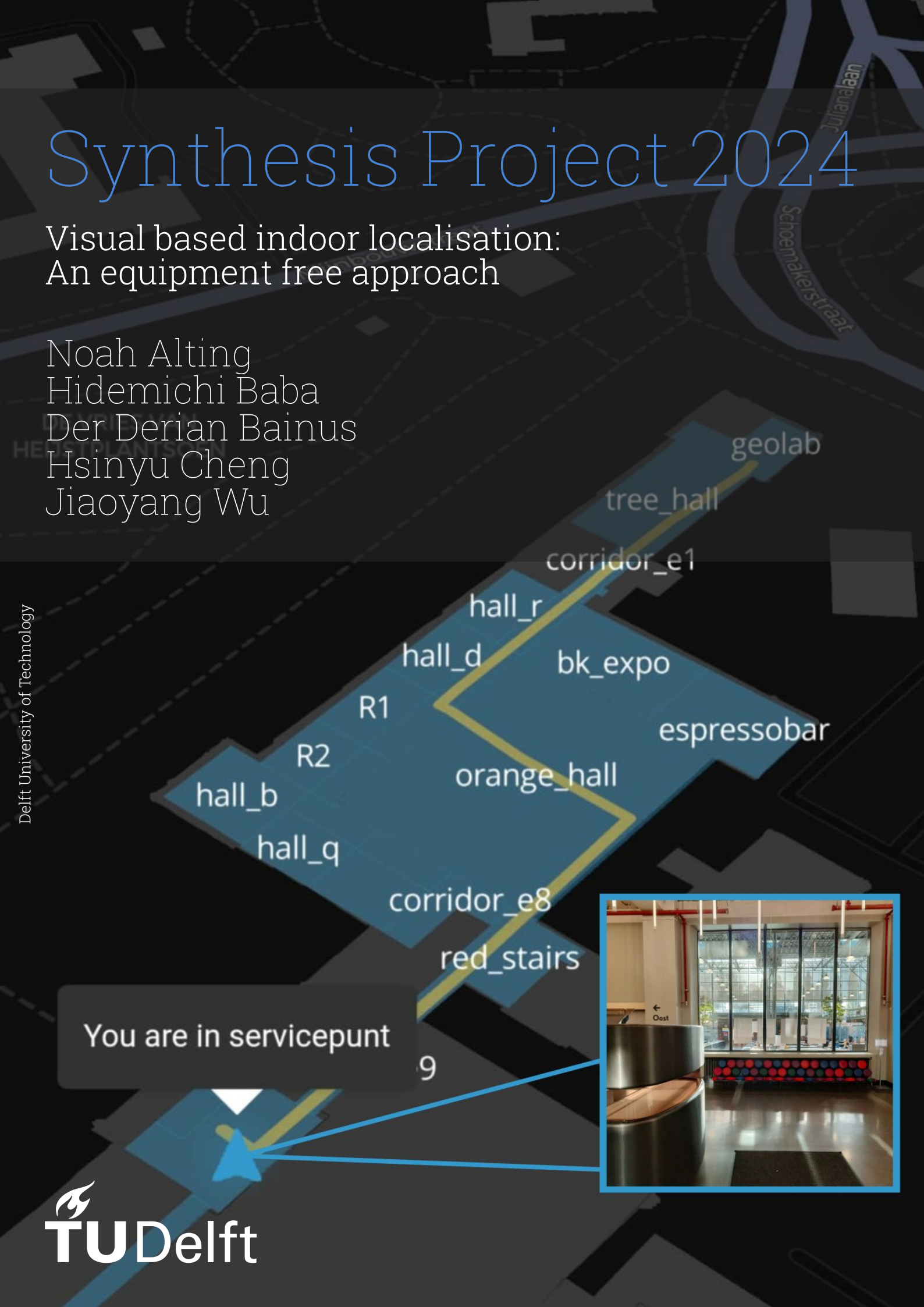
Synthesis Project 2024

Visual based indoor localisation: An equipment free approach

Noah Alting
Hidemichi Baba
Der Derian Bainus
Hsinyu Cheng
Jiaoyang Wu

Delft University of Technology



TU Delft

Synthesis Project 2024

Visual based indoor localisation: An equipment free approach

by

Noah Alting
Hidemichi Baba
Der Derian Bainus
Hsinyu Cheng
Jiaoyang Wu

Instructors:	Edward Verbree, Adibah Yunisya, Niels van der Vaart
Project Duration:	September, 2024 - October, 2024
Faculty:	Architecture and the Built Environment

TU Delft

Abstract

This project presents an indoor navigation system based on image matching, aiming to address the challenges of localization and navigation in indoor environments. The system utilizes Simultaneous Localization and Mapping (SLAM) technology to capture high-resolution images and point cloud data, combined with the VGG16 model from Convolutional Neural Networks (CNN) for image processing, feature extraction, and matching.

In our research, we conducted experiments at the Faculty of Architecture and the Built Environment of Delft University of Technology, using a SLAM scanner to obtain 360-degree panoramic images and point cloud data of the indoor environment. Through cube mapping projection, we converted the panoramic images into six planar views, selecting the front, right, and left views as positioning references. Additionally, we reconstructed the indoor environment structure and designed node networks for positioning and navigation.

The technical architecture of this system comprises three main components: VGG16-based image feature extraction, cosine similarity-based image matching, and DBSCAN algorithm for location clustering. Through this method, the system can achieve real-time localization results after image capture and provide users with optimal paths using the A* navigation algorithm.

Experimental results show that when using single image matching, the system's room localization accuracy reaches 74.65%. When employing multiple image matching and DBSCAN clustering methods, the accuracy significantly improves. In our final evaluation involving 116 positions, the system successfully matched 111 of these positions to their correct rooms, achieving a localization accuracy of 95.69%.

This research not only provides an innovative solution for indoor positioning and navigation but also points the way for future research, including support for multi-floor navigation, enhancing CNN model performance, and automating building processing. This technology has the potential for widespread application in complex indoor environments such as large buildings, conference centers, and university campuses, offering users accurate, real-time positioning and navigation services.

Keywords: Indoor Navigation, Image Matching, SLAM, VGG16, Real-time Positioning.

Contents

1

Introduction

1.1. Research Purpose

Indoor positioning technology has advanced rapidly to address the limitations of Global Navigation Satellite Systems (GNSS) in indoor environments, where signal attenuation and structural obstructions render GNSS ineffective (Khan et al. 2022). With the rise of smart devices, the demand for efficient Indoor Positioning Systems (IPS) has grown, leading to methods such as WiFi-based fingerprinting and Bluetooth Low Energy (BLE) systems (Subedi and Pyun 2020). However, each of these systems presents unique trade-offs, particularly with respect to accuracy, accessibility, and energy use.

This project aims to explore a vision-based alternative for indoor localization that bypasses the need for WiFi or BLE. Leveraging image data acquired through Simultaneous Localization and Mapping (SLAM) technology, we propose a camera-based solution that minimizes additional infrastructure while offering accurate, real-time positioning. This approach mitigates limitations of RF-based methods, such as fluctuating signal strength and power demands (Mainetti, Patrono, and Sergi 2014). To address these gaps, we come up with a research question as our guideline of this project which is:

"How effective is a vision-based indoor navigation system in achieving accurate, room-level localization for users within complex building environments?"

To implement this, we focus on a web-based application deployed at the Faculty of Architecture and the Built Environment at TU Delft. Using the VGG16 model from Convolutional Neural Networks (CNN) for image processing and matching, the system provides real-time location feedback within a browser-based app, requiring no installation and utilizing server-side resources for enhanced accessibility. The conceptual foundation of this system is rooted in the Concepts of Placement framework proposed by Sithole and Zlatanova, which emphasizes understanding placement in terms of Position, Location, Place, and Area. This framework seeks to capture the dynamic nature of indoor spaces, where human interaction with the environment involves more than simply moving from one pinpoint location to another. By integrating these concepts, this project aims to provide a more intuitive, context-aware localization experience, reflecting how users perceive and navigate through complex indoor environments (Sithole and Zlatanova 2016).

1.2. Stakeholders

In this section, we will outline the collaborators involved in developing this project, explain how the client will be engaged throughout the project, and describe how users will benefit from the product.

1.2.1. Team

The Team is made up of dedicated members who handle all the tasks and responsibilities of the project. Each person plays an important role in planning, carrying out, and checking the project's progress. They bring different skills and expertise to help achieve the project goals effectively and efficiently. Team members manage resources, stick to deadlines, and handle any challenges that come up along

the way. By working closely together, using each other's strengths, and applying what they learned in the first year of the TU Delft Master's in Geomatics program, the team aims to deliver results that are well-researched and based on reliable information. The individual information and roles of all team members are shown in the Appendix C.

1.2.2. Supervisors

The Supervisors provide important guidance and support throughout the project. They help ensure that the team's work meets high standards of quality and accuracy. Each supervisor offers valuable expertise, helping the team stay focused, solve challenges, and achieve project goals. Supervisors give feedback, monitor progress, and encourage the team to follow best practices at each stage. By sharing their knowledge, especially in areas related to the TU Delft Master's in Geomatics program, the supervisors help the team create a final result that is well-researched and reliable. Their support is key to turning the team's efforts into a successful outcome. The information of team supervisors is shown in the Appendix D.

1.2.3. Client

In this project, we identify Adibah Nurul Yunisya as our primary client, with an emphasis on foundational research to support her studies. Our approach is intended to serve as a versatile method that could be applied more broadly in the future. We define our client as follows:

- **Adibah Nurul Yunisya**: A PhD student at TU Delft, specializing in indoor navigation research, with a focus on using landmarks to enhance spatial awareness for end-users.
- **Future Clients**: Building administrators seeking a user-friendly indoor localization and navigation solution for visitors without extensive infrastructure, such as Bluetooth sensors.

1.2.4. Client's Users

Our end users are envisioned as individuals visiting a building with numerous rooms and corridors, often without prior familiarity. These users may have arrived using an outdoor navigation app, such as Google Maps, but lack resources for navigating the indoor environment. As a result, they may be unaware of their specific location within the building.

- **Spatial Awareness**: Users initially lack orientation within the building and may attempt to establish their location by consulting an indoor map, typically located at the entrance.
- **Navigation**: With no clear path to their destination, users are expected to rely on indoor maps or signage to identify their current position and find their way to their intended location.

1.3. Deliverables

This project includes several key outcomes: a web application, source code on GitHub, documentation, a dataset, and this report.

- **Web Application**: The primary deliverable is a web application that enables users to achieve room-level localization and navigation within large indoor spaces. By using image-matching technology, the application allows users to identify their current room by taking a few photos, making it possible to determine their location even without GPS. The app is accessible to anyone with the URL.
- **Source Code**: All code developed for this project is published as open-source software (OSS) and is available in the GitHub repository: GEO1101 Synthesis Project.
- **Source Code Documentation**: The code includes a README file that details each step required for developers to set up and run the application.
- **Dataset**: A collection of data acquired through SLAM and data generated manually by the team.
- **This Report**: Comprehensive documentation of the project's objectives, methodology, results, and future recommendations.

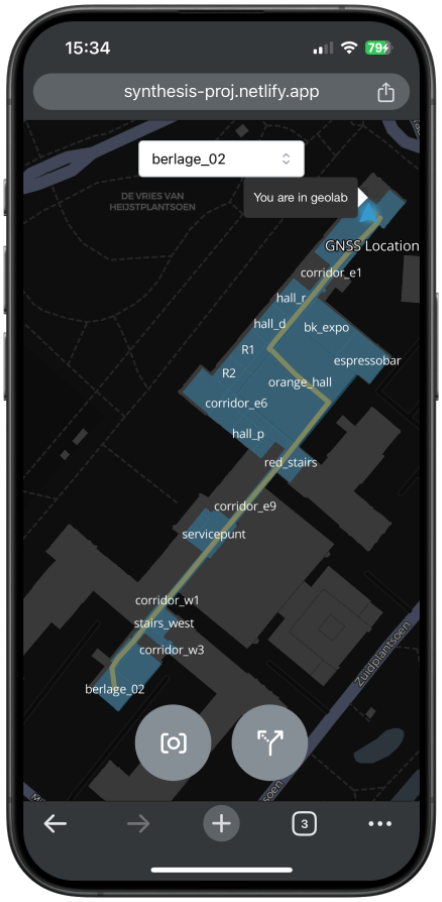


Figure 1.1: User interface of the Web app

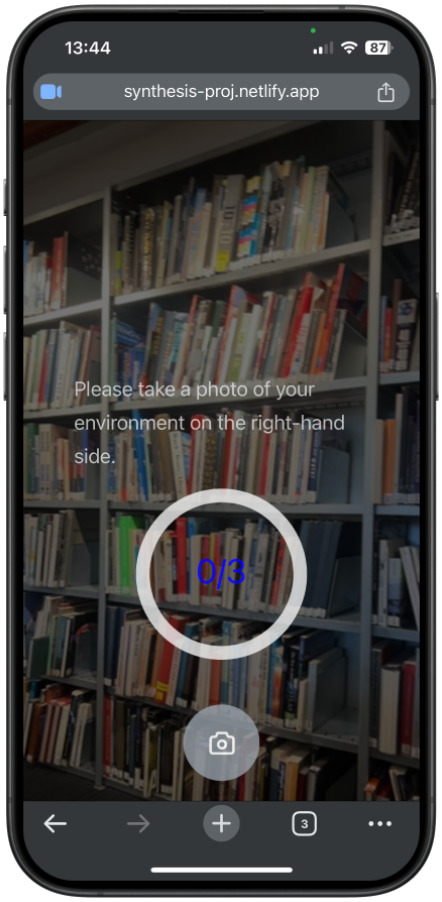


Figure 1.2: User interface of the Web app

2

Background

2.1. Problem Statement

This section outlines the key issues in implementing indoor positioning solutions, including the difficulties and challenges of using GNSS for indoor positioning; the analysis of the more mature indoor positioning technologies, such as Bluetooth and WiFi fingerprinting, and some related work with similar methods to this project.

2.1.1. GNSS Signal in Indoor Environment

It is widely known that the positioning accuracy of GNSS receivers in ”classical” outdoor environments is generally higher than in indoor environments (Drira 2006). The searching in two domains, frequency and time (delay), brings estimates of the the Doppler frequency and code delay (Puricer and Kovar 2007). In case of clear visibility there are several strong satellite signals present and the tasks of acquisition, tracking, and position computation are relatively easy.

In challenging environments such as urban canyons or indoor spaces, user receiver location becomes difficult. Satellite signals are obstructed by foliage, walls, and other structures (Abo-Zeed et al. 2019). The signals available to the user receiver in these environments mostly consist of heavily attenuated direct signals and reflected (or scattered) signals that typically follow the path of least resistance. That is why the two main phenomena that the indoor receiver must take into account are signal attenuation and the multipath effect.

Signal attenuation is one of the main challenges faced by GNSS in indoor positioning (Sakpere, Adeyeye-Oshin, and Mlitwa 2017). The GNSS signal strength in indoor environments typically ranges from -160 dBW to -190 dBW, significantly lower than in outdoor environments. Different building materials also have varying degrees of signal attenuation. For example, wooden structures may cause about 10 dB of attenuation, while brick walls can cause 20-25 dB of attenuation. This severe signal attenuation requires receivers to process extremely weak signals, posing enormous challenges to signal acquisition and tracking, greatly increasing the difficulty of indoor positioning.

Multipath effect is another major issue faced by indoor GNSS positioning (Wielandt and De Strycker 2017), with its impact being more severe in indoor environments than outdoors. Indoors, GNSS signals reach the receiver after multiple reflections, scattering, and diffractions. This complex signal propagation environment may result in the direct signal being completely obscured, or the reflected signals becoming stronger than the direct signal. This situation makes traditional multipath mitigation techniques difficult to apply effectively, as they usually assume that the direct signal is dominant. Therefore, in indoor environments, more complex channel models and signal processing techniques need to be considered to address this challenge.

2.1.2. Available Indoor Positioning Technologies

Indoor positioning and navigation systems provide precise tracking and navigation services in enclosed spaces such as shopping malls, hospitals, airports and university campuses. Because Global Navigation Satellite System (GNSS) technologies like GPS have trouble penetrating signals from buildings or other obstacles, these systems often don't work properly in indoor environments (Pasricha 2021). Therefore, positioning technology designed specifically for indoors has become particularly important.

Among these systems, Wi-Fi-based localization is popular for its cost-effectiveness and compatibility with existing Wireless Local Area Networks (WLANs), utilizing methods like fingerprinting and Received Signal Strength Indicator (RSSI) to estimate device location. However, challenges such as power consumption and signal attenuation from obstacles significantly affect positioning accuracy and performance (Mainetti, Patrono, and Sergi 2014).

Similarly, Bluetooth indoor positioning leverages Bluetooth technology widely used in devices such as smartphones and laptops to achieve low-power positioning without the need for additional hardware. Still, it faces issues such as increased latency during device discovery, short communication range so more hardware installations (such as beacons) are required to properly cover the space, and difficulties in dynamic indoor environments, making it less suitable for real-time Application (Mainetti, Patrono, and Sergi 2014).

Compared with wireless technology-based approaches, vision-based methods require only common and economical cameras as perception sensors, with minimal environmental instrumentation (L. Xu et al. 2019). A mobile camera-based solution was proposed in (Fanga et al. 2016), where a point cloud of a site was created using Structure from Motion (SfM) with drone-collected images, allowing feature matching to locate objects within the point cloud. However, SfM is time-consuming, and recovering camera poses from a complete image set makes this solution unsuitable for real-time applications, limiting continuous tracking capabilities (S. Jiang, C. Jiang, and W. Jiang 2020).

Simultaneous Localization and Mapping (SLAM) is a process where a device builds a map of an unfamiliar environment while tracking its own location within that map. Using sensors like cameras or LiDAR, SLAM identifies environmental features and updates both the map and the device's position as it moves. This makes SLAM ideal for navigation in unknown spaces without relying on pre-existing maps (Cadena et al. 2016).

We leverage a SLAM-based scanner to generate high-density point clouds in this project, offering several distinct advantages over traditional mapping approaches. This method provides real-time mapping with high accuracy, particularly in dense and feature-rich environments, while eliminating the need for pre-existing floor plans. The SLAM scanner continuously updates and refines the map as new data is collected, making it highly adaptable to dynamic environments—ideal for indoor positioning and navigation.

2.1.3. Related works

As a traditional feature matching method (Liu et al. 2018), Dynamic Bag of Words (DboW) has been applied wildly in image matching and SLAM (Simultaneous Localization and Mapping) systems, where real time positioning and mapping is crucial. It uses vocabulary trees or similar structure to classify images in the key frame of videos (Zhang, Li, and Yang 2010), representing images as vectors of visual words, allowing for efficient comparison and retrieval.

Recent advancements in indoor localization techniques have explored novel approaches. Triantafyllou's thesis on *Isovist Fingerprinting* presents an innovative method of indoor localization by capturing visibility information from specific points and analyzing the spatial layout using isovist parameters such as area, perimeter, and compactness. His research suggests that visibility-based fingerprinting, when combined with 2D representations of LiDAR-captured point clouds, can serve as a viable alternative for localization in settings where traditional methods struggle due to hardware constraints or environmental variability (Georgios Triantafyllou and Azarakhsh Rafiee 2024).

In recent years, convolutional neural networks (CNN) have been extensively used for a wide range of visual perception tasks (Rawat and Wang 2017), such as object detection/classification, action/activity recognition, etc. Behind the remarkable success of DCNN on image/video analytics are its unique

capabilities of extracting underlying nonlinear structures of image data as well as discerning the categories of semantic data contents by jointly optimizing parameters of multiple layers together (Zhu, Fang, and Ghamisi 2018). Among various CNN architectures, VGG16 has gained significant attention due to its simplicity and effectiveness. Developed by the Visual Geometry Group at Oxford, VGG16 is characterized by its depth and uniform architecture (Zhou 2024), consisting of 16 weight layers including 13 convolutional layers and 3 fully connected layers. The model's strength lies in its use of small 3x3 convolutional filters stacked in increasing depth, which allows it to learn complex features efficiently while maintaining a relatively small number of parameters (Bello 2023).

In this paper, we present an image-based approach for indoor localization to address the challenges of point cloud reconstruction. Unlike previous study (Dardavesis, Verbree, and A. Rafiee 2023) that primarily focused on ceiling images, our research utilizes images captured from the front and sides of the environment. This approach provides a more comprehensive view of the surroundings and potentially captures more diverse features for localization. Using SLAM technology, we capture images from a LiDAR scanner, which provides accurate environmental scanning even in complex or dynamic indoor settings. These images are then processed using the VGG16 convolutional neural network for feature classification.

The VGG16 model's deep architecture enables efficient visual feature extraction and matching, which is beneficial for our application due to its ability to capture hierarchical representations of visual data. This approach is robust in environments with fewer distinct 3D features but sufficient visual detail, enhancing both the accuracy and efficiency of localization. By focusing on front and side views rather than ceiling images, we aim to capture a wider range of environmental features that may be more distinctive and informative for localization purposes.

By leveraging the VGG16 model's powerful feature extraction capabilities, our method can effectively identify and classify key visual elements in the indoor environment from multiple perspectives. The model's deep layers allow for the recognition of complex patterns and structures, which is crucial for accurate localization in varied indoor settings. Furthermore, the VGG16's uniform architecture facilitates easy integration into our existing SLAM framework, allowing for seamless processing of the high-resolution LiDAR images and efficient matching of visual features across different frames. This multi-view approach, combining front and side images, potentially offers a more robust and versatile solution for indoor localization compared to methods that rely solely on ceiling imagery.

2.2. Motivation

This section outlines the team's motivation and purpose for implementing this project, and proposes the MoSCoW Framework applicable to this project.

2.2.1. Motivation for Our Works

Our project is driven by several key motivations:

- We aim to make indoor navigation more user-friendly and accessible, especially in large, complex buildings where traditional navigation methods fall short.
- By utilizing smartphone cameras, we're tapping into a device that most people already carry, eliminating the need for additional hardware or devices.
- By developing a solution that's both technologically advanced and user-friendly, we're working towards making complicated indoor navigation available for public use.

Our approach combines image matching with graph-based pathfinding to create a seamless navigation experience. Users simply capture images of their surroundings, which are then processed on our servers to determine their location and calculate the optimal route to their destination. This innovative method puts powerful navigation capabilities directly into the hands of users, transforming how people interact with and move through complex indoor environments.

2.2.2. Motivation for API and UI-Driven Approach

We opted to develop a web application to explore the feasibility of browser-based indoor localization, which offers the advantage of accessibility without requiring users to install a dedicated app. Although

accuracy improvements remain essential, our web app serves as a proof-of-concept, demonstrating the potential for delivering an effective user experience using the Geolocation API and camera as core tools.

Additionally, the Synthesis project's goal to "apply and expand knowledge acquired during core Geomatics courses" influenced this choice. By utilizing skills gained in "Geoweb Technology," we aimed to solve a real-world problem through a web-based interface, addressing both technical challenges and user accessibility within a single framework.

2.2.3. MoSCoW Framework

A widely used framework called the MoSCoW Method is used to define the boundaries of development work clearly. Software development expert Dai Clegg introduced the method to separate project requirements into four groups based on their priority (Kuhn 2009): **Must-have**, **Should-have**, **Could-have**, and **Won't-have**. The meaning of each category is explained below, followed by Table 2.1 indicating what product needs fall into each category according to our team.

- **Must have:** Non-negotiable product needs that are mandatory for the team.
- **Should have:** Important initiatives that are not vital but add significant value.
- **Could have:** Nice-to-have initiatives that will have a small impact if left out.
- **Won't have:** Initiatives that are not a priority for this specific time frame.

Priority	**Requirement**	**Achieved**
Must have	Indoor localization using image matching technology	Yes
Should have	• Create a Web App to visualize the result • Show the way to the destination	Yes
Could have	Real-time updates on position on the route to the destination	No
Won't have	Directions with visual cues	No

Table 2.1: MoSCoW Framework

3

Methodology

3.1. Methodology Overview

Our research aims to enhance indoor localization accuracy by leveraging user-captured images. To achieve this goal, we will develop a comprehensive methodology that combines advanced computer vision techniques with graph-based navigation. Our approach consists of five key stages:

1. Data Acquisition and Pre-processing: We collect and prepare a diverse set of visual data from various indoor environments.
2. Image Matching with Ground Truth: We utilize feature-based matching algorithms to compare user-captured images with a pre-established ground truth dataset, enabling accurate position estimation.
3. Localization of User Position: By combining the results of image matching with SLAM data, we determine the user's precise location within the indoor environment.
4. Navigation using Graph Data Structure: We represent the indoor space as a graph, allowing for efficient pathfinding and navigation from the user's current position to their desired destination.

3.2. Data Acquisition

Our research on localization and matching algorithms relies on accurately collected ground truth data, composed of two primary components: image data and point cloud data. Images, captured from accessible and walkable areas, are tagged with specific coordinates, while point cloud data maps room walls with the same coordinate system to support seamless integration.

The project area includes targeted sections of TU Delft's Bouwkunde Building: the east wing, encompassing the Geolab, lecture halls B, Q, P, D, R, the Orange Hall, and the espresso bar; and the west wing, covering rooms Berlage 01 and Berlage 02. With image coordinates as the entry points for localization and room bounding boxes identifying room locations, it is essential for both image and point cloud data to align within a unified coordinate system.

To ensure this alignment, we selected a SLAM scanner, which provides synchronized image positions and point cloud data. The SLAM scanner's output includes 360-degree panoramic images, image coordinates in a CSV file, and detailed point cloud data. These outputs undergo further processing steps—georeferencing, cube mapping of panoramic images, and room polygon creation—to transform raw data into usable ground truth for our methodology.

3.2.1. Georeferencing

Georeferencing aligns SLAM data with the building's coordinate system, ensuring consistency with the user interface and accuracy in mapping. Specific objects outside the building were identified as ground control objects, providing fixed reference points that connect indoor data to a geographic coordinate system. These control objects were then integrated into the georeferencing process, allowing the SLAM

scanner's indoor point cloud data and image coordinates to align with external data sources, such as the Open Street Map basemap. This integration creates a consistent spatial framework that accurately reflects the physical layout of the Bouwkunde Building.

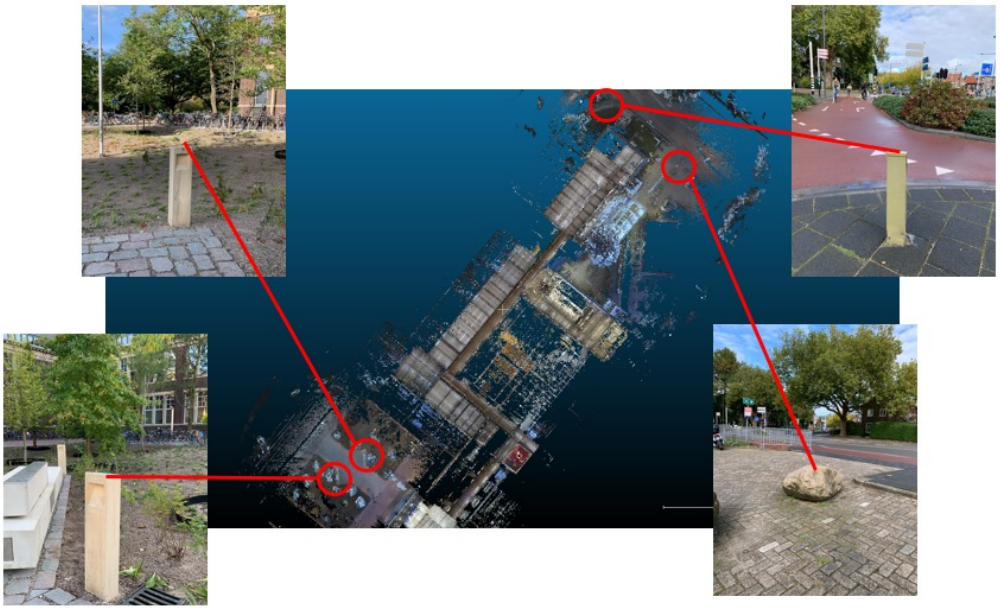


Figure 3.1: Georeferencing With 4 Ground Control Objects

3.2.2. Cube Mapping of the 360-degree Panoramic Images

Cube mapping transforms spherical panoramas into six flat views—front, back, left, right, top, and bottom—reducing distortion for easier navigation. For this project, only the front, right, and left views are retained, focusing on essential perspectives for navigation. The cube mapping process is automated using Python code, which calculates the 3D direction vectors for each cubemap face and maps these directions to the equirectangular image coordinates to sample the correct portions of the panorama. Each face is then saved as a separate image, creating a consistent dataset compatible with 3D environments.

Figure 3.2: Illustration of 360-degree Panorama Image

Figure 3.3: Illustration of Cubemap Images Projection

Each image in the dataset is tagged with coordinates that match the LiDAR data in a unified coordinate system, ensuring seamless integration of images and point clouds. This alignment simplifies georeferencing, enabling images and LiDAR scans to accurately align within the same spatial framework. It supports a range of tasks, from creating floor plans based on different point cloud datasets to merging image trajectories from various data sources, thereby maintaining consistency across datasets.

Table 3.1: Illustration of Image Code With Tts Coordinate Information

Image	Time	X	Y	Z	Heading	Roll	Pitch
p000000.jpg	1726507930	-0.1648	0.0039	0.291	-0.0193	-0.5862	1.4294
p000001.jpg	1726507931	-0.1585	0	0.2835	0.0334	-0.4847	1.4516
p000002.jpg	1726507932	-0.1576	0.0046	0.2826	0.0144	-0.5046	1.3378
p000003.jpg	1726507933	-0.1571	0.0059	0.2881	-0.1163	-0.5003	1.5259
p000004.jpg	1726507934	-0.1425	0.004	0.2861	-0.2175	-0.4012	1.4497
p000005.jpg	1726507935	-0.1568	-0.0021	0.293	-0.3146	-0.4634	1.4721
p000006.jpg	1726507936	-0.1535	-0.0128	0.2882	-0.2257	-0.4583	1.4905
p000007.jpg	1726507937	-0.1577	-0.0105	0.2914	-0.2348	-0.5008	1.4071

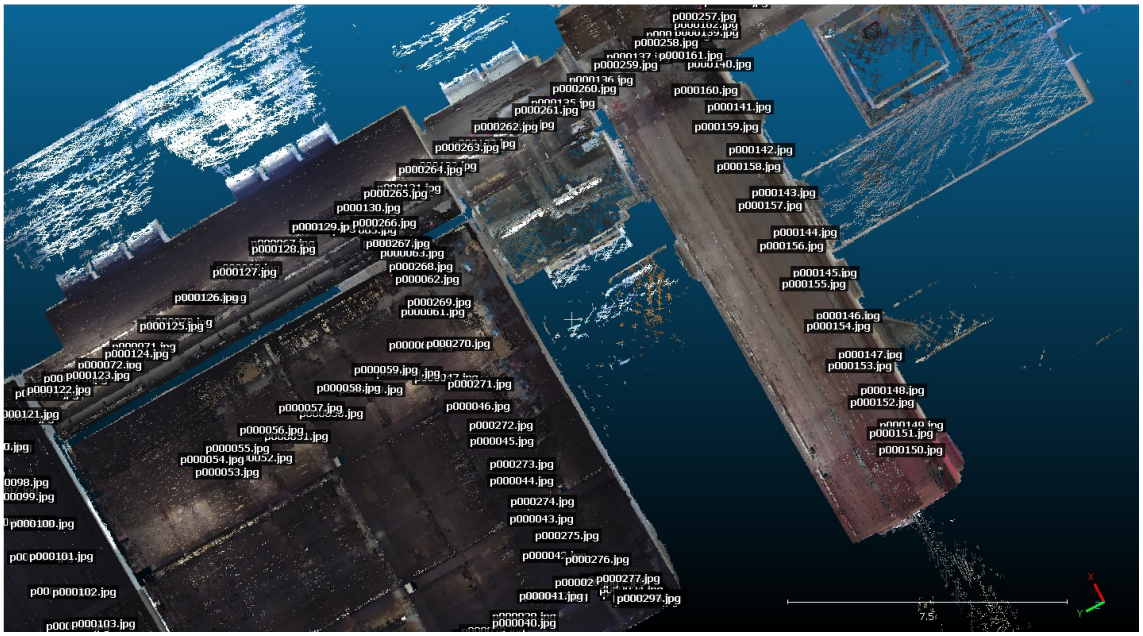


Figure 3.4: Illustration of Integrated Images Coordinate and Point Cloud

3.2.3. Creating Room Polygons for the Floorplan

Creating floorplan polygons involves transforming 3D georeferenced point cloud data into 2D representations. Starting with point cloud data aligned to real-world coordinates, we ensure seamless integration with other spatial data. The data is subsampled to reduce the number of points without losing essential details, making it easier to handle. Noise filtering is then applied to remove background noise, isolating only permanent structural elements by filtering out temporary objects such as furniture and people. This process ensures that the final floorplan accurately reflects the actual layout of the space.

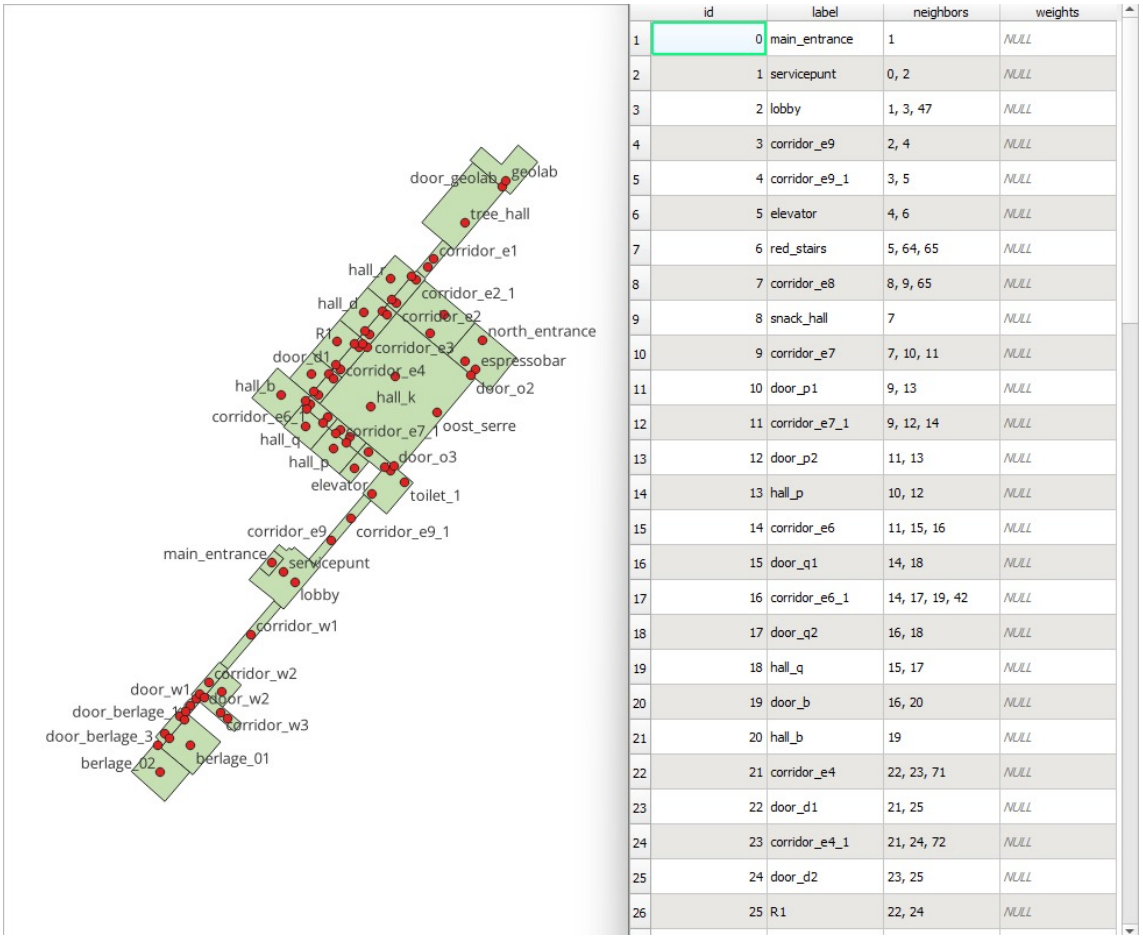
After cleaning, the data is flattened to 2D by removing the Z attribute, leaving only X and Y coordinates for the floorplan. These 2D points are then used to digitize rooms as polygons, creating a clear layout. Future datasets follow the same steps—from georeferencing to CSV conversion—to maintain consistency and ease integration with existing data. Each polygon is labeled with a unique identifier, allowing nodes to be linked with specific floorplan areas for navigation, enabling precise location referencing and routing.

To make the floorplan compatible with the API, it is converted to GeoJSON format in the EPSG 4326 coordinate system, allowing seamless integration for indoor navigation and localization.

3.2.4. Creating Nodes for Navigation

The process of creating node data for the closest path algorithm involves setting up a network of navigation nodes across the floorplan to enable efficient indoor navigation. Using the floorplan polygon layer as a reference, nodes are accurately positioned within each room or area. In QGIS, a new shapefile with multipoint geometry is created to hold these nodes, each serving as a key navigation point. Each polygon in the floorplan has at least one node to ensure complete navigational coverage.

Each node is assigned a unique identifier (ID) to allow the navigation system to reference specific locations and organize the network efficiently. Nodes are also named to correspond with their polygon (room or area), establishing a clear link essential for navigation tasks. Unique names prevent confusion in the navigation data. Neighboring nodes are connected by specifying direct connections between node IDs, forming a network that the path algorithm can use to determine efficient routes. The created nodes saved in GeoJSON format and saved in EPSG:4326 coordinate system.



	id	label	neighbors	weights
1	0	main_entrance	1	NULL
2	1	servicepunt	0, 2	NULL
3	2	lobby	1, 3, 47	NULL
4	3	corridor_e9	2, 4	NULL
5	4	corridor_e9_1	3, 5	NULL
6	5	elevator	4, 6	NULL
7	6	red_stairs	5, 64, 65	NULL
8	7	corridor_e8	8, 9, 65	NULL
9	8	snack_hall	7	NULL
10	9	corridor_e7	7, 10, 11	NULL
11	10	door_p1	9, 13	NULL
12	11	corridor_e7_1	9, 12, 14	NULL
13	12	door_p2	11, 13	NULL
14	13	hall_p	10, 12	NULL
15	14	corridor_e6	11, 15, 16	NULL
16	15	door_q1	14, 18	NULL
17	16	corridor_e6_1	14, 17, 19, 42	NULL
18	17	door_q2	16, 18	NULL
19	18	hall_q	15, 17	NULL
20	19	door_b	16, 20	NULL
21	20	hall_b	19	NULL
22	21	corridor_e4	22, 23, 71	NULL
23	22	door_d1	21, 25	NULL
24	23	corridor_e4_1	21, 24, 72	NULL
25	24	door_d2	23, 25	NULL
26	25	R1	22, 24	NULL

Figure 3.5: Created Nodes

3.2.5. List of Data

The structured data essential for the indoor navigation system includes:

1. **Cubemap Images**: Flat views (front, left, right) from panoramic images, essential for localization.
2. **Image Coordinates**: Georeferenced for accurate positioning within the layout.
3. **Floorplan Polygons**: Created from point cloud data, saved in GeoJSON format for compatibility.
4. **Node Coordinates**: Positioned within floorplan polygons, forming a navigable network for route calculations.

This dataset supports indoor localization and navigation by linking visual data with spatial information accurately.

3.3. Image matching and localisation

3.3.1. Overview of Technical Architecture

The objective of this architecture is to accurately extract and match features from user-provided input images, utilizing spatial clustering to determine the potential location of the user's image. The architecture consists of three main components, each contributing to the overall process:

1. **Image Feature Extraction (CNN Model)**:
    - A pre-trained Convolutional Neural Network (CNN), specifically `VGG16`, is used to extract high-dimensional feature vectors from both the user's input images and a reference dataset of ground truth images.

- The feature vectors for the reference images are extracted and stored in advance, enabling efficient matching when a query image is provided. These stored features serve as the basis for similarity matching and localization.

2. **Image Matching (Cosine Similarity)**:
   - After obtaining the feature vector for the query image, we calculate similarity scores by comparing it with the precomputed feature vectors from the reference images.
   - Using cosine similarity, the system selects the top N reference images with the highest similarity scores as “best matches” and retrieves their associated geographic coordinates for clustering.
3. **Location Clustering (DBSCAN)**:
   - The geographic coordinates of the top N matched images are clustered using DBSCAN (Density-Based Spatial Clustering of Applications with Noise) to identify dense regions and filter outliers.
   - The largest cluster’s centroid is then used as the predicted location for the user’s image.

Process Summary: The technical architecture workflow is as follows:

- **Data Preprocessing**: Each input image undergoes resizing and normalization to meet the input requirements of the VGG16 model, ensuring consistency for feature extraction.
- **Feature Extraction**: The preprocessed query image and each reference image in the dataset are processed through the VGG16 model to obtain their respective feature vectors.
- **Image Matching**: The feature vector of the query image is compared to reference image feature vectors using cosine similarity, allowing us to select the top N matches.
- **Coordinate Clustering**: The geographic coordinates of the best-matched reference images are clustered using DBSCAN, and the centroid of the largest cluster is identified as the predicted location.

3.3.2. CNN for Image Matching

VGG16 is a deep convolutional neural network consisting of 13 layers of 3×3 small convolutional filters and 3 fully connected layers, forming a 16-layer structure. This model performed exceptionally well in the 2014 ImageNet Challenge, achieving a Top-5 error rate of 7.3% in the classification task, ranking second; in the localization task, it achieved first place with a Top-5 error rate of 25.3% (Simonyan and A. Zisserman 2014).

Figure 3.6 illustrates the detailed structure of the VGG16 model, highlighting its 16 layers of convolutional and pooling operations, followed by fully connected layers. This diagram provides a overview of the model’s architecture and the hierarchical feature extraction it performs.

Compared to other architectures, VGG16 strikes a good balance between accuracy and computational cost. With approximately 138 million parameters, this model achieves a classification accuracy of 92.7% on the ImageNet test set. For comparison, VGG19 adds 3 convolutional layers, increasing the parameter count to around 144 million and slightly improving classification accuracy. On the other hand, ResNet50 has about 23.5 million parameters and uses residual connections to reduce training errors in deeper networks, achieving a test accuracy of 97.33% with higher computational efficiency (Mascarenhas and Agarwal 2021).

The comparison shows that VGG16 provides high accuracy while maintaining a relatively moderate number of parameters, making it suitable for resource-constrained applications where quick and accurate localization is required. ResNet50, however, is more advantageous in scenarios where extremely high accuracy is needed. Therefore, in our system, we use the pre-trained VGG16 model, a convolutional neural network (CNN), to extract deep features from images for image matching.

1. **Choice of the VGG16 Model**:

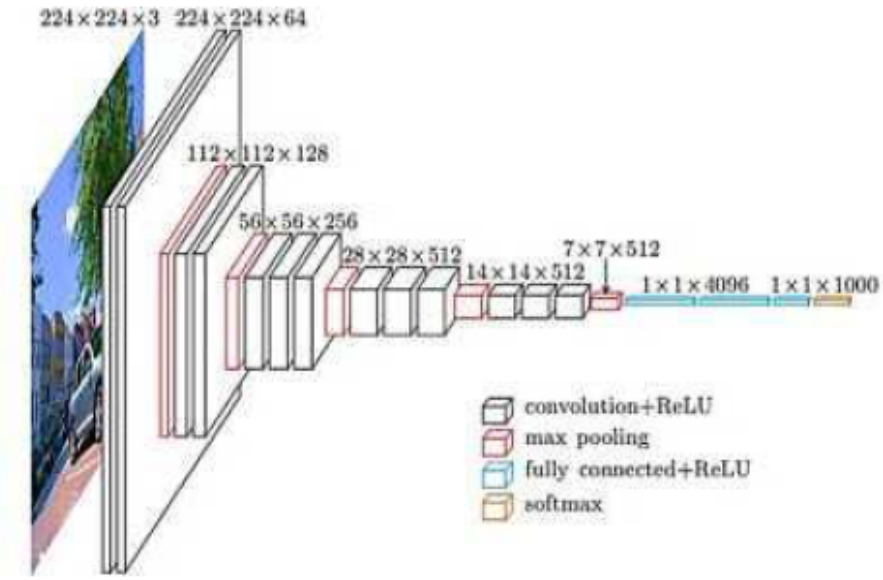


Figure 3.6: VGG16 architecture showing the sequence of convolutional, pooling, and fully connected layers Mascarenhas and Agarwal 2021.

- VGG16 is a well-known CNN model with a deep structure (16 layers), known for its strong performance in image classification tasks. This model extracts hierarchical features through a series of convolutional and pooling layers.
- In this system, we load a pre-trained VGG16 model and remove the classification layers, retaining only the feature extraction layers. This allows the model to output feature vectors that can be used for image matching rather than classification.
- The reference dataset is processed in advance, with each image passed through the VGG16 model to extract feature vectors that are stored for efficient retrieval during similarity matching.

2. **Image Preprocessing**:
   - To meet the input requirements of VGG16, both query images and reference images undergo preprocessing steps, including resizing (to 224x224), conversion to a tensor, and normalization (using the mean and standard deviation values expected by VGG16's pre-trained weights).
   - These preprocessing steps ensure a consistent format for images fed into the model, and normalization enhances the model's ability to recognize features across different images.
3. **Feature Vector Extraction**:
   - After preprocessing, the query image and each image in the reference dataset are processed through the VGG16 model, generating high-dimensional feature vectors that capture multi-level information about each image, making them suitable for similarity matching.
   - The model is set to inference mode during feature extraction to prevent any modifications to model weights, which enhances the stability of the matching process.
4. **Matching Using Feature Vectors**:
   - The feature vector of the query image is compared to the stored feature vectors of reference images using cosine similarity.
   - Higher cosine similarity indicates greater content similarity between images. The system selects the top N reference images with the highest similarity scores as "best matches."

Using CNN for image matching enables the system to accurately find images that are most similar to the content of the query image, providing high-quality candidate matches for subsequent location

clustering. This approach can also be adapted for various image datasets, with adjustable parameters in the VGG16 model and preprocessing steps to further enhance matching performance.

3.3.3. DBSCAN for Localization

In our system, the DBSCAN (Density-Based Spatial Clustering with Application of Noise) algorithm is used as a density-based clustering method to identify clusters in datasets containing noise and outliers. Unlike traditional clustering algorithms, DBSCAN does not require a pre-defined number of clusters. Instead, it defines clusters based on point density, making it effective for handling irregularly shaped clusters and noisy data in large spatial datasets. This capability makes DBSCAN particularly useful in applications like geographical data analysis and image-based localization (Ester et al. 1996).

In our application, DBSCAN is applied to identify dense regions of coordinates associated with matched images. By clustering these dense regions and ignoring outliers, DBSCAN helps accurately predict the potential location of the query image based on the spatial distribution of matched image coordinates. Reference images are obtained through SLAM technology, with known coordinates that serve as the basis for matching the user's query image location.

DBSCAN operates with two main parameters: `eps`, which defines the radius of the neighborhood around each point, and `min_samples`, which specifies the minimum number of points required to form a dense region or cluster. Points with sufficient neighbors within the `eps` radius are considered "core points," while isolated points that do not meet this threshold are marked as noise. DBSCAN can automatically ignore sparse, noisy regions, thereby enhancing the robustness of clustering results for noisy data.

1. **Principles and Advantages of DBSCAN**:
   - DBSCAN is an unsupervised clustering algorithm that does not require a predefined number of clusters. It forms clusters based on point density, meaning that points within a specified distance from each other are grouped into clusters, while sparse points are treated as noise.
   - This method is particularly well-suited for clustering geographic location data, as it groups dense areas of coordinate points into clusters and ignores isolated points, thereby reducing the impact of noise on localization results.
2. **Parameter Settings**:
   - DBSCAN has two key parameters: `min_samples` and `epsilon`.
     - `epsilon`: Defines the neighbourhood radius for clustering, so that points within this distance are considered neighbours.
     - `min_samples`: Specifies the minimum number of points required to form a valid cluster. If a point has enough neighbours within the `epsilon` radius, it becomes a "core point" and initiates cluster formation.
   - Adjusting these parameters appropriately influences the shape and number of clusters, as well as the stability of clustering results. Generally, smaller `epsilon` values and higher `min_samples` values yield tighter, more precise clusters.
3. **Calculating the Center of the Largest Cluster**:
   - After performing DBSCAN, the system generates multiple clusters, with the largest cluster typically representing the most likely location of the user. The centroid of this largest cluster is computed and used as the predicted location for the query image.
   - The centroid is calculated by averaging all coordinate points within the cluster, accurately pinpointing the "center" of the entire cluster.
4. **Ignoring Outliers**:
   - DBSCAN automatically labels outliers with the cluster ID of -1, indicating they are not part of any dense region. These points, possibly due to noise or accidental matches, are automatically ignored, and only dense regions are considered for localization.

By employing DBSCAN clustering, the system determines the potential location of the query image based on the distribution of matched image coordinates. This approach not only improves localization accuracy but also minimizes the impact of noise data, making it particularly effective for tasks like geographic data clustering and image-based localization.

3.3.4. Parameters we adjust

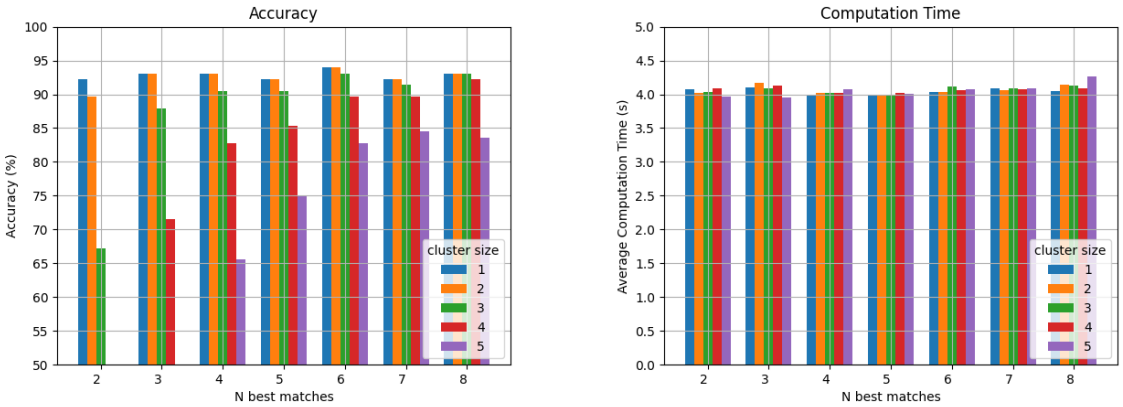
To optimize the system's performance, we conducted hyperparameter tuning to identify the best values for each parameter. This tuning process focused on three key parameters, each of which can significantly impact the final localization accuracy:

- **N Best Matches:** The number of top-matched images included in the clustering process for each user image taken.
- **DBSCAN Cluster Size:** The minimum number of neighboring points required to determine if a point is considered an outlier.
- **DBSCAN Epsilon Value:** The neighborhood search radius for each point in the clustering process.

Initially, we experimented with different values for the N Best Matches and DBSCAN Cluster Size parameters to determine the combination that yields the highest localization accuracy.

To perform this hyperparameter tuning, we used Python code to evaluate the localization accuracy by comparing the predicted position of each image to its true location label, which is stored in the validation sheet, across different parameter combinations. The results of this tuning process are presented in Figure 3.7a.

In parallel, we used the same parameter combinations to measure the system's computation time, allowing us to assess the impact of these combinations on overall computation efficiency. The results for computation time are shown in Figure 3.7b.



(a) Correct room match score with variable parameters.

(b) Computation time with variable parameters.

Figure 3.7: Accuracy and average computation time for changing N best matches and minimal cluster size.

From Figure 3.7a, we observe that accuracy fluctuates with different parameter combinations. These fluctuations vary with cluster size, as higher cluster sizes lead to lower accuracy for smaller values of N Best Matches. Despite these variations in accuracy, computation time remains stable across the different parameter combinations.

Among these combinations, the highest accuracy of 93.97% was achieved with $N = 6$ best matches per user image and a minimum DBSCAN cluster size of 2, with a computation time of approximately 4 seconds. The epsilon parameter was held constant at 2.0 during these tests. After identifying the optimal values for N Best Matches and DBSCAN Cluster Size, we used these settings to tune the epsilon parameter.

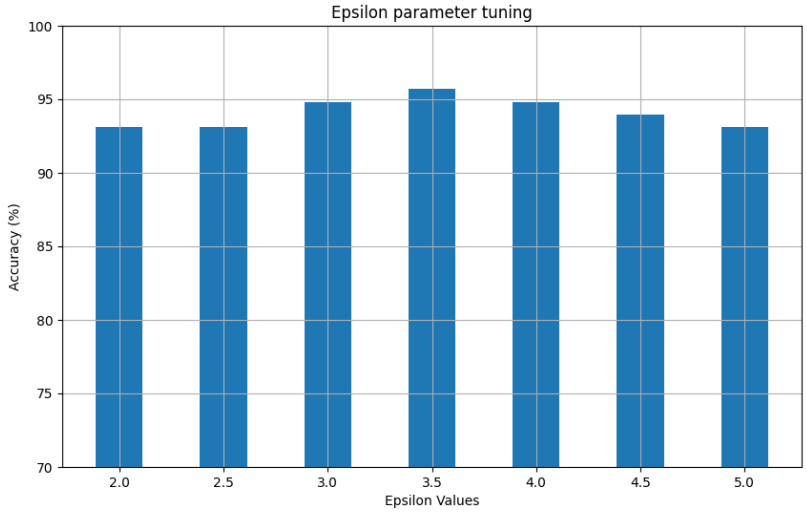


Figure 3.8: Barplot used for tuning the epsilon value of the DBSCAN clustering. Performed after finding the best match image number (N) and the cluster size at 6 and 2, respectively.

From Figure 3.8, we observe that the accuracy varies slightly across different epsilon values. Although the difference in accuracy is not drastic, there is a clear peak at 3.5. For epsilon values both lower and higher than 3.5, the accuracy remains relatively stable but does not reach the same peak level. This suggests that while the system's accuracy is fairly robust across a range of epsilon values, fine-tuning to the optimal value of 3.5 can lead to marginal improvements in performance. Through epsilon parameter tuning, we achieved the highest localization accuracy of 95.69% with an epsilon value of 3.5.

3.4. Routing and Navigation

In order to achieve route planning in indoor environments, based on the building floor plan (BK Bouwkunde Plattegrond), we first used QGIS to restore the spatial division and potential path network within the study area as much as possible. We divided the study area into compact polygonal grids to represent all classrooms, studios, and activity areas; meanwhile, we constructed node networks for indoor positioning and navigation respectively; finally, we selected the user coordinates obtained during image matching and the destination coordinates in the user interface as the starting point and destination, and then used the A* navigation algorithm to find the optimal path.

1. **Indoor Environment Structure Reconstruction**:
   - Based on the point cloud data obtained by SLAM Scanner, we process the building floor plan and divide the study area into compact polygonal grids.
   - We presented classrooms, studios, and activity areas as distinct polygons while restoring spatial divisions and potential path networks.
2. **Node Network Design**:
   - We constructed one node network for localisation validation and navigation, the distribution of node network is shown in the following Figure 3.9.
   - We positioned nodes at key locations such as room entrances, corridor intersections, and decision points.
   - We assigned attributes to both nodes (e.g., coordinates, room identifiers) and edges (e.g., coordinates, neighbors, labels) to enrich the network with relevant information. For each node in the node network, their fields are shown in the following Table 3.2.

Figure 3.9: Nodes Network

3. **A* Path-finding Algorithm**:
   - Algorithm Principle and Workflow:
     - Select starting point: User coordinates obtained from image matching
     - Choose a destination: Coordinates specified in the user interface
     - Implement the A* algorithm to find the optimal path:
       * Used a heuristic function to estimate the cost from any node to the goal
       * Explored nodes with the lowest combined cost (actual cost + heuristic estimate)
       * Maintained open and closed lists to track explored and unexplored nodes
       * Iteratively expanded the search until the goal was reached
     - Reconstructed the optimal path by backtracking from the goal to the start
     - Optimized for factors such as distance, time, or accessibility based on user preferences

Coordinates	ID	Label	Neighbors	Weights
Point (4.3705, 52.0057)	0	entrance	1	null
...	1	servicepunt	0, 2	null
...	2	lobby	1, 3, 47	null
...	3	corridor_e9	2, 4	null

Table 3.2: Node fields in the network

This approach enables efficient and accurate indoor navigation by combining detailed spatial information with path-finding techniques. The resulting system can guide users through complex indoor

environments, taking into account various constraints and user-specific requirements. For example, path planning for a specified room can avoid obvious obstacles such as walls and doorposts.

3.5. Alternative Approaches Considered

In the process of developing an effective image-based localization method, we explored several alternative approaches. This section briefly introduces the methodologies we tried that ultimately did not meet our accuracy requirements.

3.5.1. Combination of Structure from Motion and Point Cloud Comparison

Our initial approach involved using Structure from Motion (SfM) to reconstruct a 3D point cloud Negahdaripour and Lee 1992. Once reconstructed, we attempted to compare this point cloud with ground truth data obtained through SLAM scanning. However, the high computational demands of SfM and the challenges of accurately registering point clouds led us to abandon this approach. SfM required considerable processing time, which conflicted with our goal of providing localization results within approximately 5 seconds for a responsive web application. Additionally, performing point cloud registration without an initial positional guess proved extremely challenging. As a result, we pivoted to explore other methodologies that were better suited to our real-time localization objectives.

3.5.2. Image Matching Between User and Ground Truth Images

One approach we secondly experimented with was the Bag-of-Words (BoW) method for image matching Sivic and Zisserman 2003. This method involved a structured sequence of steps: first, SIFT was used to extract feature descriptors from images. A visual vocabulary was then created by clustering these descriptors from a training set using k-means, with each cluster center representing a ”visual word.” Each image was represented as a histogram of these visual words, reflecting the frequency of each word within the image. Finally, we used metrics such as cosine similarity to measure image similarity, comparing user images against pre-trained images to identify matches. Despite the systematic nature of this approach, the Bag-of-Words method did not yield the accuracy required for reliable indoor localization in our use case.

4

Results

4.1. Accuracy of Image Matching

To evaluate our algorithm's accuracy in room localization, we developed a script to perform image matching using a reference Google spreadsheet that logs each image's associated position ID, user image name, and known room location. This setup allows us to add more testing positions to our spreadsheet, enabling further refinement and targeted tests under various conditions, such as blurred images or images with noise (e.g., people in the background). Table 4.1 provides an example of the validation sheet with manually labeled true room information.

Position ID	User Image Name	True Room
0	room_val011.jpg	corridor_e1
0	room_val012.jpg	corridor_e1
0	room_val013.jpg	corridor_e1
1	room_val014.jpg	corridor_e2
1	room_val015.jpg	corridor_e2
1	room_val016.jpg	corridor_e2
2	room_val017.jpg	corridor_e5
...	...	...

Table 4.1: Example of room validation spreadsheet.

The script matches each image in the validation sheet to our database, assigning a room using a point-in-polygon algorithm with the building floor plan. The result of this localization is recorded in a new column, *found room*. To measure accuracy, we compared the *true room* against the *found room*, generating statistics on correct localization rates. In total, 355 user images were processed, of which 264 were assigned to the correct room, resulting in a room localization accuracy of 74.65% (265/355) for single-image matching, as summarized in Table 4.2.

Total test user images:	355
Correctly matched rooms:	265
Room match percentage:	74.65%

Table 4.2: Room Matching Accuracy using single image matching and best found parameters N=6. Since we want to test the localisation for 1 user image, cluster size is set to 1 and epsilon is arbitrary.

4.2. Accuracy of Localization Using Multiple Images

To improve localization accuracy, we extended our approach to use multiple image matches through DBSCAN clustering. This method clusters the calculated coordinates of each image and uses the center of the largest cluster as the user's location. To evaluate the performance of this method, our script processes all images that share a unique position ID. For each image, the 6 best matches are selected and input to the DBSCAN algorithm, with a minimal cluster size of 2. The resulting room assignment is recorded as the *found room*.

To find parameters of DBSCAN clustering and

In our final testing of this approach, we evaluated 116 positions. Of these, 111 were correctly matched to the appropriate room, resulting in a localization accuracy of 95.69% (111/116). Table 4.3 summarizes these accuracy results, including the total tested positions, correctly matched rooms, and the calculated room match percentage.

Total test user positions:	116
Correctly matched rooms:	111
Room match percentage:	95.69%

Table 4.3: Overall Room Matching Accuracy using combined image matching and best found parameters N = 6, DBSCAN cluster size = 2 and DBSCAN epsilon = 3.5.

In cases where localization did not match the true room, we further analysed the mismatches to understand the causes. Most errors stemmed from images capturing significant portions of adjacent rooms, leading the algorithm to assign coordinates in the neighbouring room polygon rather than the true room. This is understandable, as overlapping features often cause matching confusion. Additionally, some rooms share very similar layouts, such as identical tables or chairs, which sometimes results in localization errors—for example, between *Hall P* and *Hall Q*.

Table 4.4 provides a detailed report of the mismatches, including the position ID, *true room*, *found room*, test image IDs (listed in Appendix B), and the suspected reason for each mismatch. This breakdown has proven valuable for identifying systematic sources of error in localization. In several cases, discrepancies were due to human error in labelling the *true room*, suggesting that, in certain situations, our algorithm may even surpass manual (human) labelling in accuracy. These findings reinforce the robustness of our approach, showing potential for consistent performance under various conditions.

Position ID	True Room	Found Room	Test Image IDs	Reason for Mismatch
23	corridor_e2	bk_expo	{84, 85, 86}	Found better matching images in adjacent room
51	corridor_w2	corridor_w3	{168, 169, 170}	Found better matching images in adjacent room
62	hall_p	hall_q	{201, 202, 203}	Rooms are very similar
89	espressobar	hall_p	{282, 283, 284}	No clear reason found
112	corridor_w2	corridor_w2	{351, 352, 353}	Found better matching images in adjacent room

Table 4.4: Mismatch report for 116 test user positions, detailing the suspected reasons for mismatches. The test images are shown in Appendix B.

In this table, we analyze each position individually to identify the sources of error that may cause mislocalization.

For the first position, Position 23, which should be located in `corridor_e2`, the system incorrectly recognizes it as being in the `bk_expo` room, which is adjacent to the `corridor_e2` polygon. As shown

in Figure B.4, the distribution of matched image positions and the DBSCAN results indicate a higher density of matches within the bk_expo polygon, likely leading to the incorrect localization. This error originates from Figure B.3b, where the captured image includes a significant portion of bk_expo, resulting in matches with other ground truth images from bk_expo.

For the second position, Position 51, which should be located in corridor_w2, the system incorrectly identifies it as corridor_w3. Similar to the error in Position 23, one of the user's images captures a significant portion of corridor_w3, resulting in a denser distribution of matched points within corridor_w3. Consequently, the DBSCAN algorithm selects corridor_w3 as the final location. This error originates from Figure B.5b.

For the third position, Position 62, where the true location is hall_p, the system incorrectly identifies it as hall_q. This error occurs because hall_p and hall_q have similar layouts and colors, leading to many incorrect top-matched images and subsequently incorrect image positions. As shown in Figure B.8, the majority of the matched image positions are located within the hall_q polygon.

4.3. User Response Speed

To evaluate the user response time, we used browser tools to measure the response time. This metric represents the time taken by the server to process a request and return a complete response to the client, as measured with the browser's built-in tools. After taking 10 measurements, we found the average response time for localization to be 4.51 seconds. Given our initial target of having a repsonse under 5 seconds, this result is within an acceptable range.

5 Discussion

5.1. Accuracy

As described in chapter 4.2, localizing a users' position using multiple images yielded the highest room match result. Out of 116 test positions within the covered area inside the faculty building, our algorithm assigned 111 (95.69%) the correct room (Table 4.3). A detailed report of the 5 mismatches is presented in Table 4.4. Examining the cases where we fail to assign the correct room, a clear similarity can be found. Every mismatch is one of two cases;

1. **The user images show large portions of an adjacent room**
   This only happens when the user takes images in a 'virtual' room. It was a design choice to separate long corridors into sections to further increase the precision of localization. Because the corridor sections are not separated by physical obstructions, there are cases where a user is close to the border of two corridor sections when capturing their images. This causes our algorithm to find a larger cluster of image coordinates in the adjacent corridor section and thus localize the user there. Examples of these cases are shown in Figures B.3, B.5 and B.11.

   For this overarching fail set, it can be stated that this is a result of our own design. Changing the division of corridor sections, changes the matching result. Had we chosen to merge room polygons such as *corridor_w2* and *corridor_w3* (See Figure B.2), two out of the total five mismatches would have passed our test. This indicates that the manual floorplan division matters to the localization. Room polygons that are small and are openly connected to adjacent polygons are prone to be matched to neighbouring room polygons.

2. **The user images are taken in very similar rooms**
   Some rooms in the faculty can look identical. Mostly lecture halls and self study rooms show the characteristic of only having tables, chairs, chalkboards or monitors. In other words, there are few unique features to identify in the user images and the algorithm struggles to find the correct room. An examples of these cases are shown in Figures B.7.

   This overarching fail set is due to both user input quality and the closely resembling rooms. Efforts can be made to reduce these localization errors:

   - Taking more images in said rooms in the building scanning phase, offering more samples to match user images to.
   - Ask the user to take images of something rather than nothing. Images of white walls or chairs are less unique and prone to be mismatched. User images of unique room features have a better chance of a correct match.
   - Encourage the faculty to add unique features to rooms that do not have these. Adding a painting or poster can increase the identifiability of a room. After all, even for humans that know the building well, it will be hard to correctly distinguish *Hall P* from *Hall Q* (depicted in Figure B.7).

Overall, we are satisfied with the resulting room localization accuracy. A total of 95.69% correctly assigned rooms is above initial expectation. The 5 cases that did not localize correctly have a similarity among each other and can be classified as two mismatch categories. For both, there are multiple ideas to overcome the mismatching issue, and when acted upon, the accuracy will improve even further. For even better insight in the performance of our algorithm it is advised to collect more data. Ideally, more research can be done to what happens if the user images are blurry, noisy or taken in different lighting conditions.

5.2. Comparison of Indoor Navigation Methods

As we described in chapter 2, indoor positioning and navigation systems provide precise tracking and navigation services in enclosed spaces. Among these methods, WiFi-based localization, Bluetooth/Beacon technology, and vision-based approaches have gained significant popularity.

Method	Advantages	Disadvantages	Typical Use Cases
Image-based Method	- No extra hardware needed - High accuracy in feature-rich environments - Unaffected by signal interference - Effective in GPS-free environments	- Requires initial mapping - Computationally intensive - Poor in visually similar or featureless areas - Sensitive to lighting/layout changes	Navigation in places needing high precision, like museums or malls
Bluetooth/Beacon	- Low power consumption - Affordable deployment - Room-level accuracy	- Requires beacon installation - Signal affected by obstacles/interference - Limited range (10-30m) - Frequent battery replacement	Tracking in offices, shopping centers, or exhibitions
WiFi-based Method	- Uses existing WiFi infrastructure - Cost-effective for large areas - No extra device needed for WiFi-enabled devices	- Lower accuracy (5-15m) - Signal strength fluctuations - Inconsistent in crowded areas - Needs frequent calibration	General indoor positioning in large areas like airports or hospitals

Table 5.1: Comparison of Indoor Navigation Methods

Additionally, we compared the accuracy of our image-based method with the WiFi fingerprinting results from Assignment 2 of GEO1003 Positioning and Location Awareness. Both methods utilize cosine similarity for evaluating room accuracy. Figure 5.1 shows the WiFi results, where two out of the four examples failed to achieve accurate room identification. In contrast, our image-based method demonstrated superior performance, achieving a high accuracy rate of 95.67% across 36 unique polygons (rooms) and 116 images. This highlights the robustness of our approach in indoor environments and its effectiveness in overcoming some of the limitations observed with WiFi-based positioning.

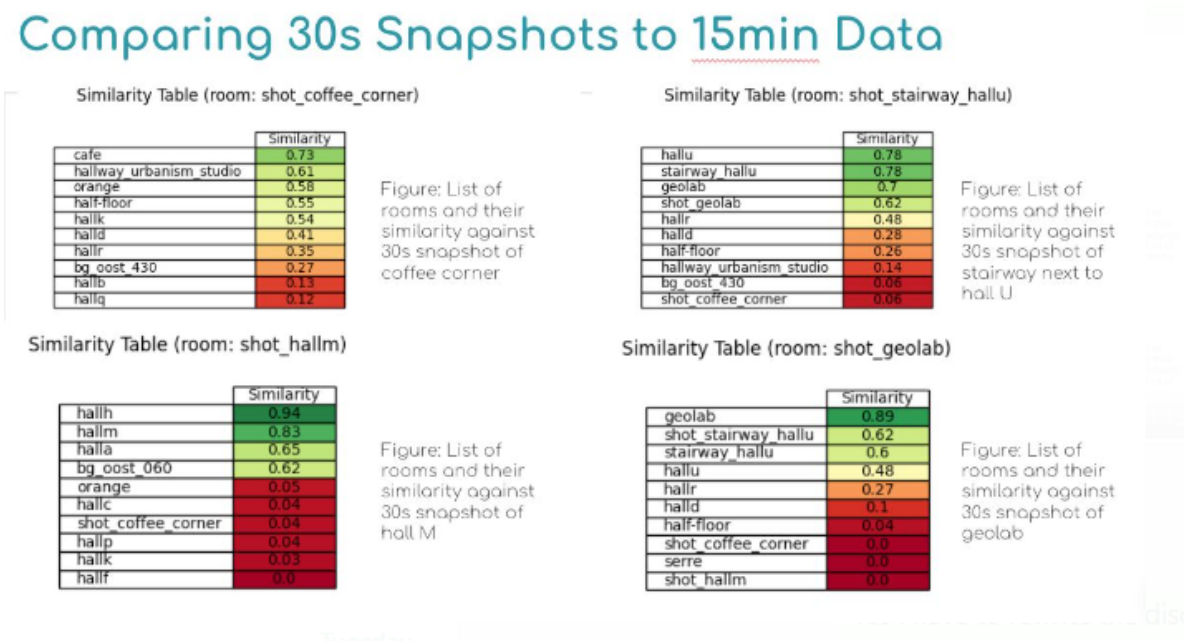


Figure 5.1: GEO1003 WiFi fingerprinting results

5.3. Aspect of Navigation

While evaluating our outcomes primarily against the research objective of providing accurate localization in indoor environments, analyzing the system's navigation experience from a user-friendly perspective is equally important. Our primary client, Adibah, specifically focuses on navigation using landmarks rather than simply providing directional prompts or distance measurements. Laraki 2024 highlights a similar issue in Google Maps' early experience in India, where users preferred navigating with landmarks over directions based solely on street names.

Yesiltepe, Dalton, and Torun 2021 provide a framework for effective landmarks, categorizing them by visibility, positioning at decision points, and salience in assisting wayfinding tasks. Applying these insights, our app currently only displays paths and room names, leaving significant room for enhancement. For example, as suggested by Yesiltepe, Dalton, and Torun 2021, incorporating distinct local landmarks within the building—such as a café, canteen, or visually distinct objects—could improve user orientation and overall navigation experience.

5.4. Limitation

The success of this system in achieving accurate positioning and navigation is influenced by several factors. Its core method relies on image-matching technology, meaning positioning accuracy heavily depends on the quality of visual data. For example, if the user captures images that are too high, include the ceiling, or are too close-up, featuring only a white wall, this can result in matching failures. Although poor lighting or furniture rearrangements may impact matching effectiveness, testing has shown that such changes have limited impact on positioning, and the system remains stable under moderate visual condition variations.

When multiple rooms share similar layouts, colors, and lighting, the system may confuse areas with similar visual characteristics, leading to positioning errors. This issue has been observed in our research with Hall P and Hall Q.

Certain steps in the system still require manual operation, such as placing navigation nodes. These manual setups are time-consuming and prone to errors, making expansion within large or complex buildings challenging. Although there are plans to automate these processes in the future, current manual setup remains a limitation.

The system's data processing requirements also add operational complexity. In large buildings or with

high-resolution images, steps such as point cloud processing and georeferencing are time-intensive and require specialized software, making it challenging to keep data updated and deploy the system effectively in large spaces, particularly in frequently changing environments.

Additionally, system performance is highly dependent on accurate initial mapping; any errors or inaccuracies in the generated map may lead to persistent positioning errors. In dynamic environments, periodic remapping may be required to maintain accuracy. Finally, integrating this image-based system with existing indoor positioning or building management systems poses challenges, including compatibility issues, data format changes, and infrastructure limitations. Addressing these integration challenges is essential to expanding the system's applicability and ensuring seamless operation across different indoor environments.

5.5. Impact of our research

Our browser-based approach offers significant usability advantages by eliminating the need for users to install a dedicated app, enhancing accessibility and user experience. This research demonstrates the potential for achieving accurate indoor localization directly within a browser, showcasing a novel direction for web-based geospatial technologies.

From an accuracy standpoint, our method performs competitively with traditional indoor localization systems, despite not relying on external equipment like Wi-Fi or Bluetooth beacons. However, while accuracy remains high, there are occasional localization discrepancies, suggesting further refinement opportunities.

Regarding cost, although our initial setup involved a SLAM scanner for ground truth data, the approach could be simplified to make data acquisition more affordable. Users without specialized equipment can still implement this localization method by taking representative indoor photos and labeling them based on room or area. Alternatively, they can rely on image matching alone, without additional DBSCAN processing, to identify the most probable room layout. Although this approach achieves an accuracy of 74%, labeled room diagrams are still required. This adaptability underscores the product's potential for scalable, cost-effective deployment in various indoor environments.

6

Future Work

6.1. Support for Multiple Floors

The indoor localization and navigation algorithm we developed has been tested on a substantial portion of the ground floor at TU Delft's Faculty of Architecture & the Built Environment. This provided a proof of concept, setting the foundation for extending our approach to cover an entire building. Achieving this broader scope would require incorporating support for multiple floors.

The primary challenge in supporting multiple floors lies in generating floor plans and constructing the navigation graph to handle pathfinding across levels. Currently, the user interface displays a 2D map of the ground floor, with the user's position based solely on the X and Y coordinates from the SLAM scanner images. While the device also provides a Z coordinate, it is currently unused. For effective 3D localization and navigation, the algorithm would need to be redesigned to integrate this height information into both image and graph node coordinates.

In addition, the user interface should be adapted to support 3D navigation. One initial approach could involve displaying a 2D map for the user's current floor, then automatically switching to the 2D plan of the next floor as the user moves to stairs or an elevator. There are two potential methods for enabling this switch:

- **User-Triggered Floor Change**: Adding a button for users to indicate when they wish to change floors could be useful for visualizing the entire route in advance.
- **Automatic Floor Detection**: Using the device's Internal Measurement Unit (IMU) for live location tracking would enable the algorithm to detect when the user reaches stairs or an elevator and automatically switch the floor view.

The first option is simpler and would require minimal modifications to the existing code, while the second option would involve integrating live tracking technology. Though more time-intensive, live tracking could enhance user experience by providing seamless, real-time navigation feedback.

6.2. Enhance CNN

While our current system achieves an accuracy of 95.69%, we could further enhance matching precision by exploring other deep learning models beyond VGG16, such as **ResNet** and **MobileNet**. These models each bring unique advantages: ResNet utilizes residual connections, enabling a deeper architecture that extracts more refined image features, making it suitable for high-resolution tasks (He et al. 2016); MobileNet, on the other hand, is a lightweight CNN architecture specifically designed for resource-constrained environments. Its use of depthwise separable convolutions maintains accuracy while significantly reducing computational load (Howard et al. 2017). These distinctive features make both models promising candidates for testing and comparison in our matching task.

Another promising approach for future improvement is **data augmentation**. By applying transformations such as cropping, brightness adjustment, and limited rotation, we can increase the variety of

training samples, which helps the model generalize better to different user images. However, care must be taken to avoid excessive rotation or transformations that would produce unnatural perspectives, as users typically do not capture images in an inverted or highly skewed orientation. Appropriate data augmentation can thus provide more robust training data without compromising the realism of user images.

For filtering matched results, DBSCAN currently leverages spatial density to remove potential mismatches. However, other strategies could be explored, such as running multiple models in parallel, allowing each model to select its top matches, followed by a majority-vote system to determine the final N best matches.

It's important to note that these improvements may increase runtime, so a balance between accuracy and processing time is necessary to ensure users don't experience extended wait times.

6.3. Automation of building processing

In this research, we generated necessary data manually, such as floorplan and graph data structures for pathfinding. However, this manual approach could be time-intensive and laborious for faculty administrators managing large facilities. Automating these data-preparation processes would significantly streamline implementation and scalability.

For instance, Bot, Nourian, and Verbree 2019 proposes a method for creating graph data structures for indoor environments by referencing BIM (Building Information Modeling) data. Additionally, Gankhuyag and Han 2020 suggest an approach to generate floorplan data from large-scale Terrestrial Laser Scanning (TLS) datasets. Alternatively, if available, floorplans could also be extracted from semantic 3D city models that contain indoor information. Integrating these automated processes would simplify data preparation for users and make our localization approach more accessible and efficient.

7

Conclusion

The research aimed to provide an accurate and accessible method for indoor localization without requiring app installation. It demonstrates that a visual-based approach, implemented through a web application, can serve as a viable alternative to infrastructure-dependent systems like WiFi or Bluetooth. By leveraging high-accuracy image matching with CNN and user localization via DBSCAN clustering, this method achieves impressive accuracy using only images and suggests optimal parameter configurations based on testing. However, challenges persist, particularly in adapting to dynamic environments, supporting multi-floor navigation, and requiring active user engagement for scanning. Additionally, considerations around spatial awareness, including how to present localized results to users, warrant further discussion. Future work should focus on enhancing CNN capabilities, automating data acquisition and processing, and expanding multi-floor support to boost usability and adaptability in more complex indoor environments.

References

Abo-Zeed, Mohammad et al. (2019). “Survey on land mobile satellite system: Challenges and future research trends”. In: *IEEE Access* 7, pp. 137291–137304.

Bello, Azeez (2023). “Recognition and classification of mechanical tools through machine learning”. In.

Bot, F. J., P. Nourian, and E. Verbree (June 2019). “A GRAPH-MATCHING APPROACH TO INDOOR LOCALIZATION USING A MOBILE DEVICE AND A REFERENCE BIM”. In: *The international archives of the photogrammetry, remote sensing and spatial information sciences/International archives of the photogrammetry, remote sensing and spatial information sciences* XLII-2/W13, pp. 761–767. DOI: 10.5194/isprs-archives-xlii-2-w13-761-2019. URL: https://isprs-archives.copernicus.org/articles/XLII-2-W13/761/2019/.

Cadena, Cesar et al. (2016). “Past, Present, and Future of Simultaneous Localization and Mapping: Toward the Robust-Perception Age”. In: *IEEE Transactions on Robotics* 32.6, pp. 1309–1332. DOI: 10.1109/TRO.2016.2624754.

Dardavesis, I., E. Verbree, and A. Rafiee (2023). “Indoor localisation and location tracking in indoor facilities based on LiDAR point clouds and images of the ceilings”. In: *AGILE: GIScience Series* 4, p. 4. DOI: 10.5194/agile-giss-4-4-2023. URL: https://agile-giss.copernicus.org/articles/4/4/2023/.

Drira, Anis (2006). “GPS navigation for outdoor and indoor environments”. In: *University of Tennessee, Knoxville*.

Ester, Martin et al. (1996). “A density-based algorithm for discovering clusters in large spatial databases with noise”. In: *kdd*. Vol. 96. 34, pp. 226–231.

Fanga, Y. et al. (2016). “A Point Cloud-Vision Hybrid Approach for 3 D Location Tracking of Mobile Construction Assets”. In: URL: https://api.semanticscholar.org/CorpusID:36655164.

Gankhuyag, Uuganbayar and Ji-Hyeong Han (Apr. 2020). “Automatic 2D Floorplan CAD Generation from 3D Point Clouds”. In: *Applied Sciences* 10.8, p. 2817. DOI: 10.3390/app10082817. URL: https://www.mdpi.com/2076-3417/10/8/2817.

Georgios Triantafyllou, Edward Verbree and Azarakhsh Rafiee (2024). “Indoor localisation through Isovist fingerprinting from point clouds and floor plans”. In: *Journal of Location Based Services* 0.0, pp. 1–20. DOI: 10.1080/17489725.2024.2320642. eprint: https://doi.org/10.1080/17489725.2024.2320642. URL: https://doi.org/10.1080/17489725.2024.2320642.

He, Kaiming et al. (June 2016). “Deep Residual Learning for Image Recognition”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Howard, Andrew G et al. (2017). “MobileNets: efficient convolutional neural networks for mobile vision applications (2017)”. In: *arXiv preprint arXiv:1704.04861* 126.

Jiang, San, Cheng Jiang, and Wanshou Jiang (2020). “Efficient structure from motion for large-scale UAV images: A review and a comparison of SfM tools”. In: *ISPRS Journal of Photogrammetry and Remote Sensing* 167, pp. 230–251. ISSN: 0924-2716. DOI: https://doi.org/10.1016/j.isprsjprs.2020.04.016. URL: https://www.sciencedirect.com/science/article/pii/S0924271620301131.

Khan, Dawar et al. (2022). “Recent advances in vision-based indoor navigation: A systematic literature review”. In: *Computers & Graphics* 104, pp. 24–45.

Kuhn, Janet (2009). “Decrypting the MoSCoW analysis”. In: *The workable, practical guide to Do IT Yourself* 5.

Laraki, Elizabeth (Sept. 2024). *Google Maps UX: The India Conundrum*. URL: https://elizlaraki.substack.com/p/google-maps-ux-the-india-conundrum.

Liu, Hong et al. (2018). “An End-To-End Siamese Convolutional Neural Network for Loop Closure Detection in Visual Slam System”. In: *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 3121–3125. DOI: 10.1109/ICASSP.2018.8462426.

Mainetti, Luca, Luigi Patrono, and Ilaria Sergi (2014). “A survey on indoor positioning systems”. In: *2014 22nd International Conference on Software, Telecommunications and Computer Networks (SoftCOM)*, pp. 111–120. DOI: 10.1109/SOFTCOM.2014.7039067.

Mascarenhas, Sheldon and Mukul Agarwal (2021). “A comparison between VGG16, VGG19 and ResNet50 architecture frameworks for Image Classification”. In: *2021 International conference on disruptive technologies for multi-disciplinary research and applications (CENTCON)*. Vol. 1. IEEE, pp. 96–99.

Negahdaripour, Shahriar and Shinhak Lee (Dec. 1992). “Motion recovery from image sequences using only first order optical flow information”. In: *International Journal of Computer Vision* 9.3, pp. 163–184. DOI: 10.1007/bf00133700. URL: https://link-springer-com.tudelft.idm.oclc.org/article/10.1007/BF00133700#citeas.

Pasricha, Sudeep (2021). *Overview of Indoor Navigation Techniques*, pp. 1141–1170. DOI: 10.1002/9781119458555.ch37.

Puricer, Pavel and Pavel Kovar (2007). “Technical Limitations of GNSS Receivers in Indoor Positioning”. In: *2007 17th International Conference Radioelektronika*, pp. 1–5. DOI: 10.1109/RADIOELEK.2007.371487.

Rawat, Waseem and Zenghui Wang (2017). “Deep Convolutional Neural Networks for Image Classification: A Comprehensive Review”. In: *Neural Computation* 29.9, pp. 2352–2449. DOI: 10.1162/neco_a_00990.

Sakpere, Wilson, Michael Adeyeye-Oshin, and Nhlanhla BW Mlitwa (2017). “A state-of-the-art survey of indoor positioning and navigation systems and technologies”. In: *South African Computer Journal* 29.3, pp. 145–197.

Simonyan, Karen and Andrew Zisserman (2014). “Very deep convolutional networks for large-scale image recognition”. In: *arXiv preprint arXiv:1409.1556*.

Sithole, G. and S. Zlatanova (2016). “POSITION, LOCATION, PLACE AND AREA: AN INDOOR PERSPECTIVE”. In: *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences* III-4, pp. 89–96. DOI: 10.5194/isprs-annals-III-4-89-2016. URL: https://isprs-annals.copernicus.org/articles/III-4/89/2016/.

Sivic and Zisserman (2003). “Video Google: a text retrieval approach to object matching in videos”. In: 1470–1477 vol.2. DOI: 10.1109/ICCV.2003.1238663.

Subedi, Santosh and Jae-Young Pyun (2020). “A survey of smartphone-based indoor positioning system using RF-based wireless technologies”. In: *Sensors* 20.24, p. 7230.

Wielandt, Stijn and Lieven De Strycker (2017). “Indoor multipath assisted angle of arrival localization”. In: *Sensors* 17.11, p. 2522.

Xu, Lichao et al. (2019). “An Occupancy Grid Mapping enhanced visual SLAM for real-time locating applications in indoor GPS-denied environments”. In: *Automation in Construction* 104, pp. 230–245. ISSN: 0926-5805. DOI: https://doi.org/10.1016/j.autcon.2019.04.011. URL: https://www.sciencedirect.com/science/article/pii/S0926580518311506.

Yesiltepe, Demet, Ruth Conroy Dalton, and Ayse Ozbil Torun (Mar. 2021). “Landmarks in wayfinding: a review of the existing literature”. In: *Cognitive Processing* 22.3, pp. 369–410. DOI: 10.1007/s10339-021-01012-x. URL: https://link.springer.com/article/10.1007/s10339-021-01012-x.

Zhang, Hong, Bo Li, and Dan Yang (2010). “Keyframe detection for appearance-based visual SLAM”. In: *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 2071–2076. DOI: 10.1109/IROS.2010.5650625.

Zhou, Nanxiang (2024). “Image recognition in depth: comparative study of CNN and Pre-trained VGG16 architecture for classification tasks”. In: *Second International Conference on Physics, Photonics, and Optical Engineering (ICPPOE 2023)*. Vol. 13075. SPIE, pp. 553–559.

Zhu, Jian, Leyuan Fang, and Pedram Ghamisi (2018). “Deformable Convolutional Neural Networks for Hyperspectral Image Classification”. In: *IEEE Geoscience and Remote Sensing Letters* 15.8, pp. 1254–1258. DOI: 10.1109/LGRS.2018.2830403.

A

Libraries and Technologies Utilized

Name	Brief Explanation
Python	Server-side language
Pytorch	Used for Convolutional Neural Network (CNN) processing
FastAPI	Web server framework for building APIs
sklearn	Used for the DBSCAN clustering algorithm
Networkx	For pathfinding and graph-based algorithms
TypeScript	Language for developing the user interface
React	JavaScript library for building user interfaces
Maplibre	For displaying interactive maps in the user interface
Docker	For containerizing the application for deployment
Google Cloud Platform	Infrastructure for server-side hosting and services
Netlify	Platform for deploying the user interface
QGIS	For creating pathfinding data and spatial analysis
CloudCompare	For processing point cloud data
Faro Connect Viewer	For pre-processing Geo SLAM data
OpenCV	For projecting 360-degree panorama into cubemap images

B

Extra Figures

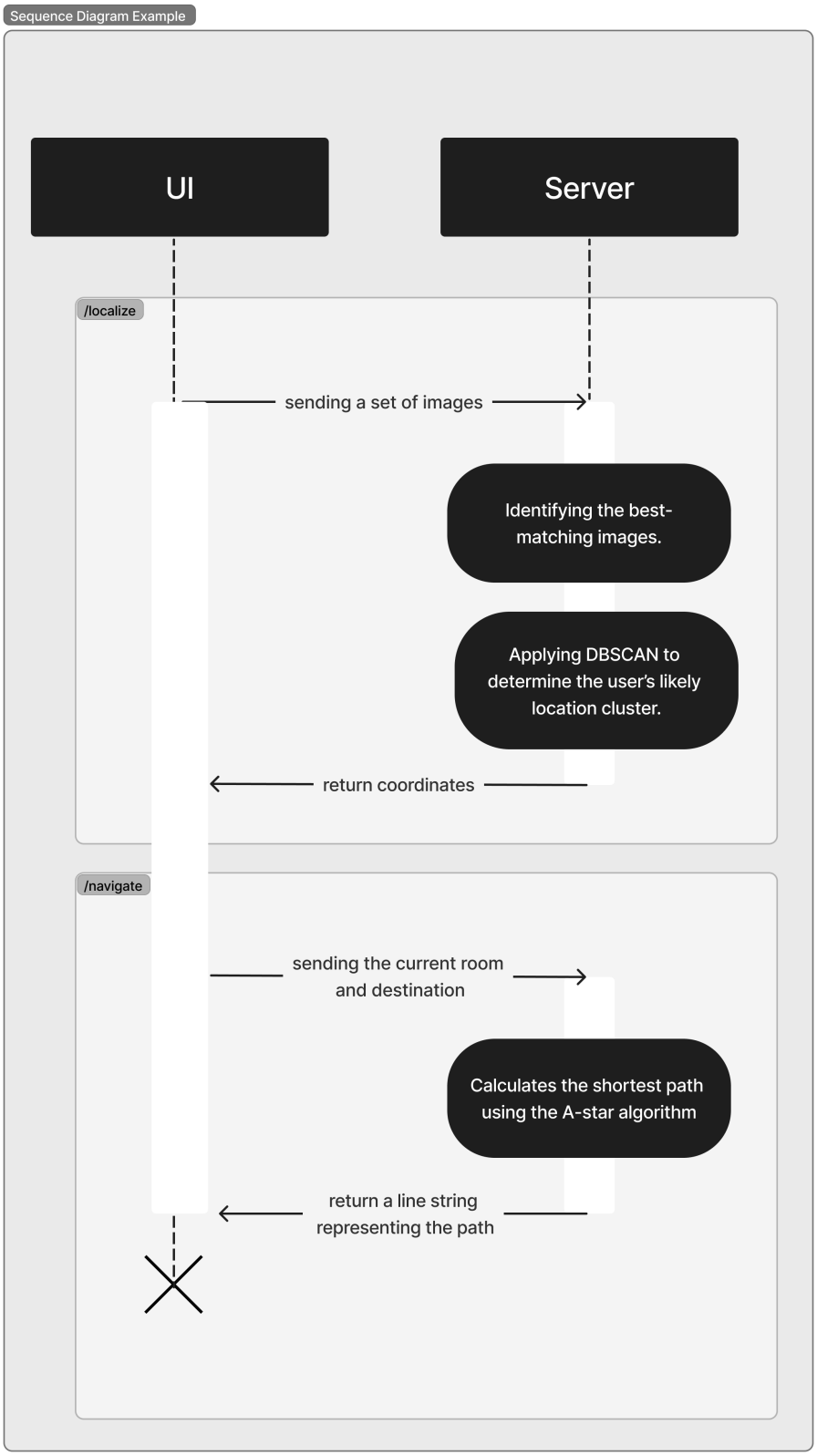
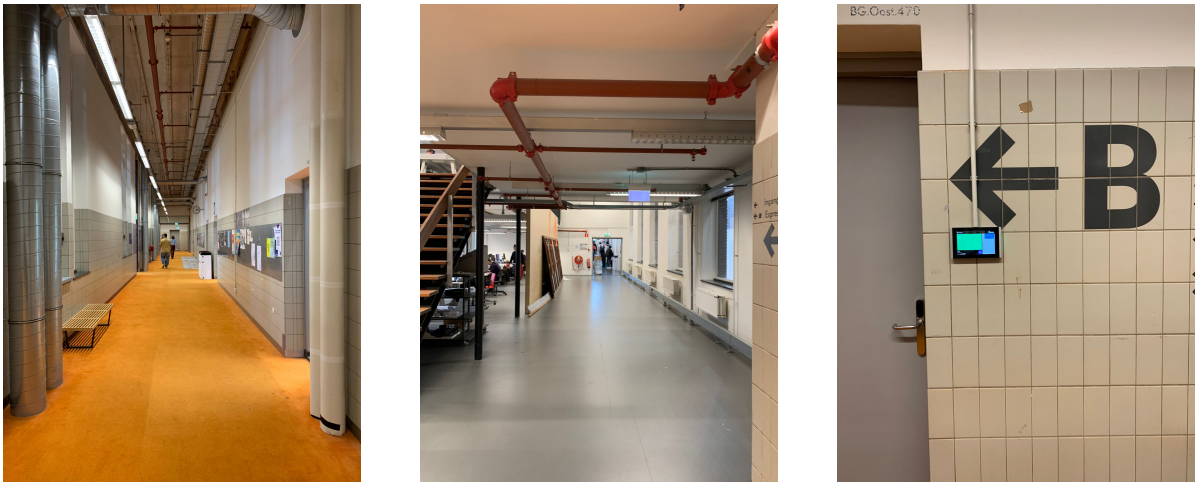


Figure B.1: Sequence Diagram for Localization and Navigation Workflow.

Figure B.2: Digitized Floorplan



(a) Test user image 84. **(b)** Test user image 85. **(c)** Test user image 86.

Figure B.3: User images used to localize test position 23. The expected room was *corridor_e2* but the outcome was *bk_expo*.

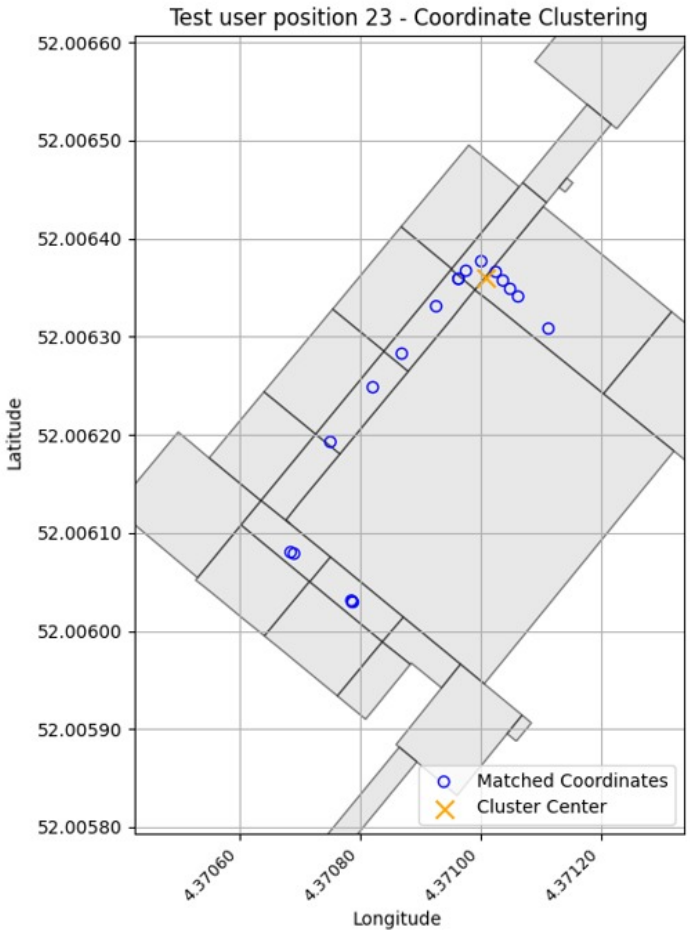
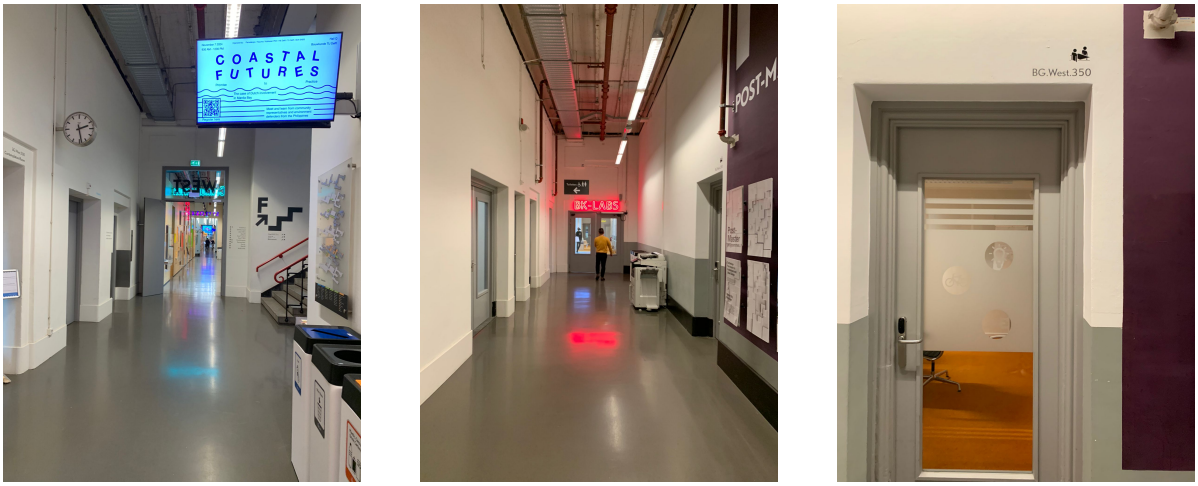


Figure B.4: DBSCAN clustering for 3 images taken on test user position 23. The coordinates of the N=6 best matches per image are plotted in blue and the largest cluster center is plotted in orange. The expected room was *corridor_e2* but the outcome was *bk_expo*. (see Figure B.2)



(a) Test user image 168. **(b)** Test user image 169. **(c)** Test user image 170.

Figure B.5: User images used to localize test position 51. The expected room was *corridor_w2* but the outcome was *corridor_w3*.

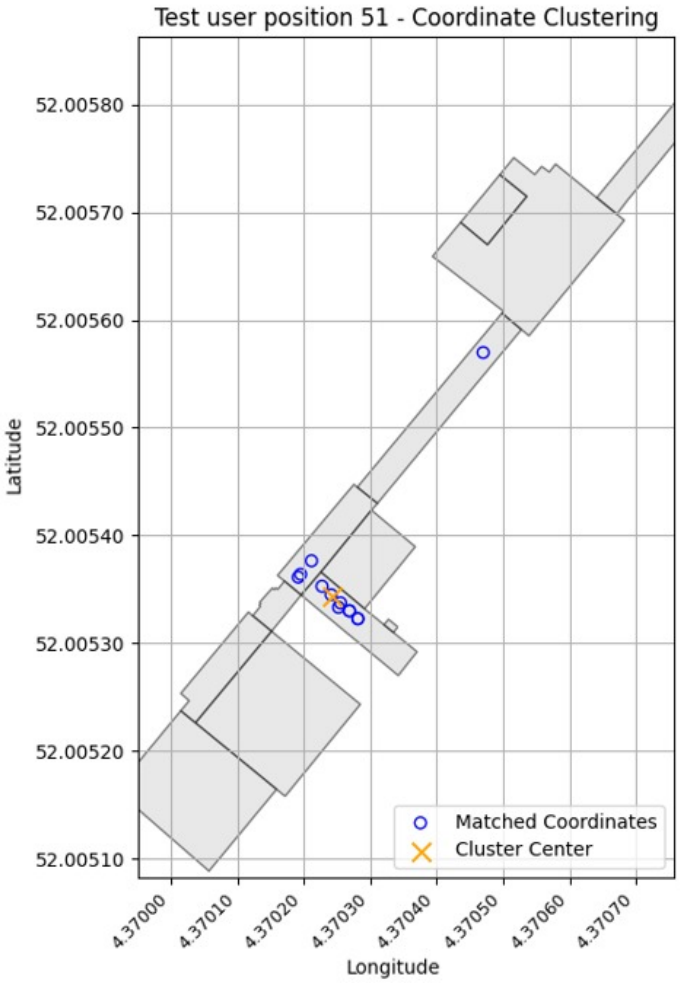
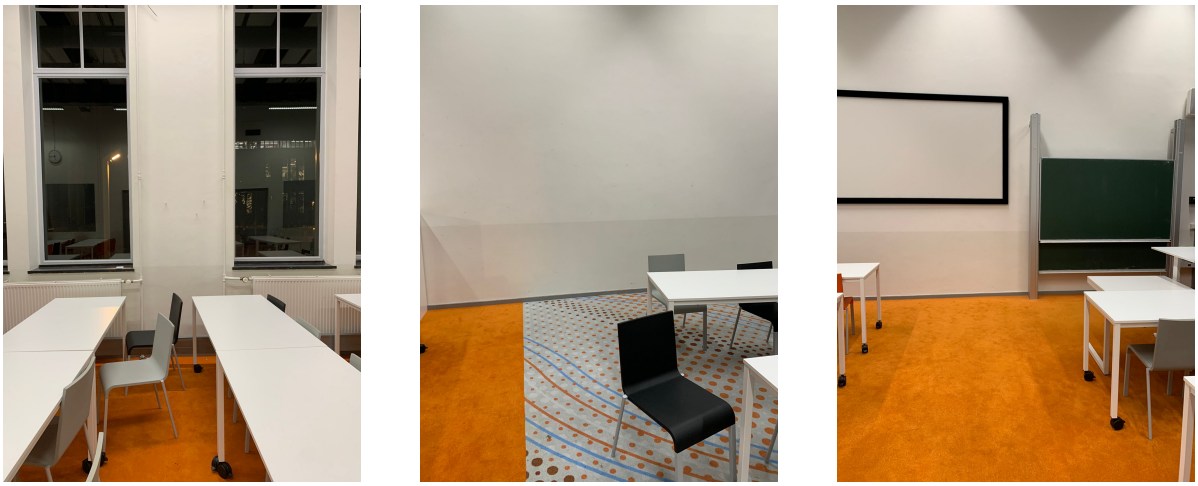


Figure B.6: DBSCAN clustering for 3 images taken on test user position 51. The coordinates of the N=6 best matches per image are plotted in blue and the largest cluster center is plotted in orange. The expected room was *corridor_w2* but the outcome was *corridor_w3*. (see Figure B.2)



(a) Test user image 201. **(b)** Test user image 202. **(c)** Test user image 203.

Figure B.7: User images used to localize test position 62. The expected room was *hall_p* but the outcome was *hall_q*.

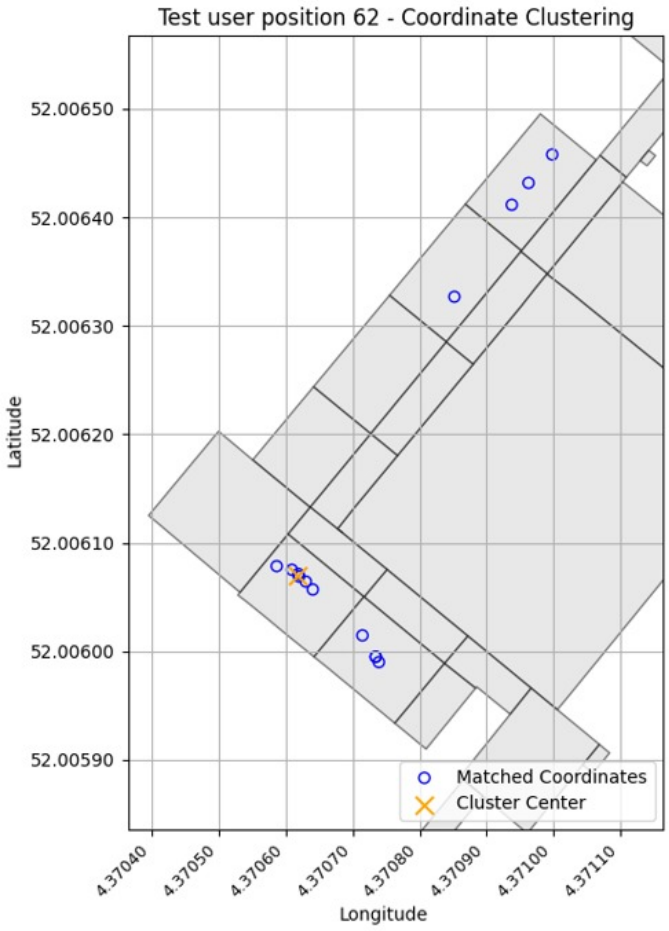
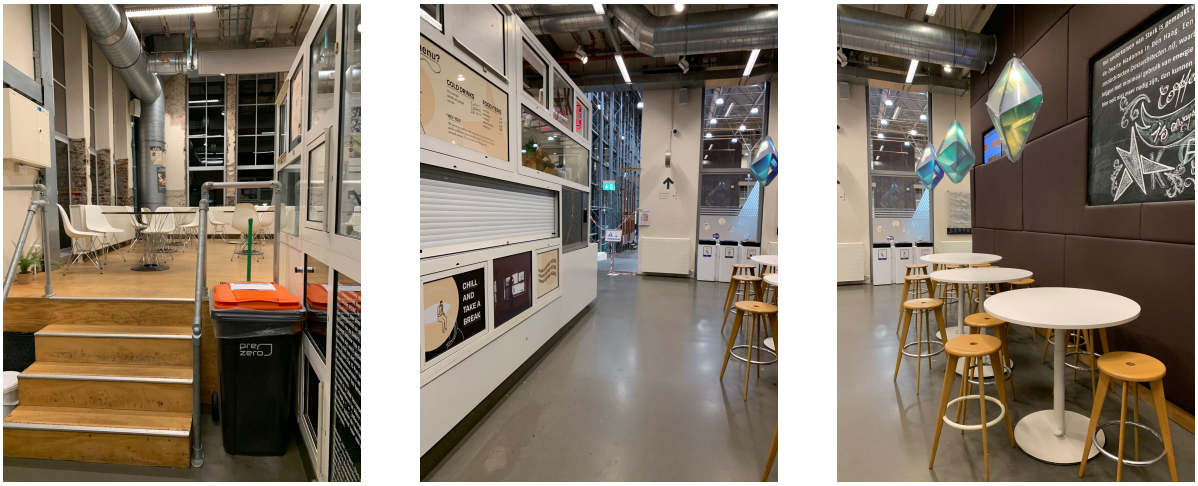


Figure B.8: DBSCAN clustering for 3 images taken on test user position 51. The coordinates of the N=6 best matches per image are plotted in blue and the largest cluster center is plotted in orange. The expected room was *hall_p* but the outcome was *hall_q*. (see Figure B.2)



(a) Test user image 282. **(b)** Test user image 283. **(c)** Test user image 284.

Figure B.9: User images used to localize test position 89. The expected room was *espressobar* but the outcome was *hall_p*.

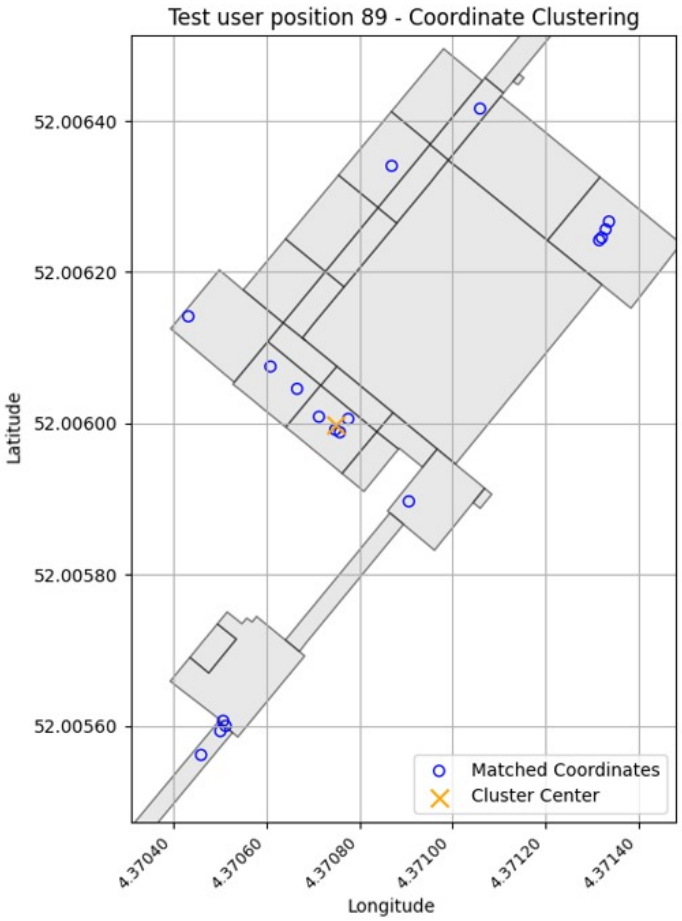


Figure B.10: DBSCAN clustering for 3 images taken on test user position 89. The coordinates of the N=6 best matches per image are plotted in blue and the largest cluster center is plotted in orange. The expected room was *espressobar* but the outcome was *hall_p*. (see Figure B.2)

(a) Test user image 351. **(b)** Test user image 352. **(c)** Test user image 353.

Figure B.11: User images used to localize test position 112. The expected room was *corridor_w3* but the outcome was *corridor_w2*.

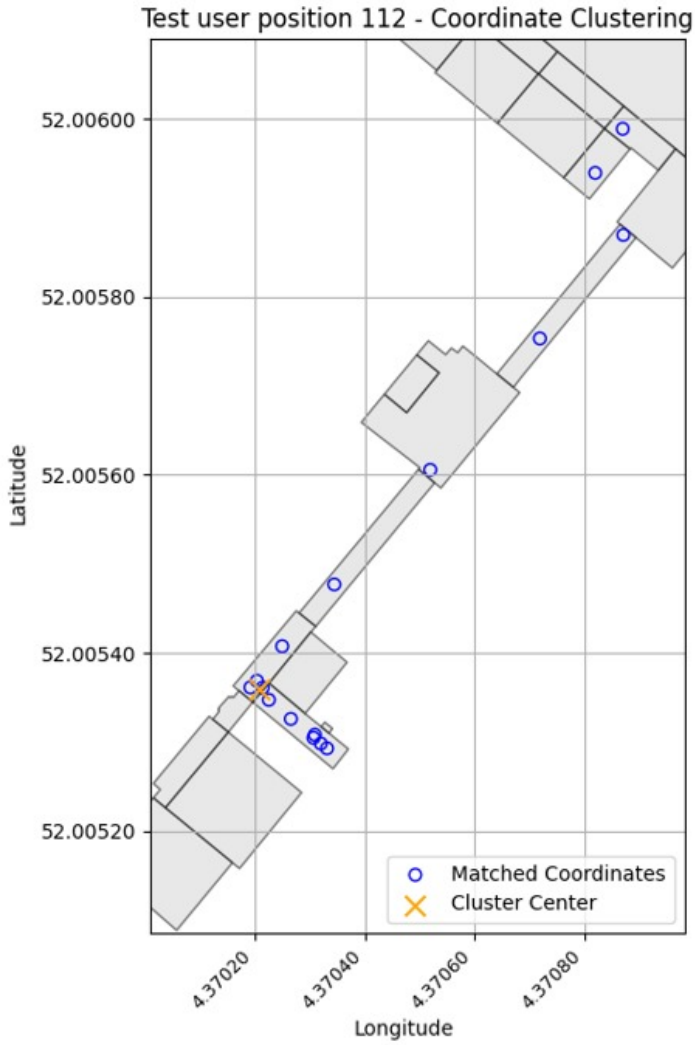


Figure B.12: DBSCAN clustering for 3 images taken on test user position 112. The coordinates of the N=6 best matches per image are plotted in blue and the largest cluster center is plotted in orange. The expected room was *corridor_w3* but the outcome was *corridor_w2*. (see Figure B.2)

C

Members and Roles

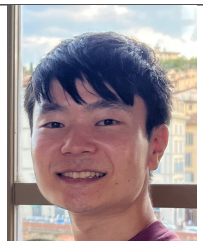
Photo	Member Information
	**Name:** Noah Alting **Nationality:** Dutch **Background:** BSc in Applied Physics **Interests:** Music, environment, programming **Sports:** (Kite-)surfing, tennis **Expertise:** Python, SQL, Web Design **Role:** Report Manager
	**Name:** Hidemichi Baba **Nationality:** Japanese **Background:** BA in Economics **Interests:** Programming, 3D Geo, hiking, camping **Sports:** Bouldering, Snowboarding, Baseball **Expertise:** Programming, Web Development, Cloud architecture **Role:** Team Leader
	**Name:** Der Derian Auliyaa Bainus **Nationality:** Indonesian **Background:** BSc in Geodesy and Geomatics **Interests:** Point Cloud, 3D modelling, cooking **Sports:** Cycling, jogging **Expertise:** Point cloud processing **Role:** Data Acquisition Manager
	**Name:** Hsin-Yu Cheng **Nationality:** Taiwanese **Background:** BSc in Geography, BA in Philosophy **Interests:** Playing games **Sports:** Juggling, fire dancing **Expertise:** Machine Learning **Role:** Secretary
	**Name:** Jiaoyang Wu **Nationality:** Chinese **Background:** BSc in Geographic Information Science **Interests:** Texas poker **Sports:** Cycling, Table tennis **Expertise:** Image Processing (RS), Neural Network **Role:** Technical Manager

Table C.1: Team Members Overview

Roles	Job Description
Team Leader	• Manage schedule • Spokesperson for external people • Creating meeting agenda
Report Manager	• Managing report and presentation structure • Dividing writing tasks and collecting texts
Secretary	• Plan a meeting with an external or internal and contact them • Taking notes during meetings
Data Acquisition Manager	• Managing data collection process • Responsible for ensuring the availability of equipment when needed
Technical Manager	• Managing GitHub repository • Testing, aligning, and modifying code to ensure the quality of the product

Table C.2: Roles and Job Descriptions

D

Supervisors

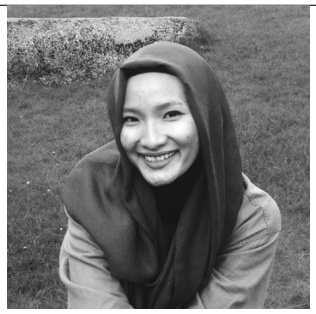
Photo	Supervisor Information
	**Name:** Edward Verbree **Nationality:** Dutch **Email:** e.verbree@tudelft.nl **Association:** TU Delft
	**Name:** Adibah Nurul Yunisya **Nationality:** Indonesian **Email:** a.n.yunisya@tudelft.nl **Association:** TU Delft
	**Name:** Niels van der Vaart **Nationality:** Dutch **Email:** c.g.vandervaart@tudelft.nl **Association:** TU Delft, Esri Nederland

Table D.1: Supervisors