Imperial College London

IMPERIAL COLLEGE LONDON

DEPARTMENT OF COMPUTING

Enhancing Visual SLAM for Autonomous Forest Navigation through Robust Feature Correspondence

Author:
Saifullah Ijaz

Supervisors:
Dr Bahadir Kocer
Dr Ronald Clark

Submitted in partial fulfillment of the requirements for the MEng degree in Electronic and Information Engineering of Imperial College London

January 2024

Contents

Chapter 1

Introduction

1.1 Motivation

Forest environments form a large part of the natural terrain throughout many areas of the world. Ecological studies in forests are essential to monitor the changing biodiversity within these ecosystems.[1] Typical forestry applications include forest monitoring, wildlife monitoring, early detection of wildfires and search and rescue operations.[2] Due to the large geographical areas covered by forests, it is often difficult for humans to perform monitoring tasks manually. There are also potential safety concerns for search and rescue operations that take place after natural disasters, where it is often unsafe for emergency response teams to carry out searches in areas prone to earthquakes and wildfires.[3] Advancements in imaging hardware and computer vision techniques have led to autonomous robotic systems, such as aerial robots and ground vehicles, being used to quickly survey large areas of forests without the need for direct human involvement.[4]

However, autonomous navigation of forest environments remains a challenging problem for mobile robots.[5] Autonomous navigation systems are inherently limited by the capabilities of the navigation sensors, since the information provided by these sensors is used to inform navigation tasks such as localisation, mapping, path planning and trajectory tracking.[6] GPS navigation systems [7] suffer from increased error and signal interference due to tree trunks and dense canopy cover within forest environments.[8]

Visual Simultaneous Localisation and Mapping (V-SLAM) [9] has been explored as an alternative approach for autonomous navigation of robotic systems within GPS denied environments. V-SLAM uses a camera as the primary sensor to provide visual data for a robot to perform SLAM [10] tasks. The frames captured by the camera can be used to perform accurate pose estimations [11] and to create meaningful dense maps of the environment.[12] However, forest environments provide challenging conditions for V-SLAM to work effectively, since the visual information captured by the camera is affected by several factors.[13] These include complex textures, variable lighting conditions and dynamic objects.[14]

1.2 Objectives

Previous work [15] explored the use of tree trunks as robust features for keypoint matching between frames, to improve the trajectory estimations of V-SLAM systems within forest environments. Tree trunks can be considered reliable landmarks that are resilient to the challenges faced by V-SLAM systems in forest environments. A real-world forestry dataset (CanaTree100) [16] was used to train a Tree Parameter Extraction Model (TPEM) to detect keypoints on tree trunks. Different feature correspondence [17] techniques were compared, and it was shown that the inclusion of the tree parameters as features improved the accuracy of the pose estimations compared to the standard ORB [18] feature correspondence method. However, the SuperGlue [19] feature correspondence method which uses SuperPoints [20] as features resulted in the smallest trajectory errors, proving it to be the most robust method out of the three which were tested.

The majority of SLAM datasets available are focused on urban environments. There is a distinct lack of real-world forestry datasets that can be used to evaluate SLAM approaches for autonomous forest navigation. Existing forestry datasets typically include RGB ground truth data but lack any data related to depth information, which means it is difficult to use these datasets to assess 3D SLAM systems. The CanaTree100 dataset is useful for machine learning tasks related to tree detection, but lacks camera pose data and ground truth 3D point clouds,[21] so is less effective for SLAM related applications. Synthetic datasets of forest environments can be utilised; however, the simulation environments usually lack photorealism and do not fully encapsulate real-world physics.

Therefore, we propose a new SLAM dataset for real-world forest environments which will provide comprehensive ground truth data, thus providing a basis to evaluate V-SLAM implementations designed for autonomous forest navigation. The dataset will comprise of frames obtained from recordings in forest environments, using an RGB-D camera,[22] alongside the corresponding ground truth 3D point clouds for each frame. Additionally, it will include location data that encompasses the trajectory of the camera's movement throughout the recorded sequence. This will be used to determine localisation accuracy based on the error between the estimated camera poses and the actual path followed.

A V-SLAM system will be implemented with the aim of using SuperGlue as a robust feature correspondence technique to generate dense 3D maps of forest environments. The resulting maps will be stored in the new dataset alongside the estimated camera poses for each testing sequence. This will facilitate fair comparisons with other V-SLAM implementations in the future, specifically in the context of forest environments.

Chapter 2

Background

In this chapter, we:

- Provide an overview of both SLAM and Visual SLAM, highlighting the different tasks that constitute each of their respective pipelines.
- Outline the functionalities of RGB-D cameras and explain the challenges with using vision-based sensors in forest environments.
- Describe traditional feature description techniques, discussing the limitations of each method.
- Review previous work in which different feature correspondence techniques were compared for a visual odometry system in the context of forest environments, with particular focus on the SuperGlue feature correspondence method.

2.1 SLAM

2.1.1 Overview of SLAM

Simultaneous Localisation and Mapping (SLAM) [10] is a problem in which a mobile robot traverses through an unknown environment to progressively construct a map of the environment whilst also self-localising with respect to the map at the same time.[23] SLAM has been applied to several different types of mobile robots, including ground, underwater and aerial vehicles.[24, 25, 26] The SLAM problem involves the robot estimating the position of other landmarks [27] within the environment as well as its own position relative to those landmarks.[28]

Mapping an unknown environment requires a detailed understanding of the position of landmarks. This then informs the localisation task, in which a robot performs state estimation [29] to predict its position and orientation to ultimately find its location within the map.[30]

Figure 2.1: SLAM Pipeline [31]

2.1.2 General SLAM Pipeline

Figure 2.1 shows the sequence of tasks that take place within a SLAM system. Practical implementations of SLAM on mobile robots usually consist of a combination of exteroceptive [32] (outward-looking) sensors which capture distance measurements to other objects within the environment, and proprioceptive [32] (inward-looking) sensors which capture data such as velocity and orientation of the robot, which are used in pose estimations. Typical exteroceptive sensors used on robotic platforms include LIDAR,[33] RADAR,[34] acoustic [35] or vision-based sensors,[12] which are used to characterise the location of objects within the environment. Examples of proprioceptive sensors include Inertial Measurement Units (IMUs),[36] accelerometers and gyroscopes.[37]

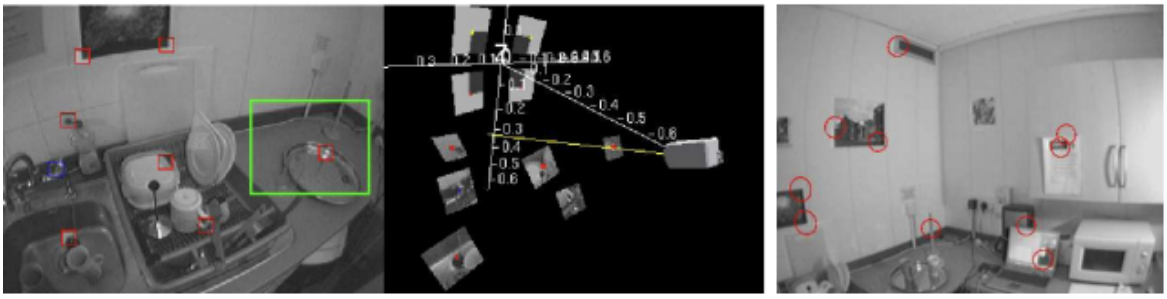

Figure 2.2: Examples of Features used within SLAM [38]

Exteroceptive sensors are used to compute distances to features [39] that are present in the environment. Features are discernible aspects of the environment that can be recognised from multiple viewpoints. Examples include wall segments and corners of objects for sensors such as LIDAR, or lines and points for vision-based sensors as shown in Figure 2.2.[38] As the robot moves through the environment, new features will be observed, features may go out of view of the sensor, and other features that were previously seen might now be observed from a different point of view.

The terms data association, feature matching and feature correspondence are used interchangeably to represent the task of establishing if two features observed at different viewpoints represent the same object in the real-world.[40] Correlating the newly observed information with the pre-established map information is crucial to acquiring an understanding of the spatial relationship between objects in the physical environment.[41] Hence, the quality of the maps generated by SLAM algorithms is directly influenced by how well the feature correspondence task is conducted.

The localisation task involves odometry,[42] in which a robot makes predictions on where it is currently located on the map based on either proprioceptive or exteroceptive sensor measurements. To accurately determine its location, a robot needs to estimate its pose, which consists of its position and orientation. Similarly to the mapping aspect of SLAM, the localisation task involves taking observations of landmarks at various positions. As the robot travels through the unknown environment, measurements of features taken from different viewpoints in the environment will allow the robot to start to self-localise through place recognition as it remembers previously seen landmarks.[43]

For small-scale SLAM systems, probabilistic methods [44] have been shown to be effective at reducing the uncertainty of the robot's predictions about its location on the map. However, odometry drift can occur due to the increased uncertainty of the robot's location at distances that can be considered far from the origin of the map. To counteract this, loop closure detection [45] is commonly employed in large-scale SLAM systems, in which re-observations of previously seen landmarks are used to correct the drift error from the sensor measurements. When a robot revisits a previously seen landmark, the uncertainty of its pose decreases due to prior knowledge about the locations of those landmarks on the map. The SLAM map is dynamically updated as additional sensor measurements are taken, with loops being closed within the map once the robot revisits a previously observed location.[46]

2.2 Visual SLAM

2.2.1 Overview of Visual SLAM

Visual SLAM (V-SLAM) involves the use of vision-based sensors such as monocular, stereo or RGB-D cameras.[47] Traditional SLAM approaches using sensors such as LIDAR or SONAR, either produce 2D maps or sparse 3D maps.[48] Visual data is especially useful for SLAM systems because it provides detailed information that is semantically useful for humans. This allows for the creation of dense 3D maps of physical environments that capture the visual experience of the human eye.[49] Using visual data enables the implementation of visual odometry,[50] where the camera frames are used to estimate the camera's pose. This, in turn, corresponds to the robot's pose, since the camera is attached to the robotic platform, thus allowing a robot to estimate its egomotion and localise itself in an unfamiliar environment using the visual input from the camera.[51]

V-SLAM can employ either direct or indirect (feature-based) methods for both sparse and dense mapping.[52] Direct methods utilise pixel intensities directly, for either the complete set of pixels in the frame or a sub-set of them. This is in contrast to feature-based methods, which only consider pixels that correspond to features. Direct methods can result in higher density maps due to the increased amount of pixel information available. However, features are easily recognised and tracked between frames, which can improve both localisation and mapping accuracy.[53]

2.2.2 Visual SLAM Pipeline

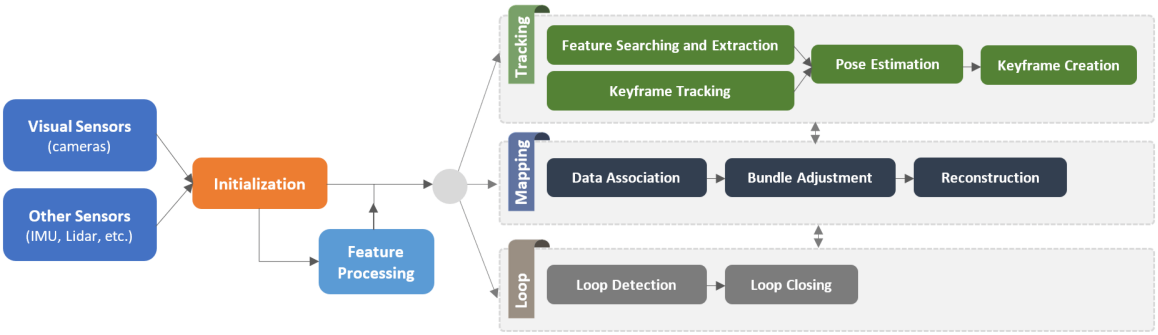

Figure 2.3: Visual SLAM Pipeline [12]

Figure 2.3 shows the sequence of tasks that take place within a V-SLAM system, although the exact tasks will vary depending on implementation choices, including whether direct or feature-based methods are used. The process will also be affected by the choice of proprioceptive sensors, since IMUs can be used alongside the vision-based sensor to implement visual inertial odometry,[54] in which the robot's ego-motion is estimated using both the IMU and visual data. Once the initial data has been acquired from the sensors, three separate threads are used for tracking (localisation), mapping and loop closure respectively, which make up the V-SLAM system.

Tracking involves feature detection and the use of keyframes to estimate the camera's pose. Traditionally, V-SLAM systems have employed filter-based methods like Kalman Filters (KF) [55] or Extended Kalman Filters (EKF).[56] These methods establish a direct connection between localisation and mapping, assuming a direct correlation between camera pose and landmarks. However, this requires the estimations of the camera's pose and locations of landmarks on the map to be updated for every camera frame which can be computationally intensive.[57]

Modern V-SLAM systems use keyframe based approaches instead, which separate the localisation and mapping tasks. The camera pose estimates still take place over every frame for the localisation task, although this is restricted to a specific region of the map, as the robot can only occupy one location at any given time. The mapping task is carried out using only keyframes instead of every camera frame.[57] Keyframes are a sub-set of the frames captured by the camera which aim to maximise the amount of useful visual information and minimise useless information, to reduce the amount of data required for mapping an unknown environment. Keyframes are usually frames in which there is a substantial change in visual data between one frame and the next, whereas frames that contain repeated visual information are usually ignored.[58] Therefore, substantially less visual data is required to perform the mapping task in keyframe based V-SLAM approaches. This results in much more computationally efficient implementations that allow keyframe based V-SLAM approaches to outperform their filter-based counterparts.

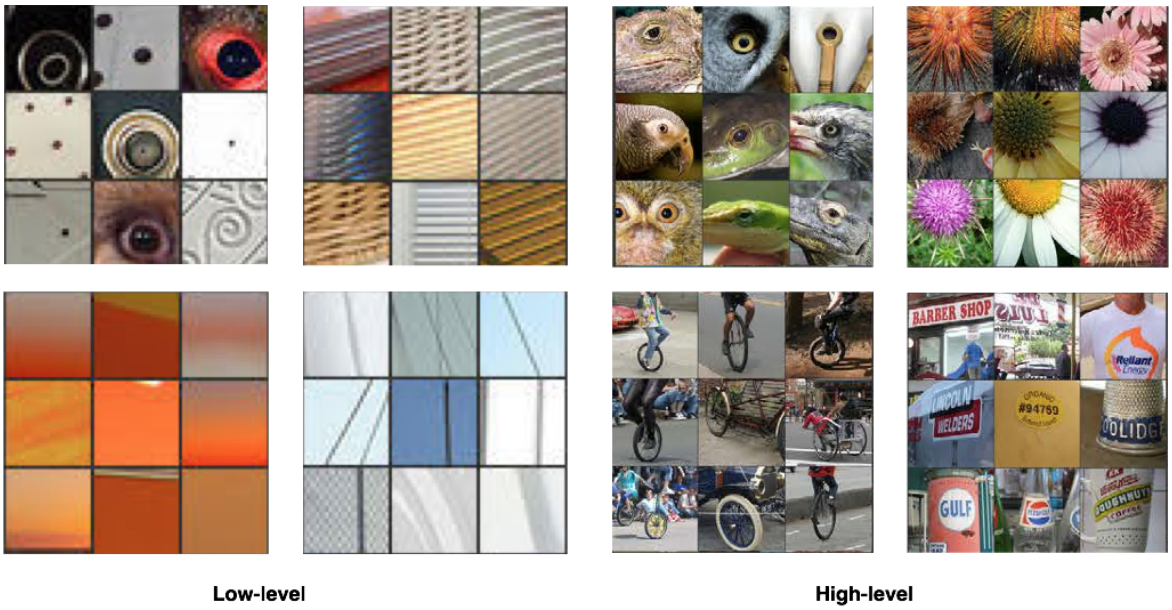

Figure 2.4: Low-level vs High-Level Features [59]

Mapping involves feature correspondence, bundle adjustment and reconstruction. Feature correspondence in V-SLAM requires matching low-level or high-level features between camera frames.[60] Elements at a lower level such as points and lines concentrate on specific details including the fundamental geometric elements of objects within the scene and aspects such as textures. High-level features are semantically labelled objects that depict scenes in a manner that represents how humans perceive the environment, as shown in Figure 2.4.[61]

The reconstruction task is use-case dependent and can be quite simple for basic 2D mapping.[62] However, photorealistic 3D mapping requires reliable depth information to ensure that the landmarks are positioned correctly within 3D space. The depth data can be collected directly to generate 3D point clouds such as in RGB-D or LIDAR based SLAM approaches. If depth information is not directly available from exteroceptive sensors, more complex computer vision techniques [63] will need to be used to infer the depth information required to accurately reconstruct the 3D scene. Additionally, computer graphics techniques [64] will need to be employed to ensure that textures are rendered correctly.

Bundle adjustment is an optimisation technique which is used to refine the visual reconstruction of the SLAM map. The estimates of the 3D feature coordinates and camera poses are optimised to minimise reprojection errors based on a cost function. However, the exact cost function used will be implementation dependent.[65] Loop closure is used to build accurate SLAM maps and to provide a mechanism that allows the robot to re-localise itself if it loses track of its current position.[66] In V-SLAM systems, similarity metrics are used to compare the features in the current frame with those that have been observed in the past. This requires accurate image matching and is usually performed by a convolutional neural network (CNN).[67]

2.3 Challenges with Visual SLAM in Forests

2.3.1 RGB-D Cameras

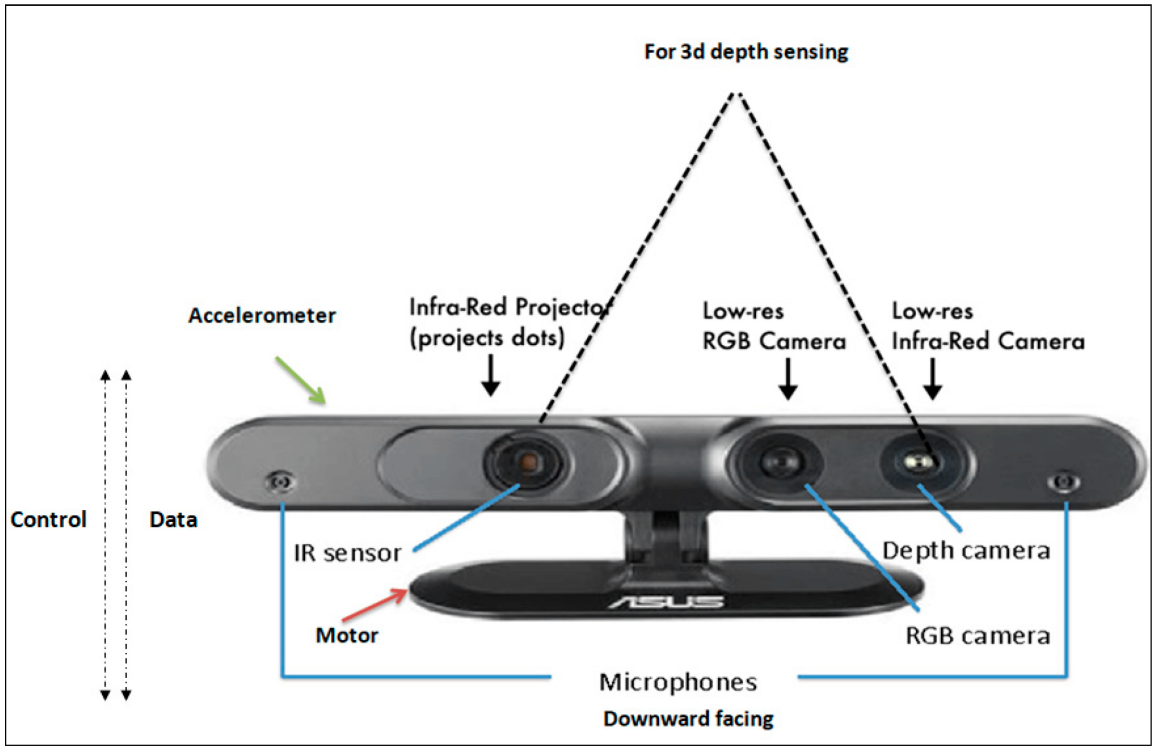

Figure 2.5: Typical RGB-D camera setup [22]

RGB-D cameras combine the functionalities of traditional RGB cameras with depth sensors, as shown in Figure 2.5, to capture the data required for 3D scene reconstruction. Each pixel is assigned a red, green and blue (RGB) colour value as well as a depth value which corresponds to a distance measured by the depth sensor to a point in the physical environment. Two different methods are typically employed to obtain depth measurements, triangulation or Time-of-Flight (TOF).[68]

Triangulation can involve either passive or active approaches. Passive approaches calculate the disparity between two images acquired from different viewpoints. Active approaches, such as structured light, involve casting a pattern of light onto the scene. Different objects will have varying depths in the physical world, which will deform the pattern of light, thus allowing depth information to be inferred from the optical distortion.[69] Structured light can take the form of visible light or other wavelengths of light from the electromagnetic spectrum such as infrared (IR). TOF methods involve measuring the time taken for light to be emitted, reflect off an object, and travel back to the detector. Different RGB-D cameras will use various depth sensors depending on factors such as cost, intended use-cases and size constraints.[70]

2.3.2 Camera Model

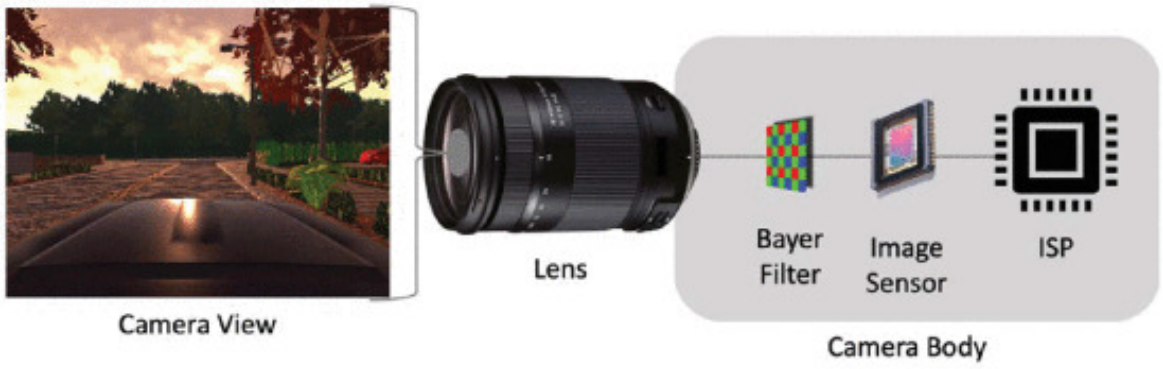

Figure 2.6: RGB camera model [71]

RGB cameras consist of a lens, Bayer filter,[72] image sensor and Image Signal Processor (ISP) as shown in Figure 2.6. Light passes through the lens onto the Bayer filter where pixels are filtered to represent either red, green or blue, as this is required for colour images.[73] The main types of image sensors are Charge Coupled Device (CCD) and Complementary Metal Oxide Semi-conductor (CMOS).[74] The image sensor contains receivers that convert photons of light into electrical signals. These are then discretised into numerical values relating to the pixel intensities. Finally, the ISP performs tasks such as denoising, lens distortion correction and encoding to achieve the final image.[71]

2.3.3 Dynamic Lighting Conditions

Lighting conditions are especially variable in forests due to dense vegetation.[75] Tree canopies often block much of the direct sunlight causing the forest floor to be darker than other areas. Some areas within the forest might have more vegetation and other areas might have less.[76] This results in rapidly changing light and dark areas which can be problematic for vision sensors. High dynamic range (HDR) is usually required to ensure that the camera captures image details in the bright areas in the environment as well as the shadows.[77] White balance and ISO also need to be managed correctly to ensure that the resulting image is exposed correctly.[78]

2.3.4 Complex Textures

The textures in forest environments are very diverse due to the presence of different plant species, trees and wildlife. Vision sensors interpret textures as local spatial variations within an image. This can include variations in colour and pixel intensities.[79] Since forests are highly textured environments, it is often difficult to capture the fine details within the scene, which can be detrimental to the quality of the maps produced by V-SLAM algorithms, especially when rendering detailed 3D maps.

2.3.5 Motion Blur

Atmospheric conditions, like the wind, cause movement among objects in forest landscapes, such as swaying tree branches and fluttering leaves. In addition to this, the camera itself will be moving relative to the environment during SLAM applications.[75] Factors such as the speed of the camera movement and whether the camera platform is stabilised will affect the quality of the footage captured. This could result in undesirable motion blur, in which areas of individual frames might appear blurry with a noticeable lack of detail.[80] Robust feature correspondence relies on sufficient detail being available in each frame to estimate the camera pose and locate the position of objects within the scene. As a result, forest landscapes prove to be challenging environments for V-SLAM systems to work effectively.

2.4 Traditional Feature Descriptors

2.4.1 Overview of Feature Descriptors

Feature description techniques aim to identify areas of an image that can be easily re-identified from different camera viewpoints. The first step involves extracting the points of interest from the image. The objective is to identify features that are robust to rotation, scale and changes of illumination in order to allow feature correspondence algorithms to accurately track features between frames. Feature description involves creating vectors or matrices to store information about the features such as the grey level of the pixels or the textures of the surrounding pixels. Descriptors can be local gradient-based, image intensity-based or learning-based.[81]

2.4.2 Harris Corner Detector

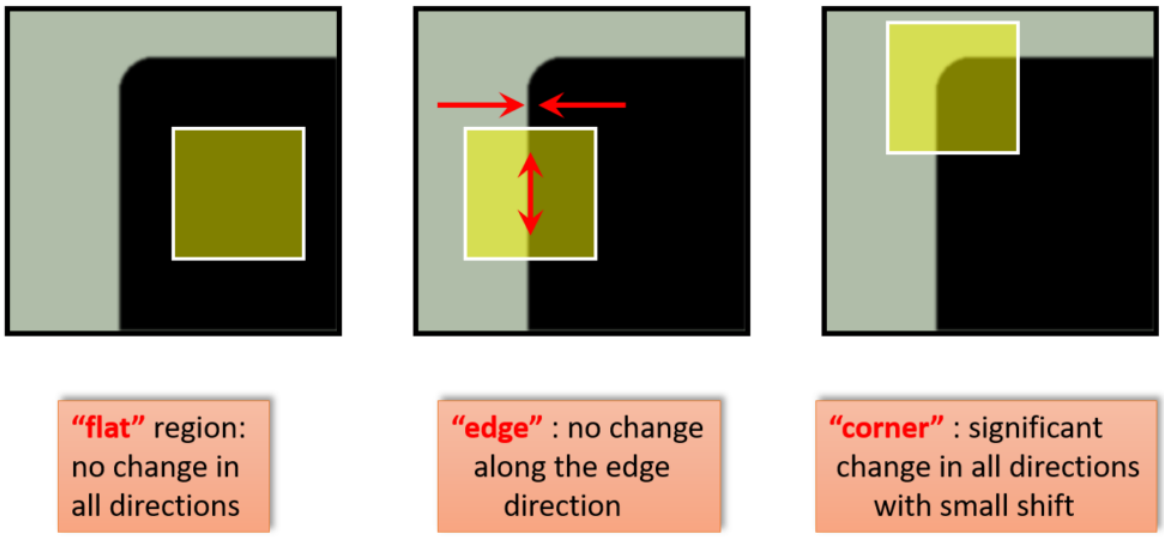

Figure 2.7: Harris Corner Detector [82]

Harris et al. proposed the Harris corner detector [83] shown in Figure 2.7 that identifies corners of objects as useful points, since they are invariant to rotation and can easily be tracked between frames.

$$E(u,v) = \sum_{(x,y)\in W} \underbrace{w(x,y)}_{\text{window function}} \Big[\underbrace{I(x+u,y+v)}_{\text{shifted intensity}} - \underbrace{I(x,y)}_{\text{intensity}}\Big]^2 \tag{2.1}$$

$$E(u,v) \approx \begin{bmatrix} u & v \end{bmatrix} M \begin{bmatrix} u \\ v \end{bmatrix} \tag{2.2}$$

Equation 2.1 shows how the window is shifted across the pixels in the image in both horizontal and vertical directions which are given by (u, v). Rectangular or Gaussian functions can be used for the window function. The aim of the Harris corner detector is to maximise $E(u, v)$ which is the difference between the original pixel intensities and the pixel intensities in the shifted window. This is achieved using the first-order approximation of the Taylor expansion of equation 2.1, the results of which can be seen in equation 2.2.[84]

$$M = \sum_{x,y} w(x,y) \begin{bmatrix} I_x I_x & I_x I_y \\ I_x I_y & I_y I_y \end{bmatrix} \tag{2.3}$$

$$R = \det(M) - k\,(trace(M))^2 \tag{2.4}$$

$$\det(M) = \lambda_1 \lambda_2 \tag{2.5}$$

$$trace(M) = \lambda_1 + \lambda_2 \tag{2.6}$$

Equation 2.3 shows that M is a 2x2 matrix which is computed using the products of the image derivatives I_x and I_y. Harris et al. described a cornerness function R as shown in equation 2.4. This can be computed using the determinant and trace of M as given by equations 2.5 and 2.6 respectively, where λ_1 and λ_2 are the eigenvalues of M. The function R can be used to determine if the change in intensities represents a flat region, an edge or a corner.[84]

If λ_1 and λ_2 are both small, $|R|$ will also be small, which means that the region is flat since there is no change in pixel intensities in any direction. Edges only occur when $\lambda_1 >> \lambda_2$ or vice-versa, so $R < 0$ which represents a change in only one direction. Corners are only present when λ_1 and λ_2 are both large and $\lambda_1 \approx \lambda_2$ which causes R to be large as there is a significant change in intensities regardless of the direction in which the window was shifted.[84]

The Harris corner detector has proven to be useful to detect corners which are invariant to rotation. Shifting a window along a rotated direction will still produce the same difference in intensities for a corner. However, the Harris corner detector is not invariant to scale. Hence, other feature description methods have been explored as alternative options.

2.4.3 SIFT

Scale-Invariant Feature Transform (SIFT) [85] is a feature description method that identifies points of interest within an image which are characterised by feature vectors. SIFT uses local features which are both scale and rotation invariant. SIFT features are also robust to variations in light intensity within an image.

Scale-space extrema detection is used to consider different scales via a difference of Gaussians (DoG) filter.[86] Keypoints are localised at potential features to improve the estimates for the location and scale. Orientations are then assigned to the features using the directions of neighbouring points which form a histogram of gradients for each sub-region within an area of the image. The size of the SIFT feature descriptor vector will be determined by the number of sub-regions as well as the number of bins used in the histograms. To maintain rotation invariant features, the dominant orientation in the histogram will be selected based on the gradient magnitude which receives the majority vote from the pixels in the neighbourhood.[81]

However, there are several calculations required to determine the gradient magnitudes and orientations for SIFT descriptors. This means that actual implementations of SIFT are often quite slow and therefore unsuitable for SLAM applications, in which real-time behaviour is desirable.[87]

2.4.4 SURF

Speeded Up Robust Features (SURF) [88] is a feature description technique designed to improve the speed of SIFT. Instead of calculating gradient magnitudes in multiple orientations as seen in the SIFT algorithm, SURF calculates gradients solely in the x and y directions through the use of Haar wavelets as shown in Fig 2.8.

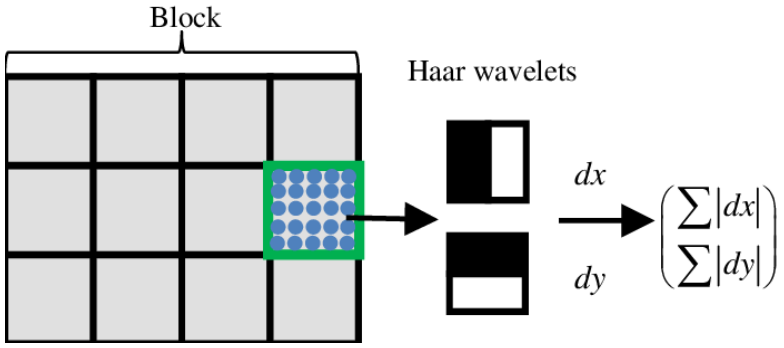

Figure 2.8: Haar wavelets in SURF feature descriptor [89]

Since there are only two orientations to consider, horizontal and vertical, pixel intensities are weighted with either $+1$ or -1. Summing these is therefore much faster for a given sub-region, thus allowing SURF to be significantly faster than SIFT. Despite this, SURF feature descriptor vectors are still quite large and feature correspondence techniques often struggle to match SURF features when there are large angles of rotation involved.[90]

2.4.5 BRIEF

Binary Robust Independent Elementary Features (BRIEF) [91] is a feature description technique designed to improve upon the performance of SURF to achieve even faster speeds. Image patches are used to obtain binary strings as the features, which can be efficiently matched using the Hamming distance,[92] which measures the number of places in which the binary strings differ in values.

$$\tau(p; x, y) = \begin{cases} 1 & \text{if } p(x) < p(y) \\ 0 & \text{otherwise} \end{cases} \tag{2.7}$$

The smoothed intensities $p(x)$ and $p(y)$ of randomly sampled pixels within a patch p are compared against each other to result in a binary output as shown in equation 2.7. n_d pairs of points x and y are randomly sampled to lead to an n_d-dimensional bitstring, where n_d is typically 128, 256 or 512.

Nevertheless, despite the accelerated computation, BRIEF descriptors are neither scale nor rotation invariant.[93] Robotic platforms moving within forest environments will inevitably rotate in order to avoid obstacles and navigate difficult terrain. Different scales will also be present in forest scenes, since background foliage will be much further away than trees that are close to the camera. As such, BRIEF descriptors are not suitable to use as features within a V-SLAM system designed to operate within forest environments.

2.4.6 FAST

Features from Accelerated Segment Test (FAST) [94] is a feature description method aimed at improving the speed of the Harris corner detector. Corners are identified based on the brightness of 16 pixels that form a ring around a pixel p that could be a potential corner within a given image patch. If the intensities of n pixels surrounding p are brighter than $I(p) + t$ or darker than $I(p) - t$ where t is a threshold, p can be considered a corner.

The top, bottom, left and rightmost pixels within the ring are evaluated first to quickly reject non-corners. If more than one of these pixels do not meet the brightness criteria, the candidate is not a corner. However, the high-speed test struggles when $n < 12$ and also depends on the distribution of corners. Hence, a machine learning model was used in order to increase reliability, in which a decision tree was trained to successfully classify corners. FAST converts the decision tree into if-then-else statements in C to identify corners quickly.

Although FAST exhibits the performance required to include it as part of a SLAM system, the use of corners as features is sub-optimal for forest scenes. Forest environments rarely contain objects with well defined corners that can easily be identified and tracked between frames. Therefore, other types of features have been explored as more robust alternatives for forest environments.

2.5 Feature Correspondence in Forests

2.5.1 Tree Parameter Extraction Model

Vision sensors in forest environments suffer from many challenges, such as moving branches and leaves which can cause motion blur, and dappled light which results in brightness changes within the image. Traditional feature descriptors often struggle to capture points that are resilient to these challenges in forest scenes, which means they cannot be reliably tracked between camera frames.

Mapping a forest environment relies heavily on accurate feature correspondence which is why Benny [15] explored the use of tree trunks as robust features that can be easily tracked between frames. Tree trunks are stationary landmarks in dynamic forest scenes, which makes them ideal to use as robust features. The Tree Parameter Extraction Model (TPEM) is a machine learning model that was trained to identify tree trunk diameters, inclination of trees and felling cut positions as distinctive landmarks to result in reliable keypoints along the tree trunks.

ResNet-50, ResNet-101 and ResNeXt-101 [95, 96] were compared as the backbone architectures for the TPEM. To reduce overfitting, models were pre-trained on a synthetic dataset (SynthTree43k) [16] before being fine-tuned on the CanaTree100 dataset. However, all three models exhibited better performance without pre-training on the synthetic dataset first, and ResNeXt-101 had the highest average precision (AP) score so it was selected as the backbone architecture of choice.

Each identified keypoint has a corresponding x and y coordinate which is then converted into an OpenCV keypoint object. BRIEF descriptors are then assigned to these tree trunk keypoints to enable them to be tracked across consecutive frames as shown in Figure 2.9.

Figure 2.9: TPEM feature correspondence in forest environment [15]

However, the TPEM uses the base of the tree trunks to determine the inclination parameter. If the base is occluded, keypoints can be positioned incorrectly, which results in incorrect feature correspondences. Incorrect positioning of keypoints on the edges of the tree trunks can cause background and foreground elements of the scene to be confused with each other, especially when depth values are included. As such, Benny also included ORB features to improve robustness.

2.5.2 ORB

Oriented FAST and Rotated BRIEF (ORB) [18] features combine the advantages of the speeds of FAST and BRIEF with the added benefit of rotation invariance. FAST is used to identify keypoints, with the N best keypoints being selected based on the responses from the Harris corner detector. To ensure that features are considered at multiple scales, a scale pyramid of the image is used.

ORB then employs the use of intensity centroids to consider orientations, as this is not done by FAST. An image patch is considered around a corner point, where the moments of the patch are calculated using Equation 2.8.

$$m_{pq} = \sum_{x,y} x^p y^q I(x, y) \tag{2.8}$$

These moments are then used to compute the centroid as shown in Equation 2.9.

$$C = \left(\frac{m_{10}}{m_{00}}, \frac{m_{01}}{m_{00}} \right) \tag{2.9}$$

A vector is formed from the centre of the corner to the centroid. The orientation can therefore be determined by the direction of this vector as given by Equation 2.10.

$$\theta = atan2(m_{01}, m_{10}) \tag{2.10}$$

BRIEF feature descriptors are then used, although traditional BRIEF does not perform well when there is rotation involved. Therefore, ORB steers the BRIEF descriptors relative to the orientation of the keypoints. ORB feature correspondence involves multi-probe Locality Sensitive Hashing (LSH) [97] where hash tables are used to store hashed feature points in distinct buckets. During feature correspondence, neighbouring buckets are selected and the elements are compared using brute-force matching to find the features. Figure 2.10 shows ORB feature correspondence being used to match features in a forest scene.

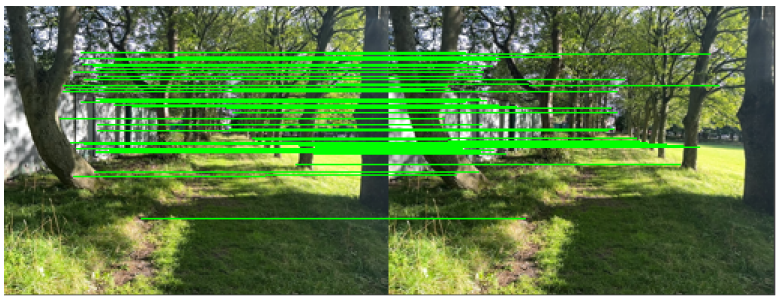

Figure 2.10: ORB feature correspondence in forest environment [15]

Benny experimented with solely using ORB features as well as ORB features in conjunction with the features from the TPEM. It was shown that the inclusion of the tree keypoints reduced both the translation and rotation errors of the camera pose estimates, since the tree trunks provided reliable landmarks. However, the state-of-the-art SuperGlue feature correspondence technique outperformed both methods.

2.5.3 SuperGlue

SuperGlue [19] is a feature correspondence technique that differs greatly from traditional methods. It attempts to understand the geometric transformations of the underlying 3D scene, which makes it ideal for challenging outdoor scenarios such as forest environments. SuperGlue is based on a graph neural network architecture which learns feature matching and rejects any points that cannot be matched. It can match many different types of low-level features but is typically used to match SuperPoint [20] features, which are interest points that have been identified via a self-supervised framework. SuperPoint uses Homographic Adaptation (HA) to consistently identify a more extensive range of interest points.

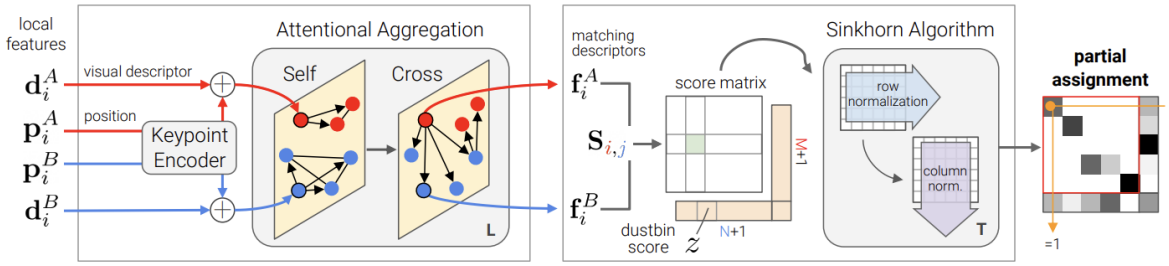

Figure 2.11: SuperGlue Architecture [19]

SuperGlue consists of two main aspects, an attentional graph neural network and an optimal matching layer as shown in Fig 2.11. A Multi-Layer Perceptron (MLP) is used to form a high-dimensional vector containing the x and y position coordinates of the keypoints as well as their descriptors. The use of self and cross attention enables the graph neural network to interpret both the appearance and position of the keypoints. Attentional aggregation is used to build a graph between the keypoints that form the nodes. This allows SuperGlue to focus on specific keypoints for certain attributes and to consider other similar keypoints nearby. This learning based approach is particularly effective, which is why SuperGlue outperforms traditional feature correspondence algorithms, since they cannot exhibit the same type of behaviour to understand the underlying scene.

The second part of SuperGlue is used to solve an optimal transport problem where the scores for the predictions of matching descriptors are maximised. Unmatched keypoints are discarded into dustbins, which allows SuperGlue to perform better in scenarios that involve occlusions. This is particularly useful in forest environments, since there might be occlusions at certain camera angles due to dense foliage. Optimisation of the score matrix is implemented through the use of the Sinkhorn algorithm. This allows SuperGlue to perform real-time feature correspondences, since all components of the SuperGlue architecture are differentiable, which means computations can be done efficiently using GPUs. This makes SuperGlue a suitable choice as a feature correspondence technique for use within a V-SLAM system, since real-time behaviour is desirable.

Figure 2.12 shows the SuperGlue feature correspondences in a forest scene. When comparing the different feature correspondence methods on the same forest scene, it is clear that SuperGlue outperforms both the TPEM and ORB. The distribution of keypoints in the scene is far more spread out in Figure 2.12 compared to both Figures 2.9 and 2.10. This shows that SuperGlue is able to understand the underlying features of the scene in a much more comprehensive way.

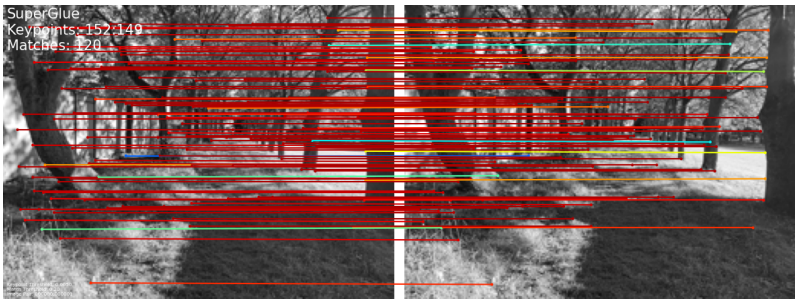

Figure 2.12: SuperGlue feature correspondence in forest environment [15]

Since the TPEM focuses solely on tree trunks, often there are very few keypoints to match, and frames in which no tree trunks are present will result in no feature correspondences. ORB feature correspondences tend to focus on more discernable aspects of the environment such as the trees and the wall. Feature correspondences that are close together in one part of the frame can be problematic, especially if that part of the frame is occluded in subsequent frames.

SuperGlue matches points not only on the trees and the wall, but also on the ground. Several feature points on the blades of grass were successfully matched by SuperGlue but ignored by the other feature correspondence methods. In forest environments, it is important to have feature correspondences that cover the entire frame. Since the scenes are very dynamic and can change dramatically from one frame to the next, it is unreliable to focus on one specific part of the scene.

	Translation Error (m)			Rotation Error (deg)		
	Min	Max	RMSE	Min	Max	RMSE
ORB	0.383	3.471	2.016	7.775	21.211	14.142
SuperGlue	0.275	3.638	1.871	7.618	17.051	11.749
Tree Parameters: TPEM + ORB	0.812	7.712	3.944	6.840	73.124	38.570
Tree Parameters: ORB	0.419	13.609	6.381	8.229	83.135	45.405

Figure 2.13: Trajectory errors for different feature correspondence methods [15]

Figure 2.13 shows the absolute trajectory errors calculated for each of the feature correspondence methods that Benny tested. SuperGlue had the lowest RMSE error for both translation and rotation. However, since the data collection setup involved the use of the iPhone Stray Scanner app, the accuracies of ground truth measurements are unclear. Nevertheless, the relative performance of SuperGlue has shown it to be the best feature correspondence method out of the ones which were tested.

However, SuperGlue is not without its disadvantages, especially when being deployed in forest environments. This is due to the fact that SuperGlue was trained on the Oxford and Paris datasets,[98] which means it generalises well for several different scenarios, but does not perform extremely well in a specific domain, such as forest environments. Much of the training examples focused on indoor scenes, whilst the outdoor examples typically only covered urban areas. This means that SuperGlue has not had sufficient exposure to rural environments for it to be able to learn how to match the features that are present in forest scenes.

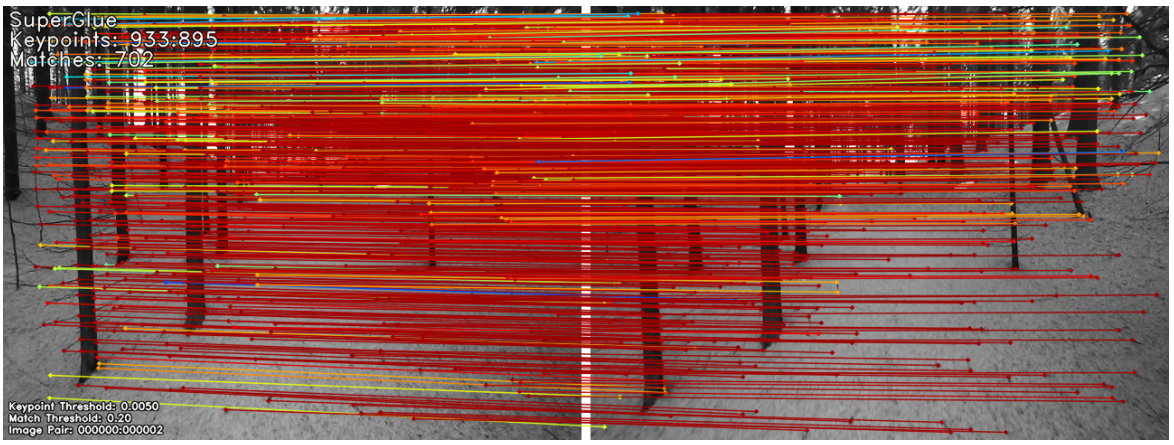

Figure 2.14: SuperGlue matches in FinnWoodlands static scene [99]

Figure 2.14 shows the SuperGlue feature correspondences on a scene taken from the FinnWoodlands dataset.[99] The red colours indicate a high confidence in the matches, whilst the blue colours indicate a lower confidence. In this scene, the trees cover much of the frame, which provide easily identifiable features for SuperGlue to match across consecutive frames, hence the large numbers of correct matches.

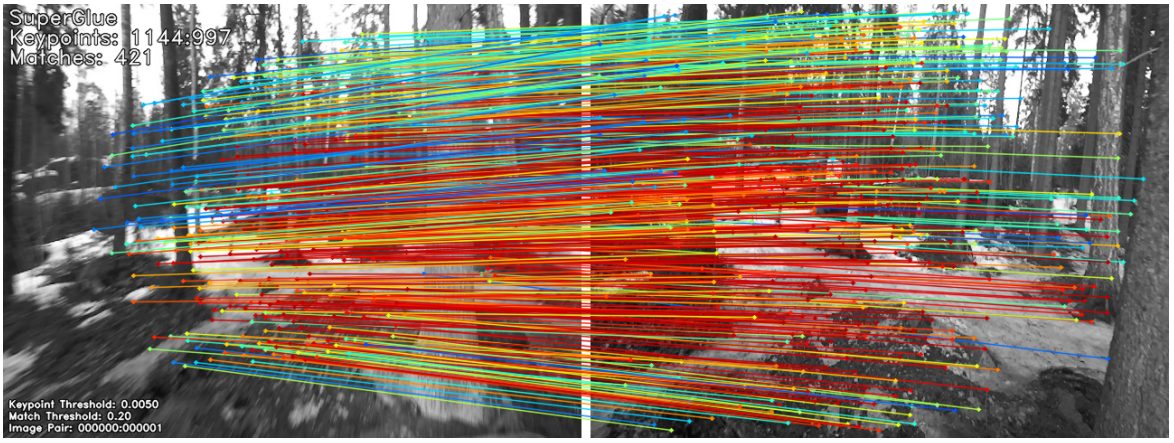

Figure 2.15: SuperGlue matches in FinnWoodlands moving scene [99]

Figure 2.15 shows far fewer matches and also a much lower confidence as shown by the colours. Motion blur is present as well as overexposed areas of the sky. SuperGlue focuses mainly on the static trees, whereas if it had learned the underlying features of forest scenes, it might be able to match other features to account for this.

Chapter 3

Project Plan

There are two main objectives for this project. The first objective is to build a dataset that will allow researchers to train and evaluate SLAM systems that are specifically designed to work in forest environments. The second objective is to implement a Visual SLAM system that incorporates the SuperGlue feature correspondence technique to perform SLAM tasks on the forest dataset.

The dataset will consist of footage captured from an RGB-D camera moving through a forest. It will contain RGB images, depth images, 3D point clouds and ground truth location data. The estimated camera poses and SLAM maps from the SuperGlue SLAM system will also be included to enable comparisons with future methods.

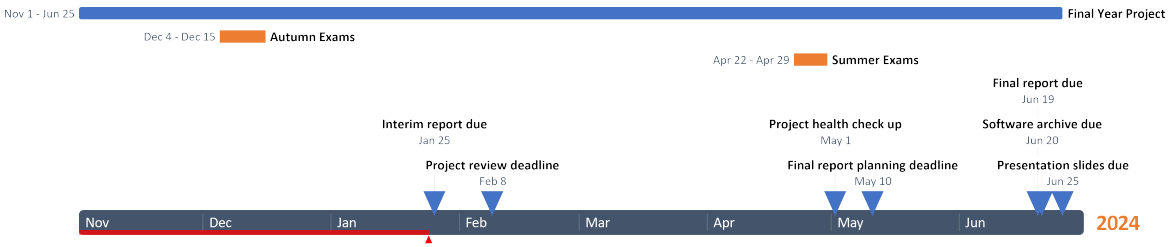

Figure 3.1: Gantt Chart with Project Deadlines

Figure 3.1 shows a timeline with the deadlines for the project deliverables. I have also included exam periods to indicate times when I will not be working on the project. This will ensure that I keep on track and meet all deadlines.

Figure 3.2: Gantt Chart with Project Plan

Figure 3.2 shows a high-level overview of the tasks that will need to be completed from now until the end of the project. I will work on the report throughout the project, but additional time has been left at the end to account for setbacks.

Data collection will initially be done by walking through a forest whilst holding an RGB-D camera. Concurrently, I will work on building a data processing pipeline, which will involve generating 3D point clouds from the depth maps and storing ground truth location data for each testing sequence.

Once sufficient data has been captured, I can start implementing the SuperGlue SLAM system. This will form the majority of the implementation work for the project due to the significant challenges posed by the complexities of SLAM systems. Thus far, I have managed to recreate the results from the SuperGlue paper as shown in Figure 3.3.

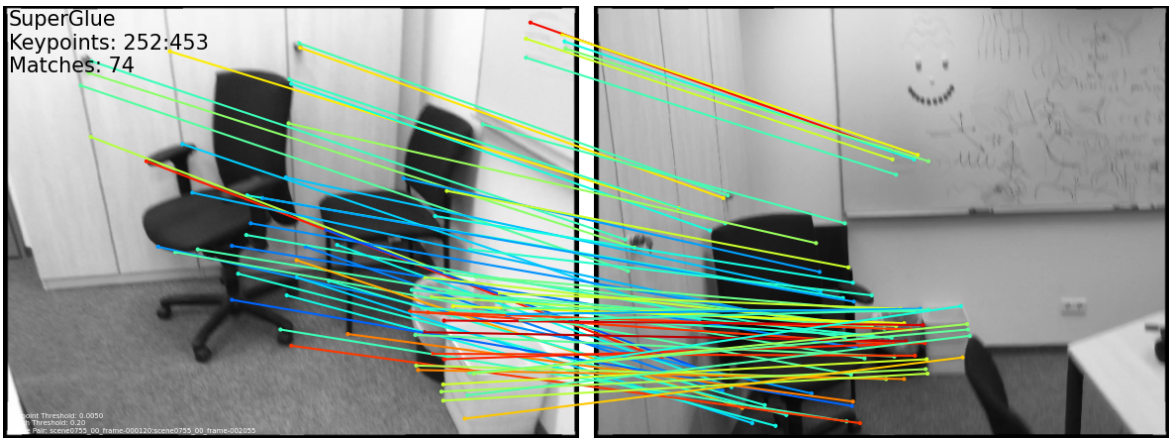

Figure 3.3: Recreating result from SuperGlue Paper [19]

The next steps will involve applying SuperGlue to the forest dataset. The SLAM system will use these matches to build a map incrementally, fusing the 3D point clouds with the RGB data to form dense 3D maps. Loop closure detection and bundle adjustment will also need to be incorporated to achieve an accurate V-SLAM system. The accuracies of the pose estimations will be compared against ORB-SLAM3,[100] which is a V-SLAM system that uses the ORB feature correspondence method. The SLAM maps generated by each system will also be compared against each other. Once the data pipeline has been setup, the dataset can be increased by doing additional data collection in forests. Depending on how much time is available, additional hand-held data can be collected, or the RGB-D camera can be mounted to a drone to collect data that is more representative of the use cases for SLAM systems.

If there is sufficient time, SuperGlue can also be re-trained on the forest dataset to see if that improves feature correspondences in forests compared to the standard model. Specific examples will need to be captured where there are fewer trees in view, so that the model learns to match other types of features in forests. Depth data can be used to obtain ground truth matches for pairs of images from the forest dataset. The performances of both models can then be compared, where negative log-likelihood is used as the loss function. As a backup plan, I have setup Microsoft Airsim [101] as a simulation environment, in case there are issues with real-world data collection due to inclement weather or malfunctioning equipment.

Chapter 4

Evaluation Plan

The successfulnesss of different aspects of the project can be evaluated qualitatively as well as quantitatively. A key goal of the project is to release a dataset that can be used by other researchers for SLAM in forest environments. To maximise the utility of the dataset, ground truth information should be provided where possible.

I am planning on using the Intel RealSense D455 depth camera [102] for real-world data collection. The ideal range of the depth measurements is listed as 0.6m to 6m. This should be sufficient for a SLAM system designed for forest environments. An IMU is also integrated into the camera platform. Having this data alongside the RGB and depth images will enable greater accuracy for pose estimations. There is also the possibility of including a Real-Time Kinematic (RTK) GPS module [103] which would provide ground truth location data, thus allowing the results of the SLAM system to be compared with the ground truth trajectories.

A useful dataset should contain a wide range of captured examples. There are several forests nearby to where I live in Slough, each containing different species of trees. Black Park Country Park has over 500 acres of woodland with different trees such as oak, beech and birch. Swinley Forest, located in Bracknell, contains coniferous pine trees spanning across 2600 acres. Burnham Beeches, is famous for beech and oak trees which make up almost 500 acres of woodland. Additionally, Wormwood Scrubs is approximately 190 acres of open parkland space situated in west London, and is particularly useful in the event that drones will be used for data collection, since it has specific time slots in which drones are permitted to fly.

Data should be captured in multiple different forest locations to allow for different tree species to be captured. Different parts of the forest should also be considered, and examples should include dense areas where there are lots of trees present, as well as semi-dense areas where the scene also includes grass and bushes. The captured examples should also cover scenarios where objects are occluded from one frame to the next, in order to test the ability of V-SLAM systems to deal with challenging scenarios. Data should be captured at different times of day and during variable weather conditions, to facilitate scenes containing dynamic lighting conditions. The weather conditions can be included as metadata for future reference.

The effectiveness of the SuperGlue SLAM system can be measured quantitatively by comparing the estimated camera poses to the ground truth trajectories, in which the RMSE error can be calculated for both translation and rotation as done by Benny. Having a dataset with sufficient examples will also enable fair comparisons with other V-SLAM algorithms such as ORB-SLAM3. The pose estimation errors can be compared quantitatively and the resultant dense maps which are generated can be analysed visually to compare the level of detail.

Demonstrating the SLAM system will involve running it in real-time on a recorded sequence from the dataset. A successful system will show a dense 3D map of the forest environment being generated incrementally on a screen. The camera pose estimates should also be displayed alongside the ground truth trajectories to enable visual analysis of how closely the predictions match the ground truth. The results should be stored in the dataset for each testing sequence, to ensure that future work has a baseline to compare against.

If sufficient time is available, data can be collected using an RGB-D camera attached to a drone that is manually controlled by a human pilot. This would be a useful addition to the dataset since it will more accurately represent the camera motion experienced by robotic platforms, which is where SLAM algorithms will actually be deployed. Proving that the SLAM system works on the data captured by a drone is therefore a big milestone towards having a fully autonomous drone that can accurately perform SLAM in forest environments.

If SuperGlue is re-trained on the forest dataset, it will need to be compared against the standard SuperGlue algorithm on the forest dataset to see if the feature correspondences have improved in the context of forest environments. If this leads to an improvement, two versions of the SuperGlue SLAM system can be compared, one with the original SuperGlue and one with the forest trained SuperGlue. As before, RMSE can be calculated for both translation and rotation errors for the camera pose estimates. This will enable a comparison to see if the additional training on the forest dataset has improved the pose estimates in the context of forest environments. Additionally, the SLAM maps can be visually compared to see if the additional domain specific training results in an improvement in the quality of the maps being generated.

Since the dataset is being designed with future research in mind, it is important that sufficient documentation is produced to enable other researchers to re-create the SLAM results. This can be done using the open-source Read the Docs [104] software documentation hosting platform. A Docker [105] container will be created to provide a consistent testing platform for users, which will eliminate the possibility of different operating systems and library versions from affecting the results. Additionally, the documentation will outline the steps needed to contribute to the dataset, as it will be quite limited initially due to a single person collecting the data manually.

Chapter 5

Ethical Issues

There are several ethical issues which need to be considered regarding the potential use cases for such technology. Forest based SLAM approaches could be used for autonomous drone navigation in military applications, which could have dire consequences, especially if a weapons system is attached to the drone. It could also be used in surveillance applications where targets are tracked without their knowledge, which could be a breach of privacy. However, the intention is that this technology is to be used solely for civilian applications in order to help humans carry out labour intensive tasks such as forest monitoring, or dangerous tasks such as search and rescue missions in order to minimise the risk exposed to humans.

Real-world data collection also poses some ethical concerns. Forest data will need to be collected in public woodland areas. There is the possibility that other members of the public who are not aware of the experiment might be captured in the footage. Steps will be taken to ensure that no people are captured in the recordings and if this does happen, the corresponding footage will be deleted. Additionally, the forest environment will be respected, and care will be taken to avoid disturbing wildlife in the area.

Health and safety is also something that needs to be considered, especially if drones are being used as part of the experimental setup. The guidelines set out by the Civil Aviation Authority (CAA) will need to be followed, and appropriate risk assessments will need to be conducted before any flights take place.

This project will use open-source code from the SuperGluePretrainedNetwork [106] repository which is available on Github. The licence permits academic usage as well as non-profit organisation non-commercial research usage. Since this project falls under the academic category, the code can be used, provided that the authors are acknowledged. To prevent code contamination, any open-source code used in this project will be accessed as a git submodule to make it clear which authors are responsible for each code file.

Bibliography

[1] Timothy J. Fahey. Forest Ecology. In Simon A Levin, editor, *Encyclopedia of Biodiversity (Second Edition)*, pages 528–536. Academic Press, Waltham, January 2013. ISBN 978-0-12-384720-1. doi: 10.1016/B978-0-12-384719-5.00058-7. URL https://www.sciencedirect.com/science/article/pii/B9780123847195000587. pages 3

[2] Barbara Koch, Matthias Dees, Jo Van Brusselen, H Leblon, and R Nilsson. Forestry applications. In *Advances in Photogrammetry, Remote Sensing and Spatial Information Sciences: 2008 ISPRS Congress Book*, pages 439–465. July 2008. ISBN 978-0-415-47805-2. doi: 10.1201/9780203888445.ch32. Journal Abbreviation: Advances in Photogrammetry, Remote Sensing and Spatial Information Sciences: 2008 ISPRS Congress Book. pages 3

[3] S. Karma, E. Zorba, G. C. Pallis, G. Statheropoulos, I. Balta, K. Mikedi, J. Vamvakari, A. Pappa, M. Chalaris, G. Xanthopoulos, and M. Statheropoulos. Use of unmanned vehicles in search and rescue operations in forest fires: Advantages and limitations observed in a field trial. *International Journal of Disaster Risk Reduction*, 13:307–312, September 2015. ISSN 2212-4209. doi: 10.1016/j.ijdrr.2015.07.009. URL https://www.sciencedirect.com/science/article/pii/S2212420915300364. pages 3

[4] Alexander Buchelt, Alexander Adrowitzer, Peter Kieseberg, Christoph Gollob, Arne Nothdurft, Sebastian Eresheim, Sebastian Tschiatschek, Karl Stampfer, and Andreas Holzinger. Exploring artificial intelligence for applications of drones in forest ecology and management. *Forest Ecology and Management*, 551:121530, January 2024. ISSN 0378-1127. doi: 10.1016/j.foreco.2023.121530. URL https://www.sciencedirect.com/science/article/pii/S0378112723007648. pages 3

[5] Chaoyue Niu, Callum Newlands, Klaus-Peter Zauner, and Danesh Tarapore. An embarrassingly simple approach for visual navigation of forest environments. *Frontiers in Robotics and AI*, 10, 2023. ISSN 2296-9144. URL https://www.frontiersin.org/articles/10.3389/frobt.2023.1086798. pages 3

[6] N Shalal, T Low, C McCarthy, and N Hancock. A REVIEW OF AUTONOMOUS NAVIGATION SYSTEMS IN AGRICULTURAL ENVIRONMENTS. pages 3

[7] Swagata Upreti and Manish Kumar. Perspectives of Global Positioning System (GPS) Applications. March 2008. pages 3

[8] Alex Souza Bastos and Hisashi Hasegawa. Behavior of GPS Signal Interruption Probability under Tree Canopies in Different Forest Conditions. *European Journal of Remote Sensing*, 46(1):613–622, January 2013. ISSN 2279-7254. doi: 10.5721/EuJRS20134636. URL https://www.tandfonline.com/doi/full/10.5721/EuJRS20134636. pages 3

[9] Weifeng Chen, Guangtao Shang, Aihong Ji, Chengjun Zhou, Xiyang Wang, Chonghui Xu, Zhenxiong Li, and Kai Hu. An Overview on Visual SLAM: From Tradition to Semantic. *Remote Sensing*, 14(13):3010, January 2022. ISSN 2072-4292. doi: 10.3390/rs14133010. URL https://www.mdpi.com/2072-4292/14/13/3010. Number: 13 Publisher: Multidisciplinary Digital Publishing Institute. pages 3

[10] Hamid Taheri and Zhao Chun Xia. SLAM; definition and evolution. *Engineering Applications of Artificial Intelligence*, 97:104032, January 2021. ISSN 0952-1976. doi: 10.1016/j.engappai.2020.104032. URL https://www.sciencedirect.com/science/article/pii/S0952197620303092. pages 3, 5

[11] Soonhac Hong and Cang Ye. A pose graph based visual SLAM algorithm for robot pose estimation. In *2014 World Automation Congress (WAC)*, pages 917–922, August 2014. doi: 10.1109/WAC.2014.6936197. URL https://ieeexplore.ieee.org/document/6936197. ISSN: 2154-4824. pages 3

[12] Ali Tourani, Hriday Bavle, Jose Luis Sanchez-Lopez, and Holger Voos. Visual SLAM: What Are the Current Trends and What to Expect? *Sensors (Basel, Switzerland)*, 22(23):9297, November 2022. ISSN 1424-8220. doi: 10.3390/s22239297. URL https://www.ncbi.nlm.nih.gov/pmc/articles/PMC9735432/. pages 3, 6, 8

[13] James Garforth and Barbara Webb. Visual Appearance Analysis of Forest Scenes for Monocular SLAM. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 1794–1800, Montreal, QC, Canada, May 2019. IEEE. ISBN 978-1-5386-6027-0. doi: 10.1109/ICRA.2019.8793771. URL https://ieeexplore.ieee.org/document/8793771/. pages 3

[14] James Garforth and Barbara Webb. Lost in the Woods? Place Recognition for Navigation in Difficult Forest Environments. *Frontiers in Robotics and AI*, 7, 2020. ISSN 2296-9144. URL https://www.frontiersin.org/articles/10.3389/frobt.2020.541770. pages 3

[15] Basil Benny, Bahadir Kocer, and Ronald Clark. Tracking and Mapping Forest Environments with Tree Trunk Parameter Estimations. September 2023. pages 4, 16, 17, 19

[16] Vincent Grondin, Jean-Michel Fortin, François Pomerleau, and Philippe Giguère. Tree detection and diameter estimation based on deep learning. *Forestry: An International Journal of Forest Research*, 96(2):264–276, April 2023. ISSN 0015-752X. doi: 10.1093/forestry/cpac043. URL https://doi.org/10.1093/forestry/cpac043. pages 4, 16

[17] Lorenzo Torresani, Vladimir Kolmogorov, and Carsten Rother. Feature Correspondence Via Graph Matching: Models and Global Optimization. In David Forsyth, Philip Torr, and Andrew Zisserman, editors, *Computer Vision – ECCV 2008*, volume 5303, pages 596–609. Springer Berlin Heidelberg, Berlin, Heidelberg, 2008. ISBN 978-3-540-88685-3 978-3-540-88688-4. doi: 10.1007/978-3-540-88688-4_44. URL http://link.springer.com/10.1007/978-3-540-88688-4_44. Series Title: Lecture Notes in Computer Science. pages 4

[18] Ethan Rublee, Vincent Rabaud, Kurt Konolige, and Gary Bradski. ORB: An efficient alternative to SIFT or SURF. In *2011 International Conference on Computer Vision*, pages 2564–2571, November 2011. doi: 10.1109/ICCV.2011.6126544. URL https://ieeexplore.ieee.org/document/6126544. ISSN: 2380-7504. pages 4, 17

[19] Paul-Edouard Sarlin, Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. SuperGlue: Learning Feature Matching with Graph Neural Networks, March 2020. URL http://arxiv.org/abs/1911.11763. arXiv:1911.11763 [cs]. pages 4, 18, 22

[20] 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, Salt Lake City, UT, USA, June 2018. IEEE. ISBN 978-1-5386-6100-0. doi: 10.1109/CVPRW.2018.00060. URL https://ieeexplore.ieee.org/document/8575521/. pages 4, 18

[21] Ziyu Li, Fangyang Ye, and Xinran Guan. 3D Point Cloud Reconstruction and SLAM as an Input, December 2021. URL http://arxiv.org/abs/2112.12907. arXiv:2112.12907 [cs]. pages 4

[22] Kyriaki A. Tychola, Ioannis Tsimperidis, and George A. Papakostas. On 3D Reconstruction Using RGB-D Cameras. *Digital*, 2(3):401–421, September 2022. ISSN 2673-6470. doi: 10.3390/digital2030022. URL https://www.mdpi.com/2673-6470/2/3/22. Number: 3 Publisher: Multidisciplinary Digital Publishing Institute. pages 4, 10

[23] Weifeng Chen, Chengjun Zhou, Guangtao Shang, Xiyang Wang, Zhenxiong Li, Chonghui Xu, and Kai Hu. SLAM Overview: From Single Sensor to Heterogeneous Fusion. *Remote Sensing*, 14(23):6033, January 2022. ISSN 2072-4292. doi: 10.3390/rs14236033. URL https://www.mdpi.com/2072-4292/14/23/6033. Number: 23 Publisher: Multidisciplinary Digital Publishing Institute. pages 5

[24] Yun Su, Ting Wang, Chen Yao, Shiliang Shao, and Zhidong Wang. GR-SLAM: Vision-Based Sensor Fusion SLAM for Ground Robots on Complex Terrain. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5096–5103, October 2020. doi: 10.1109/IROS45743.2020.

9341387. URL https://ieeexplore.ieee.org/document/9341387. ISSN: 2153-0866. pages 5

[25] Jonatan Scharff Willners, Yaniel Carreno, Shida Xu, Tomasz Łuczyński, Sean Katagiri, Joshua Roe, Èric Pairet, Yvan Petillot, and Sen Wang. Robust Underwater SLAM using Autonomous Relocalisation. *IFAC-PapersOnLine*, 54(16):273–280, January 2021. ISSN 2405-8963. doi: 10.1016/j.ifacol.2021.10.104. URL https://www.sciencedirect.com/science/article/pii/S2405896321015068. pages 5

[26] Abhishek Gupta and Xavier Fernando. Simultaneous Localization and Mapping (SLAM) and Data Fusion in Unmanned Aerial Vehicles: Recent Advances and Challenges. *Drones*, 6(4):85, April 2022. ISSN 2504-446X. doi: 10.3390/drones6040085. URL https://www.mdpi.com/2504-446X/6/4/85. Number: 4 Publisher: Multidisciplinary Digital Publishing Institute. pages 5

[27] Fabio T. Ramos, Juan Nieto, and Hugh F. Durrant-Whyte. Recognising and Modelling Landmarks to Close Loops in Outdoor SLAM. In *Proceedings 2007 IEEE International Conference on Robotics and Automation*, pages 2036–2041, April 2007. doi: 10.1109/ROBOT.2007.363621. URL https://ieeexplore.ieee.org/document/4209385. ISSN: 1050-4729. pages 5

[28] Hugh Durrant-Whyte and Tim Bailey. Simultaneous Localisation and Mapping (SLAM): Part I The Essential Algorithms. 2006. pages 5

[29] Timothy D. Barfoot. *State Estimation for Robotics*. Cambridge University Press, 1 edition, July 2017. ISBN 978-1-107-15939-6 978-1-316-67152-8. doi: 10.1017/9781316671528. URL https://www.cambridge.org/core/product/identifier/9781316671528/type/book. pages 5

[30] Andréa Macario Barros, Maugan Michel, Yoann Moline, Gwenolé Corre, and Frédérick Carrel. A Comprehensive Survey of Visual SLAM Algorithms. *Robotics*, 11(1):24, February 2022. ISSN 2218-6581. doi: 10.3390/robotics11010024. URL https://www.mdpi.com/2218-6581/11/1/24. Number: 1 Publisher: Multidisciplinary Digital Publishing Institute. pages 5

[31] Iman Abaspur Kazerouni, Luke Fitzgerald, Gerard Dooly, and Daniel Toal. A survey of state-of-the-art on visual SLAM. *Expert Systems with Applications*, 205:117734, November 2022. ISSN 0957-4174. doi: 10.1016/j.eswa.2022.117734. URL https://www.sciencedirect.com/science/article/pii/S0957417422010156. pages 6

[32] Fernando Molano Ortiz, Matteo Sammarco, Luís Henrique M. K. Costa, and Marcin Detyniecki. Applications and Services Using Vehicular Exteroceptive Sensors: A Survey. *IEEE Transactions on Intelligent Vehicles*, 8(1):949–969, January 2023. ISSN 2379-8904. doi: 10.1109/TIV.2022.3182218. URL https://ieeexplore.ieee.org/document/9795135. Conference Name: IEEE Transactions on Intelligent Vehicles. pages 6

[33] Ninad Mehendale and Srushti Neoge. Review on Lidar Technology, May 2020. URL https://papers.ssrn.com/abstract=3604309. pages 6

[34] Niraj Bhatta and Geetha Priya .M. RADAR and its applications. 10:1–9, January 2017. pages 6

[35] Yang Bai, Li Lu, Jerry Cheng, Jian Liu, Yingying Chen, and Jiadi Yu. Acoustic-based sensing and applications: A survey. *Computer Networks*, 181:107447, November 2020. ISSN 1389-1286. doi: 10.1016/j.comnet.2020.107447. URL https://www.sciencedirect.com/science/article/pii/S1389128620311282. pages 6

[36] Gerasimos G. Samatas and Theodore P. Pachidis. Inertial Measurement Units (IMUs) in Mobile Robots over the Last Five Years: A Review. *Designs*, 6(1):17, February 2022. ISSN 2411-9660. doi: 10.3390/designs6010017. URL https://www.mdpi.com/2411-9660/6/1/17. Number: 1 Publisher: Multidisciplinary Digital Publishing Institute. pages 6

[37] Ilham Faisal, Tito Purboyo, and Anton Ansori. A Review of Accelerometer Sensor and Gyroscope Sensor in IMU Sensors on Motion Capture. *Journal of Engineering and Applied Sciences*, 15:826–829, November 2019. doi: 10.36478/jeasci.2020.826.829. pages 6

[38] Andrew J. Davison, Ian D. Reid, Nicholas D. Molton, and Olivier Stasse. MonoSLAM: Real-Time Single Camera SLAM. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(6):1052–1067, June 2007. ISSN 0162-8828, 2160-9292. doi: 10.1109/TPAMI.2007.1049. URL http://ieeexplore.ieee.org/document/4160954/. pages 6

[39] T. R. Gayathri, R. P. Aneesh, and Gayathri R. Nayar. Feature based simultaneous localisation and mapping. In *2017 IEEE International Conference on Circuits and Systems (ICCS)*, pages 419–422, December 2017. doi: 10.1109/ICCS1.2017.8326034. URL https://ieeexplore.ieee.org/document/8326034. pages 6

[40] Dirk Hähnel, Sebastian Thrun, Ben Wegbreit, and Wolfram Burgard. Towards Lazy Data Association in SLAM. In Paolo Dario and Raja Chatila, editors, *Robotics Research. The Eleventh International Symposium*, Springer Tracts in Advanced Robotics, pages 421–431, Berlin, Heidelberg, 2005. Springer. ISBN 978-3-540-31508-7. doi: 10.1007/11008941_45. pages 6

[41] Wu Zhou, E Shiju, Zhenxin Cao, and Ying Dong. Review of SLAM Data Association Study. 2016. URL https://www.atlantis-press.com/article/25859609.pdf. pages 6

[42] Mordechai Ben-Ari and Francesco Mondada. Robotic Motion and Odometry. pages 63–93. January 2018. ISBN 978-3-319-62532-4. doi: 10.1007/978-3-319-62533-1_5. pages 7

[43] Viorela Ila, Josep M. Porta, and Juan Andrade-Cetto. Information-Based Compact Pose SLAM. *IEEE Transactions on Robotics*, 26(1):78–93, February 2010. ISSN 1941-0468. doi: 10.1109/TRO.2009.2034435. URL https://ieeexplore.ieee.org/abstract/document/5325904. Conference Name: IEEE Transactions on Robotics. pages 7

[44] Andrii Kudriashov, Tomasz Buratowski, Mariusz Giergiel, and Piotr Małka. SLAM as Probabilistic Robotics Framework Approach. In Andrii Kudriashov, Tomasz Buratowski, Mariusz Giergiel, and Piotr Małka, editors, *SLAM Techniques Application for Mobile Robot in Rough Terrain*, Mechanisms and Machine Science, pages 39–64. Springer International Publishing, Cham, 2020. ISBN 978-3-030-48981-6. doi: 10.1007/978-3-030-48981-6_3. URL https://doi.org/10.1007/978-3-030-48981-6_3. pages 7

[45] Aritra Mukherjee, Satyaki Chakraborty, and Sanjoy Kumar Saha. Detection of loop closure in SLAM: A DeconvNet based approach. *Applied Soft Computing*, 80:650–656, July 2019. ISSN 1568-4946. doi: 10.1016/j.asoc.2019.04.041. URL https://www.sciencedirect.com/science/article/pii/S1568494619302339. pages 7

[46] Alif Khairuddin, Shukor Talib, and Habibollah Haron. Review on simultaneous localization and mapping (SLAM). pages 85–90, November 2015. doi: 10.1109/ICCSCE.2015.7482163. pages 7

[47] Zachary Teed and Jia Deng. DROID-SLAM: Deep Visual SLAM for Monocular, Stereo, and RGB-D Cameras. February 2022. pages 7

[48] P Sankalprajan, Thrilochan Sharma, Hamsa Datta Perur, and Prithvi Sekhar Pagala. Comparative analysis of ROS based 2D and 3D SLAM algorithms for Autonomous Ground Vehicles. In *2020 International Conference for Emerging Technology (INCET)*, pages 1–6, June 2020. doi: 10.1109/INCET49848.2020.9154101. URL https://ieeexplore.ieee.org/document/9154101. pages 7

[49] Mahalakshmi Ramamurthy and Vasudevan Lakshminarayanan. Human Vision and Perception. In Robert Karlicek, Ching-Cherng Sun, Georges Zissis, and Ruiqing Ma, editors, *Handbook of Advanced Lighting Technology*, pages 1–23. Springer International Publishing, Cham, 2014. ISBN 978-3-319-00295-8. doi: 10.1007/978-3-319-00295-8_46-1. URL https://doi.org/10.1007/978-3-319-00295-8_46-1. pages 7

[50] Khalid Yousif, Alireza Bab-Hadiashar, and Reza Hoseinnezhad. An Overview to Visual Odometry and Visual SLAM: Applications to Mobile Robotics. *Intelligent Industrial Systems*, 1(4):289–311, December 2015. ISSN 2199-854X. doi: 10.1007/s40903-015-0032-7. URL https://doi.org/10.1007/s40903-015-0032-7. pages 7

[51] Mohammad O. A. Aqel, Mohammad H. Marhaban, M. Iqbal Saripan, and Napsiah Bt. Ismail. Review of visual odometry: types, approaches, challenges, and applications. *SpringerPlus*, 5(1):1897, October 2016. ISSN 2193-1801. doi: 10.1186/s40064-016-3573-7. URL https://doi.org/10.1186/s40064-016-3573-7. pages 7

[52] Jakob Engel, Thomas Schöps, and Daniel Cremers. LSD-SLAM: Large-Scale Direct Monocular SLAM. In David Fleet, Tomas Pajdla, Bernt Schiele, and Tinne Tuytelaars, editors, *Computer Vision – ECCV 2014*, volume 8690, pages 834–849. Springer International Publishing, Cham, 2014. ISBN 978-3-319-10604-5 978-3-319-10605-2. doi: 10.1007/978-3-319-10605-2_54. URL http://link.springer.com/10.1007/978-3-319-10605-2_54. Series Title: Lecture Notes in Computer Science. pages 7

[53] Rana Azzam, Tarek Taha, Shoudong Huang, and Yahya Zweiri. Feature-based visual simultaneous localization and mapping: a survey. *SN Applied Sciences*, 2(2):224, February 2020. ISSN 2523-3963, 2523-3971. doi: 10.1007/s42452-020-2001-3. URL http://link.springer.com/10.1007/s42452-020-2001-3. pages 7

[54] Zhengdong Zhang, Amr Suleiman, Luca Carlone, Vivienne Sze, and Sertac Karaman. Visual-Inertial Odometry on Chip: An Algorithm-and-Hardware Co-design Approach. In *Robotics: Science and Systems XIII*. Robotics: Science and Systems Foundation, July 2017. ISBN 978-0-9923747-3-0. doi: 10.15607/RSS.2017.XIII.028. URL http://www.roboticsproceedings.org/rss13/p28.pdf. pages 8

[55] Qiang Li, Ranyang Li, Kaifan Ji, and Wei Dai. Kalman Filter and Its Application. In *2015 8th International Conference on Intelligent Networks and Intelligent Systems (ICINIS)*, pages 74–77, November 2015. doi: 10.1109/ICINIS.2015.35. URL https://ieeexplore.ieee.org/document/7528889. pages 8

[56] Simon J Julier and Jeffrey K Uhlmann. A New Extension of the Kalman Filter to Nonlinear Systems. 1997. pages 8

[57] Georges Younes, Daniel Asmar, Elie Shammas, and John Zelek. Keyframe-based monocular SLAM: design, survey, and future directions. *Robotics and Autonomous Systems*, 98:67–88, December 2017. ISSN 0921-8890. doi: 10.1016/j.robot.2017.09.010. URL https://www.sciencedirect.com/science/article/pii/S0921889017300647. pages 8

[58] Nigel Joseph Bandeira Dias, Gustavo Teodoro Laureano, and Ronaldo Martins Da Costa. Keyframe Selection for Visual Localization and Mapping Tasks: A Systematic Literature Review. *Robotics*, 12(3):88, June 2023. ISSN 2218-6581. doi: 10.3390/robotics12030088. URL https://www.mdpi.com/2218-6581/12/3/88. Number: 3 Publisher: Multidisciplinary Digital Publishing Institute. pages 8

[59] Deep Learning for Computer Vision with Caffe and cuDNN, October 2014. URL https://developer.nvidia.com/blog/deep-learning-computer-vision-caffe-cudnn/. pages 9

[60] Wei Jiang, Kap Luk Chan, Mingjing Li, and Hongjiang Zhang. Mapping low-level features to high-level semantic concepts in region-based image retrieval. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 2, pages 244–249 vol. 2, June 2005. doi: 10.1109/CVPR.2005.220. URL https://ieeexplore.ieee.org/document/1467449. ISSN: 1063-6919. pages 9

[61] Zewen Xu, Zheng Rong, and Yihong Wu. A survey: which features are required for dynamic visual simultaneous localization and mapping? *Visual Computing for Industry, Biomedicine, and Art*, 4(1):20, July 2021. ISSN 2524-4442. doi: 10.1186/s42492-021-00086-w. URL https://doi.org/10.1186/s42492-021-00086-w. pages 9

[62] Kevin Trejos, Laura Rincón, Miguel Bolaños, José Fallas, and Leonardo Marín. 2D SLAM Algorithms Characterization, Calibration, and Comparison Considering Pose Error, Map Accuracy as Well as CPU and Memory Usage. *Sensors*, 22(18):6903, January 2022. ISSN 1424-8220. doi: 10.3390/s22186903. URL https://www.mdpi.com/1424-8220/22/18/6903. Number: 18 Publisher: Multidisciplinary Digital Publishing Institute. pages 9

[63] Ambroise Moreau, Matei Mancas, and Thierry Dutoit. Depth prediction from 2D images: A taxonomy and an evaluation study. *Image and Vision Computing*, 93:103825, January 2020. ISSN 0262-8856. doi: 10.1016/j.imavis.2019.11.003. URL https://www.sciencedirect.com/science/article/pii/S0262885619304184. pages 9

[64] Paul S. Heckbert. Survey of Texture Mapping. *IEEE Computer Graphics and Applications*, 6(11):56–67, November 1986. ISSN 0272-1716. doi: 10.1109/MCG.1986.276672. URL http://ieeexplore.ieee.org/document/4056764/. pages 9

[65] Bill Triggs, Philip F. McLauchlan, Richard I. Hartley, and Andrew W. Fitzgibbon. Bundle Adjustment — A Modern Synthesis. In Gerhard Goos, Juris Hartmanis, Jan Van Leeuwen, Bill Triggs, Andrew Zisserman, and Richard Szeliski, editors, *Vision Algorithms: Theory and Practice*, volume 1883, pages 298–372. Springer Berlin Heidelberg, Berlin, Heidelberg, 2000. ISBN 978-3-540-67973-8 978-3-540-44480-0. doi: 10.1007/3-540-44480-7_21. URL https://link.springer.com/10.1007/3-540-44480-7_21. Series Title: Lecture Notes in Computer Science. pages 9

[66] P. Newman and Kin Ho. SLAM-Loop Closing with Visually Salient Features. In *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*, pages 635–642, Barcelona, Spain, 2005. IEEE. ISBN 978-0-7803-8914-4.

doi: 10.1109/ROBOT.2005.1570189. URL http://ieeexplore.ieee.org/document/1570189/. pages 9

[67] Xiwu Zhang, Yan Su, and Xinhua Zhu. Loop closure detection for visual SLAM systems using convolutional neural network. In *2017 23rd International Conference on Automation and Computing (ICAC)*, pages 1–6, September 2017. doi: 10.23919/IConAC.2017.8082072. URL https://ieeexplore.ieee.org/document/8082072. pages 9

[68] Jianwei Li, Wei Gao, Yihong Wu, Yangdong Liu, and Yanfei Shen. High-quality indoor scene 3D reconstruction with RGB-D cameras: A brief review. *Computational Visual Media*, 8(3):369–393, September 2022. ISSN 2096-0662. doi: 10.1007/s41095-021-0250-8. URL https://doi.org/10.1007/s41095-021-0250-8. pages 10

[69] Michael Zollhöfer, Patrick Stotko, Andreas Görlitz, Christian Theobalt, Matthias Nießner, Reinhard Klein, and Andreas Kolb. State of the Art on 3D Reconstruction with RGB-D Cameras. *Computer Graphics Forum*, 37(2):625–652, May 2018. ISSN 0167-7055, 1467-8659. doi: 10.1111/cgf.13386. URL https://onlinelibrary.wiley.com/doi/10.1111/cgf.13386. pages 10

[70] Muhammad Bilal Shaikh and Douglas Chai. RGB-D Data-Based Action Recognition: A Review. *Sensors (Basel, Switzerland)*, 21(12):4246, June 2021. ISSN 1424-8220. doi: 10.3390/s21124246. URL https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8234200/. pages 10

[71] Francesco Secci and Andrea Ceccarelli. On failures of RGB cameras and their effects in autonomous driving applications. In *2020 IEEE 31st International Symposium on Software Reliability Engineering (ISSRE)*, pages 13–24, October 2020. doi: 10.1109/ISSRE5003.2020.00011. URL https://ieeexplore.ieee.org/document/9251080. ISSN: 2332-6549. pages 11

[72] Yuan-Peng Fan, Lei Wei, Lin Li, Lin Yang, Zi-Qiang Hu, Yuan-Hao Zheng, and Yu-Hao Wang. Research on the Modulation Transfer Function Detection Method of a Bayer Filter Color Camera. *Sensors*, 23(9):4446, January 2023. ISSN 1424-8220. doi: 10.3390/s23094446. URL https://www.mdpi.com/1424-8220/23/9/4446. Number: 9 Publisher: Multidisciplinary Digital Publishing Institute. pages 11

[73] Derya Akkaynak, Tali Treibitz, Bei Xiao, Umut A. Gürkan, Justine J. Allen, Utkan Demirci, and Roger T. Hanlon. Use of commercial off-the-shelf digital cameras for scientific data acquisition and scene-specific color calibration. *Journal of the Optical Society of America. A, Optics, image science, and vision*, 31(2):312–321, February 2014. ISSN 1084-7529. URL https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4028365/. pages 11

[74] Kamalkeet Kainth and Baljit Singh. Analysis of CCD and CMOS Sensor Based Images from Technical and Photographic Aspects, January 2020. URL https://papers.ssrn.com/abstract=3559236. pages 11

[75] Jakob Iglhaut, Carlos Cabo, Stefano Puliti, Livia Piermattei, James O'Connor, and Jacqueline Rosette. Structure from Motion Photogrammetry in Forestry: a Review. *Current Forestry Reports*, 5(3):155–168, September 2019. ISSN 2198-6436. doi: 10.1007/s40725-019-00094-3. URL https://doi.org/10.1007/s40725-019-00094-3. pages 11, 12

[76] Karen De Pauw, Pieter Sanczuk, Camille Meeussen, Leen Depauw, Emiel De Lombaerde, Sanne Govaert, Thomas Vanneste, Jörg Brunet, Sara A. O. Cousins, Cristina Gasperini, Per-Ola Hedwall, Giovanni Iacopetti, Jonathan Lenoir, Jan Plue, Federico Selvi, Fabien Spicher, Jaime Uria-Diez, Kris Verheyen, Pieter Vangansbeke, and Pieter De Frenne. Forest understorey communities respond strongly to light in interaction with forest structure, but not to microclimate warming. *New Phytologist*, 233 (1):219–235, 2022. ISSN 1469-8137. doi: 10.1111/nph.17803. URL https://onlinelibrary.wiley.com/doi/abs/10.1111/nph.17803. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1111/nph.17803. pages 11

[77] Paul E Debevec and Jitendra Malik. Recovering High Dynamic Range Radiance Maps from Photographs. 1997. pages 11

[78] Stephen Sagers and Ron Patterson. Film Speed or the ISO. March 2012. pages 11

[79] Omar Elezabi, Sebastien Guesney-Bodet, and Jean-Baptiste Thomas. Impact of Exposure and Illumination on Texture Classification Based on Raw Spectral Filter Array Images. *Sensors (Basel, Switzerland)*, 23(12):5443, June 2023. ISSN 1424-8220. doi: 10.3390/s23125443. URL https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10302469/. pages 11

[80] Tim Brooks and Jonathan T. Barron. Learning to Synthesize Motion Blur. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6833–6841, Long Beach, CA, USA, June 2019. IEEE. ISBN 978-1-72813-293-8. doi: 10.1109/CVPR.2019.00700. URL https://ieeexplore.ieee.org/document/8953368/. pages 12

[81] Wen Liu, Shuo Wang, Zhongliang Deng, and Hong Chen. A Review of Image Feature Descriptors in Visual Positioning. 2021. pages 12, 14

[82] datahacker.rs. OpenCV #013 Harris Corner Detector - Theory, July 2019. URL https://datahacker.rs/opencv-harris-corner-detector-part1/. pages 12

[83] C. Harris and M. Stephens. A Combined Corner and Edge Detector. In *Proceedings of the Alvey Vision Conference 1988*, pages 23.1–23.6, Manchester, 1988. Alvey Vision Club. doi: 10.5244/C.2.23. URL http://www.bmva.org/bmvc/1988/avc-88-023.html. pages 13

[84] OpenCV: Harris Corner Detection, . URL https://docs.opencv.org/4.x/dc/d0d/tutorial_py_features_harris.html. pages 13

[85] David G. Lowe. Distinctive Image Features from Scale-Invariant Keypoints. *International Journal of Computer Vision*, 60(2):91–110, November 2004. ISSN 0920-5691. doi: 10.1023/B:VISI.0000029664.99615.94. URL http://link.springer.com/10.1023/B:VISI.0000029664.99615.94. pages 14

[86] D.G. Lowe. Object recognition from local scale-invariant features. In *Proceedings of the Seventh IEEE International Conference on Computer Vision*, pages 1150–1157 vol.2, Kerkyra, Greece, 1999. IEEE. ISBN 978-0-7695-0164-2. doi: 10.1109/ICCV.1999.790410. URL http://ieeexplore.ieee.org/document/790410/. pages 14

[87] Jian Wu, Zhiming Cui, Victor S. Sheng, Pengpeng Zhao, Dongliang Su, and Shengrong Gong. A Comparative Study of SIFT and its Variants. *Measurement Science Review*, 13(3):122–131, June 2013. ISSN 1335-8871. doi: 10.2478/msr-2013-0021. URL https://content.sciendo.com/doi/10.2478/msr-2013-0021. pages 14

[88] Herbert Bay, Tinne Tuytelaars, and Luc Van Gool. SURF: Speeded Up Robust Features. In Aleš Leonardis, Horst Bischof, and Axel Pinz, editors, *Computer Vision – ECCV 2006*, Lecture Notes in Computer Science, pages 404–417, Berlin, Heidelberg, 2006. Springer. ISBN 978-3-540-33833-8. doi: 10.1007/11744023_32. pages 14

[89] Yuanxin Ye, Lorenzo Bruzzone, Jie Shan, Francesca Bovolo, and Qing Zhu. *A Fast and Robust Matching Framework for Multimodal Remote Sensing Image Registration*. August 2018. pages 14

[90] Tian Zhang, Rui Zhao, and Zhongsheng Chen. Application of Migration Image Registration Algorithm Based on Improved SURF in Remote Sensing Image Mosaic. *IEEE Access*, 8:163637–163645, 2020. ISSN 2169-3536. doi: 10.1109/ACCESS.2020.3020808. URL https://ieeexplore.ieee.org/document/9183911. Conference Name: IEEE Access. pages 14

[91] David Hutchison, Takeo Kanade, Josef Kittler, Jon M. Kleinberg, Friedemann Mattern, John C. Mitchell, Moni Naor, Oscar Nierstrasz, C. Pandu Rangan, Bernhard Steffen, Madhu Sudan, Demetri Terzopoulos, Doug Tygar, Moshe Y. Vardi, Gerhard Weikum, Michael Calonder, Vincent Lepetit, Christoph Strecha, and Pascal Fua. BRIEF: Binary Robust Independent Elementary Features. In Kostas Daniilidis, Petros Maragos, and Nikos Paragios, editors, *Computer Vision – ECCV 2010*, volume 6314, pages 778–792. Springer Berlin Heidelberg, Berlin, Heidelberg, 2010. ISBN 978-3-642-15560-4 978-3-642-15561-1. doi: 10.1007/978-3-642-15561-1_56. URL http://link.springer.com/10.1007/978-3-642-15561-1_56. Series Title: Lecture Notes in Computer Science. pages 15

[92] Mohammad Norouzi, David J Fleet, and Ruslan Salakhutdinov. Hamming Distance Metric Learning. pages 15

[93] Michal Kottman. The Color-BRIEF Feature Descriptor. pages 15

[94] Edward Rosten and Tom Drummond. Machine Learning for High-Speed Corner Detection. In Aleš Leonardis, Horst Bischof, and Axel Pinz, editors, *Computer Vision – ECCV 2006*, Lecture Notes in Computer Science, pages 430–443, Berlin, Heidelberg, 2006. Springer. ISBN 978-3-540-33833-8. doi: 10.1007/11744023_34. pages 15

[95] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, Las Vegas, NV, USA, June 2016. IEEE. ISBN 978-1-4673-8851-1. doi: 10.1109/CVPR.2016.90. URL http://ieeexplore.ieee.org/document/7780459/. pages 16

[96] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated Residual Transformations for Deep Neural Networks, April 2017. URL http://arxiv.org/abs/1611.05431. arXiv:1611.05431 [cs]. pages 16

[97] Aristides Gionis, Piotr Indyk, and Rajeev Motwani. Similarity Search in High Dimensions via Hashing. 1999. URL https://www.vldb.org/conf/1999/P49.pdf. pages 17

[98] Filip Radenovic, Ahmet Iscen, Giorgos Tolias, Yannis Avrithis, and Ondrej Chum. Revisiting Oxford and Paris: Large-Scale Image Retrieval Benchmarking. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5706–5715, Salt Lake City, UT, June 2018. IEEE. ISBN 978-1-5386-6420-9. doi: 10.1109/CVPR.2018.00598. URL https://ieeexplore.ieee.org/document/8578696/. pages 20

[99] Juan Lagos, Urho Lempiö, and Esa Rahtu. FinnWoodlands Dataset. In Rikke Gade, Michael Felsberg, and Joni-Kristian Kämäräinen, editors, *Image Analysis*, Lecture Notes in Computer Science, pages 95–110, Cham, 2023. Springer Nature Switzerland. ISBN 978-3-031-31435-3. doi: 10.1007/978-3-031-31435-3_7. pages 20

[100] Carlos Campos, Richard Elvira, Juan J. Gómez Rodríguez, José M. M. Montiel, and Juan D. Tardós. ORB-SLAM3: An Accurate Open-Source Library for Visual, Visual-Inertial and Multi-Map SLAM. *IEEE Transactions on Robotics*, 37(6):1874–1890, December 2021. ISSN 1552-3098, 1941-0468. doi: 10.1109/TRO.2021.3075644. URL http://arxiv.org/abs/2007.11898. arXiv:2007.11898 [cs]. pages 22

[101] Home - AirSim, 2017. URL https://microsoft.github.io/AirSim/. pages 22

[102] Introducing the Intel® RealSense™ Depth Camera D455, . URL https://www.intelrealsense.com/depth-camera-d455/. pages 23

[103] In-Su Lee and Linlin Ge. The performance of RTK-GPS for surveying under challenging environmental conditions. *Earth, Planets and Space*, 58(5):515–522, May 2006. ISSN 1880-5981. doi: 10.1186/BF03351948. URL https://doi.org/10.1186/BF03351948. pages 23

[104] Read the Docs Inc. Full featured documentation deployment platform. URL https://about.readthedocs.com/?ref=readthedocs.com. pages 24

[105] Docker: Accelerated Container Application Development, May 2022. URL https://www.docker.com/. pages 24

[106] magicleap/SuperGluePretrainedNetwork, 2020. URL https://github.com/magicleap/SuperGluePretrainedNetwork. original-date: 2020-03-17T19:32:12Z. pages 25