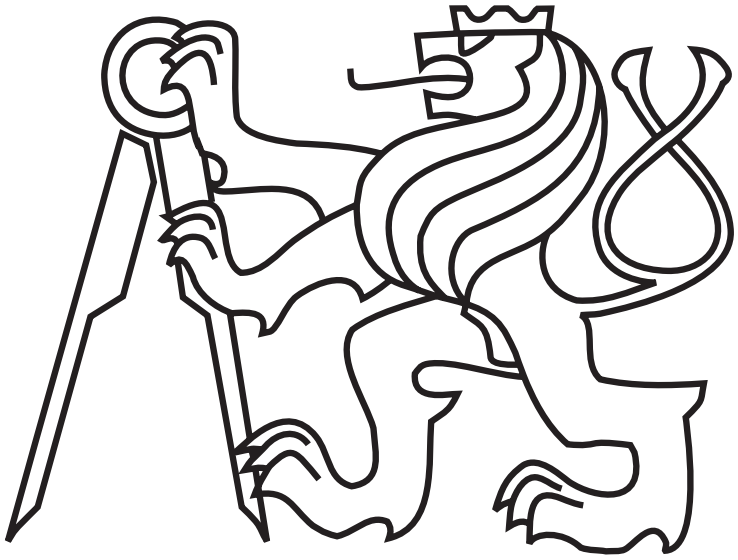
CZECH TECHNICAL UNIVERSITY IN PRAGUE

Faculty of Electrical Engineering

MASTER'S THESIS



Vivek Punia

Map Management System for Visual Teach and Repeat Navigation

Department of Cybernetics

Thesis supervisor: **Ing. Zdeněk Rozsypálek**

MASTER'S THESIS ASSIGNMENT

I. Personal and study details

Student's name: **Punia Vivek** Personal ID number: **506079**

Faculty / Institute: **Faculty of Electrical Engineering**

Department / Institute: **Department of Cybernetics**

Study program: **Cybernetics and Robotics**

II. Master's thesis details

Master's thesis title in English:

Map Management System for Visual Teach and Repeat Navigation

Master's thesis title in Czech:

Systém správy map pro vizuální navigaci

Guidelines:

1) Get acquainted with the current navigation and estimation stack for VT&R.
2) Implement improved map management for VT&R using database services.
3) Create a set of tools to extract metadata about the map from raw data.
4) Research and establish a set of performance criteria for long-term robot navigation.
5) Implement a set of planning methods to find a sequence of maps (for given time and conditions etc.) that optimize the established criteria.
6) Conduct an experiment with a real or simulated robot to prove functionality of the map management system.

Bibliography / sources:

[1] M. Gadd, P. Newman: The Data Market: Policies for Decentralised Visual Localisation. https://arxiv.org/pdf/1801.05607.pdf
[2] T. Krajnik, F. Majer, L. Halodova, T. Vintr: Navigation without localisation: reliable teach and repeat based on the convergence theorem. 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
[3] T. Krajník, J.P. Fentanes, J.M. Santos, T Duckett: Fremen: Frequency map enhancement for long-term mobile robot autonomy in changing environments. IEEE Transactions on Robotics. 2017.
[4] Z. Rozsypálek, G. Broughton, P. Linder, T. Rouček, J. Blaha, L. Mentzl, K. Kusumam, T. Krajník. Contrastive Learning for Image Registration in Visual Teach and Repeat Navigation. Sensors 2022, 22, 2975.
[5] Z. Rozsypálek, T. Rouček, T. Vintr and T. Krajník, Multidimensional Particle Filter for Long-term Visual Teach and Repeat in Changing Environments in IEEE Robotics and Automation Letters, doi: 10.1109/LRA.2023.3244418.

Name and workplace of master's thesis supervisor:

Ing. Zdeněk Rozsypálek Department of Computer Science FEE

Name and workplace of second master's thesis supervisor or consultant:

Date of master's thesis assignment: **22.02.2023** Deadline for master's thesis submission: **26.05.2023**

Assignment valid until: **22.09.2024**

Ing. Zdeněk Rozsypálek	prof. Ing. Tomáš Svoboda, Ph.D.	prof. Mgr. Petr Páta, Ph.D.
Supervisor's signature	Head of department's signature	Dean's signature

III. Assignment receipt

The student acknowledges that the master's thesis is an individual work. The student must produce his thesis without the assistance of others, with the exception of provided consultations. Within the master's thesis, the author must state the names of consultants and include a list of references.

Date of assignment receipt

Student's signature

Declaration

I declare that the presented work was developed independently and that I have listed all sources of information used within it in accordance with the methodological instructions for observing the ethical principles in the preparation of university theses.

Prague, date ....

Signature

Acknowledgements

First and foremost, I want to express my heartfelt gratitude to my parents, Sunehri Devi and Vinod Punia, for their support and sacrifices they have made throughout my academic journey. I am truly blessed to have them as my pillars of strength. I would also like to extend my thanks to my sister, Kiran, her presence in my life has been a source of comfort, and her belief in my potential has pushed me to overcome challenges. I am incredibly grateful to my girlfriend, Parnika, her encouragement and belief in me have been invaluable. Her love and presence have been a constant source of inspiration and motivation.

I am deeply grateful to my supervisor, Zdeněk Rozsypálek, for his expertise, patience, and unwavering support, without whom this work would not have been possible. His guidance and insightful feedback have been instrumental in shaping the direction of my research and the quality of this thesis. Finally, I am indebted to the researchers, and authors whose work has formed the basis of my research. Their contributions have been invaluable in shaping my understanding and informing my findings.

Abstract

In long-term deployments, autonomous robots based on Visual Teach & Repeat navigation face the challenges brought on by vast differences in visual information due to seasonal changes as well as changes due to human activities. In this thesis, we present an efficient way to store maps for Visual Teach & Repeat navigation which enables us to perform a spectral analysis on the data stored in the server to find the frequencies of the pseudo-cyclic processes of the surroundings and develop a model of the environment. The model helps in path planning to find a sequence of maps suitable for robust navigation for the current time and optimizes other criteria and hence try to mitigate the challenges brought on by seasonal changes and human activities in the long-term. The framework also provides a convenient way to switch from robust navigation to exploration mode when desired. We performed experiments to evaluate the functionality of the presented methodology and the results indicate that the planned path leads to robust navigation based on the current state of the environment.
Keywords: Map management, Path planning, Visual Navigation, Teach and Repeat, Centralised Database

Abstrakt

Při dlouhodobém nasazení čelí autonomní roboti využívající navigaci typu Visual Teach & Repeat výzvám, které přinášejí velké rozdíly ve vzhledu prostředí způsobené sezónními změnami nebo změnami způsobenými lidskou činností. V této práci představujeme efektivní způsob ukládání map pro navigaci Visual Teach & Repeat, který nám umožňuje provádět spektrální analýzu dat uložených na serveru, abychom zjistili frekvence pseudocyklických procesů daného prostředí a vytvořili jeho model. Tento model je dále využit při plánování cesty pro nalezení posloupnosti map, které optimalizují kriteria, jako je například délka trasy, robustnost průjezdu nebo prohledání prostředí. Provedli jsme experimenty k vyhodnocení funkčnosti představené metodiky a výsledky ukazují, že plánovaná cesta vede k robustní navigaci na základě aktuálního stavu prostředí. Představený rámec umožňuje zmírnit negativní dopad změn ve vzhledu prostředí na robustnost tohoto vizuálního navigačního systému.
Klíčová slova: Správa map, Plánování cesty, Vizuální navigace, Opakovaná Navigace, Centralizovaná databáze

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Autonomous mobile robots have expanded the possibilities of places where robots can be deployed. They are being deployed in environments which are hard to reach and possibly dangerous for humans. From being deployed to Mars, to an inaccessible cave for rescue efforts, to underwater missions for life and science exploration, they have significantly broadened the scope and possibilities of applications. Like humans who, to be able to navigate or explore any area, need a map of the area and know where on the map they are, the two foremost things that a robot needs to navigate any environment is to know where it is, and what the environment looks like. Once that is done, it can start moving around.

The main tasks that a robot needs to accomplish to navigate any environment are the following:

- **Localisation** : The robot needs to know where it is in the environment before starting out. The problem of finding out where it is in the environment is called as localisation.
- **Mapping** : The problem of creating a map of the environment is called as mapping. A robot needs to create the map of its surrounding before it can start working on its other directives. The map created by it is a description of the robot's surrounding. The description depends on the type of implementation. The robot can use this map to navigate around. As the robot explores the environment and encounters new unmapped areas, they are added to the map.
- **Path planning** : Once the surrounding has been mapped and the robot has localised itself, the route that the robot should take to go from one location to any desired destination is planned at this stage. The decision of which route to take may depend on many factors, one of the most common one being planning the shortest route between its current location and the destination.

- **Navigation** : After an optimum path is planned from the robot's current location to goal location, the planned path needs to be traversed to reach the goal. Steering the robot based on sensors' inputs and using appropriate control such that the planned path is traversed is known as navigation.

When initially starting out in a new environment, the robot may neither know its location nor may it have the map of the environment. Localisation can be with respect to the map that it creates, or it can be a global location. The problem of localisation (relative to its map) as well as mapping simultaneously is, aptly named as **SLAM (Simultaneous Localisation And Mapping)** problem. The problem of SLAM is a difficult and computationally expensive one due to point cloud matching with unknown correspondences, the number of which grows exponentially with time, and errors arising from control inaccuracies and sensor noise keep adding up over time leading to a decreased confidence in the maps created by it. The uncertainties arising from the sensor noise are usually addressed by probabilistic methods [1]. Nevertheless any solution to SLAM has to resort to some level of simplification to make it feasible.

Another method which presents a simpler way of navigation, **Visual Teach & Repeat** [2] has become quite popular in recent times. Developments in computer vision and deep learning have made the use of cameras a more practical option and opened up a lot of possibilities when it comes to using visual information from cameras. While SLAM can also use cameras, VT&R is reported to be more robust [3], and is a simpler system which reduces the number of points of failure in the system. It is also a cheaper system to implement in comparison [4].

In Visual Teach & Repeat methods, the robot is operated manually by a human, and as it is driven around, it records information from odometry, LiDAR etc., and stores the action commands, as well as live images from the cameras corresponding to the information collected by other sensors, during the teaching phase. During the repeat phase, the commands corresponding to the robot's position in the map are replayed by the robot to navigate the environment. The images captured by the cameras onboard the robot prove to be very useful. The action commands by themselves are not sufficient to follow the path that it is taught. Due to inaccuracies in control and noise in sensor data, just the starting pose and action commands prove to be insufficient in navigating the taught path. This is where the images captured by the robot during the learning phase are used. They are used for heading correction using any of the suitable computer vision methods [5]. Since images are used for heading correction, the need to explicitly localise is eliminated and the system becomes much simpler. In several studies, it has been shown to perform better in long-term navigation of autonomous robots [6], [3], [7], [8].

A practical large scale deployment of autonomous mobile robots can have a large fleet of robots operating in a shared environment. Teaching each robot all the paths can

be a very inefficient way of using VT&R. A way to share the knowledge gathered by a robot during its learning phase with the other robots in the fleet can help eliminate this inefficiency.

The thesis has the following two aims:

- **Develop a system to efficiently store and share the maps between the robots**: We propose a system for storing the knowledge gathered by the individual robots during their learning phase. One of the tasks of the thesis work is to implement improved map management using database services. The maps created by individual robots are uploaded to a centralised database, all of which are readily available to all the robots in the fleet. This approach presents the following benefits:
  - Knowledge sharing: In a multi-robot system, a new robot joining the fleet may not have to learn anything at all and just use the knowledge gathered by its teammates.
  - Smart decision making: The information present in the hundreds of maps gathered by the robots can be used to make smart decisions about path planning.
  - Computation offloading: Computation-heavy tasks can be done offline by more powerful computers to save time and space.

  A robot can create hundreds of maps during its teaching and exploration phases. All of which can be uploaded to the server and requested as needed. The maps and all the corresponding information related to the maps and the environment, which is extracted from the maps, are stored in the database for efficient use of the knowledge .
- **Develop an algorithm which finds an optimum sequence of maps to be served to the robot**: Just storing all the maps in a server is not enough. There has to be a way to decide which map should be served to the robots, for a given start and end point, when it reaches a certain location in the environment. A new robot joining the team has no information about the environment. All it knows is that it has been asked to go from a starting point to a goal point. Out of the hundreds of maps present in the database, decisions need to be made about finding an optimum sequence of maps for the robot. For long-term deployment, various criteria need to be optimised for before feeding the robot a sequence of maps that it can use for navigation. One of the most important factors is time since VT&R uses images, and images captured by the cameras can differ vastly depending on the time of the day/year [9] [10]. Since Visual Teach & Repeat relies heavily on images to correct the heading, it needs to be made sure that

the images in the maps being chosen are suitable. Once an optimum sequence of maps is decided, it can be served to any robot given its location and time in the environment.

The thesis is structured into five chapters. The following chapter gives an overview of the state of the art relevant to the topic of the thesis. The third chapter presents the methodology which explains in detail all about the tools, mathematics and development used to achieve the aforementioned goals of the thesis. The fourth chapter presents the experiments performed and shows the results of the implementation. The fifth chapter presents the conclusion and future recommendations.

Chapter 2

State of the Art

In this chapter, we present the state of the art relevant to the work of the thesis. Specifically we take a look at Visual Teach & Repeat, challenges faced by mobile robots in long-term deployment and the solutions available for them. We then take a look at the current strategies for handling multiple maps of the same location, and how the information gathered by robots is shared among each other for improved localisation and navigation of autonomous mobile robots. Then we discuss some of the current path planning methods, one of which is used in the current work.

2.1 Visual Teach & Repeat

The work of this thesis specifically aims for long-term deployment of robots based on Visual Teach & Repeat methods. Teaching a robot a certain task manually, which then can be replayed by the robot has been quite popular for manipulators in the industry. The same idea can be applied to mobile robots in VT&R methods. The robot is manually operated by a human during its teaching phase. The robot records the action commands given to it along with the corresponding data gathered from other sensors like odometer, LiDAR etc, along with the visual information. During the repeat phase, the robot can simply replay the action commands, corresponding to the readings from other sensors recorded during teaching phase. Due to noise in sensor data and uncertainty in the starting pose, just the action commands are not sufficient at all to repeat the path. This is where the images captured by the robot during the learning phase prove to be useful.

Methods proposed in [6], [11] rely on construction of metric maps. Metric maps place the features captured by the visual system on exact location in the Cartesian space. There are other methods which do not depend on explicitly localising the robot, or

placing the features of the surrounding in the Cartesian space. Method in [8] showed that explicit localisation is not needed for teach and repeat methods. They presented a mathematical proof showing that the robot converges to the taught path even when the visual information from the cameras are used to correct heading. But the method worked only on polygonal maps. Method presented in [2] was computationally efficient, did not require any camera calibration and worked with any shape of path. Works of [12], [13] also rely on calculating the horizontal difference of the features to steer and follow the taught path closely.

For long-term deployments, calculating the difference in heading based on features can lead to unintended results if the images being matched do not have enough correspondences. Two images from the same location can look drastically different based on the lighting, camera parameters etc. Seasonal changes also lead to a huge difference in the structure of the environment. For example, two images from the same location, with the only difference being that one of the images might be from a snowy winter while the other might be from a sunny summer day can look vastly different. Calculating feature correspondences using hand-crafted feature descriptors can prove to be rather insufficient in long-term deployments. Approaches that employ neural networks have shown that learned descriptors can perform better than the classical hand-crafted ones [14], [15], [16], [17]. Ubiquitous presence of powerful hardware in a small form-factor has made it possible to use power-hungry neural networks even in mobile robots which are usually restricted in their use of resources. Methods presented in [18], [5] use convolutional neural networks to extract feature descriptors. The method presented in [5] uses a fully convolutional neural network and achieves state of the art performance in calculating horizontal displacement between two images, which is used to steer the robot on a prerecorded path.

2.2 Long-Term Deployment and Spectral Analysis

Long-term deployment of robots presents a difficult problem when it comes to autonomous navigation. Environments change over time, either due to man-made causes or due to natural causes. Static maps created by a robot are ultimately going to hit their limit to be able to describe the environment in long term. To overcome this limitation, maps need to continuously evolve with time. Works of [19], [20] propose methods that keep the features which stay the same over long periods of time to deal with the dynamic nature of environment. Method in [21] proposes continuously adapting the maps with time, adding and removing new features as required. They propose several map management strategies and show that modelling the cyclic nature of the changes in the environment to predict the appearance of features performs better than other strategies.

Work of [9] also shows that modelling the environment improves the performance in the long run. They propose that any process in the environment is pseudo-cyclic in nature. For example, the processes in a parking lot of a factory are periodic, the number of cars present in the parking lot is going to be almost the same on every weekday during office hours, whereas it is going to be almost the same after hours and on every weekend. The method proposes using Fourier transform of each environment state to calculate the frequency spectra. Since the robot might observe the environment at any random time, standard Fourier transform can not be used to calculate the frequency spectra since it assumes that the samples are uniformly spaced in time. To deal with this problem, non-uniform Fast Fourier Transform is used to calculate the frequency spectra [22]. The frequencies and their corresponding amplitudes and phases thus calculated are used to describe the underlying pseudo-periodic processes of the environment and can be used to predict the future state of the environment.

While [9] is applicable only to binary state representations and could model only the probability of only a single state over time, [10] presents a method to model the spatio-temporal continuous representation of the environment and can predict the probability distribution of a given random variable at a particular time and location. The repetitive nature of the environment is represented by projecting the time into a set of circles where the circle is derived from frequency of change calculated using the method presented in [9].

Other approaches that deal with the robustness issues related to varying lighting conditions use pre-trained neural networks. Method in [23] uses pre-trained networks to predict visual features which are used in VT&R for reliable navigation in varying lighting conditions. Method in [16] also proposed a trainable feature descriptor based on evolutionary methods and binary comparison tests and showed that it performs better than hand-crafted feature extractors in long term deployments with varying lighting conditions.

2.3 Expertise Sharing

The work presented in this thesis aims to implement a system where the robots can share their expertise with each other. Work in [24] presented a decentralised versioning system for maps where each robot updates its belief about the environment, which the robots can use as a "pricing" strategy and decide about the credibility of the information based on these "prices". The method presented though doesn't take into account the modelling of the environment itself. The maps created by Visual Teach & Repeat are big in size, so storing hundreds of maps on all the robots probably is not feasible in our case. Our work is not about a decentralised system of sharing expertise, we propose a centralised system which stores maps, which in turn provides the benefit of offloading

computation to a central server hence time is not wasted by resource-limited robot in computing when it is online. But the idea of assigning a "price" to the belief of every robot serves as an inspiration.

Method presented in [25] on the other hand collect "experiences". When a new place is visited the output of visual odometry is saved. When the place is visited again, the live stream of images from the camera are compared with the saved "experience" which helps in localisation. An unsuccessful localisation means that the saved "experiences" are vastly different visually from the the live feed and the output of the visual odometry is then instead saved as a part of "experience" which might help in localisation in the future. While the new "experience" is being saved, localisation is also attempted continuously and if at some point localisation is successful, saving is stopped. Method in [26] proposed a similar approach where they create non-metric topological graphs called as "experience map". However, unlike [25], here a node is referred to as an "experience". The node itself contains its visual information of the surrounding and other relevant information needed for localisation. When the robot reaches a node and the visuals from the live feed differ by a lot from the saved visual of that node, the new visual information is added to the node for future localisation attempts.

Method presented in [27] proposed creating a full-scale map by fusing the data recorded during map creation stage on an off-vehicle computer. From this full-scale map, a "summary map" is created which is feasible enough for navigation when the robot is online. The "summary map" is created by using a scoring function that determines the usefulness of landmarks and decides which features would lead to robust navigation. Work of [28] proposes a similar method where they perform offline localisation using the dataset available at the time. If the localisation does not perform well, that means the available dataset does not have enough landmarks for robust navigation so new landmarks are added as part of a new dataset. If the localisation does perform well enough then no new landmark is added to the map, though the new information encountered during this session is still added to the map since it proves useful in making a distinction between which landmarks are important and which are not. The importance of a landmark for navigation is decided by a ranking function. To keep the size of maps in check, they apply map summarisation techniques.

2.4 Path Planning

The work presented in this thesis represents an environment by a topological graph, where each edge is a map created by VT&R and each node a place in the environment. There are a number of path finding algorithms that can be used to find an optimum path between any two nodes of a graph.

Rapidly-exploring random tree (RRT) [29] is an algorithm that creates a tree-based data structure which is used for path planning. It is a probabilistic path planning algorithm that randomly samples the search space. The tree starts from a root, and random samples are drawn from space around it. If there is a feasible path between the sample and its nearest neighbour, then the connection is made. One of the advantages of RRT is that it can be adapted to be used in wide variety of path planning problems [29].

The A^* algorithm is one of the most popular path finding algorithms that finds the shortest path between two points in a graph using a heuristic function to guide the search towards the goal. A cost is evaluated for a node, which is based on the sum of costs of all the nodes from the starting node to the current node and a heuristic cost that estimates the effort to go from the current node to the goal node. The path found by the algorithm is optimal as long as the heuristic cost doesn't overestimate the effort it takes to reach the goal [30]. We use A^* algorithm in this thesis since it seems suitable and simple enough for the task.

Chapter 3

Methodology

This chapter outlines the approach and techniques employed to achieve the goals of the thesis. The first step is to setup a database and implement a way to efficiently manage all the maps created by any robot working in an environment. Section 3.1 details how this is achieved. Maps created by VT&R contain all the relevant information needed for a robot to navigate during repeat phase. Metadata of all the maps, which is extracted, plays a key role later on. The map, along with the metadata, is then uploaded to a centralised database. As the map is uploaded, the graph of the environment is updated, with the new map being a new edge between the nodes mentioned during the upload process. The graph is constantly updated as each new map is uploaded. Section 3.2 details how all the data uploaded to the database is used to develop a model of the environment using the metadata of all the maps present in the database. The model of the environment is recalculated and updated with each addition of a new map. Section 3.3 details how all the computationally heavy task done by neural networks is offloaded to the server and how Fourier analysis is used to calculate the periodicities of the cyclic processes of the environment, results of which are used in the model of the environment. Section 3.4 talks about the established optimisation criteria that are used to decide which path should be taken by the robot for navigation. When a robot reaches a location in the environment and wishes to find a optimum path to its destination, a cost is calculated for each map of the environment stored in the database. Using the cost as a heuristic for the path finding algorithm, an optimum path is calculated between its current position and the destination. This is discussed in section 3.5.

The sequence of maps corresponding to the optimum path is downloaded to the robot from the database which can be used for navigation. The maps generated during learning phase which would lead to a robust navigation, or exploration, are thus used. A schematic showing how each component of the process is connected to each other is shown in Figure 3.1.

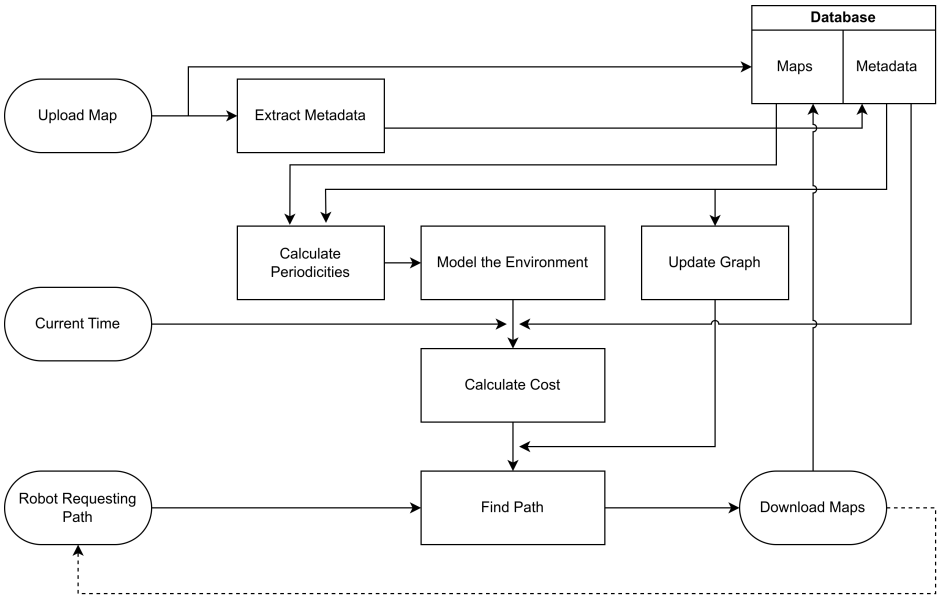


Figure 3.1: Schematic showing the whole pipeline

3.1 The Database

Ability to share the maps recorded by Visual Teach & Repeat with other robots in the fleet so that they don't have to learn the same environment, if other robots have already done so, is required. To enable the utilisation of all the maps/paths taught to every robot that operates in an environment and find an optimum path for a robot to follow, some way to store all this expertise in a way that can be accessed by any robot in the environment is needed. Even in the case when just one robot operates, we might have several maps for the same path and may want to store it off the robot for various reasons, some of which are listed below:

- Save disk space on an already resource starved mobile robot.
- Offload computation-heavy tasks to a more powerful server.
- Saving time on computation done by the robot when it is online by pre-computation that can be done by the server.
- Any number of robots can join or leave the fleet in the future without affecting the functionality for any other member.

To be able to share expertise/maps among the robots, the maps created by individual robots need to be stored in a centralised database. The ubiquitous presence of high-speed WiFi networks makes using a centralised server feasible for storing and fetching

the expertise of the robots. We implement a client-server model, visualised in Figure 3.2, where each robot in the fleet acts as two way node which can upload and download maps from the server.

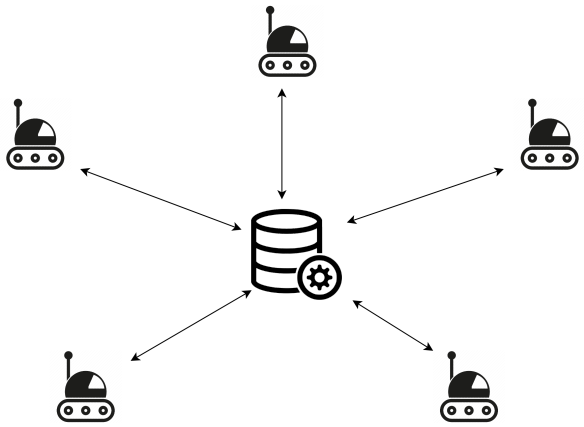


Figure 3.2: A client-server relationship between the robots and the database

The choice of the database service that should be used is a critical first step. The maps generated by the Visual Teach & Repeat stack contains JPG images, rosbags, neural network representations of the images, and zipped files. The database service needs to be open-source and suitable for unstructured data which can store data as objects. Choosing a wrong service has the potential to lead to problems down the line. MinioDB [31], which is a high performance, S3 compatible object storage, proves to be suitable for the task. It is easy to use, versatile when it comes to the type of data it can handle, and provides an intuitive dashboard which the user can use to explore the files in the database.

3.1.1 Metadata Extraction

As shown in Figure 3.1, when a map is uploaded to the database, its metadata is also extracted and uploaded in a separate bucket in the database. When a path is requested by a robot, instead of doing computations on all the maps, just the metadata of all the maps is used to do all the work presented in the following sections which ultimately helps in finding an optimal path for the given time. Hence, the metadata plays a key role in enabling the entire process. Clearly, just having all the maps stored in some database is not enough for navigation, specially when the goal is to find a robust path for long-term deployments. One of the biggest advantages of this framework is that the decision making related to which maps the robot needs can be done offline leading to reduced computation time for the robot when it is online.

We propose that the following metadata should be extracted during upload:

- **Timestamp of the map**: As discussed before, a model of the environment is created. A cost is calculated for a map by using the model of the environment based on the timestamp of the map. The cost is one of the criteria which is used in the heuristic function of the path finding algorithm. This is one of the most important metadata that is used to find an appropriate sequence of maps for robust visual navigation.

- **Length of the path**: Length is also one of the optimisation criteria which is used in finding an optimal path. Hence this needs to be extracted so that it can be used later by the path finding algorithm.

- **Name of the map**: Each map is given a unique name. It also acts as a tag for the edge in the graph representation of the environment. The name of the map is used as a reference to the map and facilitates the fetching, uploading and deletion of maps from the database as and when required.

- **Neural Network representation of the images**: The current implementation of VT&R framework uses neural networks to represent the images captured by the cameras. While this particular metadata is not utilised in the current work, it might prove to be useful in future to extend functionality.

- **Action commands**: The action commands given by the human operator during the teaching phase of VT&R, which are used during the repeat phase for navigation, are stored as metadata.

- **Times**: The timestamp corresponding to each action command.

- **Distances**: The distance corresponding to each action command which is utilized during navigation.

- **Starting node of the map**: The human operator enters a starting node name when uploading the map. The name is used in creating and updating the graph for the environment.

- **End node of the map**: The human operator enters an end node name when uploading the map. It is also used in creating and updating the graph for the environment.

As a map is uploaded, the graph for the environment is updated by adding a new edge and nodes corresponding to the current map.

3.1.2 Contents of the database

There are mainly two buckets in the database. One is for simply storing the maps created by VT&R as zipped files. The other is for storing the metadata. Figure 3.3 demonstrates how information is stored in the database. The following details the contents of each bucket:

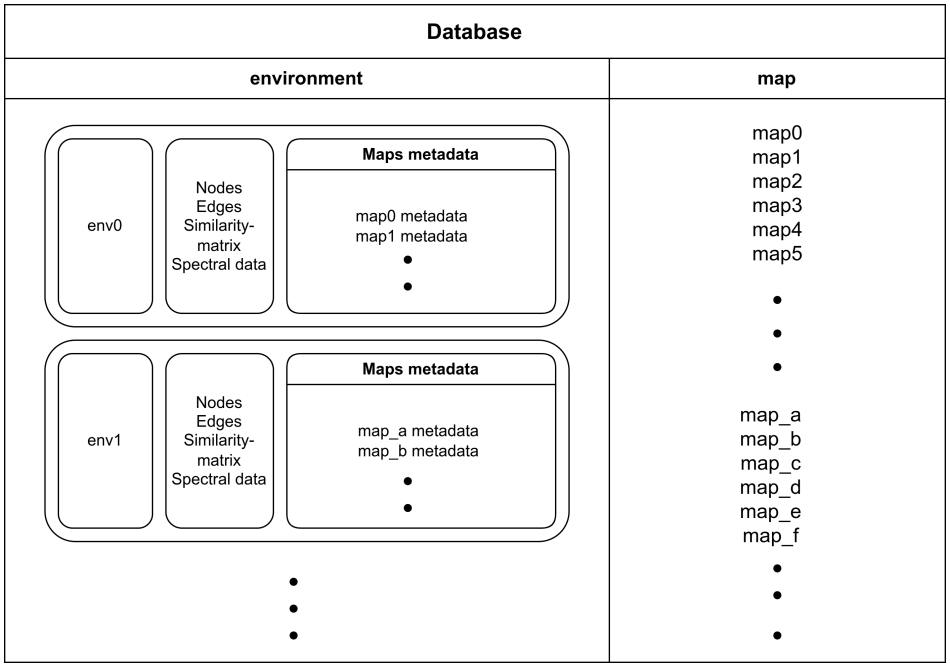


Figure 3.3: Buckets' contents in the database

1. **environment** : This bucket contains all the relevant information (including but not limited to the metadata of all the maps created) about all the environments that the robots are deployed in. Each object in the bucket contains the following data:

    - **Name**: During upload, each environment is given a unique name which helps in keeping track of which map in the map bucket belongs to which environment.
    - **Nodes**: This contains information about all the nodes belonging to the environment. It is used to maintain and update the graph of the environment, and ultimately in path planning.
    - **Edges**: This contains information about all the edges belonging to the environment. It is used to maintain and update the graph of the environment.

- **Similarity matrix**: It is a matrix that contains information about the similarity in visual information between each map of an environment. It is used to calculate the periodicities of the processes in the environment using a neural network. How this is done is discussed in section 3.3. This is stored as a metadata of the environment so that model of the environment, which uses the periodicities, can be updated with each new map that is added to the environment. The model is continuously evolving as more data is added to the database. Calculating the similarities between maps using neural networks is the most time consuming part of the whole process. This is where the results of the computation are stored, thereby reducing computation time when the robot is online since the results can just be directly used during navigation.
- **Spectral data**: This contains the results of the spectral analysis done on the similarity matrix. As similarity matrix gets updated as more maps are added, so are the results of the spectral analysis, which updates the model of the environment.
- **Maps metadata**: As discussed in section 3.1.1, when a map is uploaded, the metadata corresponding to that map is added to the environment object corresponding to environment to which the map belongs. This contains the metadata of all the maps that belong to this environment.

Every time a new map is uploaded, the object in the bucket corresponding to the environment is updated.

2. **map** : All the maps created by a robot, for all the environments that it is deployed in, are uploaded as zipped files to this bucket without making any changes to it so that the integrity of the map is retained. The sequence of maps outputted by the path planning algorithm during navigation provides the information about which maps should be downloaded to the robot. The zipped files are downloaded and extracted which are then used for navigation by VT&R framework.

3.2 Model of the Environment

The accuracy of navigation for the Visual Teach & Repeat depends on how similar the live feed of the camera is to the map created by the robot during teaching phase. Method in [2] calculates the horizontal shift in pixels of the live feed vs the stored map. If the visual information of the stored map is completely different from the live feed, the robot cannot calculate the horizontal shift and hence is not able to correct its heading during navigation. Even if the live feed is not completely different from the

images in the stored map, there could still be a vast difference between the both. The images in a map that are created during day time can be significantly different from the live images of night time. This can lead to poor results in calculation of horizontal shift between the two images which results in a poor performance during navigation. Though there are methods, like the one presented in [5], that prove to work well in finding horizontal shift in the images which are taken even at different seasons and times of day. It is evident that the day and time on which the robot is deployed plays a significant role in determining the performance of the method in long-term deployments if the navigation is based on VT&R. If the robot has maps which look visually similar to the live feed of current time, it performs well. One naive solution would be to just compare the images of all the maps in the database to the live feed and just select the one that is the most similar to it. The image similarity comparison is done using a neural network. Computing the likelihood of each map's images to the live feed is computationally very expensive, the robot would spend a lot of time just standing and computing which in turn would cost money. On top of that, the database containing the maps from all the robots will grow in size with time. It can contain hundreds of maps, from different times of the day and different seasons with varying illumination levels, for all the environments that the robot is deployed in. The fact that time plays an important role in determining the performance of navigation can be exploited.

Methods in [9] and [10] propose that processes in an environment can be pseudo-periodic in nature. For example, 09:00 of today is visually similar to the 09:00 of yesterday, 1st January of this year is visually similar to the 1st January of the last year. The activities caused by humans are also pseudo-periodic. The number of people at a place, for example, could be the highest during office hours, and low during weekends, both of which are periodic in nature. Such patterns are often tied to periodicities of 12 hours, 24 hours, 1 week, 1 month, or 1 year, at least for outdoor environments.

Using the fact that processes in an environment can be periodic, we propose a mathematical model of the environment, inspired by [9], that can help in selecting a sequence of maps that is the most suitable for the given current system time, without having to compare each map's visual information with the live feed's visuals using a neural network. Modelling the environment leads to an improved performance in long term deployment of robots [21], [9]. The model can help in predicting what the state of the environment will be in future at any given time. When the robot reaches a certain location and is waiting for a path for it to navigate, out of the hundreds of maps in the database, a sequence of maps which leads to a closest likely match between the live feed and the visual information stored in the aforementioned maps should be fetched from the database. Given that the periodicities of the processes in the environment are known, a cost is assigned to every map in the database using the model, with respect to the current system time. The cost calculation is done at the time when the robot requires a sequence of maps to go from a starting point to a goal point. The model

presented here helps select the map that is the most likely to be visually similar for the current time of the environment, with the added benefit that other criteria can also be optimised for, while being computationally efficient.

A sinusoidal function is a good candidate to model the cyclic nature of environment, frequency and phase shift of which will be calculated in section 3.3. The timestamp of each map, that is extracted and uploaded to the database along with the map, is used to calculate a cost and it takes into account the cyclic nature of the environment, for each map. The cyclic cost, c_k^c, is assigned to the k^{th} map using equation 3.1.

$$c_k^c = \frac{1}{2} - \frac{1}{2N}\sum_{i=1}^{N} w_i cos(\frac{2\pi}{T_i}(t - s_k) - \phi_i), \tag{3.1}$$

where

- N is the number of prominent periodicities of the environment. It is a parameter that is decided by the user,
- w_i is the weight for i^{th} periodicity of the environment,
- T_i is the time period corresponding to i^{th} periodicity,
- t is the current system time,
- s_k is the timestamp of k^{th} map,
- ϕ is the phase shift of the i^{th} periodicity.

For an environment with daily and weekly periodicities i.e $N = 2, T_1 = 24\,hours, T_2 = 168\,hours$, Figure 3.4 demonstrates the cyclic cost calculated for the current system time using equation 3.1 for maps with different timestamps. ϕ for each periodicity is assumed to be zero, w for both the periodicities is assumed to be the same.

While equation 3.1 captures the cyclic nature, it does not account for the permanent changes that occur with time. Every environment changes slowly with time. Construction of a new building, installation of new furniture, plantation of a new tree are just some of the examples of permanent changes. So even two maps which were created at the same location and similar times of the day but in different years are most likely not going to have the same similarity in visual information in long-term. A map created five years ago may not have the same cost as a map created one year ago. To account for this trend, we propose incorporating an exponential cost as well, called as "trend cost" c_k^τ, calculated for the k^{th} map using equation 3.2. This assigns a higher cost to the maps which are likely to have been created before any permanent changes occurred.

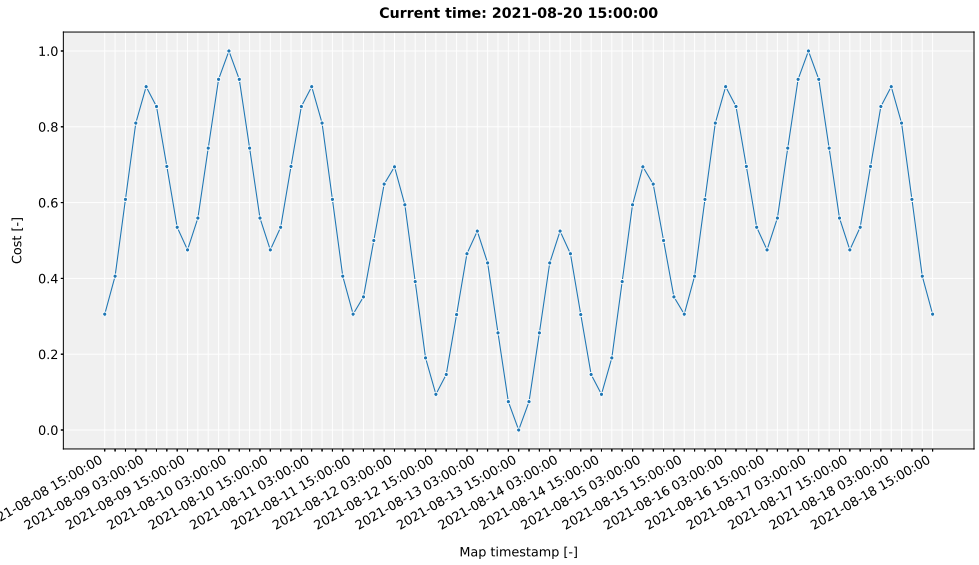


Figure 3.4: Cyclic cost of each map calculated for the current time 2021-08-20 15:00, given that the processes in the environment are periodic with daily and weekly periodicities. As expected, the cost of the map from the same time exactly a week ago is the lowest, followed by the maps which are exactly 24 hours apart from that day.

$$c_k^\tau = 1 - e^{-a(t-s_k)}, \tag{3.2}$$

where t is the current system time, a is a parameter that determines the rate at which permanent changes occur in the environment, s_k is the timestamp of k^{th} map.

The lower the value of parameter a, the slower the exponential cost c_k^τ rises with time. Figure 3.5 demonstrates the dependence of c_k^τ on a. The value of a should be estimated from the data present in the database. We do not estimate this parameter in the present work.

Hence the similarity cost, C_k, for each map incorporates both the cyclic cost and "trend cost" and is given by equation 3.3.

$$C_k = c_k^\tau(t, s_k, a) + c_k^c(t, s_k, T, w, \phi, N), \tag{3.3}$$

where t is the current system time, s_k is the timestamp of k^{th} map, c_k^τ is the trend cost given by equation 3.2, a is a parameter that determines the rate at which permanent changes occur in the environment, c_k^c is the cyclic cost given by equation 3.1, T is the time period of a process of the environment, w is the weight for the periodicity, ϕ is

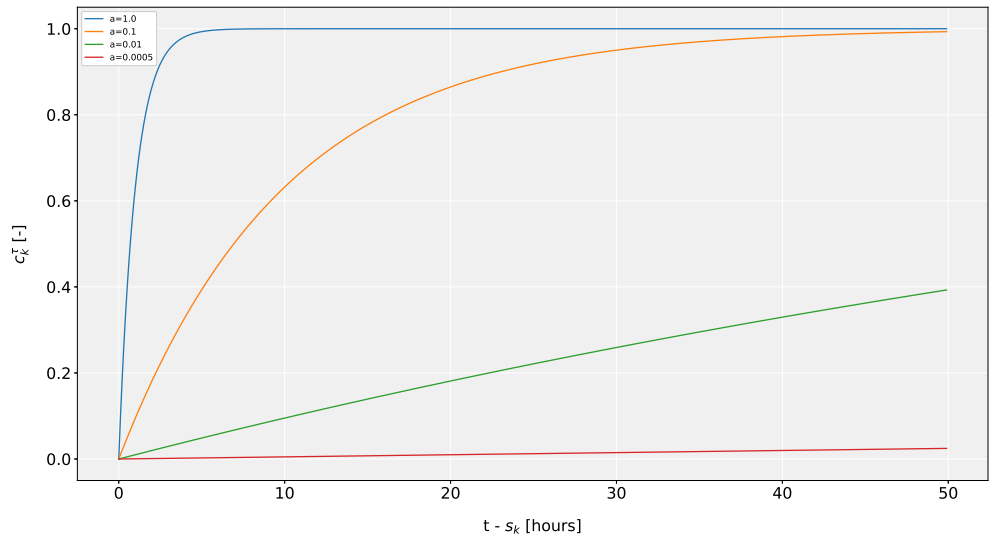


Figure 3.5: Trend cost vs time difference between the current time and the map's timestamp.

the phase shift of the periodicity. N is the number of prominent periodicities of the environment. Calculation of parameters T, w, ϕ, N is demonstrated in section 3.3.

3.3 Parameter Estimation

The cost for each map calculated by the model of the environment given by equation 3.3 depends on the following parameters:

- T: Time period of a periodic process of the environment.
- w: Weight (magnitude) of the frequency corresponding to the time period T.
- ϕ: Phase shift corresponding to the time period T.
- N: The number of prominent frequencies present.
- a: Parameter that determines the rate of permanent change in the environment. This is not calculated in the present thesis.

For a general outdoor setting, the processes repeat with some periodicities that makes common sense to assume, the common periodicities being that of 24 hours, 1

week, 1 year. But for a general application, periodicities cannot be assumed. If the robot is deployed inside a factory that runs 24 hours a day, the environment may not follow the daily and weekly periodicities. The database stores all the maps and the meta-data of the environment. Everything that can be known, including the visual information, about the environment is present in the database. Converting the visual information into a time series and performing spectral analysis on it should output the frequency content of the processes of the environment. Timestamp of each map is readily available from the metadata, just the visual information of each map needs to be converted to a single number to construct a time series. The following subsection 3.3.1 details how this is done.

3.3.1 Similarity Matrix

Method presented in [32] computes the likelihood, $\mathcal{L}$, of similarity between two images, using equation 3.4. The more the two images look alike, higher is the output value and vice versa.

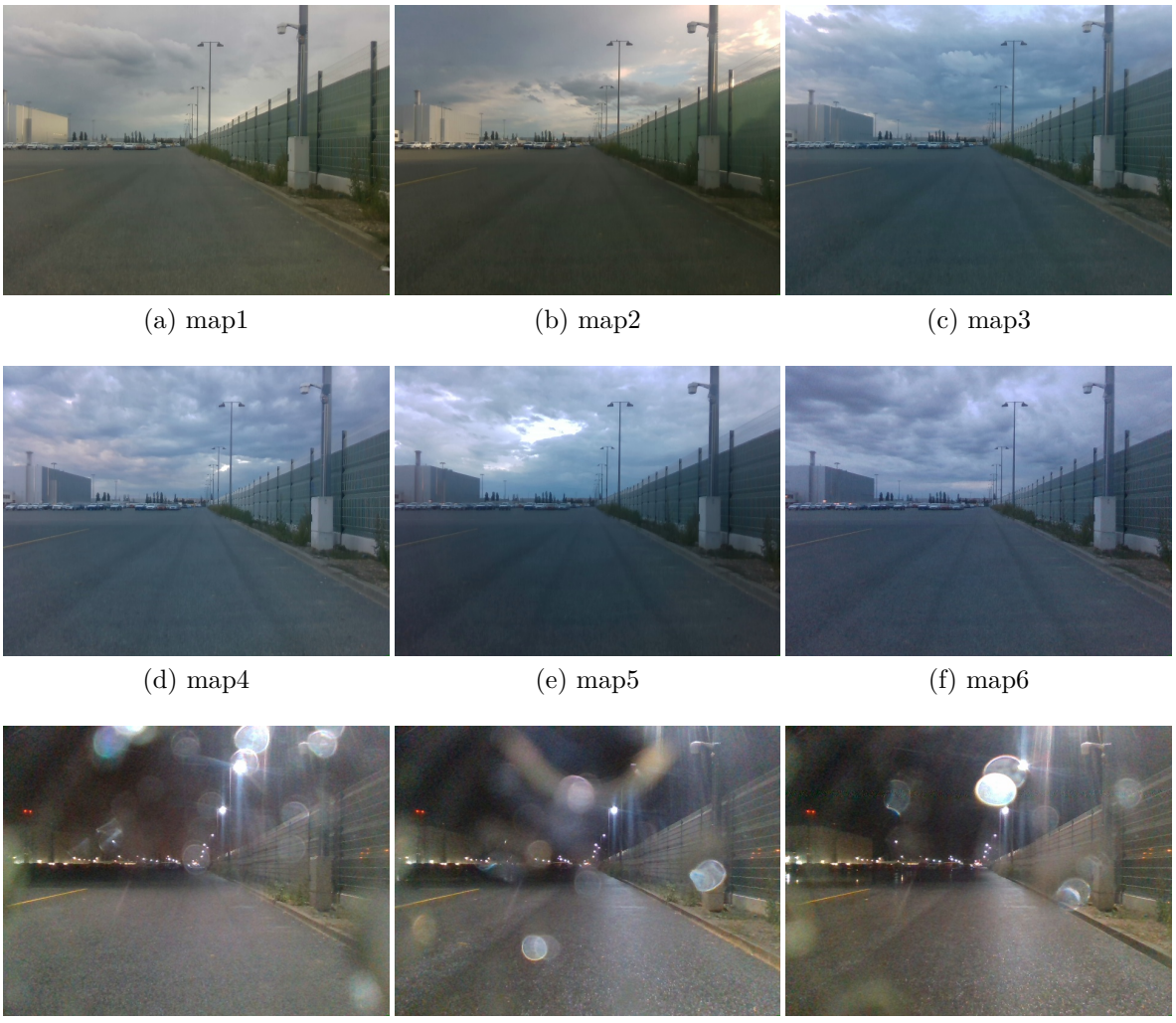
$$\mathcal{L}(I_1, I_2) = \sum_{i=1}^{c} \sum_{j=1}^{h} \sum_{k=1}^{w} R(I_1)[i,j,k] \cdot R(I_2)[i,j,k], \tag{3.4}$$

where $R(I)$ is the representation of image I by the neural network, c, h, w are the number of channels, height, width of $R(I)$. An application of this is shown for images in Figure 3.6 with corresponding calculated likelihood values given in Table 3.1. The more the images look similar to each other, the higher is the number.

I_1	I_2	$\mathcal{L}(I_1, I_2)$
map1	map1	248.6
map1	map2	217.4
map1	map3	214.2
map1	map4	209.0
map1	map5	211.6
map1	map6	205.2
map1	map7	145.5
map1	map8	144.0
map1	map9	144.8

Table 3.1: Likelihoods calculated for the images in Figure 3.6 using equation 3.4

If an image from a specific map from a location is compared with an image from another map from the same location, equation 3.4 can be used to calculate the similarity in visual information of the two maps. Following the same logic, calculating the



(a) map1 (b) map2 (c) map3

(d) map4 (e) map5 (f) map6

(g) map7 (h) map8 (i) map9

Figure 3.6: Images used for likelihood calculation. Pictures taken from maps created at Škoda factory.

similarity of an image from a specific map with select images from each map, along with the fact that timestamp of each map is known, a time series can be constructed. The resulting time series indicates how the similarity between a specific map and all the other maps varies with time. Calculating the time series by comparing the second map with all the other maps will result in another time series, which, given enough data is available in the database, should result in a time series that has the same periodicities as the first one, except with different phase shifts. In this way, a similarity matrix S_e for a given environment can be constructed, where the elements of the i^{th} row indicates the similarity between the i^{th} map to every other map in the environment, the diagonal values being the highest in every row since an image is the most similar to itself.

As can be seen from Table 3.1, likelihood values outputted by equation 3.4 can vary quite in range. Each row of the similarity matrix can be normalized using the softmax function given by equation 3.5.

$$\sigma(z)_i = \frac{e^{\beta z_i}}{\sum_{j=1}^{K} e^{\beta z_j}}, \tag{3.5}$$

where z_i is the i^{th} element of a row, K is the total number of elements in the row, β is a parameter that controls the sharpness of the output probability distribution $\sigma(z)$. For the nine images shown in Figure 3.6, the similarity matrix represented as a heat map is shown in Figure 3.7. The heat map for the nine images shows visually which images look alike each other and which do not. It is divided into four major regions in this example: the lighter regions in the top-left and bottom-right, and darker regions in top-right and bottom-left. For a specific row, the images corresponding to the lighter regions are not likely to look similar to the images corresponding to the darker region. The images corresponding to a specific lighter region are likely to be more visually similar to each other.

The similarity matrix provides a way to construct a time series for the visual information of the maps. Instead of assuming that the periodicity of the environment is a fixed value, this provides a general way of finding the actual periodicities of the environment which can be used by the model of the environment which is given by equation 3.3, to fetch time-appropriate maps for the robot. The similarity matrix is updated every time a map is uploaded to the environment, hence the model of the environment is continuously evolving. Even if the periodicities of the processes that take place in the environment change, the model adapts itself.

Calculation of the similarity matrix is the most computation-heavy part of the entire pipeline. Instead of performing this computation when the robot is online, it is done offline and the pre-computed results are used to decide the suitability of a map using the model of the environment, given by equation 3.3, which saves precious time.

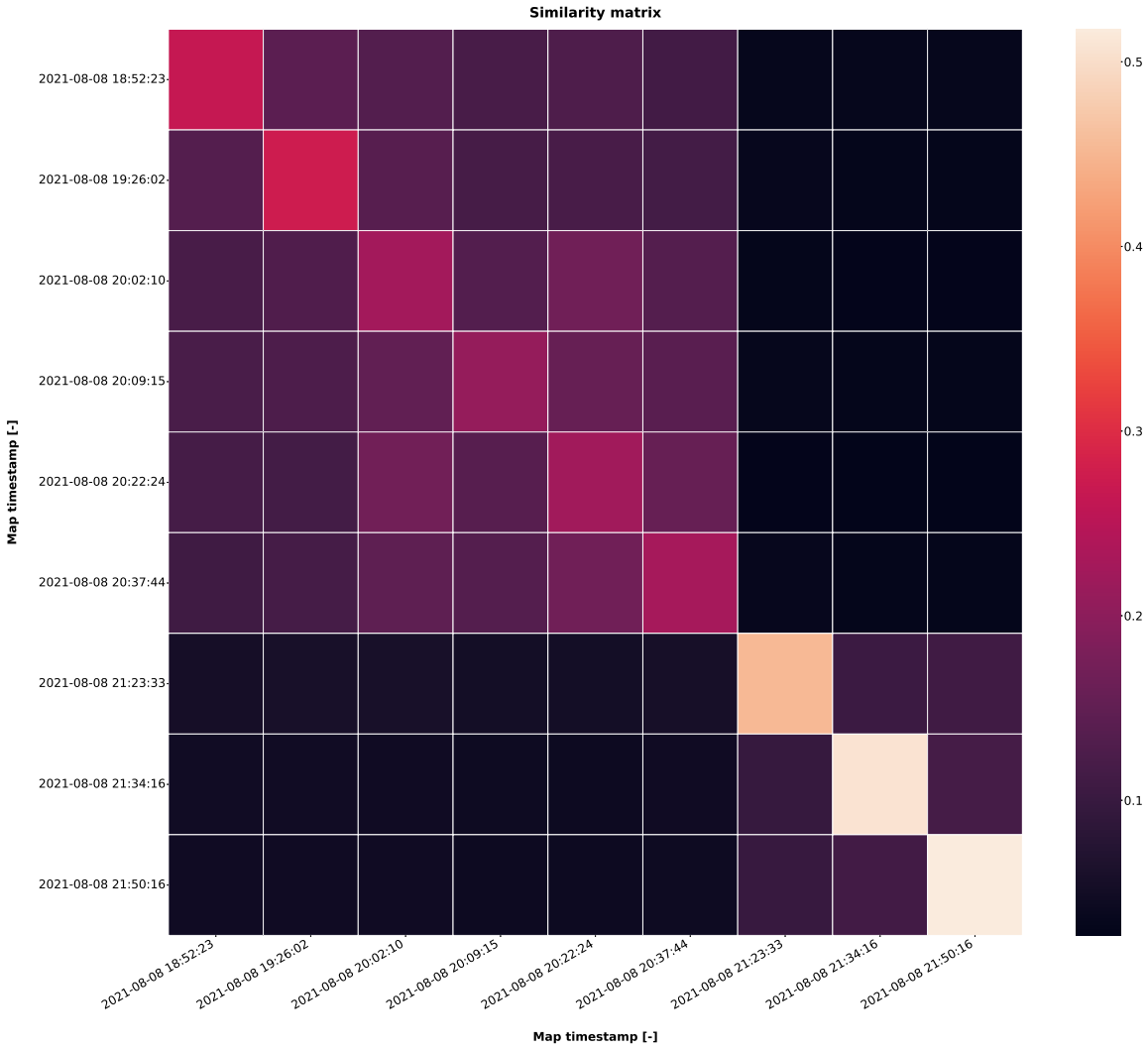


Figure 3.7: Similarity matrix calculated for the nine images from Figure 3.6. As the time difference increases between the maps, the visual differences increase, hence values for similarity gradually decrease as one moves from the first map to the last map.

3.3.2 Spectral Analysis

The similarity matrix calculated in section 3.3.1 is used to find the periodicities present in the data. Time series are generated from the similarity matrix by using the difference in timestamps of its elements as the x-axis values and the element values as the y-axis values for each data point in the time series. For each element A_{ij} in the similarity matrix S_e, the x-axis value for the time series corresponding to i^{th} row is:

$$x_{ij} = s_i - s_j,$$

where, s_i is the map timestamp corresponding to the i^{th} column of the similarity matrix, s_j is the map timestamp corresponding to the j^{th} column of the similarity matrix.

The y-axis value for the time series corresponding to x_{ij} is:

$$y_{ij} = A_{ij},$$

where A_{ij} is the value of the element corresponding to the i^{th} row and j^{th} column of the similarity matrix. For each element A_{ij} of a row i in S_e, a data point (x_{ij}, y_{ij}) in the time series is created. Each row of the matrix is a time series, just with a different phase shift. Figure 3.8 shows a time series constructed from the first row of a similarity matrix for an environment with maps recorded over a duration of three days.

Fourier analysis of the time series should be able to help us find the frequencies present in the data. But the standard Fourier Transform requires that the samples should be uniformly spaced in time. A large fleet of robots deployed in an environment might observe it at any random time. The maps created by robots using VT&R in a real-life scenario are almost never uniformly spaced in time from each other. An example of it can be seen in Figure 3.8. To deal with this problem, [9] uses the method used in [22] which uses a non-uniform Fast Fourier Transform (FFT) to calculate the frequency spectra.

The output of the non-uniform FFT employed on the time series is equal to the parameters w_i, T_i, ϕ_i and the number of prominent frequencies that are selected is equal to N in equation 3.3. As more maps are added to the database and the similarity matrix gets bigger, time series becomes more dense leading to a better spectral analysis. With this we have everything that is needed to choose time-accurate maps for robust visual navigation, at least when it comes to dealing with the cyclic nature of any general environment. Other criteria are discussed in the next section 3.4.

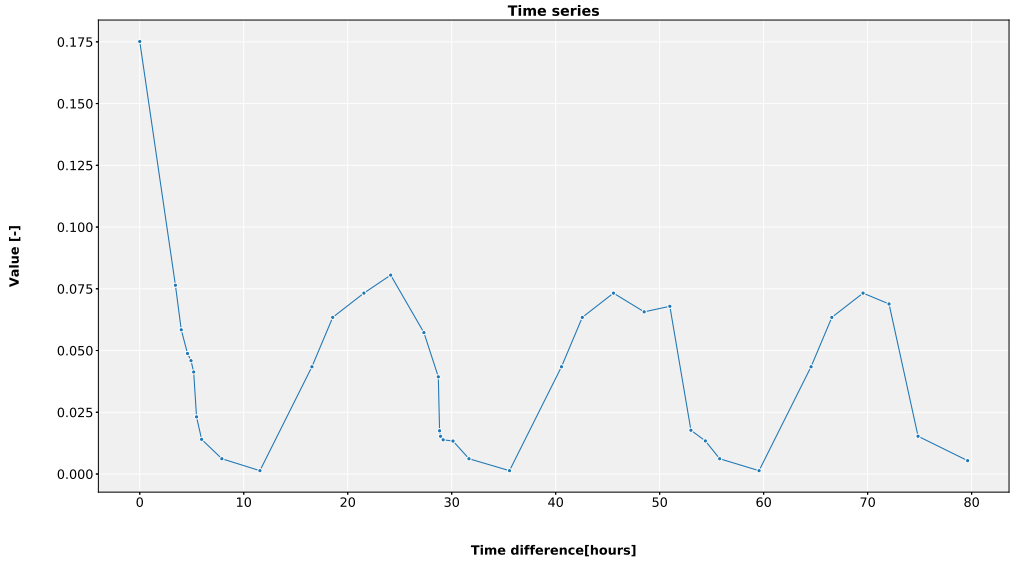


Figure 3.8: Time series constructed from the first row of a similarity matrix calculated for an environment containing 39 maps which are non-uniformly spaced in time over a period of 3 days.

3.4 Optimisation criteria

3.4.1 Criteria

Sections 3.2 and 3.3 exploit the visual information of maps created by VT&R system to find the periodicities of the pseudo-cyclic processes of any environment in an attempt to choose maps that are the most likely to lead a robust navigation of the environment. While this is an important criteria to take care of when planning for an optimal path between any two node in the environment, depending on the application we may want to optimise for other criteria.

We propose the following criteria:

- **Map similarity,** C: The similarity between visual information of a map captured by the robot during teaching phase and the visuals from live feed determines how closely the robot can navigate the taught path. The similar they are, the better the heading correction of the robot. So, this is one of the key criteria that plays an important role in VT&R navigation.
- **Distance travelled,** d: Distance travelled by the robot is almost always one of the criteria during any path planning strategy. Shorter path is usually preferred as

it likely to be energy and time efficient. Given that all other criteria are optimised, the path with a shorter length is preferred.

- **Drift,** δ: When a robot repeats the path it was taught, it also creates a new map simultaneously. With the time, multiple maps are stored in the database for the same path for the same time of the day/year. Now the question arises about which map should be selected out of these maps which are all roughly from the same time. [21] suggests various strategies, one of them being just selecting the latest map. But as mentioned in the paper, when the robot repeats a path it was taught, it can never be absolutely accurate and there are always some offsets. If each next map is created based on the last map, eventually the accumulated errors will be large enough that the navigation quality will be drastically reduced. This problem is known as drift.

3.4.2 Final cost function

The final cost J_k for a map k is given by equation 3.6.

$$J_k = \theta_1 C_k(t, s_k, w, T, \phi, N, a) + \theta_2 d_k + \theta_3 \delta(n_k), \tag{3.6}$$

where

- t is the current time,
- s_k is the map's timestamp,
- C_k is the map similarity cost calculated using the model of the environment (3.3). The lower the value, the more likely is the map to look similar to the live camera feed. It is a function of t, s_k, and the parameters w, T, ϕ, N, a obtained from spectral analysis,
- θ_1 is the weight for the map similarity cost. The higher the value, the more importance is given to map similarity calculated based on time in path planning,
- d_k is the length of the map,
- θ_2 is the weight for length of the map. The higher the value, the more importance is given to shorter paths in path planning,
- δ is the drift which is function of n_k,

- n_k is the n^{th} version of map k created at the same location in the environment that is to be used during repeat phase. The 50^{th} map is likely to have drifted more compared to the 3^{rd} map from the original map for the same location,

- θ_3 is the weight for the drift.

3.4.3 Robust navigation vs Exploration

Robust navigation in the context of Visual Teach & Repeat navigation refers to the ability of the robot to closely repeat the paths it is taught during teaching phase. Robust navigation prioritizes accuracy over speed or exploration. So the most robust way to navigate would be to use the maps stored in the database which are the most likely to be visually similar to the live feed of the robot during navigation. While robust navigation is definitely desired, another important concept in autonomous robotics is that of exploration. Using only maps that result in robust navigation ultimately leads to a narrow number of options in path planning. There may be other paths that are better than the current known robust paths. In a system where maps are being shared among robots using a centralised database, the ideal case would be to have maps from as many different times as possible for all the paths of the environment. But that is usually not the case, specially in the starting.

Values of $\theta_1, \theta_2, \theta_3$ can be varied to change what mode the robot should operate in. With $\theta_1 = 1, \theta_2 = 0, \theta_3 = 0$, a path that is the most likely to lead to robust navigation is planned and the corresponding sequence of maps are downloaded to the robot from the database. Similarly, with $\theta_1 = 0$ and $\theta_2 = 1, \theta_3 = 0$ would find the shortest path between the current position and destination without placing any importance on accuracy. While repeating the shortest path the robot can create a new map that would contain the visual information for the current time which can be used next time. So if the robot has to navigate the same path next time it has a path that is both robust and shortest. For $\theta_1 = -1, \theta_2 = 0, \theta_3 = 0$, the path that is the least likely to lead to robust navigation is planned without putting any importance on distance. This is the exploration mode where the robot navigates the path for which the maps stored in the database do not look visually similar to the live feed, and hence can create new maps which can be later used for robust navigation. Varying the values of $\theta_1, \theta_2, \theta_3$ in relation to each other leads to different paths by the path planning algorithm. Striking a balance between these is still up for research.

3.5 Path Planning

We are creating a topological map to represent the spatial information about the environment. When a map is uploaded to the database, the user provides a start node and an end node for the map, which is added to a directed graph. The nodes of the graph are the ends of the individual maps of the environment, with edges of the graph being the maps of the environment. Nodes and edges are continually added to the graph as more and more maps are added to the environment. The final step in the process is to find a path (a sequence of maps) between any two nodes of the graph. When a robot reaches a certain location in the environment, it needs a path along with the action commands to navigate between its current location and its goal. The action commands and visual information needed to traverse a path between any two nodes are implicitly contained in the edge between the two nodes of the graph.

There can exist multiple maps even between two consecutive nodes since the same path can be recorded at different times during the teaching phase of VT&R. And since VT&R depends on the images recorded during the teaching phase, the same path is indeed taught multiple times at different times of the day and year to have various versions of maps containing as many possible variations in visual data as possible. A common situation a robot might encounter during navigation is shown in Figure 3.9.

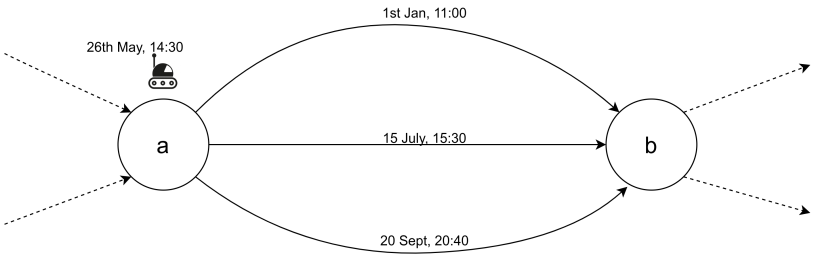


Figure 3.9: The robot reaches node a at a certain time. There are multiple options of maps that it can fetch from the database, to use for traversal from node a to node b, each of which were recorded at a different time in the past.

Assuming that all other factors like distance between the nodes, [energy consumption etc.] are the same for all the three options of maps, the map which would contain images that look most similar to the current time of 26th May 14:30 is the map which was recorded on 15th July 15:30. The equation of model of environment (3.3) assigns a cost to all three of the maps for the current time. In this simple case, it should assign the lowest cost to the map with timestamp 15th July 15:30. For a general case with a large number of nodes, all with maps recorded at different times in the past, a *similarity cost* is calculated for each map in the environment.

Finding a path between any two nodes on the directed graph is just a matter of using any of the path finding algorithms. A^* is well suited for this task due to its ability to incorporate heuristics [30]. Usually the heuristic used is the actual cost of a optimal path from the current node to the goal. The heuristic function in our case is given by equation 3.6.

Based on the path found by the algorithm, a sequence of maps is returned and downloaded from the database to the robot, that it can use to traverse the path. The cost given by equation 3.6 takes care of the criteria to optimise for and hence maps suited for the current system time form the sequence of maps that are returned.

Chapter 4

Experiments

This chapter contains the experiments performed to evaluate the functionality of the methodology proposed in the previous chapter. The presented experiments are designed to test the proposed model of the environment (3.3), efficacy of the parameters calculated from the using spectral analysis of time series created from the similarity matrix, and path planning based on the results of the model of the environment.

4.1 Experiment 1

To test the efficacy of model given by equation 3.3, which in turn depends on the results of the spectral analysis, a somewhat dense dataset is needed, a dataset containing maps for at least 2 consecutive days with sufficient number of maps so that spectral analysis results in a sufficiently accurate prediction. Therefore the first experiment is performed on a real dataset with sufficient density. All of the maps in the dataset are from the same path, just at different times of different days, hence this dataset is not used to test the path planning capabilities. But the dataset does provide a good opportunity to verify the pseudo-cyclic nature of an environment and verify the results the spectral analysis and the cost assigned to each map using the time aspect based on the model of the environment.

4.1.1 Dataset

The dataset used for this experiment was collected by real robot prototype used for transporting cars at the Škoda car manufacturing plant in Mladá Boleslav during the time period of August 8th, 2021 to August 11th, 2021, between the hours of 3:30 PM

and 11:30 PM. The dataset has 24 maps with varying illumination levels. An image from each map of the dataset is shown Figure 4.1:

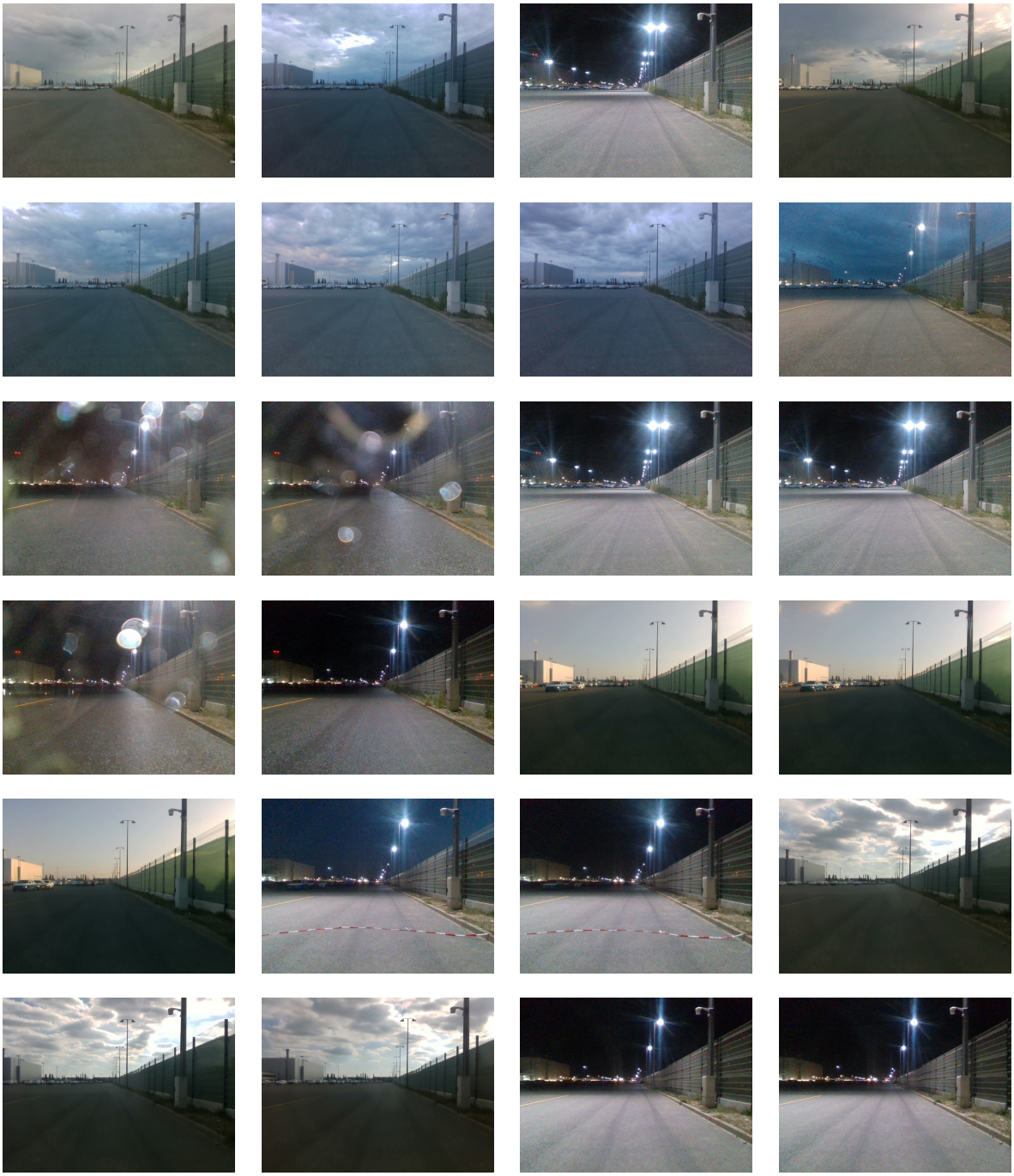


Figure 4.1: First image from each map of the dataset

4.1.2 Similarity Matrix and Spectral Analysis

The first step towards calculating the cost for each map, which would ultimately help in finding an optimal path, is to calculate the similarity matrix for the dataset. We use the neural network [5] to calculate the similarity of each map to every other map in the dataset. In this experiment, only the first image (the ones shown in Figure 4.1) of each map is considered to represent the visual content of the map, since each map was recorded over a short period of time to calculate the similarity of the map against every other map using the neural network. The resulting similarity matrix for the dataset is represented as a heat map in figure 4.2. One can note some of the following observations from the heat map:

- The diagonal elements have the highest values since those represent the similarity of a map with itself, which obviously would be the highest.
- There are "islands", like the one in the top left or the one in the middle, indicating the intensity of how close the maps belonging to them look to each other.
- The periodic nature of the visual information contained in the maps seem to emerge as one moves along a row (or a column).

A time series, as shown in Figure 4.3, is created using the similarity matrix and non uniform Fourier analysis is performed. The spectral analysis outputs all the frequencies, along with the magnitude and phase information, present in the time series. Selecting a few of the frequencies with the largest magnitudes should be sufficiently good for the present application [33]. We select the two most prominent periodicities, i.e ($N = 2$), from the output of the spectral analysis. Therefore the parameters for equation 3.3 are as mentioned in Table 4.1.

T_i (hours)	12	24
w_i	0.03727	0.02893
ϕ_i	3.1616×10^{-5}	-4.3708×10^{-5}

Table 4.1: Parameters estimated for the dataset for $N = 2$

A time series can be constructed using the values in Table 4.1 which approximates the actual time series created using the data from the similarity matrix. This time series, referred to as 'predicted time series' along with the time series created from the similarity matrix shown represented by Figure 4.2 are shown in Figure 4.3. The predicted series tries to approximate the actual time series. If more number of frequencies are used from the output of the spectral analysis, the more accurate the prediction will

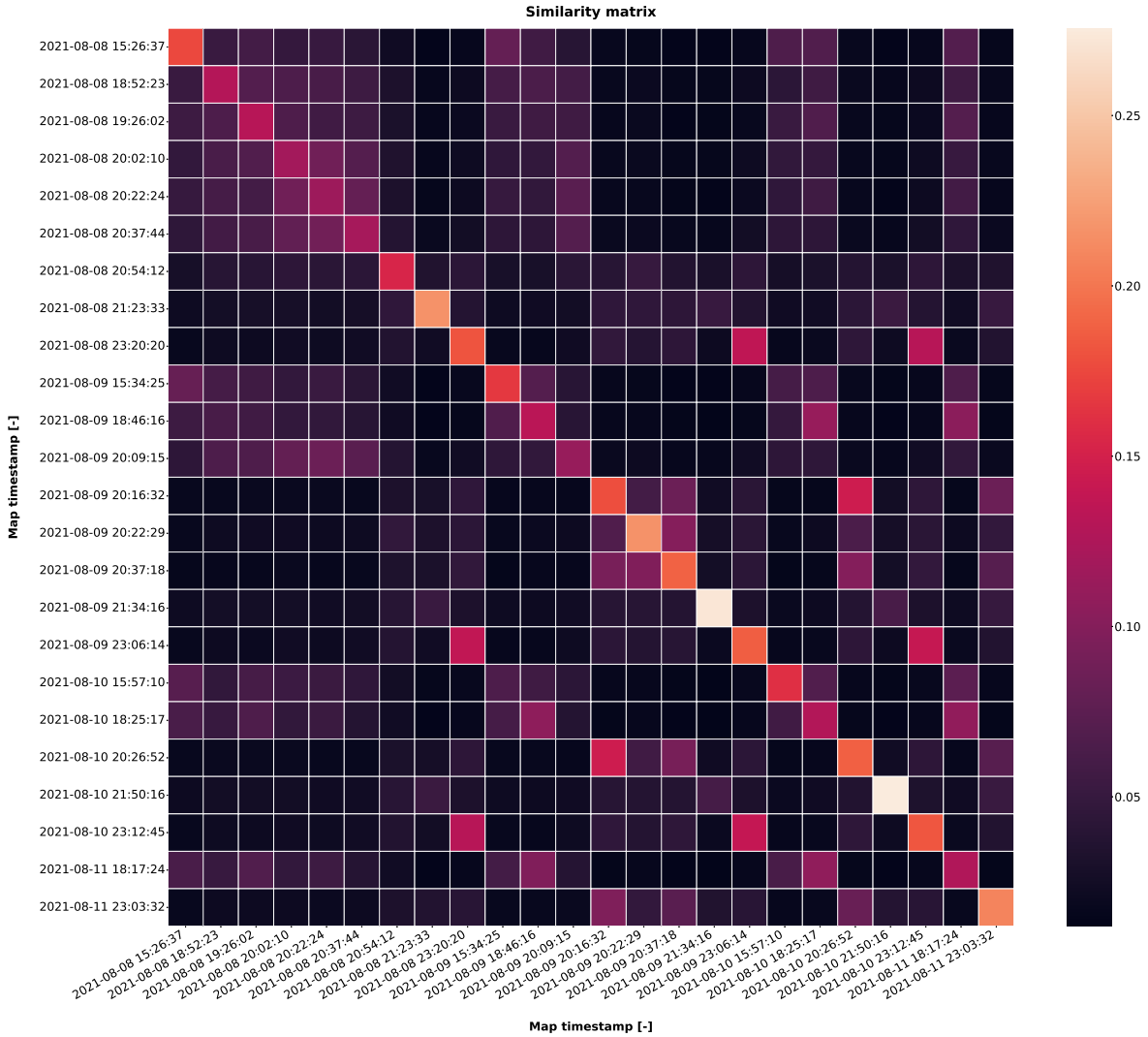


Figure 4.2: Similarity matrix for the dataset

look. As the number of samples increases the better are the results of Fourier analysis. A higher number of samples, i.e more maps in the database for the environment which leads to a more dense time series, would also lead to a predicted series which approximates the actual time series more accurately.

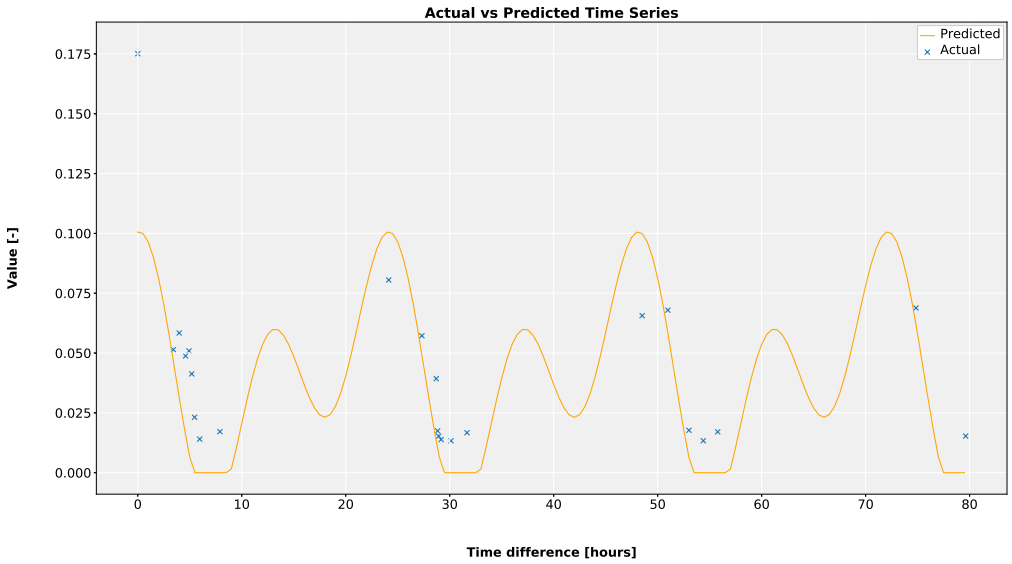


Figure 4.3: x-axis values denote the time elapsed from the timestamp of the first map of the similarity matrix. y-axis denotes the similarity of a map with first map in the matrix.

The periodic nature can again be observed in the time series, also because it has been derived from the similarity matrix. It can be argued that the decision of selecting just two frequencies doesn't seem to result in a 'predicted series' that closely predicts the actual series. Deciding on a higher number of frequencies would have resulted in a better prediction, though with added computation, but from the results in the following section it can be seen that for the current experiment it does seem to give expected results. The number of prominent frequencies can be changed as needed.

4.1.3 Map Similarity using the Model of the Environment

Since the robustness of a Visual Teach & Repeat navigation system depends on the quality of heading correction using the live feed from the camera on the robot and the visual information of stored maps, we want to use a map from the database whose visual information (images) is most likely to be similar to the current view of the robot. To enable this we developed the model of the environment given by equation 3.3. The

model assigns a cost to each map of the environment, the lower the cost the most likely it is to look similar to the current view based on time. All of the components needed to do so have been calculated in the section 4.1.2.

Using the parameters from Table 4.1 with equation 3.3, a cost is assigned to each of the 24 maps of the dataset. The cost for each map calculated for two distinctly different times of the day are shown in Figures 4.4 and 4.5. Cyclic nature of the environment can also be noticed in the cost assigned to each map since the model itself is based on the idea that the processes in any environment are pseudo-cyclic in nature.

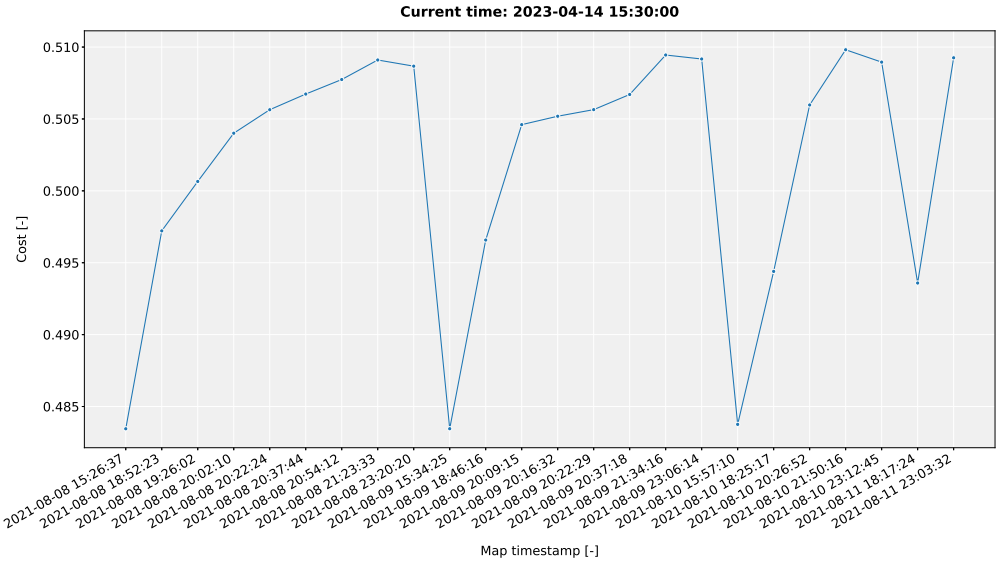


Figure 4.4: Cost for each map based on similarity for system time 2023-04-14 15:30

From Figures 4.4 and 4.5, it can be seen that cost of maps whose timestamps are closer to the current time is lower than the maps whose timestamps are farther from the current time, and the pattern repeats according to the frequencies outputted by the Fourier analysis (4.1). This intuitively makes sense since a map, if one exists in the database, from the same time as current time is the most likely to contains images that would look similar to the live feed of the camera on-board the robot. If the database doesn't contain a map from around the same time, then the next best is selected based on the cost. This would not always lead to the ideal result. For example, if it the robot needs to navigate an environment at 03:00 am, but all the maps were created during morning time. A map will still be selected since a cost will be assigned to every map and the map with a timestamp closest to 03:00 am will have the lowest cost. But this problem takes care of itself with time as more maps are created during exploration phase. These costs contribute to the final cost in equation 3.6, which in turn is used

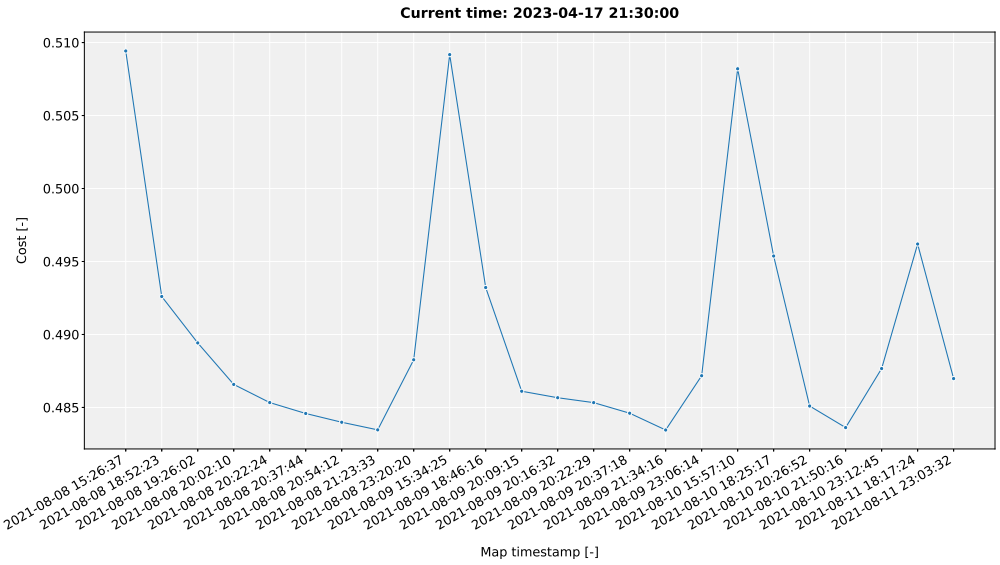


Figure 4.5: Cost for each map based on similarity for the system time 2023-04-17 21:30

for path planning.

4.2 Experiment 2

Experiment 2 and the following experiment 3 were conducted at Karlovo náměstí campus of Czech Technical University in Prague using the Spot robot by Boston Dynamics (Figure 4.6) carrying a payload and equipped with Basler Ace 2 camera. The robot was taught various paths at two different times of the day in the first week of May 2023. Some of the paths were taught before the sunset, and some of the paths were taught after the sunset so that there is distinct difference in the illumination levels. These experiments are also able to prove the efficient map management abilities of the system. Each map created by the robot is uploaded to a database, which can be fetched as and when needed during navigation.

While experiment 1 was performed on a dataset containing maps collected over three days and proved to show the capabilities of estimating parameters using spectral analysis needed for the model of the environment (3.3) to compute cost of each map, all the maps were between the two same start and end nodes. Experiments 2 and 3 are designed to bring everything together and show the end result of using the model of the environment (3.3) followed by path planning to select a sequence of maps that can lead to a robust visual navigation.

Figure 4.6: Spot by Boston Dynamics used for experiments

The location (environment) of experiment is shown in Figure 4.8 along with the topological directed graph representation of the environment superimposed over it. There are nine segments/edges and seven nodes, with each node being the starting point or end point of a segment. A map is created for each segment by manually operating the robot so that it can be taught the path which can later be used to repeat the path. Out of the nine maps, five are created before the sunset, and the remaining four after the sunset. The details about the sections are as given in Table 4.3. The aim is to go from node A to node G using a sequence of maps that optimises all the criteria mentioned in section 3.4.

Map name	**Start node**	**End node**	**Timestamp**	**Distance**	**Day/Night**
map1	A	C	2023-05-07 17:08:32	18.723	Day
map2	C	D	2023-05-07 17:10:49	13.598	Day
map3	D	E	2023-05-07 17:23:01	19.345	Day
map4	E	B	2023-05-07 17:24:36	24.703	Day
map5	B	G	2023-05-07 17:26:09	26.871	Day
map6	E	F	2023-05-07 21:02:47	13.331	Night
map7	F	G	2023-05-07 21:04:21	20.298	Night
map8	A	B	2023-05-07 21:07:37	18.001	Night
map9	A	D	2023-05-07 21:09:25	26.146	Night

Table 4.3: Details of the graph of the environment

(a) map1 (b) map2 (c) map3

(d) map4 (e) map5 (f) map6

(g) map7 (h) map8 (i) map9

Figure 4.7: Images from the nine maps. map1 to map5 were created before sunset, and the rest after sunset.

An image from each of the nine maps is shown in Figure 4.7. Each image shown is the visuals the robot's camera captured at the starting position (not all the images) of a map during the teaching phase.

To navigate from node A to node G, a path needs to be planned. There are five possible routes it can take: ABG, ADEBG, ADEFG, ACDEBG, ACDEFG. If it was just a matter of distance the obvious choice would have been the shortest path ABG but instead we want to choose a path that is the most likely to lead to robust visual navigation. The maps are stored in the database which can be used by any other robot operating in the same environment. To select such a sequence of maps the model of the environment (3.3) is used to assign a cost to each of the nine maps. Since its just 1 map between each node pair, and all the maps have been created on a single day just within a few hours of each other, creating a similarity matrix and constructing a time

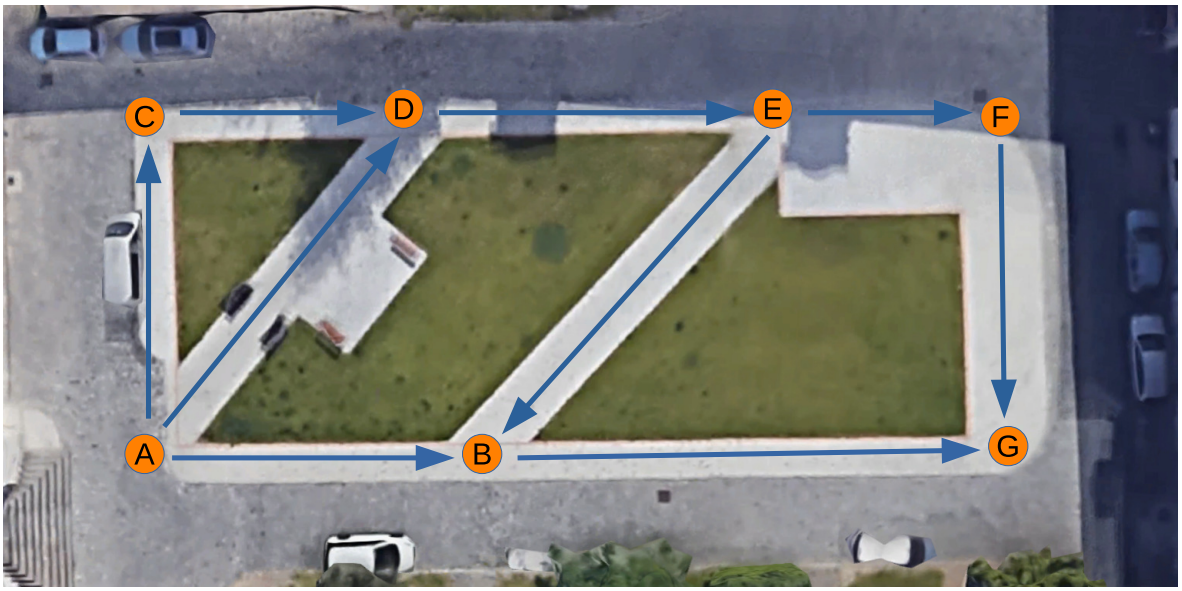


Figure 4.8: Karlovo náměstí CTU Campus in Prague with the graph representation of the environment. Taken from [34].

series to do Fourier analysis would not lead to any meaningful result. For the purpose of this experiment, since it is performed in an outdoor environment we assume that the processes of the environment repeat every 24 hours, hence the parameters can be assumed to be $T = 24\ hours$, $w = 1$, $\phi = 0$, $N = 1$.

For the given parameters, a different cost is assigned to each map depending on the current state of the environment. Similarity costs calculated for each map of the environment at two different times is shown in Figure 4.9. If the robot is asked to go from node A to node G at 16:00:00 the next day, the model assigns higher costs to the maps created after the sunset, and lower cost to the ones collected before the sunset, which is as expected. If the robot is asked to go from node A to node G at 22:00:00, the model assigns higher costs to the maps created before the sunset, and lower cost to the ones collected after the sunset this time, which is also as expected. Assuming that we want the maps that would lead most robust visual navigation i.e we do not care about the distance the robot has to cover to reach its goal, the weights in equation 3.6 can be set to $\theta_1 = 1, \theta_2 = 0, \theta_3 = 0$. With these values, the final cost for each map is as shown in Table 4.5 for the two different times of the day.

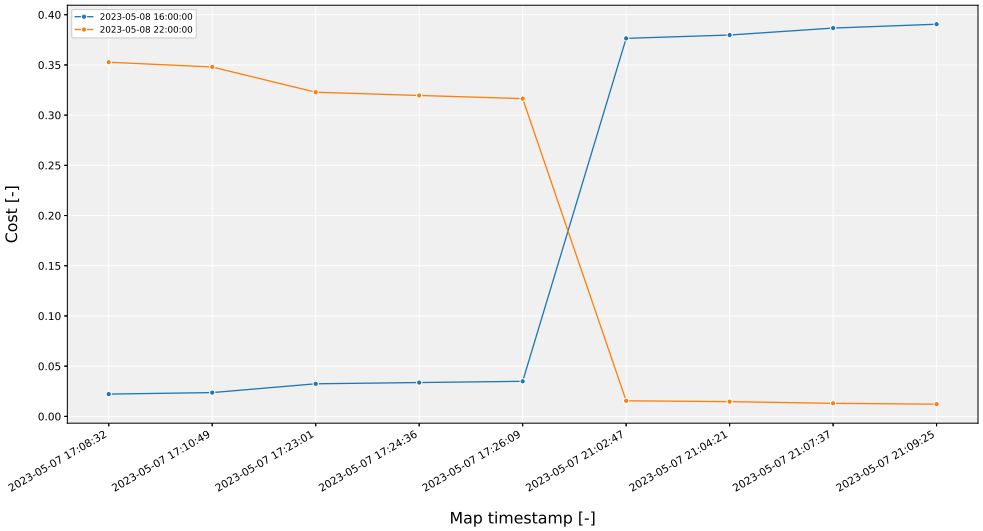


Figure 4.9: Similarity cost calculated for each map at two different times of the day.

Map	Final cost at 16:00	Final cost at 22:00
map1	0.022	0.352
map2	0.023	0.347
map3	0.032	0.322
map4	0.033	0.319
map5	0.034	0.316
map6	0.376	0.015
map7	0.379	0.014
map8	0.386	0.013
map9	0.390	0.012

Table 4.5: Final cost for each map for the current system time 2023-05-08 16:00:00 and 2023-05-08 22:00:00

Using the final cost of each map as heuristic cost in A^* algorithm to find a path between node A and node G, the planned path for the robot if navigation is desired at 2023-05-08 16:00:00 is as shown in Figure 4.10. With $\theta_1 = 1, \theta_2 = 0, \theta_3 = 0$, again using the final cost of each map as heuristic cost in A^* algorithm to find a path between node A and node G, the planned path at 2023-05-08 22:00:00 is as shown in Figure 4.11.

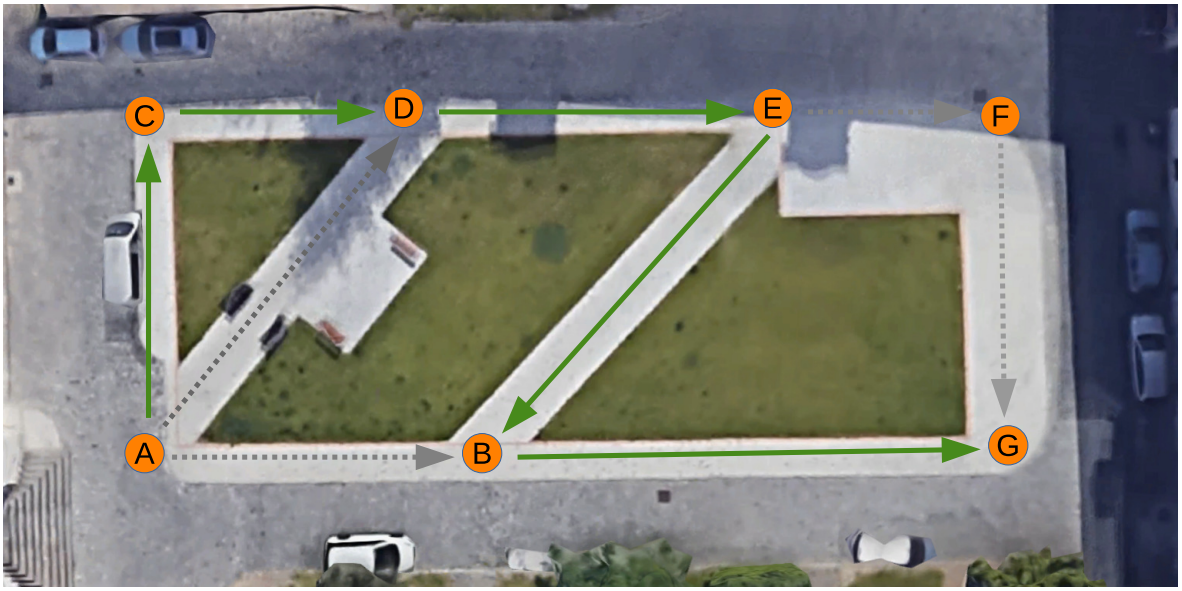


Figure 4.10: The sequence of maps fetched when the robot requests for a path between A and G at time 2023-05-08 16:00:00. From looking at the timestamps of each map in Table 4.3, the path with the maps that were captured before the sunset (17:00-17:30) is chosen to traversal.

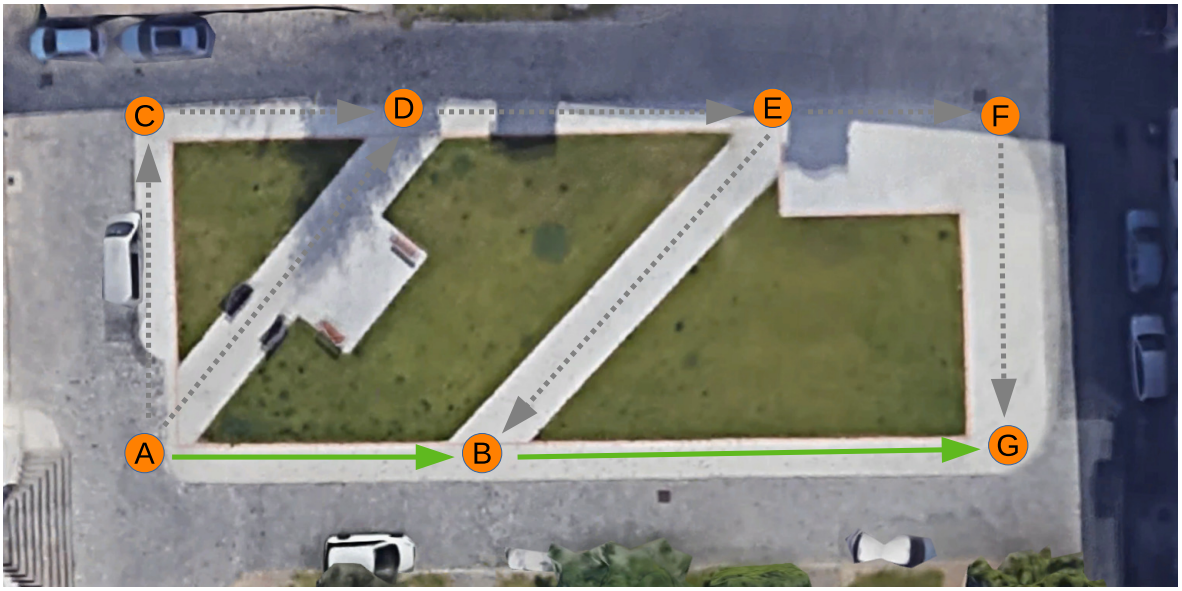


Figure 4.11: The sequence of maps fetched when the robot requests for a path between A and G at time 2023-05-08 22:00:00

4.3 Experiment 3

While experiment 2 was designed to show the path planning capabilities of system using all the information it has available to it, this experiment is designed to show how the weights in equation 3.6 affect which path is selected for traversal. The experiment 3 is conducted at the same location using the same robot with Visual Teach & Repeat system. It is taught 3 paths at two different times. The graph representation of the environment (superimposed on the satellite image of the site) is shown in Figure 4.12. The details about the paths are as given in Table 4.7. As is evident from the data in the table, out of the three maps, two are created before the sunset, one after the sunset and each path is significantly different in length from each other.

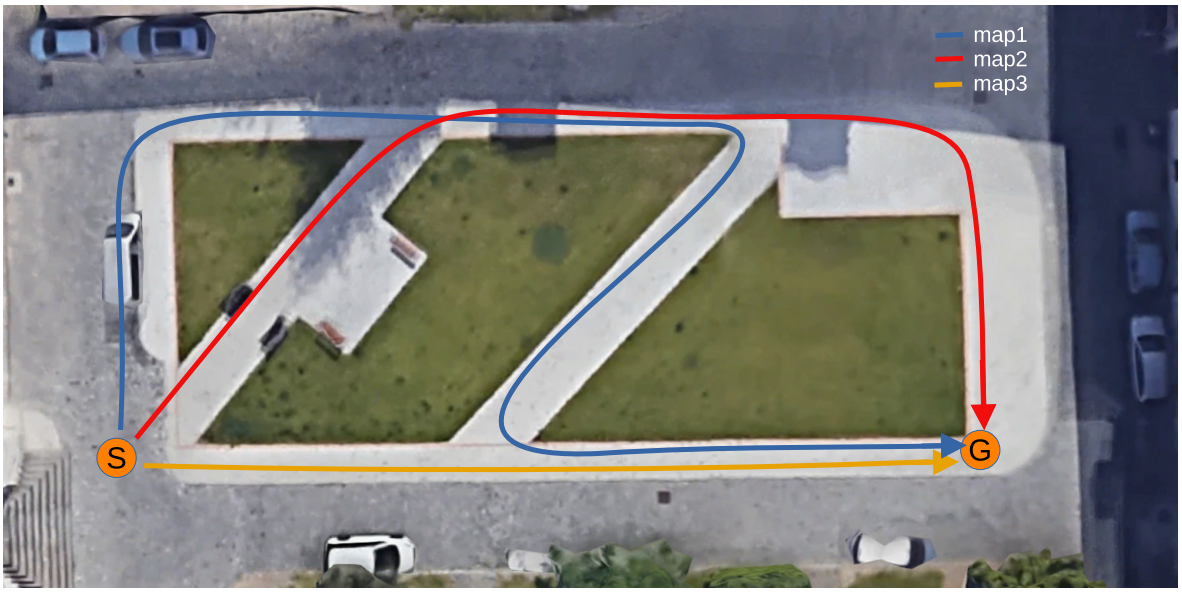


Figure 4.12: Three paths taught to the robot during teaching phase.

Map	Start node	End node	Timestamp	Distance	Day/Night
map1	S	G	2023-05-07 18:04:03	100.392	Day
map2	S	G	2023-05-07 18:09:54	69.692	Day
map3	S	G	2023-05-07 21:18:07	44.469	Night

Table 4.7: Details of the three paths for the experiment.

An image from each of the three maps is shown in Figure 4.13. Each image shown is the visuals the robot's camera captured from just the starting position (not all the images) of a map during the teaching phase.

(a) map1 (b) map2 (c) map3

Figure 4.13: Images from the three maps

Again, since all the maps have been created on a single day just within a few hours of each other, creating a similarity matrix and constructing a time series to do Fourier analysis would not lead to accurate results. We assume that the processes of the environment repeat every 24 hours, hence the parameters are assumed to be $T = 24\ hours, w = 1, \phi = 0, N = 1$.

For the given parameters, the cost assigned to each map (for two different times of the day) based on just the likely visual similarity between the live feed and the stored maps, if the robot is asked to traverse from node S to node G at 16:00:00 and 22:00:00 the next day, are shown in Figure 4.14.

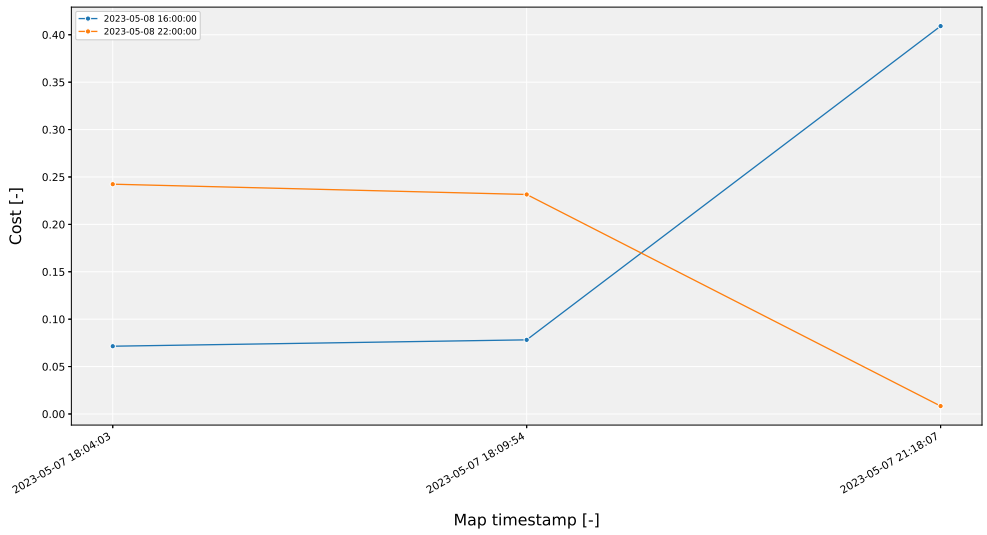


Figure 4.14: Similarity cost calculated for each map for system time 2023-05-08 16:00:00 and 2023-05-08 22:00:00

We test what effect varying $\theta_1, \theta_2, \theta_3$ has on which map is selected for navigation. Since the experiments are performed over just a single day, the effect of drift in equation 3.6 can be ignored, hence we assume $\theta_3 = 0$. If more weight is given to distance, the shortest path is selected. As is evident from from the values in Figure 4.14 and Table 4.9, if all the weight is given to visual similarity the path shown in 4.15 is selected even if it is the longest distance that the robot has to travel. The map created for this path would lead to the most robust visual navigation.

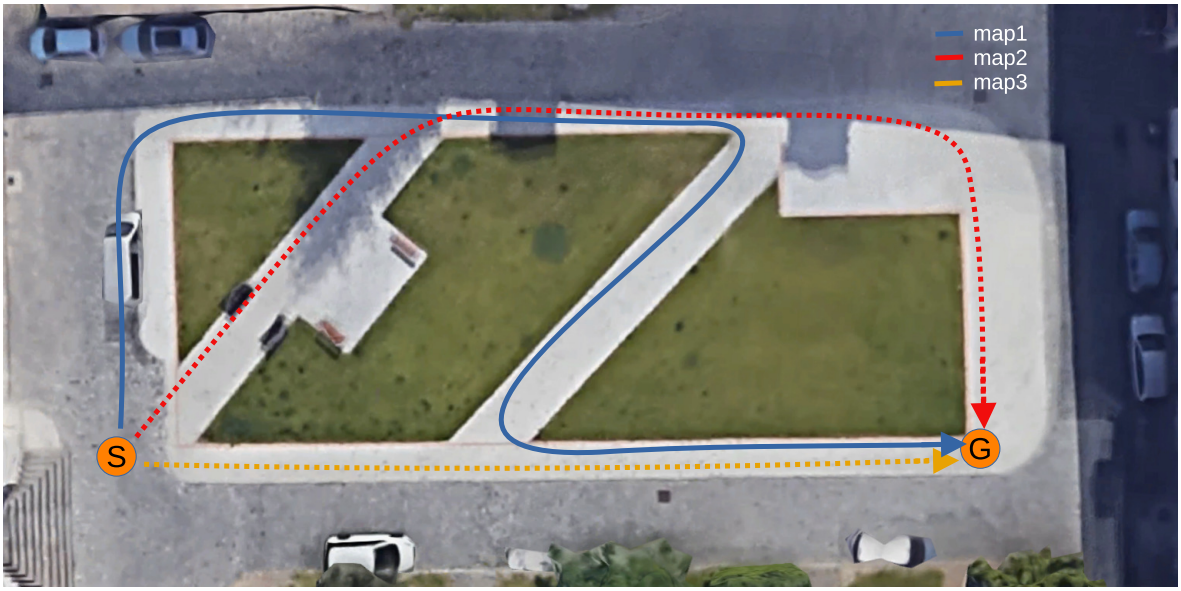


Figure 4.15: For $\theta_1 = 1, \theta_2 = 0, \theta_3 = 0$, map1 is selected for navigation at time 2023-05-08 16:00:00

Map	Similarity cost	Normalized distance	Final cost
map1	0.071	1.000	0.071
map2	0.078	0.694	0.078
map3	0.409	0.422	0.409

Table 4.9: Cost of each map when $\theta_1 = 1.0, \theta_2 = 0, \theta_3 = 0$ at current system time 2023-05-08 16:00:00

As weight for distance is increased, visual similarity is not the only factor being considered anymore and path shown in Figure 4.16 is selected since it is not as visually similar as the previous path but it is close enough with the advantage being that is shorter is length. The final costs for the maps are given in Table 4.11.

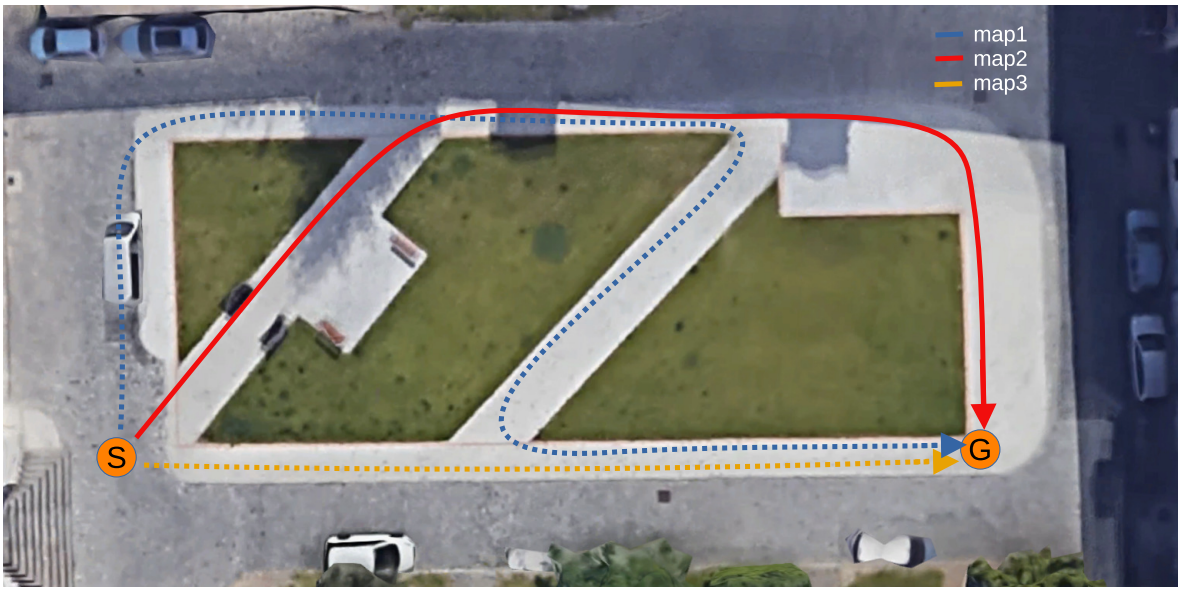


Figure 4.16: For $\theta_1 = 0.7, \theta_2 = 0.3, \theta_3 = 0$, map2 is selected for navigation at time 2023-05-08 16:00:00

Map	**Similarity cost**	**Normalized distance**	**Final cost**
map1	0.071	1.000	0.350
map2	0.078	0.694	0.262
map3	0.409	0.422	0.419

Table 4.11: Cost of each map when $\theta_1 = 0.7, \theta_2 = 0.3, \theta_3 = 0$ at current system time 2023-05-08 16:00:00

On the other hand, if distance is the only factor we care about and not about the robustness of navigation, most of the weight can be given to distance. The navigation is this case can be the least accurate when it comes to the robot repeating the taught paths. This can be a desired strategy in case we want to create more maps for a given segment of the environment which can later be used to robust navigation in the future. An example of this is shown in Figure 4.17. Values for the cost are given in Table 4.13.

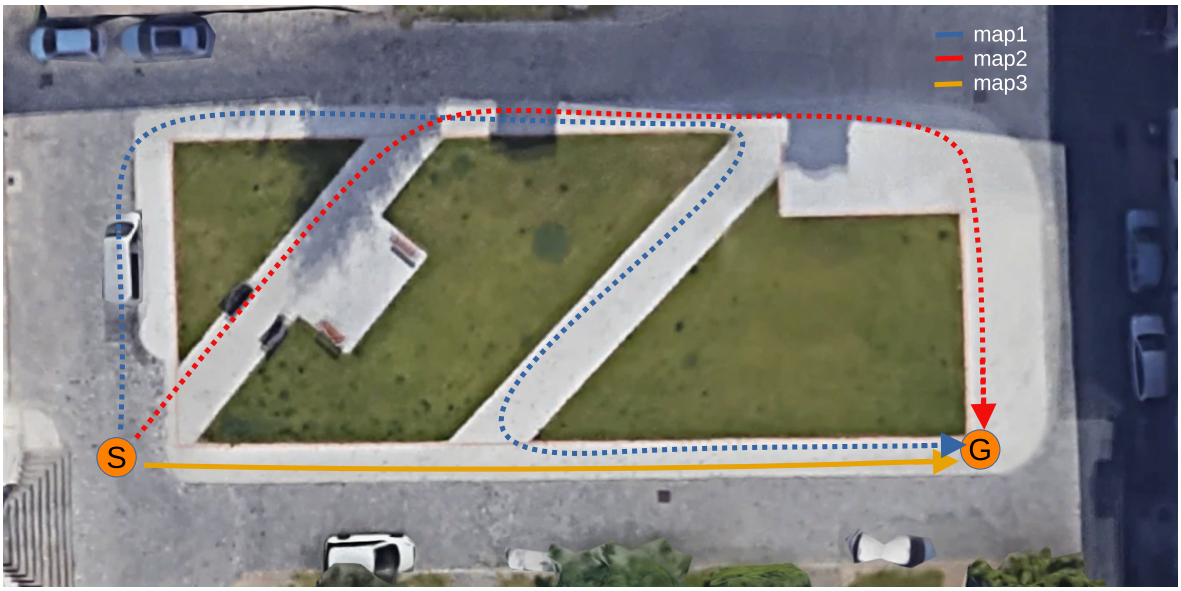


Figure 4.17: For $\theta_1 = 0.3, \theta_2 = 0.7, \theta_3 = 0$, map3 has the lowest final cost at time 2023-05-08 16:00:00

Map	Similarity cost	Normalized distance	Final cost
map1	0.071	1.000	0.721
map2	0.078	0.694	0.509
map3	0.409	0.422	0.432

Table 4.13: Cost of each map when $\theta_1 = 0.3, \theta_2 = 0.7, \theta_3 = 0$ at current system time 2023-05-08 16:00:00

During this traversal the robot can create a new map simultaneously for the current time which can be used in future with the advantage of being the most visually similar as well as being the shortest path for the current time. A scientific method on how to the select values of $\theta_1, \theta_2, \theta_3$ is up for further research.

Chapter 5

Conclusions and Recommendations

In conclusion, this thesis presented an improved map management strategy for Visual Teach & Repeat using a database service and an algorithm to find a sequence of maps using a model of the environment that exploits the periodic nature of the processes that take place in the environment, for navigation in long-term deployment of autonomous mobile robots. The implemented map management strategy enabled performing spectral analysis [9] on all the maps created by VT&R system during teaching phase which helps to find the periodicities present in the environment. Experiments were conducted to test all the aspects of the work: detection of periodicities of an environment, performance of the model of the environment, functionality of the path planning using the heuristic cost of each map, benefits of using a centralised database for managing maps. The results of the experiments showed that detection of periodicities works reasonably well given that enough maps are present in the database for the environment, specially from around the similar time at which navigation is to be performed. If the calculated periodicities are reasonably accurate, the heuristic cost assigned to each map by the model of the environment is as expected and thus results in path planning that returns a sequence of maps that optimizes the established criteria. The framework also allows to switch from navigation mode to exploration mode in a convenient way. The algorithm's adaptability to continuously evaluate the data present in the database and update the model of the environment accordingly makes it suitable for long-term deployment in any type of environment, whether indoor or outdoor. Managing maps using a database service enables multi robot systems to share expertise with each other. Repository with the code is available at [35].

Future work may involve improving the accuracy for the case when there aren't enough maps in the database to precisely calculate the periodicities present in the environment. Another improvement can be done in determining optimal value of weights in the final cost function 3.6 for the cases of robust navigation and exploration. Work on how to incorporate the effect of drift in assigning the cost to each map can also be

done. We have only used the first row of the similarity matrix here to construct the time series. Figuring out how to exploit the other rows of the matrix is also another problem, solution to which might result in better results.

Bibliography

[1] Sebastian Thrun, Wolfram Burgard, and Dieter Fox. *Probabilistic robotics*. MIT Press, Cambridge, Mass., 2005.

[2] Tomáš Krajník, Filip Majer, Lucie Halodová, and Tomáš Vintr. Navigation without localisation: reliable teach and repeat based on the convergence theorem. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1657–1664, 2018.

[3] Michael Paton, Kirk MacTavish, Laszlo-Peter Berczi, Sebastian Kai van Es, and Tim D. Barfoot. I can see for miles and miles: An extended field test of visual teach and repeat 2.0. In *International Symposium on Field and Service Robotics*, 2017.

[4] Maxim Simon, George Broughton, Tomáš Rouček, Zdeněk Rozsypálek, and Tomáš Krajník. Performance comparison of visual teach and repeat systems for mobile robots. In *Modelling and Simulation for Autonomous Systems: 9th International Conference, MESAS 2022, Prague, Czech Republic, October 20–21, 2022, Revised Selected Papers*, page 3–24, Berlin, Heidelberg, 2023. Springer-Verlag.

[5] Zdeněk Rozsypálek, George Broughton, Pavel Linder, Tomáš Rouček, Jan Blaha, Leonard Mentzl, Keerthy Kusumam, and Tomáš Krajník. Contrastive learning for image registration in visual teach and repeat navigation. *Sensors*, 22(8), 2022.

[6] Paul Timothy Furgale and Tim D. Barfoot. Visual teach and repeat for long-range rover autonomy. *Journal of Field Robotics*, 27:534–560, 2010.

[7] Tomáš Krajník, Sol Pedre, and Libor Přeučil. Monocular navigation for long-term autonomy. In *2013 16th International Conference on Advanced Robotics (ICAR)*, pages 1–6, 2013.

[8] Tomáš Krajník, Jan Faigl, Vojtěch Vonásek, Karel Kosnar, Miroslav Kulich, and Libor Preucil. Simple yet stable bearing-only navigation. *Journal of Field Robotics*, 27, 2010.

[9] Tomáš Krajník, Jaime Pulido Fentanes, João Machado Santos, and Tom Duckett. Frequency map enhancement: introducing dynamics into static environment models. *IEEE Transaction on Robotics*, 2016.

[10] Tomáš Krajník, Tomas Vintr, Sergi Molina, Jaime Fentanes, Grzegorz Cielniak, Oscar Mozos, George Broughton, and Tom Duckett. Warped hypertime representations for long-term autonomy of mobile robots. *IEEE Robotics and Automation Letters*, PP:1–1, 07 2019.

[11] Tim D. Barfoot, Braden Stenning, Paul Timothy Furgale, and Colin McManus. Exploiting reusable paths in mobile robotics: Benefits and challenges for long-term autonomy. In *2012 Ninth Conference on Computer and Robot Vision*, pages 388–395, 2012.

[12] Zhichao Chen and Stanley T. Birchfield. Vision-based path following without calibration. In Alejandra Barrera, editor, *Mobile Robots Navigation*, chapter 21. IntechOpen, Rijeka, 2010.

[13] Zhichao Chen and Stanley T. Birchfield. Qualitative vision-based path following. *Trans. Rob.*, 25(3):749–754, jun 2009.

[14] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In *Communications of the ACM*, volume 60, pages 84 – 90, 2012.

[15] Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. Superpoint: Self-supervised interest point detection and description. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 337–33712, 2017.

[16] Tomáš Krajník, Pablo de Cristóforis, Keerthy Kusumam, Peer Neubert, and Tom Duckett. Image features for visual teach-and-repeat navigation in changing environments. *Robotics Auton. Syst.*, 88:127–141, 2017.

[17] Nan Zhang, Michael Warren, and Tim D. Barfoot. Learning place-and-time-dependent binary descriptors for long-term visual localization. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 828–835, 2018.

[18] Mihai Dusmanu, Ignacio Rocco, Tomás Pajdla, Marc Pollefeys, Josef Sivic, Akihiko Torii, and Torsten Sattler. D2-net: A trainable cnn for joint detection and description of local features. *ArXiv*, abs/1905.03561, 2019.

[19] Feras Dayoub and Tom Duckett. An adaptive appearance-based map for long-term topological localization of mobile robots. In *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3364–3369, 2008.

[20] David M. Rosen, Julian Mason, and John J. Leonard. Towards lifelong feature-based mapping in semi-static environments. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1063–1070, 2016.

[21] Lucie Halodová, Eliška Dvořráková, Filip Majer, Tomáš Vintr, Oscar Martinez Mozos, Feras Dayoub, and Tomáš Krajník. Predictive and adaptive maps for long-term visual navigation in changing environments. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 7033–7039, 2019.

[22] Sylvio Ferraz-Mello. Estimation of periods from unequally spaced observations. *The Astronomical Journal*, 86:619, 1981.

[23] Mona Gridseth and Timothy D. Barfoot. Keeping an eye on things: Deep learned features for long-term visual localization. *CoRR*, abs/2109.04041, 2021.

[24] Matthew Gadd and Paul Newman. The data market: Policies for decentralised visual localisation. *CoRR*, abs/1801.05607, 2018.

[25] Winston Churchill and Paul Newman. Experience-based navigation for long-term localisation. *The International Journal of Robotics Research*, 32:1645 – 1661, 2013.

[26] Michael Milford and Gordon Wyeth. Persistent navigation and mapping using a biologically inspired slam system. *The International Journal of Robotics Research*, 29(9):1131–1153, 2010.

[27] Peter Mühlfellner, Mathias Bürki, Michael Bosse, Wojciech Derendarz, Roland Philippsen, and Paul Furgale. Summary maps for lifelong visual localization. *J. Field Robot.*, 33(5):561–590, aug 2016.

[28] Mathias Bürki, Marcin Dymczyk, Igor Gilitschenski, Cesar Cadena, Roland Siegwart, and Juan I. Nieto. Map management for efficient long-term visual localization in outdoor environments. *CoRR*, abs/1808.02658, 2018.

[29] Steven M. LaValle. Rapidly-exploring random trees : a new tool for path planning. *The annual research report*, 1998.

[30] Peter E. Hart, Nils J. Nilsson, and Bertram Raphael. A formal basis for the heuristic determination of minimum cost paths. *IEEE Transactions on Systems Science and Cybernetics*, 4(2):100–107, 1968.

[31] MinIO. Minio s3 object storage.
`https://min.io/`.

[32] Zdeněk Rozsypálek, Tomáš Rouček, Tomáš Vintr, and Tomáš Krajník. Multidimensional particle filter for long-term visual teach and repeat in changing environments. *IEEE Robotics and Automation Letters*, 8(4):1951–1958, 2023.

[33] Tomas Krajnik, Jaime Pulido Fentanes, Grzegorz Cielniak, Christian Dondrup, and Tom Duckett. Spectral analysis for long-term robotic mapping. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3706–3711, 2014.

[34] Google maps. `https://www.google.com/maps`. Accessed: 30 Mar 2023, 21:10:07.

[35] Vivek Punia. Chronorobotics-project. GitHub repository, 2023. `https://github.com/puklu/Chronorobotics-project`.