Robotic Navigation Using Privacy Preserving Vision

ASHWIN GUR

B.Eng (Mechatronics) Hons, B.Science (Computer Science)

Supervisor: Donald G. Dansereau

A thesis submitted in fulfilment of
the requirements for the degree of
B.Engineering (Mechatronics) Hons, B.Science (Computer Science)

School of Aeronautical, Mechanical and Mechatronic Engineering
Faculty of Engineering
The University of Sydney
Australia

2 November 2024

Abstract

Autonomous robotic systems frequently rely on camera vision to navigate environments. Collecting images of the scene introduces privacy risks if malicious actors gain access to the data. Privacy is valued across a range of application domains, from automatic vacuum cleaners in households to warehouse robots in commercial settings. There is a need to develop robots that can steer through environments using privacy-preserving vision. Existing methods in the literature apply privacy-preserving transformations after the sensitive data has already been digitised, which is vulnerable to attacks that can directly obtain the raw sensor data on the system.

In this thesis, we build upon a paper that proposes an inherently privacy-preserving architecture where images are captured and hashed entirely in the analogue domain. There is a knowledge gap in this domain because there is no proposed method to use privacy-preserving images to steer a robot within an environment. We present a reinforcement learning-based pipeline to train a robot to navigate indoor environments using only privacy-preserving images. Our system is capable of following closed-loop trajectories and identifying semantic waypoints along the path. We propose and evaluate various privacy-preserving hash functions to assess their effectiveness in successfully navigating the environments in a simulation. Additionally, we compare the performance of the privacy-preserving method against models trained on conventional camera images to demonstrate its feasibility.

Our results reveal that steering with full privacy-preserving vision is possible, but further work is required to refine the approach and enable it in practical scenarios. The contributions in this thesis are a step forward in developing privacy-preserving robots. This technology will positively impact commercial and household settings where privacy and confidentiality is valued.

Statement of Contributions

I hereby declare that the intellectual content of this thesis is the product of my own work and that all the assistance received in preparing this thesis and sources have been acknowledged. The work I contributed to includes

- I designed the research topic that was investigated in this thesis, with guidance and suggestions from my supervisor.
- I conducted the relevant background research and literature review presented in this thesis.
- I setup the reinforcement learning pipeline using open source tools.
- I implemented the privacy-preserving hashes, collected training data and evaluated results.

Acknowledgements

I would like to thank my supervisor, Dr. Dansereau, for his support and guidance. Your constant feedback has helped me immensely. I would also like to thank my peers who are also doing their thesis alongside me, our discussions have provided me with useful ideas and motivation.

Contents

List of Figures

CHAPTER 1

Introduction

1.1 Motivation

As advancements are made in autonomous robotics systems, privacy issues are also a growing concern. The benefit autonomous robotic systems can provide is immense, but widespread adoption is challenging until the trust and security of this technology can be guaranteed. From autonomous vacuum cleaners to warehouse assistants, robotic vision is utilised in a variety of applications requiring navigation through an environment. A critical concern with these systems is the compromise of user privacy if a malicious attacker gains access to the camera feed.

Robotic vacuum cleaners have gained popularity in recent years. These systems are equipped with many sensors including GPS and camera that collect detailed data on indoor environments [1]. The data collected is often sent to cloud servers for the training and improvement of robot models. This poses privacy concerns for the users of these products who are usually unaware of the data they are sharing. Data breaches can reveal sensitive data. In the case of house floor maps and camera images, there is abundant information to reveal a person's identity and day to day activities. There have been many recorded instances of privacy breaches in such systems, with robot camera feeds being hacked to spy on people [2], [3]. Privacy breaches extend beyond camera vision; a paper by Sami et al. [4] introduces a novel acoustic side-channel attack using lidar sensors, which are commonly installed on robot vacuum cleaners.

A survey by Eick et al. [5] revealed that less than 0.5% of robotics papers published since 1982 have mentioned the word ‘privacy’. Denning et al. [6] explore a series of potential security risks and attacks such as spying and psychological attacks. Prior work in the area of privacy-preserving vision includes methods such as face blurring, RGB-Depth cameras and 3D line clouds. These approaches do not provide the highest level of privacy and pose some vulnerabilities, which we discuss in further detail in the literature review. There is also a knowledge gap in using privacy-preserving vision for a robot steering application. A previous paper by Taras el a. [7] show that the localisation problem is solvable with their specific implementation of privacy-preserving vision. Further work is required to extend this to a moving robot.

Methods such as encryption are a low cost solution that can be handled entirely in software. However, the data is still susceptible to breaches even when it is stored on a cloud system with dedicated security [8], or due to human error and social engineering [9]–[11]. A recent report by the Office of the Australian Information Commissioner revealed the highest number of notifiable data breaches between January to June 2024 since late 2020 [12]. This motivates the research direction of this thesis, which focuses on using data that is inherently captured and stored in a secure format, which means any data breaches are unlikely to yield useful information.

For the context of this thesis, we define ‘robotic navigation’ as the process of controlling a robot’s steering direction and velocity to move from one point to another without colliding into objects in an environment. We aim to build a method of robotic navigation that uses only privacy preserving sensor data. In particular, we will use privacy preserving images that have gone through an analogue image hashing process developed by Taras et al. [7]. This process involves a single-pixel imaging technique that will provide enough information for a robot to understand where it is in the environment. There will not be enough information in the image to understand specific details in a scene such as human faces, furniture layout and

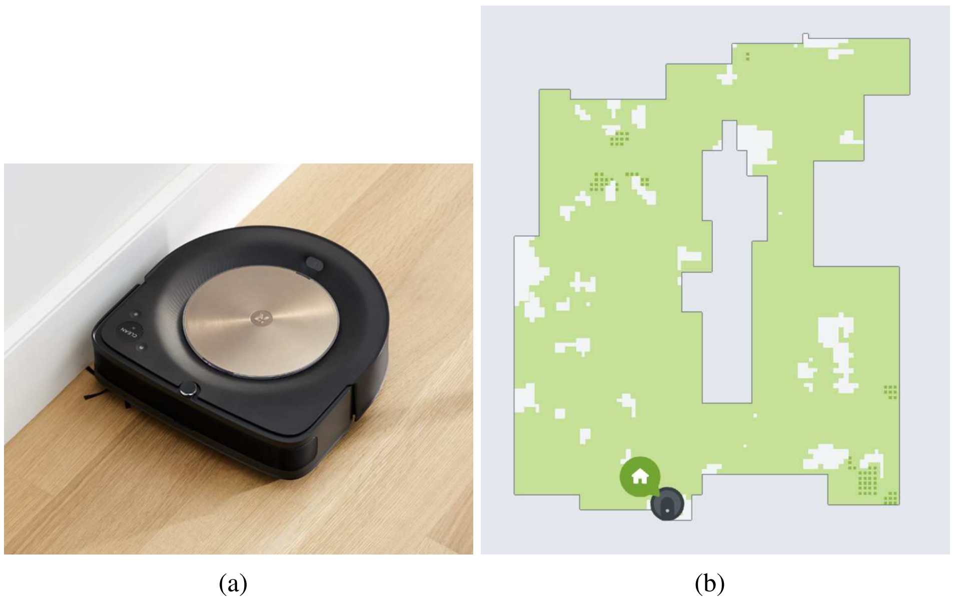


FIGURE 1.1: Example of how an automatic household vacuum cleaner captures sensitive data that is capable of mapping out an entire environment in high detail. (a) is a Roomba s Series [13] which provides a Clean Map Report [14] of the entire house. This sensitive data can be exploited by malicious actors as it provides a detailed layout of a house, including obstacles.

sensitive documents. The goal of this approach is to prevent an attacker that has complete access to the sensor feed from gaining any privacy breaking data from the scene. The focus of this thesis will be to demonstrate privacy preserving robotic navigation within a household environment. However, the principle contributions in this thesis can extend to other scenarios such as outdoor environments and drones.

1.2 Contributions

This thesis aims for the following contributions to knowledge:

(1) Design and implement a reinforcement learning pipeline to control a robot's steering and velocity using only privacy-preserving vision.

(2) Develop and test a number of hash functions compatible with the privacy-preserving architecture.
(3) Train and store a reinforcement learning model that can be re-used in future runs.
(4) Validate and establish the most promising privacy-preserving approach amongst various hash functions in a simulated environment.
(5) Compare the effectiveness and viability of a privacy-preserving approach against a non-privacy approach.

The contributions from this thesis will enable privacy preserving robots to be developed and used in real environments. As a consequence, greater trust and security in robotic systems can be enabled. This is a step forward in the wider adoption of robotics systems in many applications outlined below.

The development of privacy preserving robotic vision will have a positive impact in a variety of applications such as households, healthcare, businesses, governments and humanitarian efforts. The household sector has the greatest potential to benefit from this technology. Users can have autonomous driving robots in their homes to help with chores, elderly and disability support, delivery, entertainment and education. Ensuring user privacy will be crucial for the large scale use of such systems. Furthermore, business and government sectors can also greatly benefit from the productivity gained from privacy preserving autonomous robots. Tasks involving intellectual property and confidential data can be completed without the risk of attacks that could steal such information. Benefits also extend to humanitarian efforts, which include disaster relief and search and rescue operations.

1.3 Thesis outline

Chapter 2 provides the background information necessary to understand the methodology and results of the thesis. Chapter 3 provides an analysis of existing work that has been done to address the issue and its limitations. Chapter 4 provides the methodology of experiments performed to achieve the thesis goals. Chapter 5 explores the results of our experiments and compares the different approaches used. Chapter 6 discusses the results in the context of the literature. Chapter 7 concludes with the contributions this thesis has made, significance, limitations and future work.

CHAPTER 2

Background

2.1 Defining privacy

The International Association of Privacy Professionals defines the concept of privacy as “the right to be let alone, or freedom from interference or intrusion” [15]. A distinction exists between security and privacy: security primarily aims to protect data from malicious attacks and prevent the exploitation of stolen data for profit, while privacy focuses on the use and governance of personal data.

To what extent should a method or solution be deemed privacy-preserving? For example, one person may view 24/7 location tracking on their phone as acceptable, while another may consider it highly intrusive and only enable it selectively. A study by O’Neil [16] found that privacy concerns also vary across different demographics. In the context of this thesis, we define privacy preservation as the inability of an attacker to reconstruct the layout of an environment, including any human activities and objects within it.

2.2 Convolutional neural networks

Convolutional neural networks (CNNs) are a class of deep neural networks primarily designed to process and analyse visual data. In our scenario, this will be a privacy preserving image that has been hashed using a method outlined by Taras et al. [7] (further details in chapter 3).

CNNs excel at capturing spatial hierarchies of features in images, that will make it suitable for the task of mapping an input image to a steering direction for our approach. We will use the open source PyTorch library to create and train our model. The CIFAR10 example model outlined in a PyTorch tutorial was used as a starting point for our model architecture [17].

2.3 Reinforcement learning

Reinforcement learning (RL) is a type of machine learning where an agent learns to make decisions by interacting with its environment. After the agent takes an action, it receives a reward or penalty as feedback to help improve future runs. This reward function is different depending on desired task to be achieved. The key components of RL include:

- Agent: The learner or decision-maker (e.g.a robot).
- Environment: The world or system the agent interacts with (e.g.a race track).
- Actions: The choices an agent can make (e.g.steering and accelerating).
- State: The current situation of the environment (e.g.sensor input to a robot).
- Reward: Feedback from the environment based on the agent's action (e.g.a robot is rewarded for staying on the track and penalised for crashing).

Reinforcement learning differs from other machine learning approaches such as supervised and unsupervised learning. The agent learns through a trial and error process by interacting with the environment. It does not have labelled data, feedback is instead received in the form of rewards and penalties after performing actions.

2.3.1 Proximal policy optimisation

Proximal policy optimisation (PPO) is a specific reinforcement learning algorithm that uses policy gradient methods. The goal is to directly optimise the policy that governs the agent's actions. The algorithm is designed to update the policy without deviating too far from the existing policy which improves stability. PPO provides a good balance between performance and computational efficiency, which will allow for a higher volume of experimentation. Its versatility also means it is suited for continuous control tasks such as controlling a robot. A CNN can be used in combination with PPO. It acts as a feature extractor that can process our privacy-preserving images to help the agent make a decision on the action to perform.

CHAPTER 3

Literature review

The following literature review presents prior work relevant to our approach and existing methods for privacy-preserving vision. We discuss the limitations of previous work and show how our approach addresses and overcomes these challenges.

3.1 Privacy-preserving vision using analogue circuits

A novel idea that can drive our privacy preserving approach is the use of single-pixel imaging. Latorre-Carmona et al. [18] have demonstrated a method of reconstruction-free single pixel image classification. A classification can be performed by digitising the smallest amount of data. The confidence measure of the classification increases as more measurements are made. The approach outlined by Carmona et al. emphasises the computational advantage of single-pixel classification rather than the privacy benefits it could provide.

Taras et al. [7] have proposed a concept in the single-pixel imaging domain that is focused on maintaining privacy of a visual scene. This is done through a hashing method that prevents scene reconstruction, but provides enough details for different images in a scene to be differentiated. They demonstrated that this hashing method enables localisation to be performed. A new hashed image can be matched against an existing set of hashed images using a bag of words approach. A notable advantage in Taras' hashing process is its capability of being run in the analog domain. Analog processing provides greater resistance to cyberattacks, which we require for privacy-preservation. Upon digitisation, the image is already hashed so a malicious attacker in the system cannot intercept any direct representation of the scene. Because the hashing function does not store pixel locations and involves a degree of randomness, a brute force attack does not guarantee image reconstruction. Many images can map to the same input, but small changes in the scene leads to different hashed images. This characteristic makes single-pixel image hashing suitable for our scenario. The method by Taras et al. [7] has some limitations this thesis will aim to address. A method of navigation using this approach was not addressed. Furthermore, an important factor to consider is the rotational invariance of the hashing method. This is likely to cause issues with steering in scenarios where multiple parts of the environment have similar patterns. With some modifications, we will use this approach to steer a robot through a scene using only privacy-preserving images.

3.1.1 Single-pixel imaging

Single-pixel imaging is an alternative approach to traditional camera systems, focusing on simplified image acquisition methods that leverage computational techniques to reconstruct images. This approach allows imaging across a broader spectral range, reduces hardware complexity, and has privacy-preserving applications. Single-pixel imaging typically relies on compressive sampling, spatial light modulation, and a single photodetector to capture information.

Compressive sampling and the single-pixel camera

Duarte et al. [19] introduced a framework for single-pixel imaging through compressive sampling, enabling high-quality image reconstruction from limited data. In their design, a digital micro-mirror device (DMD) acts as a spatial light modulator, projecting a sequence of light patterns onto the scene. A single photodetector (or "single pixel") captures the intensity of light reflected from the scene. This data, although limited compared to a conventional camera sensor, can be computationally reconstructed to form a complete image. The compressive

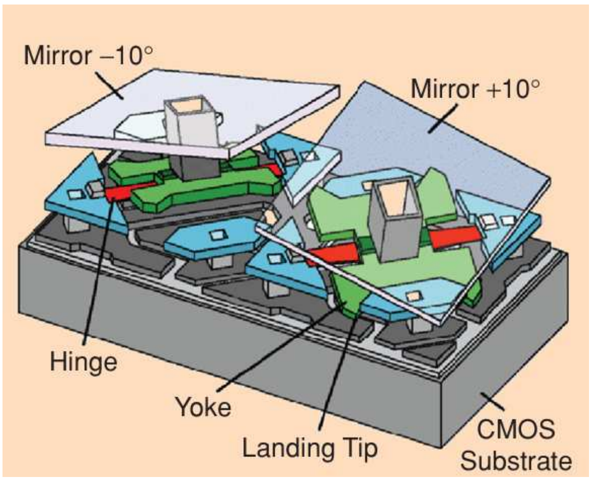


FIGURE 3.1: Digital micro-mirror device schematic [19].

sampling approach allows the single-pixel camera to efficiently capture information across diverse spectral ranges, where traditional silicon-based sensors are less effective, and enables the camera to operate with simpler, smaller, and less costly components.

Photodiode-based single-pixel cameras

Building upon these ideas, Jauregui-Sánchez et al. [20] explored single-pixel cameras (SPCs) using photodiodes as detectors. In this setup, the SPC illuminates the scene using a sequence of microstructured light patterns, encoded by a spatial light modulator, and then collects reflected light with a photodiode. The photodiode generates an electrical signal based on the captured light, which is processed computationally to reconstruct an image. This setup leverages the photodiode's simplicity, making it suitable for low-cost and low-power applications. However, signal quality is crucial for effective imaging, as various noise sources—such as photocurrent noise, dark current, and thermal noise—can degrade the output. The authors model these noise factors and study how the signal-to-noise ratio (SNR) varies with incident light power, wavelength, and temperature, comparing their numerical model's predictions with experimental SPC results.

Relevance and tradeoffs

Single-pixel imaging holds promise in fields requiring privacy-preserving or spectral-specific

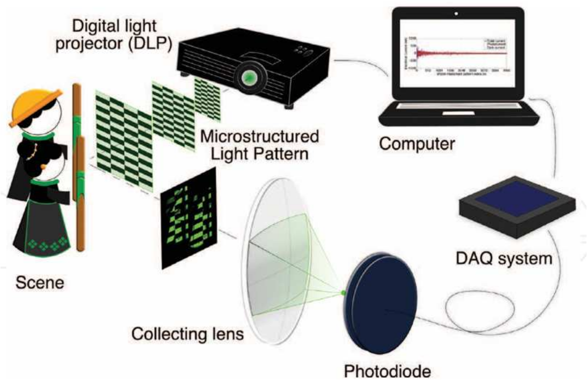


FIGURE 3.2: Single-pixel camera setup using photodiodes as detectors [20].

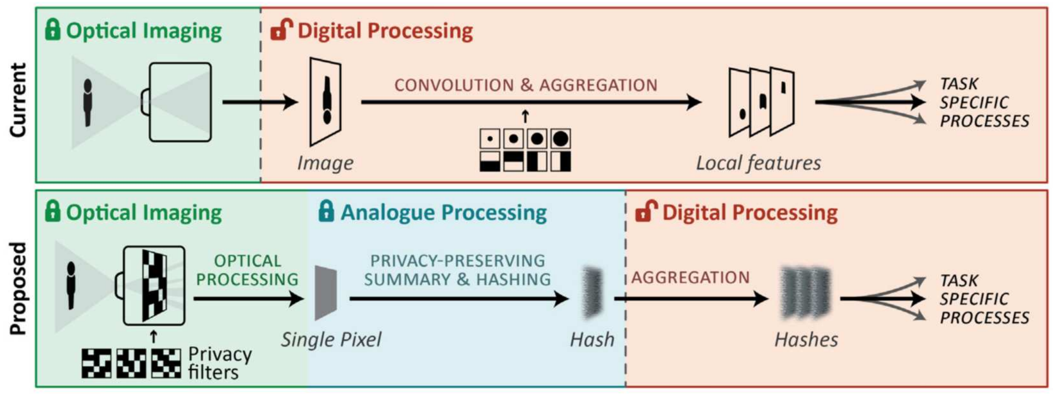


FIGURE 3.3: Figure from [7]. Taras et al.'s proposal of an inherently privacy-preserving architecture. Privacy-preserving image hashing is performed in the analogue domain and only the hashed image is digitised. Therefore an attacker with access to the system can only observe the privacy-preserving hashes which are used for task specific processes. In this thesis the task will be to steer a robot around a scene using only privacy preserving images.

imaging, as the single-pixel approach captures limited information about the scene, often preventing detailed reconstruction without complex algorithms. The tradeoffs include a dependency on computational reconstruction quality and potential noise impact, which may limit resolution and image quality in some contexts. However, by optimizing signal quality and accounting for environmental factors, single-pixel imaging presents a powerful alternative to conventional imaging, particularly in spectral regions where traditional sensors struggle or where privacy concerns are paramount.

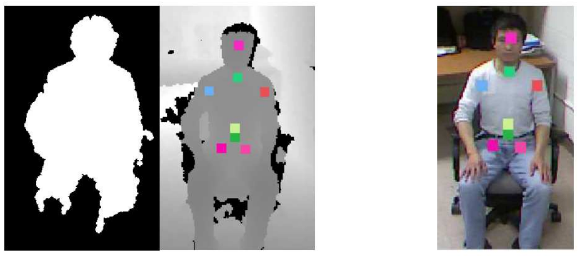


FIGURE 3.4: Figure from [21]. The left side shows a privacy-preserving image taken with an RGBD camera that does not reveal a person's identity.

3.2 Privacy-preserving vision using an RGBD camera

A single-pixel image hashing approach overcomes many limitations of existing privacy preserving approaches. There have been many privacy-preserving robotic vision approaches implemented previously. One such example is by Zhang et al. [21], who use a red-green-blue depth (RGBD) camera to analyse the environment using 3D depth information. Their approach is able to mask out human features to keep them from being identified (Figure 3.4), but human actions can still be understood if the video feed is viewed. A limitation of using this approach is that human actions remain trackable which is a vulnerability in a privacy preserving approach. Our approach will overcome this limitation by using privacy-preserving vision that does not allow for humans or their actions to be tracked in the scene.

3.3 Privacy-preserving line clouds

3D point clouds are often used to map out an environment and enable localisation. Regular 3D point clouds are easy to interpret by humans and reveal sensitive details about the layout of an environment. Speciale et al. [22] propose an approach that lifts the map representation from a 3D point cloud to a 3D line cloud. Their paper focuses on image-based localisation, which is a crucial step in the success of our approach and also has wider applications in robotics [23]–[25].

Although the 3D line cloud method is unintelligible to humans, Chelani et al. [26] successfully demonstrate image reconstruction from the line clouds to a high degree of similarity with the original image. As seen in Figure 3.6, the line cloud approach is not secure for privacy-preserving applications. The obfuscation method of this approach is therefore not suitable to use for privacy-preserving scenarios. Taras et al. [7] overcomes the limitations of this approach, because it removes and scrambles information rather than obfuscating it. Removing information results in more challenging scene reconstruction for attackers.

Furthermore, Pittaluga et al [27] show how multi-channel images can be reconstructed when information about colour and SIFT descriptors are given. The method works with highly sparse and irregular 2D point distributions. The reconstruction is also capable of rendering

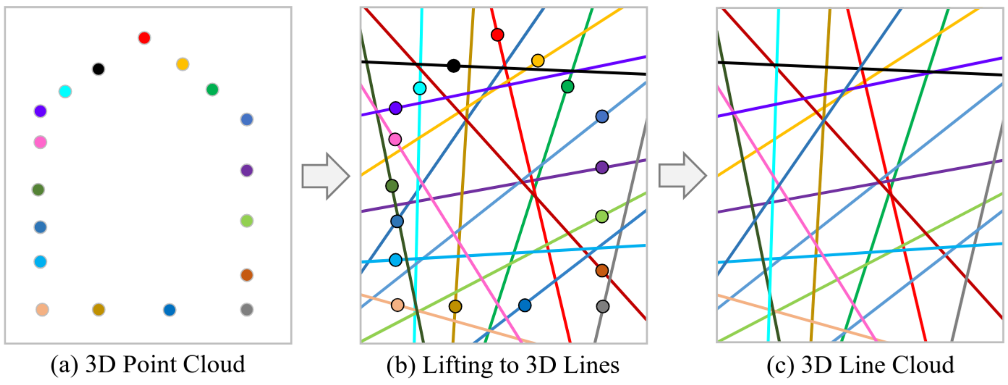


FIGURE 3.5: Figure from [22]. The 3D point cloud is lifted to random 3D lines that pass through individual points in order to obfuscate the geometry of the map and preserve privacy.

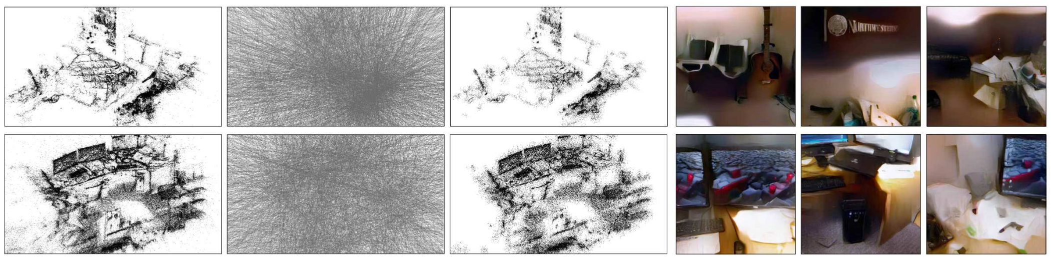


FIGURE 3.6: Figure from [26]. It shows 3D point clouds obfuscated into 3D line clouds using the approach by [22]. Chelani et al. [26] show the underlying 3D point clouds can be recovered and processed through a point cloud-to-image translation which recovers image details.

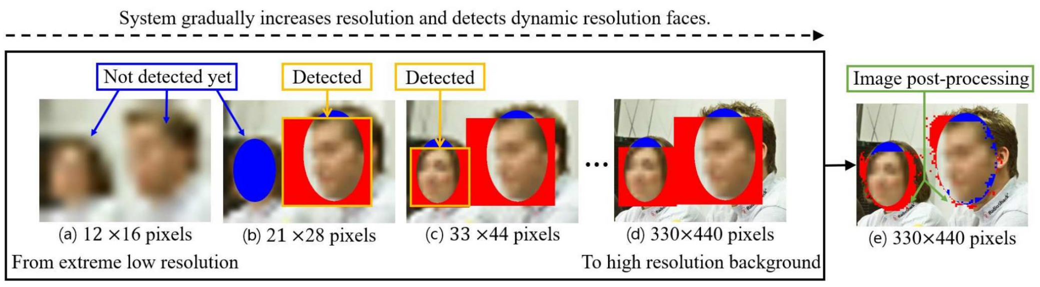


FIGURE 3.7: Figure from [28]. A privacy-preserving pipeline is shown, where the starting image is low resolution and gradually increases in resolution. Detected faces are left at a low resolution.

novel views of the scene, which causes further issues with privacy as a 3D scene holds more privacy revealing information than an image on its own. By using Taras' approach, scene reconstruction from the hashing data alone will not be viable as the method destroys information about the location of features.

3.4 Privacy-preserving vision using blurring

Kim et al. [28] provide an approach to privacy-preserving robotic vision that involves a deep-learning model trained to detect and block human faces. The image is processed to keep faces at a very low 15×15 resolution, whilst the rest of the scene remains at a high resolution. This allows for details in the scene to be processed while keeping humans anonymous. The privacy preservation in this approach does not extend beyond keeping user faces from being identifiable. This is an issue because details in the scene can still be viewed at full resolution, which would not be appropriate if non-human features in the scene must retain privacy. Our approach will preserve the privacy of the entire scene instead of only humans.

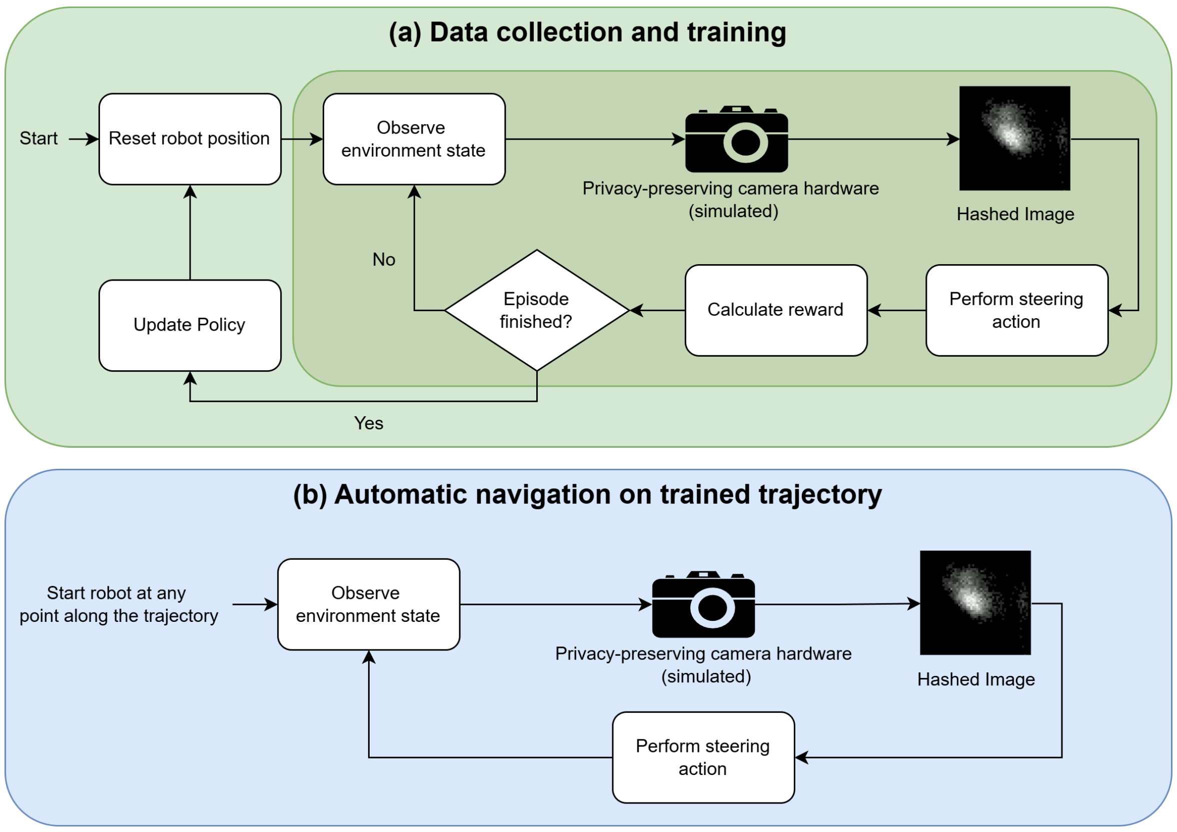
3.5 Locality-sensitive hashing

The work by Shahbazi et al. [29] on loop closure detection demonstrates a practical application of Locality-sensitive hashing (LSH) in Simultaneous Localization and Mapping (SLAM). In

SLAM, loop closure detection ensures that a robot can recognize previously visited locations, a critical aspect of creating accurate maps. Shahbazi's approach uses LSH to efficiently identify similar visual features, enhancing real-time visual SLAM performance with faster matching and higher precision than traditional Bag-of-Words (BOW) methods. Integrating LSH in this manner could similarly enhance our navigation system by enabling efficient, privacy-preserving recognition of scenes within the trajectory.

CHAPTER 4

Methodology



(a) Data collection and training
Start
Reset robot position
Observe environment state
Privacy-preserving camera hardware (simulated)
Hashed Image
No
Episode finished?
Calculate reward
Perform steering action
Update Policy
Yes
(b) Automatic navigation on trained trajectory
Start robot at any point along the trajectory
Observe environment state
Privacy-preserving camera hardware (simulated)
Hashed Image
Perform steering action

FIGURE 4.1: The experiment is split into 2 sections, the data collection and training process (a), and the automatic navigation process after training (b). In the training process (a) we run a reinforcement learning algorithm where the robot explores the scene based on a hashed-image input. An episode completes after a timeout, out of bounds event, or a crash. The navigation policy is updated after episode completion. In (b) we execute the trained model that successfully steers the robot through the entire loop-trajectory without crashing or going out of bounds. The focus of this thesis is on using the hashed images to train a machine learning model, so the privacy-preserving camera is simulated in this setup.

We will create a method of navigating through a scene using the privacy-preserving hashing method proposed by Taras et al [7]. The work by Taras et al. provides a good basis for our approach, however it does not define a method of navigating a robotic platform through a scene. The method we develop involves reinforcement learning-based approach where the robot is rewarded for staying as close as possible to a closed-loop trajectory. The trajectory for a given scene is predefined and the reward function requires the robot's position to determine deviation from the trajectory. The testing process will only take a privacy-preserving hash for input to determine all navigation commands.

Incorporating locality-sensitive hashing (LSH) into this framework could help address the requirement that "small changes between images result in small changes between the resulting hashes." Unlike standard cryptographic hashing methods, such as SHA-2, which produce outputs with high sensitivity to any input variation (known as the avalanche effect), LSH is designed to map similar inputs to similar hashes. This characteristic makes it a promising option for privacy-preserving navigation, where continuity between frames in a trajectory needs to be preserved.

Locality-Sensitive Hashing divides data into buckets such that items within each bucket are more likely to be similar. For navigation, this means that small changes in the robot's environment, such as slight positional shifts, would produce hashes close to those of previous frames. This continuity is essential for a reinforcement learning-based navigation system that relies on smooth transitions and predictable visual information for accurate trajectory tracking.

Several criteria that need to be fulfilled by the privacy-preserving hashing function to enable secure navigation:

(1) Small changes between images results in small changes between the resulting hashes

(2) The hash should destroy as much information as possible, but leave enough information to differentiate between points in a trajectory

This restricts our ability to use standard encryption methods such as the Secure Hash Algorithm 2 (SHA-2). These hashing methods exhibit an avalanche effect, where a small change in the input leads to a completely different output. This is not suitable for our CNN approach which relies on associating patterns to similarly looking values. In the following subsection, we justify a possible approach that meets the criteria above.

4.1 Privacy-preserving hashing concept

Taras et al. [7] outline an approach that is possible with a single-pixel camera architecture. By designing an approach that is compatible with analogue processing, we minimise the risk of an attack on that part of the system. We only digitise the image after computing the hash in the analogue domain. Therefore an attacker will only see the hashed image, requiring a brute force attack that does not guarantee a privacy breach.

The first method proposed by Taras et al. is a random lines approach. The method involves measuring pixel intensity, so we read the image as a single channel monochrome. Assuming 8 bits of precision, there are 256 possible intensity values for any given pixel. We construct our hash as a 256×256 zeros matrix with one axis representing maximum intensity and the other axis representing minimum intensity. We then measure the minimum, i, and maximum, j, pixel intensity along a random, straight line and increment the matrix element at (i, j). This is repeated for N lines. Given a large enough N, we can use this fingerprint to identify an image from a sequence of images. A key feature of this approach is the detection of edges along lines. The global structure of the image can be discerned but the locations of the edges remain unknown as that information is removed in the hashing process.

One characteristic of Taras et al.'s approach is the rotational invariance of the captured data. The statistical aggregation is not affected by the rotation of the original image frame as no pixel location data is captured. In specific situations, this will contribute to erroneous classifications by the reinforcement learning model. For instance, if we have two separate locations in an environment that appear as visual mirror images of each other, the hashing method proposed by Taras et al. will classify them as the same image since the hashes will be identical. To mitigate this issue, we will also test hashes that are not rotationally invariant by means such as breaking up the original image into multiple regions or using multiple feature extractors.

4.2 Privacy-preserving hash functions

We have designed several privacy-preserving hashes to use. The extent of privacy on these hashes vary based on the amount of information they give away. We have primarily chosen patch and line hashes because they can utilise the performant matrix operations in NumPy.

All hashes are based on the minimum and maximum pixel values of features on an 8-bit monochrome image. For a standard 8-bit monochrome image, the resulting hash would be 256×256. A downside to very high precision in the hash output is the susceptibility to noise. The resulting hash is more grainy which can cause issues with the CNN feature extractor. One way to mitigate this issue is to 'bin' the hash so pixels within a certain range are treated as the same value. For example, with a bin size of 4 the pixel values 0–3 are treated as 0, 4–7 are treated as 1 etc.. An example hash function is shown in Figure 4.2. This results in a 64×64 hash. An example of applying various bin sizes to the same underlying image is shown in Figure 4.4. No bins results in a noisy hash that can be hard to learn features from. A bin size that is too high will result in a low resolution hash that loses a lot of distinguishing information. Therefore a middle-ground value should be used for the bins.

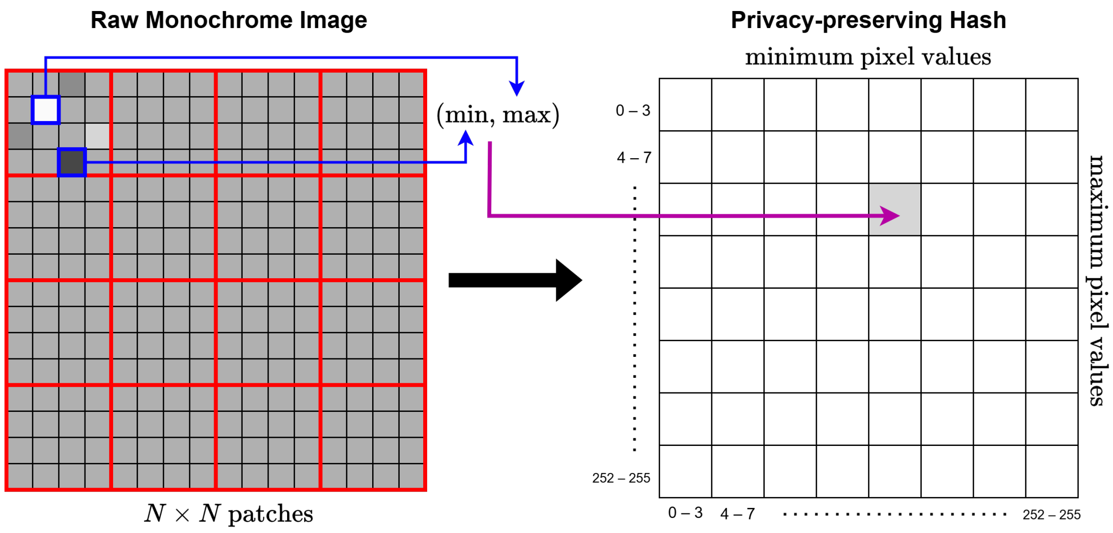


FIGURE 4.2: Diagram showing the hashing process. In this example we take the minimum and maximum in each patch of the original image and map it to the hash.

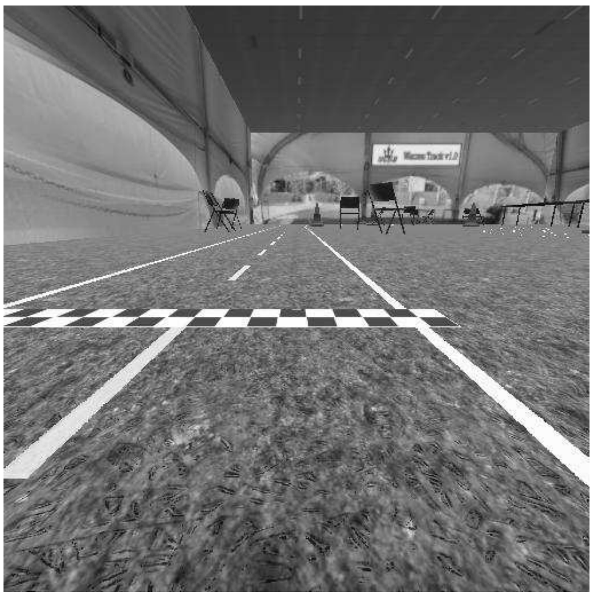


FIGURE 4.3: Example greyscale image of the environment. All hashes in this section will be based off this image.

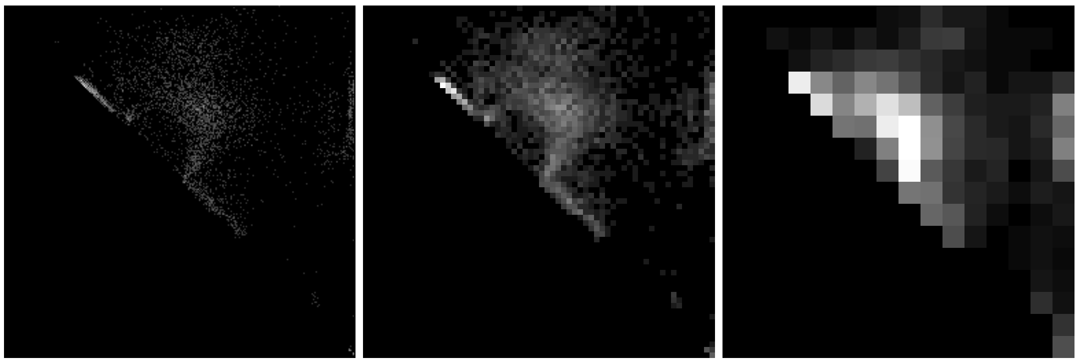


FIGURE 4.4: The diagram depicts hashes with varying bin sizes. (a) is a 256×256 hash with no bins, (b) is a 64×64 hash with a bin size of 4, and (c) is a 16×16 hash with a bin size of 16.

4.2.1 Patch hash

The patch hash involves sampling every $k \times k$ patch on an $N \times N$ image such that no patches overlap and all pixels in the image are covered. The minimum and maximum pixel value in every patch is measured for the hash. An example of a patch hash is shown in Figure 4.2.

4.2.2 Line hash

The line hash is similar to the patch hash but instead samples every $k \times 1$ horizontal or vertical line on an $N \times N$ image.

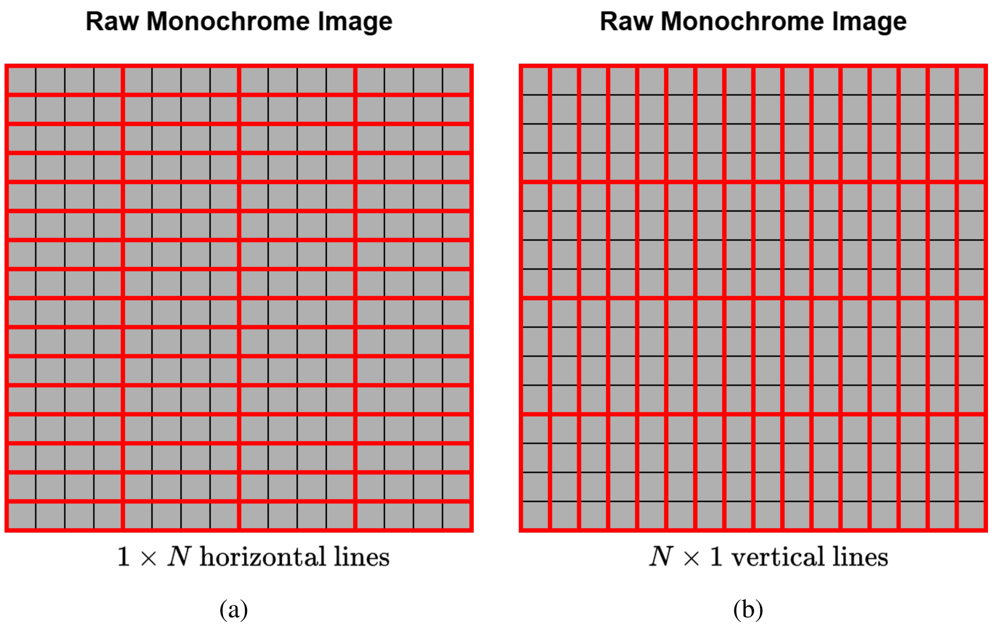


FIGURE 4.5: Diagram visualising the line hashing method. (a) shows a horizontal line hash and (b) shows a vertical line hash.

4.2.3 Multi-channel line hash

The multi-channel line hash builds upon the regular line hash by sampling both horizontal and vertical lines. The resulting hash has 2 channels, for each of the horizontal and vertical hashes. The motivation behind combining the horizontal and vertical hashes into a single hash is to provide the underlying CNN feature extractor with more features to differentiate frames by. For example, the vertical line hash can provide more information about horizontal edges in the raw image, while the horizontal line hash provides information about the vertical edges in the raw image.

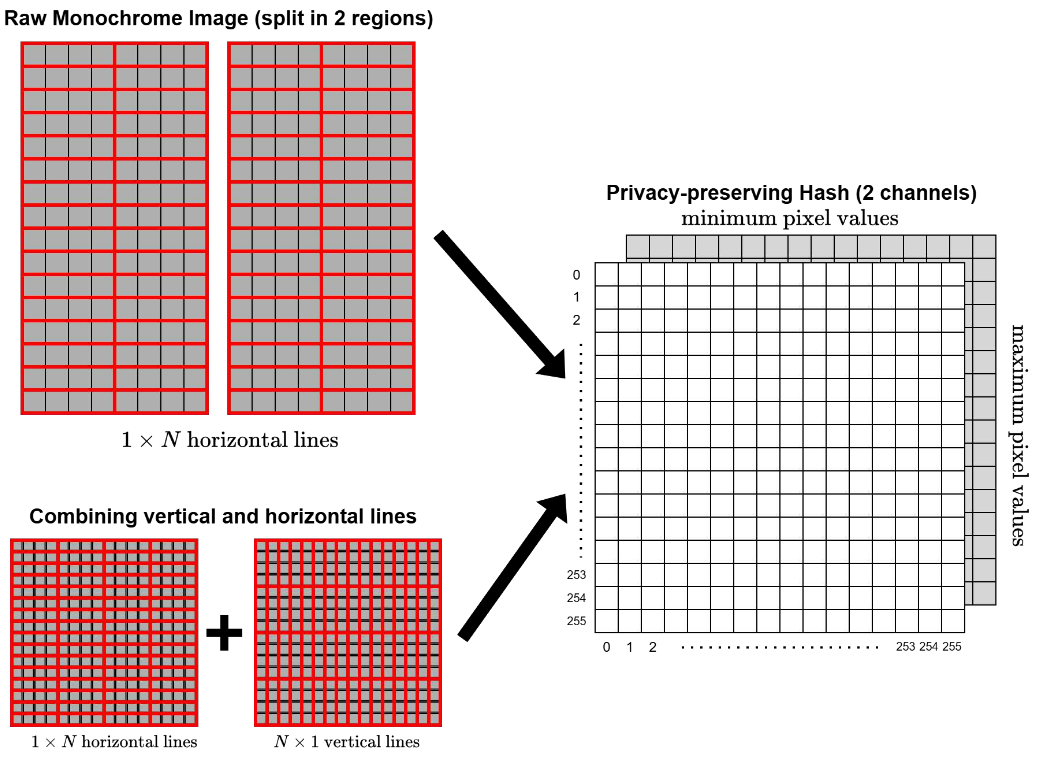


FIGURE 4.6: The diagram shows 2 examples of doing a multi-channel line hash. The first method splits the original image in 2 halves and performs the line hash on each half. The second approach performs horizontal and vertical line hashing on the entire image. Each hash feature is represented by its own channel in the resulting hash.

4.2.4 Region-based line hash

The region-based line hash also builds upon the regular line hash method. The line hash method results in a 'global' representation of the frame. If we have a scene with some object in the top left corner that we move to the bottom right corner, the hash is likely to remain very similar. Although this effectively maintains privacy by hiding an object's exact location, the loss of this information provides a challenge in a navigational scenario. For example, a robot operating with regular camera vision may use the location of a wall feature to direct its steering. A left wall and a right wall may appear the same on our line hash which results in ambiguity if those scenarios require a different steering command. Therefore we can break up the image into multiple regions and run the hashing method on each region separately. The result is combined into a multi-channel hash where each channel represents a different region of the image. We note that increasing the number of regions the image is broken up into will lead to a higher amount of local information being revealed. Thus we limit the regions to 2 (left and right half) and 4 (top-left, top-right, bottom-left and bottom-right quadrants). In addition, this approach can also be combined with both horizontal and vertical lines to gain greater detail for the agent to learn.

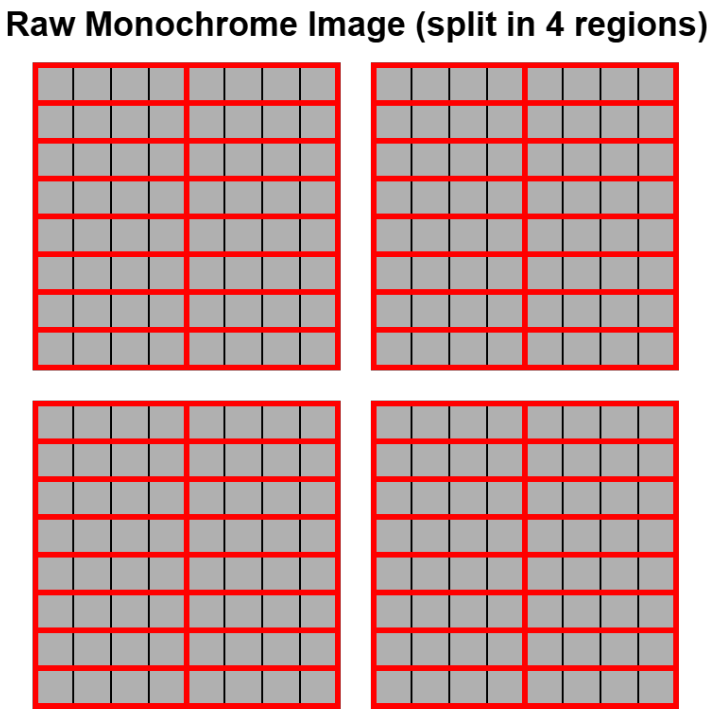


FIGURE 4.7: The diagram shows one example of a region based line hash. The original image is broken into 4 quadrants, and the horizontal line hashing algorithm is applied to each separately. This will result in a 4 channel hash.

4.3 PPO model training

We chose Proximal Policy Optimization (PPO) as the reinforcement learning (RL) algorithm for training the robot to follow a desired trajectory. PPO is ideal for this task due to its balance of robustness and efficiency in environments that require safe, stable learning. It optimizes the policy (the agent's decision-making strategy) in a controlled way, avoiding drastic changes that could lead to erratic or unsafe actions. This is especially important in continuous control tasks like trajectory following.

PPO's main advantage is its use of a "proximal" objective function. This function restricts updates to the policy, preventing large, sudden changes in learned actions. PPO achieves this by clipping the probability ratios of actions between new and old policies, keeping each step within a set threshold. This stabilizing effect makes it well-suited for teaching the robot to stay on a smooth trajectory. By discouraging abrupt deviations, PPO helps ensure that the robot stays on course without losing control. The gradual updates allow the agent to learn complex behaviors safely, which supports our goals of trajectory alignment and obstacle avoidance.

Other reinforcement learning approaches for trajectory following include Deep Deterministic Policy Gradient (DDPG) and Twin Delayed Deep Deterministic Policy Gradient (TD3). Both work well for continuous control, though TD3 is more stable. Soft Actor-Critic (SAC) promotes exploration through entropy maximization, useful in dynamic environments. Trust Region Policy Optimization (TRPO) provides stability by limiting large policy updates but is more computationally demanding. Each method has distinct strengths depending on the specific requirements for stability, exploration, or efficiency in navigation.

4.4 CNN model training

We have chosen to use a Convolutional neural network (CNN) architecture for training our model. The key advantages of CNN networks include:

(1) Effective for image classification. Our network will involve inputting and image and output a corresponding steering value associated with the robot state in that frame.
(2) Automatic feature detection, eliminating the need for manual feature extraction.
(3) Able to process large amounts of data to produce highly accurate predictions.

As the robot platform is capable of continuously variable steering angles rather than being limited to a set of predefined, discrete angles, a regression-based CNN model is well-suited for this application. Unlike a classification model, which would predict one of a limited set of categories, a regression model outputs a continuous value, allowing for nuanced control over the robot's steering. This continuous output is essential for smooth, precise adjustments that adhere to the closed-loop trajectory. The regression model can provide any steering angle within the robot's physical constraints, maximizing flexibility and accuracy in path-following. This approach enables the CNN to learn subtle variations in steering, leading to more refined navigation that responds fluidly to changes in the scene while following the predefined path.

4.5 Evaluating success

There are several ways to verify the performance of the proposed method.

A high level verification involves evaluating the entire end-to-end process. More specifically we observe:

(1) The robot successfully completes at least 1 lap of the trajectory, starting from any point along the trajectory.
(2) The robot stays within a maximum distance of the intended trajectory at all times.
(3) The robot does not collide with any objects in the scene.

We can also compare the end-to-end result against a model that uses regular images instead of hashed images. Our goal is to achieve a performance that is just as good as a model using regular camera images, demonstrating the viability of our approach.

The reinforcement learning pipeline can be evaluated by graphing the mean reward and mean episode length over the training session. We expect the mean reward to increase over the training session because the robot improves its ability to steer on the track before crashing or going off course. Qualitatively, we can plot the training trajectories of the robot for each hashing method and observe how far into the track it explores. This should approximately correlate with the mean episode length because a higher mean episode length indicates the robot spend more time on the track (the speed parameters for all training instances will remain the same).

CHAPTER 5

Results

5.1 Defining the experiment scope and requirements

We define a scope containing the intended functionality, assumptions and constraints posed on the simulation. Performing the experiments in a simulation allows for repeatability and precise control of robot and environment parameters. In this thesis, we specify some hardware and software constraints on the robot platform to validate the underlying principles of our approach. The future work section explores avenues for expanding the capabilities of this system.

5.1.1 Robot platform

The robot platform involves the specific hardware and software capabilities of the chosen robot.

(1) The robot platform is able to steer left, right and forwards at any angular velocity within the steering range.
(2) The robot is able to move forward at any velocity between the zero velocity and maximum velocity.
(3) The robot has a single forward facing camera capable of capturing data up to 15 frames per second.

(4) The robot trains and stores a reinforcement learning model locally, which can be reused for future runs.

(5) The camera feed will be post-processed digitally into the image hashes. A complete pipeline would involve the analogue processing system discussed by Taras et al. [7] to prevent digital attacks. This thesis will focus on the training and navigation pipeline and assume the digital hashing process can be swapped out by an analogue privacy-preserving camera without affecting the functionality.

5.1.2 Environment

We define assumptions about the environment the robot will operate in.

(1) The environment has sufficient visual texture for navigation to operate successfully.
(2) The environment is on a flat plane (e.g. the floor of a house).
(3) All objects in the environment (excluding the robot platform) remain static.
(4) The environment should have sufficient lighting to enable visual navigation.

5.1.3 Intended functionality

We define the intended functionality of the robot in terms of training, navigation and points of interest.

(1) The robot shall be trained to steer through a specific trajectory in a scene.
(2) A reinforcement learning algorithm is used to train the robot to follow the trajectory.
(3) The trajectory shall form a closed loop and will not contain any overlapping paths.
(4) The robot shall be able to start at any point on the trained trajectory and successfully follow it without collisions.
(5) The robot should be on the trained trajectory at all times, deviating no further than 0.2m from the intended path.

5.2 Simulation environment setup

We perform the experiments in a simulated environment. A simulated environment allows for repeatability of experiments because all constraints can be precisely controlled. Donkey Simulator is chosen as the software that will serve as the environment for our reinforcement learning pipeline. The simulator is able to fulfil the methodology requirements and provides detailed data. There is a forward facing camera with adjustable resolution and field of view. This will serve as the initial observation that will pass through our privacy-preserving hashing algorithm before it enters the training step. Settings such as range of forward and angular velocities can be adjusted and there are a variety of indoor maps to choose from.

We will use the Stable-Baseline3 python library which provides a set of reinforcement learning implementations compatible with PyTorch. We are required to provide information such as the observation space, action space, how to reset the environment and how to process observations. Successful demonstrations of privacy-preserving navigation in the simulated environment provide a good basis for hardware implementation of the system, which will provide more challenges for lighting setup and repetition of training episodes.

5.3 Reinforcement learning setup

5.3.1 Environment

As seen in Figure 5.1, we have a closed-loop track in which all obstacles are static. The robot platform is the only object that moves around the scene.

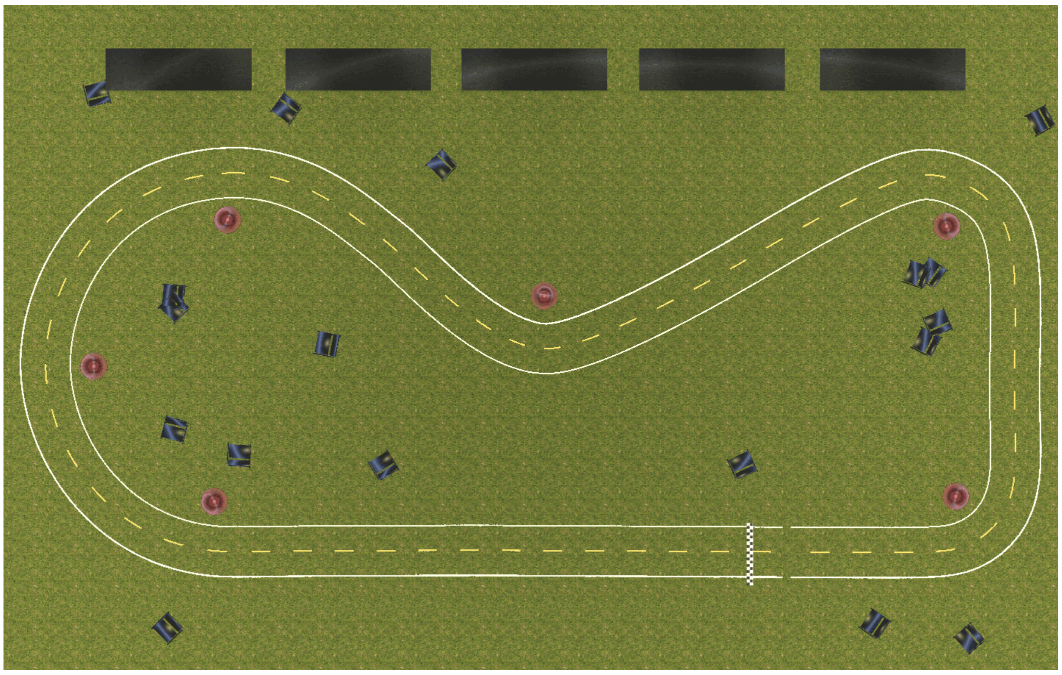


FIGURE 5.1: Top-down perspective of the scene. Note the track is a closed-loop circuit with various obstacles.

5.3.2 Agent

The agent has a single, forward facing camera that captures images at 15 frames per second. The input to the reinforcement learning agent is a privacy-preserving image. As this thesis focuses on the navigational problem with privacy-preserving vision, a regular camera is used to capture an image of the scene which is then passed through the hash function to simulate an analogue-based hashing circuit. The reinforcement learning pipeline only processes the hashed image and does not use any additional information from the raw camera image.

The agent is a 4-wheeled robot with Ackerman steering. The control algorithm is able to command the steering and speed of the robot. We represent this as an action vector

$$\mathbf{a} = \begin{bmatrix} s \\ v \end{bmatrix}, \tag{5.1}$$

where the values are normalised between the range:

$$s \in [-1, 1] \text{ represents the steering.} \tag{5.2}$$

$$v \in [0, 1] \text{ represents the speed.} \tag{5.3}$$

-1 is the sharpest left turn and 1 is the sharpest right turn the robot can take. The robot is stationary at 0 and maximum forward speed at 1. The steering and speed magnitude can be reduced for easier control.

5.3.3 Reward

The aim of our reward function is to keep the robot on the desired trajectory. We penalise the robot for going off trajectory and crashing into obstacles.

Let:

- e_{ct} be the cross-track error (the distance of the car from the centre of the lane).
- $e_{\text{max ct}}$ be the maximum allowable cross-track error before failure.
- v_{forward} be the forward velocity of the car.
- hit be a condition representing whether the car has collided with an obstacle or not.
- done be a flag indicating the end of an episode (either the car finishes or fails).

$$R = \begin{cases} -1.0 & \text{if done} = \text{True} \\ -1.0 & \text{if } |e_{\text{ct}}| > e_{\text{max ct}} \\ -2.0 & \text{if hit} \neq \text{none} \\ (1.0 - \frac{|e_{\text{ct}}|}{e_{\text{max ct}}}) \cdot v_{\text{forward}} & \text{if } v_{\text{forward}} > 0.0 \\ v_{\text{forward}} & \text{otherwise} \end{cases} \tag{5.4}$$

5.4 Hash visualisations

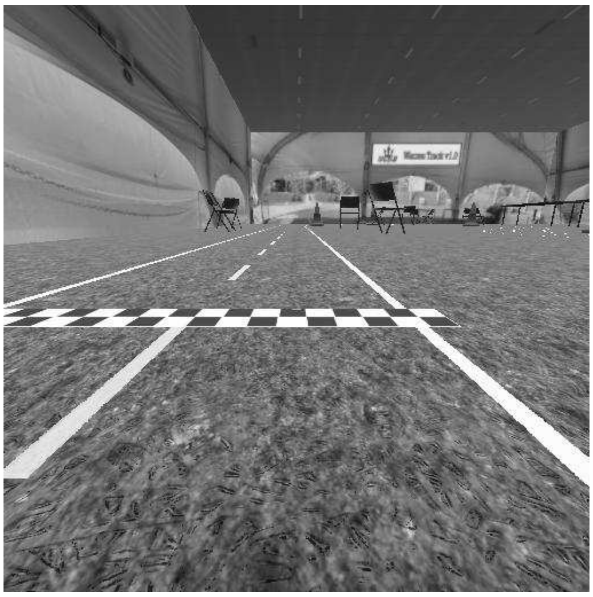
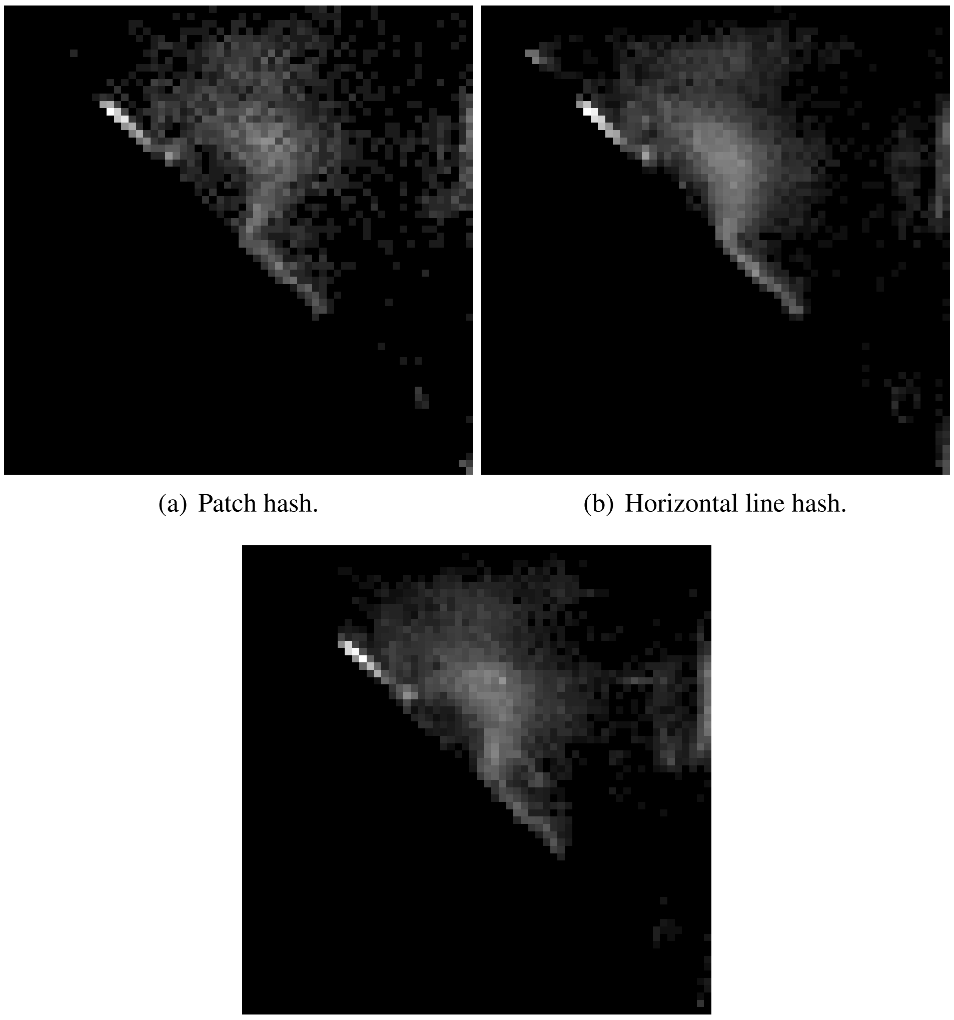


FIGURE 5.2: Example greyscale image of the environment. All hashes in this section will be based off this image.

The figures below show visualizations of the different hashes, transformed from the reference image in Figure 5.2. Each hash is displayed as an 8-bit monochrome image, though cell values in the hash may exceed 255 in the reinforcement learning model. To improve clarity, a gamma correction filter highlights the darker regions. Hashes with multiple channels are arranged in a 2D grid for easier viewing. Note that the bottom left triangle of each hash channel is black. This is due to the hashing method: within each patch or line, the minimum pixel value cannot be greater than the maximum.

We expect the multi-channel hashes to perform better than single channel versions by virtue of the higher amount of information for the reinforcement learning algorithm to train from. For example, if we use the region based method and split the image into left and right halves, then an object which goes from the left half of the image into the right half will be detected. If the same scenario involved a single channel hash, an object going from the left half of the frame to the right half of the frame would not be easily detected. This is because our hash functions destroy all location revealing data of features within the image.

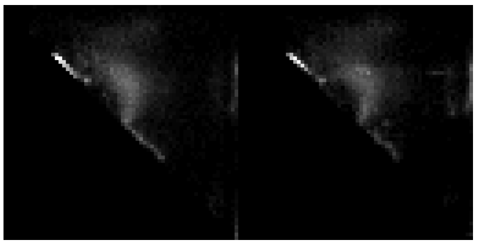


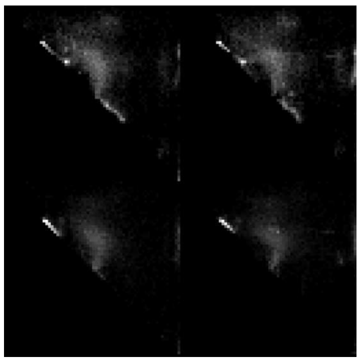
(a) Patch hash.

(b) Horizontal line hash.

(c) Vertical line hash.

FIGURE 5.3: Visualisation of single-channel hashes. Each hash function results in a relatively similar but unique hash. By comparing the patch hash in (a) with the line-based hashes in (b) and (c), we observe the greater amount of noise in the patch hash. The line based hashes contain more feature samples due to the lower area of each line, which results in a smoother gradient over the hash.


(a) Horizontal and vertical line hash.


(b) Horizontal and vertical line hash (2 regions, left and right).

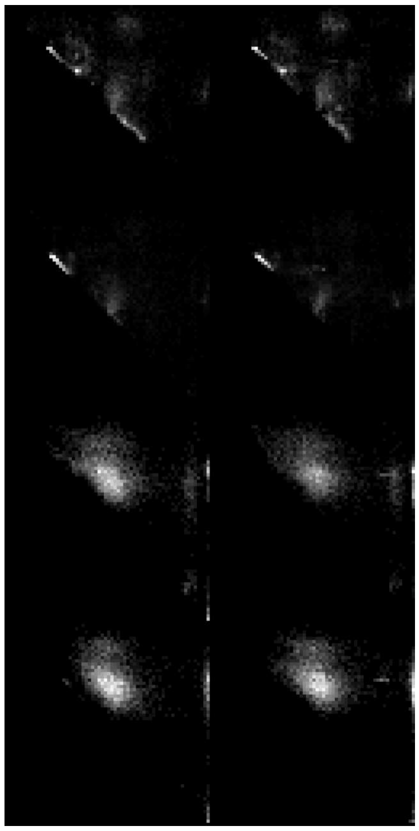

(c) Horizontal and vertical line hash (4 regions).

FIGURE 5.4: Visualisation of multi-channel hashes. Each channel in the hash represents a different feature or region of the original image. A multi-channel hash provides greater detail as the original image is aggregated in multiple ways. In (a) for example, the horizontal line feature on the left will be more sensitive to vertical edges in the scene and the vertical line feature will be more sensitive to horizontal edges. (b) and (c) split the original image into multiple regions with the aim of increasing the information provided to the reinforcement learning algorithm. Notably in (c), the top 4 channels and bottom 4 channels are significantly different because the bottom two quadrants are taken up by the floor (which has a more uniform texture) and the top two quadrants show finer details from the rest of the environment.

5.5 Experimental results

We run the various hashing methods outlined in the methodology on the same track to compare effectiveness. We also perform a run without any hashing technique, using a regular 256×256 monochrome image. This provides a point of reference when comparing the performance of the privacy versus non-privacy navigation.

For the experiment we train each hashing method for a total of 75000 time steps where each policy update is after 2048 steps. In each step a new image frame is captured and the observation and reward is calculated. The reinforcement learning policy updates after a minimum of 2048 frames have passed. The benefit in updating our policy after a fixed time horizon is to allow for more stable and efficient learning. By collecting data from many episodes, proximal policy optimisation learns from a more diverse set of states and actions. Some episodes can be very short if the robot crashes of goes off track early, leading to a slower convergence. This is less likely to occur with mini-batch updates.

Two quantitative metrics we use to evaluate the reinforcement learning model is the mean episode length (Figure 5.5) and mean episode reward (Figure 5.6). In our scenario, both metrics are correlated because the further the robot travels the higher the reward. The episode length may be inflated with the car travelling in a wave-like motion. However the effect of this is negligible as our track width is narrow compared to the total length of the loop.

As shown in Figure 5.6, the privacy-preserving images did not perform as well as the unaltered camera images. Despite this, all tested hashes show a positive trend, indicating that the robot is gradually learning the track. The difference in reward gain rate can be attributed to the level of information available to the reinforcement learning algorithm. A full monochrome image provides significantly more detail, such as shape, texture, and spatial information, which aids learning.

A key trade-off with privacy-preserving hashes is the loss of locational data about the scene. The simulated track setup includes distinct lines that the robot needs to follow. A CNN feature extractor can efficiently interpret these details in the unaltered images, but this becomes more challenging with hashed representations. For instance, a line indicating an upcoming right turn is much easier to interpret in a high-detail image than in a privacy-preserving hash.

In Figure 5.6, the top-performing hashes were the single-channel horizontal line hash and a multi-channel hash incorporating multiple features. While multi-channel hashes were expected to perform best due to the higher volume of information, the privacy-preserving hashes continue to show a positive learning trend toward the end of training. Therefore, in subsection 5.5.3, we extend the training times for the top two hashes to test if this trend persists.

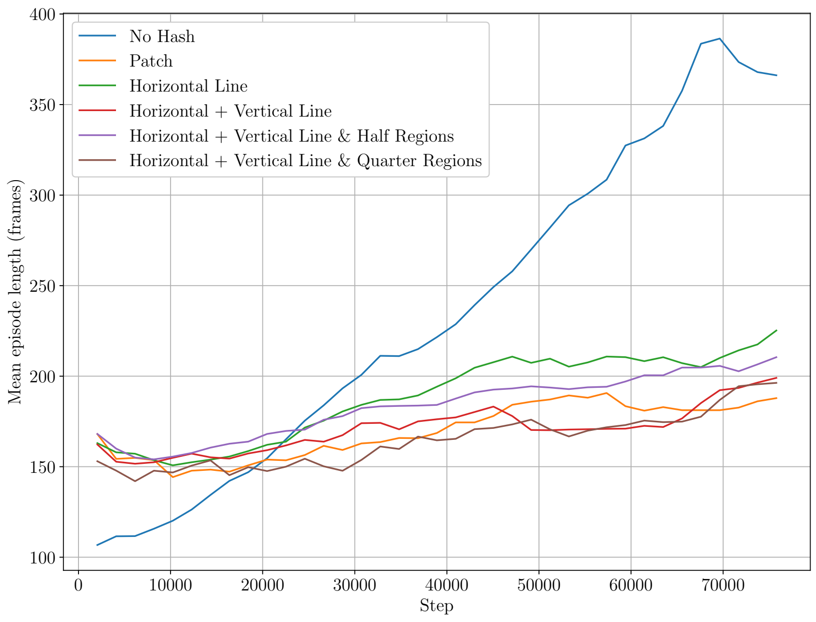


FIGURE 5.5: The mean episode length over the whole training session. The positive gradient indicates the robot goes further in the track as the environment is learnt.

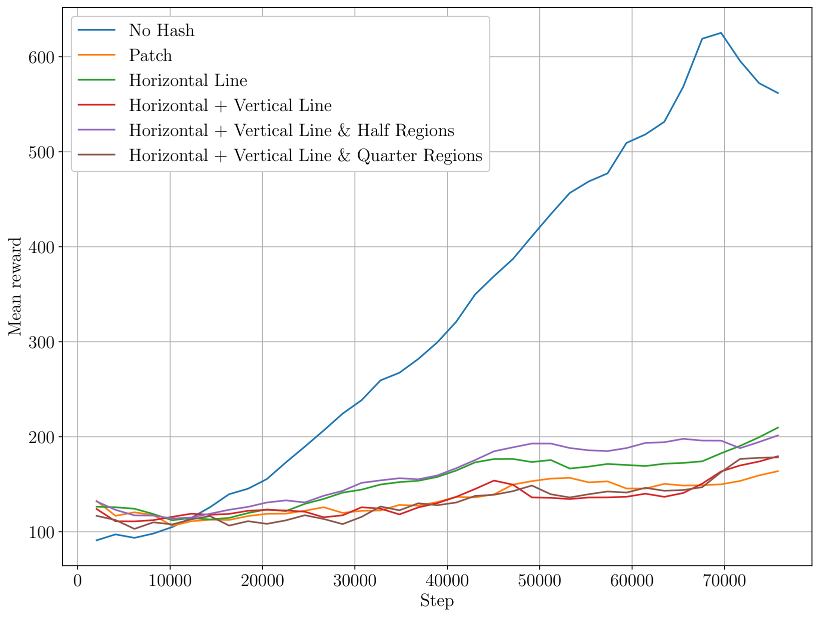
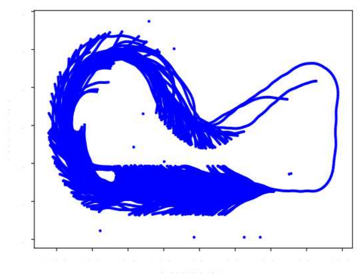


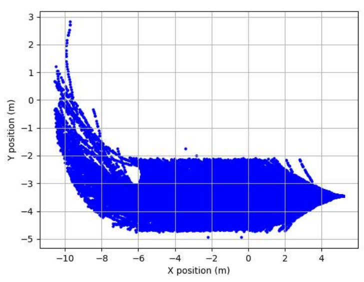
FIGURE 5.6: The mean episode reward over the whole training session. The positive gradient indicates the robot learns to maximise the reward function the longer it spends training.

5.5.1 Robot training trajectories

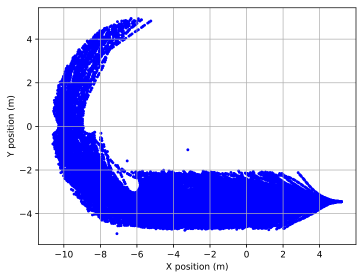
The following set of figures show the environment space explored by the robot using variations of the privacy-preserving hash.



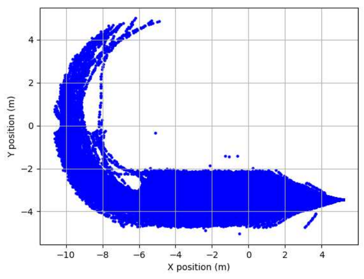
(a) No hash.



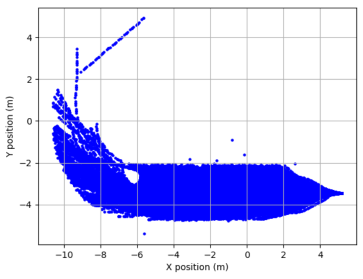
(b) Patch.



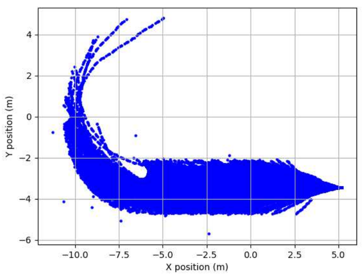
(c) Horizontal line.



(d) Vertical line.



(e) Horizontal and vertical line.

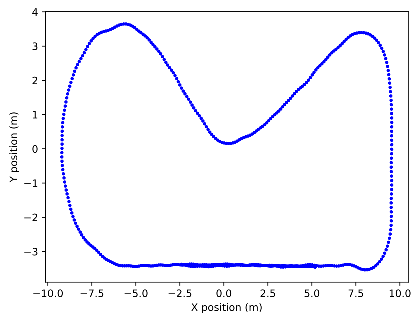


(f) Horizontal line with two regions.

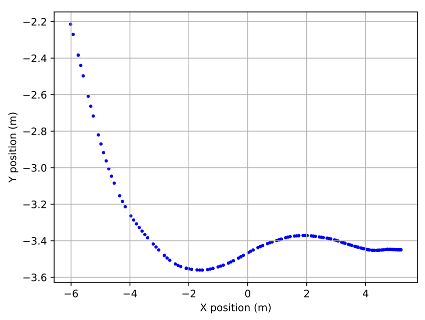
FIGURE 5.7: Robot trajectories with various hashes during the training process. The robot successfully learns the entire track using regular camera vision, as shown in (a). In contrast, the hashing methods could not complete the full loop, with the best performance at 75,000 timesteps achieved by the horizontal line hash, depicted in (c).

5.5.2 Robot validation trajectories

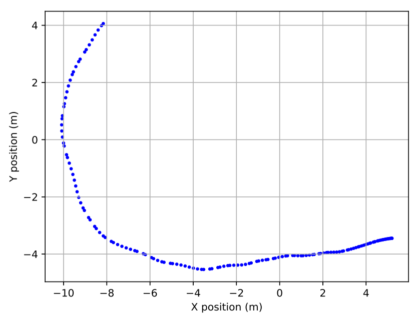
The following set of figures show the trajectory of the robot using the fully trained model for each hash. The run terminates when the robot completes the track or goes off course.



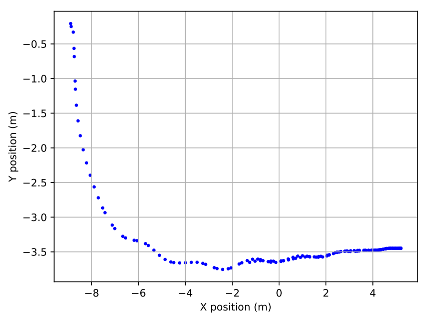
(a) No hash.



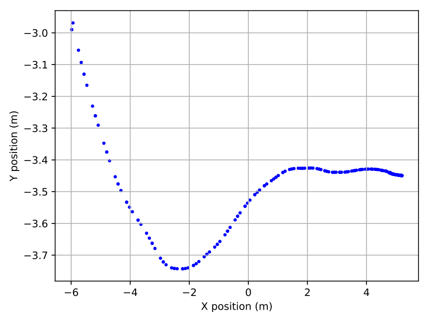
(b) Patch.



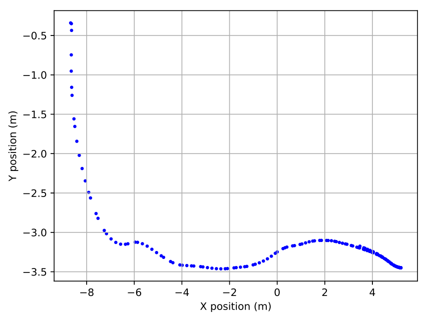
(c) Horizontal line.



(d) Vertical line.



(e) Horizontal and vertical line.



(f) Horizontal line with two regions.

FIGURE 5.8: Robot trajectories with different hashes after training the reinforcement learning model are shown. The full closed-loop track was completed only with the non-hashing approach, as seen in (a). The horizontal line hash in (c) made it the furthest along the track before deviating off course.

5.5.3 Extended training time for highest performing hashes

The positive gradient in Figure 5.6 indicates the hashing methods are still learning the environment when the training session terminates. Therefore, we run the top two privacy-preserving hashes for a 150,000 step training session and observe the behaviour. Observe in Figure 5.10, the positive trend after 75,000 time steps does not continue. The multi-channel hash model was able to explore a larger space in this run (Figure 5.9) than all the previous results with a lower training time.

We attribute the following issues with the current design that produced these results:

- The hash functions destroy too much detail in the pursuit of maximising privacy, which poses a challenge for the reinforcement algorithm to associate hashes with steering directions.
- There is not enough distinguishing information in the scene for various points in the trajectory. For instance one point at the start of the track may have an almost identical hash to another point halfway through the track. However the steering command for both are different which can cause conflict when trying to learn the steering pattern for both points in the trajectory. Two ways to address this issue is to increase the details extracted in the hash, or increase the texture variation within the simulation environment.

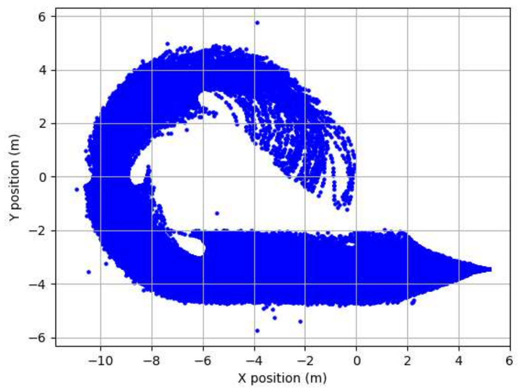


FIGURE 5.9: Trajectory of multi-channel hash with extended training time. The space explored is a noticeable improvement to the robot trajectory hashes in 5.5.1.

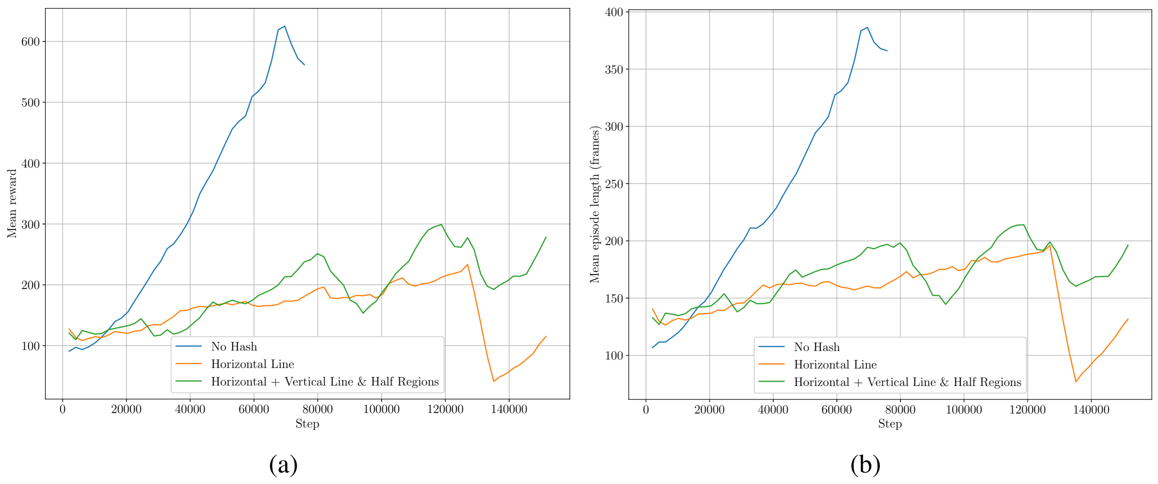


FIGURE 5.10: The mean episode reward, (a), and mean episode length, (b) for an extended training session. Note the significant dips after timestep 80,000 for both the hashes. This indicates the model is overtraining and loses its ability to effectively navigate the start of the track.

CHAPTER 6

Discussion

6.1 Feasibility

The robot was unable to navigate the entire track using privacy-preserving vision. However, we still manage to achieve partial success and show the reinforcement learning model is capable of learning using only privacy-preserving images. The efficiency in learning the environment is less than regular greyscale images. This result is to be expected as regular images contain significantly more informative details to learn from. In order to maintain privacy, our approach destroys local geometric information which reduces the mean reward gain.

A variety of handcrafted hash functions were tested and compared. The underlying principles of each hash remained the same, with all feature types being mapped to minimum and maximum pixel intensities. Hashes with more features were expected to perform better. This is due to the larger number of features allowing the underlying CNN to recognise more patterns to associate each frame to a steering direction. However, we saw in Figure 5.6, the single channel hash using horizontal line features performed as well as the multi channel hashes. The horizontal line feature is more sensitive to changes in vertical edges. The inherent motion of our robot is to perform left and right steering commands which results in the camera also turning left and right. There is a greater rate of change in vertical edges crossing the horizontal patch boundaries, which leads to a more noticeable transition between frames as the robot is turning. This also explains why using only the vertical line hash had worse performance than the horizontal line hash.

Extending the training over a larger volume of runs resulted in both the single channel and multi-channel hashes to drop in performance. The likely cause of this issue is due to overtraining the model with similar hashes originating from different sections of the track. The chosen simulation environment has a large amount of uniformity, which can work against our chosen feature extractor which summarises frames without taking object locations within the frame into account. The experimental setup can benefit from using a multitude of coloured objects around the scene. This increases the likelihood of each frame in the scene resulting in a unique hash, preventing ambiguity in the machine learning model.

We have ensured the hashing functions that have been tested are compatible with the inherently privacy-preserving architecture proposed by Taras et al. [7]. The hardware implementation relies on lower complexity image operations such as scanning minimum and maximum sensor values. Since our experiment was executed entirely in a simulated environment, the privacy-preserving hardware was also simulated. The privacy-preserving camera can be treated as a black-box in our setup, which means a real hardware implementation of the approach can be substituted for the simulated privacy-preserving camera.

An earlier approach tested in this thesis involved manually generating the training data. The robot would be set to a data collection mode and teleoperated from a starting point and closely follow an intended trajectory. This method only required a CNN for training the resulting image dataset against the steering angles of the robot. However, a limitation we found with such an approach was the feasibility in generating a large volume of data required for a trajectory. It is not enough to consider all the positive cases where the robot is strictly following the planned trajectory. An incorrect steering command for a single frame would lead to the robot going off course without the knowledge to correct itself. Therefore, a reinforcement learning approach was deemed more viable because the robot will automatically generate the training data and explore the state space. This training approach inherently accounts for off policy scenarios where the robot crashes and repeats such scenarios over a large volume of runs to optimise the reward.

Overall, we have shown it is indeed possible to partially navigate a robot through a scene using only privacy-preserving vision. The following sections will address future work and suggestions to improve the current method and allow for successfully navigating an entire track.

6.2 Comparison with other privacy-preserving vision approaches

When comparing our approach (built upon the proposal of Taras et al. [7]) against other approaches discussed in the literature review, we have the most secure privacy-preserving techniques.

One method in the literature was to start with a low resolution image and gradually increase the image resolution. When a face is detected by a deep-learning algorithm, that section of the image remains low resolution while the rest of the scene becomes high resolution. This approach is practical for scenarios such as surveillance cameras where it may be desirable to hide the identities of people, but still be informed of any events occurring in the scene. However, a robot traversing an environment will not necessarily require this level of information to perform its tasks. As demonstrated in our thesis, a robot can be trained to steer through an environment using privacy-preserving cues that destroy revealing geometric information about the scene.

Another method discussed in the literature also utilises aspects of a hardware based approach. Using only the depth channel of an RGB-Depth camera can inherently mask the identities of individuals. The paper by Zhang et al. [21] applies the technique on an automatic fall detection scenario for elderly people. This scenario requires the detection of a human in the frame and associating a falling motion with a detection action. Using a depth channel for training a privacy-preserving robot to navigate a scene is viable as key geometric information would be preserved. However, the level of privacy-preservation is not the most secure compared to the hashing approach in this thesis. Critical details and movement of humans in the scene can still be exposed with the use of an RGB-Depth camera.

One direction that can be taken towards maintaining user or environment privacy is to focus on increasing security of the systems themselves. This approach would be to prevent an attacker from gaining access to the system or data. Although state of the art systems can be designed in this regard, there will always be newer vulnerabilities discovered. For example, a robotic vacuum system can collect and stores sensitive sensor data in a cloud system. The data can be stored in an encrypted format to ensure anyone interception does not breach privacy. The data could still be breached through other means, such as gaining credentials to the cloud system and then having full unrestricted access to the files. The motivating idea in our thesis assumes an attacker already has full access to the digital system. Therefore the underlying data we collect should already be in a privacy-preserving format. No point in the digital system handles privacy-revealing data since the data collection system inherently captures the data in a privacy-preserving manner.

Robots are not limited to camera vision for localisation. As noted in the literature review, 3D point clouds are also commonly used to perform localisation tasks that enable navigating environments. Privacy-preserving methods in this space have been proposed, such as lifting the 3D point clouds to 3D line clouds. Such methods have been successfully attacked and allowed for image or scene reconstruction. This does not rule out a privacy-preserving way of capturing 3D point cloud data, but further research needs to be done in this area to achieve the same level of privacy as our approach.

As highlighted in the literature review, our system architecture is designed with a privacy first approach in mind. This sets it apart from many existing approaches that apply privacy-preserving methods in the digital layer. While we demonstrate the feasibility of our system to an extent, future work is required to consolidate the system design and evaluate it in real-world scenarios.

6.3 Limitations and tradeoffs

There are several tradeoffs with designing a heavily privacy-preserving approach.

The hashing functions we use destroy any local feature information in the frame and the locations of such features. For example, we can encounter a scenario with two or more similar frames. The only significant difference in the frames is the location of an object which changes as the robot moves in the environment. It is trivial to notice the object's position changes in the regular images. On the other hand, our hashed image represents a statistical summary over the entire frame. Therefore these hashes would appear very similar. This characteristic was intentionally chosen for the approach, since a machine learning method needs to detect common patterns with common frames (which is why we cannot use cryptographic hash functions). The downside of destroying detailed scene data is the potential loss of accuracy when identifying steering actions associated with specific frames. This is also the reason why our privacy-preserving method learns the environment at a lower rate than regular camera images. Regular images contain a significantly higher amount of useful information that can be learned.

The hash functions we use are dependent on pixel intensities. The pixel intensities of an image are by nature, highly sensitive to changes in brightness of a scene. In our simulated experiments, we only trained and tested scenes with a constant brightness level. We predict a change in lighting conditions results in the hash features translating along the x and y-axis, as these axes represent the minimum and maximum brightness of features in the real image. With the current setup, the reinforcement learning model will not be able to handle translated features in the hash automatically. The model will need to either be trained on varying lighting conditions or utilise different hashing methods which are not entirely dependent on the pixel intensity of an image. One possible approach is to make use of the HSL (hue, saturation, light) colour space which explicitly separates the light channel for a pixel. A consideration to keep in mind is having more information results in less privacy as this approach will process multi channel colour data instead of a single channel monochrome space.

During the initial experiments for this thesis, we tested a simpler approach that uses only a CNN model trained on a large dataset of hash data. A challenge encountered with this approach was the requirement to generate and save all the training data beforehand. The robot had to be manually teleoperated through the scene to collect this data. An issue with this approach is the robot going off trajectory and being unable to recover due to lack of training data for such a situation. It proved unfeasible to account for these scenarios when manually collecting the data. This was the motivating factor to switch to a reinforcement learning approach, where it is more logical to set a reward function for following a specific path. The robot will automatically learn from off-policy scenarios as it is penalised for crashing of going off the track.

One limitation brought on by a reinforcement learning approach is the reward function setup. As we need to know the cross track error, we require the xy position of the robot and the closest point on the track. Although the fully trained runs of the robot navigate entirely off the privacy-preserving images, the training process still requires additional positional data for the reward function. In our simulated environment, it is easy to obtain the exact position of the robot for training purposes, however this would be a greater challenge when training a robot with the same reward function in a physical scenario. Privacy would need to be sacrificed for the training phase.

Furthermore, reinforcement learning poses a challenge in a real world environment. If the robot crashes or goes off course in simulation, it is trivial to reset the environment and robot position to try again. However, in a real situation the robot would need to have some way of recovering its position. Collisions in a real training scenario would also have a tangible impact, requiring any hazards in the environment to be minimised to prevent damage.

The method we have proposed is also restricted to following pre-planned trajectories. In most practical scenarios, a robot would have many possible paths it is required to take. For example, in a warehouse there would be many decision points in a path that branch out and change directions. Further work is needed to implement a method of determining when the robot is at a branch in the path and how it should decide on the direction to steer.

CHAPTER 7

Conclusion

7.1 Summary of contributions

This thesis has demonstrated a possible approach to navigating a robot around an environment using only privacy-preserving vision in simulation. The method was designed to be compatible with an analogue circuit architecture, which ensures a higher level of privacy compared to other approaches in the literature which digitally process.

We setup a reinforcement learning pipeline and a simulated environment for our privacy-preserving robot. The privacy-preserving camera proposed by Taras et al.'s work was simulated and provided us with an image hash from the scene. We trained the robot to steer a closed-loop trajectory around a room using only these privacy-preserving images. Several hash functions were designed and evaluated under identical conditions. While we were unsuccessful in navigating the entire track, the reinforcement learning model was capable of learning part of the track before losing performance. The privacy-preserving models were compared against a standard image approach and were found to be less efficient during the learning process.

Overall, we provide an initial basis of navigation using privacy-preserving vision and invite other researchers to build upon our model and ideas.

7.2 Significance

The work presented in this thesis is a step forward towards privacy-preserving technology. Although technologies have been progressing at a rapid rate, there has been little consideration for integrating privacy-preserving techniques. Maintaining privacy in robot platforms will positively impact many sectors, from households to commercial settings. Greater trust can be placed in these systems when they are built from a privacy-preserving standpoint.

7.3 Directions for future work

7.3.1 Hash functions

In this thesis, we only tested hash functions that processed feature minima and maxima. We are not limited to this approach only, as there are many other ways to hash an image. The key characteristics of the hash still need to hold to successfully train a reinforcement learning model with it. Small changes in the original image should result in small changes in the hash, and large differences between images should result in large changes in the hash.

The hash functions we chose were handcrafted for the purpose of this thesis. There is a further avenue to find better feature extraction methods that still maintain the same degree of privacy. One way to build upon our pipeline is to also automate the hash design process. Automating this process will prevent the need to test handcrafted hash functions and will assist with finding the most effective hash. This may introduce further challenges such as significantly larger convergence time for training the model and the increase in complexity.

7.3.2 Light and moving object invariance

Our proposed model and hash function is not designed to work with changing lighting conditions. In a physical scenario we cannot guarantee a constant lighting setup. The pipeline would benefit from being robust against various lighting if the system is to be developed for practical scenarios. Similarly, the method assumes no objects other than the robot platform are moving in the scene. This ensures every position in the environment will result in the same privacy hash every time. However this assumption will not hold in physical scenarios where there is likely to be moving objects or humans in the scene. While our approach does not have a guaranteed way to address this issue, we can improve the training process to allow for a certain amount of tolerance for moving objects in the frame. For example, introducing scenarios where an object moves across the sensor's field of view and then associating those robot positions with the same steering action required to stay on the track. Our system would also benefit from pause or stop condition which detects if an unexpected object appears in the robot's path. If such patterns in the hash are learnt, it will prevent the robot from necessarily colliding into random objects found in its path. A notable paper addressing localization under varying visual conditions is SeqSLAM by Milford and Gordon [30]. The principles from this work could be applied in future research to enhance the system's capabilities.

7.3.3 Navigating more complex paths

In this thesis, we only experiment with a simple closed-loop trajectory as a starting point. Clearly this limits the range of applications for a robot. The task of solving privacy-preserving navigation for more complex paths would address a knowledge gap following this thesis. Free form navigation using privacy-preserving vision may pose as a difficult challenge, so we recommend expanding the complexity of a closed loop circuit. For example a circuit with branching paths which require knowledge of the current location and also which direction to turn to reach the intended destination. Currently, the 'map' of the environment learnt by our reinforcement learning model is stored as a model file with the weights and parameters for the neural network. A more informative format or additional data is required to allow for more complex logic such as understanding the current robot position within the environment and deciding where to turn next to efficiently reach the destination.

Bibliography

[1] R. Cuprik. 'Gathering dust and data: How robotic vacuums can spy on you.' (2023), [Online]. Available: https://www.welivesecurity.com/en/privacy/gathering-dust-and-data-how-robotic-vacuums-can-spy-on-you/.

[2] J. Fell, 'We hacked a robot vacuum — and could watch live through its camera,' *ABC News*, Oct. 2024. [Online]. Available: https://www.abc.net.au/news/2024-10-04/robot-vacuum-hacked-photos-camera-audio/104414020.

[3] P. Arntz, 'Robot vacuum cleaners hacked to spy on, insult owners,' *Malwarebytes Labs*, Oct. 2024. [Online]. Available: https://www.malwarebytes.com/blog/news/2024/10/robot-vacuum-cleaners-hacked-to-spy-on-insult-owners.

[4] S. Sami, Y. Dai, S. R. X. Tan, N. Roy and J. Han, 'Spying with your robot vacuum cleaner: Eavesdropping via lidar sensors,' in *Proceedings of the 18th Conference on Embedded Networked Sensor Systems*, ser. SenSys '20, Virtual Event, Japan: Association for Computing Machinery, 2020, pp. 354–367, ISBN: 9781450375900. DOI: 10.1145/3384419.3430781. [Online]. Available: https://doi.org/10.1145/3384419.3430781.

[5] S. Eick and A. I. Antón, 'Enhancing privacy in robotics via judicious sensor selection,' in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, 2020, pp. 7156–7165. DOI: 10.1109/ICRA40945.2020.9196983.

[6] T. Denning, C. Matuszek, K. Koscher, J. R. Smith and T. Kohno, 'A spotlight on security and privacy risks with future household robots: Attacks and lessons,' in *Proceedings*

of the 11th International Conference on Ubiquitous Computing, ser. UbiComp ’09, Orlando, Florida, USA: Association for Computing Machinery, 2009, pp. 105–114, ISBN: 9781605584317. DOI: 10.1145/1620545.1620564. [Online]. Available: https://doi.org/10.1145/1620545.1620564.

[7] A. K. Taras, N. Sünderhauf, P. Corke and D. G. Dansereau, ‘Inherently privacy-preserving vision for trustworthy autonomous systems: Needs and solutions,’ *Journal of Responsible Technology*, vol. 17, p. 100 079, 2024, ISSN: 2666-6596. DOI: https://doi.org/10.1016/j.jrt.2024.100079. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2666659624000052.

[8] S. Subashini and V. Kavitha, ‘Kavitha, v.: A survey on security issues in service delivery models of cloud computing. journal of network and computer application 34(1), 1-11,’ *Journal of Network and Computer Applications*, vol. 34, pp. 1–11, Jan. 2011. DOI: 10.1016/j.jnca.2010.07.006.

[9] F. Breda, H. Barbosa and T. Morais, ‘Social engineering and cyber security,’ in *INTED2017 Proceedings*, IATED, 2017, pp. 4204–4211.

[10] N. Kostic, ‘15 Examples of Social Engineering Attacks,’ *phoenixNAP Knowledge Base*, Jul. 2023. [Online]. Available: https://phoenixnap.com/blog/social-engineering-examples.

[11] R. Ayyagari, ‘An exploratory analysis of data breaches from 2005-2011: Trends and insights,’ *Journal of Information Privacy and Security*, vol. 8, pp. 33–56, Jul. 2014. DOI: 10.1080/15536548.2012.10845654.

[12] Office of the Australian Information Commissioner (OAIC), ‘Report shows highest number of data breaches in 3.5 years,’ *Office of the Australian Information Commissioner (OAIC)*, Sep. 2024. [Online]. Available: https://www.oaic.gov.au/news/media-centre/report-shows-highest-number-of-data-breaches-in-3.5-years.

[13] *Roomba® s Series*, https://www.irobot.com.au/roomba/s-series, Accessed: May 9, 2024.

[14] *What is a Clean Map™ Report?* https://homesupport.irobot.com/s/article/1467, Apr. 2024.

[15] International Association of Privacy Professionals (IAPP), *About the IAPP*, 2024. [Online]. Available: https://iapp.org/about/what-is-privacy/.

[16] D. O'Neil, 'Analysis of internet users' level of online privacy concerns,' *Social Science Computer Review*, vol. 19, no. 1, pp. 17–31, 2001. DOI: 10.1177/089443930101900103. [Online]. Available: https://doi.org/10.1177/089443930101900103.

[17] *Training a Classifier*, https://pytorch.org/tutorials/beginner/blitz/cifar10_tutorial.html.

[18] P. Latorre-Carmona, V. J. Traver, J. S. Sánchez and E. Tajahuerce, 'Online reconstruction-free single-pixel image classification,' *Image and Vision Computing*, vol. 86, pp. 28–37, 2019, ISSN: 0262-8856. DOI: https://doi.org/10.1016/j.imavis.2019.03.007. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0262885619300356.

[19] M. F. Duarte, M. A. Davenport, D. Takbar *et al.*, 'Single-pixel imaging via compressive sampling,' *IEEE signal processing magazine*, vol. 25, no. 2, pp. 83–91, 2008.

[20] Y. Jauregui-Sánchez, P. Clemente, P. Latorre-Carmona, J. Lancis and E. Tajahuerce, 'Single-pixel imaging using photodiodes,' in *Advances in Photodetectors-Research and Applications*, IntechOpen, 2018.

[21] C. Zhang, Y. Tian and E. Capezuti, 'Privacy Preserving Automatic Fall Detection for Elderly Using RGBD Cameras,' in *Computers Helping People with Special Needs*, K. Miesenberger, A. Karshmer, P. Penaz and W. Zagler, Eds., Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 625–633, ISBN: 978-3-642-31522-0.

[22] P. Speciale, J. L. Schonberger, S. B. Kang, S. N. Sinha and M. Pollefeys, 'Privacy preserving image-based localization,' in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 5493–5503.

[23] M. Cummins and P. Newman, 'Fab-map: Probabilistic localization and mapping in the space of appearance,' *I. J. Robotic Res.*, vol. 27, pp. 647–665, Jun. 2008. DOI: 10.1177/0278364908090961.

[24] R. Dube, D. Dugas, E. Stumm, J. Nieto, R. Siegwart and C. Cadena, 'Segmatch: Segment based place recognition in 3d point clouds,' May 2017, pp. 5266–5272. DOI: 10.1109/ICRA.2017.7989618.

[25] M. Schreiber, C. Knöppel and U. Franke, 'Laneloc: Lane marking based localization using highly accurate maps,' in *2013 IEEE Intelligent Vehicles Symposium (IV)*, 2013, pp. 449–454. DOI: 10.1109/IVS.2013.6629509.

[26] K. Chelani, F. Kahl and T. Sattler, 'How privacy-preserving are line clouds? recovering scene details from 3d lines,' in *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021, pp. 15 663–15 673. DOI: 10.1109/CVPR46437.2021.01541.

[27] F. Pittaluga, S. J. Koppal, S. B. Kang and S. N. Sinha, 'Revealing scenes by inverting structure from motion reconstructions,' in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 145–154.

[28] M. U. Kim, H. Lee, H. J. Yang and M. S. Ryoo, 'Privacy-Preserving Robot Vision with Anonymized Faces by Extreme Low Resolution,' in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2019, pp. 462–467. DOI: 10.1109/IROS40897.2019.8967681.

[29] H. Shahbazi and H. Zhang, 'Application of locality sensitive hashing to realtime loop closure detection,' in *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, IEEE, 2011, pp. 1228–1233.

[30] M. J. Milford and G. F. Wyeth, 'SeqSLAM: Visual route-based navigation for sunny summer days and stormy winter nights,' in *2012 IEEE International Conference on Robotics and Automation*, 2012, pp. 1643–1649. DOI: 10.1109/ICRA.2012.6224623.