Parking Spot Detector and Classifier

Patrick Noonan
20355426
p.noonan2@universityofgalway.ie

I. INTRODUCTION

This project guides you through the process of analysing multiple images of car parks to detect parking spaces and predict their occupancy. We will discuss these images in detail, along with the image processing pipeline and the various approaches considered to successfully identify as many parking spots as possible throughout the assignment. The pipeline maintained a consistent architecture across most images, with slight adjustments for different sets due to variations such as parking space size or overall image dimensions. For instance, the kernel size might need to change, but the core pipeline structure remained consistent. The project employs several embedded image processing techniques, including morphological operations, edge detection, thresholding, and Hough Line Transforms.

II. RESEARCH AND DEVELOPMENT

The majority of the research for this project was built upon the concepts explored in the three assignments leading up to the final project. Techniques such as intensity transforms**,** edge detection**,** Hough transforms**,** colour spaces, and morphological processing were integral to those assignments and played a crucial role in developing this project.
The aim was to consolidate these skills into a cohesive and effective image processing pipeline. While there were slight variations in implementation—such as the method used to calculate the crop regions of the parking spots—the core concepts and knowledge were directly drawn from the prior assignments.

III. EMBEDDED IMAGE PROCESSING PIPELINE

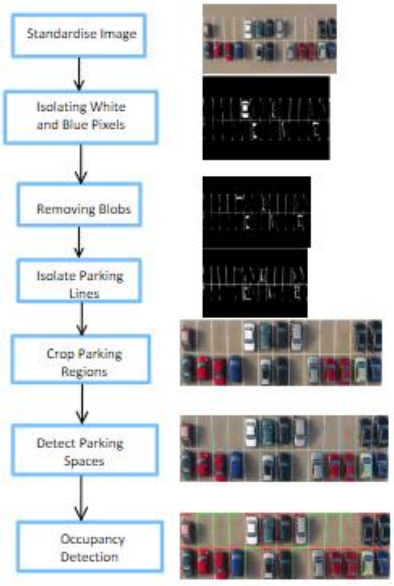


Figure 1 Processing Pipeline

A. Standardising Image Sizes

The first step in the process was to standardise the image sizes within each set, as the original images varied in dimensions. This variability would affect the impact of kernels and other operations, as the effect depends on the image size and pixel count.
Initially, I chose a size of 600x900 pixels for standardisation, which worked well for the first two sets. However, as the project progressed, it was realised that increasing the image size could capture more detail, especially for images with smaller parking spaces, thereby improving the accuracy of the subsequent processing steps.
Once the images were standardised to the chosen dimensions, consistent processing could be applied across all images, ensuring uniformity in the operations. This standardisation was crucial for maintaining consistency in the image processing pipeline.

B. Isolating White and Blue

The next step was to isolate the white colour, which represents the parking lines, from the rest of the image. This was achieved by converting the image to the HSV colour space, which helped to bring more definition to the white colour. Thresholding was applied to retain the white colour, and setting all other pixels to zero. This step isolated the parking lines, leaving them as the primary feature in the image. However, this method also retained other white elements, such as white vehicles and various pieces of white noise. The subsequent stage of the pipeline focused on eliminating large white blobs, like cars and other features, to ensure that only the parking lines remained.

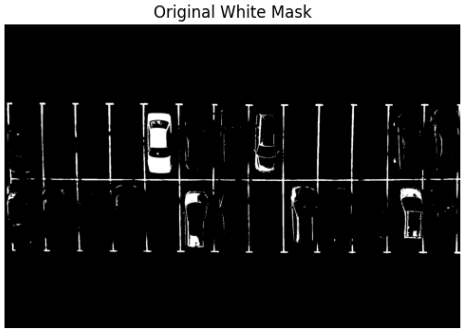


Figure 2 Isolating White Colour in the Image

C. Removing White Blobs

To remove these blobs, we applied a morphological opening, which eliminated thin white lines, leaving only the car blobs. Next, we used morphological closing to restore the car blobs to their original size. These blobs were then inverted, making them the opposite colour of the original image. By XOR-ing the original image with this modified version, we successfully removed the car blobs, leaving the parking lines isolated.

With the parking lines isolated, we could proceed with further image processing to identify the parking regions.

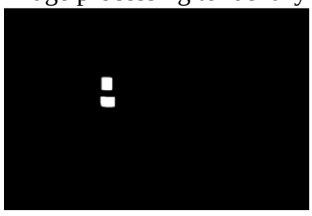


Figure 2 Blob isolated

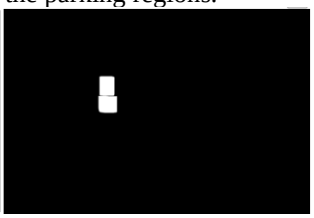


Figure 3 Blob Closed

Figure 4 Blob Inverted

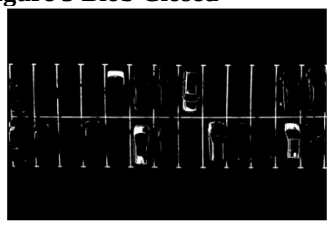


Figure 5 Blob Removed

Once these operations were carried out, we observed that the large blob was removed. However, some noise remained in the image. This noise will need to be addressed in the subsequent steps. Despite the remaining noise, removing the large blob was the most critical part of this stage in the processing chain.

D. Isolating Parking Lines

The next stage of the pipeline focuses on isolating the parking lines from the noise. This is achieved by first applying a Gaussian blur to the image, followed by Canny edge detection. The strategy is that the parking lines will produce more distinct and continuous edges compared to the noise, which should appear as smaller, fragmented spots. This approach aims to highlight the parking lines while minimising the noise, making it easier to remove in subsequent steps.

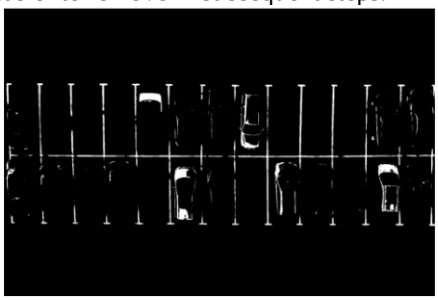


Figure 6 Before Edge Detection

Figure 7 After Edge Detection

From the images above, it is evident that the parking line information remains clear, but there is still significant information from the originally parked cars. To address this, a Hough Line Transform is applied to connect all the parking lines in the image. This technique primarily detects horizontal and vertical lines, which helps minimise the edges of the remaining cars, as their edges are less likely to be perceived as continuous straight lines than the parking lines.

Once this is complete the algorithm then isolates horizontal and vertical lines from the transformation and puts them onto the same image.

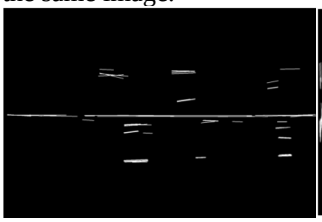


Figure 8 Horizontal Lines

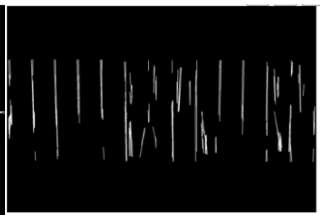


Figure 9 Vertical Lines

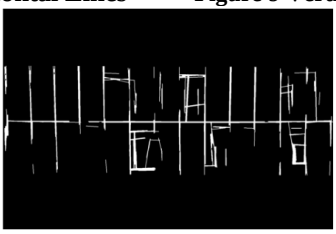


Figure 10 Lines Combined

Once this step is complete, the parking lines become very vivid. Although some noise remains, it can likely be removed through thresholding when drawing the parking lines. At this stage, we have a clear depiction of the parking area, which can now be easily cropped. The next stage in the pipeline involves cropping the individual parking areas for further processing.

E. Cropping Parking Areas

The cropping of the parking areas was achieved by projecting the sum of white pixels along the horizontal rows and vertical columns of the binary image. These projections were normalised to ensure that the values were comparable. A thresholding operation was then applied to isolate areas with a high density of white pixels, which likely represent parking areas.

The horizontal lines with significant white pixel counts were grouped based on their proximity to each other, allowing the identification of continuous horizontal regions. The vertical lines were used to determine the left and right boundaries of the parking area.

For each identified horizontal group, the topmost and bottommost pixels were used to define the vertical boundaries, while the leftmost and rightmost significant vertical lines were used for the horizontal boundaries. This approach allows for the cropping of each distinct parking area, resulting in multiple cropped regions from the original image.

Each crop corresponds to a potential parking area, effectively isolating these regions from the rest of the image. This makes it easier to further analyse the parking lines and determine the occupancy status of each parking spot, without interference from non-parking elements like road markings or other noise.

Figure 11 Before Cropping

Figure 12 Crop 1

Figure 13 Crop 2

The parking spaces have been successfully isolated from the roadways, footpaths, and other noisy elements. This isolation significantly simplifies the subsequent steps of detecting parking lines and determining the occupancy status of each parking space. By removing noisy features, now there is a cleaner, more controlled environment for accurate parking space analysis.

F. Detect Car Parking Spaces

Projections are performed on each cropped image to identify the parking lines within the crop. This is achieved by detecting spikes in the projection graphs, as shown below. These spikes correspond to the parking lines, which are then drawn on the image. The coordinates of these lines are stored in an array for occupancy assessment in the later stages of the pipeline.

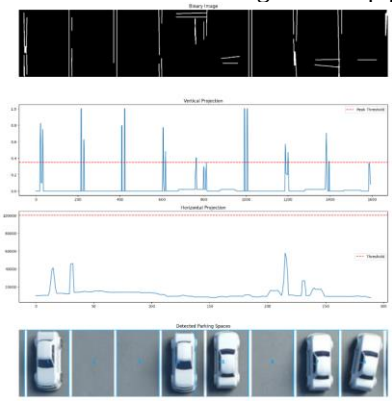


Figure 14 Detecting Parking Spaces

The projection graphs above show that the parking lines are generally clear. However, in the middle of the graph, a noisy line surpasses the threshold, causing one of the parking space lines to misalign with its actual position. This issue arises from the earlier morphological operations not eliminating enough noise. Balancing the retention of essential information with noise minimisation is a delicate task throughout the pipeline.

To mitigate false positives, several checks are implemented. These include ensuring peaks in the projections are not too close together by using a scanning window and selecting the highest peak within the window. These measures, along with additional checks further down the pipeline, help prevent false or misaligned parking lines. For example, in the projections below, even when multiple peaks are close together, the algorithm selects the highest and most appropriately spaced peaks to accurately define parking spaces.

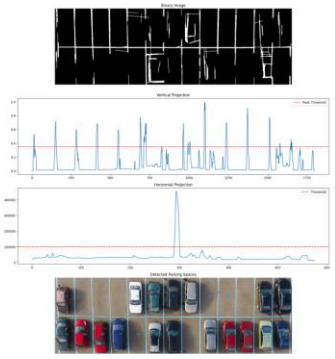


Figure 15

The easiest solution to this problem is to improve the noise reduction earlier on in the pipeline as it becomes difficult to differentiate and remove the noise as the pipeline continues. However, since the majority of the parking spots are identified further checks can be carried out before occupancy detection takes place.

G. Occupancy Detection

Once the parking spaces are successfully identified, tests can be conducted to determine which parking spaces are occupied and which are vacant. Various methods were evaluated, including tests based on pixel intensity, variance, and edge density. Among these, the combination of edge density and variance proved to be the most reliable and consistent.
The thresholds for these tests may need to be adjusted for different image sets. A parking spot is sometimes considered occupied if it surpasses both the variance and edge density thresholds. In other image sets, meeting either one of these thresholds is sufficient to classify a spot as occupied.
If a parking spot meets the thresholds, it is classified as engaged, and a red rectangle is drawn around the space. Conversely, if the spot does not meet the thresholds, it is classified as vacant, and a green rectangle is drawn, with "Vacant" labelled on the parking space.

Figure 16 Occupancy Analysis

As shown in the image above, the occupancy analysis performs well when the parking spots are correctly detected. However, there were some challenges when the camera angle was skewed, causing cars from one row to overlap with another, which impacted the accuracy of the occupancy detection.

To address this issue, the occupancy analysis was conducted separately on the top and bottom halves of each parking spot. Both halves needed to meet the thresholds for the spot to be classified as engaged. This approach proved effective for the intended use case. However, care had to be taken to avoid excessively large crops, as this could result in one-half of the spot appearing empty, incorrectly classifying an engaged spot as vacant.
With the completion of the occupancy analysis, the pipeline reaches its conclusion.

H. Conclusion of Processing Pipeline

The image processing pipeline demonstrated strong performance across most test images, particularly excelling with the first set through effective use of intensity projections, edge detection, and aspect ratio analysis. However, challenges emerged with images from sets 2 and 3, where noise and visual artifacts led to unreliable spot detection and occupancy analysis. The main failure points occurred when parking area crops included extraneous features like road markings and shadows, highlighting the pipeline's dependency on clean, well-defined regions of interest for reliable

IV. Processing Pipeline Adjusted

A. Distortion

The second image in set 2 had some distortion in it with a straight parking lot being turned into a curved line which led to some trouble in terms of trying to crop the region and also in terms of the projections as they can only draw in straight lines and the intensity transform calculations are carried out in straight lines in the horizontal direction only.

Figure 17 Distortion in Image 2 Set 2

As can be seen in the image above, the angle of the camera creates a slight curvature or warping effect across the image, which requires special attention. To address this issue, a perspective transform was applied to the crop. This transform removes distortion and standardises the image in the vertical and horizontal directions, effectively eliminating the distortion caused by the camera angle.
Although it would have been beneficial to implement this perspective correction automatically within the pipeline, due to time constraints, this enhancement was not attempted. Instead, the perspective transform was applied specifically to this image to ensure a clean, undistorted view of the parking lot.

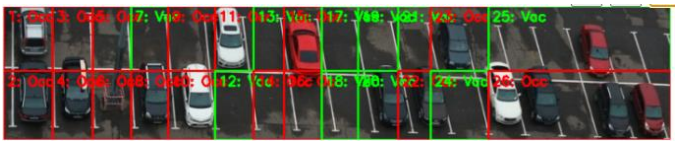


Figure 18 Before perspective transform

Figure 19 After Perspective Transform

B. Rotation

Image 1 in the final set required rotation to align parking lines horizontally and vertically, as the algorithm can only process and draw orthogonal lines. While this manual rotation improved performance, an automated system for detecting and correcting off-centre lines would enhance the pipeline's capabilities. A rotation operation was also applied to the final image in set 3 to improve performance.

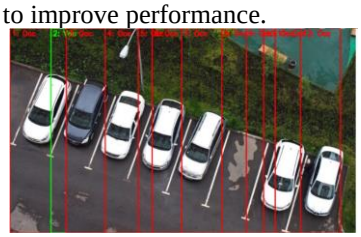


Figure 20 Before Rotation Operation

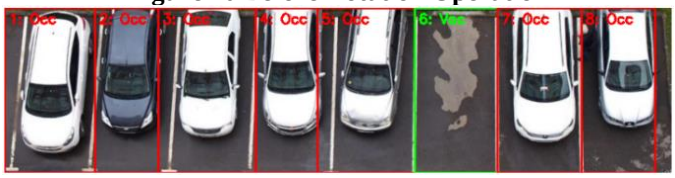


Figure 21 After Rotation Operation

C. Rescaling Resolution

Image 2 in the final set, is significantly zoomed out compared to the other two images, this required resolution upscaling from 600x900 to 3000x4500 to preserve crucial details for processing. While this increased computation time, it allowed the kernels to function effectively.

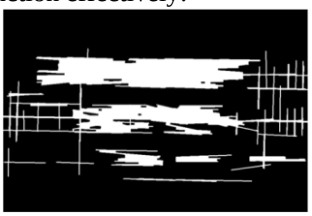


Figure 22 Combined Lines before an increase in resolution

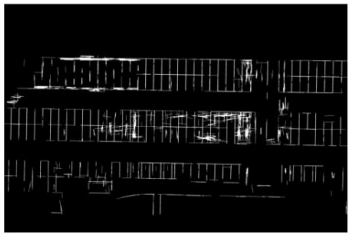


Figure 23 Combined lines after an increase in resolution

D. Adjustments to occupancy detection

As previously discussed in the Occupancy Analysis section of the report, the algorithm required adjustments for image two in set two due to the awkward angle at which the image was taken. Some parking spaces appeared occupied by cars parked in the spaces directly in front of the vacant spots. To counter this, the occupancy analysis was split into the top and bottom

halves of each parking space, requiring both halves to meet the threshold to deem the space occupied. This implementation worked effectively, with minimal instances of overlapping cars causing vacant spots to be incorrectly marked as occupied. However, darker cars with fewer discernible features struggled to be detected as engaged due to this approach.

E. *Adjustments to kernel size and threshold values*

The main adjustments between image sets focused on threshold values and kernel sizes throughout the processing pipeline. At the initial stage, threshold values were carefully tuned, particularly for detecting the blue parking spaces in set 3. These thresholds were generally reduced in later sets to preserve more parking space information, though this resulted in increased noise - creating a crucial balance between information retention and noise minimisation.
Kernel sizes required similar adjustments across sets to effectively handle white blob removal and noise management. The Hough Line Transform parameters, including Canny edge detection thresholds, needed precise tuning as noise significantly impacted performance. Images in sets 2 and 3 contained more noise, requiring stricter thresholds, though distinguishing between noise and actual parking lines often proved challenging. The key strategy was achieving consistent results after the initial thresholding and blob removal stages, as success here minimised the need for adjustments later in the pipeline, though this wasn't achievable for all images.

F. *Conclusion*

Although the image processing pipeline performs well when the crop is correct and the parking lines are visible, there are still areas where the algorithm could be improved. The concepts discussed above were applied to specific images as needed. However, for the algorithm to function effectively in real-world scenarios, these features should be automatically detected and the images processed accordingly. Implementing such checks would make the algorithm more intricate, but it would significantly enhance its adaptability and accuracy. For the images in this project, performance and accuracy improved measurably once the appropriate adjustments were made.

V. Results

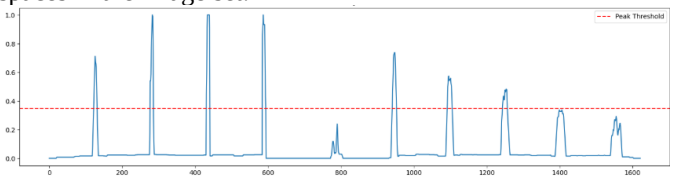
A. *Image Set 1*

	Algorithm	Actual
Parking Spots Detected	78	80
Vacant	37	41
Engaged	41	39
False Negatives	2	
False Positives	0	

The first set performance was very good considering we only detected two false negatives in the occupancy detection and we failed to detect two parking spaces. The false positives occur in the image below,

Figure 24 False Positives

The false positives occur due to the tyre markings on one of the spots and also the white line failing to be detected in the far-right of the image. This causes the edge density to go above the threshold and the occupancy to be deemed as engaged. Attempts were made to decrease the threshold to detect the white line however this effected other parking spaces in the image set.



In the graph above with a slight decrease in the peak threshold the parking line would be detected but this effected other parking spaces in the set so it was decided to leave the threshold at the current value.

B. *Image Set 2*

	Algorithm	Actual
Parking spots Detected	90	96
Vacant	53	58
Engaged	37	38
False Negatives	2	
False Positives	3	

The first two images in this set performed very well, but the last image posed significant challenges, particularly with one specific crop shown below, which led to a high failure rate in detecting parking spaces on the left side of the image. As a result, the algorithm performed poorly in terms of parking space detection for that section.

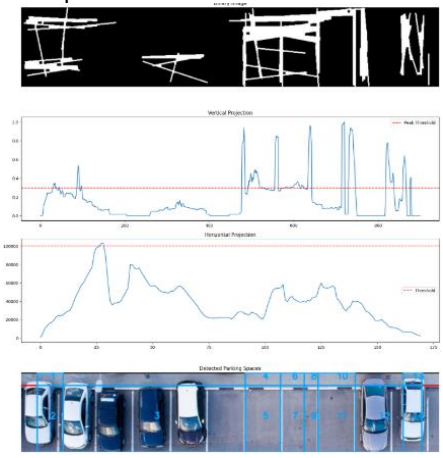


Figure 25 Poor Performing Crop

The primary issue with the image above is the inclusion of the footpath and curbing in the crop, which introduces substantial

noise into the projection. This noise obscures the peaks of the parking lines, leading to inadequate detection. The crop below is a parking lot in the same image which performs well without the noise provided by the footpath.

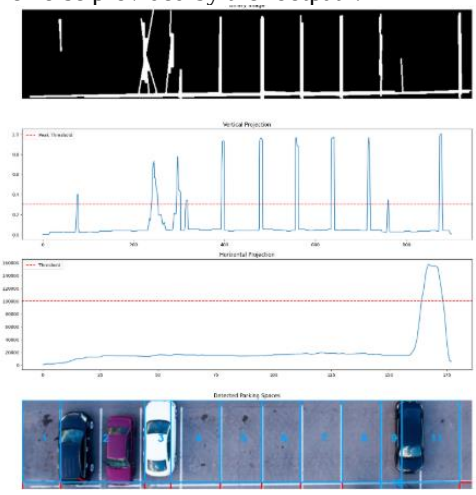


Figure 26 Parking lot in the same image

When the footpath is excluded, the peaks of the parking lines are much clearer, resulting in a higher detection rate. If there were a way to manually crop the image or remove this noise earlier in the algorithm, it would significantly enhance the algorithm's performance.

C. Image Set 3 for images 1 and 3

	Algorithm	Actual
Parking Spot Detected	45	44
Vacant	21	30
Engaged	24	14
False Negatives	11	
False Positives	0	

Image 2 was not used in the investigation of results, as it was not in a suitable state for detecting parking spaces due to poor time management. The increase in resolution did improve the retention of information for each parking space; however, aligning the kernels used in the other two images with those applied to Image 2 proved challenging. Further analysis of processing Image 2 independently with its kernels for parking space detection is necessary. Due to time constraints, it is unlikely I will complete this part of the project, but I prioritised optimising the performance of Images 1 and 3.
Another approach being considering for detecting parking spaces is the barcode detection method used earlier in the semester. This involves using morphological closing and dilation to create large contours and then applying the large contours for image cropping. While this method did not work well for the more zoomed-in images in the set, it may be more effective for Image 2 due to the smaller parking areas
Despite these issues, the other two images performed well, with most false detections occurring in the third image. Both images had rotations applied to them, with the third image receiving the rotation later in the project to improve performance. The false detections are shown in the image below.

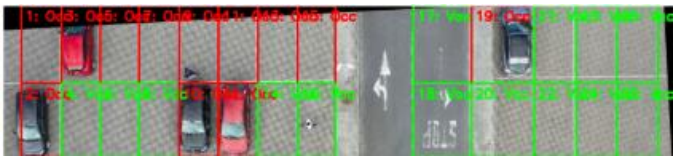


Figure 27 Crop with a high number of false positives

The rotation of the image causes the crop to extend beyond the frame, resulting in a heavy black edge across all the parking spaces in the top left corner. This edge exceeds the edge density threshold, causing all spaces in this area to be incorrectly marked as occupied. Increasing the threshold to mitigate this issue leads to parking spaces in other images being falsely identified as vacant. The only viable solution is to implement safety checks during the rotation process to ensure that the image remains within frame boundaries. Also the image wrongly detects the roadway as a parking space also.
Image 1 in Set 3 performs very well successfully identifying all the spaces and calculating the correct occupancy for each image.

VI. Impact on Embedded Applications

The implementation would perform reasonably well on embedded applications, with several design choices supporting efficient embedded processing. The standardisation to 600x900 pixels provides a relatively low resolution for the majority of images, helping manage computational requirements. This fixed resolution ensures predictable memory usage and processing demands across different input images. However, when dealing with zoomed-in images, increasing the resolution to higher levels would significantly tax the embedded system's resources, potentially causing performance bottlenecks and increased processing times.
The implementation's cropping strategy is particularly beneficial for embedded applications. By isolating and processing only the relevant parking space regions rather than the entire image, the system substantially reduces the computational workload. This selective processing approach is crucial for embedded systems with limited processing power, as it minimises unnecessary calculations and memory usage.

VII. Conclusion

In conclusion, the processing pipeline performs well for the first set of images, but its performance progressively declines in the subsequent sets. It remains effective in Set 2, except for one crop affected by footpath noise. In Set 3, while the pipeline processes one image efficiently, it struggles with the second image due to insufficient time management to implement an adequate solution. Although the third image detects parking spaces accurately, occupancy detection suffers under the current pipeline configuration.
Key improvements to the pipeline include enhancing noise removal in the initial stages after image standardization. Currently, only one iteration of erosion is applied to remove parking lines and blobs. An iterative approach to gradually removing smaller blobs could be more effective, as many white cars remain in the image after this stage, causing complications further down the pipeline.

Another improvement is the automatic implementation of features like rotation and distortion removal. Currently, these abnormalities are addressed manually for each image, which is sufficient for this assignment but impractical for real-world applications. Automating this process would enhance the pipeline's robustness and scalability.
The occupancy detection could also be refined. While it works well in most cases, it produces false detections in edge cases. More rigorous testing could help eliminate these inaccuracies, improving overall reliability.
Finally, better time management would have allowed more comprehensive development, particularly for the challenging third set. Although ample time was allocated, poor time management led to a rushed implementation for the second image in Set 3 and missed opportunities for simple improvements.
More extensive research into alternative approaches could have significantly enhanced performance. The current implementation leaned heavily on previous assignments, but deeper exploration could have introduced more effective or optimized methods. A more substantial investment in research and planning at the project's outset would likely have saved time and improved outcomes throughout.

VIII. References

Solomon, C. and Breckon, T. (2011) Fundamentals of digital image processing a practical approach with examples in Matlab. West Sussex: John Wiley & Sons.

Gonzalez, R. and Woods, R.E. (2018) *Digital Image Processing FOURTH EDITION Vol Fourth*. 330, Hudoson Street, New York, New York: Pearson.

Noonan, Patrick (2024) '*Assignment 1 – Image Signal Processor*', University of Galway. Unpublished individual project.

Noonan, Patrick (2024) *'Assignment 2 – Barcode Detector'*, University of Galway. Unpublished individual assignment.

Noonan, Patrick (2024) 'Assignment 3 – Laneway Detection', University of Galway. Unpublished individual assignment.

OpenCv (2015) *Geometric Transformations of Images*, *OpenCV*. Available at: https://docs.opencv.org/4.x/da/d6e/tutorial_py_geometric_transformations.html#:~:text=Perspective%20Transformation&text=To%20find%20this%20transformation%20matrix%2C%20you%20need%204%20points%20on,getPerspectiveTransform. (Accessed: 08 December 2024).

GeeksforGeeks (2023) *Python opencv - morphological operations*, *GeeksforGeeks*. Available at: https://www.geeksforgeeks.org/python-opencv-morphological-operations/ (Accessed: 25 November 2024).

Price, S. (1996) *Edges: The Canny Edge Detector*, *Informatics Homepages Server*. Available at: https://homepages.inf.ed.ac.uk/rbf/CVonline/LOCAL_COPIES/MARBLE/low/edges/canny.htm (Accessed: 20 November 2024).

Mathworks (no date) *Understanding Color Spaces and Color Space Conversion*, *Understanding color spaces and color space conversion*. Available at: https://uk.mathworks.com/help/images/understanding-color-spaces-and-color-space-conversion.html (Accessed: 20 November 2024).

IX. Appendix