of their populations hiked at least once during 2021 [41, 55]. Strava, one of the most downloaded fitness tracking applications on iOS and Android, reported a year-on-year increase of 190% in uploads of hiking activities during the Covid-19 pandemic's first 12 months [125].

Hikers frequently bring cameras to their adventures. As a result of the recent improvements to the integrated camera systems of modern smartphones, these have become the camera of choice for many. These high-technological devices include a wide selection of hardware components aiding to capture the best of the outdoors. *Global Positioning System* (GPS) receivers and multiple camera sensors are components commonly included in smartphones of all price ranges, and these are utilized to enable high-resolution photographs supplemented with geographical references to the location where the images were taken. By leveraging the smartphones' powerful processing performance, these devices captures and processes high-resolution imagery at incredible speeds. Using advanced image processing techniques, these handheld computers are able to recognize in-image objects and locate key features, amounting to operations that required super-computers just a few decades ago. Such advancements in processing speeds have led to the development of in-camera image stitching capabilities, which use common key points in multiple images shot from the same scene to create ultra-wide images, henceforth called *panoramas*. Most modern smart phones are capable of creating such images without the need of additional tools other than the standard camera application. Thus, panoramas have become a popular method for visualizing wide scenes, easily accessible for any smartphone user [98].

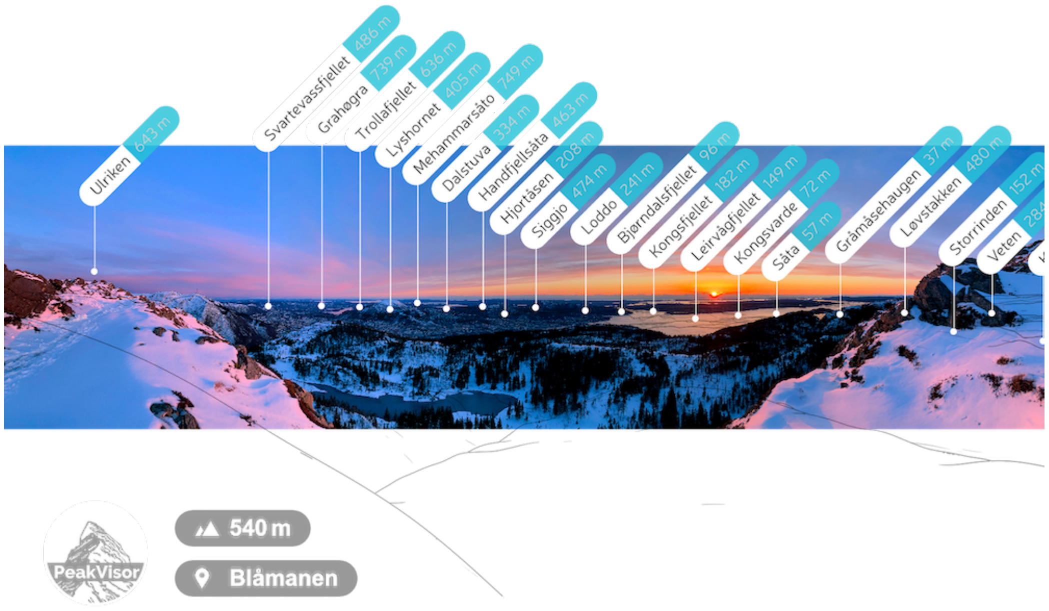


Figure 1.1: A panorama automatically annotated using the PeakVisor [83] web interface. The annotated images include geographical place names, elevations, and contours from the nearby terrain superimposed onto the image.

Capturing panoramas in the mountains can produce images revealing a large portion of the terrain surrounding the photographer, where many mountain peaks are visible. Applications exist which combine the geographical reference of an image with geospatial information about the area to annotate them with information about the visible mountain tops. PeakVisor [83] is one such example, a service that combines a vast amount of geospatial data collections to superimpose relevant information onto the uploaded photographs. Figure 1.1 presents an image annotated using PeakVisor, and includes data about the location of the image and the visible mountains nearby. PeakFinder [104] uses similar data sets to create virtual terrain images that users interactively explore and presents geographically referenced information about the peaks surrounding the virtual viewpoint. Such applications and services may inspire hikers who desire to explore unfrequented areas, giving the users an overview of the terrain and its features.

1.1 The problem addressed in this work

Visually appealing photographs provide insights and inspiration to those seeking adventures in the mountains, and hikers utilize online image services to explore unfrequented terrain in the search for their next hiking adventure. Numerous popular photo-sharing services, such as Flickr, Instagram, and Google Photos, allow uploading ultra-wide images that are multiple times wider than their height. Although these services supports the uploading of images in a wide variety of sizes

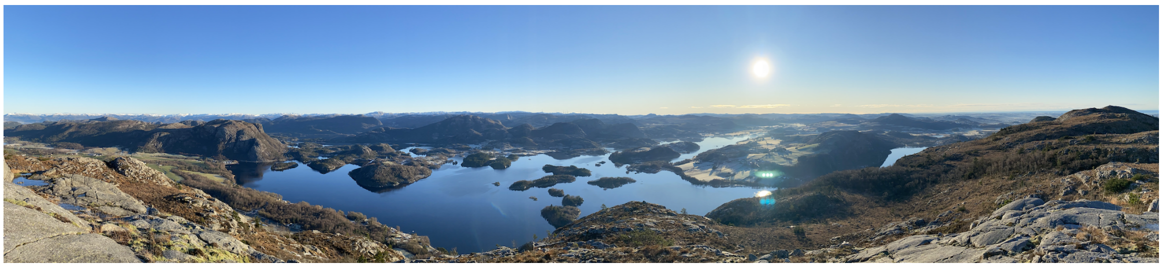


Figure 1.2: A panorama with an approximately 200-degrees wide field of view.

and aspect ratios, all images are commonly treated the same. For the ultra-wide panoramas, this means that the images often are heavily downscaled to fit within the bounds of the screen area, resulting in narrow images that do not reflect the views from the image location. To cope with the narrowness of such images, some services offers special panorama viewers allowing users to pan and zoom to achieve more detailed views of the images.

Illustrating how ultra-wide panoramas weakens the ability to present details when compressed into smaller areas, Figure 1.2 presents an image with such *Field Of View* (FOV). Displaying such ultra-wide imagery on flat surfaces does not justify how the views were at the panorama location, curving straight lines and cramping multiple sky directions into the same image. Panoramas are commonly created by capturing overlapping images by rotating the camera around its vertical axis and stitching them together in post-processing, resulting in images effectively presenting the world captured as a cylinder wrapping around the photographer. Ideally, panoramas could be displayed wrapping around a virtual viewer in a similar way as it was created, and several researchers have proposed methods for displaying images in such a manner [47, 100].

Only looking at an image is not sufficient enough to understand everything a hiker needs to know about the area surrounding the image's shooting location. Photo-sharing services provide additional context to images by exploiting the geographical reference embedded in modern photographs. Converting image coordinates to local place names gives users an indication about the photographer's location at the time of image creation, and plotting these coordinates onto maps enables viewers to get a general overview of the area. Such geographical information can be useful for hikers and works as a starting point for retrieving the information needed to go to the specific area themselves.

For hikers seeking the best views along the trails, knowing whether a particular mountain peak is visible or not from a given viewpoint might be a deciding factor when picking an area to explore. Identifying the different mountain tops displayed in images might be hard without knowing the *camera pose*, meaning the camera's orientation at the time of image creation. Without the direction the camera is pointed or where the photograph was created, it can be a hard task to locate the mountains in topographic maps in the search for additional information about the mountains in the image scene.

Worbs' *Mountainpanorama* [122] combines spherical panoramas with geospatial information about nearby peaks to allow its users to teleport between the different images in a database. Their service solves one of the problems with the more popular photo-sharing services by displaying full-fledged panoramas that encircles the virtual observer, enabling users to look around and explore the terrain through such imagery. On the other hand, the service limits the uploading of new panoramas to a few selected photographers, limiting the usability for hikers not handpicked by the sites moderators.

A hike's degree of difficulty is also a crucial factor for both hiking beginners and the highly experienced ones [75, 89]. It is essential to consider variations in the terrain, adapt the route based on the group's hiking experience level, and avoid natural features such as cliffs or marshes. Properties that describe the peaks and the areas around them, such as the degree of ascent along the mountainsides, typically called *slope*, may be possible for the trained eyes to interpret by examining the height curves on a map. However, such information quickly becomes burdensome to comprehend without earlier experiences from using maps. Therefore, some mapping services like the Norwegian Mapping Authority's Høydedata [40], provide visualization tools for overlaying slope information on interactive maps to ease the interpretation of such data.

A service displaying panoramas encircling the viewpoint while providing important information about the image's shooting location is desirable. It would enable hikers to use ultra-wide imagery as the starting point for their new adventures while also being a tool for exploring their images after completing a hike.

1.2 A geovisualization-based solution

The problems introduced in the previous section called for a solution combining geographically referenced images with terrain data and its features. Such solutions involve data types that are not necessarily of the same format, even though the common denominator is their reference to geography. Due to the geographical aspect of such data, we decided to develop the application based on research from *geographic visualization*. When employed effectively, such visualizations enable interfaces that engage users in exploring geographic information [22].

Geographic visualization, commonly abbreviated as *geovisualization*, was introduced as *cartography* transitioned into the digital era [15]. Cartography is the study and technique of making and utilizing maps and is among the most well-studied visualization methods available to scientists due to the long field time of the domain [66]. When computers evolved to become public possessions, interactivity opened new possibilities and challenges for cartography.

The new possibilities introduced by computers motivated a search for a new name to express the differences between the traditional cartography and this new multimedia cartography, resulting in several new visualization subfields, including geovisualization. It focuses on visually representing spatial data and exploiting available cartographic techniques as part of the interactive graphical representation of the data. The interactive aspect of geovisualization defines a characteristic that separates it from traditional cartography, where the exploratory capabilities are limited [22].

This thesis presents *PanoVis*; a prototype geovisualization web application providing hiking inspiration through panoramic imagery and terrain analysis. Its name is a portmanteau of "panorama" and "visualization," reflecting one of its key features. PanoVis includes functionality for annotating panoramas with information about visible peaks, methods for estimating the images' FOV, and visualizing the terrain surrounding the photographer as three-dimensional surfaces, among others. All the application's features are combined into a seamless geovisualization web interface, providing relevant geospatial information to the users, reducing the need for navigating through multiple sites and services in the search for new hiking adventures.

1.3 Scope

We split the application into three distinct components; *the File Handler*, *the Panorama Aligner*, and *the Panorama Explorer*. By uploading new panoramas to the application using the file handler, it triggers the panorama alingment module and loads this into view. Here, the users are invited to recognize some features in the recently uploaded image, followed by the display of a computer-generated image presenting a digital terrain surrounding the panorama's photographing location. The users are asked to select the same set of feature points that were selected in the panoramic image, a task that might require some local knowledge about the image's motives.

Therefore, we assumed that the images being uploaded to PanoVis would be added by the creator of the panoramas themselves, easing the feature recognition process due to the photographer being more likely to have some local knowledge about the objects visible in the image. Moreover, we assumed that the panorama explorer component could be utilized by hikers seeking inspiration for their next mountainous adventures.

1.4 Outline

This thesis is seven chapters, as briefly described in the following:

Chapter 2, Background and related work, describes the background of our project. Furthermore, we present an overview of related work in geovisualization and image registration.

Chapter 3, The PanoVis system, analyses the problem and results in a set of requirements for solving it. Furthermore, we present the methods employed to meet the requirements.

Chapter 4, Implementation, describes the implementation and the technologies employed in the application, supplemented with the reasoning behind our choices.

Chapter 5, Evaluation, presents how we evaluated the application, followed by an analysis of the study results.

Chapter 6, Discussion, discusses the evaluation results, mentioning aspects of the application's design that could have been solved differently and where we see the application's potential for future development.

Chapter 7, Conclusion, concludes our work and highlights the features that make PanoVis stand out.

Chapter 2

Background and related work

This chapter presents the research context in which our application was built, presenting the relevant pieces of previous work and an overview of the state of the art in geovisualization, image registration, and image analysis.

2.1 What is geovisualization?

The term geovisualization emerged around 1990, when *cartography* transitioned into the computer age [15]. It results from a paradigm shift from *communication* in traditional cartography with a focus on explanation, to *exploration* and knowledge crystallization in geovisualization [16, 63]. Today, users can make on-demand changes to the display and access a variety of coordinated views in real-time, enabling data exploration from multiple perspectives. Hence, geovisualization situates at the core of visual information processing to facilitate thinking in complex decision-making tasks [2].

Several definitions of geovisualization exists. The term is used to refer to maps, a way of using maps, an academic discipline, or visualization techniques, to name a few [16]. MacEachren et al. [62] define it as a process for leveraging geospatially referenced data to meet scientific and societal needs and a research field developing visual methods and tools to support a wide array of geospatial data applications. Despite the term's ubiquity, most of the context in which the term appears has a relationship to interaction with geospatially referenced data. The most widely accepted definition is the one introduced by the International Cartographic Association (ICA) Commission on Visualization and Virtual Environments, a commission that has played an important role in stimulating geovisualization research [62]: "Geovisualization integrates approaches from visualization in scientific computing (ViSC), cartography, image analysis, information visualization, exploratory data analysis (EDA), and geographic information systems (GISystems) to provide the-

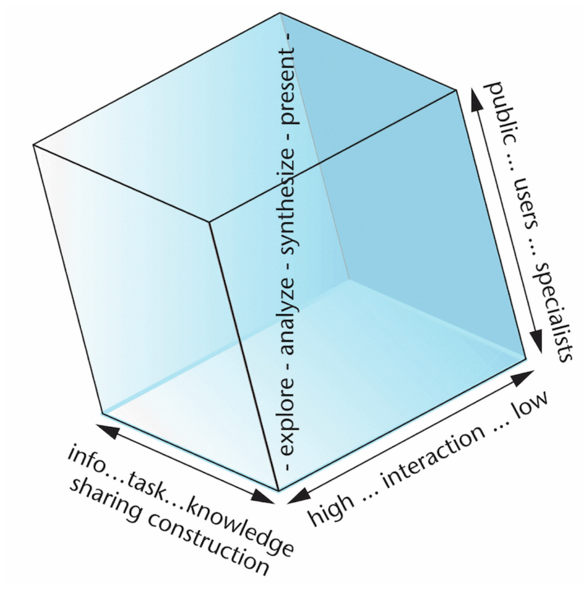

- explore - analyze - synthesize - present -
public ... users ... specialists
high ... interaction ... low
info...task...knowledge
sharing construction

Figure 2.1: *Functions of geovisualization* by MacEachren et al. [62, Figure 1]

ory, methods, and tools for visual exploration, analysis, synthesis, and presentation of geospatial data [65, p.3]."

MacEachren et al. [62] depict the four functions of geovisualization in their *geovisualization-use space* illustration, presented in Figure 2.1. The figure models the space of visualization goals concerning three dimensions; *task*, *interaction*, and *users*. The task ranges from sharing existing knowledge and information to revealing unknowns and constructing new knowledge. Interaction ranges from passive visual interfaces with low interactivity to active visual interfaces with high interactivity. Thirdly, the user space ranges from a single, private user to a larger public audience [77].

2.1.1 Driving forces of geovisualization

Nöllenburg [77] reports that there are three driving forces in geovisualization. The first is the rapid advances in graphics and display technology during the last few decades. High-performance graphics hardware has plummeted in costs, making efficient personal computers highly accessible, triggering the development of highly immersive three-dimensional virtual environments. With high-performance hardware being public property, researchers have investigated these technologies' potential for visualizing geospatial data.

An ever-increasing amount of geospatial data is the second driving force of the domain. Such data are collected continuously by numerous scientific and governmental institutions, private companies, and individuals and need to be analyzed and explored. MacEachren and Kraak [65] estimated in 2001 that up to 80 percent of the generated data includes geospatial references, such as geographic coordinates, addresses, and postal codes. Rhyne and MacEachren [93] stated that the Earth Observatory System generated more than a terabyte a day of geospatially referenced data. A common property of geospatial data is that they are multivariate, making them a rich source of potentially valuable information for research and decision making. During the recent COVID-19 pandemic, for example, geospatially referenced data has been valuable for predicting the spread of the disease [58]. Confronted with lots of data, users quickly meet the limits of their capacity in analyzing and understanding the data [77].

However, computers perform poorly in detecting and interpreting unknown patterns in noisy data, which the human brain excels in, in comparison. The goal of geovisualization is to combine the strengths of human vision and knowledge with the storage capacity and computational power of modern computing to explore large geospatial data sets. One way of doing this is to simultaneously represent many graphical representations of the data, allowing the user to look at the data from different perspectives to gain insight and draw conclusions.

The third driving force of geovisualization, as described by Nöllenburg [77], is the development of the Internet and its prominent position as a medium for disseminating geospatial data and maps [53]. The Internet facilitates the collaboration of expert users at different places, one of the ICA Commission's research challenges.

2.2 Visualizing geospatial data

We live in a time where technology is advancing at an unprecedented pace. Modern *Geographic Information System* (GIS), GPS, and remote sensing technologies enable us to capture large amounts of geospatial data almost effortlessly, and researchers strive to create methods presenting these data digitally [66]. There are numerous techniques for visualizing this kind of data, ranging from simple maps using data of low dimensionality to more complex ones combining a wide array of variables for each data point.

2.2.1 Cartographic visualizations

In the sense of "making something visible," all maps can be considered a visualization [64]. In academic cartography, maps have been a key tenet of scientific studies for a long time. Kraak et al. [54] define maps as "a symbolized representation of geographic reality, representing selected features or characteristics, resulting from the creative effort of its author's execution of choices, and is designed for use when spatial relationships are of primary relevance".

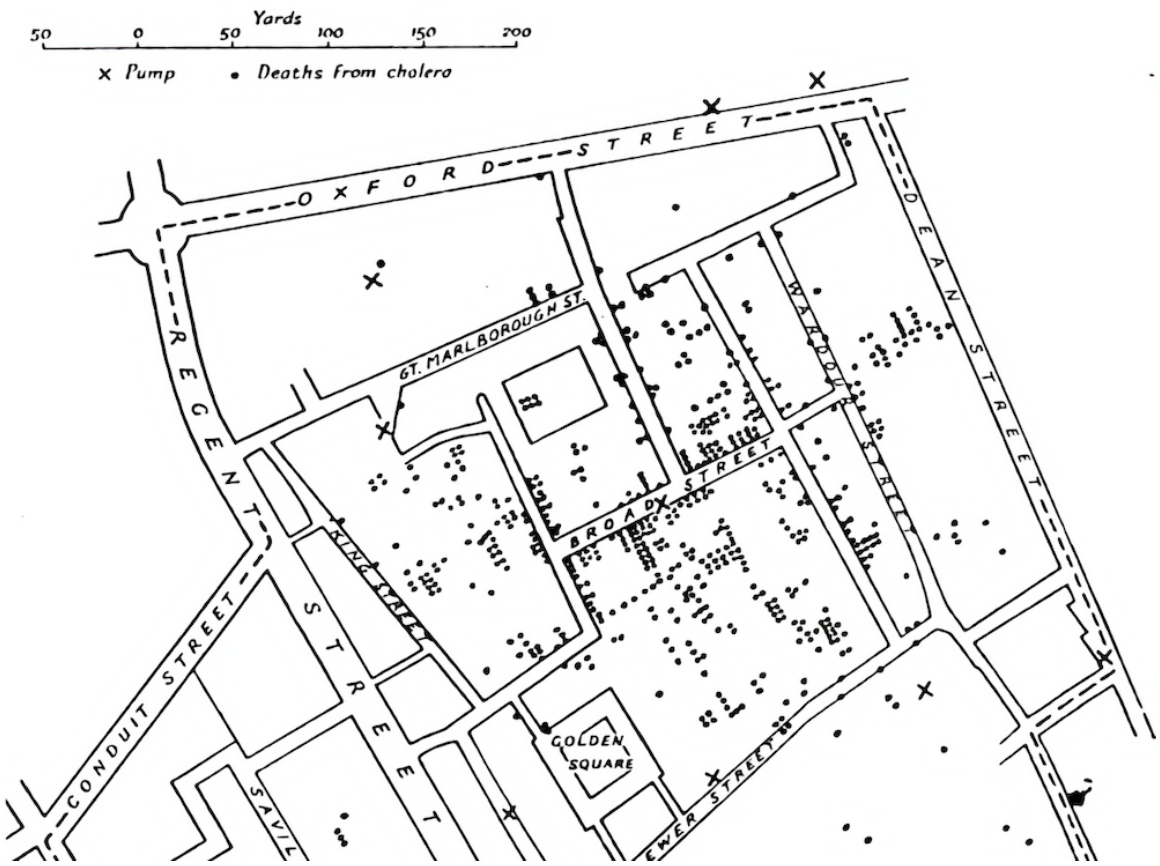


Figure 2.2: Dr. Snow's [102] map of cholera deaths in London from 1854. Deaths are marked by dots and water pumps by crosses. Recreation by Gilbert [31].

An early example of geospatial data visualization is Dr. Snow's [102] cholera dot-map from 1854. Gilbert's [31] recreation of the map is shown in Figure 2.2 and presents a section of a map of London. The water pumps and the cholera victims' location of death are on the map. By plotting this data on the map, Dr. Snow gained critically important insight into the origin of the cholera outbreak. Identifying one of the water pumps as the source of infection resulted in the officials closing the water pump and eradicating the cholera outbreak.

2.2.2 Visualizing three-dimensional geospatial data

Three-dimensional geospatial data incorporates the third dimension, storing more information than the traditional two-dimensional geospatial data consisting of a pair of x and y values. In geovisualization, the third dimension usually represents real-world elevation values, such as a surface's height above sea level or geological depth in oceanic regions [29].

Some data sets include elevation values for each geospatial coordinate. Such data sets are called *Digital Elevation Models* (DEM), where each elevation value represents the approximate vertical distance between a geographic reference point [20].

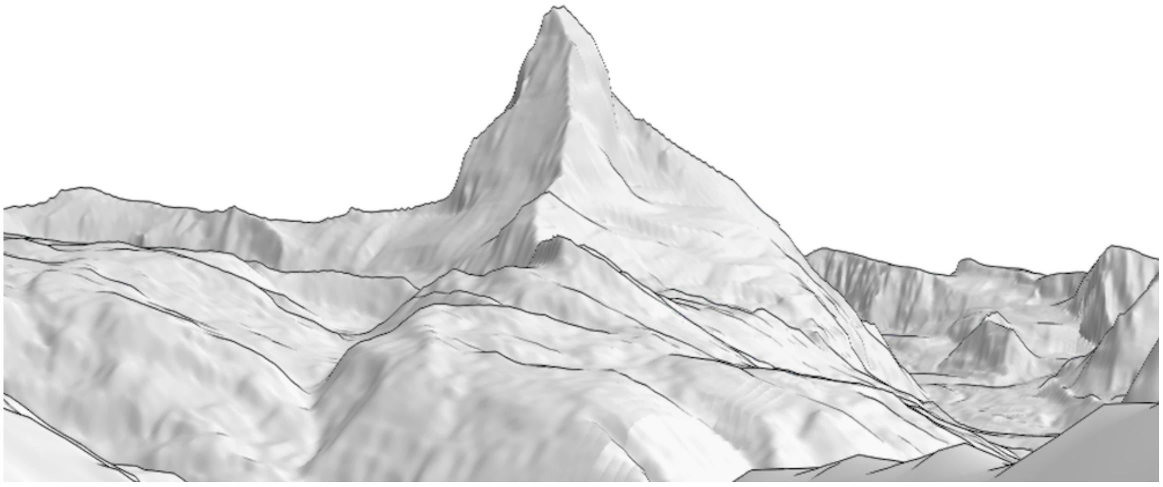


Figure 2.3: A rendering of the three-dimensional nature of the Matterhorn's surface, created using Deuschle's web tool for creating digital terrain panoramas [18].

Digital Surface Models (DSM) and *Digital Terrain Models* (DTM) commonly adopt the term DEM as a generic terminology, even though there are some differences that are worth mentioning. When fine details about the terrain are needed, DSMs are used as they include natural and built features on the Earth's surface, such as trees, stones, buildings, or bridges. DTMs are used for more coarse-grained terrain, such as the oceanic surface, and are more suitable for mapping the Earth's surface.

Wróżyński et al. [123] combine DSMs with rendered images of wind turbines to quantify the visual impact of wind turbines. By factoring in the wind turbines' size and position, the authors generate a viewshed of the wind turbines' potential area of visibility. DSMs are also commonly used in runway approach zones to determine possible obstructions in the flight path [86]. Our work uses the term DEM to refer to the digital terrain models without buildings and vegetation.

These three-dimensional data have a wide range of applications and are commonly used with ray-tracers to create synthetic visual images of the terrain, illustrated in Figure 2.3. The figure shows a landscape representing the surface of the Matterhorn, created by interpolating elevation values to create a smooth and seamless surface. The edges are colored in contrasting colors to emphasize the differences in the terrain, and some researchers utilize such edges to identify the visible peaks in a scene [5, 24, 25, 74, 91].

2.3 Mashup technique

The panorama explorer found in PanoVis builds upon several different visualization tools. It draws similarities to a highly popular geovisualization technique that combines content from more than one source to create an integrated end-user ex-

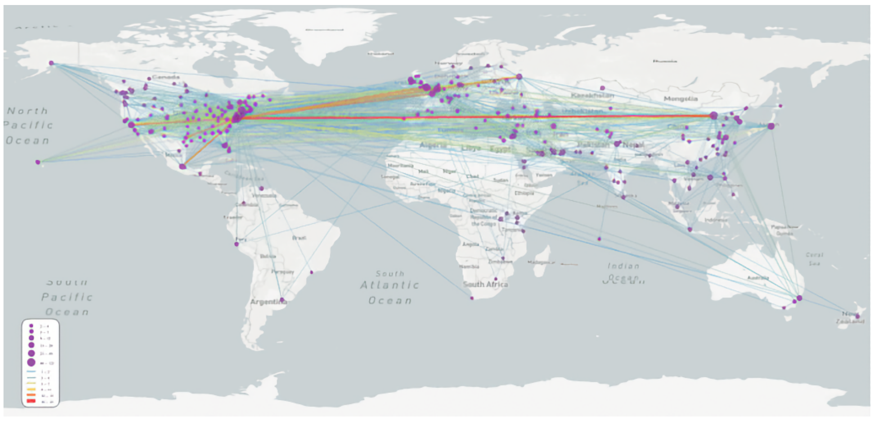


Figure 2.4: Geo-mashup example by Zhang et al. [127, Figure 7(a)].

perience displayed in a single graphical interface, typically called a mashup [27, 59, 126]. Mashups commonly use some form of geographical representation to integrate applications and data sources and provide a visual interface [59, 120]. Mashups can rely on static and dynamic web content, with the static content often being background cartographic material as Figure 2.4 illustrates [33]. During the last decade, more and more public *Application Programming Interfaces* (API) have enabled mashups to integrate more dynamic content, such as news feeds, traffic information, or meteorological reports [87].

With the increased popularity of mashups, several tools exist to simplify the development of mashup-like applications. Gahegan et al. [30, 110] introduced *GeoVISTA Studio*, a codeless visual programming environment for analyzing and visualizing geoscientific data. Their goal was to give non-programmers the ability to create sophisticated mashup applications easily and rapidly manipulated. GeoVISTA Studio gives its users tools for designing and implementing interactive interfaces using visual programming, using drag-and-drop techniques to form the program's outlook without writing any code. Fekete [26] presented the *InfoVis Toolkit*, a coherent software architecture and a set of components designed to support the creation of information visualization applications. It has some similarities with GeoVISTA Studio and includes mechanisms and components for direct manipulation of visualizations. The InfoVis Toolkit also features functionality for selecting and filtering the data and performing a broad array of well-known generic information visualization tasks on the data.

Other researchers have created mashups with a more specific area of use. Wood et al. [120] exploit an array of existing functionality and data to create new applications tailored for their specific task of exploring millions of geographically referenced search queries. The resulting application includes several methods for interacting with and analyzing the data, including techniques that the authors name *tag clouds*, *tag maps*, *data dials* and *multi-scale density surfaces*. In addition to the proposed geovisualization techniques, they also provide methods for evaluating their mashup approach, which will be discussed later in this chapter.

In the mashup-like panorama explorer found in PanoVis, one of the visual components is annotated with marks representing additional geographical information about the image in view. To calculate the positions of these marks, knowing the *camera pose* comes in handy.

2.4 Camera pose estimation

Estimating the camera pose is a crucial point in our work, and minor errors in this estimate can lead to inaccurate annotation placements. There are several ways to estimate the pose of an image, including image-based methods and methods utilizing data of multiple modalities. Different methods have been studied to solve this problem, and we list several examples and their common elements in the following. While some of these examples focus on estimating the camera's location, our work will only estimate the viewing orientation based on the assumption that the uploaded panoramas contain a geographical reference.

2.4.1 Image-based methods

Kopf et al. [50] proposed Deep Photo, a system that includes functionality for annotating images with information about buildings and roads, among others. Their algorithm relies on estimating the camera pose using the image along with a three-dimensional model covering the visible scene. Using a technique called *image registration*, which we will cover in the next section, the authors are able to map the image to the model. This mapping allows them to read geographical information from the underlying three-dimensional model, which they then use to annotate the image. Their technique relies on manual user interaction to control the mapping process, which is similar to our approach for PanoVis.

Hays and Efros [36] propose IM2GPS, a method for solving a different part of the photograph pose estimation task, namely to estimate its geographical location. Their algorithm relies on a data-driven scene matching approach, leveraging a data set containing millions of geotagged crowdsourced images from online photo services such as Flickr. The algorithm is applied to non-geotagged images to estimate their geographic location using global image descriptors only. Due to the Flickr database mostly consisting of images of landmarks and tourist attractions, their algorithm is negligibly accurate when querying photos from lesser photographed spots, such as those taken in rural areas. Lin et al. [60] unite the IM2GPS algorithm using aerial imagery for a cross-view matching algorithm. The authors report that their differential translation approach trained on ground-level scenes successfully geolocates 17% of query images from isolated areas, compared to 0% for existing methods.

2.4.2 Geo-localization using multiple modalities

Unlike image-based geo-localization methods, the methods leveraging data of multiple modalities rely on additional input data to estimate the camera location for a query image. Most of these methods use a cross-domain matching of a query image and a DEM, where features such as horizon lines and edge maps are exploited for pinpointing the camera location. Stein and Medioni [107] use horizon lines from a query image together with the DEM to create a matching method verifiable by geometry.

Nagy [74] proposes a method for improving the accuracy of the azimuth angle provided by mobile devices in augmented reality applications. Using the on-device camera and a DEM, matching the panoramic skyline with the skyline vector from the DEM determines the correct azimuth angle.

2.5 Image registration

As briefly described in Section 1.3 and mentioned above, the PanoVis system includes a module for aligning a panorama with a computer-generated terrain image. The backbone for our method is *image registration*, which Brown [10] defines as the process of overlaying two or more images of the same scene concerning a particular reference image [73]. The images can be taken at different times or from different viewpoints, using a different camera sensor or focal length [128]. Image registration aligns the two images, called the *reference* and *sensed* images. Zitová and Flusser [128] claim that the majority of image registration techniques consist of the following steps, which the same authors illustrate in Figure 2.5:

1. *Feature detection.* Distinctive features in the sensed and reference images, such as edges, corners, and line intersections, are manually or automatically detected.

2. *Feature matching.* The correspondence between the features detected in the images is established.

3. *Transform model assessment.* The type and parameters of the mapping function are estimated and used to align the sensed image with the reference image, utilizing the established correspondence between the features.

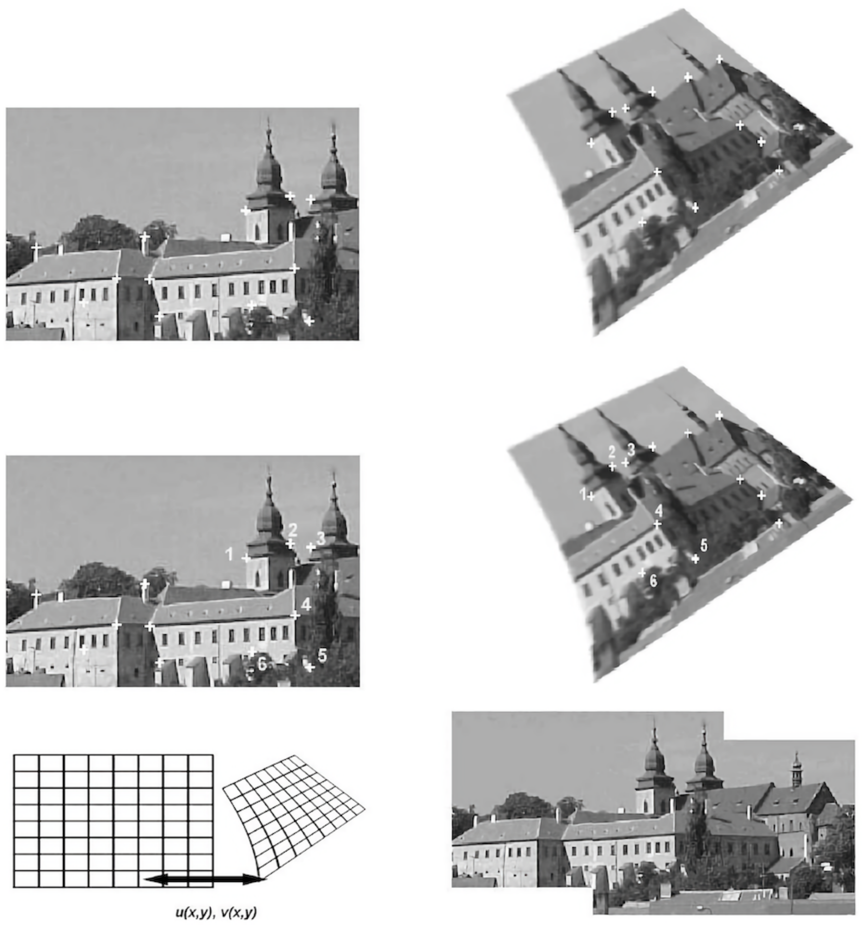


Figure 2.5: The four steps of image registration as illustrated by Zitová and Flusser [128]. The top row shows feature detection using corners as features, the middle row illustrates feature matching by invariant descriptors, and the bottom row presents the transform model estimation and the final image. The mapping exploits the established correspondence between the images' features.

4. *Image resampling and transformation*. The mapping function is applied to the sensed image, aligning the sensed image with the reference image. Appropriate interpolation methods compute the image values for the pixels outside the reference image's bounds.

With the camera pose being established through image registration, the next step is to identify the pieces of geographical data that should be annotated on the images. In our case, this means calculating the visibility of a set of mountains from a given viewpoint.

2.6 Identifying mountain peaks

Most camera pose estimation research uses high accuracy three-dimensional urban models, and several of the studies aim to recognize constructions in larger

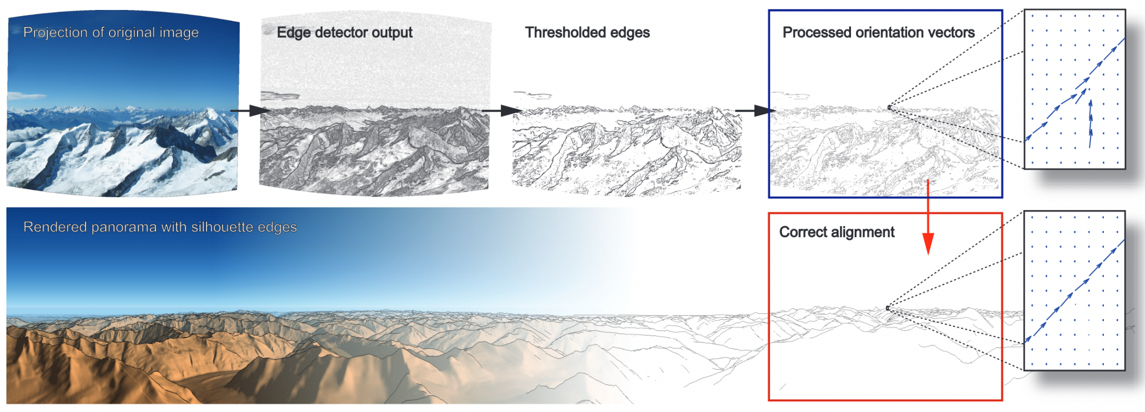

Projection of original image
Edge detector output
Thresholded edges
Processed orientation vectors
Rendered panorama with silhouette edges
Correct alignment

Figure 2.6: Illustration by Baboud et al. [5, Figure 8] showing how their algorithm aligns panoramas with terrain models using silhouette edges.

cities. When it comes to identifying mountain peaks, many studies accomplish this leveraging DEMs. Presenting the data from a DEM as a three-dimensional model is quick and easy, and newer technologies make this a feasible task for most consumer-friendly computers. In such presentations, we can quickly segment the nearby mountains from the mountains on the far horizon, and using this depth data can be convenient in many use-cases. We discuss how this information can be implemented in pose estimation to recognize the mountain peaks in a photograph.

2.6.1 Leveraging digital elevation models

Baboud et al. [5] propose a method for the automatic annotation and augmentation of mountain photographs, relying on a method using the characteristics of the visible horizon. Their algorithm uses the known location of the photographer and field of view to create silhouette edges retrieved from a DEM and matches these with the photograph's edges to find the viewing direction. They illustrate this process in Figure 2.6. A search space reduction on the extracted edges decreases the computation time. The authors report a highly accurate terrain alignment, indicating the use of a high-resolution DEM.

Fedorov et al. [25] present a method for identifying mountain peaks in geographically referenced photographs using a coarse DEM sampled at a spatial resolution between 30 m and 90 m. A 360 degrees panorama created from the photograph's location data and the surroundings visible in the panorama allows for matching the skyline from the DEM terrain with the photograph. Using a *Vector Cross-Correlation* (VCC) technique to find the best matching candidates, the VCC for each peak creates labels for the visible mountain peaks. The presented method is completely unsupervised, allowing it to analyze large image sets without the need for manual annotation.

Fedorov et al. [24] combine photograph-to-model matching with the on-device smartphone sensors to present real-time mountain peak metadata in an augmented reality (AR) application. The method results in a content-based reality augmentation algorithm more robust than the ones only relying on the smartphone sensors because the digital magnetic compass is prone to errors if larger magnetic objects are close to the smartphone [5]. The GPS sensor can also be error-prone in mountainous regions if the photographer is in a valley with occlusions between the smartphone and the satellites [5].

Hofmann [38] created GIPFEL, an application for annotating mountainous images relying on intervention from the user. Given an image, it requires the user to input a location and select a minimum of two visible mountains from the database. The problem with the mountain selection method is that the user needs to know the names of at least two of the mountains in view, and these two need to exist in the database. Therefore, selecting mountains by name can be cumbersome for images taken in unfamiliar areas. After the selection process is complete, a drag-and-drop interaction moves the mountain objects to their corresponding image spots. Given the placement of these mountains, the program guesses the camera's focal length, view direction, tilt, and roll. Mountain metadata from the database uses these parameters to create an annotated overlay for the inputted image. If the program guesses the parameters wrongly, manual adjustments are possible for fine-tuning the overlaid mountain marks.

2.7 Summary

This chapter has presented relevant pieces of previous work, including the state-of-the-art in geovisualization, the main research area for the realization of PanoVis. We have also provided some insights into how other researchers use visualization techniques with geospatial data, and how they combine multiple uses of visualization to create mashups. In the next chapter, we present how PanoVis builds upon some of the methods presented above, as well as providing reasoning for the choices made in the design of the system.

Chapter 3

The PanoVis system

This chapter presents a set of requirements for PanoVis and introduces the novel techniques that were developed to meet the requirements. Additionally, we provide details about the visual components that builds on the proposed techniques and how they make up the PanoVis system.

3.1 System requirements

Chapter 1 introduced the need for combining geographically referenced images with terrain data in order to provide hikers the information they need to be inspired for their next adventures. In order to find a solution that achieves that, it was important to understand the hikers' needs and requirements. We used methods from requirements engineering to obtain better insights into the hikers' needs, combining interviews, task analysis, and domain analysis [67]. The interviews were semi-structured, consisting of predefined and unplanned questions [92]. Three hikers that reported they were hiking multiple times a month were interviewed, and each interview included the questions found in Appendix A.1, followed by unstructured discussions concerning the points they highlighted at the beginning of the interview. The interviews allowed us to understand the hikers' expectations for tools that helped finding a solution.

For the application to be valued as useful, implementing as many as possible of these requirements was obviously a goal. However, we also concluded that the application would achieve our overall goal despite lacking some of the requested features. The following list defines and describes the requirements that resulted from the interviews, as well as requirements introduced to strengthen the geovisualization aspect of the system. They are ranked from most important to least, weighted by the knowledge obtained through the interviews.

R_1 **Annotated Panoramic Images** As panoramic images of mountainous landscapes function as a great resource of hiking inspiration, giving viewers a larger grasp of an area's views and terrain, we deemed the display of such imagery as the most important feature of our prototype. Presenting panoramas in a non-flat manner was highly emphasized to maintain the cylindrical aspects of its contents. To provide additional details about the panoramas' motifs, enriching the images with metadata about the visible peaks from the point of image capture was also accentuated.

R_2 **Three-Dimensional Terrain** Exploring mountainous areas requires a good understanding of the terrain. Visualizing geospatial data in three dimensions is a proven and reliable method for integrating heterogeneous data sets into a single view [37]. It provides users quick and effective insights into the geography of an area, and thus, a system that facilitates the discovery of new hiking areas must include functionalities that support a three-dimensional view of the geospatial data.

R_3 **Topographic Maps** Hikers use topographic maps to plan routes, estimate travel times, and locate water sources and suitable campsites [119]. Hence, our application should provide methods for exploring the local geography of an area using two-dimensional map views. Embedding the maps with the same geospatial information projected onto the panoramas is desirable.

R_4 **Hike Trail Visualizations** In addition to using panoramas as a source of hiking inspiration, other hikers' routes provide deeper insights into where the photographer was going in advance of capturing the panorama. Combining the hike's GPS path with the corresponding images photographed during the hike would make it easier for users to understand how to access the mountain and increase the geographical understanding of the area.

R_5 **Coordinated Views** Accommodating multiple of the above requirements introduced a need for synchronizing the distinct views when navigating between multiple images. As the different views reflect the data centered around a certain viewpoint, changing the image-in-view should update all views to reflect the data belonging to the newly selected viewpoint [116].

R_6 **Immediate Visual Feedback** It could be disadvantageous if there was a lack of visual feedback while the application processes data behind the scenes [1]. Loading animations should be present during time-consuming processes, creating an illusion of a more responsive application.

R_7 **Emphasize Interactivity** Interaction could increase the effectiveness of visual representations [12]. It was therefore desirable to emphasize user interaction

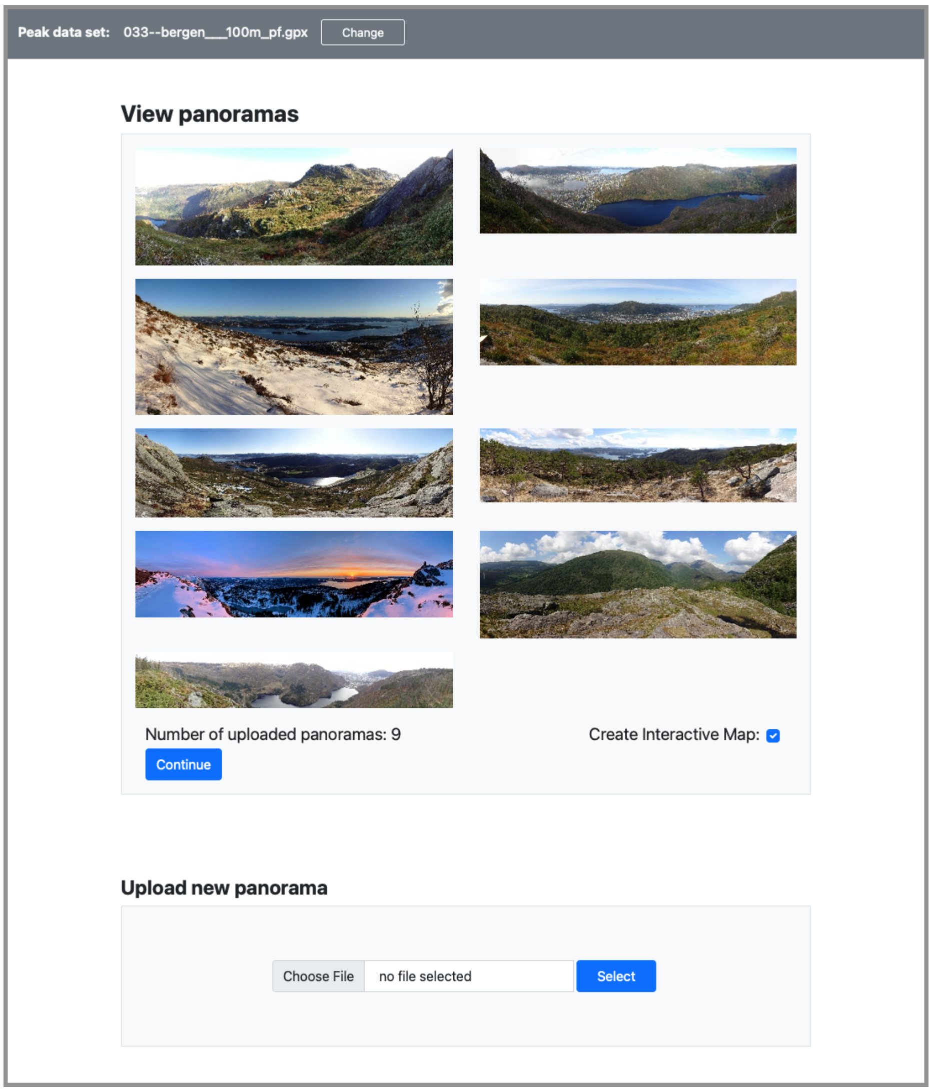

Peak data set: 033--bergen___100m_pf.gpx
Change
View panoramas
Number of uploaded panoramas: 9
Continue
Create Interactive Map: ✓
Upload new panorama
Choose File
no file selected
Select

Figure 3.1: The File Handler window used to add or remove images to PanoVis.

throughout the application, leading to a more intuitive user experience for exploring the data sets.

3.2 Application overview

PanoVis consists of three separate windows: *the File Handler*, *the Panorama Aligner*, and *the Panorama Explorer*. In the following, we focus on the algorithms and processes used to align the images and how we generate a variety of geovisualizations presented along with the panoramic images, which are described in detail further below. These visualizations are created with the aforementioned requirements in mind, focusing on user and domain requirements. Upon application launch, users are greeted with a *Graphical User Interface* (GUI) for handling existing images, adding new ones, removing old ones, and controlling which mountain peak data set to use for the application. Figure 3.1 illustrates how the File Handler looks with nine panoramas added to the service. This view also provides additional controls on which kinds of visualizations to display.

Adding new panoramas redirects the user to the panorama aligner, a tool needed to estimate a panorama's viewing direction. The alignment process combines user interaction with automatic methods, which are described in Section 3.3, to align panoramas with static images of a digital terrain covering the same views as the panoramas. Because of the default characteristics of these digital terrain images, shifting the panorama to its corresponding position in the digital terrain allows us to extract the information needed for estimating a panorama's viewing direction.

Once panoramas have been added to PanoVis, the user can navigate to the panorama explorer to examine the photographs and their counterpart geovisualizations. Upon loading this view, the mountain peak data set provided by the user is processed to identify which mountain peaks are visible from the panorama's capturing location. The resulting data is then projected onto the panorama, including the peak's name, elevation, and distance from the viewpoint. Every visual component of the panorama explorer supports user interaction for displaying additional information about the data sets, enabling the user to explore the underlying data on demand, avoiding information overload by default [14].

3.3 Azimuth estimation

Panoramic images play a key role in the PanoVis system and act as a gateway to explore multiple instances of geospatial data. We have developed a technique for superimposing geospatial information onto the panoramas to fulfill the requirement of R_1 (Annotated Panoramic Images). Thus, understanding the position of these objects relative to the photographer's viewpoint was important for positioning the annotations on the panoramic images. In this thesis, our method for computing geospatial objects' relative positions from a viewpoint depends on a panorama's viewing direction, called the *azimuth* [4]. To achieve this, we estimated panorama azimuths by combining automatic and manual processes.

As presented in Section 2.4, the viewing direction is one of the *Degrees Of Freedom* (DOF) in camera pose estimation. In our application, exploring the data through interactive methods is heavily emphasized, referring to requirement R_7; thus, we want to include the photographer in estimating the panoramas' azimuths. Our technique for accomplishing this begins with rendering a static image of the terrain surrounding the photographer, which we refer to as the *rendering*, similar to the technique presented by Fedorov et al. [25]. Renderings are 360 degrees

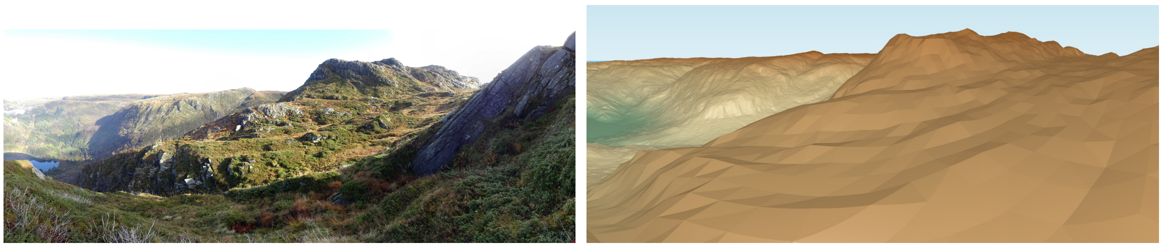

(a) Panorama (b) Rendering

Figure 3.2: A panorama and its corresponding render.

static images generated using elevation values of an underlying DEM with a common viewing direction. Exploiting these characteristics facilitates the conversion of pixel coordinates along the x-axis to azimuth approximations. Figure 3.2 presents a panorama and its corresponding rendering. Mapping the panorama to the rendering through image registration techniques allows us to transfer the same characteristics to the panoramic image.

Recalling the image registration steps introduced in the previous chapter and illustrated by Zitová and Flusser [128] as well as in Figure 2.5, the image registration process begins with feature detection and feature matching. Popular feature-based matching algorithms, such as SIFT [61], tend to be less usable for image-to-rendering registrations [128], due to the image motives being highly dissimilar in their looks. Therefore, our method places the photographer in charge of these tasks by presenting panoramas and renderings in an image viewer with functionality for recording the user's actions. To successfully locate the same features in the panorama and a computer-generated rendering requires the rendering to be easily recognizable and mimic the landscape in the panoramic image as accurately as possible.

3.3.1 Terrain rendering

Creating renderings with easily recognizable features called for carefully considered choices of parameters that can determine the terrain's appearance. Our method utilizes the digital terrain images for manually selecting a set of control points in the distant landscape, and it is thus essential that the render's viewpoint is placed as close as possible to the photograph's location. We used the geographical location reference from the panoramas as the render's viewpoint because most modern cameras have built-in GPS receivers [91]. Even though GPS receivers in modern cameras are prone to errors in mountainous areas, most readings locate the camera with a deviation of fewer than ten meters [56]. Such deviations can have a significant impact on the appearance of nearby geography.

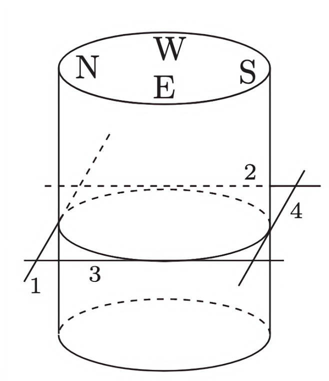
Mostly, however, our focus is on recognizing objects in the distance. Thus, these inaccuracies are small enough and do not impact the appearance of the distant terrain substantially. We accept that discrepancies may occur, since they have any significant impact on PanoVis' functionality.

A panorama's location consists of a pair of longitude and latitude values. In addition to these planar two-dimensional coordinates, the renderer also needs an elevation to position the viewpoint in the three-dimensional terrain accurately. The viewpoint's vertical position is sensitive to small elevation changes and must be carefully selected. A too low viewpoint placement ensues in visual artifacts where sightlines pass through the ground, rendering parts of the terrain transparent. Viewpoints positioned too high introduce a risk of losing some key features in the nearby terrain, making the renderings less recognizable. To avoid these scenarios, we exploit the elevation values of the DEM to position the viewpoint just above the terrain.

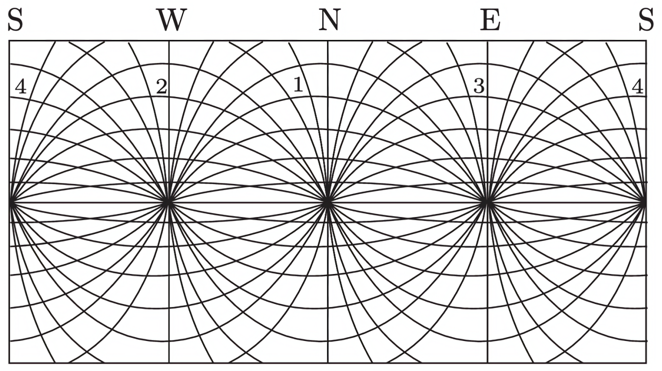
In cases where the viewpoint is placed on the edge of a raster cell, using the cell's elevation as the vertical viewpoint value might result in renderings with a too low viewpoint elevation. Imagine a $3x3$ grid representing a part of the DEM, with the viewpoint located at the intersection of three cells at the matrix' corner. If the cells represent a hilly terrain, there can be large elevation changes between two neighboring cells. As the cells of a DEM are discrete values representing the continuous terrain, using the elevation of a cell as vertical viewpoint placement may result in a viewpoint submerged under the terrain surface if with large changes in elevation between two cells. To avoid positioning the viewpoint below the ground, we introduce a loop that generates smaller 200 by 100 pixel renderings and verify that the viewpoint is not submerged in the terrain. The smaller rendering sized allows the loop to iterate at quick paces, reducing the processing time at this stage. If the viewpoint is submerged, we increase the viewpoint's elevation by 50 centimeters and repeat the process until the viewpoint is above the ground. When a satisfying viewpoint elevation is found, we generate a full sized rendering using these updated parameters.

Rendering parameterization

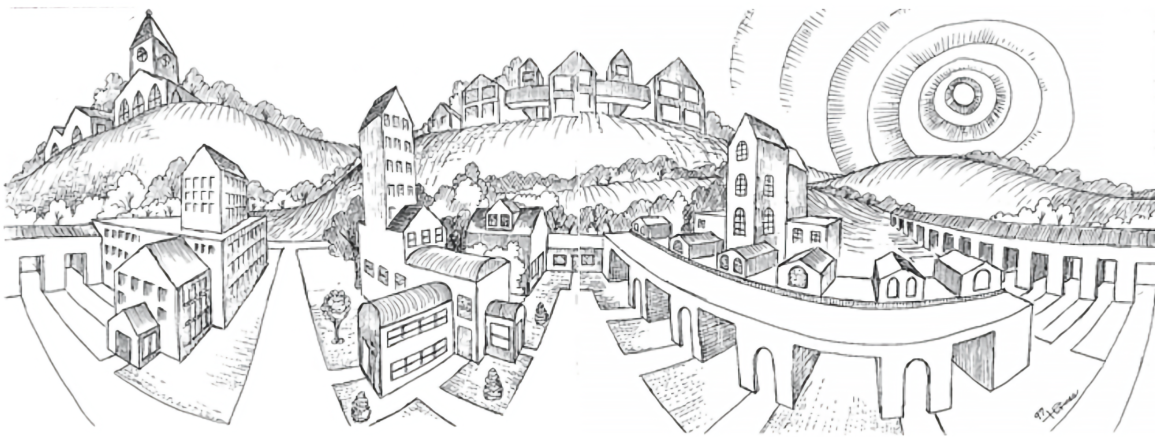
Several parameters influence a render's look after viewpoint positioning in a digital terrain. Selecting an appropriate projection is a task that we examined carefully, as the two images should share similar projections to recognize and match features easily. The most common panorama type in digital photography is the one of *segmented panoramas* [44], created by capturing overlapping images, based on rotating the camera around its vertical axis while taking the images. These images are joined using image registration techniques to create *cylindrical panoramas* [57], illustrated



(a) Four segments



(b) Cylindrical projections of horizontal segments



(c) Cylindrical panoramic projection drawn by Termes [97, p. 172, Figure 4.24]

Figure 3.3: Illustrations showcasing cylindrical panoramic projection presented by Salomon [97, p. 171-172]

in Figure 3.3(c). We take advantage of the fact that panoramas are most often of cylindrical projection and use this projection also for our renderings.

Figure 3.3(a) and (b) illustrate how straight segments become curved when unrolling a cylindrical panorama. There are four different segments, each representing one of the sky directions. The first segment tangents part of the northern part of the cylinder, and following the segment between East and West demonstrates the segment's curvature. The same applies to the remaining segments but with different endpoints. The curving increases towards the top and bottom of the cylinder. In Figure 3.3(b), the segments drawn at the cylinder's extreme ends fall outside the panorama. It is possible to avoid losing potentially important information positioned at the cylinder's ends by increasing the render's vertical FOV. By doing so, our program handles all types of panoramas and guarantees that the rendering contains the entire scene shown in the panoramas.

Another parameter that highly influences rendered terrain images is their amount of *vertical exaggeration*, a technique used to discern subtle topographic fea-

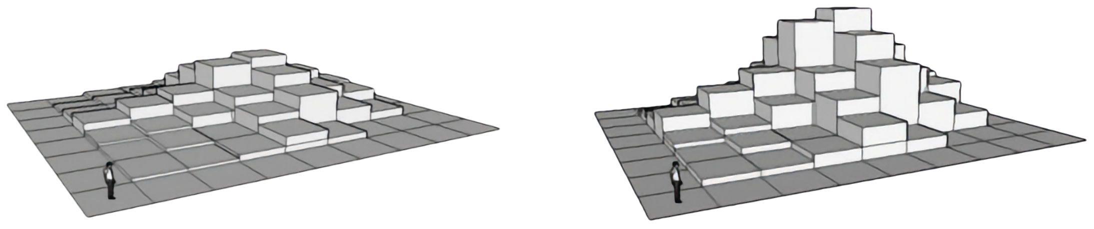

(a) No vertical exaggeration (b) Vertically exaggerated

Figure 3.4: Two illustrations by Nutsford et al. [79] of a digital elevation model representing a hilly surface and a vertically exaggerated copy.

tures in digital terrains. Vertical exaggeration increases a surface's vertical scaling while maintaining the same horizontal scaling and assists in creating images that look similar to how our eyes perceive landscapes in the real world [82]. Therefore, three-dimensional cartography and virtual reality environments have actively used vertical exaggeration [19]. However, some researchers oppose the technique describing it as a tool that makes oblique visualizations dramatic at the cost of misleading the viewer about the landscapes [71].

Figure 3.4 presents subfigures illustrating how vertical exaggeration affects the appearance of a scene. Both illustrations represent the same DEM with a steep slope stretching towards the upper-right corner of the DEM, and with an observer positioned at the lower-left corner. In Figure 3.4(a), it can be hard to spot the slight slope beginning right next to the observer. Applying vertical exaggeration makes these slight increments more prominent, as illustrated in Figure 3.4(b).

Although some researchers oppose vertical exaggeration, we concluded that using this technique for the renderings is beneficial. Vertical exaggeration will help PanoVis's users select the image registration's control points. Therefore, we vertically exaggerated the scenes by a factor of two to take advantage of the benefits this entails without making the terrain look otherworldly.

Based on our choices of parameters for the renderings, we ended up with a result that includes a horizontal FOV of 360 degrees, a vertical one of 180 degrees, and a viewpoint positioned just above the terrain at the panorama's location of creation. To increase the visual differences between low-lying areas and the mountains, we textured the terrain with a gradient from green to brown depending on the height data in the different raster cells. These choices provide us with images that are well suited for detecting and matching features.

3.3.2 Registering image control points

As previously mentioned, our image registration method relies on user interaction. The reference and sensed images, respectively the rendering and the panorama,

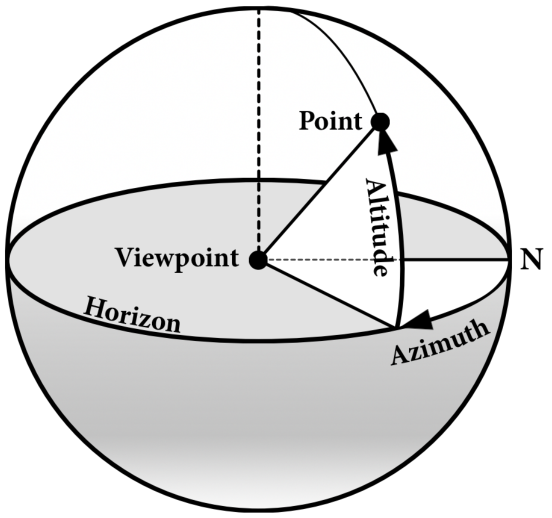


Figure 3.5: A schematic diagram illustrating *azimuths* and *altitudes*. The illustration is based on user TWCarlson's work uploaded to Wikimedia Commons [112].

are displayed in the application's frontend for manual feature detection and correspondence establishment between these. Due to the render's wide vertical FOV, we selected a spherical image viewer for displaying images that utilize the entire FOV around the user's point of view, meeting requirement R_1. The spherical image viewer lets PanoVis record user actions, such as keyboard and mouse clicks.

A set of control points in each image is required to map the panorama to the render. To create a set of control points, the user interacts with the spherical image viewer, where each mouse click returns a pair of values. These value pairs consist of *azimuth* and *altitude*, describing the degree of rotation around the vertical axis from the viewpoint and its vertical position relative to the horizon, as Figure 3.5 illustrates. Before deriving the actual mapping, we need to convert these control points to planar coordinates. To convert these values to planar pixel coordinates, we use the formula of Petroff et al. [85] given in Equation 3.1.

$$\begin{aligned} pixel_x &= (azimuth/FOV_{horizontal} + 0.5) * panorama_{width} \\ pixel_y &= (0.5 - altitude/FOV_{vertical}) * panorama_{height} \end{aligned} \tag{3.1}$$

Selecting an image point returns a pair of x and y values, which we use to perform the third step of Zitová and Flusser's image registration pipeline [128]: the transform model assessment. Our method depends on a function for creating transformation matrices provided by the *Open Source Computer Vision* (*OpenCV*) [42] library. Chapter 4 presents further details about the implementation of this library and how its used to accomodate our needs. The function for creating the transform

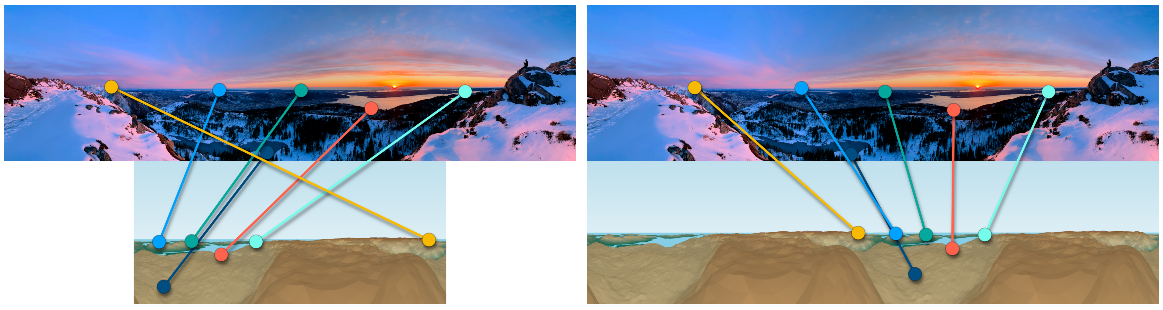
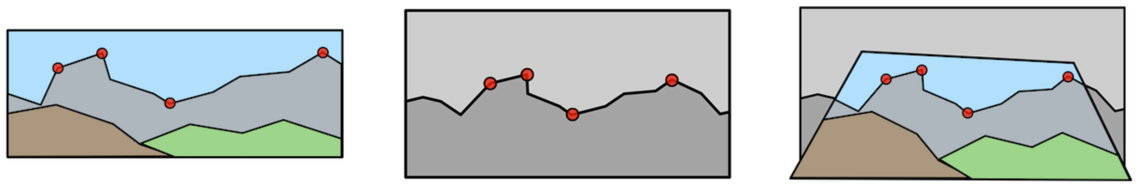

(a) Rendering with 360 degrees FOV (b) Rendering with 720 degrees FOV

Figure 3.6: Corresponding control points visualized using one panorama with two renderings. The rendering in (a) provides a 360 degrees FOV, while the one in (b) has duplicated the rendering from (a) to create a rendering with a 720 degrees FOV.

matrices takes two lists of coordinates as arguments, and maps the coordinate at index x in the first list to the coordinate found at the same index in the second list. Therefore, the selection of points must be in ascending order along the x-axis, so that the coordinates are transformed to their corresponding point in the reference image. Because we are dealing with images of ultrawide FOV, two neighboring points on each side of the image's border might be selected. In this case, two of the rightmost control point would return a value less than the previous, resulting in a heavily warped image which in no case would align perfectly with the underlying synthetic panorama. To avoid this, we implemented a few safety mechanisms that prevent the user from being able to warp the images.

The x coordinates are left-to-right, so the x values should be in ascending order. If one value is less than the previous, we know that the rest of the values belongs to the image's other side. To make these pixels' coordinates compatible with the homography transformation provided by OpenCV [80], we iterate through the x coordinates and store each previously seen x value. If the current x value is less than the previous, we know that the selection has crossed the rightmost edge of the image, and that the following x values are positioned in ascending order from the leftmost image edge. Such image egde crossings can lead to coordinates being mapped to the opposite side of the images, resulting in distorted images. To avoid such distortions, we duplicate the rendering in the x-direction to create a 720 degrees view, which is shown in Figure 3.6. Because the viewing sphere presenting the rendering to the user maxes out at 360 degrees, the worst-case scenario would be the selection of $(image_{width}, y)$ followed by $(image_{width} - 1, y)$. The application then interprets this as $(image_{width}, y)$ followed by $(image_{width} * 2 - 1, y)$, which is therefore within the bounds for any of the following selections.



(a) The sensed image (b) The reference image (c) Aligned images

Figure 3.7: The panorama alignment method. (a) and (b) present four corresponding control points selected in each image. (c) illustrates the panorama aligned with the terrain in the render.

3.3.3 Image transformation

After the control point registration has been performed by the user, the program automatically estimates the parameters of the mapping function needed to transform the panorama to the render. This automatic process utilizes the established correspondence between the images to compute a transformation matrix, called a homography matrix [80]. Planar homography relates to the transformation between two image planes. By choosing a control point coordinate from the rendering, (x_1, y_1), and its corresponding point in the panorama, (x_2, y_2), we mapped the first coordinate to the second using Equation 3.2, where **H** is a $3x3$ matrix [68]. Lying on the same plane, Equation 3.2 is valid for all corresponding control points in both images. Figure 3.7 illustrates the transformation process.

$$\begin{bmatrix} x_1 \\ y_1 \\ 1 \end{bmatrix} = \mathbf{H} \begin{bmatrix} x_2 \\ y_2 \\ 1 \end{bmatrix} \tag{3.2}$$

3.3.4 Employing renderings for azimuth estimation

Above, we introduced some of the renderings' default characteristics. The panorama's center-point, where the x value of a pixel location is $image_{width}/2$, always looks towards the north with an azimuth of zero degrees, as also illustrated in Figure 3.3(b). The edges, where the x-value is either 0 or $image_{width}$, have an azimuthal angle of +/-180 degrees. For azimuth estimation, our method uses the upper and lower bounds of a panorama's FOV to extract the centermost x values between these edges. Hence, a method for calculating a panorama's FOV is required. To approximate a FOV, we exploit these known image characteristics and combine them with what we know about the transformed image. For any x value, we compute azimuth

a for x value p_x using the following formula, where IW is the render's width, given by the number of x-axis pixels:

$$\alpha = (p_x * 360/IW) \bmod 360 \tag{3.3}$$

Observing Figure 3.7(c), we see that the panorama's shape looks like a trapezium where no sides of the image are parallel. It is infrequent for panoramic images to position essential compositional elements, such as the horizon line in mountains in our case, towards the lower part of the image, as this results in images with very much sky and little foreground or terrain. We can not guarantee this will never happen, but most images position the horizon in the upper two-thirds of the image. Thus, transforming the images leads to geometric shapes with either bit of difference in the length of the horizontal edges or where the top edge is shorter than the bottom, as illustrated in Figure 3.7(c). In images where the upper edge is significantly longer than the lower one, there will be a more significant deviation in the viewing angle of the lower and upper edges. In some cases, using the longest edge might approximate the FOV wider than the panorama's actual one.

To reduce the risk of over-estimating a panorama's FOV, we used its transformed counterpart's shortest horizontal edge and extract its extremities. We estimated the edges' azimuths using Equation 3.3, subsequent by the following formula to estimate the FOV f between leftmost pixel lb and rightmost pixel rb:

$$f = (ub - lb) \bmod 360 \tag{3.4}$$

Given the lower-bound heading lb computed using Equation 3.3 on the pixel coordinate $bound_{left}$, along with the FOV f computed using Equation 3.4, a panorama's final azimuth estimation is done using this formula:

$$v = ((lb + (f/2)) + 180) \bmod 360 \tag{3.5}$$

Upon completing the image registration and mapping process, the program has generated a selection of images and additional metadata. The panorama explorer includes visualizations that present the transformed image and its digital terrain render. To avoid the need for reselecting control points and transform the images more than once, the additional files created in the process are stored, and the panorama's azimuth is appended to its metadata for easier retrieval later in the program.

3.4 Visibility determination

Fulfilling R_1's sub-requirement of superimposing information related to the visible mountain peaks in a panorama requires a method for determining geospatial objects' visibilities. As presented in Section 2.6, there exists multiple methods utilizing DEMs to identify visible peaks. Others rely on images' contents, and some combine the mentioned methods [5]. Because PanoVis already uses a DEM for creating renderings, as mentioned earlier in this chapter, we used the same elevation data sets for visibility determination. Our process begins with calculating a visibility map of the area surrounding an image's location, called a *viewshed*.

Viewsheds are binary images computed from DEMs, where each raster cell is assigned a value of *visible* or *invisible* from a given viewpoint. They are computed using several methods, but the most common techniques utilize *Line Of Sight* (LOS) or *reference planes* algorithms [124]. The two most popular open-source implementations for creating viewsheds build on the LOS algorithm proposed by Haverkort et al. [35] or the reference plane algorithm proposed by Wang et al. [115]. The most noticeable difference between the two is that the reference plane algorithm computes viewsheds in constant time for all viewpoints on the DEM, in contrast to the LOS algorithm, in which the computing time varies with the viewpoint placement. We favored the more predictable computing times introduced with the reference plane algorithm and used this method for creating viewsheds in our application.

Computing viewsheds are time-consuming, but this process is only needed once per viewpoint in our data set and is therefore done when uploading a new panorama. Once the viewshed is created, we can do the much faster task of coordinate look-ups in the binary image rasters to determine if a given object is visible, or not. The mountain peak data used in our program consists of objects embedded with geographical world coordinates in the *World Geodetic System 1984* (WGS 84) format [118]. The WGS 84, a type of *Coordinate Reference System* (CRS), is rarely used for DEMs covering smaller areas, in which case more region-specific ones are used. To generalize the coordinate look-up to support DEMs covering any parts of the Earth, a method for converting coordinates between different CRSs is required. As this task is fairly common in geodesy, we rely on third-party libraries for this task. Chapter 4 presents an overview of the libraries implemented for converting coordinates, but the methods implemented in these libraries are outside our scope to be described.

After converting a mountain peak's coordinates, these coordinates are used to index the viewshed's cell that includes the given location. As described earlier in this section, viewsheds are binary images representing either visible or invisible cells of a DEM. From every viewpoint in our data set, meaning the location of panorama creation, a set of visible mountains is created and stored along the image metadata. Additionally, the locations of all other viewpoints in the data set are added to another set, if visible, to be used for annotating the panoramas with smaller thumbnails of the other images visible from the given viewpoint.

3.4.1 Calculating visible objects' spherical positions

Once a list of visible objects is created, the resulting data set is used in several of the application's visual components. One of these components is the panorama sphere, where multiple geospatial data about the visible items are superimposed onto the panoramas to meet R_1. It was preferable to position these annotations as close as possible to the object's location in the image. Thus, a method for converting their geospatial coordinates to spherical coordinates relative to the location of the viewpoint was required.

Figure 3.5 illustrates that spherical coordinates consist of a pair of azimuthal and altitude values. In order to obtain the azimuth for positioning an object in the image sphere, we used the formula presented by Bullock [13]. Longitudes are denoted by L and latitudes by ϕ, where positive latitudes represent the northern hemisphere, and negative latitudes represent the southern hemisphere. In contrast to Bullock's formula, east longitude is considered positive to reflect how longitudes are formatted in the WGS 84 format, and west is considered negative. The viewpoint is denoted by A and includes the latitude ϕ_A and longitude L_A. The object to be positioned is denoted by B and includes the latitude ϕ_B and longitude L_B. We obtain an object's azimuthal angle θ using the following formula, where the values of A and B are given in radians, and ΔL equals $L_B - L_A$:

$$\begin{aligned} S &= \cos\phi_B \sin\Delta L \\ C &= \cos\phi_A \sin\phi_B - \sin\phi_A \cos\phi_B \cos\Delta L \\ \theta &= \operatorname{arctan2}(S, C) \end{aligned} \tag{3.6}$$

Because of the short distances between the viewpoints and the visible objects nearby, we computed an object's altitude by approximating a flat Earth and Pythagoras' theorem, as illustrated in Figure 3.8. H_A denotes the viewpoint's elevation, and H_B denotes the object's elevation. The Euclidean distance between the points is denoted by d, and ΔH represents the difference between the two elevations. With this information, we compute the altitude angle α by $\arcsin \frac{\Delta H}{c}$.

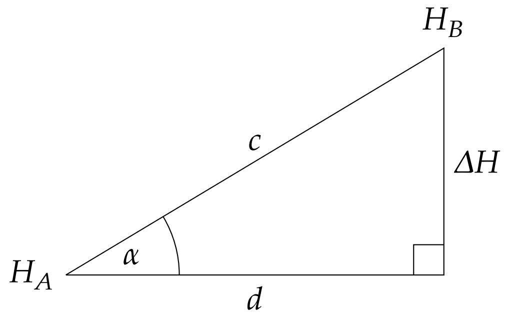


Figure 3.8: The altitude angle α of an object H_B relative to a viewpoint H_A.

3.5 Visual components

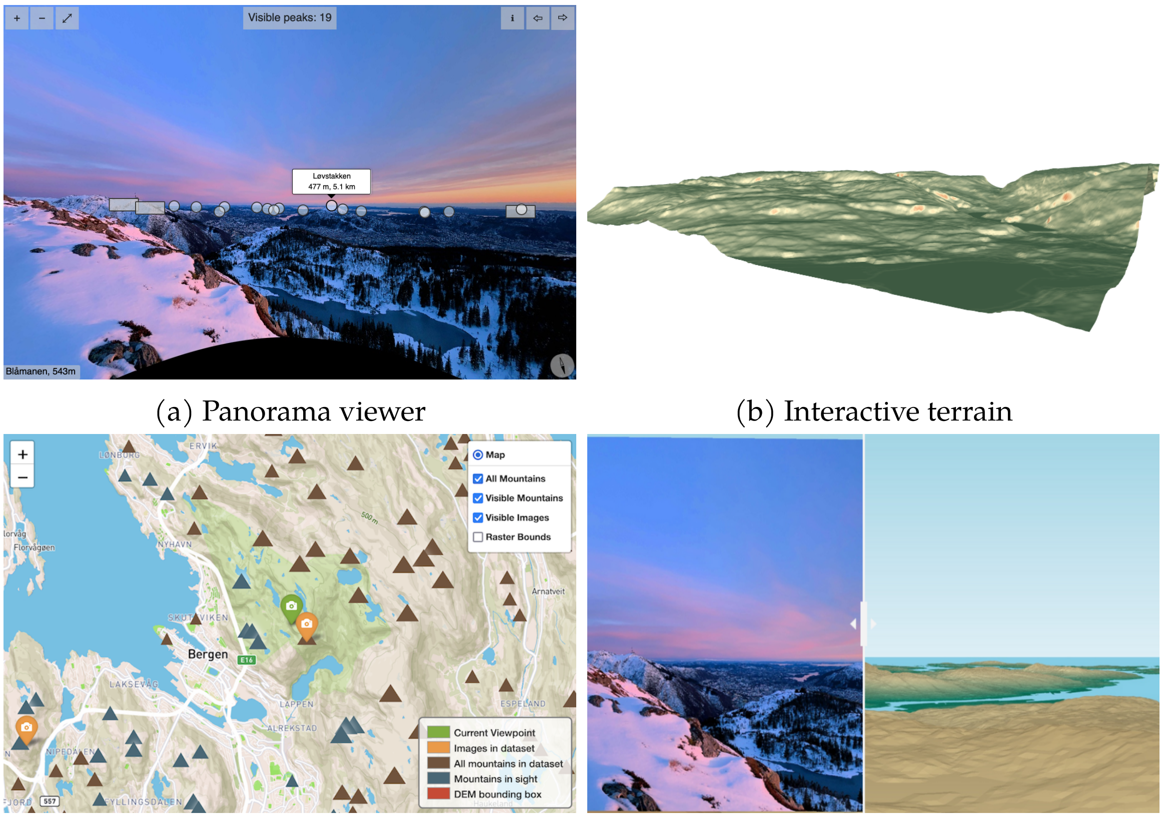
As aforementioned, the panorama explorer consists of multiple visual components. Figure 3.9 presents a preview of each component sorted after the order of occurrence in the panorama explorer.

3.5.1 Panorama viewer

Upon opening the panorama explorer, the first visual component that comes into view is the panorama sphere. Due to our great emphasis on the utility of panoramic images as the root of hiking inspiration in our application, we consider this view the main component of PanoVis. The panorama sphere, illustrated in Figure 3.10(b), has been added to the application to meet R_1 and fulfills the goal of displaying panoramas in a non-flat manner.

Several methods can present the images that wind around a virtual viewpoint. One alternative is to use cylindrical panorama viewers, illustrated in Figure 3.10(a). As panoramic images typically use cylindrical projections, cylindrical panorama viewers are a good choice for preserving the panoramas' looks. It reflects how the images are created when the camera is rotated around its vertical axis. However, modern cameras have no standard focal lengths, and smartphones often include multiple cameras. If an image's vertical FOV reaches values close to 180 degrees, cylindrical panorama viewers tend to make the landscape look more flat. This is not favorable when viewing images of mountainous landscapes, as it may drastically impact the viewers' impressions of the area.

Spherical panorama viewers utilize images with fixed characteristics to provide unrestricted views horizontally and vertically. For best results, such image viewers require equirectangular photographs, providing unrestricted views both horizontally and vertically. For images with narrower FOVs, trimming can be applied to avoid stretching the image, resulting in images not covering the whole viewport, as illustrated in Figure 3.10(b). Due to the flexibility of spherical panorama viewers



(a) Panorama viewer (b) Interactive terrain

(c) Map view (d) Juxtapose view

Figure 3.9: The panorama explorer's visual components

compared to the more restricted cylindrical ones, we favor the spherical method as it supports both cylindrical and spherical panoramas, even though cylindrical panoramas might be a bit distorted when projected onto a sphere.

To display panoramas in the panorama sphere, we either need a method for calculating the amount of FOV trimming needed to preserve the panorama's appearance, or we need to map the panoramas to equirectangular images. Recalling the image registration process described earlier in this chapter, the panoramas were mapped to their corresponding control points in equirectangular renderings. Using the same transformation matrix computed through the image registration process, we can map the panoramas to images with the same dimensions as the renderings but with all black pixels. This transformation makes panoramas look semi-cylindrical for narrow images with limited vertical FOVs. The results for images with horizontally and vertically large FOVs are images with spherical characteristics.

The visible objects' spherical positions described in the previous section are used to annotate the photographs displayed in the panorama sphere. These annotations include information about visible mountain peaks and, if applicable, functionalities for "teleporting" to other image locations visible in the panorama. Hovering an

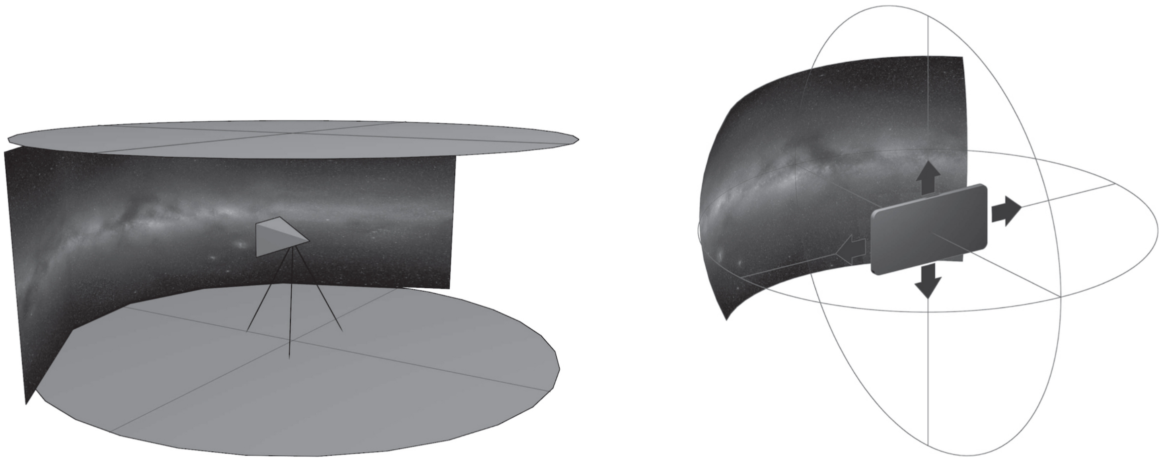
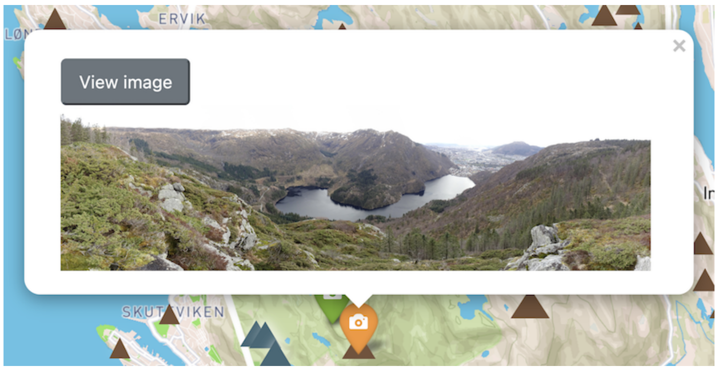

(a) Cylindrical panorama viewer (b) Spherical panorama viewer

Figure 3.10: Illustrations by Kent [47] showing how the panoramas wrap around a viewpoint in a cylindrical or spherical image viewer.

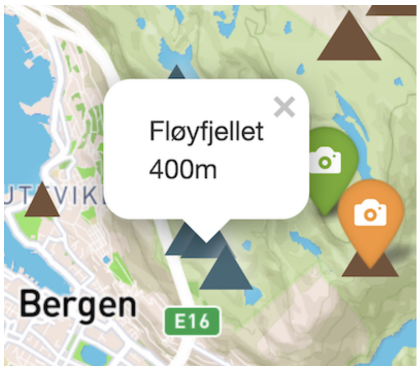
image annotation object presents a preview of the panorama, and clicking it triggers a scene switch replacing the current image with the clicked one. This teleport effect enables the user to explore the panorama more immersively, similar to the three-dimensional virtual tours presented by several other researchers [8, 28, 45, 49].

3.5.2 Three-dimensional surface view

Another visual component of the panorama explorer is the three-dimensional surface plot seen in Figure 3.9(b), fulfilling R_2. This geovisualization utilizes the gridded data stored in the DEM used to create the renderings. Rather than showing the individual data points, it shows a functional relationship between a designated dependent variable x and the two independent variables y and z. The z values of the surface plots are determined by the elevation values of the DEM and interpolated between each cell of the data set, creating a continuous surface representing the elevations. The area on the surface model corresponding to the individual cells can contain multiple data attributes, not only the elevations from the DEM. Therefore, we want to include a metric for presenting the slope of the terrain to provide more information about the terrain. As the DEM can be interpreted as a two-dimensional array, its slope is computed using external libraries implementing the techniques described in NumPy's documentation [78]. Because of the continuous aspect of the slope and elevation data, we combined the two different grids into one visualization, using the slope values to color the surface grid. The reasoning for the color choices is explained in Subsection 3.5.6



(a) Image popup



(b) Peak popup

Figure 3.11: The map view superimposed with glyphs representing multiple instances of geospatial data.

3.5.3 Topographic map

Roth [95] defines *cartographic interaction* as the dialogue between a human and a map, mediated through a computer. He argues that cartographic interaction adds value to exploratory geovisualization. We provided the user with a different data perspective by including an interactive geographical map superimposed with geospatial data, which also accomodate the requirement of R_3 (Topographic Maps). The interactive map, illustrated in Figure 3.13, provides controls for further data analysis through filtering and details on demand. We group the data using appropriate visual encodings, such as colors, shapes, icons, and sizes.

We emphasized the term *interactive* in the map view to meet R_7. User interaction is used to zoom and pan the map, filter the data, and show details on demand, following the guidelines of Shneiderman's famous mantra [101]. Figure 3.13(b) and (c) illustrate how additional details about the mountain peaks are shown on user interaction and how the image locations reveal the corresponding image if an image icon is pressed. The image popup includes a clickable button, and pressing this triggers a global scene switch. The scene switch updates the viewpoint to the selected image location and the other views in the mashup, acommodating the requirement of R_5 (Coordinated Views).

3.5.4 Juxtapose view

The fourth view in our application is the *juxtapose view*. Juxtapose corrsponds to placing two objects side by side to highlight differences or similarities. We use this technique to visualize the image alignment following the image registration initialized on panorama upload. The juxtapose view presents a panorama overlaid on top of the corresponding synthetic terrain matching the image location and includes a slider that controls the visible portion of each image, illustrated in Figure 3.9(d). Horizontally sliding the vertical divider increases and decreases the visible portions of the panorama overlaid on top of the render.

The juxtapose view does not necessarily lead directly to better insight into the underlying data sets. However, it is very important in some cases if the image registration has been performed incorrectly. Incorrect image transformations may result in inaccurate placements of annotations in the spherical image viewer, and if such errors occur, the juxtapose view comes in handy for the user. By comparing the transformed panorama with the render, the user gets visual feedback on the quality of the image transformation, seeing where the image should have been transformed and comparing it to the actual transformation result. Therefore, we included an option for re-doing the image transformation for a given image, easily accessible from the juxtapose viewer. In addition, the same view is presented after the transformation process is complete to provide the user with a method to confirm that the transformation is sustained.

3.5.5 Marks and channels

Munzner [72] defines marks as basic geometric primitives depicting items or links, with *channels* controlling their appearances. In Figure 3.12, the author presents a set of channels grouped by their channel types, ranked by the attributes' effectiveness. Though depending on their data types, some of the most effective channels are the *color hue*, *spatial region*, *position on a common scale*, and *shape*. With this in mind, we carefully selected the channels that best suit our data for the various visual components of the panorama explorer.

To highlight the visible geospatial objects in a panorama, the panorama viewing sphere includes a set of marks representing the objects that were marked as visible through visibility determination. These objects are either mountain peaks or other geographically referenced images in the data set. To highlight the mountain peaks visible in a panorama, we utilize the identity channel *shape*. Each mountain peak object is marked with a semi-transparent circle. The reason behind its circular shape was to limit the marks standing out too much while superimposed onto the panoramas. By default, the marks only present the information they include through user interaction, such as hovering. If the user prefers to display mountain peaks' names and their elevations without needing to hover on each mark, the panorama sphere includes a "show all" button. Pressing this button appends an arrow-like banner to be displayed above each mark to present the information otherwise only accessible on hover-interactions.

Marks representing visible panorama locations are supplements to the way of

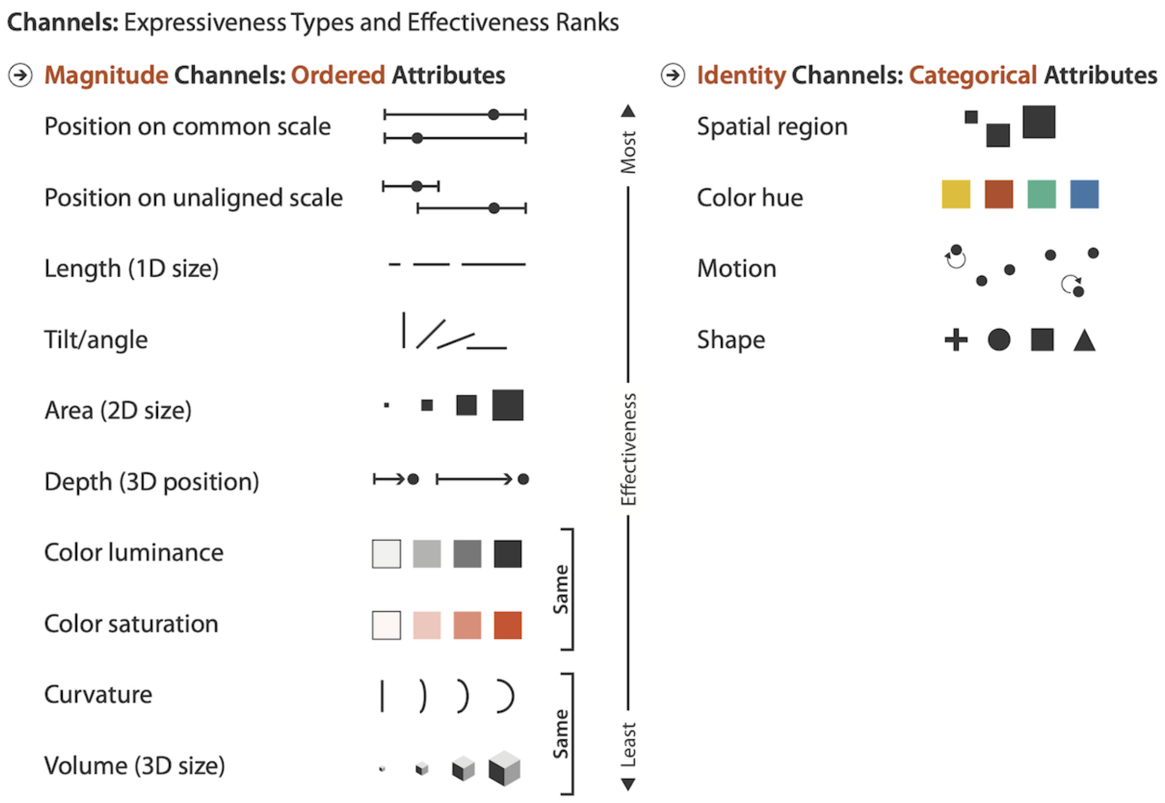


Figure 3.12: Munzner's [72] channel effectiveness ranking.

navigating between the photographs. As these marks include a thumbnail of the image corresponding to the location and shortcuts for teleporting to the given panorama, these marks are given a more protruding shape. A larger rectangular shape distinguishes the image marks from the ones representing mountains. Due to the panoramas being a core element of the geospatial data exploration, these marks are displayed at a relative distance closer to the virtual observer. This means that if a mountain mark is positioned at the same position as an image mark, the image mark will be easier to access for the user.

To differentiate the visible peaks from the non-visible ones in the data set, we introduced an identity channel to control the marks; the shape. The triangular shape is dissimilar to most other items on the map, making the peak marks easily distinguishable from the background. We also included a magnitude channel utilizing the elevation values of the data set. The ordered data is used to control the size of the peak marks, scaled accordingly to the minimum and maximum elevations in the data set.

Our processed data sets include both visible and non-visible peaks. By default, our interactive map visualization only presents the peaks visible from the viewpoint, corresponding to the set of peaks annotated onto the panorama in the spherical image viewer. We expanded the user's data insight by including s a settings panel where the user can toggle the visibility of specific parts of the data set using

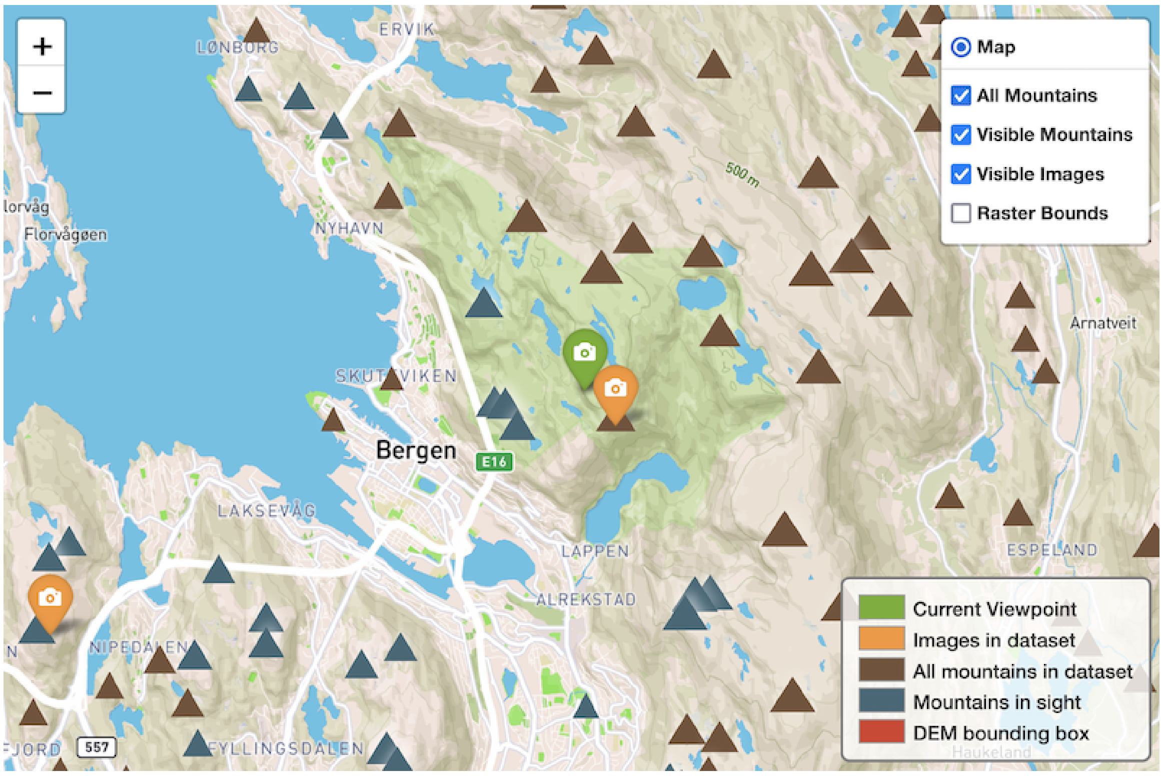


Figure 3.13: The interactive map view superimposed with glyphs representing multiple instances of geospatial data.

categorical attributes. The non-visible peaks are also shaped as triangles to symbolize the looks of a mountain top and scaled according to the elevation values. We use color as our identity channel to differentiate the visible peaks from the non-visible ones, as seen in Figure 3.13.

Other marks superimposed onto the map include the geographical locations of all other images in the data set. A colored pin containing a camera symbol is used to represent the location of each image, and a color hue is applied to distinguish the current view from the other image locations. Further reasoning about the colors selected for the application is given later in this chapter.

3.5.6 Color choices

As color affects every aspect of visualization [117], describing our color choices is important. The panorama explorer employs a selection of colormaps and themes throughout its visual components. The map view features a theme carefully crafted to showcase the topography of mountainous regions. We designed our custom map using the third-party mapping utility MapBox [69], which features vast amounts of crowdsourced geospatial data regarding landmarks, national parks, and mountains, among others. Encoding such features in the color hue channel ensures that these visual elements stand out among the other data displayed on the map, a phenomenon described as *popout* by Ware in his oft-cited *Perception for Design* [117]. Accordingly, a light-grey color was selected as the map's base color, laying the foundation for using darker colors for marks and labels added on top of the map to make these easily distinguishable.

Furthermore, we used a green color palette for all elements categorized as natural environments, including greenspace and wildlands. Including green in the map's color theme needed us to be cautious about what other colors to include. About 10% of the world's male population and 1% of the female population have some form of color blindness, of which the most common ones are protanopia and deuteranopia [117]. Both these dichromatic color vision conditions cause inabilities to discriminate red from green [113]. Following the inclusion of a green color family as described earlier, we avoided using red coloring for the most important pieces of information on the map.

Ware's basic guidelines were used to achieve sufficient distinctiveness for the marks representing mountain peaks from the background. Their guideline 4.13 [117, p. 124, G4.13] reads as follows: *"When color coding large background areas to be overlaid with small symbols, consider using all low chroma, high-value (pastel) colors for the background, together with high-chroma darker colors for the overlaid symbols."* Resultingly, darker colors were used to color the locations of both visible and non-visible mountain peaks.

The three-dimensional terrain features a color theme somewhat distinctive from the one used for the map view. Following the use of green colors for natural environments in the map view, this view employs a darker-green color as the surface's base. However, the terrain's coloring depends on its slope degree; therefore, steeper terrain features color towards the red color spectrum to signify more extreme elevation differences. Although green-red colormaps are usually deemed a not-so-good choice, red is commonly associated with danger [117]. Blue-green colormaps could also be used to highlight the differences in the degree of slope, but we preferred to have a base color of green to to its similarities with the natural areas of the topographic map. Hence, a green-red colormap was used to highlight certain parts of the terrain that the users should recognize to avoid too steep terrain.

We included a "you are here" mark on the three-dimensional terrain to indicate the viewpoint's positions in the landscape, following another one of Ware's guidelines [117, p. 374, G10.9]. The mark features a highly saturated red color for maximum discrimination. It avoids confusion with the reds that color the slope degrees by being much more saturated than the colormap. Resultingly, users can locate the position of the viewpoint quickly.

3.5.7 Immediate visual feedback

R_6 states that the application should provide visual feedback at all times to signal that the application is always running, even when heavy processing is done in the background. *Perceived performance* is the term used for how quickly an application appears to perform given a specific task and is an integral element of building user trust and holding attention [39]. To increase the perceived performance of a program, researchers have developed a set of techniques that can provide visual feedback to the user. Continuous feedback reassures the user that the system is working and gives them something to look at while waiting [7].

The most time-consuming processes of our application is rendering the digital terrain images, creating the viewshed, and visibility determination. These processes all require a couple of seconds to complete, and the application's user interface lacks interactiveness during these tasks. When such delays are experienced, users can easily become annoyed [7, 39, 48, 109]. To decrease the user's perceived time during these processes, our application relies on a set of techniques that can be used to provide visual feedback to the user.

Söderström et al. [103] researched how different loading indicator speeds affect the users' perception of time. The results showed that the faster the animation speed is on a passive loading screen, the more likely a user would prefer it. The perceived time a user gets on a passive loading screen also correlates to the loading animation. Therefore, in the object identification process, we include an animated loading indicator to visually provide feedback to the user that the application is still processing the data.

Kim et al. [48] studied how a loading symbol's progress function significantly affects the user's perception of waiting time and concluded that loading symbols showing progress estimates feel significantly faster than repetitive spinning symbols. They highlight that design factors, such as symbols and shapes, are less important, but more preferable design choices may make the user feel less bored while waiting. Our application provides a loading bar to reduce the perceived time waiting for the object identification process to complete.

For rendering the virtual terrain and creating the viewshed from the panorama location, we used another technique for reducing the perceived time. While these processes are happening in the background, the user is prompted with the first step in the panorama alignment process, where control points in the panorama will be selected. This includes the user in the process without knowing that time-consuming processes are happening in the background. If the user finishes selecting control points before the background processes are complete, the continue button is disabled, and a message telling the user to wait is displayed. If the user finishes selecting control points after the background processes are complete, the continue button is enabled, and the user has not experienced any delay.

3.6 Summary

In this chapter, the methods to fulfill most of the requirements we deemed important are presented and how they are realized in an application for exploring panoramic images accompanied by multi-modal geospatial data collections. The panorama explorer's visual components are presented, along with details concerning their visual encodings and justifications for the made choices. These visual components aid the users of PanoVis in exploring several instances of geospatial data centered around geographically referenced panoramas, providing important information about the area surrounding the image location. Coordinated views facilitate that the presented data always corresponds to the image in view, and interactivity integrates the user in an explorative way of analyzing this. Additionally, the interactive and explorative aspects expand to the stage of adding new panoramas to the system. The users engage in the image registration process and participate in estimating the panoramas' azimuths. In summary, the visual components satisfy all but requirement R_4, and in Chapter 6, we discuss our reasoning for not fulfilling all of these.

Chapter 4

Implementation

This chapter describes the implementation and the technologies employed in the application, supplemented with the reasoning behind the choices. We present an overview of the software architecture and analyze the technologies in detail.

4.1 Software architecture

We developed our application with an emphasis on rapid development and multi-platform support. Thus, we adopted Python as our programming language of choice due to its high flexibility and ease of use. We implemented our application in a modular fashion, with some modules relying on the output of the others. Figure 4.1 illustrates the data flow between the modules, and in the following we list them as they occur in the application pipeline:

- *Azimuth estimation*: The user aligns the panorama with a virtual terrain by interactively selecting a set of control points in each image as presented in a *Pannellum* [85] spherical image viewer. A transformation matrix is computed using the popular *Open Source Computer Vision* (*OpenCV*) [42] library, transforming the panorama to its corresponding position in the render. Leveraging the renderings' general properties, we used the pixel coordinates of the transformed panorama's edges to compute an azimuth estimate.

- *Calculating object visibilities*: The application relies on a pre-computed viewshed to detect visible objects in the nearby terrain. Our algorithm exploits this viewshed to calculate the mountain peaks' visibilities from a given viewpoint.

- *Generating visual components*: Pre-computes the necessary data structures and lays the foundation for the three-dimensional terrain view and the topographic map. It speeds up both processes by exploiting the filtered data from the earlier stages of the pipeline.

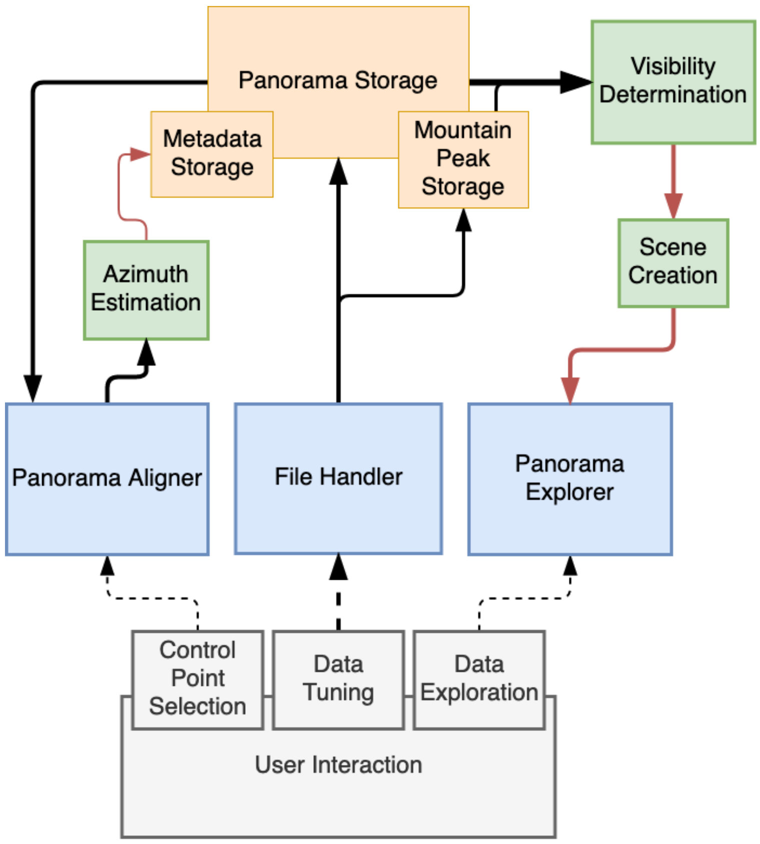


Figure 4.1: The overall software architecture of PanoVis. Black arrows represents user input data, e.g., panoramas and data sets containing geospatial information about mountain peaks. Red arrows represent processed data and dashed arrows represent user interaction. Yellow boxes represent data storage, green boxes represent data processing, and blue boxes represent client-side GUI.

- *Scene creation*: The backend computes the relative coordinates of each visible object to position the annotations in the spherical image viewer correctly. All relevant data is structured in the web-friendly JSON format and passed to the frontend.

- *Panorama explorer*: The geovisualization mashup combines the pre-computed scenes, topographic map, and the terrain view recipe to display the main window of PanoVis. A selection of JavaScript frameworks enables highly-interactive visual components throughout the application and glues the visual components together for achieving a coordinated view experience.

4.2 Backend

Some developers prefer low-to-middle-level programming languages like C++ with direct access to the computer's memory and *Graphics Processing Unit* (GPU) when high performance is paramount. Akin languages need compiling before code execution, ensuing real-time rendering of surface models synthesized from a DEM. Such implementations often require more powerful computers to run smoothly, and the implementation may be more time-consuming than high-level programming languages like Python. High performance was an advantage, but not a requirement for our prototype, as the amount of data that runs through the program will not be very large. With our emphasis on rapid development, we focused on more high-level programming languages that made it possible to quickly implement what was needed for a prototype.

Java is an example of a high-level programming language developers often use to create applications. Numerous add-on packages and libraries exist for it, enabling rapid development by relying on others' software instead of reinventing the wheel. Java, like C++, is a compiled language highly suitable for developing large and robust systems because of its strict syntax rules, blocking the ability to compile if errors are detected. As rapid development was a criterion for us, we deemed Java suitable for our needs, but before deciding, it was important to look at its highly popular alternative, Python.

Python consistently ranks as one of the most popular programming languages because its core philosophy emphasizes less-cluttered syntax and grammar [84, 106]. Compared to both Java and C++, Python is an interpreted programming language that directly executes instructions line by line without needing to compile them in advance. Interpreted languages are platform-independent, feature automatic memory management and garbage collection, and are easier to debug during development because of the lack of waiting for compiling between each run.

Java and Python support various add-on packages and libraries, and the largest open-source tools are mostly available for both programming languages. Thus, few features distinguish them, apart from Python's extra flexibility with *Just In Time* (JIT) compilation or Java's adequate performance. Because development pace was weighted more than performance for our prototype, Python was selected as the programming language for PanoVis' backend.

4.3 Frontend

The backend needed a web framework for communicating with the frontend, the part of the application that is visible to the user. GUIs use visual constructs that mimic physical objects such as buttons and sliders to provide intuitive methods for interacting with the computer [111]. GUIs present information in a manner that allows rapid assimilation and manipulation and is well suited for applications that

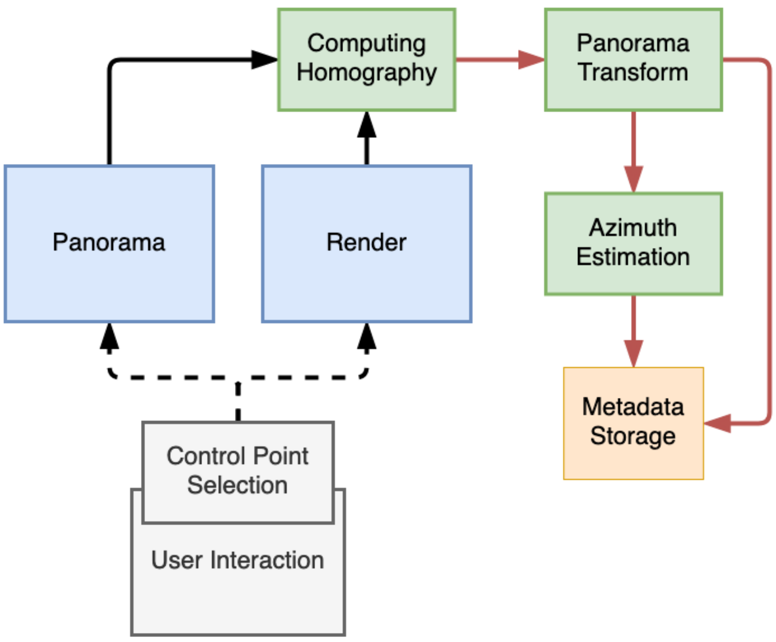


Figure 4.2: The azimuth astimation pipeline.

require user interaction.

Advantageously, our application provides visual methods for interacting with panoramas and other visualizations. Examples of such visual methods are native desktop applications or web application frameworks talking to the backend using an API. The latter was the preferred method, as it was more platform-agnostic and allowed remote access to the application through a three-tier architecture.

In a three-tier architecture, the client tier is the GUI. The GUI is where user interaction sends instructions to the application logic tier. The application logic tier communicates with the database to read and write data and the client tier to display the results.

A web application framework represents a collection of libraries and modules that enable the rapid development of web applications. Our web framework of choice for Python applications was *Flask* [81] because it is lightweight and enables a speedy setup using only a few lines of code. Flask runs a local web server that hosts the websites to interact with our application.

4.4 Estimating panorama azimuth

Our application requires an azimuth value for correctly positioning the annotations on the panoramas and keeping the compass in view up to date. As presented in Chapter 3, a panorama aligned with the render is used to estimate the azimuth. Therefore, estimating the azimuth begins with aligning the images through an im-

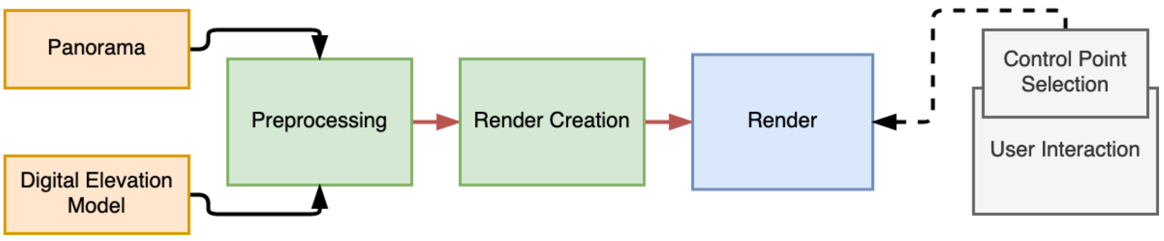


Figure 4.3: Render creation pipeline.

age registration technique, resulting in a transformation matrix needed for aligning the images.

The transformation matrix originates from the set of corresponding pixel coordinates in the panorama and renderings, demanding a GUI that supports registering the user's selections. A library sporting such functionality is Pannellum [85], a lightweight and open-source panorama viewer enabling interactive presentation of ultra-wide images in any web application. Although the library is lightweight, it includes several functionalities such as annotating the images with interactive marks and registering the position of mouse clicks performed by the user.

4.4.1 Rendering virtual terrain images

Simultaneously as the user selects control points in the panorama, the backend initiates a process for rendering an image of the digital terrain. To correctly interpret the DEM data, we used a combination of GDAL [96] and *Rasterio* [32]. Both are popular open-source libraries for reading, writing, and manipulating geospatial data sets. GDAL is written in C and provides a wide range of functionality through Python bindings. Although writing programs in C may lead to increased efficiency because of its low-level characteristics, it may also introduce several issues concerning C's dangling pointers leading to potential crashes [32]. As an improvement on GDAL, Rasterio desires to offer the same functionalities tailored for Python, succeeding in a more reliable environment with fewer program crashes. We considered such characteristics highly advantageous and thus preferred Rasterio over GDAL. However, Rasterio did not have a complete overlap of functionalities with GDAL, so in some cases, we resorted to a combination of both libraries in the application.

The rendering process begins with loading the provided GeoTIFF into the application, which is used with the renderer to create an image of the virtual terrain. We extract the panorama's geographical location by reading its *Exif* metadata to calculate the viewpoint used when rendering. We lighten the load of the program by cropping the DEM because it reduces the rendering time. The closest areas around the point of view are most relevant; hence, the scope of the DEM is limited to an area beginning twenty kilometers northwest of the viewpoint and extends equally far in a south-easterly direction. These values were carefully selected during the early development of PanoVis when an array of different DEM sizes were tested. We quickly learned that the larger DEMs added significant processing time to the rendering process, while adding a neglegible amount of ditant information to the virtual scenes. While performance was not a major focus for our prototype, DEM cropping was used to reduce the chance of finishing panorama control point selection before the render has been created.

For rendering the DEM in three dimensions, we selected the *Persistence of Vision Raytracer*, abbreviated POV-Ray [11], as our raytracing tool of choice. POV-Ray provides a wide selection of tools for rendering three-dimensional scenes, including support for visualizing digital terrains. It is open source and multi-platform, making it a viable option for our application. We used the panorama's location as our viewpoint and the elevation from the raster cell matching the location to adjust the height of the viewpoint. Subsequently, the backend handles the render to the application's frontend before displaying it to the user on completion of selecting the panorama's control points.

4.4.2 Image registration

On completion of control point selection in both images, the frontend passes two arrays of pixel coordinates to the backend for creating the transformation matrix. It uses the OpenCV [42] library, a huge open-source library that offers many tools used in computer vision to compute the transformation matrix. Consequently, the matrix is applied to each pixel in the panorama to align it with the rendered terrain. After aligning the images, the frontend presents the result, letting the user verify that the transformation was successful.

In addition to creating the transformation matrix and performing the image alignment, OpenCV is also responsible for locating the pixel coordinates of the corners in the newly-aligned image. We exploited these pixels to calculate the azimuth described in Section 3.3.

4.5 Visible objects detection

The visibility determination module is the key tenet for the application's image annotation functionalities. It exploits the geospatial aspects of the multi-modal data to determine an object's visibility, with the objects being mountain peaks or the other images in the data set. The application interprets both data sets as a list of `Item` data objects. As illustrated in Figure 4.4, an `Item` object includes a name, a location in the WGS 84 CRS, a location in the CRS specified in the DEM, and a path used for

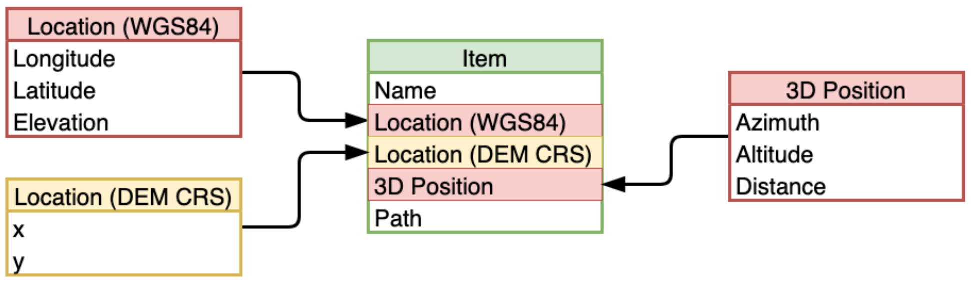


Figure 4.4: The `Item` data class. Yellow box represents data read directly from the underlying DEM, and red boxes represent data computed through helper functions.

URLs or file paths. Additionally, it includes the relative position of an object from a viewpoint computed using the method described in Section 3.4. We included two separate locations for each object, due to the function that converts coordinates from one CRS to another being computationally intensive to instantiates. To avoid unnecessary waiting times upon each coordinate conversion, we converted all coordinates while loading the data sets into the applciation.

While selecting control points in the render in the previous module, the backend generated a viewshed with the panorama's location as the viewpoint. Utilizing GDAL's built-in viewshed analysis tool, the backend appended a binary raster image to the panorama's metadata. Our object detection algorithm leverage this image to determine an object's visibility from the panorama location. We used the `Location (DEM CRS)` coordinates to look up the object's visibility value in the viewshed. Each pixel's color value can be either 0 or 255, translating to black or white. In our implementation, the white pixels are the visible ones.

The objects we wanted to determine the visibility for were the mountains and the other images in the data set. Recalling the `Item` object in Figure 4.4, both of these objects were structurally akin. Therefore, we implemented a generalized method for determining visibility for all objects in our data set. Looping through the set of objects created a subset containing only those whose location matches the pixel value of 255 in the viewshed.

4.6 Additional visual components

The panorama explorer includes multiple visual components. Two of these components, the three-dimensional terrain view and the map view, are created by the Python backend and loaded into the frontend on demand. Using a small portion of the DEM centered around the location of the panorama, the application uses Plotly [90], a graphing library available for both Python and JavaScript, to create a JSON file to be read by the browser for rendering the DEM data as a three-

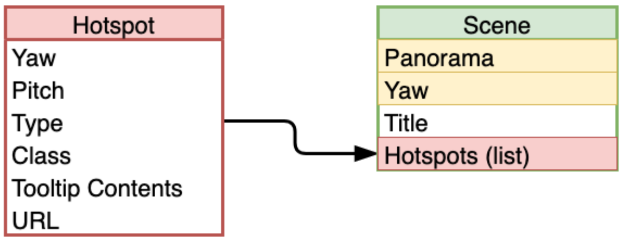


Figure 4.5: Scene data structure.

dimensional terrain. This rendering happens in real-time, allowing user interaction with the displayed data with the help of Plotly's JavaScript library, named *plotly.js*.

The backend utilizes *Folium* [108] to create the interactive maps presented in the mashup. Folium supports superimposing glyphs, icons, and images to interactive maps. Using Folium, we can group the different data sets to enable filtering of the presented data. The visualization departs the Python backend as a plain HTML file, which can be presented as a standalone visualization or as a view included in the geovisualization mashup.

4.7 Creating and presenting scenes

To present the scenes in the spherical image viewer and update the other views accordingly, the data needs to be formatted so that the frontend can easily interpret them. Figure 4.5 presents how Pannellum requires a scene to be formatted, where the term `Hotspot` is used for the annotations to be placed on the panorama. The yaw and pitch are the same as azimuth and altitude illustrated in Figure 3.5. A `Scene`'s yaw represents the panorama's azimuth used for initial viewing direction when loading a scene.

The Python backend prepares a set of all relevant data in a JSON file, transferring the prepared data to the GUI. When the browser initializes loading the mashup website, the backend delivers the JSON and triggers a global scene switch using a default `panorama-id` specified when constructing the scenes. The scene switch updates all coordinates views in the mashup to include the relevant images and metadata corresponding to the specific panorama. The user can now enjoy all the interactive views and geovisualizations processed in the previous modules of the application.

Chapter 5

Evaluation

The following chapter describes the results of user studies conducted to evaluate PanoVis. A pilot study was conducted, which led to some changes to the evaluation questionnaire. After that, a user study evaluated the application.

5.1 Evaluation methodology

An indication of whether the prototype works is necessary to evaluate the application's interactivity and explorative functions with respect to R_7 [3]. In our context, the user can explore panoramic images and interact with them to learn and discover information about the multi-modal data it combines through its user interface. Among a user interface's characteristics, usability is one of the most important [34]. The International Standardization Organization (ISO) defines usability as *"the extent to which a system, product or service can be used by specified users to achieve specified goals with effectiveness, efficiency, and satisfaction in a specified context of use"* [43], and a variety of methods exists to evaluate usability for numerous types of applications.

Given the exploratory nature of our application and geovisualization environments in general, user-based evaluations are the most suitable approaches to assess its usability and usefulness, according to the framework presented by Koua and Kraak [52]. User-based evaluations involve users completing tasks in the environment, enabling researchers to obtain quantitative data about an artifact's functionality. The industry standard for measuring usability is the *System Usability Scale* (SUS) [21], a questionnaire consisting of ten statements, each to be responded to using a *Likert* scale ranging from *strongly agree* to *strongly disagree*.

For the evaluation of PanoVis' usability, we conducted a user study where the evaluation participants got to explore the application and its features. The participants were selected to target a population of people who were most likely the application's users. We preferred that they were familiar with similar software for hiking inspiration and that they hike regularly. It was also advantageous that participants had experiences with panoramic photography, in which there was a chance that they would like to upload their images to the application.

Nielsen [76] reported in his oft-cited article that five user study participants were enough to catch 85% of a website's usability problems. Several researchers have suggested that the number of participants should be higher if the artifact being evaluated is to be used by several highly distinct groups of users, such as groups of both children and adults [23, 76, 105, 121]. Therefore, testing as many as possible was highly anticipated in evaluating the PanoVis system.

Although, due to limited time and resources, only five participants were selected for our evaluations. Three participants were informatics students with great experience using similar applications, and two were non-informatic students with less technical experience. All participants told us they were hiking at least once a month, therefore likely to be in the target group for the application.

To get started, we gave the participants the task of navigating from the launch window to the panorama explorer. There, the users would get some minutes to navigate through the images of the data set and gather information about the peaks visible from each image location. Our goal was to observe how the participants interacted with the spherical image viewer. As van Sommeren et al. [114] suggested, participants were encouraged to think aloud and verbalize their thoughts throughout the user study. They were allowed to ask questions and were given up to 20 minutes to explore the application. If the participants were stuck exploring only one of the views of the panorama explorer for more than half the allotted time, a verbal reminder alerted the participants that there were more visual components to explore.

Upon completing the image exploration, the participants reported that they had explored what they wanted in the panorama explorer. Afterwards, participants received instructions to upload a new panoramic image to the application. As presented in Chapter 4, this task positioned the application's user in command of the image registration needed for FOV and azimuth estimation, a central component of PanoVis. The participants followed the on-screen instructions and selected control points in the panorama and its corresponding render.

As suggested by Brooke [9], the participants filled out the SUS questionnaire after they had been given an opportunity to use the application but before any debriefing or discussion took place. After completing the questionnaire, the evaluation participants' responses were converted to a usability score using the standard SUS scoring formula listed in Appendix B. The scoring formula computes scores in the interval from 0 to 100, translated to the different types of usability scores given in

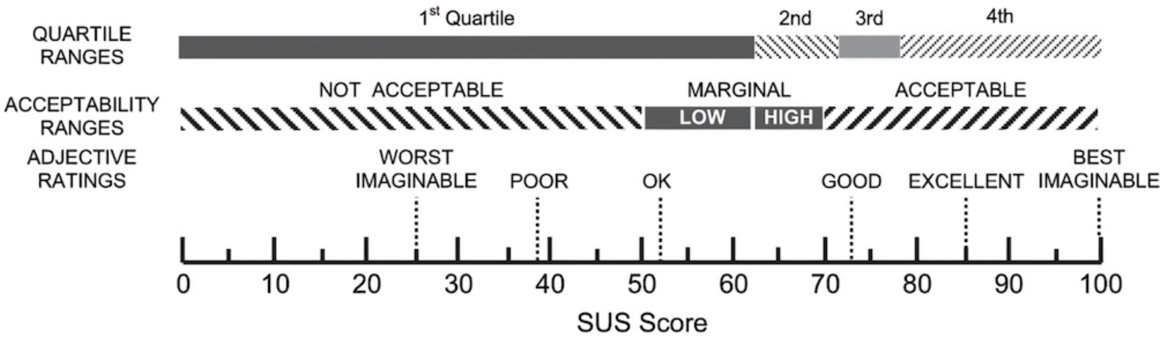


Figure 5.1: A comparison of mean System Usability Scale scores as presented by Bangor et al. [6].

Figure 5.1. In addition to SUS, the study's participants completed a semi-structured interview. Specific questions and discussion topics were prepared, which can be found in Appendix A.

5.2 Data adopted in the evaluation

The data set used in the user study consisted of twenty panoramic images photographed in or around the Bergen region of Norway. Before the user studies, nineteen of these panoramas were added to the application to provide the test participants a selection of images to explore without needing to upload all images themselves. The test participant added the final image to PanoVis's image storage during the evaluation using the image uploader. Figure 5.2 shows a selection of the panoramas used in the evaluation.

To annotate the panoramas with information about the visible mountain peaks, PanoVis required a predefined data set with such geospatial data. We used the online crowdsourcing platform for hike-related data to obtain such data, Peakbook [70]. The mountain peak data set retrieved consisted of 89 mountain objects, including the mountain's name, geographical coordinates, elevation, and a link to the Peakbook page for additional information.

Rendering the digital terrain images needed for the image registration task required a DEM. Because the images were all photographed around Bergen, a DEM covering this region was required. We obtained such a DEM from the Norwegian Mapping Authority [46], downloaded at a spatial resolution of ten meters.

5.3 Pilot study

In June 2022, a pilot study was conducted to verify that SUS was applicable to provide the information we needed about PanoVis' usability and that the following in-

Figure 5.2: Some of the panoramic images in the data set.

terview questions provided useful insights about the evaluation participant thinking about when using the application. The pilot study showed us that SUS was perfectly applicable for our application and that no changes needed to be made to the phrasings or the order of the statements. As a result, the standard SUS was kept as it was.

The follow-up questions presented to the evaluation participant on completion of the SUS evaluation were the four following questions:

- Is this something you can imagine can be used as a tool for planning hikes? (Why / why not?)
- How was your experience of the interaction in the program? (What was good / what could have been done differently?)
- Is there anything you would like to change in the application?
- Is there anything you want to add beyond the questions?

After completing the pilot study, the feedback provided lacked specific comments about PanoVis' visual components and its main view; the panorama explorer. Therefore, we included two additional interview questions for the final evaluations. The following questions were added; *"What could be done to improve the visual components on the application's main screen?"* and *"How did you feel about the panorama explorer?*." Appendix A lists the final interview questions used in the full user study.

5.4 Results

Table 5.1 presents the usability evaluations. After applying the scoring formula, the final row lists the individual scores for each participant. The SUS results range from

70 to 92.5, with the overall average being 79. As the table presents, two statements stand out positively; the fourth and the fifth statements. As every other statement in SUS is negatively loaded where ratings of 1 are the best, these statements have been inverted to positive phrasings to color the scores meaningfully, and their response scorings are also inverted. The fourth statement's actual phrasing was "I think that I would need the support of a technical person to be able to use this system," which all test participants strongly disagreed with. With the fifth statement being odd, the table presents it with the exact phrasing used in the questionnaire; "I found the various functions in this system were well integrated." Four out of five participants answered this statement with "Strongly agree," and the final participant's answer was "Agree." The lowest rated statement was "I think that I would like to use this system frequently," with an average score of 3.

	Statements:	P1	P2	P3	P4	P5	Avg.
S01	I think that I would like to use this system frequently.	4	2	4	2	3	3.00
S02	I found the product to be simple.*	3	4	5	4	5	4.20
S03	I thought the system was easy to use.	4	3	4	4	5	4.00
S04	I think that I could use the product without the support of the technical person.*	5	5	5	5	5	5.00
S05	I found the various functions in this system were well integrated.	4	5	5	5	5	4.80
S06	I thought there was a lot of consistency in the product.*	5	4	3	4	5	4.20
S07	I would imagine that most people would learn to use this system very quickly.	2	4	5	4	5	4.00
S08	I found the product very intuitive.*	3	4	4	4	4	3.80
S09	I felt very confident using the system.	4	4	4	4	5	4.20
S10	I could use the product without having to learn anything new.*	4	4	5	4	5	4.40
SUS	System usability scale results	70	72.5	85	75	92.5	79.00

Table 5.1: User-based evaluation results of the System Usability Scale. Statements marked with a star is rephrased to their positive form using the phrasings presented by Kortum et al.[51], and their scores have been inverted. The last row presents the traditional SUS scores calculated using the participants' original degrees of agreement or disagreement on the Likert scale.

The average overall score of 79 can be converted to the adjective rating "Good" using Bangor's scale in Figure 5.1, and we can consider PanoVis' usability acceptable. On the other hand, it is important to consider that the selection of participants is relatively small and cannot necessarily be regarded as generalizable. Therefore, the acceptability rating is given with some reservations.

5.4.1 Feedback from the respondents

Following the SUS questionnaire, the interviews provided useful insights into certain aspects of the application's components and functionalities. The study participants' feedback was majorly positive regarding our emphasis on user interaction in all of the application's components. P1, P4, and P5 expressed that they had a good experience with the interaction of the program and that it was easy to use. Furthermore, P1 reported no problems interacting with the application, although the same respondent highlighted that the image registration felt somewhat cumbersome. The participant experienced this process as cumbersome due to the constraint restricting the selection of control points in any given order, forcing the points to be selected left-to-right as described in Chapter 3. In contrast, P5 reported that the task of adding new images was a fun and unique experience that the study participant had not encountered before.

Despite all respondents scoring PanoVis' usability as "OK," "Good," or "Excellent," the feedback on when to use the application was somewhat polarized. P1, P2, and P5 expressed that they would see themselves using the application in advance of their hiking adventures, seeking inspiration through the images and gaining useful insights about the area by looking at the maps and three-dimensional terrain. On the contrary, P3 found a greater potential for the application's features after completing a hike, highlighting the panorama annotations as a great tool for displaying and sharing images photographed during hiking adventures.

In summary, the feedback from the test participants and the SUS questionnaire indicates that the application's overall usability is acceptable. As mentioned, the interview questions handed out to the participants included some more open questions, which most participants filled out. The next chapter discusses the open questions and their responses.

Chapter 6

Discussion

This chapter discusses the feedback from the study participants from the evaluation described in Section 3.1. Furthermore, we discuss multiple aspects of the application and the methodologies acquired during the project before highlighting where we see the potential for further development.

6.1 Design considerations and consequences

Chapter 3 described the reasoning behind the design choices in creating a prototype for the PanoVis system. However, some aspects of the current application design could be solved in multiple ways, and we can not state that the final design was the best way to solve the introduced problems. Nevertheless, our method was *one* way to solve these, and the evaluation results suggest that the application fulfilled its purpose.

6.1.1 Image registration

As previously noted in Section 5.4, one study participant expressed that the image registration task needed for aligning the panoramas and renderings was a bit confusing due to the strict constraints concerning the ordering of the selection of control points. The constraint resulted from technical limitations of how the control points are used to construct a transformation matrix. Our unanticipated findings during the evaluation indicated that it restricts the users' natural workflow at this application stage. An ideal solution for omitting these constraints could be introducing automatic methods for locating similar features in the panoramas and renderings. One such distinct feature of mountainous landscapes is the horizon edge, which Baboud et al. [5] exploits to achieve their robust image alignment algorithm aligns mountain pictures with renderings. Using comparable techniques would remove the need for manual interaction in this stage. However, such approaches' accuracy

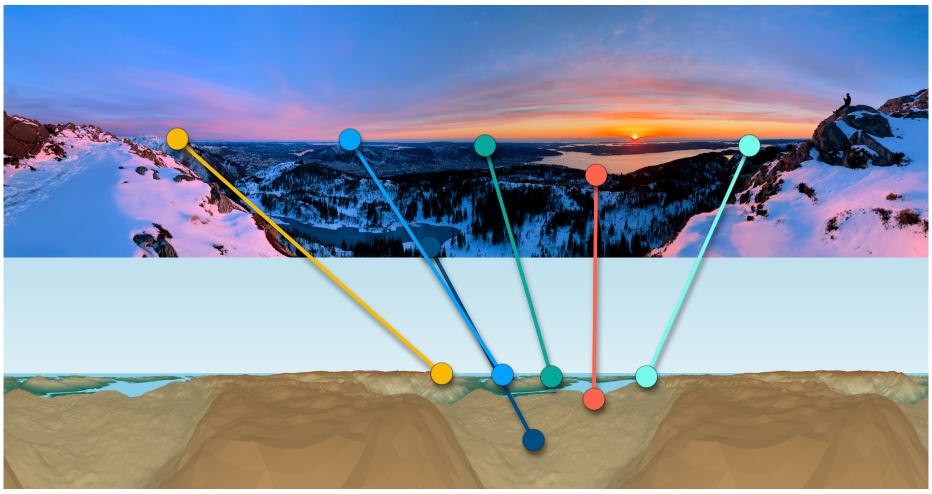


Figure 6.1: Example image of how automatic feature detection algorithms could be used to add control points to the panorama and render, where the points are connected by lines. Each point could be manually dragged to a new position if positioned ambiguously by the automatic method.

relies on several factors, including the image composition, quality, and the resolution of the digital terrain. Alternatively, a semi-automatic method could be used to automatically add a set of control points to the images while allowing users to modify the position of these through drag-and-drop interaction for fine-tuning eventual ambiguities from the automatic method, as illustrated in Figure 6.1.

Another aspect of our image registration technique is that it uses panoramas as sensed images and aligns them with renderings used as reference images. The method produces panoramic images with the same characteristics as the renderings, mimicking the equirectangular projection by applying a border around before displaying the transformed panorama in a sphere. While this method delivers promising results when the control points are selected carefully, it is prone to distorting the panoramas in some cases. Such cases include when the panoramas' horizon is too high or too low in the images or when the control points are not accurately placed. When a panorama's horizon is located towards the vertical extremes of an image, the transformation commonly results in images with heavily distorted regions towards the horizontal edges due to the render's horizon being positioned at its vertical center. Reducing such distortions is possible by selecting more control points towards these edges, but in practice, selecting and remembering two-digit amounts of control points is difficult and time-consuming. As a result, the annotations towards the edges of heavily distorted panoramas are often mispositioned. In some extreme cases, the annotations are positioned outside of the panorama.

Perfecting the control point selections through automatic methods could help reduce image distortions. On the contrary, another solution that would completely avoid the problems with image distortions is to reverse the roles of the panoramas and renderings in the image registration, aligning renderings with distortion-free panoramic images using the renderings as sensed images and panoramas as the reference. From here, calculating the position of annotations to be superimposed on the photographs would begin with the same method we presented in Section 3.4. In contrast, additional methods would be required to transform the annotation positions using the same transformation matrix applied to the render. Consequently, annotation positions would be calculated similarly to the method we used, requiring a conversion from azimuth and altitude to pixel coordinate in the render before shifting the pixels to their correct position in the panorama. After that, the annotations' pixel locations would be converted back to a pair of azimuth and altitude, allowing for superimposing annotations directly on the original panoramas. Moreover, changes would have to be made to how the panoramas are displayed in the panorama viewer, as it currently requires them to be displayed as a part of an equirectangular image frame.

Even though some panoramas are displayed with distortion towards the edges resulting in inaccurate positioning of marks, this was not a problem for the participants of our user study. We interpret the SUS scores as these ambiguities not affecting the overall experience of the prototype.

6.1.2 Visualizing digital elevation models

Our proposed application introduced multiple methods relying on geographical data gathered from DEMs, one of which was the raytracer responsible for creating the digital terrain renderings used for the image registration necessary for the azimuth estimation introduced in Section 3.3. Currently, the raytracer produces static images of equirectangular characteristics, which are displayed as a full sphere in the image viewer. Using the DEMs' data in such a way was a design choice for streamlining the display of both panoramas and renderings, enabling the application to use the same methods for extracting control points selected by the user in each image type. While potentially much more demanding to realize from a development perspective, an ideal solution would be to render the terrain directly in the browser. Visualizing the three-dimensional data in real-time is costly, often requiring high-performance GPUs. However, it would enable users to navigate along the virtual terrain instead of being locked to a specific viewpoint, possibly improving the explorative aspect of the application.

As a response to the open question at the end of the user study, one of the participants expressed that teleporting to any mountain peak and presenting a full 360 degrees view from that given viewpoint would be a great way to explore the terrain. Such functionalities could be realized through real-time terrain rendering while also being possible with the current project techniques. The method used to create the 360 degrees static terrain renderings described in Section 3.3 could easily be adapted to create such views from any viewpoint the users would like to explore. Although, rendering the renderings is time-consuming and could result in a user experience that is not as satisfying as the current one where panoramas are loaded instantly.

6.1.3 Scalability

During the development of PanoVis, the maximum number of uploaded panoramas never exceeded a hundred. Throughout our testing and evaluation of the application, this fair amount of panoramas added to the service was not majorly affecting PanoVis' performance. Currently, the application's most time-consuming processes include the control point selection during image uploading and the render creation happening in the background while the user aids in selecting control points. These tasks only happen once per image and do not affect the application's scalability. On the contrary, determining an image's visibility relies on a method that loops through all other uploaded images. Thus, adding new panoramas to the service add one more location visibility determination operation per image, which does not scale very well.

Moreover, computing an image's visibility happens every time the user navigates to the panorama explorer. When no new images are added or removed from the service, this leads to doing the same computations over and over again. Several measures could be taken to reduce the number of operations needed to determine an image's visibility. One such measure could be only to compute the image visibilities when images are added or removed and store the results for quicker lookups when no changes are done to the data set. Another measure could be to store the visibility results for each image and update the results every time a new panorama is added.

6.2 Future work

Several aspects of PanoVis can be improved in future development, and new features can be added to the application. Recalling the requirements given in Section 3.1, PanoVis failed to meet only one of these. Requirement R_4 refers to the usefulness of having real-world hike paths superimposed onto the panoramas, supplementing the images with even more geospatial information. Due to technical constraints and our emphasis on rapid development, developing such a feature would require refactoring the methods used to position and draw annotations onto

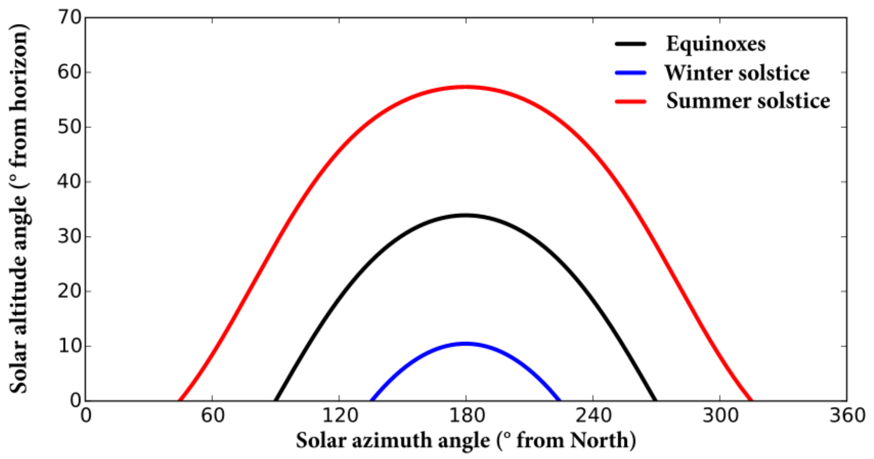


Figure 6.2: A diagram demonstrating three day arcs viewed from 56°N latitude. The illustration is based on user Deditos's work uploaded to Wikimedia Commons [17].

the panorama sphere. Thus, it was not prioritized for the development of our prototype. Although, we deem that developing such functionalities would add highly important information to any hiking-related application and that meeting R_4 should be emphasized in future work.

One participant of our user study expressed that it would be useful to project the Sun's path, sometimes called a *day arc*, onto the panorama sphere. Day arcs visualize the path of the Sun throughout the day as Figure 6.2 illustrates and enables quick readings for retrieving information about where the Sun rises and sets. Such a feature would enable hikers with a passion for photography to better plan when and where to photograph sunrises or sunsets and provide useful information about the available daylight hours for a given location. Using the panoramas' Exif metadata, the day arc could be based on the time and date the photographs were created and position the Sun as a mark on its corresponding position in the panorama. Additional settings could be used to adjust the time and date to provide insights into where the Sun could be seen at any given time.

One study participant noted that the juxtapose view would be even more useful if the user could experiment and select different blend modes for the images instead of the current solution, which uses the vertical slider to separate the transformed panoramas and the renderings. Using appropriate blending techniques, we could texture the digital terrain displayed in the renderings with colors extracted from the panoramas. One such technique could exploit the hue and saturation of the transformed panoramas and mix these with the luminance of monochrome renderings to create more realistic render images [88], as Figure 6.3 illustrates.

Furthermore, extracting the panorama colors could be exploited to create unique

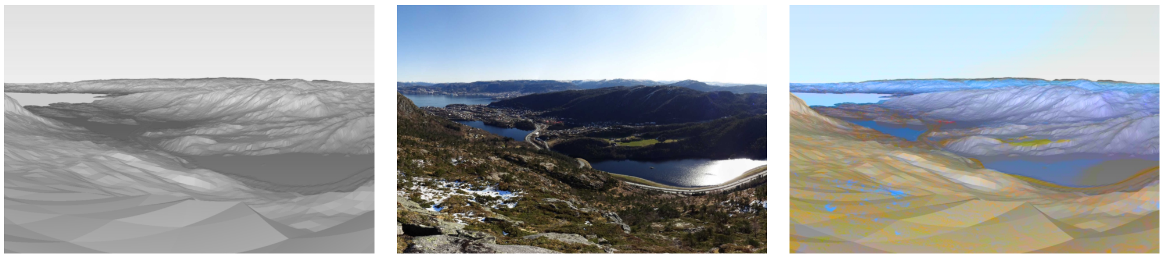

(a) The base layer. (b) The blend layer. (c) Color blend result.

Figure 6.3: Mixing the monochrome base layer's luminance with the hue and saturation of the blend layer results in an image that preserves the gray levels of the image, which can be used for coloring black and white renderings [88].

color palettes for the renderings. Currently, the color palette used to texture the digital terrain displayed in the renderings is based solely on the approximate vertical distance from the geographic reference point defined for the DEM and the terrain itself. Coloring the terrain in such a manner results in a terrain image not reflecting the actual terrain, and lakes positioned above the DEM's reference point are colored the same as the terrain surrounding it. Therefore, using a color palette based on the terrain could lead to more realistic render images. Additionally, some researchers have explored using satellite imagery to create naturally looking textures for rendering three-dimensional terrains [99]. Applying such textures to the terrain render displayed to the user for the image registration process could enable the user to more easily orient themselves in the terrain and, thus, possibly more confidently identify the common features needed for selecting control points.

Chapter 7

Conclusion

In this thesis, PanoVis is presented, an application that combines data of multiple modalities to create a seamless geovisualization mashup interface. Users can navigate the multi-modal data collections using a selection of coordinated views, with each view building on visualization principles. The presented approach includes a method for determining the azimuthal orientation of mountain panoramas through image registration techniques that exploit digital elevation data and user interaction, used for annotating panoramic images with marks representing geospatial data. Our approach was evaluated by five participants through a user study, whereof each participant reported they regularly hike. The evaluation resulted in an average SUS score of 79. Based on the evaluation results, we conclude that the presented approach is a viable solution for exploring multi-modal data collections facilitated through a geovisualization mashup interface.

As PanoVis was implemented as a web application using modern technologies, it lays the foundation for further development of a full-fledged geovisualization application suitable for the visual exploration of geospatial data and their corresponding panoramic images. With its modular fashion, PanoVis is developed to allow for the integration of future features and functionality.

Glossary

Azimuth: An angular measurement in a spherical coordinate system, measured clockwise from a north base line. True north is measured as a 0° azimuth, east 90°, south 180°, west 270° [4].

Camera pose: The position and orientation of a camera relative to some coordinate system.

Mashup: A geovisualization technique that combines content from more than one source to create an integrated end-user experience displayed in a single graphical interface [126].

Panorama: A photograph with a horizontally elongated field of view, providing an unobstructed or complete view of an area in every direction.

Rendering: In our context, a static image displaying a 360-degree view of a digital terrain synthesized from a digital elevation model.

Acronyms and abbreviations

API: Application Programming Interface.

CRS: Coordinate Reference System.

DEM: Digital Elevation Model.

DOF: Degrees Of Freedom.

DSM: Digital Surface Model.

DTM: Digital Terrain Model.

FOV: Field Of View.

GIS: Geographic Information System.

GPS: Global Positioning System.

GPU: Graphics Processing Unit.

GUI: Graphical User Interface.

JIT: Just In Time.

LOS: Line Of Sight.

SUS: System Usability Scale.

WGS 84: World Geodetic System 1984.

Bibliography

[1] D. An, "Find out how you stack up to new industry benchmarks for Mobile Page Speed," 2017-02, [Online]. Available: `https://www.thinkwithgoogle.com/_qs/documents/2294/64237_mobile-page-speed-new-industry-benchmarks.pdf` (visited on 2022-04-26).

[2] G. Andrienko, S. I. Fabrikant, A. L. Griffin, J. Dykes, and J. Schiewe, "GeoViz: Interactive maps that help people think," *International Journal of Geographical Information Science*, vol. 28, no. 10, pp. 2009–2012, 2014-10-03, ISSN: 1365-8816, DOI: `10.1080/13658816.2014.937719`.

[3] N. Andrienko, G. Andrienko, H. Voss, F. Bernardo, J. Hipolito, and U. Kretchmer, "Testing the Usability of Interactive Maps in CommonGIS," *Cartography and Geographic Information Science*, vol. 29, no. 4, pp. 325–342, 2002-01-01, ISSN: 1523-0406, DOI: `10.1559/152304002782008369`.

[4] *Azimuth*, in *Wikipedia*, 2022-09-21, [Online]. Available: `https://en.wikipedia.org/w/index.php?title=Azimuth` (visited on 2022-07-17).

[5] L. Baboud, M. Čadík, E. Eisemann, and H.-P. Seidel, "Automatic photo-to-terrain alignment for the annotation of mountain pictures," in *CVPR 2011*, 2011-06, pp. 41–48, DOI: `10.1109/CVPR.2011.5995727`.

[6] A. Bangor, P. T. Kortum, and J. T. Miller, "An Empirical Evaluation of the System Usability Scale," *International Journal of Human-Computer Interaction*, vol. 24, no. 6, pp. 574–594, 2008-07-29, ISSN: 1044-7318, 1532-7590, DOI: `10.1080/10447310802205776`.

[7] A. Bouch, A. Kuchinsky, and N. Bhatti, "Quality is in the Eye of the Beholder: Meeting Users' Requirements for Internet Quality of Service," p. 8, 2000.

[8] J. Brejcha, M. Lukác, Z. Chen, S. DiVerdi, and M. Cadík, "Immersive Trip Reports," in *Proceedings of the 31st Annual ACM Symposium on User Interface Software and Technology*, ser. UIST '18, New York, NY, USA: Association for

Computing Machinery, 2018-10-11, pp. 389–401, ISBN: 978-1-4503-5948-1, DOI: 10.1145/3242587.3242653.

[9] J. Brooke, *SUS: A 'Quick and Dirty' Usability Scale*. CRC Press, 1996-06-11, pp. 207–212, ISBN: 978-0-429-15701-1, DOI: 10.1201/9781498710411-35.

[10] L. G. Brown, "A survey of image registration techniques," *ACM Computing Surveys*, vol. 24, no. 4, pp. 325–376, 1992-12-01, ISSN: 0360-0300, DOI: 10.1145/146370.146374.

[11] D. K. Buck, A. A. Collins, and A. Enzmann, *Persistence of Vision Raytracer*, version v3.8.0-beta.2, Williamstown, Victoria, Australia: Persistence of Vision Pty. Ltd., 2021-08, [Online]. Available: https://github.com/POV-Ray/povray/releases/tag/v3.8.0-beta.2.

[12] A. Buja, J. McDonald, J. Michalak, and W. Stuetzle, "Interactive data visualization using focusing and linking," in *Proceeding Visualization '91*, 1991-10, pp. 156–163, DOI: 10.1109/VISUAL.1991.175794.

[13] R. Bullock, "Great Circle Distances and Bearings Between Two Locations," *MDT*, vol. 5, pp. 1–3, 2007-06-05, [Online]. Available: https://dtcenter.org/sites/default/files/community-code/met/docs/write-ups/gc_simple.pdf (visited on 2022-07-13).

[14] D. Carr, "Guidelines for designing information visualization applications," in *Proceedings of the 1999 Ericsson Conference on Usability Engineering*, 1999, [Online]. Available: http://urn.kb.se/resolve?urn=urn:nbn:se:ltu:diva-39533 (visited on 2022-06-20).

[15] A. Çöltekin, S. Bleisch, G. Andrienko, and J. Dykes, "Persistent challenges in geovisualization – a community perspective," *International Journal of Cartography*, vol. 3, pp. 115–139, sup1 2017-10-30, ISSN: 2372-9333, DOI: 10.1080/23729333.2017.1302910.

[16] A. Çöltekin, H. Janetzko, and S. I. Fabrikant, "Geovisualization," *Geographic Information Science*, vol. 2018, no. Q2, online, Q2 2018-04-01, ISSN: 2577-2848, DOI: 10.22224/gistbok/2018.2.6.

[17] Deditos, *Solar Declination*, 2015-01-17, [Online]. Available: https://commons.wikimedia.org/wiki/File:Solar_declination.svg (visited on 2022-07-31).

[18] U. Deuschle. "MakePanoramas." (2018), [Online]. Available: https://www.udeuschle.de/panoramas/makepanoramas_en.htm (visited on 2022-06-13).

[19] J. Döllner, "Chapter 16 - Geovisualization and Real-Time 3D Computer Graphics," in *Exploring Geovisualization*, ser. International Cartographic Association, J. Dykes, A. M. MacEachren, and M.-J. Kraak, Eds., Oxford: Elsevier, 2005-01-01, pp. 325–343, ISBN: 978-0-08-044531-1, DOI: 10.1016/B978-008044531-1/50434-6.

[20] Y. Doytsher, S. Dalyot, and Y. Katzil, "Digital Terrain Models: A Tool for Establishing Reliable and Qualitative Environmental Control Processes," in *GeoSpatial Visual Analytics*, R. D. Amicis, R. Stojanovic, and G. Conti, Eds., Dordrecht: Springer Netherlands, 2009, pp. 215–234, ISBN: 978-90-481-2899-0, DOI: 10.1007/978-90-481-2899-0_18.

[21] M. R. Drew, B. Falcone, and W. L. Baccus, "What Does the System Usability Scale (SUS) Measure?" In *Design, User Experience, and Usability: Theory and Practice*, ser. Lecture Notes in Computer Science, A. Marcus and W. Wang, Eds., vol. 10918, Cham: Springer International Publishing, 2018, pp. 356–366, ISBN: 978-3-319-91797-9, DOI: 10.1007/978-3-319-91797-9_25.

[22] J. Dykes, A. M. MacEachren, and M.-J. Kraak, "Exploring Geovisualization," in *Exploring Geovisualization*, ser. International Cartographic Association, J. Dykes, A. M. MacEachren, and M.-J. Kraak, Eds., Oxford: Elsevier, 2005-01-01, pp. 1–19, ISBN: 978-0-08-044531-1, DOI: 10.1016/B978-008044531-1/50419-X.

[23] L. Faulkner, "Beyond the five-user assumption: Benefits of increased sample sizes in usability testing," *Behavior Research Methods, Instruments, & Computers*, vol. 35, no. 3, pp. 379–383, 2003-08-01, ISSN: 1532-5970, DOI: 10.3758/BF03195514.

[24] R. Fedorov, D. Frajberg, and P. Fraternali, "A Framework for Outdoor Mobile Augmented Reality and Its Application to Mountain Peak Detection," in *Augmented Reality, Virtual Reality, and Computer Graphics*, L. T. De Paolis and A. Mongelli, Eds., ser. Lecture Notes in Computer Science, Milan, Italy: Springer International Publishing, 2016, pp. 281–301, ISBN: 978-3-319-40621-3, DOI: 10.1007/978-3-319-40621-3_21.

[25] R. Fedorov, P. Fraternali, and M. Tagliasacchi, "Mountain Peak Identification in Visual Content Based on Coarse Digital Elevation Models," in *Proceedings of the 3rd ACM International Workshop on Multimedia*

Analysis for Ecological Data, ser. MAED '14, New York, NY, USA: Association for Computing Machinery, 2014-11-07, pp. 7–11, ISBN: 978-1-4503-3123-4, DOI: 10.1145/2661821.2661825.

[26] J.-D. Fekete, "The InfoVis Toolkit," in *IEEE Symposium on Information Visualization*, 2004-10, pp. 167–174, DOI: 10.1109/INFVIS.2004.64.

[27] D. Fichter, "What Is a Mashup?" In *Library Mashups: Exploring New Ways to Deliver Library Data*, N. C. Engard, Ed., UK ed, London: Facet Publ, 2009, ISBN: 978-1-85604-703-6.

[28] A. Fineschi and A. Pozzebon, "A 3D virtual tour of the Santa Maria della Scala Museum Complex in Siena, Italy, based on the use of Oculus Rift HMD," in *2015 International Conference on 3D Imaging (IC3D)*, 2015-12, pp. 1–5, DOI: 10.1109/IC3D.2015.7391825.

[29] "Fundamentals of 3D data." (2021), [Online]. Available: https://desktop.arcgis.com/en/arcmap/latest/extensions/3d-analyst/fundamentals-of-3d-data.htm (visited on 2022-06-08).

[30] M. Gahegan, M. Takatsuka, M. Wheeler, and F. Hardisty, "Introducing GeoVISTA Studio: An integrated suite of visualization and computational methods for exploration and knowledge construction in geography," *Computers, Environment and Urban Systems*, vol. 26, no. 4, pp. 267–292, 2002-07-01, ISSN: 0198-9715, DOI: 10.1016/S0198-9715(01)00046-1.

[31] E. W. Gilbert, "Pioneer Maps of Health and Disease in England," *The Geographical Journal*, vol. 124, no. 2, pp. 172–183, 1958, ISSN: 0016-7398, DOI: 10.2307/1790244, JSTOR: 1790244.

[32] S. Gillies, "Rasterio Documentation," p. 113, 2022-05, [Online]. Available: https://github.com/rasterio/rasterio.

[33] J. Gliet, A. Krüger, O. Klemm, and J. Schöning, "Image geo-mashups: The example of an augmented reality weather camera," in *Proceedings of the Working Conference on Advanced Visual Interfaces - AVI '08*, Napoli, Italy: ACM Press, 2008, p. 287, ISBN: 978-1-60558-141-5, DOI: 10.1145/1385569.1385615.

[34] L. Hasan, A. Morris, and S. Probets, "A comparison of usability evaluation methods for evaluating e-commerce websites," *Behaviour & Information Technology*, vol. 31, no. 7, pp. 707–737, 2012-07, ISSN: 0144-929X, 1362-3001, DOI: 10.1080/0144929X.2011.596996.

[35] H. Haverkort, L. Toma, and Y. Zhuang, "Computing visibility on terrains in external memory," *ACM Journal of Experimental Algorithmics*, vol. 13, 5:1.5–5:1.23, 2009-02-23, ISSN: 1084-6654, DOI: `10.1145/1412228.1412233`.

[36] J. Hays and A. A. Efros, "IM2GPS: Estimating geographic information from a single image," in *2008 IEEE Conference on Computer Vision and Pattern Recognition*, 2008-06, pp. 1–8, DOI: `10.1109/CVPR.2008.4587784`.

[37] G. Hobona, P. James, and D. Fairbairn, "Web-based visualization of 3D geospatial data using Java3D," *IEEE Computer Graphics and Applications*, vol. 26, no. 4, pp. 28–33, 2006-07, ISSN: 1558-1756, DOI: `10.1109/MCG.2006.94`.

[38] J. Hofmann. "Gipfel." (2005), [Online]. Available: `https://flpsed.org/gipfel.html` (visited on 2022-03-08).

[39] J. Hohenstein, H. Khan, K. Canfield, S. Tung, and R. Perez Cano, "Shorter Wait Times: The Effects of Various Loading Screens on Perceived Performance," in *Proceedings of the 2016 CHI Conference Extended Abstracts on Human Factors in Computing Systems*, ser. CHI EA '16, New York, NY, USA: Association for Computing Machinery, 2016-05-07, pp. 3084–3090, ISBN: 978-1-4503-4082-3, DOI: `10.1145/2851581.2892308`.

[40] *Høydedata*, The Norwegian Mapping Authority, 2021, [Online]. Available: `https://hoydedata.no/LaserInnsyn/` (visited on 2022-05-14).

[41] "Idrett og friluftsliv, levekårsundersøkelsen," SSB. (2021-12-08), [Online]. Available: `https://www.ssb.no/kultur-og-fritid/idrett-og-friluftsliv/statistikk/idrett-og-friluftsliv-levekarsundersokelsen` (visited on 2022-06-09).

[42] Intel Corporation, *Open Source Computer Vision Library*, 2022, [Online]. Available: `https://opencv.org/` (visited on 2022-06-03).

[43] "ISO 9241-11:2018(en), Ergonomics of human-system interaction — Part 11: Usability: Definitions and concepts." (2018), [Online]. Available: `https://www.iso.org/obp/ui/#iso:std:iso:9241:-11:ed-2:v1:en` (visited on 2022-06-28).

[44] C. Jacobs, "Introduction," in *Interactive Panoramas* (X.Media.Publishing), X.Media.Publishing. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004, pp. 1–7, ISBN: 978-3-642-18665-3, DOI: `10.1007/978-3-642-18665-3_1`.

[45] C. Jacobs, "Virtual tours," in *Interactive Panoramas* (X.Media.Publishing), X.Media.Publishing. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004, pp. 159–178, ISBN: 978-3-642-18665-3, DOI: 10.1007/978-3-642-18665-3_9.

[46] Kartverket. "DTM 10 Terrengmodell (UTM32)." (2022-01-13), [Online]. Available: https://kartkatalog.geonorge.no/metadata/dtm-10-terrengmodell-utm32/fd851873-f363-46f9-9fc6-bb1b403575df (visited on 2022-06-28).

[47] B. R. Kent, "Spherical Panoramas for Astrophysical Data Visualization," *Publications of the Astronomical Society of the Pacific*, vol. 129, no. 975, pp. 1–10, 2017-04, ISSN: 1538-3873, DOI: 10.1088/1538-3873/aa5543.

[48] W. Kim, S. Xiong, and Z. Liang, "Effect of Loading Symbol of Online Video on Perception of Waiting Time," *International Journal of Human–Computer Interaction*, vol. 33, no. 12, pp. 1001–1009, 2017-12-02, ISSN: 1044-7318, 1532-7590, DOI: 10.1080/10447318.2017.1305051.

[49] M. Koeva, M. Luleva, and P. Maldjanski, "Integrating Spherical Panoramas and Maps for Visualization of Cultural Heritage Objects Using Virtual Reality Technology," *Sensors*, vol. 17, no. 4, p. 829, 4 2017-04, ISSN: 1424-8220, DOI: 10.3390/s17040829.

[50] J. Kopf, B. Neubert, B. Chen, M. Cohen, D. Cohen-Or, O. Deussen, M. Uyttendaele, and D. Lischinski, "Deep photo: Model-based photograph enhancement and viewing," *ACM Transactions on Graphics*, vol. 27, no. 5, 116:1–116:10, 2008-12-01, ISSN: 0730-0301, DOI: 10.1145/1409060.1409069.

[51] P. Kortum, C. Z. Acemyan, and F. L. Oswald, "Is It Time to Go Positive? Assessing the Positively Worded System Usability Scale (SUS)," *Human Factors*, vol. 63, pp. 987–998, 2021, DOI: 10.1177/0018720819881556.

[52] E. Koua and M. Kraak, "A usability framework for the design and evaluation of an exploratory geovisualization environment," in *Proceedings. Eighth International Conference on Information Visualisation, 2004. IV 2004.*, London, England: IEEE, 2004, pp. 153–158, ISBN: 978-0-7695-2177-0, DOI: 10.1109/IV.2004.1320138.

[53] M.-J. Kraak, "Geovisualization illustrated," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 57, no. 5-6, pp. 390–399, 2003-04, ISSN: 09242716, DOI: 10.1016/S0924-2716(02)00167-3.

[54] M.-J. Kraak and S. I. Fabrikant, "Of maps, cartography and the geography of the International Cartographic Association," *International Journal of Cartography*, vol. 3, pp. 9–31, sup1 2017-10-30, ISSN: 2372-9333, DOI: 10.1080/23729333.2017.1288535.

[55] "Kraftig ökning av vistelser i skog och mark under covid-19-pandemin," Statistiska Centralbyrån. (2022-04-21), [Online]. Available: http://www.scb.se/hitta-statistik/statistik-efter-amne/levnadsforhallanden/levnadsforhallanden/undersokningarna-av-levnadsforhallanden-ulf-silc/pong/statistiknyhet/undersokningarna-av-levnadsforhallanden-ulfsilc-2021/ (visited on 2022-06-09).

[56] Y. Kunisada and C. Premachandra, "High Precision Location Estimation in Mountainous Areas Using GPS," *Sensors*, vol. 22, no. 3, pp. 1–11, 2022-02-02, ISSN: 1424-8220, DOI: 10.3390/s22031149.

[57] A. La Salandra, D. Frajberg, and P. Fraternali, "A virtual reality application for augmented panoramic mountain images," *Virtual Reality*, vol. 24, no. 1, pp. 123–141, 2020-03-01, ISSN: 1434-9957, DOI: 10.1007/s10055-019-00385-x.

[58] Y. Lan, M. R. Desjardins, A. Hohl, and E. Delmelle, "Geovisualization of COVID-19: State of the Art and Opportunities," *Cartographica: The International Journal for Geographic Information and Geovisualization*, vol. 56, no. 1, pp. 2–13, 2021, ISSN: 1911-9925, [Online]. Available: https://muse.jhu.edu/article/790382 (visited on 2022-05-18).

[59] R. Lerner, "At the forge: Creating mashups," *Linux Journal*, vol. 2006, no. 147, p. 10, 2006-07-01, ISSN: 1075-3583.

[60] T.-Y. Lin, S. Belongie, and J. Hays, "Cross-View Image Geolocalization," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2013, pp. 891–898, [Online]. Available: https://openaccess.thecvf.com/content_cvpr_2013/html/Lin_Cross-View_Image_Geolocalization_2013_CVPR_paper.html (visited on 2022-03-10).

[61] D. Lowe, "Object recognition from local scale-invariant features," in *Proceedings of the Seventh IEEE International Conference on Computer Vision*, vol. 2, 1999-09, 1150–1157 vol.2, DOI: 10.1109/ICCV.1999.790410.

[62] A. MacEachren, M. Gahegan, W. Pike, I. Brewer, G. Cai, E. Lengerich, and F. Hardistry, "Geovisualization for knowledge construction and decision support," *IEEE Computer Graphics and Applications*, vol. 24, no. 1, pp. 13–17, 2004-01, ISSN: 1558-1756, DOI: `10.1109/MCG.2004.1255801`.

[63] A. MacEachren and D. Taylor, *Visualization in Modern Cartography* (ISSN). Elsevier Science, 1994, ISBN: 978-1-4832-8792-8, [Online]. Available: `https://books.google.no/books?id=3cP-BAAAQBAJ`.

[64] A. M. MacEachren and M.-J. Kraak, "Exploratory cartographic visualization: Advancing the agenda," *Computers & Geosciences*, Exploratory Cartograpic Visualisation, vol. 23, no. 4, pp. 335–343, 1997-05-01, ISSN: 0098-3004, DOI: `10.1016/S0098-3004(97)00018-6`.

[65] A. M. MacEachren and M.-J. Kraak, "Research Challenges in Geovisualization," *Cartography and Geographic Information Science*, vol. 28, no. 1, pp. 3–12, 2001-01-01, ISSN: 1523-0406, DOI: `10.1559/152304001782173970`.

[66] R. Maciejewski, "Geovisualization," in *Handbook of Regional Science*, M. M. Fischer and P. Nijkamp, Eds., Berlin, Heidelberg: Springer, 2021, pp. 1651–1670, ISBN: 978-3-662-60723-7, DOI: `10.1007/978-3-662-60723-7_70`.

[67] M. Maguire and N. Bevan, "User Requirements Analysis," in *Usability: Gaining a Competitive Edge*, ser. IFIP — The International Federation for Information Processing, J. Hammond, T. Gross, and J. Wesson, Eds., Boston, MA: Springer US, 2002, pp. 133–148, ISBN: 978-0-387-35610-5, DOI: `10.1007/978-0-387-35610-5_9`.

[68] S. Mallick. "Homography examples using OpenCV (Python / C ++) |." (2016-01-04), [Online]. Available: `https://learnopencv.com/homography-examples-using-opencv-python-c/` (visited on 2022-06-22).

[69] "Mapbox GL JS," Mapbox. (2022), [Online]. Available: `https://docs.mapbox.com/mapbox-gl-js/guides/` (visited on 2022-03-10).

[70] M. Meland. "Bergen ≥ 50m pf. og ≥ 100 m.o.h." (2022), [Online]. Available: `https://peakbook.org/index.php?module=index.lists.view&pbeItem=1&id=208` (visited on 2022-06-28).

[71] D. Morrison, ""Flat-Venus Society" organizes," *Eos, Transactions American Geophysical Union*, vol. 73, no. 9, pp. 99–99, 1992, ISSN: 2324-9250, DOI: `10.1029/91EO00076`.

[72] T. Munzner, *Visualization Analysis and Design* (A.K. Peters Visualization Series), 1st ed. A. K. Peters/CRC Press, 2014-12-01, ISBN: 978-0-429-08890-2, DOI: 10.1201/b17511.

[73] S. Nag, "Image Registration Techniques: A Survey," version 1, 2017, DOI: 10.48550/ARXIV.1712.07540.

[74] B. Nagy, "A New Method of Improving the Azimuth in Mountainous Terrain by Skyline Matching," *PFG – Journal of Photogrammetry, Remote Sensing and Geoinformation Science*, vol. 88, no. 2, pp. 121–131, 2020-04-01, ISSN: 2512-2819, DOI: 10.1007/s41064-020-00093-1.

[75] "Need to know - The Norwegian Trekking Association." (2022), [Online]. Available: https://english.dnt.no/need-to-know/ (visited on 2022-06-10).

[76] J. Nielsen. "Why You Only Need To Test With Five Users, Jakob Nielsen's Alertbox." (2000-03-19), [Online]. Available: https://web.archive.org/web/20120112080106/http://www.useit.com/alertbox/20000319.html (visited on 2022-06-28).

[77] M. Nöllenburg, "Geographic Visualization," in *Human-Centered Visualization Environments: GI-Dagstuhl Research Seminar, Dagstuhl Castle, Germany, March 5-8, 2006, Revised Lectures*, ser. Lecture Notes in Computer Science, A. Kerren, A. Ebert, and J. Meyer, Eds., Berlin, Heidelberg: Springer, 2007, pp. 257–294, ISBN: 978-3-540-71949-6, DOI: 10.1007/978-3-540-71949-6_6.

[78] NumPy. "Numpy.gradient — NumPy v1.23 Manual." (2022), [Online]. Available: https://numpy.org/doc/stable/reference/generated/numpy.gradient.html (visited on 2022-07-25).

[79] D. Nutsford, F. Reitsma, A. L. Pearson, and S. Kingham, "Personalising the viewshed: Visibility analysis from the human perspective," *Applied Geography*, vol. 62, pp. 1–7, 2015-08-01, ISSN: 0143-6228, DOI: 10.1016/j.apgeog.2015.04.004.

[80] "OpenCV: Basic concepts of the homography explained with code." (), [Online]. Available: https://docs.opencv.org/4.x/d9/dab/tutorial_homography.html (visited on 2022-03-15).

[81] Pallets, *Flask*, 2010, [Online]. Available: https://flask.palletsprojects.com/en/2.2.x/ (visited on 2022-02-28).

[82] T. Patterson, “Designing 3D Landscapes,” in *Multimedia Cartography*, W. Cartwright, M. P. Peterson, and G. Gartner, Eds., Berlin, Heidelberg: Springer, 1999, pp. 217–229, ISBN: 978-3-662-03784-3, DOI: 10.1007/978-3-662-03784-3_21.

[83] “PeakVisor App,” PeakVisor App. (2021), [Online]. Available: https://peakvisor.com/en/about.html (visited on 2022-06-09).

[84] T. Peters. “PEP 20 – The Zen of Python.” (2004-08-22), [Online]. Available: https://peps.python.org/pep-0020/ (visited on 2022-06-30).

[85] M. Petroff, G. D. BURE, Tortila90, J. Wienke, J. Cederberg, Didac, D. V. Oheimb, Vanessasaurus, Strarsis, W. Calderbank, P. Alexandru, PhobosK, D. Naber, A. Molnár, N. Pascual, Clashlab, J. Lehtinen, G. Cangussu, D. Morgenstern, B. Yang, N. Sherlock, Idler, J. Bowman, I. Vong, Hrumpa, D. Perry, DLar, A. Peters, A. Cortelyou, and A. Zhang, *Mpetroff/pannellum: Pannellum 2.5.6*, version 2.5.6, Zenodo, 2019-11-26, DOI: 10.5281/ZENODO.3334433.

[86] A. Petrovsky, *Eurocontrol Terrain and Obstacle Data Manual*, Eurocontrol Guidelines, Manual, 2019-11-28, pp. 1–224, [Online]. Available: https://www.eurocontrol.int/sites/default/files/2019-12/eurocontrol-terrain-obstacle-data-manual-v2-2.pdf.

[87] E. Pietroniro and D. Fichter, “Map mashups and the rise of amateur cartographers and mapmakers,” *ACMLA Bulletin*, no. 127, pp. 26–30, 2006-09, [Online]. Available: https://www.researchgate.net/publication/268180653_Map_mashups_and_the_rise_of_amateur_cartographers_and_mapmakers.

[88] Pixelmator. “Change the blend mode of a layer.” (2022), [Online]. Available: https://www.pixelmator.com/support/guide/pixelmator-pro/905/ (visited on 2022-07-31).

[89] “Planning Your Hikes & Goal Setting For Your 52 Hike Challenge,” 52 Hike Challenge. (2020-03-19), [Online]. Available: https://www.52hikechallenge.com/blogs/blog/planning-your-hikes-goal-setting-for-the-52-hike-challenge (visited on 2022-06-10).

[90] Plotly Technologies Inc., *Collaborative data science*, Montréal, Quebec, Canada: Plotly Technologies Inc., 2015, [Online]. Available: https://plot.ly (visited on 2022-03-18).

[91] L. Porzi, S. R. Buló, P. Valigi, O. Lanz, and E. Ricci, "Learning Contours for Automatic Annotations of Mountains Pictures on a Smartphone," in *Proceedings of the International Conference on Distributed Smart Cameras*, ser. ICDSC '14, New York, NY, USA: Association for Computing Machinery, 2014-11-04, pp. 1–6, ISBN: 978-1-4503-2925-5, DOI: `10.1145/2659021.2659046`.

[92] T. ur Rehman, M. N. A. Khan, and N. Riaz, "Analysis of Requirement Engineering Processes, Tools/Techniques and Methodologies," *International Journal of Information Technology and Computer Science*, vol. 5, no. 3, pp. 40–48, 2013-02-03, ISSN: 20749007, 20749015, DOI: `10.5815/ijitcs.2013.03.05`.

[93] T. M. Rhyne, A. MacEachern, and T.-M. Rhyne, "Visualizing geospatial data," in *Proceedings of the Conference on SIGGRAPH 2004 Course Notes - GRAPH '04*, Los Angeles, CA: ACM Press, 2004, 31–es, ISBN: 978-0-11-145678-1, DOI: `10.1145/1103900.1103931`.

[94] P. Ronto, "Hiking in the US has Never Been More Popular," 2021-08-06, [Online]. Available: `https://runrepeat.com/hiking-never-more-popular` (visited on 2022-04-09).

[95] R. E. Roth, "Interactive maps: What we know and what we need to know," *Journal of Spatial Information Science*, no. 6, pp. 59–115, 6 2013-06-30, ISSN: 1948-660X, DOI: `10.5311/JOSIS.2013.6.105`.

[96] E. Rouault, F. Warmerdam, K. Schwehr, A. Kiselev, H. Butler, and M. Łoskot, *GDAL*, Zenodo, 2022-01-20, DOI: `10.5281/zenodo.5884352`.

[97] "Nonlinear Projections," in *Transformations and Projections in Computer Graphics*, D. Salomon, Ed., London: Springer, 2006, pp. 145–220, ISBN: 978-1-84628-620-9, DOI: `10.1007/978-1-84628-620-9_5`.

[98] F. E. Sandnes and Y.-P. Huang, "Translating the viewing position in single equirectangular panoramic images," in *2016 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, 2016-10, pp. 389–394, DOI: `10.1109/SMC.2016.7844272`.

[99] M. Schneider and R. Klein, "Enhancing Textured Digital Elevation Models Using Photographs," in *Proceedings of the Fourth International Symposium on 3D Data Processing, Visualization and Transmission (3DPVT'08)*, 2008, pp. 1–8.

[100] M. Shih and L.-C. Chen, "An Interactive and Spherical Panorama for Multicultural City Exploration," in *Proceedings of the 2nd International Conference on Education and Multimedia Technology*, ser. ICEMT 2018, New York, NY, USA: Association for Computing Machinery, 2018-07-02, pp. 39–43, ISBN: 978-1-4503-6525-3, DOI: `10.1145/3206129.3239417`.

[101] B. Shneiderman, "The eyes have it: A task by data type taxonomy for information visualizations," *Proceedings 1996 IEEE Symposium on Visual Languages*, pp. 336–343, 1996, [Online]. Available: `http://hdl.handle.net/1903/5784` (visited on 2022-03-09).

[102] J. Snow, *On the Mode of Communication of Cholera*. John Churchill, 1854, 216 pp.

[103] U. Söderström, M. Bååth, and T. Mejtoft, "The Users' Time Perception: The effect of various animation speeds on loading screens," in *Proceedings of the 36th European Conference on Cognitive Ergonomics*, Utrecht Netherlands: ACM, 2018-09-05, pp. 1–4, ISBN: 978-1-4503-6449-2, DOI: `10.1145/3232078.3232092`.

[104] F. Soldati. "PeakFinder," PeakFinder. (2022), [Online]. Available: `https://www.peakfinder.org/about/peakfinder/` (visited on 2022-06-09).

[105] J. Spool and W. Schroeder, "Testing Web Sites: Five Users Is Nowhere Near Enough," New York, NY, USA: Association for Computing Machinery, 2001-03, pp. 285–286, ISBN: 1-58113-340-5, DOI: `10.1145/634067.634236`.

[106] "Stack Overflow Developer Survey 2021," Stack Overflow. (2021), [Online]. Available: `https://insights.stackoverflow.com/survey/2021/?utm_source=social-share&utm_medium=social&utm_campaign=dev-survey-2021` (visited on 2022-06-30).

[107] F. Stein and G. Medioni, "Map-based localization using the panoramic horizon," *IEEE Transactions on Robotics and Automation*, vol. 11, no. 6, pp. 892–896, 1995-12, ISSN: 2374-958X, DOI: `10.1109/70.478436`.

[108] R. Story, *Folium*, 2013, [Online]. Available: `https://python-visualization.github.io/folium/`.

[109] A. J. Szameitat, J. Rummel, D. P. Szameitat, and A. Sterr, "Behavioral and emotional consequences of brief delays in human–computer interaction," *International Journal of Human-Computer Studies*, vol. 67, no. 7, pp. 561–570, 2009-07-01, ISSN: 1071-5819, DOI: `10.1016/j.ijhcs.2009.02.004`.

[110] M. Takatsuka and M. Gahegan, "GeoVISTA Studio: A codeless visual programming environment for geoscientific data analysis and visualization," *Computers & Geosciences*, Shareware and Freeware in the Geosciences II. A Special Issue in Honour of John Butler, vol. 28, no. 10, pp. 1131–1144, 2002-12-01, ISSN: 0098-3004, DOI: 10.1016/S0098-3004(02)00031-6.

[111] B. H. Toby, "EXPGUI, a graphical user interface for GSAS," *Journal of Applied Crystallography*, vol. 34, no. 2, pp. 210–213, 2 2001-04-01, ISSN: 0021-8898, DOI: 10.1107/S0021889801002242.

[112] TWCarlson, *English: A schematic diagram of the terms "Azimuth" and "Altitude" as they relate to the viewing of celestial objects.* 2020-03-07, [Online]. Available: https://commons.wikimedia.org/wiki/File:Azimuth-Altitude_schematic.svg (visited on 2022-05-28).

[113] "Types of Colour Blindness," Colour Blind Awareness. (2022), [Online]. Available: https://www.colourblindawareness.org/colour-blindness/types-of-colour-blindness/ (visited on 2022-07-28).

[114] M. W. van Someren, Y. F. Barnard, and J. a. C. Sandberg, *The Think Aloud Method: A Practical Approach to Modelling Cognitive Processes*. LondenAcademic Press, 1994, ISBN: 978-0-12-714270-8, [Online]. Available: https://dare.uva.nl/search?identifier=7fef37d5-8ead-44c6-af62-0feeea18d445 (visited on 2022-06-27).

[115] J. Wang, G. J. Robinson, and K. White, "Generating Viewsheds without Using Sightlines," *Photogrammetric Engineering and Remote Sensing*, vol. 66, no. 1, pp. 87–90, 2000-01.

[116] M. Q. Wang Baldonado, A. Woodruff, and A. Kuchinsky, "Guidelines for using multiple views in information visualization," in *Proceedings of the Working Conference on Advanced Visual Interfaces*, ser. AVI '00, New York, NY, USA: Association for Computing Machinery, 2000-05-01, pp. 110–119, ISBN: 978-1-58113-252-6, DOI: 10.1145/345513.345271.

[117] C. Ware, *Information Visualization: Perception for Design*. Morgan Kaufmann, 2019-12-19, 560 pp., ISBN: 978-0-12-812876-3.

[118] "WGS84 - World Geodetic System 1984, used in GPS - EPSG:4326." (2019), [Online]. Available: http://epsg.io (visited on 2022-03-02).

[119] "Why are Topographic Maps Important for Hiking and Backpacking?" SectionHiker.com. (2021-08-29), [Online]. Available: https://sectionhiker.com/topographic-maps-hiking-backpacking/ (visited on 2022-06-18).

[120] J. Wood, J. Dykes, A. Slingsby, and K. Clarke, "Interactive Visual Exploration of a Large Spatio-temporal Dataset: Reflections on a Geovisualization Mashup.," *IEEE Transactions on Visualization and Computer Graphics*, vol. 13, no. 6, pp. 1176–1183, 2007-11, ISSN: 1941-0506, DOI: 10.1109/TVCG.2007.70570.

[121] A. Woolrych and G. Cockton, "Why and When Five Test Users aren't Enough," *Proceedings of IHM-HCI 2001 conference*, vol. 2, p. 4, 2001.

[122] T. Worbs. "Mountainpanoramas." (2010), [Online]. Available: https://www.mountainpanoramas.com/index.html (visited on 2022-06-30).

[123] R. Wróżyński, M. Sojka, and K. Pyszny, "The application of GIS and 3D graphic software to visual impact assessment of wind turbines," *Renewable Energy*, vol. 96, pp. 625–635, 2016-10-01, ISSN: 0960-1481, DOI: 10.1016/j.renene.2016.05.016.

[124] C. Wu, L. Guan, Q. Xia, G. Chen, and B. Shen, "PDERL: An accurate and fast algorithm with a novel perspective on solving the old viewshed analysis problem," *Earth Science Informatics*, vol. 14, no. 2, pp. 619–632, 2021-06-01, ISSN: 1865-0481, DOI: 10.1007/s12145-020-00545-7.

[125] "Year In Sport 2021." (2021), [Online]. Available: https://www.strava.com/yis-community-2021 (visited on 2022-06-08).

[126] N. Zang, M. B. Rosson, and V. Nasser, "Mashups: Who? what? why?" In *CHI '08 Extended Abstracts on Human Factors in Computing Systems*, Florence Italy: ACM, 2008-04-05, pp. 3171–3176, ISBN: 978-1-60558-012-8, DOI: 10.1145/1358628.1358826.

[127] H. Zhang, L. Li, W. Hu, W. Yao, and H. Zhu, "Visualization of Location-Referenced Web Textual Information Based on Map Mashups," *IEEE Access*, vol. 7, pp. 40 475–40 487, 2019, ISSN: 2169-3536, DOI: 10.1109/ACCESS.2019.2907570.

[128] B. Zitová and J. Flusser, "Image registration methods: A survey," *Image and Vision Computing*, vol. 21, no. 11, pp. 977–1000, 2003-10, ISSN: 02628856, DOI: 10.1016/S0262-8856(03)00137-9.

Appendix A

Appendix: Interview questions

A.1 Interview with hikers

As a part of forming the system requirements, we conducted interviews with hikers in the early stages of the project. The following list of predefined questions was used to guide the interviews. Additionally, some unplanned questions were asked to clarify the answers to the predefined questions.

Q_1 Do you find hiking inspiration through online images?

Q_2 What online tools do you regularly use in advance of a hike?

A.2 Interview during user study

The following listing presents the set of questions study participants were asked after interacting with PanoVis for the first time.

Q_1 Do you think the application could be used as inspiration for people seeking new hiking adventures? (Why / why not?)

Q_2 How did you feel about the panorama explorer?

Q_3 How was your experience of the interaction in the application? (What was good / what could have been done differently?)

Q_4 What could be done to improve the visual components on the application's main screen?

Q_5 Is there anything you would like to change in the application?

Q_6 Is there anything you want to add beyond the questions?

Appendix B

Appendix: The System Usability Scale

The System Usability Scale (SUS) [9] was used to evaluate the proposed application. The questionnaire and how the score is computed are included for reference.

B.1 Questionnaire

		Strongly disagree				**Strongly agree**
1.	I think that I would like to use this system frequently	1	2	3	4	5
2.	I found the system unnecessarily complex	1	2	3	4	5
3.	I thought the system was easy to use	1	2	3	4	5
4.	I think that I would need the support of a technical person to be able to use this system	1	2	3	4	5
5.	I found the various functions in this system were well integrated	1	2	3	4	5
6.	I thought there was too much inconsistency in this system	1	2	3	4	5
7.	I would imagine that most people would learn to use this system very quickly	1	2	3	4	5
8.	I found the system very cumbersome to use	1	2	3	4	5
9.	I felt very confident using the system	1	2	3	4	5
10.	I needed to learn a lot of things before I could get going with this system	1	2	3	4	5

Table B.1: The System Usability Scale used to evaluate the application's usability.

B.2 Scoring Formula

The SUS questionnaire consists of ten statements which are answered using a *Likert Scale*. The user evaluates at which level the statement is agreeable using a scale from one to five, where one represents the user strongly disagreeing with the statement and five meaning that the user strongly agrees. The given answers are collected and interpreted using the following formula. For the odd-numbered statements, the value from the Likert scale is subtracted by 1. For example, if the fifth statement has been evaluated as 4, the score will be interpreted as $4 - 1 = 3$ for this statement. For the even statements, the value is subtracted from 5, meaning a score of 3 is interpreted as $5 - 3 = 2$. Finally, the total score is computed by calculating the sum of all statements and multiplying the result by 2.5.