Visual Navigation under Image Compression

Presented by:
Saylin Pillay

Prepared for:
Dr Paul Amayo
Dept. of Electrical and Electronics Engineering
University of Cape Town

Submitted to the Department of Electrical Engineering at the University of Cape Town in partial fulfilment of the academic requirements for a Bachelor of Science degree in Bsc Mechatronics

October 14, 2019

Declaration

1. I know that plagiarism is wrong. Plagiarism is to use another's work and pretend that it is one's own.
2. I have used the IEEE convention for citation and referencing. Each contribution to, and quotation in, this report from the work(s) of other people has been attributed, and has been cited and referenced.
3. This report is my own work.
4. I have not allowed, and will not allow, anyone to copy my work with the intention of passing it off as their own work or part thereof.

Signature:.....

S. Pillay

Date:.....14/10/2019.....

Acknowledgments

I would firstly like to thank my supervisor, Dr Paul Amayo. Your constant guidance and helping me set the scale of this project, helped me plan and eventually performed the requirements of this project to my expectations. You also opened up a a topic in a field that I was not too familiar with and at the end of this project, I can say that it would definitely be something I would want to look into the future.

Thank you to my parents for their ever-lasting support and for the opportunity to reach my dreams. My brother, Sereshan for visiting me when he gets time off work and always being there when I need support.

Lastly, my friends who gave me advise me when I needed help on this project and listening to my rants when things went wrong. Also for helping me balance work with fun when I needed to clear my head.

Abstract

The current research in the field of visual navigation is focused on accurately determining a robots position in the environment that surrounds it. In order to accomplish this, more sensors are being utilized on the robots to that the data captured gathers every bit of detail that is present around the robot. This raises an issue of a vast amount of data being computed and stored on the robot which leads to inefficiency.

The study presented in this paper implements a single aspect of visual navigation in which the data is compressed. This will investigate the performance of a visual navigation system under compression.

The system will be created by using an image-based localisation technique in order to determine the location of a robot. The proposed system makes use of data in the form of images to use as an input into a visual localisation pipeline. This data will consist of various conditions such as weather, time of day and image manipulation to test the robustness and accuracy of the pipeline.

To match the images, the use of a proposed algorithm, SIFT (Scale Invariant Feature Transform) will be used where it will be able to match images taken by cameras on a robot and match it appropriately with images stored in a database.

Following, the images will be compressed and tested under the proposed conditions to investigate how this will affect the performance of the pipeline. This will be done by determining the distance covered by route found by the image matching pipeline and comparing it to the actual distance measured from a GPS.

Finally, the results will be evaluated and a conclusion will be drawn. It was found that the requirements of the system were met and recommendations and future work were made.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

This chapter will give the reader an overview on the project's procedure and goals. It will highlight the projects scope and demonstrate how the project is undertaken.

1.1 Background to the study

Today's technology and robots are beginning to become more advanced where robots are being used to complete physical tasks that humans were only able to do. A vital component to achieve this, is the use of computer vision. This is a technique that is formed to allow computers to "see" and understand visually what is around them through the use of images.

Computer vision started in the early 1970's where it was created to mimic human vision. The goal of it is not just to see but to gain useful results based on the observations that is found. It can create a 3D image from multiple 2D images which is useful for applications such as autonomous cars. Robots like autonomous cars have a large number of sensors on it, which makes use of data such as images. This will help the robot to navigate itself according to the surroundings.

Robot navigation is crucial as it allows the robot to determine its own frame of reference and then be able to plan a path to reach its destination. If it does not know its exact position at the beginning of the planned trajectory, it will encounter problems in getting to the destination. To be able to navigate around an environment, the robot would require a type of representation such as localisation or map-building.

This leads to the transfer and processing of a large scale of data which results in a lack of efficiency as it takes longer for a computer to handle it. The number of sensors and data continue to increase in robots with time which requires a solution to help manage this in order to allow the robot to be efficient.

This study focuses on creating a visual navigation system to enable a robot such as an autonomous car to be able to locate its location by using images acquired from its sensors. Once this is established, we can test the effects of image compression on the system.

1.2 Objectives of this study

The research shown in this report aims to implement a system capable of visual navigation and localisation pipeline through the use of images attained from a robot. The images will be of different conditions and then undergo compression to understand how this will affect the pipelines performance.

1.2.1 Problems to be investigated

The first challenge will be to create an image-based localisation pipeline which will mimic the likes of a "live" platform. This means that the system will act as if images from robot sensors were directly being fed into the pipeline which gives this study a real-life situation.

There is no access to technology to gain the relevant data such as images and GPS coordinates taken from a robot. This data will need to provide different conditions such as weather, lighting and quality to create diversity in the testing process. A data source will need to be identified to meet the specifications of this study which is trustworthy and relevant.

The next step will be to test the pipeline under different conditions to test the robustness of the pipeline. Compressing these images with different condition permutations will allow us to grasp to how compression will affect the pipelines performance.

1.2.2 Purpose of the study

Computer vision technology is using more sensors and data transfer as the technology improves. Robot navigation requires this in order to accurately establish its location and then determine a path. This study will give an indication on the effects of compressing data such as images in a visual localisation pipeline which plays a vital role in a number of robot navigation applications.

The results can help enable robot navigation technology the access of having more data to be used and also increase the efficiency while not compromising the storage available. The research presented in this report can be utilized for any type of robot ensuing visual navigation which is becoming a common feature in the present and future robotics.

1.3 Scope and Limitations

The use of data is required to conduct this study, however the acquisition of the data falls outside the scope of this project. The discussion and use of the data do fall within the scope. This study focuses on using this data, manipulating it, in order to find out how it would affect the system it will be used for. The main scope is making use of algorithms in order to assess the performance of the pipeline under various conditions.

The limitation comes with the datasets being used, this study is limited to the reliability of data sources and therefore cannot set conditions for the images. Thus, setting up conditions or acquiring images and GPS conditions at a chosen interval cannot be achieved. The system will need to be compatible with the pre-determined format of the data being used.

A major limitation that comes with this project, is the 12 weeks that is given in order to design, test and show this system. This is run parallel with other course work in the first 6 weeks. This study surveys and explores the testing of different conditions on a visual localisation pipeline. Therefore, only a limited number of tests can be conducted whereas it would be beneficial to partake in as many different experiments as possible.

1.4 Plan of development

The report begins with the requirements and challenges of the research presented. It is followed with an introduction and insight of the theory and methods used to accomplish the visual localisation pipeline. This includes topics on the acquisition of data and various techniques to undertake image feature matching where, new and old methods are discussed. The discussion of image manipulation and methods to compress images is done. This should give the reader a good understanding of the concepts and work that will be conducted in the remainder of the report.

This is followed by the system design used to conduct this study. A detailed process is presented which include explanations, methods, pseudocode and flowcharts. By the end of this chapter, the reader should be able to replicate this system and knowledgeable of the manner of results to follow.

Following, is the results of the proposed system design. Various experiments are conducted and a description along with a quantification of the results are displayed. With each result, is an analysis of the outcome and effects of the experiment. This looks to assess the questions posed at the beginning of the report.

Finally, a conclusion of the study is done in order to meaningfully validate the results and give an overall impression on the systems performance linking it to the hypotheses stated above. Recommendations and future work on this topic and system is then discussed.

Figure 1.1 below summaries this:

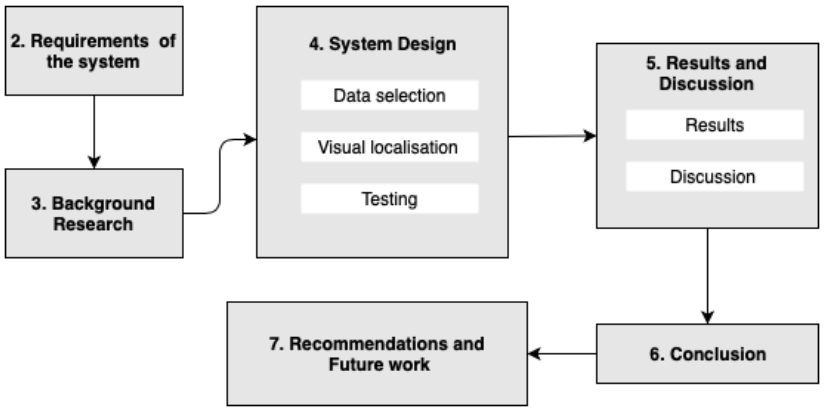


Figure 1.1: Figure illustrating the plan of development for this report

Chapter 2

Requirements

2.1 Specifications

2.1.1 User Requirements

Table 2.1: User Requirements

ID	Requirement
UR1	Input data-set images
UR2	Match images with database
UR3	Work under different conditions
UR4	Robust and accurate

The end-user requirements are an accurate visual localisation pipeline that can be used in various situations as outlined in the table above.

2.1.2 Functional Requirements

The functional requirements are based on the user requirements.

Table 2.2: Functional Requirements

ID	Requirement
FR1	The system needs to attain images under different conditions
FR2	The system should match images correctly and establishits location relevant to the database
FR3	The system should maintain the correct output under different conditions such as weather
FR4	The localisation must be accurate

2.2 Design Requirements

The pipeline will need to store a database of images and then input images in which the two databases will be compared. The input images will act as "live" images as if it was taken at that moment in time by a robot. The results of the system are driven by the requirements outlined above.

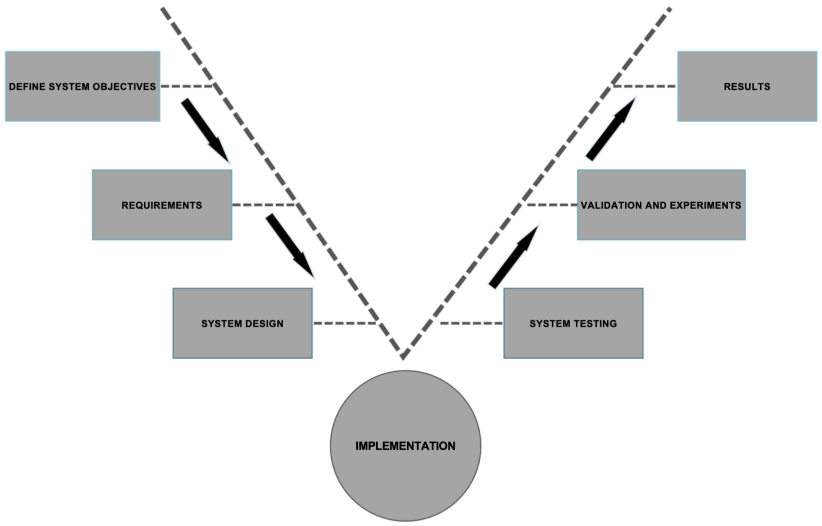


Figure 2.1: V-Diagram illustrating the design process

2.3 Known Challenges

Visual localisation is difficult to implement as the pipeline need to be accurate to ensure that the results of the study is relevant and exact. Image matching can often yield incorrect matches through errors which is why the testing and validation needs to cover

all possibilities. The data-sets being used should be relevant and of nature of what if expected to be obtained from a robot.

Chapter 3

Background Research

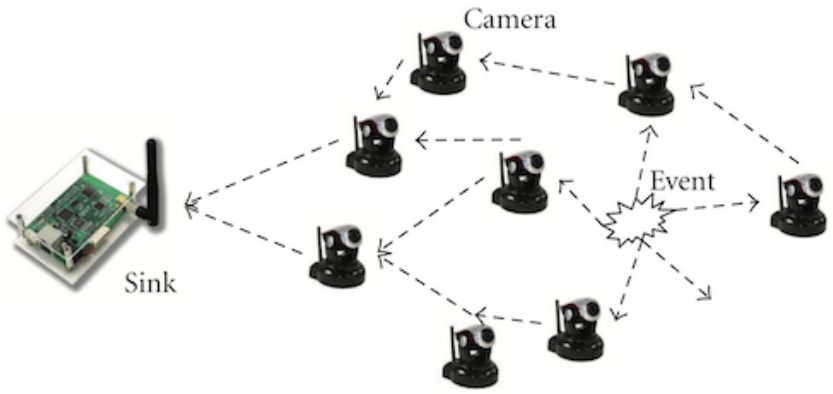
Visual navigation is an essential tool used in robots in the present and will continue to be used more regularly in time. The biggest issue in self-moving robots is localising itself in an environment and establishing a trajectory. To solve this issue, more sensors and data are being used to allow the robot to work more efficiently and accurately. Different methods are developed in order to allow robots to use sensors to attain data such as images and videos and use that in order to navigate itself.

This chapter will discuss methods used to achieve this, which will lead to various options available to implement for this study. Numerous articles, papers and textbooks were assessed to write this chapter. The literature that was found to be relevant to this research is referenced in the bibliography.

3.1 Image Acquisition

Image stitching is a computer vision application where it is the process of combining multiple images which overlap with one another. It then combines to become a seamless photo mosaic [2]. It forms sort of a panoramic image but requires a myriad of regular photographic images to be able to view the whole space. In [2] showed that photographs must be taken using a camera which is rotated 20 degrees after each image is taken. At least an overlap of 50 percent needs to be maintained between corresponding images. Alternatively, a Visual Sensor Network (VSN) can be created by using multiple camera which are pointed at different direction angles which can cover up to 360 degrees. It works with a wireless network where all the images captured is taken in by the sink and

transferred as shown in Fig. 3.1 [3].



1.png

Figure 3.1: A diagram illustrating a Visual Sensor Network [3]

Image matching is used to combine images taken on the same scene by different sensors at different times and viewpoints.

3.2 Conventional Methods for Image Matching

- Professor Barnea proposed the idea of a similarity detection algorithm (SDA) in 1972. It has a fast template matching speed at the cost of matching precision, besides, it is susceptible to noise interference [4] [5].
- Harris corner point detection algorithm, small univalue segment assimilating nucleus (SUSAN) corner point detection algorithm, and iterative closest point (ICP) algorithm. It does not feature too much of information but it does have strong robustness and can be used for many applications [8] [9].
- Normalised gray degree correlation algorithm, Fourier-Mellin phase transformation correlation algorithm and Levenberg-Marquardt optimization algorithm is found in [6] [7]. It has a high matching precision although it does have a high computation overhead, time consumption and complexity.
- Algorithms based on regional primary colour features, edge detection, neural network, wavelet transformation and image segmentation. It is found to have good precision, however the matching process is complex and universally poor [8].

One of the most efficient algorithms that was found us the scale invariant feature transform (SIFT) [10]. It was proposed by David Lowe in 1999. This approach transforms an image into a large collection of local feature vectors, each of which is invariant to image

translation, scaling, and rotation, and partially invariant to illumination changes and affine or 3D projection [10].
The principle of the algorithm detects a series of key points from a multiscale image representation. Each key point has a centre position (x, y) and a characteristic scale σ . SIFT computes the dominant orientation θ over a region surrounding these key points [11].
For each key point, (x, y, σ, θ) defines the centre, size and orientation where the SIFT descriptor is computed. SIFT key point descriptors are in theory invariant to any translation, rotation and scale change [11].

The algorithm consists of five major steps:

1. Construction of SIFT scale space, the images are convolved with a Gaussian filter.
2. Detection of SIFT feature points.
3. Allocation of key point principal.
4. Calculation of feature descriptors.
5. Matching of feature points.

The scale space of the image is defined as L(x, y, σ) and the Gaussian function is :

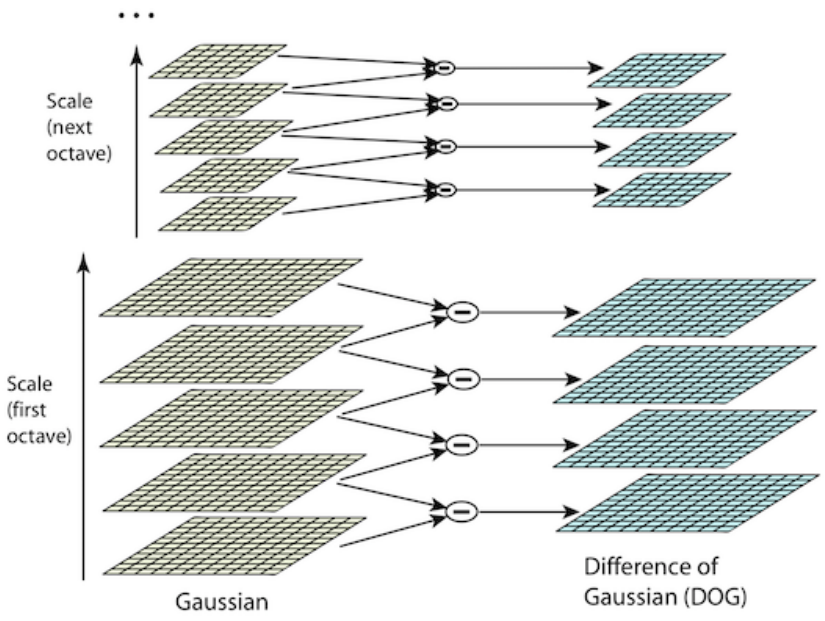
$$G(x, y, \sigma) = e^{1-(x^2+y^2)/2\sigma^2}$$

The input image is I(x, y). Convoluting this with the Gaussian function will give the scale space image. The difference-of-Gaussian function is defined as

$$D(x, y, \sigma) = (G(x, y, k\sigma) - G(x, y, \sigma))\bullet I(x, y)$$

$$= L(x, y, k\sigma) - L(x, y, \sigma)$$

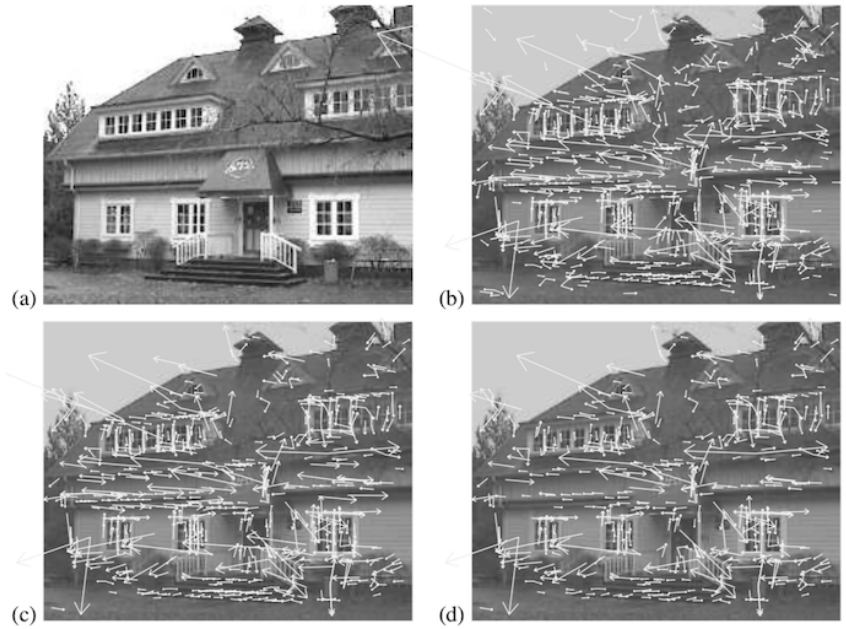
[10]



2.png

Figure 3.2: For each octave of scale space, the initial image is repeatedly convolved with Gaussians to produce the set of scale space images shown on the left. Adjacent Gaussian images are subtracted to produce the difference-of-Gaussian images on the right. After each octave, the Gaussian image is down-sampled by a factor of 2, and the process repeated. [10]

In step 1, key points are localised to the nearest pixel depending where the features were found in the scale-space. In the second step, the algorithm refines the location of the feature points to sub-pixel accuracy while removing any poor features. The sub-pixel localization proceeds by fitting a Taylor expansion to fit a 3D quadratic surface (in x, y and σ) to the local area to interpolate the maxima or minima [12].



3.png

Figure 3.3: This figure shows the stages of key point selection. (a) The 233x189 pixel original image. (b) The initial 832 key points locations at maxima and minima of the difference-of-Gaussian function. Key points are displayed as vectors indicating scale, orientation, and location. (c) After applying a threshold on minimum contrast, 729 key points remain. (d) The final 536 key points that remain following an additional threshold on ratio of principal curvatures. [10]

An orientation histogram is formed from the gradient orientations of sample points within a region around the key point. The orientation histogram has 36 bins covering the 360 degree range of orientations. Each sample added to the histogram is weighted by its gradient magnitude and by a Gaussian-weighted circular window with a σ that is 1.5 times that of the scale of the key point [10].

The final stage of the SIFT algorithm is to generate the descriptor which consists of a normalised 128-dimensional vector. At this stage of the algorithm, we are provided with a list of feature points which are described in terms of location, scale and orientation. This allows us to construct a local coordinate system around the feature point which should be similar across different views of the same feature [12]. A feature vector is found and then reduced to a unit length.

[2] explored this method and the following results were attained as shown in Table 1.1:

Table 3.1: Image Quality Analysis

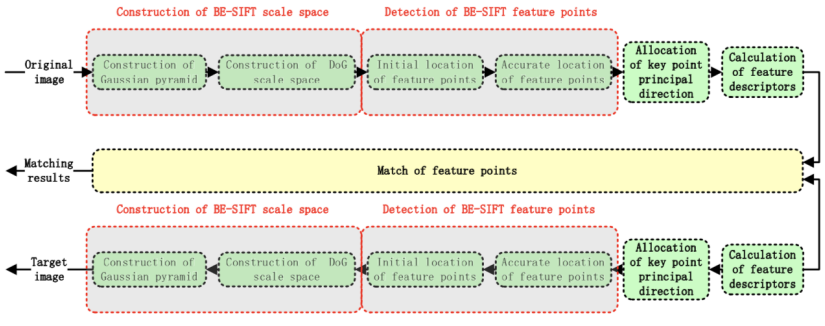
Type	Size(KB)	PSNR(db)	Entropy
Original Image	170	–	0.068425
Compressed Image (Rank 50)	85.6	27.0752	0.056359
Compressed Image (Rank 60)	88.7	27.9953	0.059114
Compressed Image (Rank 70)	91.3	28.8329	0.061192
Compressed Image (Rank 95)	95.2	30.6217	0.063911
Compressed Image (Rank 120)	97.4	32.0030	0.065805

SIFT has some downfalls as papers [4] [13] find that SIFT blurs a lot of details of images and generates isotropic Gaussian scale space.

3.2.1 Proposed improvements of SIFT

The following is methods done by researchers in attempt to improve the inefficiencies of SIFT that was described above. These papers use SIFT as the main basis of their algorithm, tweaking certain steps to improve SIFT.

- BE-SIFT is a proposed more efficient SIFT algorithm researched in [4]. This method includes all five stages that takes place in SIFT but more emphasis is put on stage 2,3 and 4. They make use of the Affine Covariant Features dataset provided by Oxford VGG Group.



4.png

Figure 3.4: Block Diagram of BE-SIFT

- An Improved Anisotropic Gaussian-SIFT (IAG-SIFT) algorithm is assessed in [13]. The standard SIFT method generates a isotropic Gaussian scale space which tends to blur images, this method aims to eliminate that from happening

- [18] researches the prospect of SIFT simplifying and matching algorithm improvement. This paper simplifies the descriptors as used in Lowes [10] where he used 4x4 sub-regions, this research uses $2 \times 2 + 1$ sub- regions. On the matching, Lowe calculates descriptor's nearest neighbour and next nearest neighbour distance whilst the IAG-SIFT divides neighbour region into four sectors and a ring, forming a 32+8 dimensional feature descriptor. The results from this paper show an improvement in image rotations, blur, affine transformations and the computational time is reduced by around 50 percent on average compared to OpenCV SIFT's computational time.

3.3 Different Concepts

3.3.1 FAB-MAP

FAB-MAP written by Mark Cummins and Paul Newman introduces a completely new method compared to SIFT [14]. FAB-MAP is a probabilistic appearance based simultaneous and localisation mapping system. This algorithm uses the Bayes Filter to compute the probability of being at a specific location given the history of observations. The visual features (vocabulary) is created by clustering speeded up robust features (SURF) extracted from images [15]. The visual vocabulary model treats an image as a "bag of words" much like a text document, where a "word" corresponds to a region in the space of invariant descriptors [14].
The joint distribution of the words in the vocabulary is approximated using a tree-structured Bayesian Network. The network is learned using the Chow-Liu tree algorithm that gives an optimal approximation of the actual joint distribution from all possible tree-structured networks [15]. A few probability schemes are defined below:

1. Sets of observations are conditionally independent given position:

$$p(Zk|Li, Zk-1) = p(Zk|Li)$$

.

2. Detector behaviour is independent of position:

$$p(zj|ej, Li) = p(zj|ej)$$

.

3. Location models are generated independently by the environment.

Mark Cummins and Paul Newman [14] concluded that their results from this research proved positive as the system proved to be robust for loop closure detection. They compared their solution to a simultaneous localisation and mapping (SLAM) technique.

3.3.2 Semantic Label

When using the SIFT algorithm, the scene needs to stay the same for detection meaning that a chance in light, weather or colour can produce incorrect results. Semantic labelling tries to eliminate this and uses detailed feature descriptors. [16] Introduces adopt the pipeline show in the figure below:

5.png

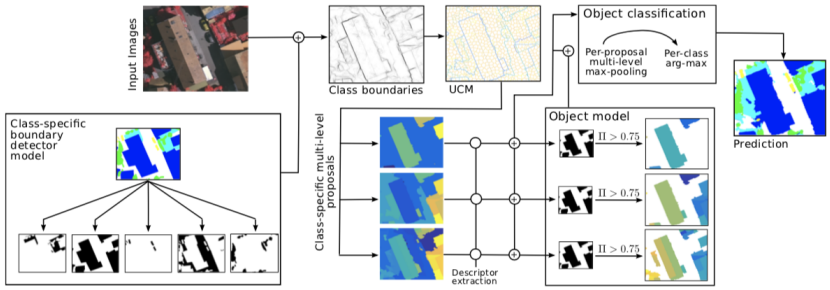


Figure 3.5: Pipeline of the proposed class-specific object extraction and classification framework

The class specific takes in an image and identifies the boundaries of it – boundary descriptors. This allows the image to be smoothen out which can lead to a Ultrametric Contour Map (UCM) mapping. It can build a map using SIFT methods but then change the descriptors to semantic labels.

3.3.3 Simultaneous Localisation and Mapping

One of the most common techniques used today is SLAM (Simultaneous Localisation and Mapping) algorithm which creates a model of its surroundings where it can then build a map and estimate its state at the same time [19]. All landmarks in the environment is estimated in real time and no prior knowledge of the location is required.

SLAM consists of several functions such as landmark extraction, data association, state estimation, state update and landmark update. SLAM uses a similar probability distribution that was discussed for FAB-MAP in order to build a map of its surroundings [20]. The main component of the SLAM process is an EKF (Extended Kalman Filter) which is responsible for updating the map based on features. It is also responsible for estimating uncertainty in the robots positioning and the features found [20].

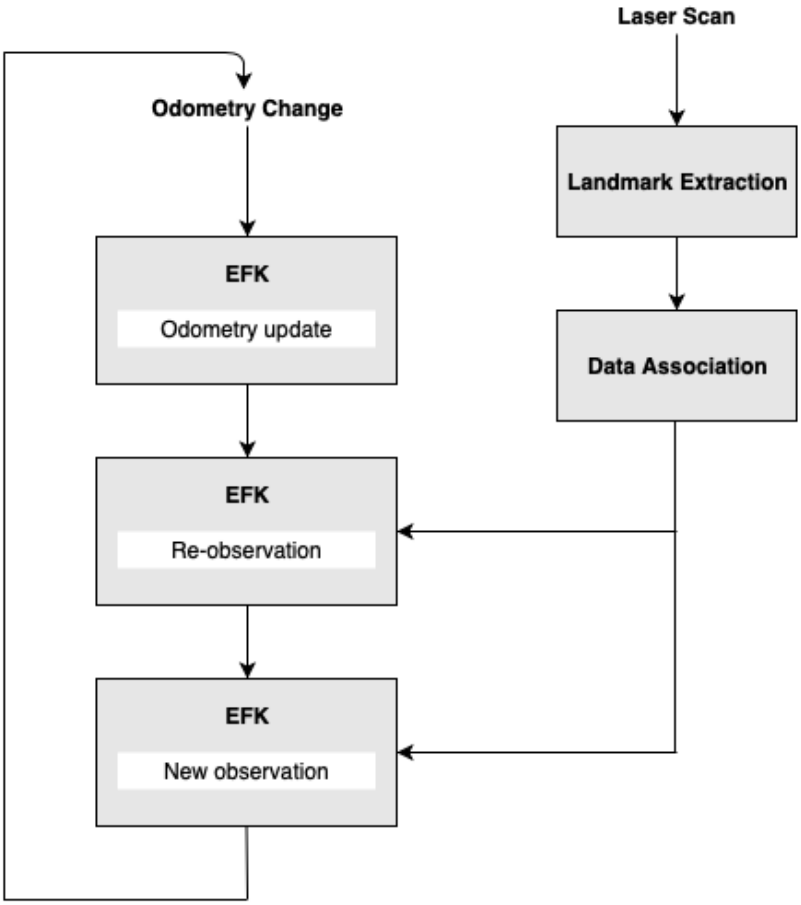


Figure 3.6: Figure Illustrating the SLAM process

The odometry deals with determining the robots exact position measured from the change in direction. Landmarks are attained by using a laser. The laser covers the environment surrounding the robot, it then measures the distance and features of the objects found by the laser. landmarks should be re-observable so it can detected from different angles and positions but unique enough to not mix them up at different times. This is able to develop data on the results found and then change the odometry.

To the describe the motion of a robot, EFK defines it as:

$$P(x_k|x_k - 1, u_k) \longleftrightarrow x_k = f(x_k - 1, u_k) + w_k$$

, [21]

where f(·) models vehicle kinematics, w_k are additive, zero mean uncorrelated Gaussian motion disturbances with covariance Q_k. The observation is described as:

$$P(z_k|x_k, m) \longleftrightarrow z(k) = h(x_k, m) + v_k$$

, [22]

where h(·) models the geometry of the observation, v_k are additive, zero mean uncorrelated Gaussian observation errors with covariance R_k.

[23] found that a SLAM system has to be accurate and efficient because it can be dangerous if the system calculates an incorrect environment in real-time and misguides the robot.

3.3.4 Convolutional Neuro Networks

Artificial Intelligence is the new and upcoming feature in robots nowadays. It a hugely researched field at the moment as it bridges the gap between humans and machines. Computer vision and robot navigation branches from this and the idea of combing the both is essential.

A Convolutional Neural Network is a new method that enables image classification, object detections and matching. It takes in an image as an input which it then converts into a matrix of pixels in grayscale [25].

It makes use of deep learning – part of machine learning methods which us used for artificial intelligence networks. Once an image is input into the system, it is then passed through a series on convolution layers. The process is shown in the figure below:

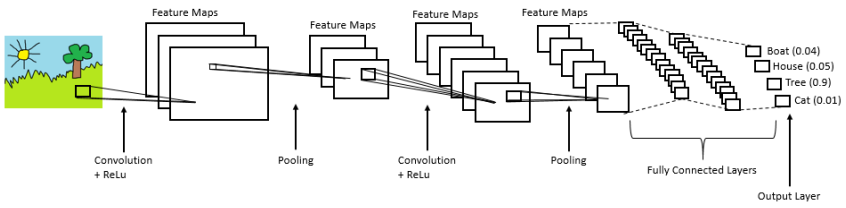


Figure 3.7: Complete CNN architecture

In order to extract features from image, the convolution layer is the first to do so. Each layer is made of a set of neurons which is connected to another set in the next layer. The output of each layer is the input into the next. It learns image features and edge detection as more images are assessed. Each layer has 3 dimensions: height, width and depth.

ReLU (Rectified Linear Unit) is a non-linear operation which is the layer responsible to make sure the convolution of layers is correct and to eradicate any non-linearities.

The pooling layer reduces the dimension size when adding all the feature map together. Doing this reduces the computational power required to process data. It is also a strong feature dominant feature extractor.

CNNs are becoming more popular to use and studies found nowadays on this topic use the concept of CNN as a base to create new algorithms or combine it. More research is done on this in [26].

Chapter 4

Methodology

4.1 Data Selection

The Oxford Robocar Dataset was selected for this project. This data was captured in 2015 in the streets of Oxford. The Robocar captured images in frequent intervals of its route along with the GPS locations at regular intervals.

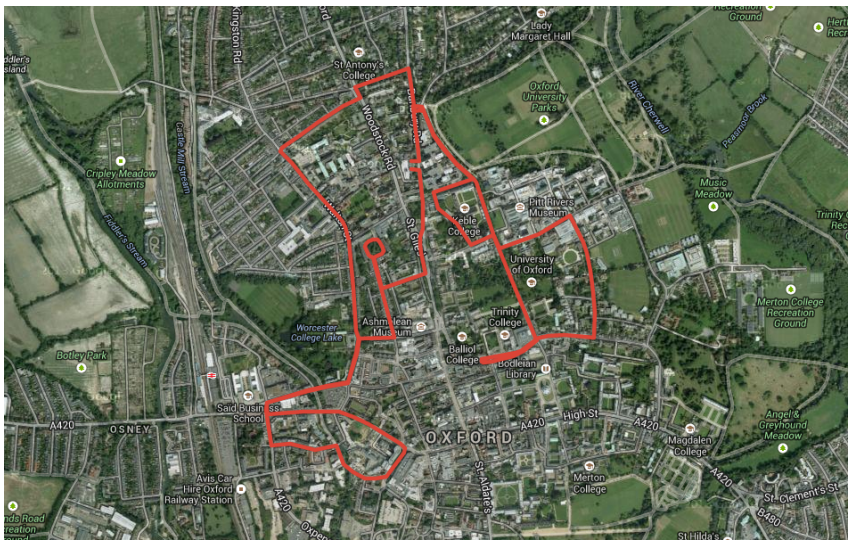


Figure 4.1: Map of the route used for dataset collection in central Oxford [24]

Thousands of images are accumulated once the route is completed, therefore only a subset of it was used for testing due to the time constraints. The images used were sized at a width of 1280 pixels and height of 960 pixels as a PNG (Portable Network Graphics)

image. PNG is a file format used for lossless image compression. This means that it restores all of the image information when it is decompressed upon viewing. These images are of high quality and does not do save much storage space.

The subsets used in the route covered approximately 300 meters to ensure that image environment differed from one another. The GPS locations from the dataset were not taken as frequently as the images which meant to gain the location for each image, the GPS co-ordinates had to be interpolated in order to gain the positions. Therefore, to ensure accuracy, the subsets taken were generally a straight line journey as interpolating it would yield more accurate GPS locations.

The data offered by Oxford Robocar has various weather conditions of the route which can help test the robustness of the pipeline as well as explore how image compression will affect it differently with the conditions.

Figure 4.2: Images at the same location under different conditions [24]

4.2 Visual Navigation

The visual Localisation pipeline will be set up such that two different datasets will be used. One data set will act as a "database" in which these images of the route will be stored along with its respective GPS co-ordinates. The second dataset will act as "live images" as if it was getting fed directly from the sensors of a robot with its GPS co-ordinates – in this case an autonomous car. The "database" will consist of images running through a route of approximately 300 meters and the "live" set will be random

images within that route in which will be matched with the “database”.

To speed up the process, the GPS co-ordinates will be incorporated to help the program choose a certain radius of the “database” route to find a specific image appose to running through the whole database. This will be achieved by locating the nearest “database” co-ordinates with the co-ordinates of the “live” image. By doing this, the live image will only be compared to a subsection of the database which has the relevant images to compare it to opposed to comparing the “live” image to the whole database. This will save a great amount of time. A visual illustration of this method is shown in the figure below:

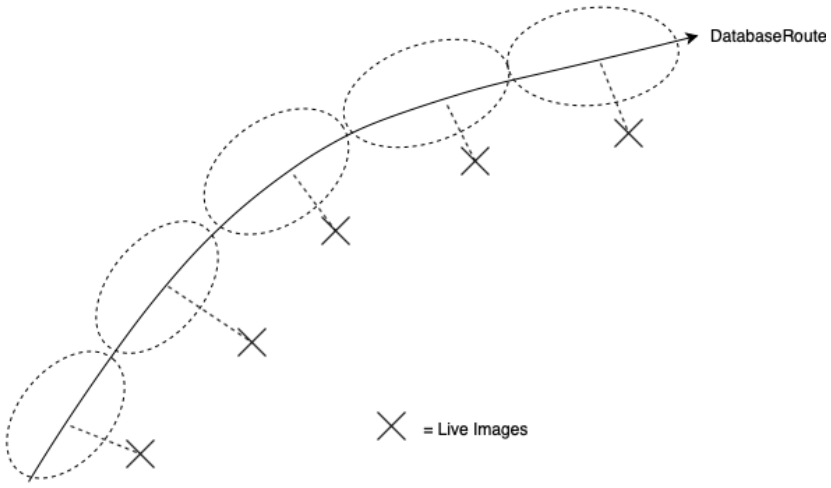


Figure 4.3: Demonstration of matching comparisons

Once the nearest co-ordinates of the database is found from the “live” image, it will then compare that image to a radius of roughly 30 meters as shown in the diagram above. The objective of this is to ensure that each “live” image has an opportunity to be compared to an image that is not similar to it while still succeeding to accelerate the visual Localisation process. Creating this opportunity allows the prospect of an error to occur. Any error that occurs will show a deficiency in the pipeline and will prove vital once the testing of images under compression in the pipeline occurs.

The database is prepared by selecting a subsection of a route taken from the Oxford RobotCar Dataset. The database needs to consist of images that would not have bias on an particular weather condition or time. Therefore, these images need to be clear, without an anomalies such a vehicles, people or conditioning. Once this is found and put together, it can be stored in the pipeline. The images used for the “live” images can be of any manner provided it falls within the database route.

4.2.1 Image Matching

To compare each image in the route, the Scale Invariant Feature Transform (SIFT) algorithm will be implemented as introduced in Chapter 2. SIFT is used to detect and describe local features in images. It locates key points which it then allocates it with quantitative information called descriptors. Key points considers four parameters: the centre co-ordinates (x, y), the scale and orientation and denotes each point. The descriptors describe how the local gradients around the key points are aligned at different scales.

Python will be used as the chosen programming language used for this project and implementation. Python offers many libraries which can assist in achieving the implementation of the image matching and the overall pipeline. Various python libraries are used in order to implement the image matching technique. The main library used is *OpenCV* which is designed to process and solve computer vision problems. Below is the pseudocode used to implement image matching for the visual localisation pipeline.

Algorithm 1 Visual Localisation Pipeline

```
1: Input: database_images array, live_images array, GPS coordinates
2: Interpolate Database GPS co-ordinates
3: Match closest GPS co-ordinates between the two image sets
4: Manipulate Image sets by compression or noise
5: for i do in live_images
6:     for j do in (database_images within 10m of live_images[i])
7:         keypoint1, descriptors1 ← image[i]
8:         keypoint2, descriptors2 ← image[j]
9:         matches ← Closest match between descriptors1 & descriptors2
10:        Set threshold                          ▷ Value to define accuracy required
11:        if (distance of matches) < threshold * (Distance of matches) then
12:            good_match++
13:        similarity = len(good_matches)/num_of_keypoints * 100
14:        if similarity > best_match then
15:            best_match = similarity
16:        gps_match[i] = best_match
17: Determine distance of live_images route
18: Determine distance of gps_match route
```

The best match of an image from the database and a live image is found in step 5 from the code above. It then records the GPS location found at that database image and appends it to an array. A metric used for this study can be established as the outcome of the code above will demonstrate the route found from the visual localisation pipeline and compare it to the actual route from the live images. If the pipelines route does not match

the route from the GPS of the live images, the pipeline has an error in its matching and therefore has inaccurate results.

The images are converted to gray-scale when being compared. This allows for better detection of important edges or features as it notes lamination better. Grayscale also possesses less complexity compared to colour which can assist the processing speed and algorithm to determine similarities. 4.4 shows two images taken at the same location and different times.

Figure 4.4: Two Images at same location and different times

Once these picture are compared using SIFT as described in the pseudocode above, the following feature matching result is produced.

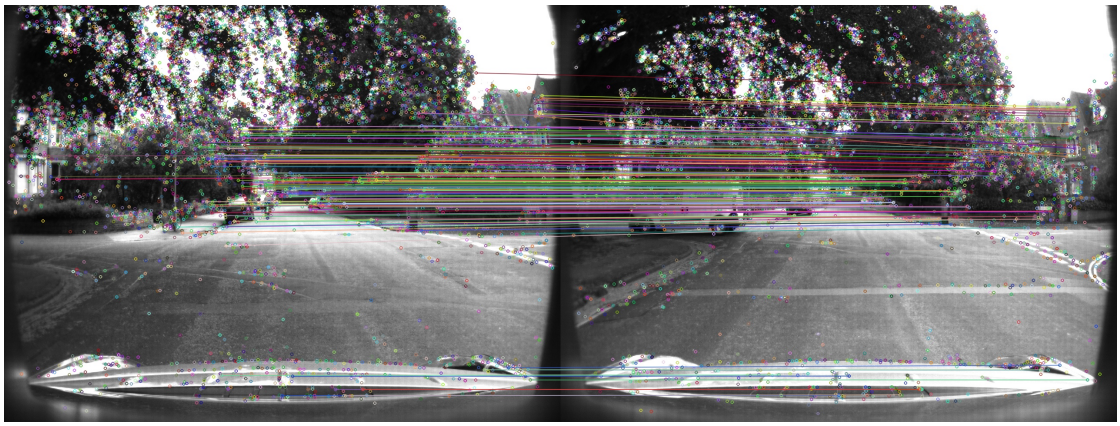


Figure 4.5: Feature matching of the images in 4.4

These are the "good" matches found from implementing the SIFT algorithm. This example has produced 223 good matches as highlighted in the coloured lines in 4.5.

4.2.2 Image Manipulation

Different weather conditions are assessed in the separate datasets being used. To further test the systems robustness and error margin, noise and illumination changes will be evaluated on the pipeline. Noise will be present in the real system if there is a malfunction in a sensor or the data becomes corrupt and a higher illumination can be caused when there is a reflection of light on the camera or directly from the sun itself.

A Gaussian noise is added to the image to manipulate the image. This is a random variable that has a normal distribution which is controlled by its mean and standard deviation. This creates an array of random numbers which is added to each pixel of the original image. A python library assists implementing the noise. *NumPy* is a library that helps with the computation of arrays used in python coding. This library adds the noise array to the pixels of the original image and also clips the image to a to the specified range of pixels. The pseudocode to add Gaussian noise to an image is provided below.

Algorithm 2 Addition of Gaussian Noise

1: **Input:** image filename
2: Create array representation of image
3: $mean = 30$
4: $standard\_deviation = 50$
5: noisy_image $\leftarrow$ image + noise
6: noisy_image is clipped between 0 and 255
7: **Output:** noisy_image

Figure 4.6: Noise added to an image

The above figure demonstrates the effects that Gaussian noise has on an image used in this study. The effects of the noise can be controlled by adjusting the mean and standard deviation value.

4.2.3 Image Compression

The method of JPEG (Joint Photographic Experts Group) compression will be used to compress the images involved in the pipeline. JPEG utilizes a lossy compression scheme meaning that it does not retain its original quality. The quality of an image can be set between 0-100.

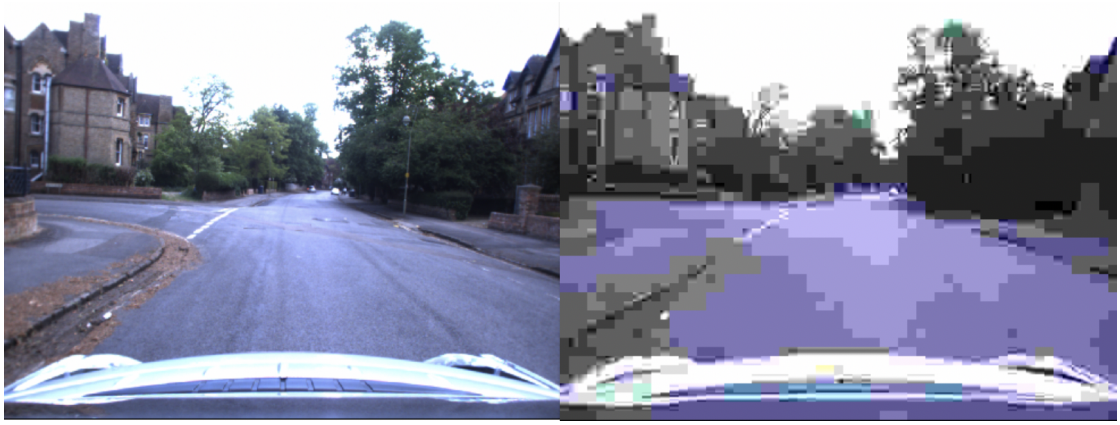


Figure 4.7: Example of an image converted to 1% quality

Table 4.1: Image size for different image qualities

Image Quality(%)	Size (Kilobytes)
100	759
50	144
10	43
1	28

Table 4.1 showcases the image size along with its respective image quality. It shows that as the image quality decreases, the size will also decrease. The 100% quality image is the original image that is classified as a "live" image in PNG format. The following qualities are JPEG and converted using a an *OpenCV* function. The difference in image size once the quality is decreased proves to be substantial and which creates the opportunity to make a great impact on this study if the pipeline is able to produce a low error margin for the visual localisation when the images are compressed.

4.3 Testing

The purpose of this study is to observe and understand how compressing the data images will affect a visual localisation pipeline. Compressing images will reduce its size which can lead to better efficiency and more data being available for the system to utilize.

The testing will begin by running the pipeline without any compression or image manipulation during different weather or time of day conditions. This will test the performance and robustness of the visual localisation pipeline. A satisfactory performance will provide a good benchmark to evaluate effects of the images when it undergoes compression.

Extra illumination will be added to the images as part of another test, those images will also be compressed. Added illumination could be caused by any glare due to sunshine, lighting or reflections that are directed at the camera or sensors being used to capture the images. Adding noise to the images will also be used for testing as sensors picking up the image or any malfunction in the equipment being used can result in an image gaining noise. Using noise and the different levels of compression, different permutations can be tested to investigate the consequences it will have on the visual localisation pipeline.

Using the techniques demonstrated above, different permutations can be investigated on the accuracy of the localisation pipeline. The results are assessed by tracking the distance covered by the "live" images and that of the matched images from the database. If there is a difference in distance covered, it can be understood that there was an error in the pipeline. This will allow us to understand how much of errors can be caused by the compression and noise of images being used in this system.

The more tests completed, would be beneficial as it will demonstrate the weak spots of the system and what it would take to disrupt the pipeline and prevent it from working accurately or outside of the tolerance accepted by a robots specifications.

Chapter 5

Results and Discussion

The project purpose is to investigate visual navigation under image compression. In order to do so, a single aspect of visual navigation was explored in an image-based visual localisation pipeline which was implanted and designed as discussed in Chapter 4. This chapter will now use the implemented pipeline to run numerous tests to determine the pipelines accuracy.

Distinct weather conditions were tested on the pipeline, which included sun, overcast, snowy, dusk and night-time. Furthermore, these data sets then went under image compression which also included the addition of noise and added illumination. This ensured enough variety in the conditions that would be present to a robot requiring visual navigation. A route of approximately 300 meters were taken and each "live" image is compared to images within a radius of 30 meters of it. The radius allowed a "live" image to be able to be compared to an environment that is dissimilar to it, allowing an opportunity to test if the pipeline will incorrectly match it.

The database is made of images taken from the route on a clear day in which the images are not prone to excess sunshine or shadows. The quality of the images chosen to be tested is the original, 50%, 10% and 1%. A total of 10 "live" images were inputted into the pipeline and the database compromised of 750 images. The Gaussian noise selected had a mean of 30 and standard deviation of 50. The compression of all images was experimenting where the database and the "live" images underwent compression. The image manipulation such as the added illumination and noise was only inserted into the "live" images.

The goal of using different datasets is to survey by what means the image compression of

the data will affect the accuracy of the pipeline. These results can produce an overview of how image compression will fare on an image-based localisation without the experiment being biased to any particular conditions.

PyCharm is used as an IDE (integrated development environment) which will compile and run the code for the experiments. This was all ran by an Apple MacBook Pro 2018.

Table 5.1: Computer Specifications

Description	Details
Processor	2.3 Ghz Intel Core i5
Ram Installed	8GB
Operating System	macOS
Graphics	Intel Iris Plus Graphics 655
Processor Clock	2133 Mhz

5.1 Sun

The first set of "live" images tested consisted of a route that was covered in sunshine. This could raise issues in the pipeline as the images can have the sun reflected of windows or the sun itself directly blinding the camera. Shadows can also cause parts of the image to be unclear.

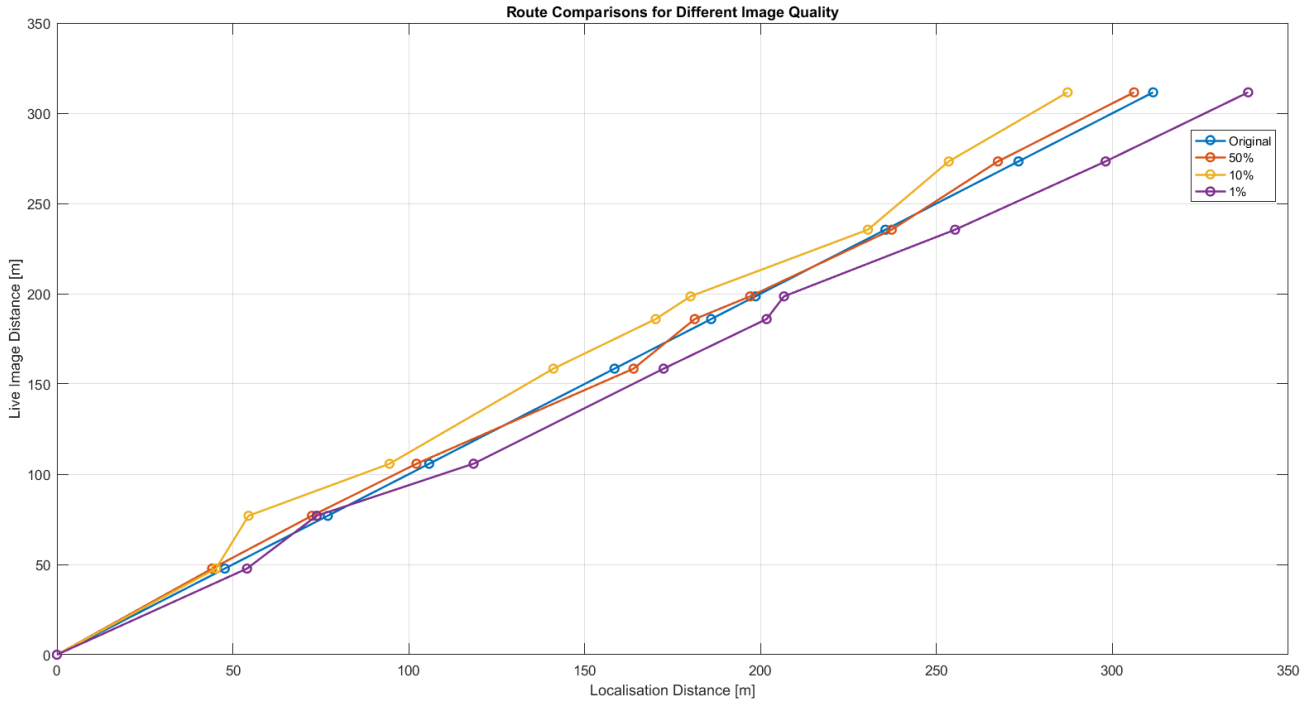


Figure 5.1: Route Comparison

Figure 5.1 Figure 1 demonstrates the results of the distance covered by the "live" images on the y-axis and the distance calculated by the pipeline on the x-axis. All distances are measured in meters and the different qualities are displayed in the coloured lines. The original line is result of the pipeline which includes no compression or image manipulation. Ten "live" images were used as input and each of the images are noted with a circle on the figure where the first image is taken as reference point zero.

The "live" distance is the actual distance being travelled by the robot whereas the pipeline distance is calculated by the system through image matching. As each image is processed, a distance covered from the previous image is calculated and added to the total distance covered. Ideally, both values for image would be the same as that would mean the visual localisation pipeline produced the correct results.

For each image quality, it would have the same "live" distance but the difference as shown above is in the pipeline distance. The distance used for the "live" distance is taken from the GPS readings from the sensors and is used as the images actual position in time. Therefore, all the "live" images used will have the same location for the live distance as it will not be affected by the conditioning. This means that there is an error produced by the pipeline under that image quality. The Appendix further investigates the error

produced for each image under different compression levels from the diagrams presented in this section. This normalisation correlates to a maximum error of 30m as that is the radius in which the "live" images are compared to.

In the figure above, it is shown that the distance calculated from the pipeline per image differs more as the image quality decreases. This is reasonable, because when the quality of the image decreases, it becomes harder for the pipeline to find good matches between the images and therefore cannot accurately locate the correct match. Rarely, the lower quality might perform a better match than a quality that is higher but will be dependent of the conditions and that specific image.

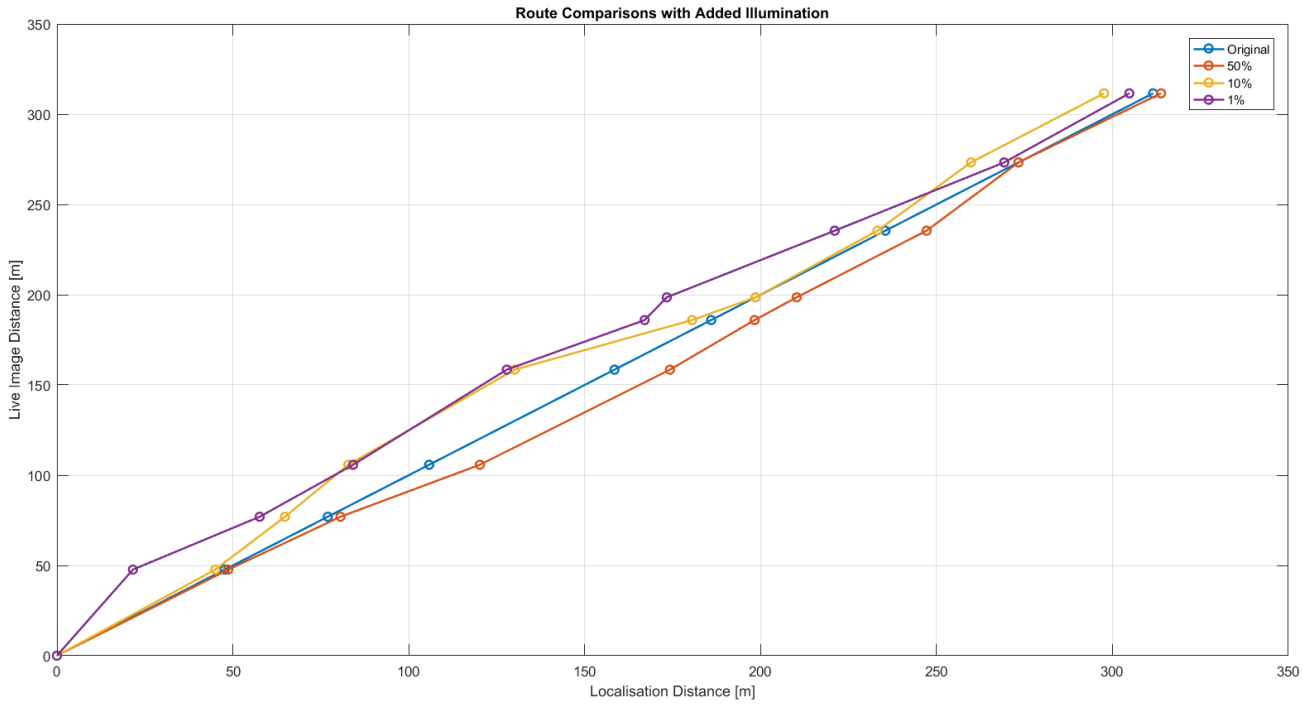


Figure 5.2: Added Illumination to Figure 5.1

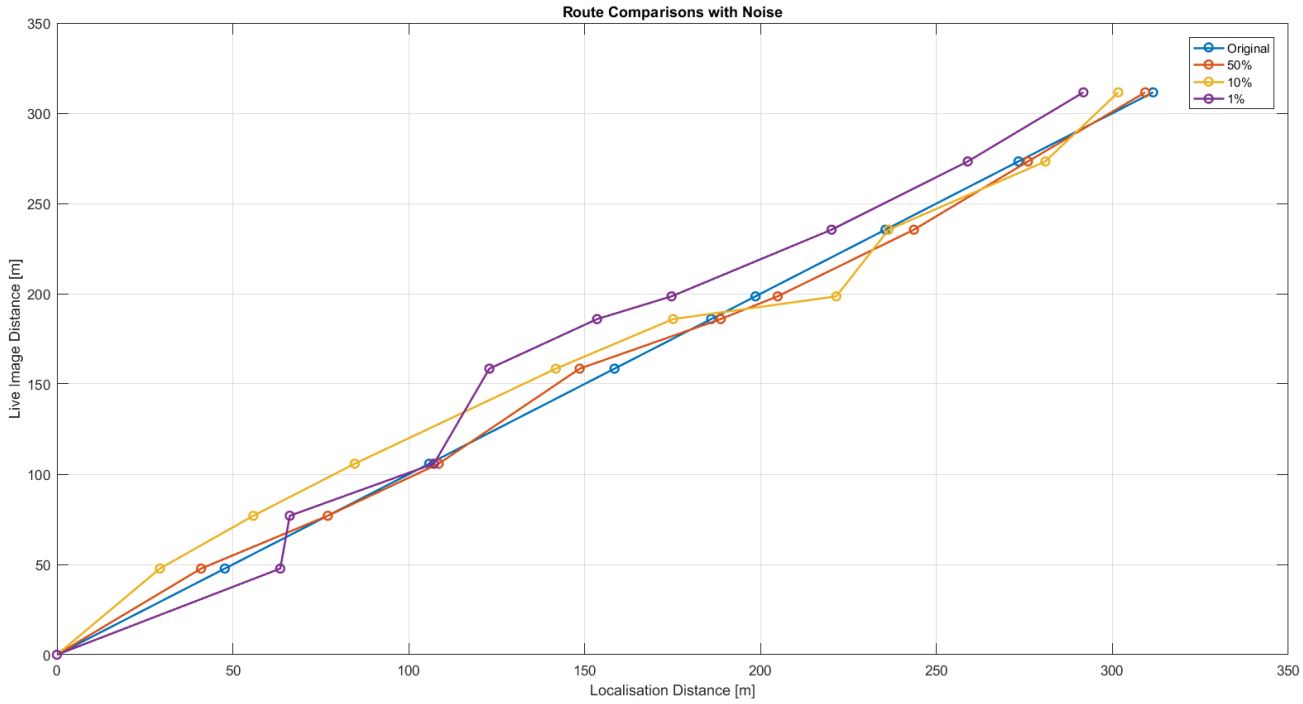


Figure 5.3: Noise added to Figure 5.1

The above diagrams showcase the difference in route of different image qualities compared to the original trajectory. The localisation pipeline worked accurately, being able to correctly match the right images and attain the correct GPS locations even when noise and illumination were added. The errors attained for sunny weather is found to be high. This due to the illumination that is present in the images due to the sun. As seen in Figure 5.2, when extra illumination is added, the error is superior compared to the other conditions tested. This could be caused by the images already containing plenty of light due to the sunny conditions. When added illumination is added, the image becomes significantly illuminated causing issues for the pipeline when trying to match the images.

Table 5.2: Overall Error

Image Quality(%)	Error (%)
50	16.94
10	23.95
1	24.31

Table 5.2 Table 2 demonstrates the overall error found in sunny condition. across all image manipulation techniques. The error percentages are high and affects the pipeline

greatly. Extra illumination played a major role in the overall error as the margins were high that region. This will not be suitable to be used in a real-world visual navigation system as the errors found is quite inaccurate.

5.2 Overcast and Rainy

This dataset was taken on a cloudy day and a slight drizzle of rain was present. This dataset is found in average lighting due to the cloud cover and offers a bit of a variety with the wet surroundings.

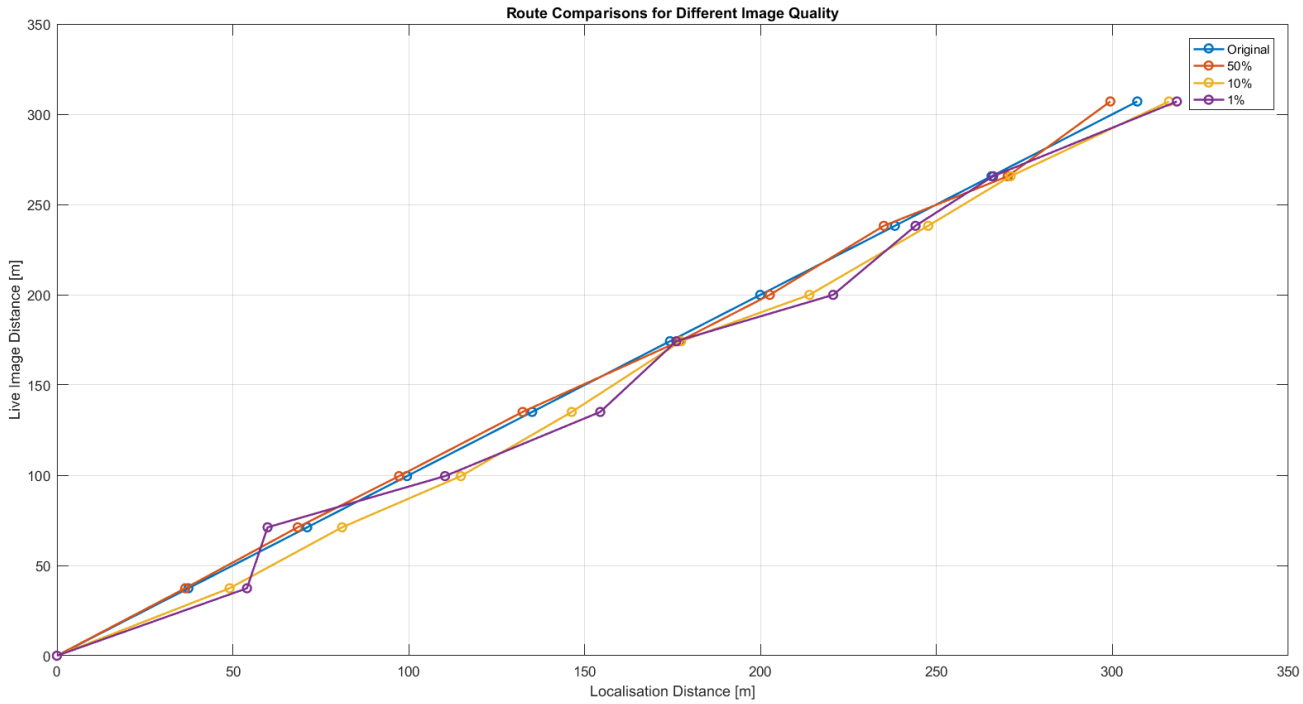


Figure 5.4: Route Comparisons in Overcast Weather

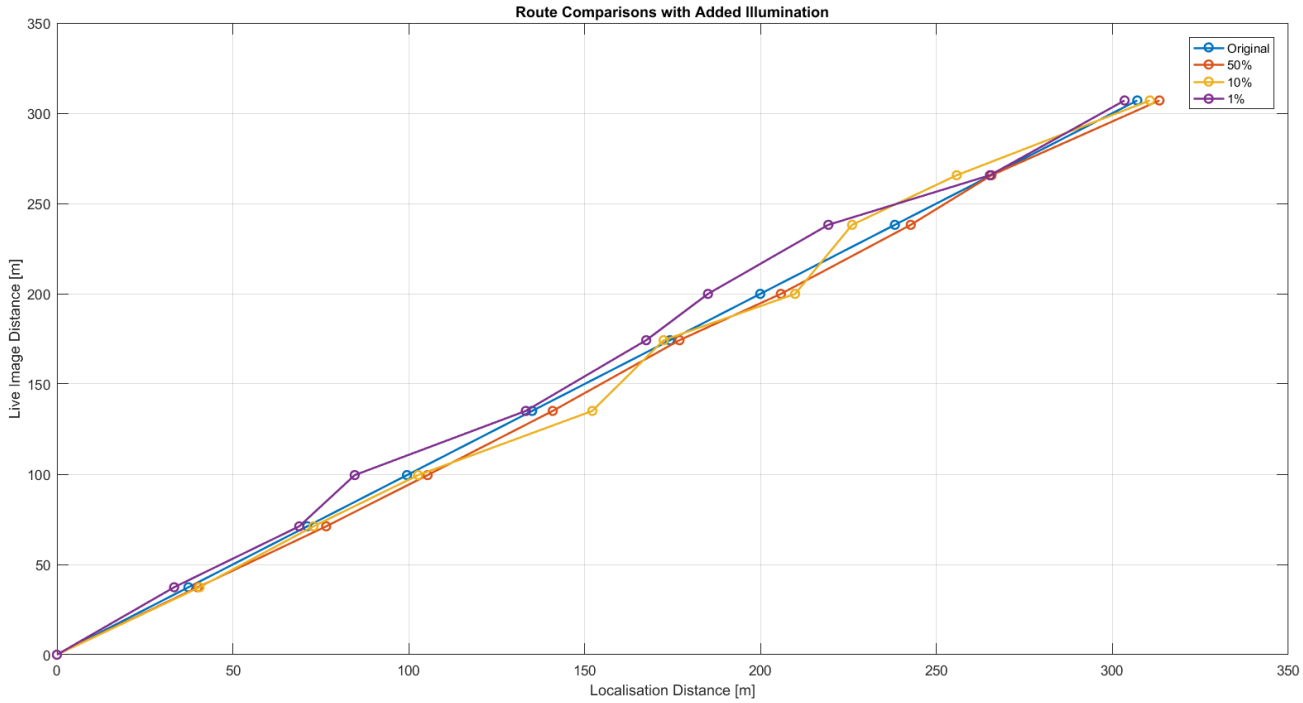


Figure 5.5: Added Illumination to Figure 5.4

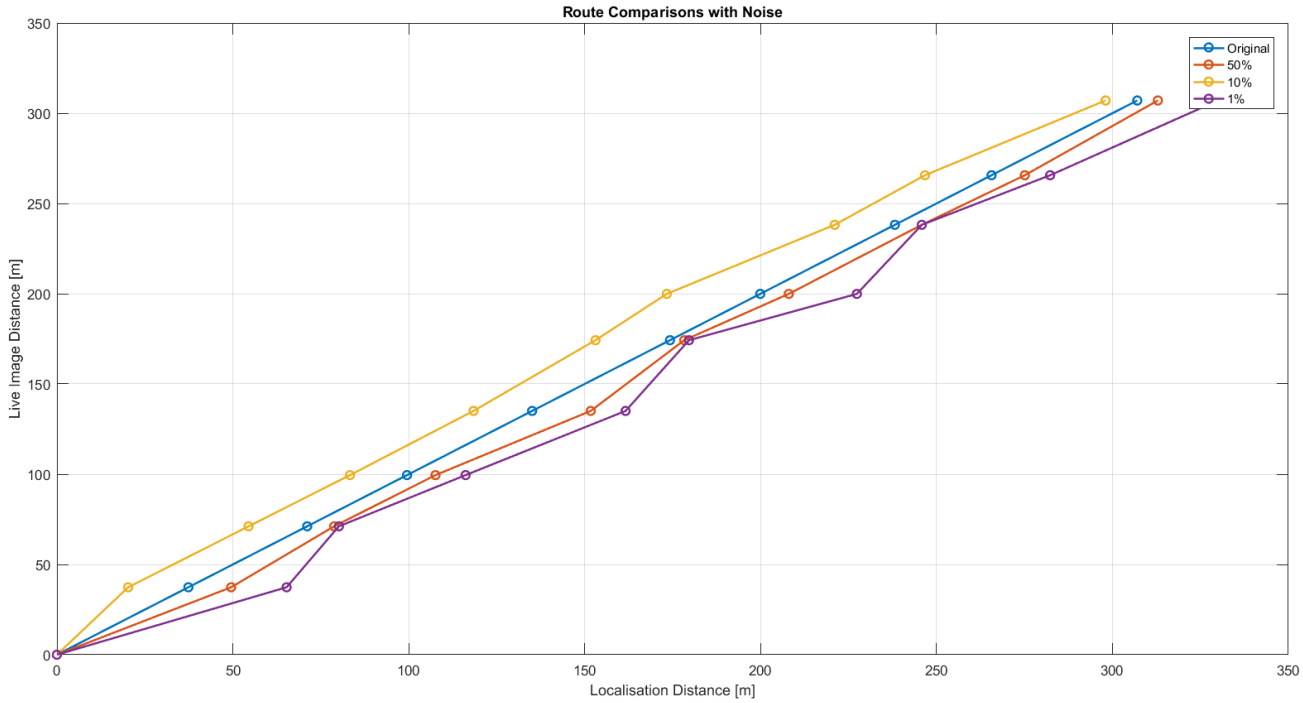


Figure 5.6: Noise added to Figure 5.4

The pipeline managed to match the "live" images precisely with the those of the database which includes when noise and illumination is added. As expected, as the quality decreased, the error margins increase. The increase, however, was substantial for these conditions. The 50% quality was able to stay adjacent to the original trajectory but struggled to maintain this when noise was added to the images. The reasons could be that the surroundings in the wet condition posed irregularities in the environment and when noise is added, disturbs the visuals even more than the other datasets. This provides a complication for the pipeline and therefore does not perform well.

Table 5.3: Overall Error

Image Quality(%)	Error (%)
50	8.54
10	15.36
1	25.41

Overall, the 50% quality images resulted in an error of less than 10% which for the use of this application where a few meters off course is tolerable. The biggest error margins were found once noise was added to the images. This was the biggest contributor to the overall error found for overcast conditions.

5.3 Snowy

Snowy conditions provide an interesting disparity in the datasets as it was taken in the winter where the leaves of the trees have fallen off. The route consists of many trees along the roads and the removal of leaves can change an image significantly. The images also have snow present which can change the appearance and it blocks the ground with a white colour which contrasts from the conditions taken in the summer.

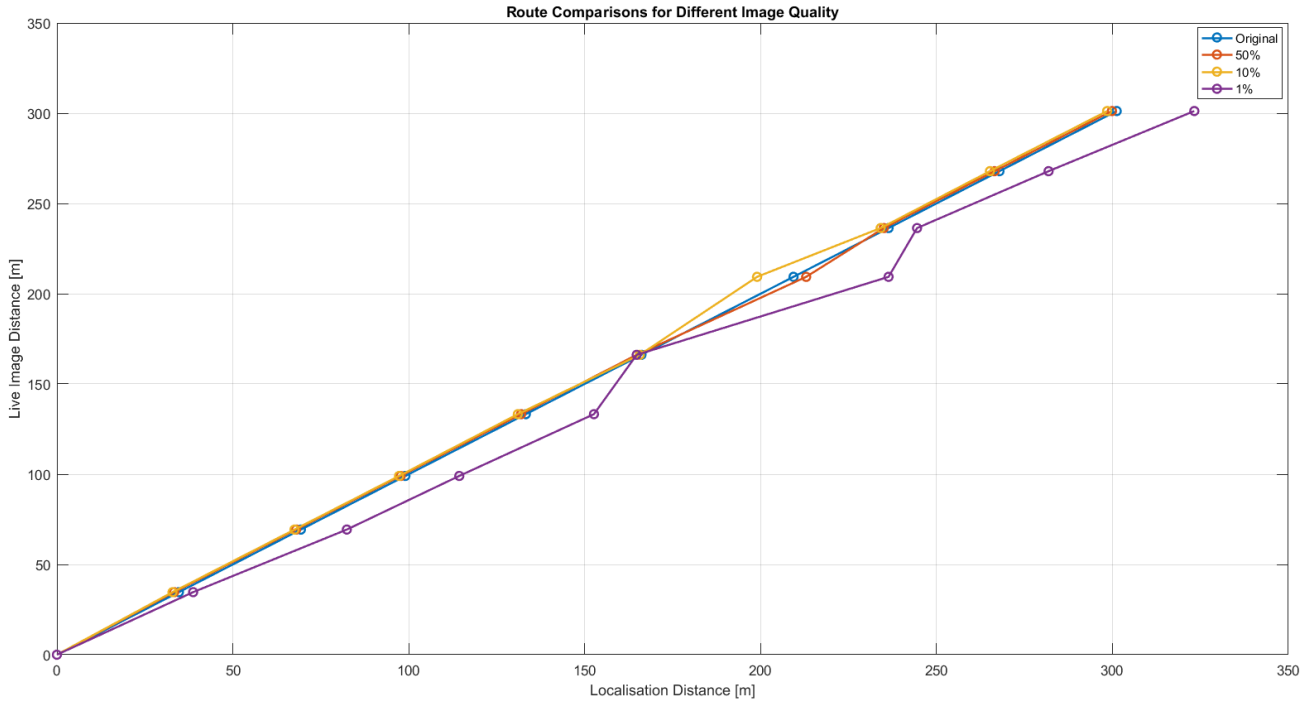


Figure 5.7: Route Comparisons in Snowy Weather

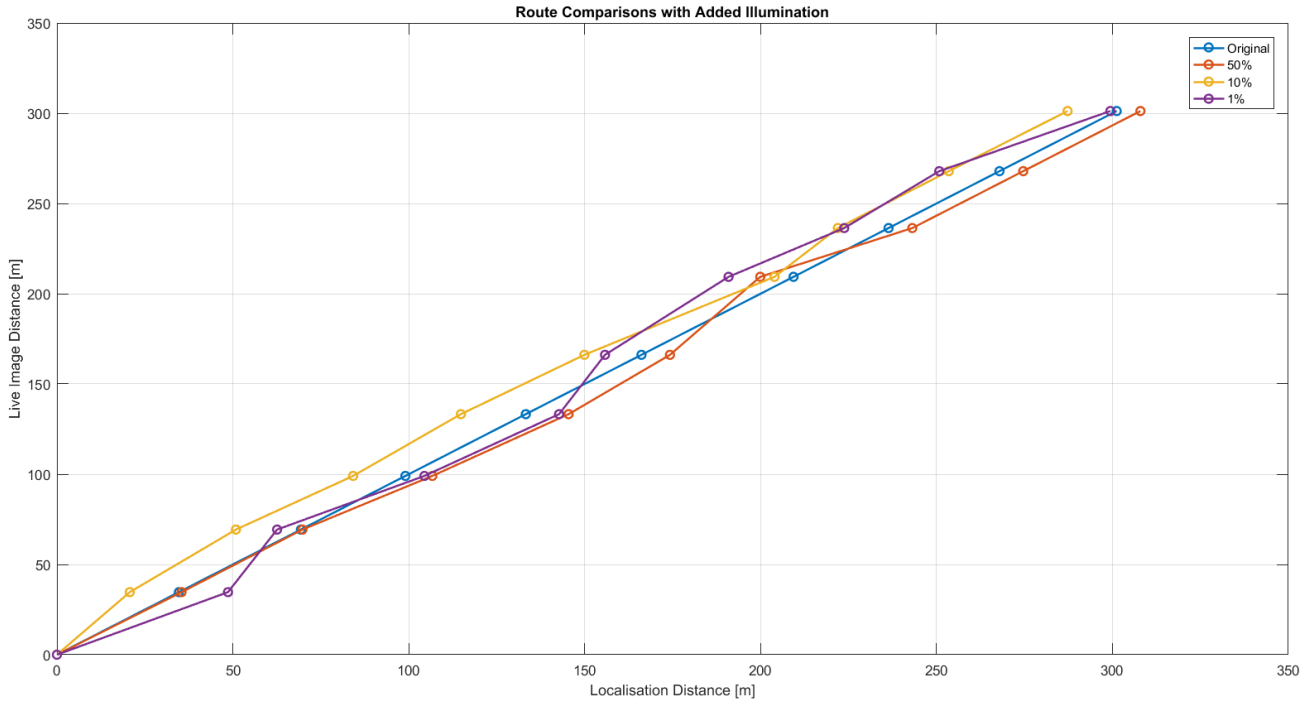


Figure 5.8: Added Illumination to Figure 5.7

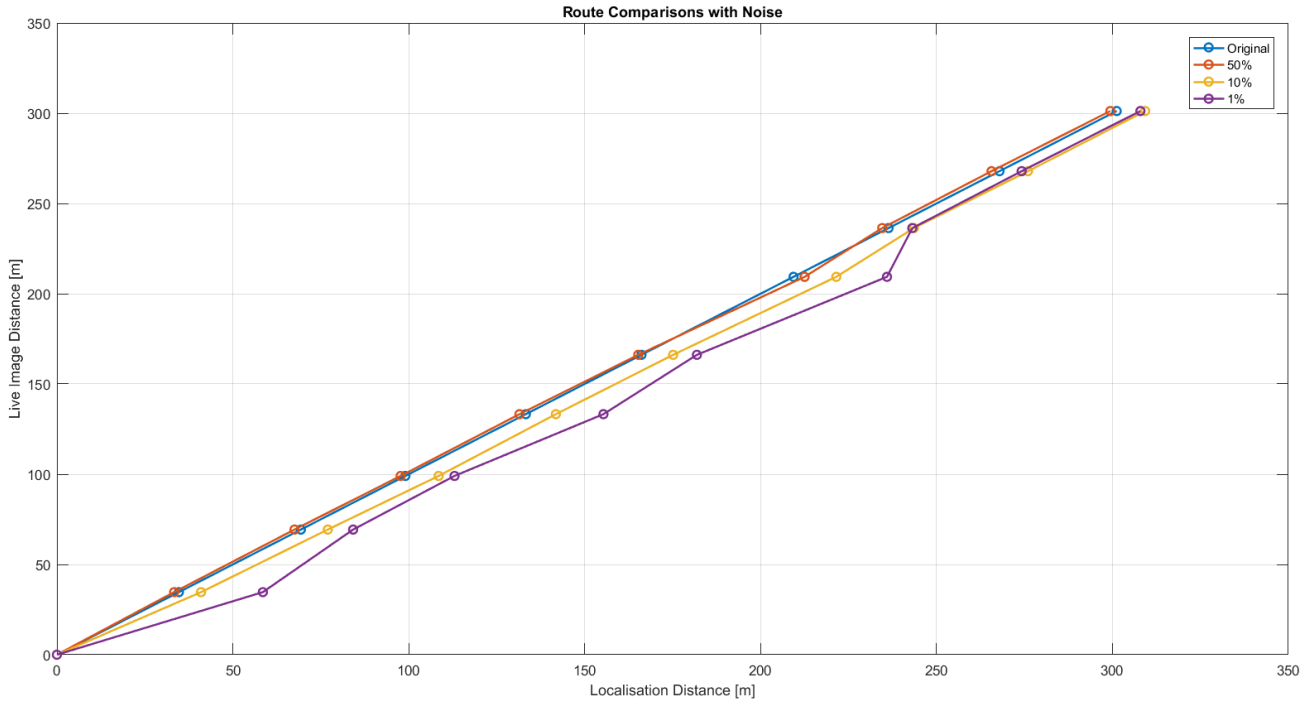


Figure 5.9: Noise added to Figure 5.7

The system fared well for the original trajectory including noise and extra illumination, managing to correctly match the correct images which was surprising given the snow present in the environment. Once the quality decreased below 10%, the errors were immense. It coped fine when noise was added although struggled when extra illumination was supplemented to the images. The extra illumination can mimic a sunny day in which snow is present as the whiteness of the snow provides a bright feature in the images and when the added illumination takes place, it can be over bearing for the pipeline to handle.

Table 5.4: Overall Error

Image Quality(%)	Error (%)
50	5.13
10	14.00
1	26.96

With the exception of the added illumination, the error margin decreased for quality higher than 10% compared to the results taken thus far. The 50% quality proved to be admirable, substantially lower than the conditions assessed prior to this proving to be robust. However, as the quality dropped, the errors yielded became much higher which

could mean that if the snowy conditions went under more image manipulation and variety of conditions, it could be found to be problematic, even for the 50% quality type.

5.4 Night

The night images were found to be poorly lit, with only the vehicles headlights and occasionally, streetlamps being able to light up the environment. This was established to be not enough for the pipeline to be able to distinguish a good amount correct similarity between the images. This caused the images to match incorrectly with a huge error margin for original images without compression or image manipulation. Below demonstrates the lack of good matches where it is seen incorrect matched being calculated.

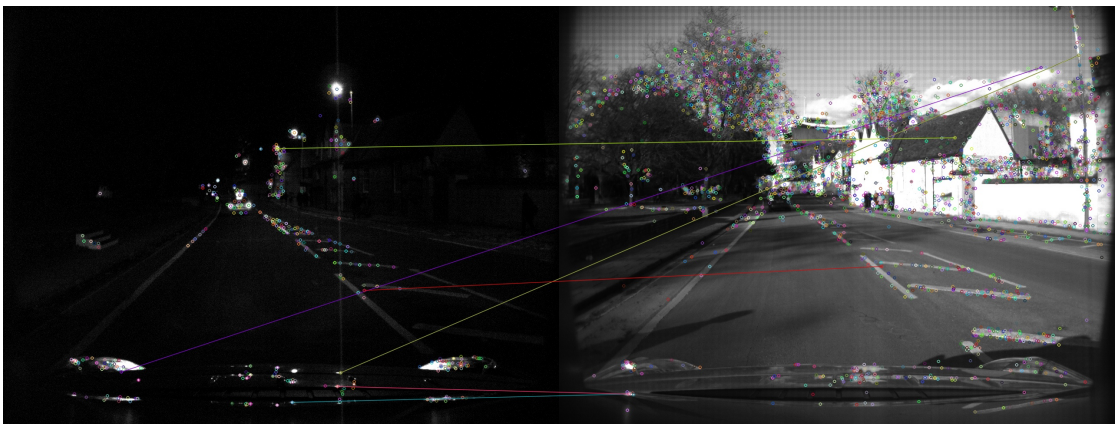


Figure 5.10: Night-time image matching

Therefore, the night-time dataset was discarded as the results would have been completely out of line giving the bounds of the experiment.

5.5 Dusk

The best alternative to replacing the night-time data was to use the dusk dataset. This provided a dark enough environment that differed from other datasets which was still adequately bright for the pipeline to yield accurate results. This dataset was completed during spring on a clear day.

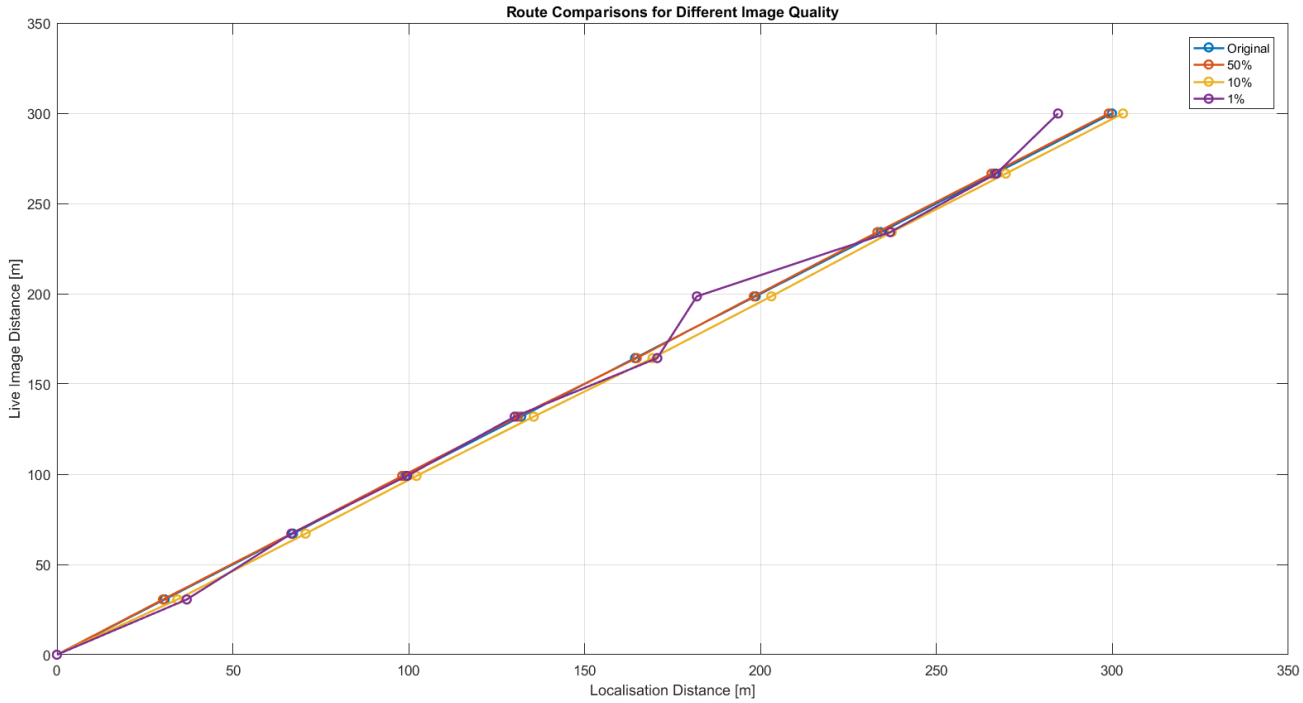


Figure 5.11: Route Comparisons during Dusk

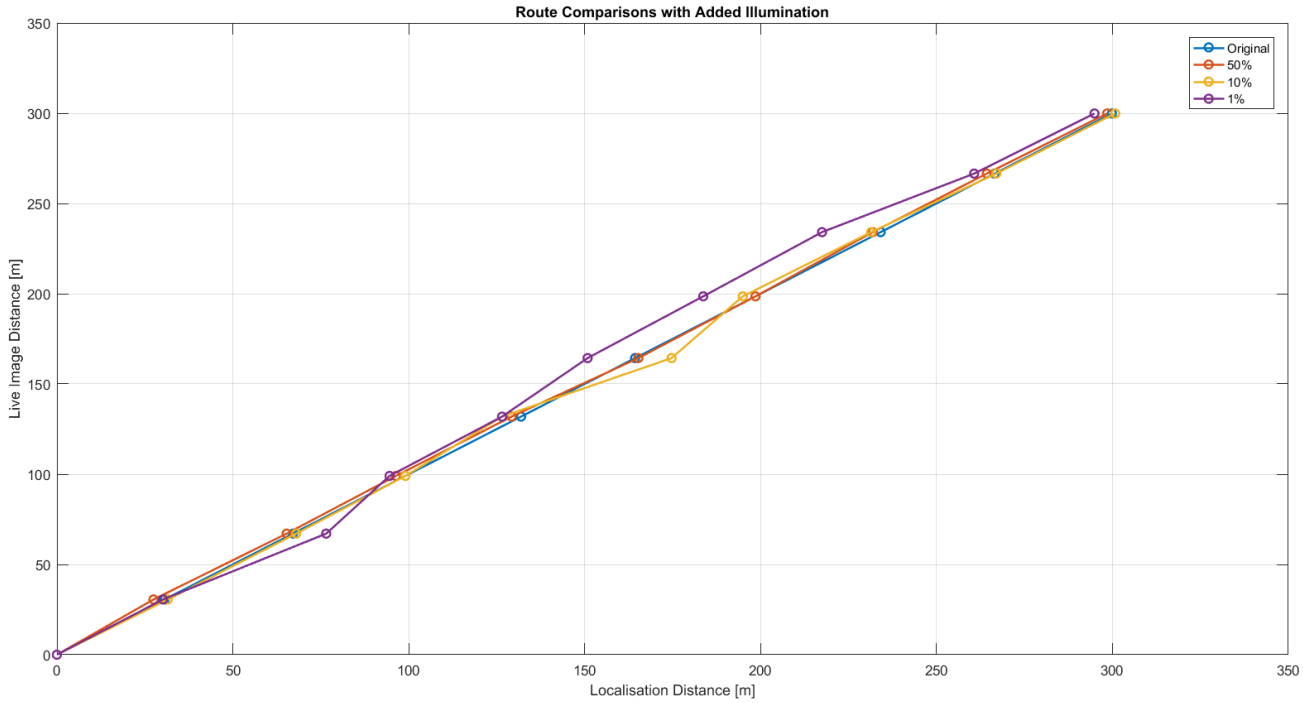


Figure 5.12: Added Illumination to Figure 5.10

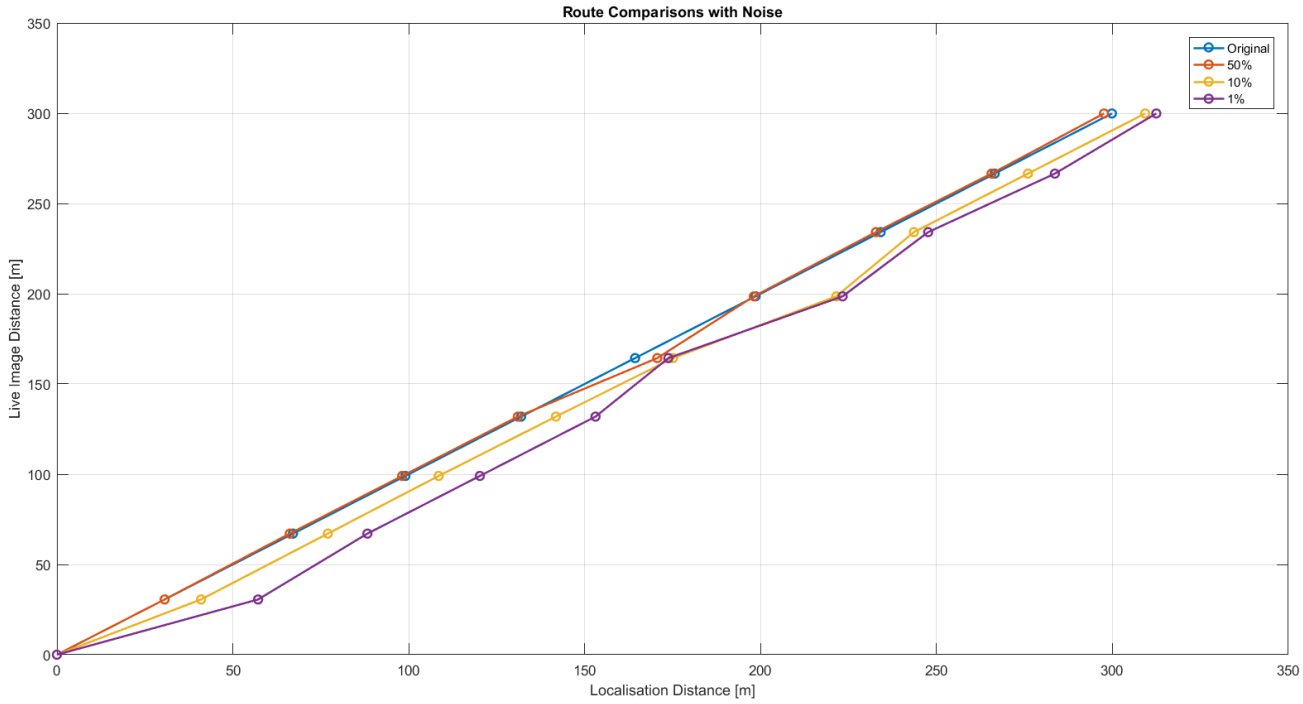


Figure 5.13: Noise added to Figure 5.10

The pipeline was able to accurately determine the correct image matching under all image manipulations. This leads to a desirable original trajectory and the 50% image quality produced remarkable results on all of the route comparisons shown above. The extra illumination still managed to be tracked well for most of the image qualities as the lack of light that the weather condition in which dusk provides. Thus, the extra illumination will not affect the pipelines performance extensively. However, with image quality below 50%, the presence of noise in images yields large errors which compromises the overall results.

Table 5.5: Overall Error

Image Quality(%)	Error (%)
50	3.45
10	17.76
1	31.31

Overall, 50% quality provides marginal error whereas the error increases significantly as the quality decreases. This could be due to the lack of lighting present in dusk, as reducing the quality, results in a similar situation found when comparing the night-time

images where the pipeline struggles to match images accurately. The biggest contributor to error was the compression of images, especially when noise was added. The error found in qualities of 10% and lower were extremely high.

5.6 Overall Performance

The performance of the pipeline in various weather and time of day conditions resulted in accurate results when not induced to compression or image manipulation. Once the images were compressed and undergone image manipulation, the errors increased accordingly with the decrease in quality. For each weather and time condition, the pipeline performed differently with the changed images. In general, the 50% quality images fared well in most conditions, and could be an option for visual navigation robots. Doing this could save a mammoth amount of storage as showcased in the method was that the 50% quality image saves 615 kilobytes of data per image. The database of a fully operational visual localisation system would consist of a myriad of images to cover the course of a route.

Chapter 6

Conclusions

The main purpose of this study was to create an image-based localisation pipeline and to then assess how the compression of images would affect the performance of the pipeline.

Due to the lack of exposure to the equipment required to attain the data needed to test the pipeline, a data source was identified in the Oxford RobotCar Dataset which provided images of a route taken in the streets of Oxford, United Kingdom which was taken over a course of a year. Utilizing many of the datasets made available, various weather conditions of the route was used.

The first step was to create a database of images of the subsection of a route for the pipeline to store along with a GPS coordinates for each image. A second subsection of the route acted as "live" images which was used as an input into the pipeline.

Following so, the "live" images were then compared to the database images in order to establish the images positions. This was achieved by utilizing the method of SIFT (Scale Invariant Feature Transform) which is a type of feature matching to find similarities between images. The robustness of the pipeline was assessed by testing different weather conditions, adding noise and increasing the illumination to the images. Subsequently, the images were compressed using the JPEG compression scheme to study the effects this will have on the pipeline.

The performance of the pipeline without image compression was accurate and had marginal error. Therefore the implementation of a visual localisation pipeline was achieved. Thereafter, the images were compressed and the accuracy started to diminish where constant errors were found in the image matching process meaning it was matching the "live" image with

an incorrect database image in a particular route.

The image quality tested were for 50%, 10% and 1%. It was found that the 50% image quality fared well and error was low meaning it could be a benchmark for a more efficient pipeline. However, any lower quality yielded an immense increase in error. After testing the pipeline with compression for various conditions, the purpose of the study was achieved by surveying the effects of image compression in a single aspect of visual navigation.

Significance of Research

This study can explore the opportunity to compress data in robots which require visual navigation. This will enable robots to have more sensors present in it without compromising the efficiency and speed of the performance or even improve it.

Chapter 7

Recommendations

Future work done on this topic should include running the pipeline through more tests by evaluating more datasets. This can include weather conditions, image quality, different noise and more types of image manipulation. The "live" images should be compared to the whole database appose to a radius of 30m. The more data that used for testing will give a further insight into this experiment.

For the system presented in this report, it would be beneficial if the data used was captured as part of this study compared to using a data source which was done in this study. This would allow the specifications of the images and frequency of the GPS locations to be controlled to work accordingly with the pipeline.

It would be beneficial to parallelize the code used to run the pipeline – for the image matching section. These consist of nested for loops which would be ideal to parallelize. This could be achieved by implementing openMP or MPI. The time taken to prepare the data and execute the pipeline for each weather condition took approximately 4 hours to achieve. The program could be executed on a faster language such as C++ where the efficiency can be improved.

Different techniques used for feature matching can be utilized. These are discussed in Chapter 3 in which some of the newer methods prove to be more accurate and efficient compared to SIFT which was implemented in this report.

All of the suggestions above would lead to a more robust and efficient visual localisation and add value to the research being done into the compression of data used in visual navigation robots.

Bibliography

[1] Y. Li, Q. Hu and Y. Gao, "An imaging sensor-aided vision navigation approach that uses a geo-referenced image database,", 2016

[2] C. Raghavachari and Shanmugha Sundaram G.A., "Efficient use of bandwidth by image compression for vision-based robotic navigation and control," 2014 International Conference on Communication and Signal Processing, Melmaruvathur, 2014, pp. 513-517.

[3] A. Mammeri, B. Hadjou and A. Khoumsi, "A survey of image compression algorithms for visual sensor networks," 2012

[4] J. Zhao, H. Liu, Y. Feng, S. Yuan and W. Cai, "BE-SIFT: A More Brief and Efficient SIFT Image Matching Algorithm for Computer Vision," 2015 IEEE International Conference on Computer and Information Technology; Ubiquitous Computing and Communications; Dependable, Autonomic and Secure Computing; Pervasive Intelligence and Computing, Liverpool, 2015, pp. 568-574.

[5] C. Harris and M Stephens, "A combined corner and edge detector," Alvey Vision Conference , 1988.

[6] M. Armstrong and A. Zisserman, "Robust robot tracking," Proceedings of the Second Asian Conferenceon Computer Vision , 1995, 15:50

[7] E. Rosten and T. Drummond, "Machine learning for high-speed detection," Proceedings of the European Conference on Computer Vision, 2006, 420-443.

[8] D. Marr and E. Hildreth, "Theory of edge detection," Proceedings of the Royal Society of London , 1980, 207(1167):187-217

[9] D.G Lowe, "Object recognition from local scale-invariant features," *IEEE*, pp. 1150-1157, 1999.

[10] D.G Lowe, "Distinctive image features from scale-invariant keypoints," *IEEE*, University of British Columbia, 2004.

[11] I Rey-Otero and M. Delbracio, “Anatomy of SIFT method,” Duke University, 2014

[12] Towards Data Science, ”SIFT (Scale-invariant feature transform”. [Online]. Available: https://towardsdatascience.com/sift-scale-invariant-feature-transform-c7233dc60f37.

[13] S. Paul and U. C. Pati, ”SAR Image Registration Using An Improved Anisotropic Gaussian-SIFT Algorithm,” 2017 14th IEEE India Council International Conference (INDICON), Roorkee, 2017, pp. 1-5.

[14] M. Cummins and P. Newman, “Probabilistic localization and mapping in the space of appearance,” Oxford University Mobile Robotics Group, 2008

[15] I. Enchev, M. Pfingsthorn, T. Luczynski, I. Sokolovski, A. Birk and D. Tietjen, ”Underwater place recognition in noisy stereo data using FAB-MAP with a multimodal vocabulary from 2D texture and 3D surface descriptors,” OCEANS 2015 - Genova, Genoa, 2015, pp. 1-8.

[16] M. Volpi and D. Tuia, ”Semantic labeling of aerial images by learning class-specific object proposals,” 2016 IEEE International Geoscience and Remote Sensing Symposium (IGARSS), Beijing, 2016, pp. 1556-1559.

[17] E. Stenborg, C. Toft and L. Hammarstrand, ”Long-Term Visual Localization Using Semantically Segmented Images,” 2018 IEEE International Conference on Robotics and Automation (ICRA), Brisbane, QLD, 2018, pp. 6484-6490.

[18] X. Zhou, K. Wang and J. Fu, ”A Method of SIFT Simplifying and Matching Algorithm Improvement,” 2016 International Conference on Industrial Informatics - Computing Technology, Intelligent Technology, Industrial Information Integration (ICIICII), Wuhan, 2016, pp. 73-77.

[19] J. K. Makhubela, T. Zuva and O. Y. Agunbiade, ”Framework for Visual Simultaneous Localization and Mapping in a Noisy Static Environment,” 2018 International Conference on Intelligent and Innovative Computing Applications (ICONIC), Plaine Magnien, 2018, pp. 1-6.

[20] H. D. Whyte and T. Bailey, “Simultaneous localization and mapping (SLAM): Part I the essential algorithms,” IEEE Robot. Automat. Mag., vol. 13, no. 3, pp. 108–117, Jun. 2006.

[21] R. Chatila and J.P. Laumond. Position referencing and consis- tent world modeling for mobile robots. In Proc. IEEE Int. Conf. Robotics and Automation, pages 138–143, 1985.

[22] K.S. Chong and L. Kleeman. Feature-based mapping in real, large scale environments using an ultrasonic array. International Journal Robotics Research, 18(1):3–19, 1999.

[23] N. Ayache and O. Faugeras. Building, registrating, and fusing noisy visual maps. Int. J. Robotics Research, 7(6):45–65, 1988.

[24] W. Maddern, G. Pascoe, C. Linegar and P. Newman, "1 Year, 1000km: The Oxford RobotCar Dataset", The International Journal of Robotics Research (IJRR), 2016

[25] M. A. Nielsen, "Neural networks and deep learning," 2015

[26] Z. Cao, T. Simon, S.-E. Wei, and Y. Sheikh, "Realtime multi-person 2d pose estimation using part affinity fields," in 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), p. 1302—1310, IEEE, Jul 2017

Appendix A

GitHub Link

The following link is a GitHub repositry which contains the code used in this study: https://github.com/PLLSAY005/PLLSAY005_EEE4022S

Appendix B

Addenda

B.1 sun

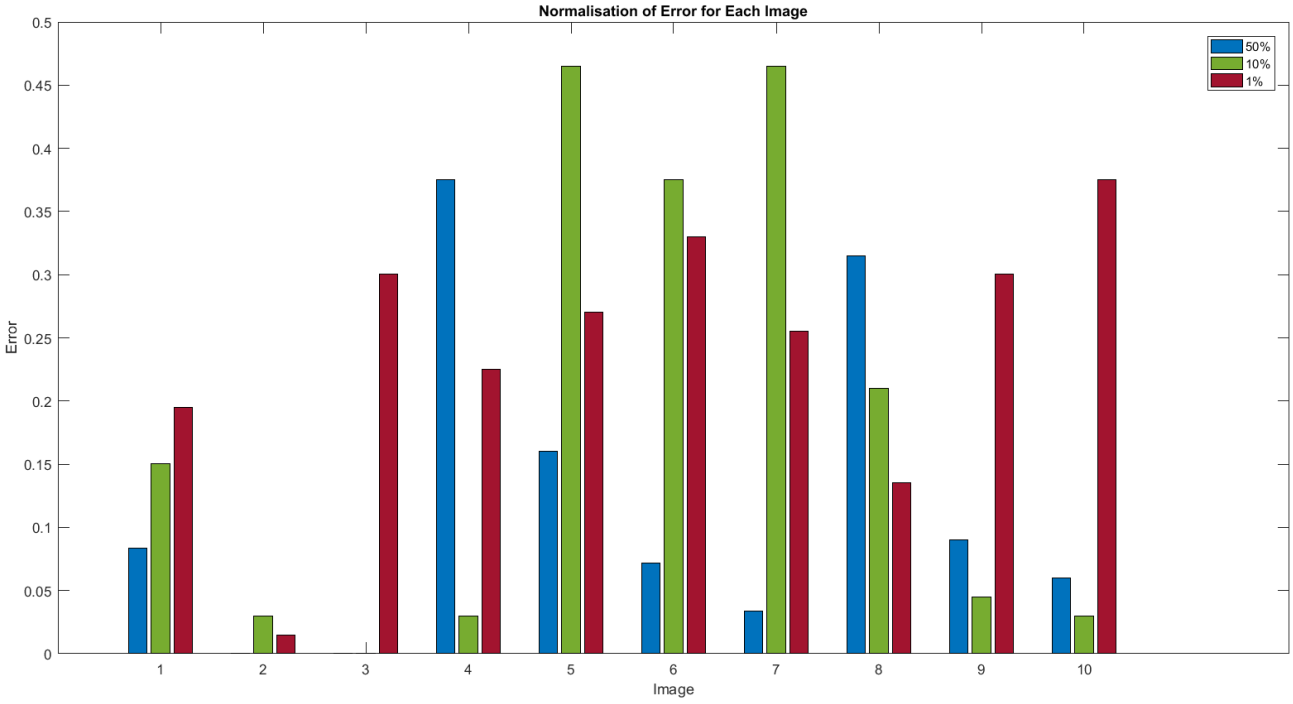


Figure B.1: Normalisation of Errors

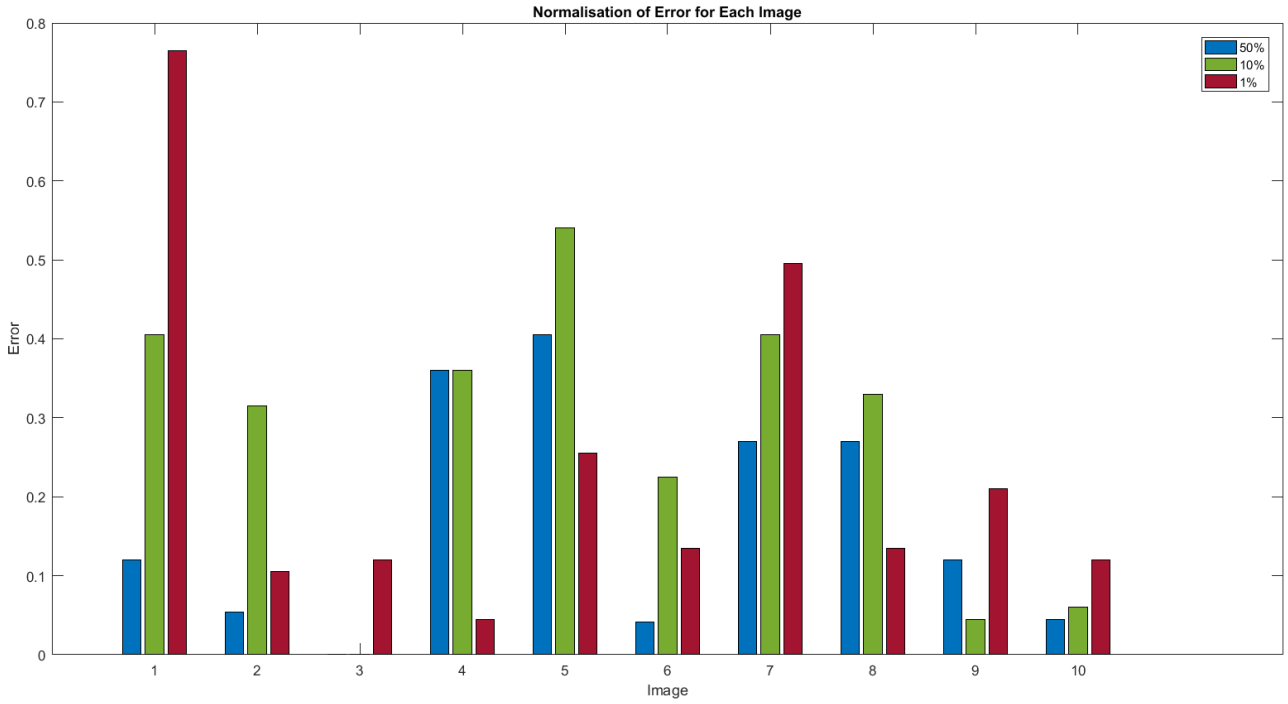


Figure B.2: Added Illumination

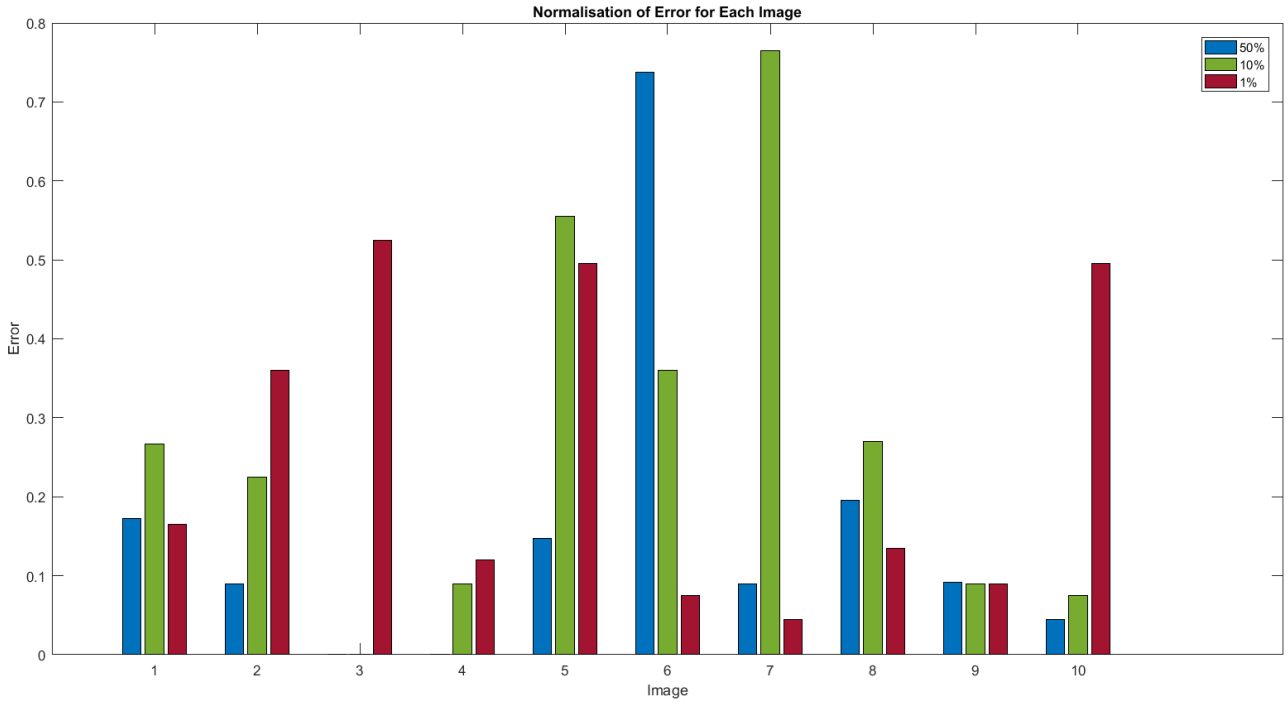


Figure B.3: Noise Added

B.2 Overcast

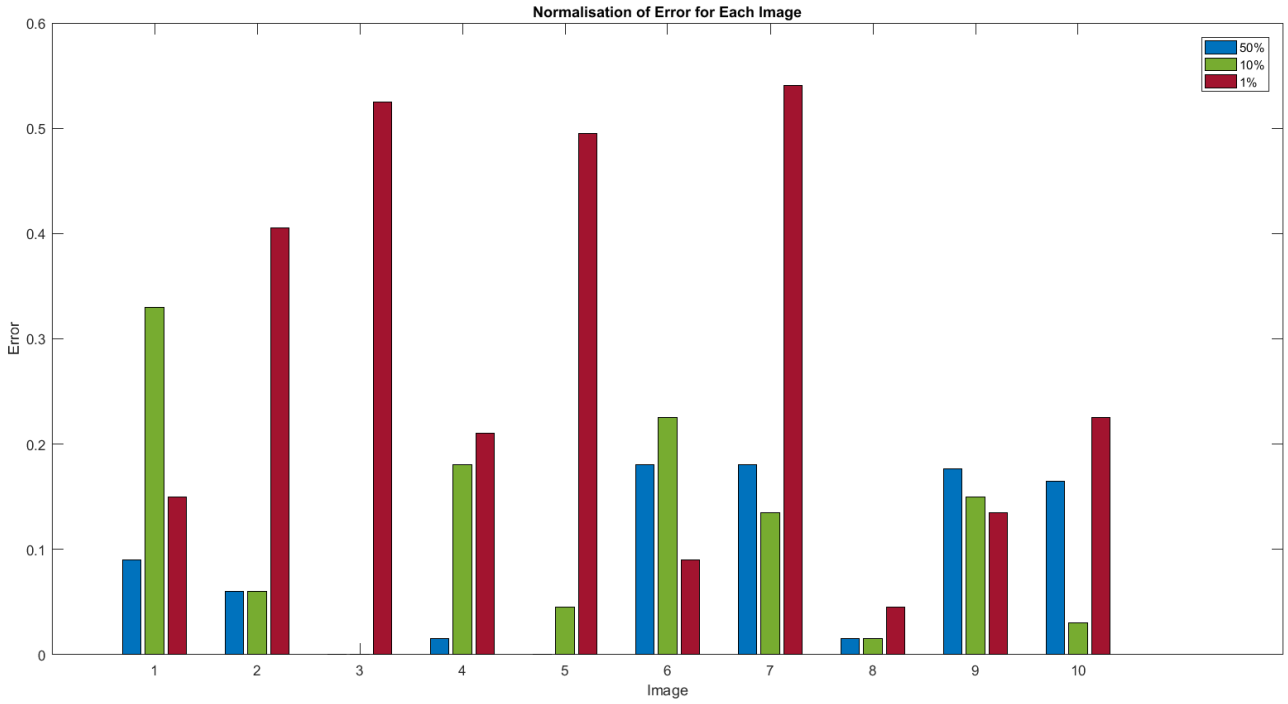


Figure B.4: Normalisation of Errors

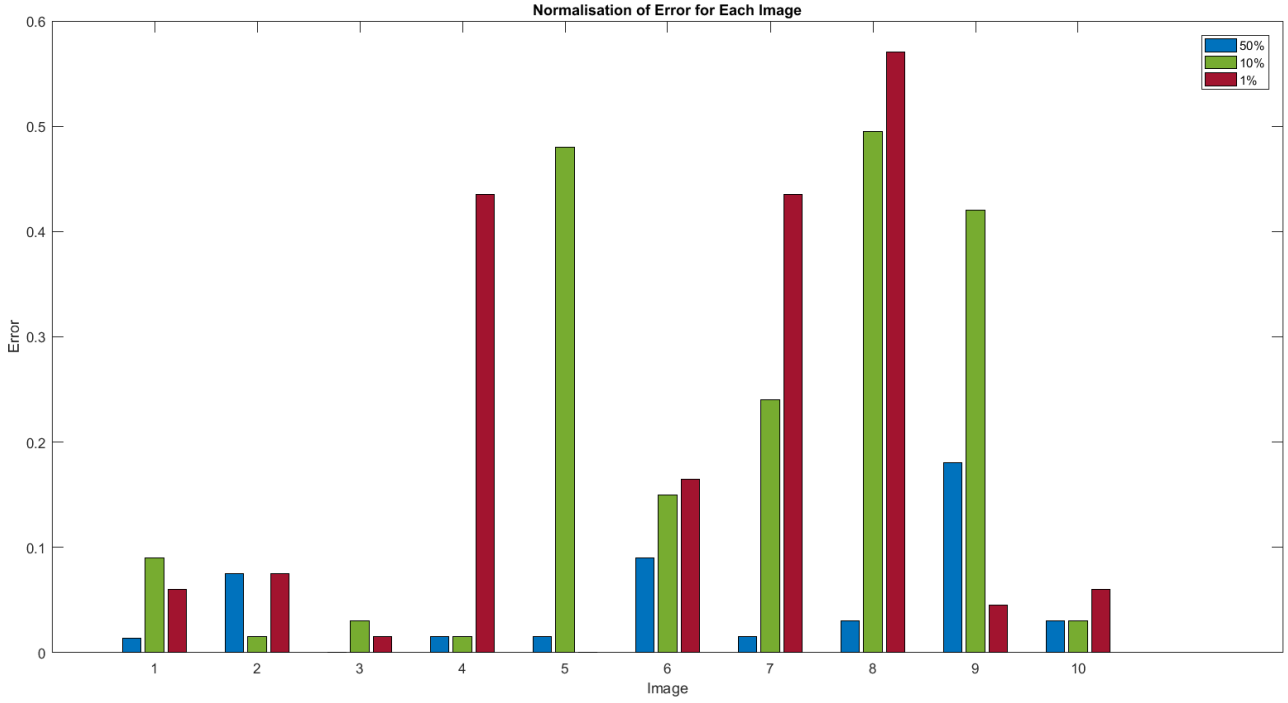


Figure B.5: Added Illumination

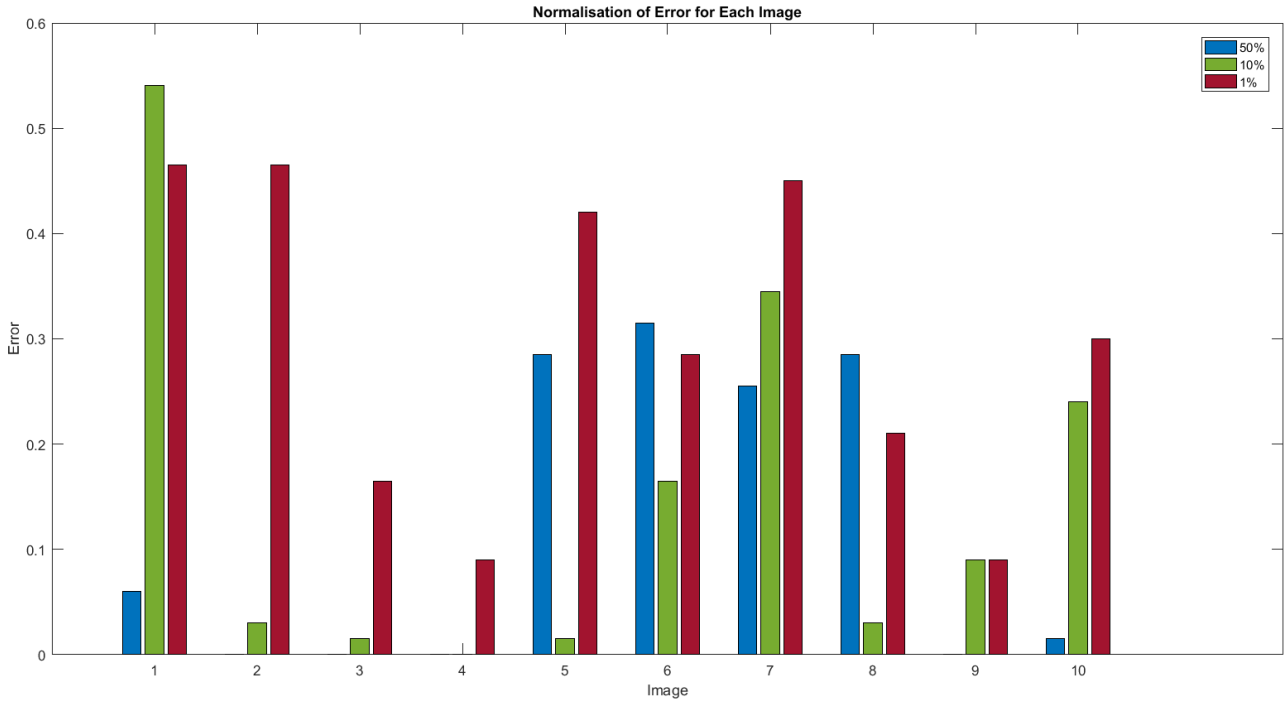


Figure B.6: Noise Added

B.3 Snowy

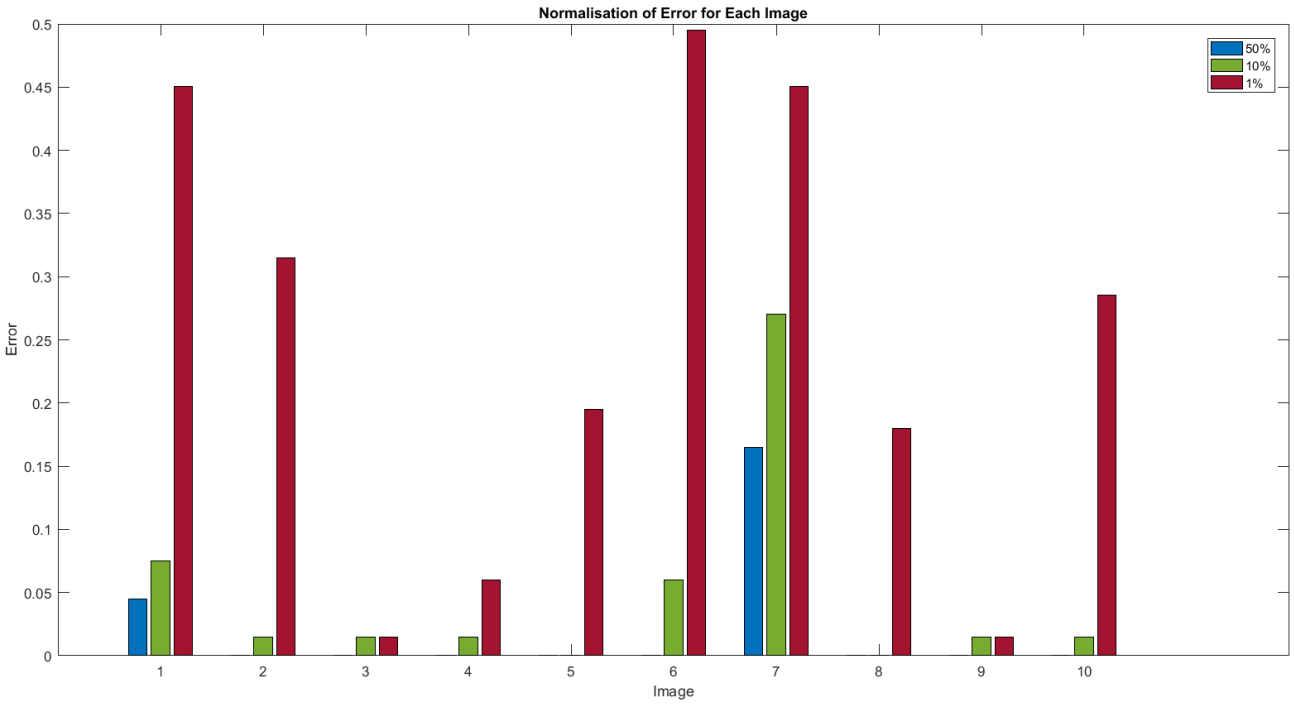


Figure B.7: Normalisation of Errors

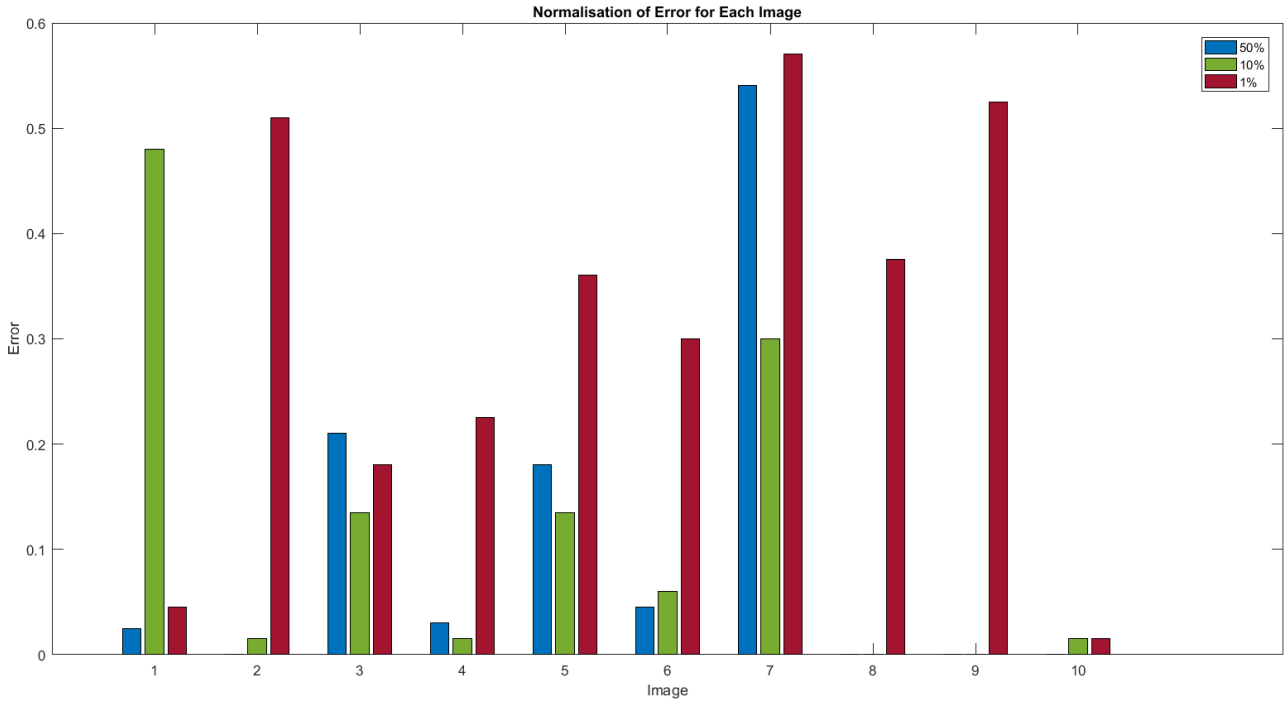


Figure B.8: Added Illumination

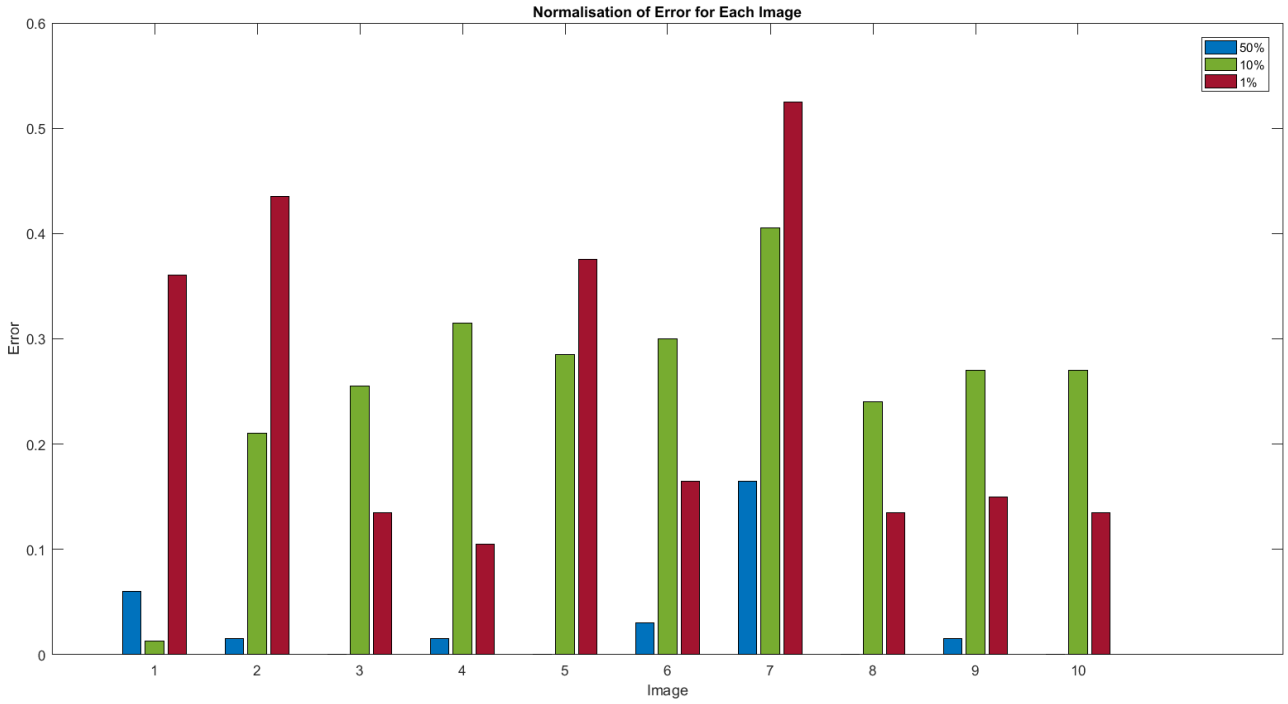


Figure B.9: Noise Added

B.4 Dusk

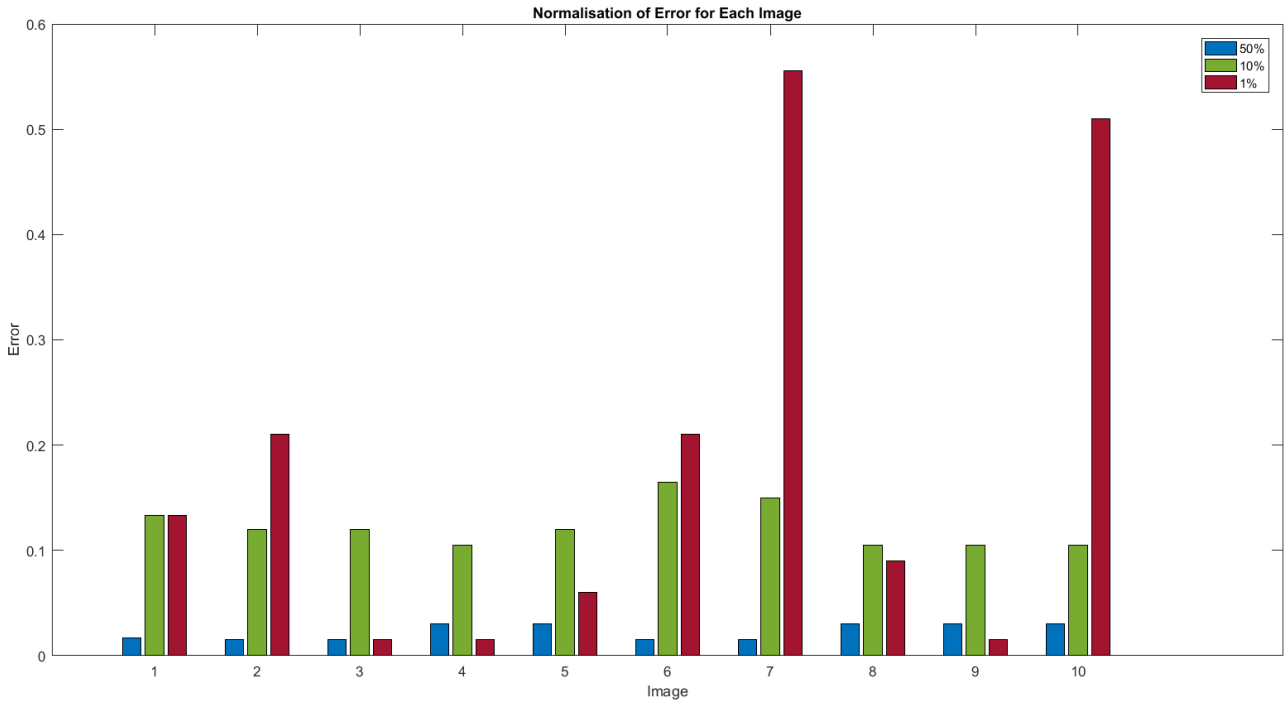


Figure B.10: Normalisation of Errors

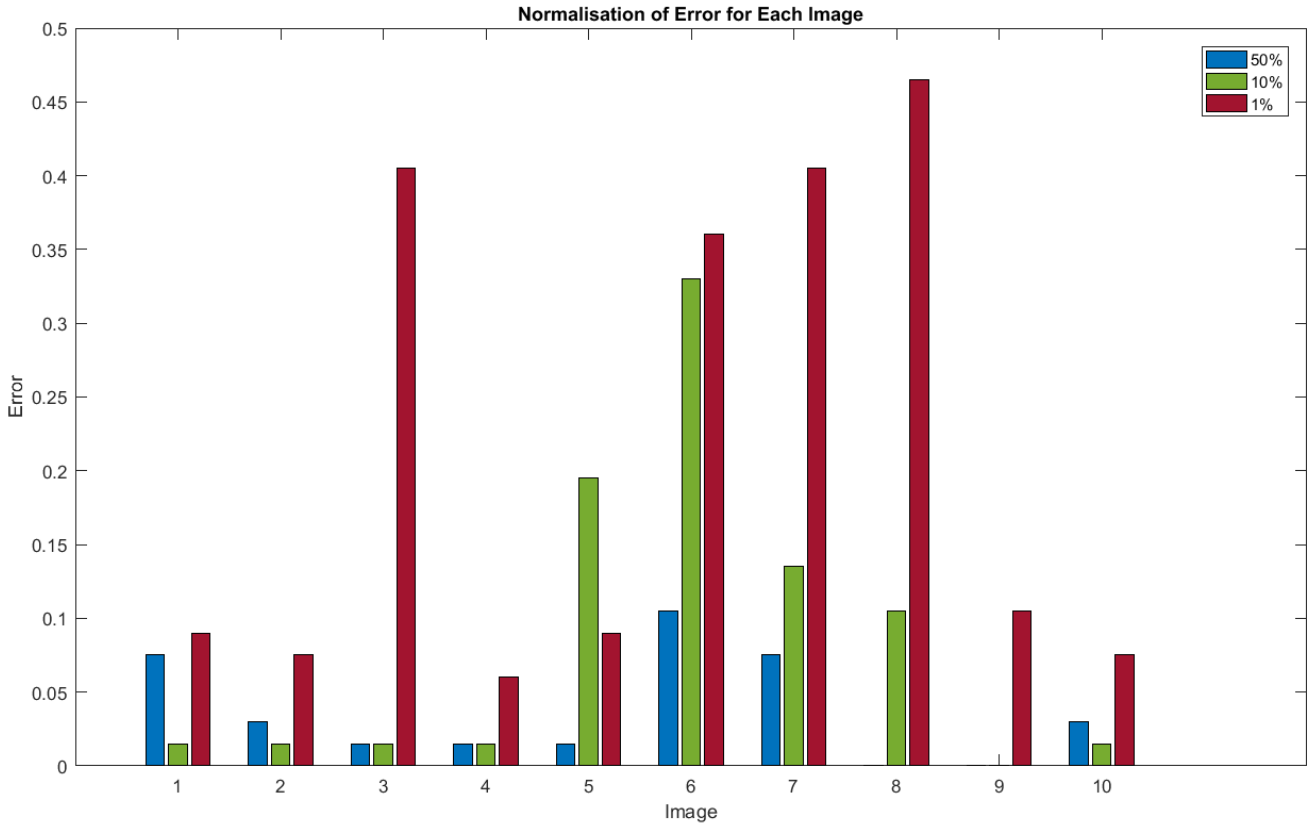


Figure B.11: Added Illumination

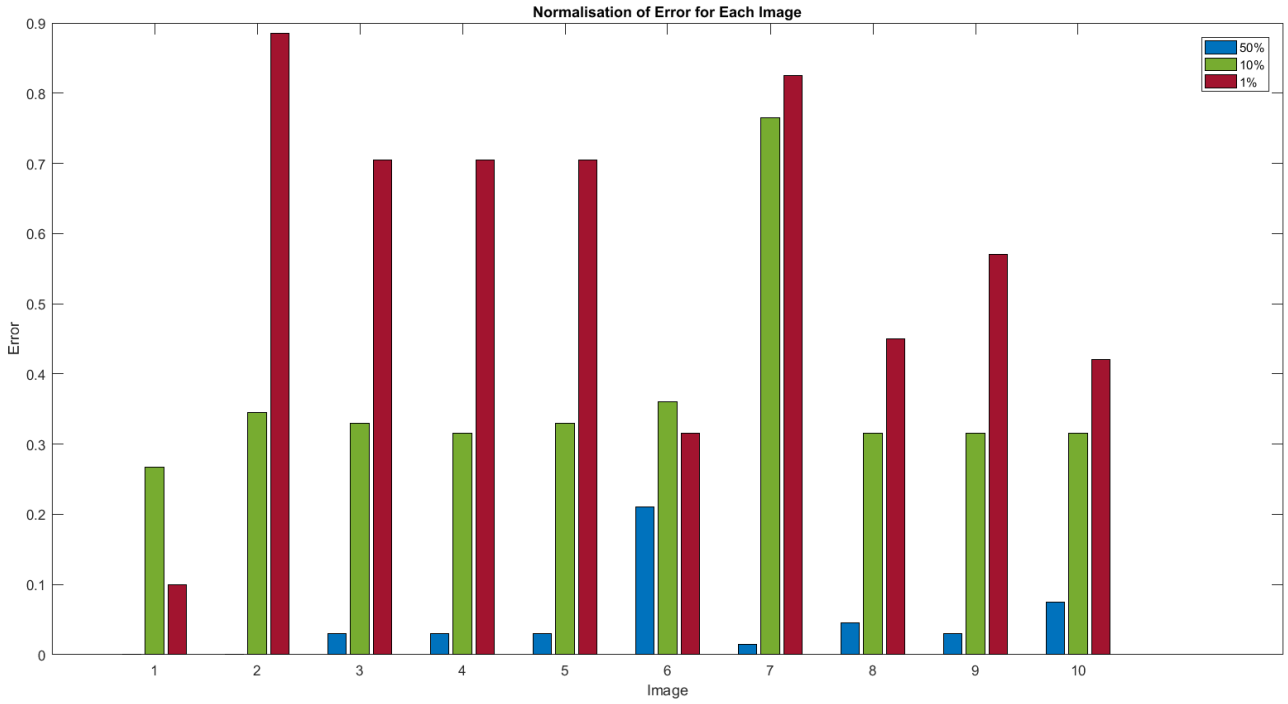


Figure B.12: Noise Added

B.5 Ethics Forms

Application for Approval of Ethics in Research (EiR) Projects
Faculty of Engineering and the Built Environment, University of Cape Town

ETHICS APPLICATION FORM

Please Note:
Any person planning to undertake research in the Faculty of Engineering and the Built Environment (EBE) at the University of Cape Town is required to complete this form **before** collecting or analysing data. The objective of submitting this application *prior* to embarking on research is to ensure that the highest ethical standards in research, conducted under the auspices of the EBE Faculty, are met. Please ensure that you have read, and understood the **EBE Ethics in Research Handbook** (available from the UCT EBE, Research Ethics website) prior to completing this application form: http://www.ebe.uct.ac.za/ebe/research/ethics1

APPLICANT'S DETAILS		
Name of principal researcher, student or external applicant		Saylin Pillay
Department		Electrical Engineering
Preferred email address of applicant:		saylin11@gmail.com
If Student	Your Degree: e.g., MSc, PhD, etc.	Bsc Mechatronics
	Credit Value of Research: e.g., 60/120/180/360 etc.	40
	Name of Supervisor (if supervised):	Paul Amayo
If this is a researchcontract, indicate the source of funding/sponsorship		
Project Title		Visual Navigation Under Image Compression

I hereby undertake to carry out my research in such a way that:

- there is no apparent legal objection to the nature or the method of research; and
- the research will not compromise staff or students or the other responsibilities of the University;
- the stated objective will be achieved, and the findings will have a high degree of validity;
- limitations and alternative interpretations will be considered;
- the findings could be subject to peer review and publicly available; and
- I will comply with the conventions of copyright and avoid any practice that would constitute plagiarism.

APPLICATION BY	Full name	Signature	Date
Principal Researcher/ Student/External applicant	Saylin Pillay		25/07/2019
SUPPORTED BY	Full name	Signature	Date
Supervisor (where applicable)	Paul Amayo		28/07/2019

APPROVED BY	Full name	Signature	Date
HOD (or delegated nominee) Final authority for all applicants who have answered NO to all questions in Section 1; and for all Undergraduate research (Including Honours).			
Chair: Faculty EIR Committee For applicants other than undergraduate students who have answered YES to any of the questions in Section 1.			

Page **1** of **1**