UNIVERSITEIT•STELLENBOSCH•UNIVERSITY
jou kennisvennoot • your knowledge partner

Road Detection in Images Using a Probabilistic Graphical Model

N.T.N. Smal
21558388

Report submitted in partial fulfilment of the requirements of the module Project (E) 448 for the degree Baccalaureus in Engineering in the Department of Electrical and Electronic Engineering at Stellenbosch University.

Supervisor: Dr Corné Van Daalen

October 2021

Acknowledgements

I would like to thank Dr Corné Van Daalen for all the help and guidance throughout this project. I would like to thank my family and friends for always being there for me whenever I need any help. I would also like to thank Larry Page and Sergey Brin for creating Google. Finally I would like to thank Dr Herman Kamper for being an inspiring lecturer and for this amazing report template. :)

UNIVERSITEIT•STELLENBOSCH•UNIVERSITY
jou kennisvennoot • your knowledge partner

Plagiaatverklaring / *Plagiarism Declaration*

1. Plagiaat is die oorneem en gebruik van die idees, materiaal en ander intellektuele eiendom van ander persone asof dit jou eie werk is.
   Plagiarism is the use of ideas, material and other intellectual property of another's work and to present is as my own.

2. Ek erken dat die pleeg van plagiaat 'n strafbare oortreding is aangesien dit 'n vorm van diefstal is.
   I agree that plagiarism is a punishable offence because it constitutes theft.

3. Ek verstaan ook dat direkte vertalings plagiaat is.
   I also understand that direct translations are plagiarism.

4. Dienooreenkomstig is alle aanhalings en bydraes vanuit enige bron (ingesluit die internet) volledig verwys (erken). Ek erken dat die woordelikse aanhaal van teks sonder aanhalingstekens (selfs al word die bron volledig erken) plagiaat is.
   Accordingly all quotations and contributions from any source whatsoever (including the internet) have been cited fully. I understand that the reproduction of text without quotation marks (even when the source is cited) is plagiarism

5. Ek verklaar dat die werk in hierdie skryfstuk vervat, behalwe waar anders aangedui, my eie oorspronklike werk is en dat ek dit nie vantevore in die geheel of gedeeltelik ingehandig het vir bepunting in hierdie module/werkstuk of 'n ander module/werkstuk nie.
   I declare that the work contained in this assignment, except where otherwise stated, is my original work and that I have not previously (in its entirety or in part) submitted it for grading in this module/assignment or another module/assignment.

21558388	
Studentenommer / *Student number*	Handtekening / *Signature*
N.T.N Smal	April 25, 2022
Voorletters en van / *Initials and surname*	Datum / *Date*

Abstract

English

The aim of this project is to identify which regions of an image taken by a car-mounted camera are road. The purpose of determining which regions of the images are road is to provide a means for self-driving cars to reliably perceive the road and any potential hazards in the road, allowing them to operate safely. To identify the road in images, one can use several characteristics that are typical to roads found in images such as the road colour, texture, or position in the image.

This report focuses on the use of probabilistic graphical models (PGMs) as a technique to model and reason about the uncertain relationships between the typical road characteristics. Superpixels were used to segment images taken by a car-mounted camera, to reduce the processing power required to identify where the road is in the images. A logistic regression model was trained to predict whether a superpixel consists of road pixels based on the superpixel's colour characteristics. The Canny edge detector was used to obtain the "edges" between neighbouring superpixels. The logistic regression model predictions and information about "edges" between neighbouring superpixels were then combined using a PGM to infer whether a superpixel is road based on these uncertain characteristics. The PGM was able to reliably identify where the road is in images taken by a car-mounted camera using these characteristics; however, the results can be improved by extending the PGM to include other typical road characteristics.

Afrikaans

Die doel van hierdie projek is om te identifiseer watter streke van ’n beeld wat deur ’n motorgemonteerde kamera geneem is, pad is. Die doel om te bepaal watter streke van die beelde pad is, is om ’n manier te bied vir selfbesturende motors om die pad en enige potensiële gevare in die pad betroubaar waar te neem, sodat hulle veilig kan funksioneer. Om die pad in beelde te identifiseer, kan ’n mens verskeie kenmerke gebruik wat tipies is vir paaie wat in beelde voorkom, soos die padkleur, tekstuur of posisie in die beeld.

Hierdie verslag fokus op die gebruik van Grafiese waarskynlikheidsmodelle (GWM’e) as ’n tegniek om te modelleer en te redeneer oor die onsekere verwantskappe tussen die tipiese pad-eienskappe. Superpixels word gebruik om beelde te segmenteer wat deur ’n motorgemonteerde kamera geneem is, om die verwerkingskrag te verminder wat nodig is om te identifiseer waar die pad in die beelde is. ’n Logistiese regressiemodel is opgelei om te voorspel of ’n superpixel uit padpixels bestaan, gebaseer op die superpixel se kleureienskappe. Die Canny rand bespeurder is gebruik om die ”rande”tussen naburige superpixels te verkry. Die logistiese regressiemodel se voorspellings en inligting oor ”rande”tussen naburige superpixels is dan gekombineer met behulp van ’n PGM om af te lei of ’n superpixel pad is gebaseer op hierdie onsekere eienskappe. Die PGM kon betroubaar identifiseer waar die pad is in beelde wat deur ’n motorgemonteerde kamera geneem is deur hierdie kenmerke te gebruik. Die resultate kan egter verbeter word deur die PGM uit te brei om ander tipiese padeienskappe in te sluit.

Contents

List of Figures

List of Tables

Nomenclature

Variables and functions

$p(x)$	Probability density function with respect to variable x.
$p(A)$	Probability of event A occurring.
$p(x\|y)$	The probability of x conditional on y.
$\text{pa}(x)$	The parents of node x.

Acronyms and abbreviations

NN	Neural network
PDF	probability density function
MRF	Markov random field
BN	Bayesian network
RGB	Red, green, and blue
SLIC	Simple linear iterative clustering
HSV	Hue, saturation, and value
PGM	Probabilistic graphical model
DAG	Directed acyclic graph
EMDW	Elementary My Dear Watson (Stellenbosch PGM library)
CSV	Comma-separated values
GLCM	Gray-level co-occurrencematrix

Chapter 1

Introduction

There are approximately 1.35 million fatal car crashes every year worldwide and an additional 20-50 million people are injured or disabled as a result of traffic accidents [1]. One solution to help prevent traffic accidents is by using self-driving cars. Self-driving cars help reduce the number of traffic accidents because they remove sensing and perception errors that often cause drivers to have accidents. Self-driving cars are also able to eliminate crashes that are caused by the driver getting distracted or driving under the influence of drugs or alcohol; these crashes account for a large percentage of the crashes that occur worldwide each year.

1.1. Project Aim

Self-driving cars need to be able to reliably perceive the road and any hazards on the road to operate safely. The primary goal of this project is to reliably identify which regions of an image taken by a car-mounted camera are road. This should be accomplished by making use of several characteristics and relationships which are typical to road surfaces and using a probabilistic graphical model to combine the uncertain characteristics and relationships in a principled way.

There are two primary techniques for segmenting images; these techniques are known as semantic segmentation and instance segmentation. The basic difference between these two techniques is the way pixels are labelled; semantic segmentation treats multiple objects of the same class as a single entity, while instance segmentation dives a bit deeper and identifies which object instance each pixel belongs to and thus treats multiple objects of the same class as distinct instances [2]. Figure 1.1 provides an example of semantic and instance segmentation. Since the focus of this project is to identify which regions of an image are road, the instances of the road class are not relevant, hence semantic segmentation will be used.

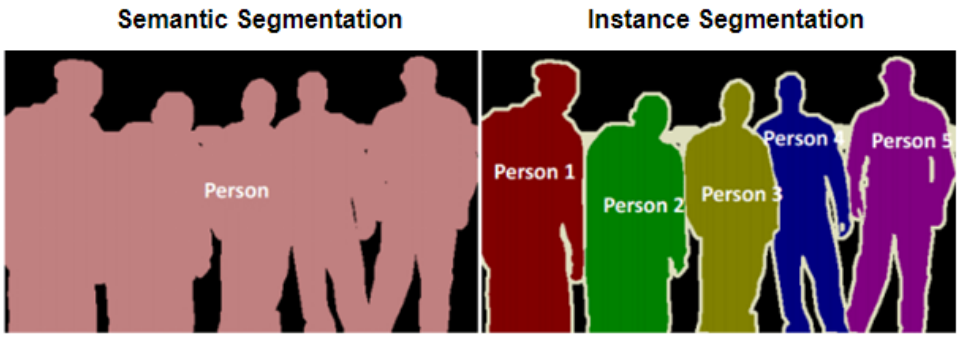


Figure 1.1: Example of semantic segmentation vs instance segmentation, image reproduced from [3].

1.2. Project Scope

The project involves the detection of road in images taken by a car-mounted camera, the assumptions made to ensure a good solution for this project are:

- The car-mounted camera is attached to the front of the vehicle. This assumption has no effect on the results, but it ensures that self-driving cars may use the results to determine where the road ahead of them is.
- A single image is considered at a time
- The images will only contain tarred roads
- Road lines are not considered as road, road lines share very different colour characteristics to the rest of the road and self-driving cars will not want to be driving on the road lines.

1.3. Project Specifications

The specifications for this project are:

- The solution should work in all environments where there are tarred roads such as rural highways, suburban streets, or city streets.
- The solution should work in all lighting and weather conditions.
- The solution should be able to execute offline.
- No accuracy is specified; however, it is important that the solution is very accurate before it can be used by self-driving cars.

1.4. Solution Overview

In this project the images are first segmented using superpixels, probabilities of whether each superpixel consists of road pixels are then obtained by making use of typical road characteristics. Typical road characteristics are obtained by firstly making use of a logistic regression model and secondly the "edges" found between road and non-road superpixels. The information obtained regarding the typical road characteristics found in the images are then combined using a probabilistic graphical model to infer where the road is in the images. Figure 1.2 depicts an overview of the solution.

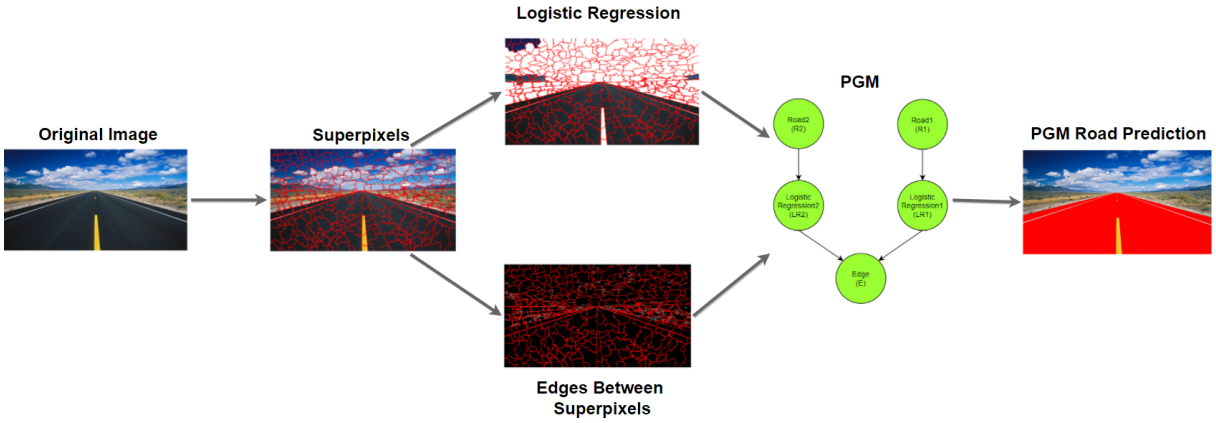


Figure 1.2: Overview of the process followed to obtain the final road prediction solution.

1.5. Report Layout

The report is structured as follows:

- Chapter 2 provides the information necessary to understand what superpixels are, why superpixels are used, the superpixel algorithms that are currently most used, how to implement the superpixel algorithms, and how the superpixel algorithm used for identifying where the road is in an image was chosen.
- Chapter 3 discusses how a logistic regression model can be trained to predict which superpixels consist of road pixels, and the accuracy of the predictions made by the logistic regression model.
- Chapter 4 introduces the concept of edges in an image and discusses how the Canny edge detector can be used to identify where the edges in images taken by a car-mounted camera are.
- Chapter 5 discusses how the edges obtained in Chapter 4 can be applied to superpixels, and how to determine the neighbours of each superpixel.

- Chapter 6 provides an overview of probabilistic graphical models, and discusses the main graphical structures used for modelling systems with uncertain relationships, and how to infer latent random variables from these graphical structures.
- Chapter 7 discusses how a probabilistic graphical model was setup to infer whether a superpixel consists of road pixels given the information obtained in Chapters 3 and 5.
- Chapter 8 discusses potential improvements or extensions that can be made to the existing PGM.
- Chapter 9 provides a summary of the project and concludes on the effectiveness of using a probabilistic graphical model to identify where the road is in images taken via a car-mounted camera.

Chapter 2

Superpixels

This chapter introduces superpixels and explores how they may be used to reduce the amount of processing power necessary to extract meaningful information from images. The method of determining which superpixel algorithm will work best for segmenting an image containing road surfaces will be described, and the results will be analysed.

2.1. Superpixel Definition

There are thousands of pixels in an image, many of these pixels often appear similar since it takes hundreds or possibly thousands of pixels to represent one object. It would be very useful to group pixels that share similar visual properties, such as colour, to reduce the processing power required when trying to identify pixels that may belong to a specific object.

One method of grouping pixels that share similar visual properties is superpixels. Superpixels are formed by grouping pixels that have similar colour characteristics and are in the same region of an image. As a result, superpixels often align better with the edges in an image than regular pixels; the concept of what is meant by an "edge" in an image is introduced later in Chapter 4. Because all pixels in a superpixel share common characteristics such as colour and location, superpixels frequently carry more information. Superpixels are useful for providing a compact representation of images, resulting in far less computational power being required to identify objects in images. Determining whether a superpixel belongs to the road is also significantly easier than trying to determine whether a single pixel belongs to the road due to there being more information available when looking at superpixels. Since the pixels that form the road and the pixels that form the road lines share very different colour characteristics it was decided that the road lines will not be considered as part of the road.

2.2. Image and Superpixel Representation in Computer Vision

Images are represented in computers as 3D matrices with 3 channels for red, green, and blue (RGB). The RGB values range from 0 to 255, where each value represents the intensity of the red, green, or blue channel for a pixel. In the range of 0 to 255, 0 represents the lowest intensity, indicating that the colour channel is not visible, and 255 represents the highest intensity, indicating that the colour channel is most visible. The output obtained after running each of the superpixel algorithms is a 2D matrix (the shape of the original image) filled with a value for each pixel representing which segment it belongs to. Figure 2.1 depicts a basic example of a segmented image. As illustrated in Figure 2.1, the original image is split into five segments, and all pixels that belong to a segment are represented by the same value as the number of that segment.

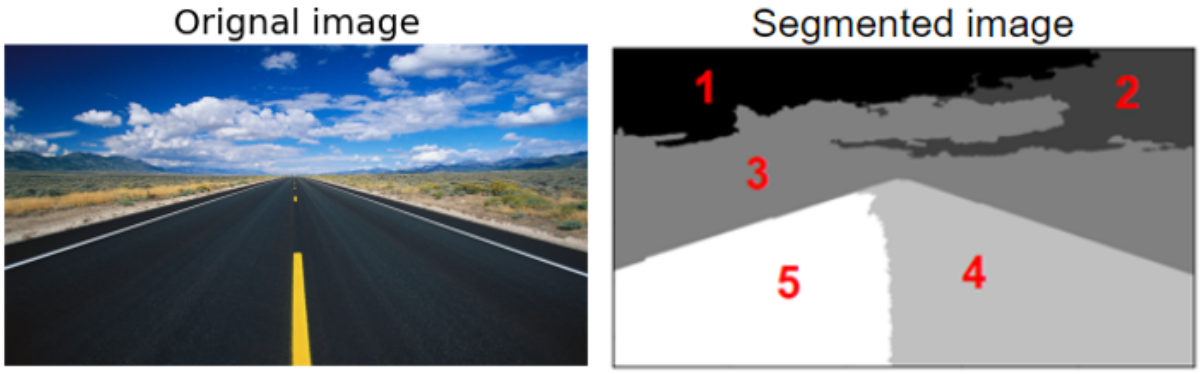


Figure 2.1: Example of an image partitioned into five segments, with each shade corresponding to the labelled number.

2.3. Superpixel Algorithms

There are several methods for obtaining superpixels from images. Four methods will be tested and analysed to determine which method is best suited for identifying which segments of an image are road. The four methods that will be tested and analysed are Felzenszwalb's efficient graph-based segmentation [4], Quickshift image segmentation [5], simple linear iterative clustering (SLIC) segmentation [6], and lastly compact watershed segmentation [7].

Felzenszwalb's efficient graph-based segmentation is popular in computer vision as it is a fast image segmentation algorithm; however, the size and number of segments obtained can vary greatly depending on local contrast. Quickshift image segmentation is a relatively new image segmentation algorithm that is based on an approximation

of kernelized mean shift and is thus part of the local mode-seeking algorithm family. Mean Shift is a centroid-based algorithm that adjusts centroid candidates to be the mean of the points in a particular region. The quickshift method is applied to the 5D space consisting of colour information and image location. A benefit of using the quickshift algorithm is that it computes a hierarchical segmentation on multiple scales simultaneously. The important aspect of hierarchical segmentation is that it preserves spatial and neighbouring information among segmented regions. The SLIC algorithm is very efficient and works by simply performing k-means clustering in the 5D space of colour information and image location. Unlike the other algorithms the compact watershed algorithm requires a grayscale gradient input image, where bright pixels denote a boundary between segments. Both the SLIC and compact watershed algorithms are expected to produce more regularly shaped segments due to there being an additional compactness argument that makes it more difficult for large segments to form.

With both the quickshift and SLIC algorithms, there is a trade-off between distance in colour space and distance in image space. Because the purpose of using the segmentation algorithms will be to detect which regions of an image are road and all road segments are expected to have similar colour characteristics, the colour space will be given more weight. With more weight given to the colour space in the quickshift algorithm, it is expected that the quickshift algorithm will outperform the other three algorithms since its clustering method excels when there are many clusters compared to the k-means clustering used by the SLIC algorithm [8].

2.4. Results Obtained from the Superpixel Algorithms

To obtain superpixels from images containing road using the four superpixel algorithms discussed the scikit-image library [9] was used. Scikit-image is an open-source image processing library with a large collection of algorithms available for the Python programming language. The four superpixel algorithms were applied to the same image taken by a car-mounted camera to allow for an easy comparison of the different algorithms. The results obtained are shown in Figure 2.2.

As expected, both the SLIC and compact watershed algorithms produce more regularly shaped superpixels which causes them to perform poorly when differentiating between road and non-road segments particularly on the border of the road. This poor performance is most noticeable when looking at superpixels near the road lines as depicted in Figure 2.3. Road lines are very narrow which makes it extremely difficult to obtain regularly shaped superpixels that do not group sections of the road lines in the road superpixels.

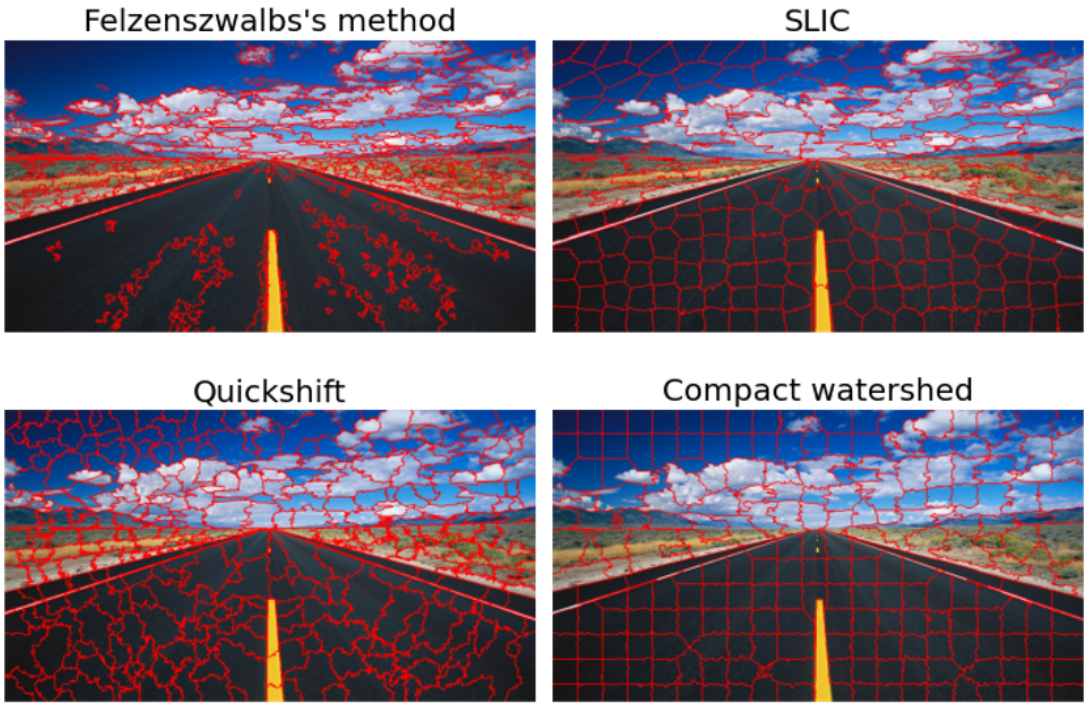


Figure 2.2: Comparison of superpixels obtained from the four superpixel algorithms.

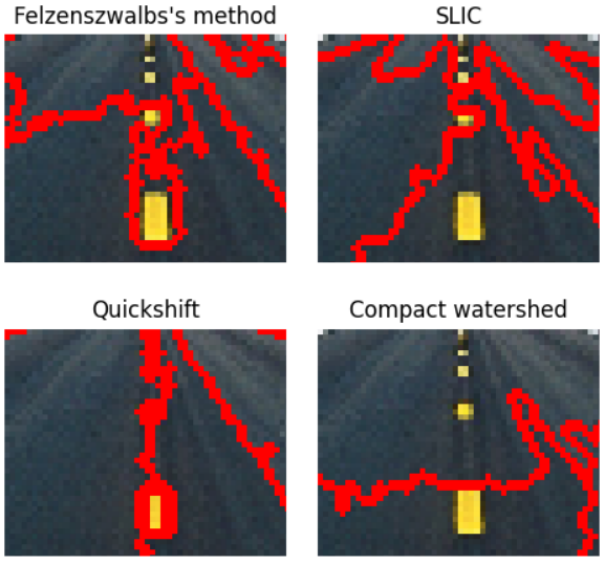


Figure 2.3: Comparison of the four superpixel algorithms' performance when zoomed in on road lines.

Since the quickshift algorithm places more weight on colour space it outperforms the other three methods, especially when looking at the road lines which are not considered as part of the road for this project. Although it may be difficult to see due to the thickness of the superpixel boundaries indicated in red in Figure 2.3, all the yellow road lines are different segments for the quickshift algorithm, whereas that is clearly not the case for the other three algorithms. It was decided that the quickshift algorithm will be used when trying to identify which regions of an image are road as it seldom groups road and non-road pixels and is therefore best suited to our application.

Chapter 3

Logistic Regression

This chapter discusses logistic regression as a solution to the problem of predicting whether a superpixel belongs to road. The performance of using logistic regression for identifying which superpixels belong to road is evaluated and the areas where the logistic regression performs poorly and the reason for it performing poorly is discussed. An overview of the process followed to train the logistic regression model and how the training of the model affects the performance of the algorithm is given. A PGM will combine the results from this chapter with the results from the subsequent chapters to determine if a superpixel is made up of road pixels.

3.1. How and Why Logistic Regression Is Used

Since images are represented in computers as matrices with specific RGB values for each pixel, the computer cannot know which superpixels belong to road and which superpixels belong to other objects in the image. Two methods were considered for identifying which superpixels in an image consist of road pixels. The two methods considered were either training a neural network (NN) or training a logistic regression model to predict whether a superpixel consists of road pixels.

It is expected that a properly trained NN will perform much better than a logistic regression model, however, a NN requires much more resources and time to train than a logistic regression model. A NN will most likely still not perform perfectly once properly trained due to the fact that some road and non-road superpixels share almost identical colour characteristics. Because the focus of this project is to use a PGM to combine many uncertain characteristics, it was decided that logistic regression would be used. It is more effective to use a method that performs slightly worse but takes less resources and time to train, allowing more time to be spent obtaining other characteristics typical to road to use in the PGM. Logistic regression is a very convenient classification algorithm since the classification problem is a binary classification problem, the superpixel is either road or not road. Logistic regression is a linear model, however, unlike other linear models such as linear regression, instead of fitting a line to the data, logistic regression instead uses the

sigmoid function. A sigmoid function is a mathematical function having the characteristic that can take any real value and map it to a value between 0 and 1 using an "S" shape. The equation of the sigmoid function is given by

$$S(x) = \frac{1}{1 + e^{-x}} \tag{3.1}$$

where e is the base of the natural logarithm. An example of how a logistic regression model can be used to predict the probability of a superpixel being road based on the intensity of red in the superpixel is shown in Figure 3.1.

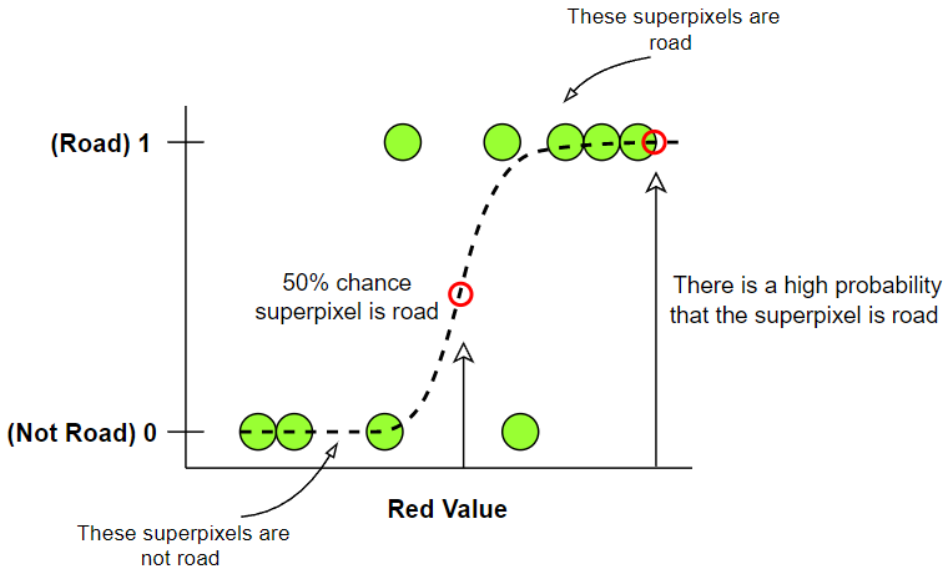


Figure 3.1: Example of logistic regression applied to superpixel classification

In the example given in Figure 3.1, the probability of a superpixel belonging to road can be obtained by mapping the red value of a superpixel to the sigmoid function and reading the probability from the vertical axis. In this example superpixels that have high red values have a high probability of belonging to road and superpixels with low red values have a low probability of being road.

3.2. Training a Logistic Regression Model to Predict Whether a Superpixel is Road

To train a logistic regression model data labelled as being either "road" or "not road" is required. The data labelled as road was obtained by cropping out road sections from different images containing road to ensure that there is a wide enough range of RGB intensities that belong to road. The RGB intensities of each pixel in the cropped images were then extracted using Python and added to an Excel spreadsheet using XlsxWriter [10], which is a Python module that can be used to write data to an Excel spreadsheet. The

difference between the red and green values for each pixel was also added to the Excel spreadsheet since pixels that belong to road often have very similar red and green values. If the difference between red and green intensities is very large the logistic regression model can use that information to predict that a pixel has a low probability of being road. The difference between the red and green intensity values can be found as shown in Equation 3.2.

$$\text{Difference between red and green intensity values} = |\text{red} - \text{green}|. \tag{3.2}$$

The data labelled as "not road" was initially obtained by using images that had very intense RGB values, however after testing the model, images that shared similar colour characteristics to road but are not road were also used. This is because if pixels with similar colour characteristics to road but are not road are not used to train the logistic regression model, the model will be biased to always predict that pixels with a high intensity are not road and pixels with a low intensity are road. The data labelled as "not road" was then added to the Excel sheet with the data labelled as "road" using Python.

After obtaining all the labelled data, the logistic regression model was trained using the logistic regression classifier found in the scikit-learn library, which is a very useful machine learning library for the Python language. Once the model has been trained it can be used to predict whether a superpixel belongs to road, the arguments for the trained logistic regression model are the red, green, and blue intensity values and the difference between the red and green values for each superpixel. To obtain the RGB values of each superpixel the average intensities for all the pixels in the superpixel were used, thus the red intensity value for a superpixel is the average of the red intensity values for all the pixels that make up a superpixel. After the RGB values for a superpixel is obtained the difference between the intensities values of the red and green channels can easily be found following the same procedure shown in Equation 3.2. To analyse the performance of the logistic regression model all the superpixels with a probability greater than or equal to 0.5 of being road were classified as road and are displayed alongside the original image the results can be observed in Figure 3.2.

To revisit the previous chapter and ensure that the correct decision was made regarding the superpixel algorithm, the same was logistic regression model was used to predicted whether superpixels obtained from the SLIC algorithm were road. To ensure that the test was fair, a similar number of segments were obtained from each algorithm; for the quickshift algorithm 376 segments were obtained and from the SLIC algorithm 377 segments were obtained. The results of which superpixels had a probability greater than 0.5 of being road is using the SLIC algorithm can be seen in Figure 3.3, it is observed that

the SLIC algorithm performs very poorly compared to the quickshift algorithm especially when looking at segments that contain road lines. Sections of the road are grouped with the road line segments resulting in chunks of the road being removed. From these results it is clear that decisions made in Chapter 2 were indeed correct and the quickshift algorithm does perform best when obtaining superpixels on images containing road.

Figure 3.2: Prediction of where the road is made by the logistic regression model. In the image on the right superpixels shown in colour have a probability greater than or equal to 0.5 of being road according to the logistic regression model.

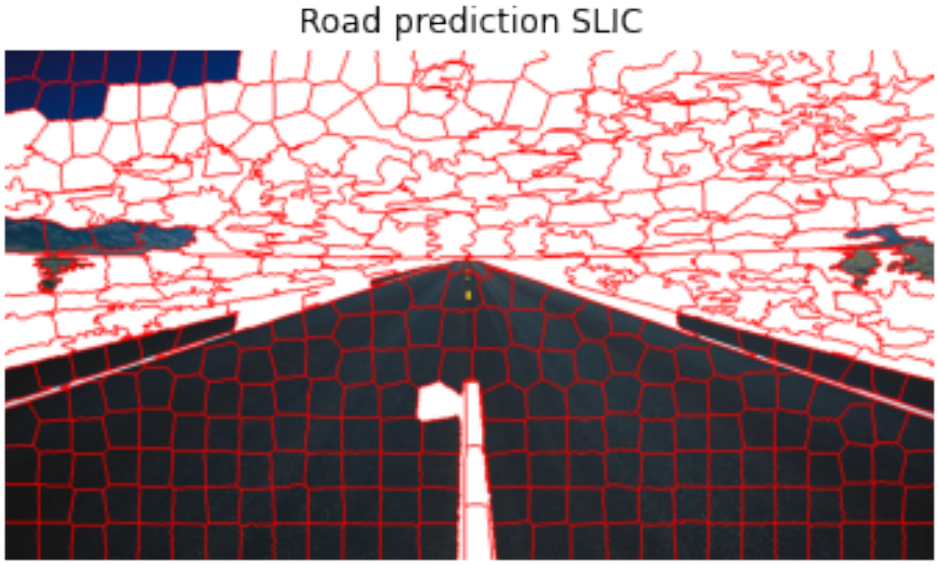


Figure 3.3: Prediction of where the road is made by the logistic regression model for SLIC superpixels. Superpixel shown in colour have a probability greater than or equal to 0.5 of being road.

3.3. Evaluation and Conclusions

The predictions made by the from logistic regression model are accurate, however, there are certain instances that cause the prediction to be incorrect one instance is when a superpixel that is not road shares very similar colour characteristics to a superpixel that is road. Another instance in which the logistic regression model performs poorly is in images where the road pixels have very intense RGB values this can be seen in Figure 3.4 where a lot of superpixels are incorrectly predicted to not be road. The instances that cause the logistic regression model to perform poorly can perhaps be addressed in the future by pre-processing the images, for example using the hue, saturation, and value (HSV) colour space instead of the RGB colour space. The advantage of using the HSV colour space is that it separates the intensity of the image from the colour information in the image, this is very useful for resilience to changes in lighting.

Figure 3.4: Performance of logistic regression model on images containing bright road segments.

Chapter 4

Canny Edge Detection

This chapter introduces the Canny edge detector and discusses the problems that can be addressed using the Canny edge detection algorithm. Detecting sharp changes in image brightness is very useful since a sudden change in the brightness of an image typically indicates a change in objects. A sharp gradient in the colour intensity of an image is known as an "edge" and is likely to indicate discontinuities in depth or a change in material properties, this can be very helpful when looking at images containing road since it can potentially indicate where the border of the road surface is. The Canny edge detector, created by John F. Canny as a means to extract meaningful structural information from different objects and reduce the amount of data required to be processed [11], is one solution for handling this issue.

4.1. How the Canny Edge Detector Functions

The Canny edge detection algorithm is a multi-stage algorithm that can be broken down into four steps [12]:

1. **Noise Reduction:** A 5×5 Gaussian filter is applied to the image in order to smooth and remove noise in the image. This step is essential since edge detection algorithms are easily affected by noise, and the Gaussian filter helps reduce the effects of noise and prevent any false detection caused by noise in the image.

2. **Finding Intensity Gradient of the image:** After applying the noise reduction filter to the image, the smoothened image is then filtered with a Sobel kernel in both the horizontal and vertical direction to obtain the first derivative in the horizontal direction (G_x) and the vertical direction (G_y). After determining the first derivative in both the horizontal and vertical directions, the edge gradient magnitude and direction for each pixel can be found as follows:

$$\text{Edge Gradient} = \sqrt{G_x^2 + G_y^2} \tag{4.1}$$

$$\text{Angle} = tan^{-1}\left(\frac{G_x}{G_y}\right) \tag{4.2}$$

3. **Non-maximum Suppression:** After determining the magnitude and direction of the gradient, a complete scan of the image is performed to eliminate any undesired pixels that may not represent the edge. To do this, each pixel is examined to determine if it is a local maximum in its neighbourhood in the gradient's direction. The process of examining whether a pixel is a local maximum in its neighbourhood is demonstrated in Figure 4.1.

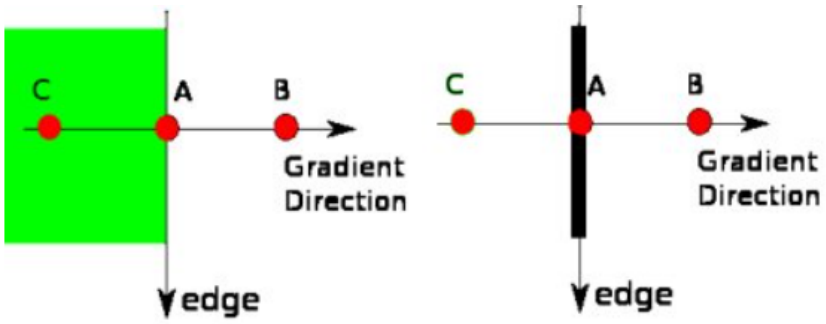


Figure 4.1: Process of non-maximum suppression image reproduced from [12].

Point A in Figure 4.1 is on the vertical edge, and the gradient direction is perpendicular to the edge. Points B and C are aligned along the gradient. To establish if point A is a local maximum, it is analysed in conjunction with points B and C. If point A is a local maximum, it is considered for the next step; else, it is suppressed.

4. **Hysteresis Thresholding:** This step decides which edges are indeed edges and which are not. This requires two threshold values, minVal and maxVal. Any edges with an intensity gradient more than maxVal are guaranteed to be edges, and those with an intensity gradient less than minVal are certain to be non-edges and should be eliminated. Edges and non-edges are categorized based on their connection if they fall within these two thresholds. If they are connected to "sure-edge" pixels, they are considered edges if this is not the case they are discarded. Figure 4.2 depicts this procedure.

   The edge A in Figure 4.2 is above the maxVal and is thus considered as a "sure-edge". Although edge C is less than maxVal, it is connected to edge A, it is therefore also regarded as an acceptable edge. Despite the fact that edge B is above minVal and is in the same region as edge C, it is not connected to any "sure-edges" and is thus discarded. This step also eliminates small pixel noises on the premise that edges are typically long lines.

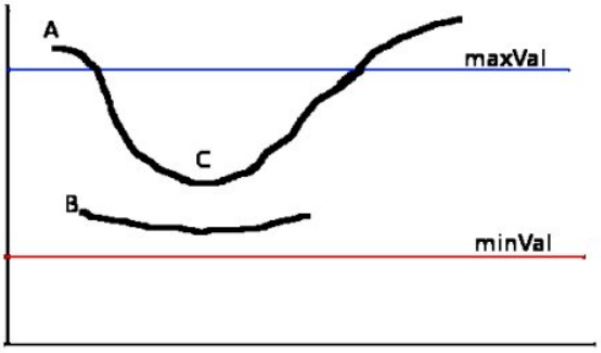


Figure 4.2: Hysteresis thresholding procedure image reproduced from [12]

4.2. Canny Edge Detector Implementation

The OpenCV library, which is a set of programming functions primarily targeted at real-time computer vision, was used to apply the Canny edge detector to images taken by a car-mounted camera. After applying the Canny edge detector to the images containing road surfaces the results as can be seen in Figure 4.3 were obtained.

Figure 4.3: Canny edge detector applied to an image taken by a car-mounted camera.

From Figure 4.3, it is observed that the Canny edge detector is able to effectively detect all sharp changes in the colour gradient of the image. These edges can be very helpful when trying to determine whether a superpixel is consists of road pixels since neighbouring superpixels that both consist of road pixels are not expected to have an edge between them, whereas neighbouring superpixels with one consisting of road pixels and the other consisting of pixels that do not belong to road are expected to have edges between. A further use of Canny edge detection is in identifying road lines for a self-driving car to remain in the correct lane, this is, however, not the current aim of the project but can be considered for an addition to the road detection software in the future.

Chapter 5

Edges Between Superpixels

The previous chapter discussed how the Canny edge detector functions and how to implement it. This chapter will discuss how the results obtained from the Canny edge detector can be applied to superpixels.

5.1. How Superpixels Can use the Canny Edge Detector Results

After applying the Canny edge detector to images taken by a car-mounted camera, a 2D matrix (the shape of the image) is received as an output. In the binary image output, zeros indicate regions with no edges and ones indicate pixels that are edges in the image. This output does not yet help to infer which superpixels consist of road, since it provides no information regarding superpixels.

To make use of the information acquired from the Canny edge detector, relationships between this information and superpixels must be identified. One typical characteristic of roads that was used is the relationship between superpixels and their neighbours; this relationship is that two neighbouring superpixels that both consist of road have a low probability of having an edge between them. In contrast road superpixels typically have an edge between themselves and neighbouring non-road superpixels. Non-road superpixels that are expected to have a slightly higher probability of having an edge between themselves and other neighbouring also non-road superpixels. This is because there are sharp changes in colour space intensity in the regions that are not road whenever there is a change in the surrounding environment, such as the change from ground to sky or trees to sand. The probability of there being an edge between two non-road superpixels is, however, not as high as the probability of there being an edge between road and non-road superpixels.

5.2. Applying the Canny Edge Detector Results to Superpixels

To determine if there is an edge between a superpixel and one of its neighbours, the first step is to determine the neighbours of each superpixel. To do this it is useful to remember how superpixels are represented in computer vision as shown in Figure 2.1 in Chapter 2. The neighbours of each superpixel were found by looping through all pixels in the segmented image and determining firstly if a pixel corresponds to the superpixel we attempting to identify the neighbours of. Secondly, if the pixel does correspond determining whether the neighbours of that pixel also correspond to the same superpixel. If any of the neighbouring pixels do not correspond, then the pixels that did not correspond were added to a list containing all the pixels that meet the same requirements for each superpixel. Once the list has been created, we are left with the numbers of each neighbouring superpixel, however, the numbers representing each neighbour are repeated multiple times. We thus need to remove all the repeated numbers, this can easily be done in Python by using the numpy.unique [13] function found in the NumPy library which adds support for dealing with large arrays and matrices. Figure 5.1 is a basic example that demonstrates this process. In this example, each superpixel consists of only one pixel to simplify things. Green represents the superpixel that we are trying to determine the neighbours of, while red is used to indicate the pixels that are being analysed to establish whether they belong to the superpixel shown in green. In the case of Figure 5.1, pixels 2 and 4 are determined as the neighbours of superpixel 1, as they do not belong to the same superpixel as pixel 1. Diagonal pixels are not considered to be neighbours since there cannot be an edge between diagonal pixels.

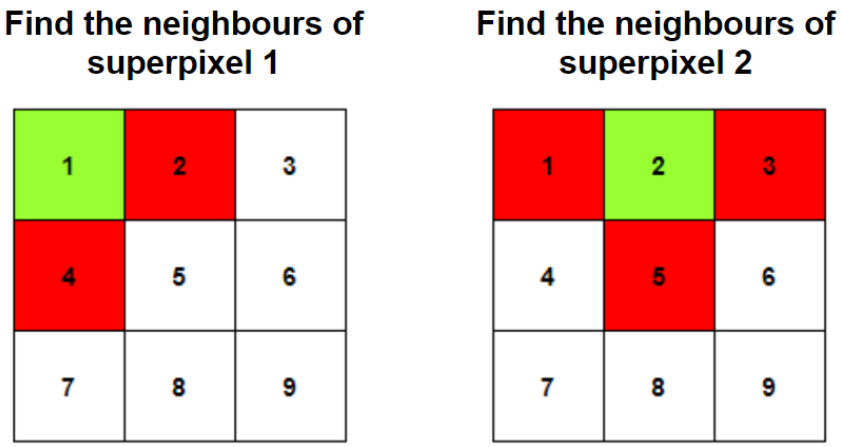


Figure 5.1: Example showing which pixels are checked to determine whether they belong to the same superpixel. Green indicates the pixel we are currently looking at, while red indicates the neighbours of that pixel.

The second step is to determine whether there is an edge between a superpixel and each of

its neighbours. This was done by creating a list containing all boundary pixels between a superpixel and each of its neighbours. The boundary pixels are determined using the same method as in the previous step. The boundary pixels were then analysed to determine whether they are edge pixels according to the Canny edge detector. It is not expected that all the pixels between a superpixel and one of its neighbours will be edge pixels; thus, a threshold for the number of boundary pixels required to be edge pixels before determining that there is an edge between two superpixels was chosen. Since there is also a small number of edge pixels between all superpixels, we need to ensure that the threshold is high enough such that it prevents these edge pixels from causing a false edge between superpixels. When deciding on the threshold, some important things to consider are the resolution of the image and the number of superpixels. A higher threshold is required for a higher resolution image since more pixels are required to make a visible edge. In contrast a lower threshold is required for an image with more superpixels because the more superpixels there are in an image, the smaller the superpixels become, and thus fewer pixels are required to make an edge between superpixels.

5.3. Evaluation and Conclusions

Figure 5.2 depicts superpixels that do not have an edge between themselves and any of their neighbours; this is used to ensure that the method of determining whether there is an edge between superpixels works. If the chosen relationship is correct, the majority of superpixels obtained is expected to be superpixels made up of road pixels, as they are the least likely to have an edge between themselves and their neighbours. This relationship is valid, as the majority of superpixels depicted are indeed made up of road pixels, as can be observed in Figure 5.2.

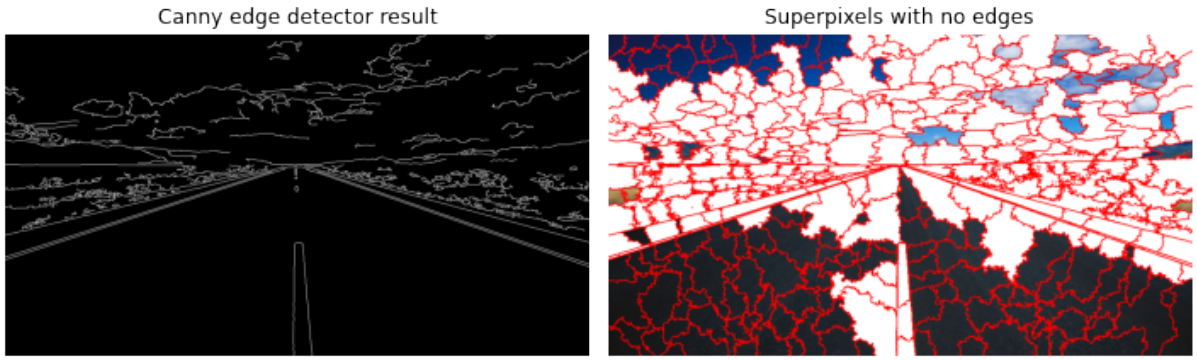


Figure 5.2: Superpixels with no edges between themselves and any of their neighbours shown in colour.

Chapter 6

Probabilistic Graphical Models

This chapter introduces probabilistic graphical models (PGMs), which will be used in subsequent chapters as a technique to combine the characteristics obtained in the previous chapters to infer whether a superpixel consists of road pixels. This chapter is largely based on the work done by Barber [14], as well as Koller and Friedman [15].

6.1. Overview

Probabilistic graphical models are statistical models that are used to encode complex joint multivariate probability distributions, allowing them to capture conditional independence relationships between interacting random variables. This makes PGMs a very powerful tool when trying to model systems with uncertain characteristics and relationships in a principled manner. By knowing the graphical structure of a PGM, many tasks can be solved with inference by computing the marginal distribution of one or more random variables. Using a PGM for inference requires two steps, firstly specifying the model of a system in terms of a graphical structure and secondly converting the model to another type of PGM, in which inference is performed. Two of the main types of graphical structures used for modelling systems with uncertain relationships are Bayesian networks (BNs) and Markov random fields (MRFs).

6.2. Bayesian Networks

Bayesian networks are directed acyclic graphs (DAGs) that represent a set of random variables and their conditional dependencies. Random variables in a BN may be latent, unknown parameters, or observable parameters. Latent random variables are random variables that cannot be directly observed but are rather inferred from other parameters that have been observed. The nodes in a BN represent the random variables and the directed edges represented by arrows to illustrate the conditional dependence between the two random variables, nodes that are not connected represent conditionally independent variables. Figure 6.1 shows a basic example of a BN.

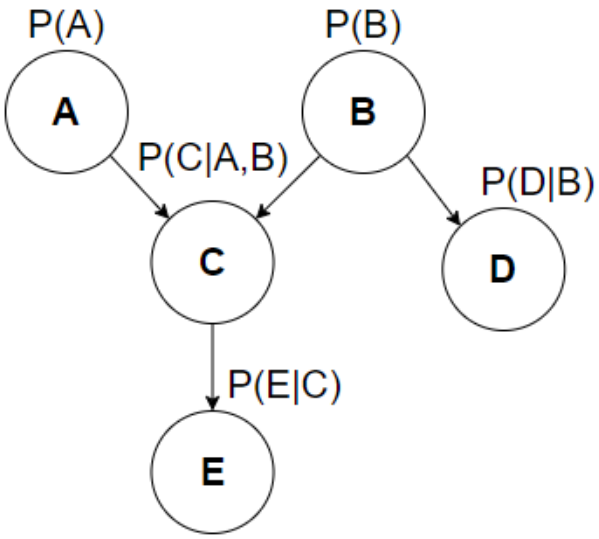


Figure 6.1: Example of a Bayesian network.

In Figure 6.1 nodes A, B, and C are the ancestors of node E, or conversely, node E is the descendant of nodes A, B, and C. Ancestor nodes that are connected through a single edge are called parents, node C is therefore the parent of node E, conversely E is the child of node C. In a Bayesian network, every node is associated with the conditional probability distribution $p(x|\text{pa}(x))$, where x refers to the node being observed, and $\text{pa}(x)$ refers to the parents of node x.

To determine the joint probability distribution over all the random variables in a BN the chain rule of probability can be used. Thus, for the BN in Figure 6.1, the joint probability distribution $p(A, B, C, D, E)$ is determined as

$$p(A, B, C, D, E) = p(E|A, B, C, D)p(D|A, B, C)p(C|A, B)p(B|A)p(A). \tag{6.1}$$

From the BN we know that random variables not connected via an edge are conditionally independent, thus the joint probability distribution can be simplified as

$$p(A, B, C, D, E) = p(E|C)p(D|B)p(C|A, B)p(B)p(A). \tag{6.2}$$

To find the marginal distribution over a subset of random variables in the BN, we need to integrate over all other variables, for discrete random variables integration is replaced by summation. The marginal distribution over the random variables A and B can be found as

$$\int_{C=\ \infty}^{\infty} \int_{D=-\infty}^{\infty} \int_{E=-\infty}^{\infty} p(A, B, C, D, E)\ dC\ dD\ dE. \tag{6.3}$$

Another important concept is that of a factor, denoted by phi(x). A factor is a function of random variables, these random variables are known as the scope of the factor. The conditional probability distribution for each node may be thought of as a factor, which is a helpful generalisation.

$$\phi_x(x, \text{pa}(x)) = p(x|\text{pa}(x)) \tag{6.4}$$

where its scope is defined as

$$\text{Scope}(\phi_x) = \{x, \text{pa}(x)\} \tag{6.5}$$

We can now describe a system using a BN to represent the probabilistic relationships between a set of random variables. This BN may then be converted into another type of PGM, in which inference can be performed.

6.3. Markov Random Fields

Markov random fields are undirected graphical models, MRFs are similar to BNs in their representation of dependencies, however, BNs are directed and acyclic whereas MRFs are undirected and may be cyclic. MRFs are frequently used to model tasks in image processing and computer vision because models in image processing and computer vision generally have non-casual or poorly defined relationships. The nodes in a MRF represent random variables, while edges between the nodes specify factors. An example of a MRF is shown in Figure 6.2 with each edge representing dependency.

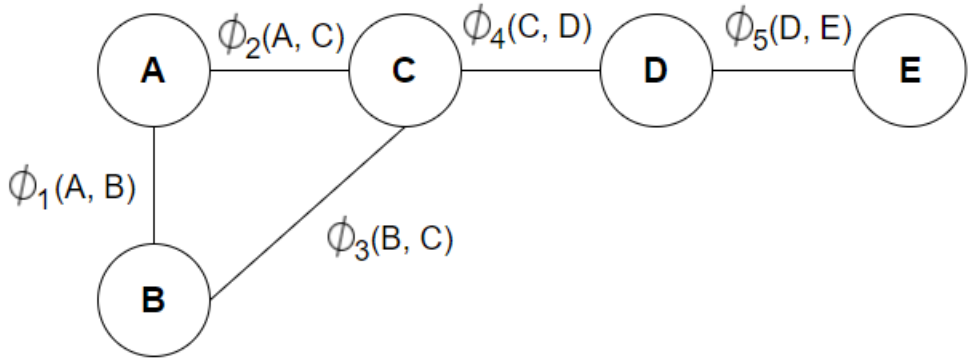


Figure 6.2: Example of a Markov random field.

The joint probability distribution $p(A, B, C, D, E)$ can be found by using the unnormalised measure $\tilde{p}(A, B, C, D, E)$. The unnormalised measure $\tilde{p}(A, B, C, D, E)$ is found by using product of factors

$$\tilde{p}(A, B, C, D, E) = \phi_1(A, B) \times \phi_2(A, C) \times \phi_3(B, C) \times \phi_4(C, D) \times \phi_5(D, E). \tag{6.6}$$

The unnormalised measure can then be converted to the joint probability distribution $p(A, B, C, D, E)$ by normalising it. To normalise the unnormalised measure the partition function is used

$$p(A, B, C, D, E) = \frac{1}{Z}\tilde{p}(A, B, C, D, E), \tag{6.7}$$

where $\frac{1}{Z}$ is the partition function. With this knowledge, we may now describe systems with undirected and perhaps cyclic relationships.

6.4. Performing inference on Models described by Bayesian Networks

We will only be looking at performing inference on models described by BNs since for this project a BN was chosen to model the typical road characteristics found in images. To perform inference on a model described by a BN, the BN is converted into a cluster graph. A cluster graph is an undirected network with nodes representing variable subsets known as clusters. The clusters for a BN representing the random variables $X_1, ..., X_n$ are indicated as

$$C_i \subseteq \{X_1, ..., X_n\}. \tag{6.8}$$

A sepset denoted by $S_{i,j}$ is a subset of variables between clusters C_i and C_j that are in the intersection of the scopes of both nodes.

$$S_{i,j} \subseteq (\text{Scope}(C_i) \cap \text{Scope}(C_j)). \tag{6.9}$$

In Figure 6.3 the conditional probabilities for the example BN shown in Figure 6.1 have been replaced by factors and the cluster divisions have been illustrated. The cluster divisions shown in Figure 6.3 is only one of many possible arrangements. The cluster graph for the BN shown in Figure 6.3 is depicted in Figure 6.4. Once a cluster graph has been set up it can be used for inference, by allowing clusters to share their beliefs regarding common variables with other clusters. The process of clusters sharing their beliefs regarding shared random variables is known as message passing, for this project the message passing algorithm used is called belief propagation also referred to as loopy belief propagation. Messages are passed by calculating the marginal distribution for each unobserved random variable that is conditional on observed variables. Messages do not change from one iteration to the other, they are passed repeatedly until they converge. The converged messages are then used to calculate beliefs.

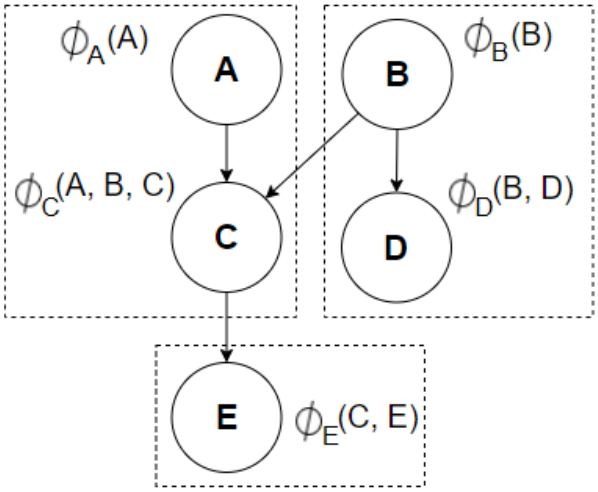


Figure 6.3: Bayesian network with cluster divisions shown and conditional probabilities replaced by factors.

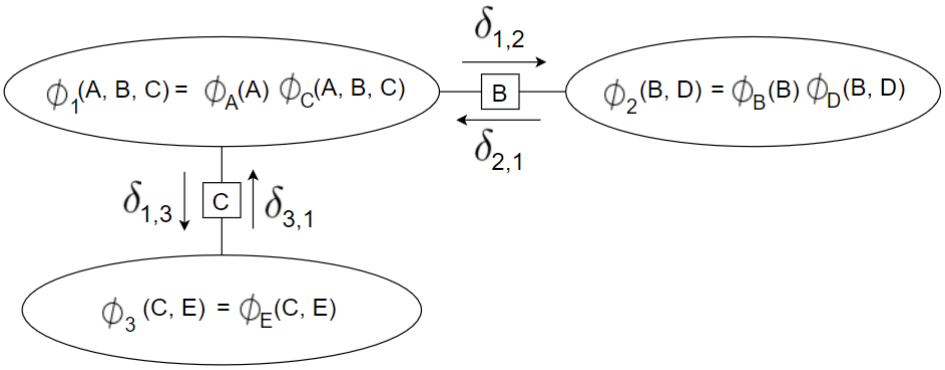


Figure 6.4: Bayesian network converted into a cluster graph showing cluster potentialis (elliptical nodes) and sepset messages (rectangle nodes).

6.5. Chapter Summary

Bayesian networks represent random variables with causal relationships, whereas Markov random fields are commonly used to model random variables with non-causal relationships since they allow for more flexible relationships. A Bayesian network representing a set of random variables may be converted to a cluster graph by organizing the conditional probability distributions into clusters, which can then be used for inference. After observing random variable message passing is used to allow clusters to share their beliefs about shared variables with nearby clusters, this is used to infer the posterior belief of latent random variables.

Chapter 7

Road Prediction Using a PGM

In the previous chapters, we have extracted features from images taken by a car-mounted camera to predict where the road is in an image; this chapter discusses how a PGM can use those predictions to infer which superpixels in an image consist of road pixels. In this chapter the PGMs were encoded using the EMDW library [16]. The process followed will be outlined and finally conclusions will be made on the accuracy of the PGM.

7.1. Setting Up a PGM for the Logistic Regression Predictions

The PGM was created using a Bayesian network. This decision was made due to the causal relationship between the road and the results obtained in the previous chapters. This causal relationship is that if a superpixel consists of road pixels, the logistic regression model has a higher probability of predicting that the superpixel is a road superpixel. The Bayesian network depicted in Figure 7.1 is one possible solution for configuring a Bayesian network to infer whether a superpixel consists of road pixels based on the results from the logistic regression model. In Figure 7.1 the random variable R is used to denote whether a superpixel consists of road pixels. $p(R)$ was obtained by counting the number of superpixels that consist of road pixels in images taken via a car-mounted camera and dividing it by the total number of superpixels in the images. R is our latent random variable and can thus never be directly observed but can only be inferred from other observed variables. The random variable LR is used to denote whether a superpixel consists of road pixels according to the logistic regression model and will be our observed parameter. In the example depicted in Figure 7.1, LR can only be observed as either road (1) or not road (0), which is achieved by thresholding the logistic regression model output. If the logistic regression model output was more than or equal to 0.5, the output was set to 1 and if the output was less than 0.5, the output was set to 0. This is, however, not ideal as it will cause the PGM to be either more or less certain about the logistic regression results than it should be. The threshold was used so that the basic PGM could be set up, the next section addresses how the probabilities from the logistic regression model can be used.

Figure 7.1: PGM used to infer whether a superpixel is road based on the observed logistic regression model predictions.

To perform inference on the BN shown in Figure 7.1, Bayes' theorem can be used, Bayes theorem states

$$p(R|LR) = \frac{p(LR|R)p(R)}{p(LR)}. \tag{7.1}$$

In Equation 7.1 $p(R|LR)$ is called the posterior distribution over R, $p(LR|R)$ is called the likelihood function of R, and $p(R)$ is called the prior probability distribution of R. Computing the posterior distribution over R is known as the inference problem. Solving the inference problem for the example shown in Figure 7.1 will help to ensure that the EMDW library is operating as expected and providing the same results as our calculations. The probability that a superpixel is road given that the logistic regression model predicts it to be road can thus be solved as follows:

$$\begin{aligned} p(R=1|LR=1) &= \frac{p(LR=1|R=1)p(R=1)}{p(LR=1|R=1)p(R=1) + p(LR=1|R=0)p(R=0)} \\ &= \frac{(0.9)(0.4)}{(0.9)(0.4) + (0.1)(0.6)} \\ &= 0.857 \end{aligned} \tag{7.2}$$

To implement the example PGM shown in Figure 7.1 using the EMDW library, two factors need to be set up, one factor for the random variable R, and the other for the probability of LR conditioned on R. A cluster graph is then set up so that the posterior distribution $p(R|LR)$ can be inferred by querying the cluster graph using the EMDW library. When the solution obtained in Equation 7.2 is compared to the results returned by the EMDW library, the results are identical, indicating that the EMDW library is operating as intended.

7.2. Beta distribution

As discussed in Chapter 3, the logistic regression model outputs the probability that a superpixel consists of road pixels. It is much more difficult to use these results since it requires there to be a probability $p(LR|R)$ associated with every probability LR observed.

Using a beta distribution is one method of associating a probability $p(LR|R)$ with every probability observed from logistic regression model LR. The beta distribution is a continuous probability distribution, and because it is frequently used model probabilities, its domain is bound between 0 and 1. The beta distribution is parameterised by two positive shape parameters represented by α and β. The shape parameters α and β appear as exponents of the random variable and control the shape of the distribution

$$\frac{x^{\alpha-1}(1-x)^{\beta-1}}{B(\alpha,\beta)} \tag{7.3}$$

where $B(\alpha,\beta) = \frac{\Gamma(\alpha)\Gamma(\beta)}{\Gamma(\alpha+\beta)}$ and Γ is the Gamma function [17].

There are few methods to determine the optimal values for the α and β parameters. Some of the methods considered are maximum likelihood estimation (MLE), maximum a posteriori (MAP) estimation and adjusting the two parameters by hand until the best results are achieved. MLE operates by maximising a likelihood function such that the observed data is most probable. MAP estimation operates in a similar manner to ML; however, it incorporates a prior distribution that quantifies the additional information available. Both MLE and MAP estimation require a lot of labelled data to be accurate. Despite the fact that MAP estimation performs better than MLE for small amounts of data due to the regularisation effect of the prior distribution, it still requires a fairly large amount of data to provide good results. It was thus decided to adjust the parameters by hand since MLE and MAP estimation are not that effective for small amounts of data. An example of how a beta distribution is used to obtain the probability distribution $p(LR|R)$ given the observed probability LR is depicted in Figure 7.2.

As shown in Figure 7.2, two beta distributions are required, one beta distribution is used to obtain the probability $p(LR|R=1)$, while the other is used to obtain the probability $p(LR|R=0)$. The horizontal axis is the probability that a superpixel consists of road pixels according to the logistic regression model, and the vertical axis represents the probability distribution $p(LR|R)$. For an observed logistic regression probability, for instance $LR = 0.6$, the horizontal axis value is mapped to the vertical axis of both beta distributions, and the related probabilities $p(LR=0.6|R=1)$ and $p(LR=0.6|R=0)$ are read off the vertical axis.

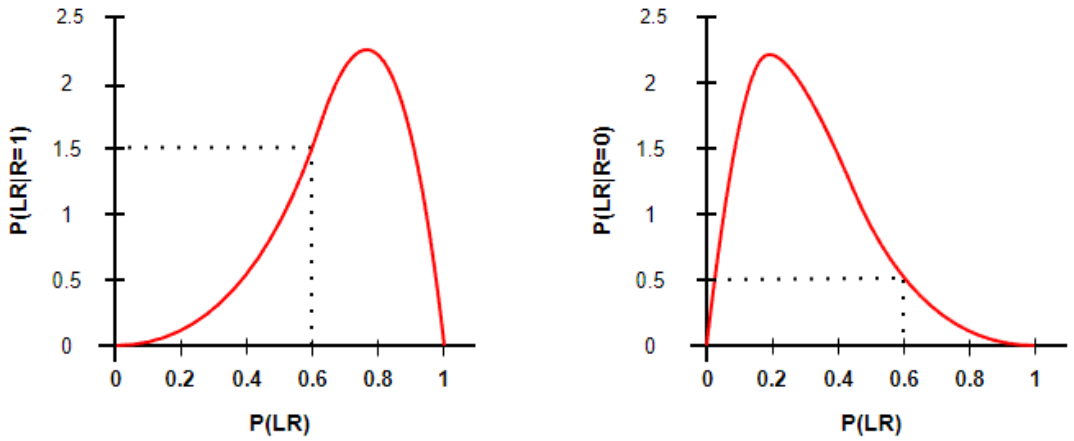


Figure 7.2: Example of how a beta distribution was used to obtain the probability $p(LR|R)$.

As can be seen in Figure 7.2, the values of $p(LR = 0.6|R = 1)$ and $p(LR = 0.6|R = 0)$ read off the y-axis do not sum to one and thus must be normalised before they can be used as probabilities for the PGM. The probabilities read from the beta distributions are normalised by summing the probabilities to obtain a normalising factor. The two probabilities are then divided by the normalising factor. This process is demonstrated in Equations 7.4 – 7.6.

$$\begin{aligned} \text{Normalising factor} &= p(LR|R = 1) + p(LR|R = 0) \\ &= 1.5 + 0.5 \\ &= 2 \end{aligned} \tag{7.4}$$

$$\begin{aligned} \text{Normalised } \phi_{LR}(LR = 0.6, R = 1) &= \frac{\phi_{LR}(LR = 0.6, R = 1)}{Normalising factor} \\ &= 1.5/2 \\ &= 0.75 \end{aligned} \tag{7.5}$$

$$\begin{aligned} \text{Normalised } \phi_{LR}(LR = 0.6, R = 0) &= \frac{\phi_{LR}(LR = 0.6, R = 0)}{Normalising factor} \\ &= 0.5/2 \\ &= 0.25 \end{aligned} \tag{7.6}$$

As can be observed in Equations X and X, the normalised probabilities obtained for $\phi(LR = 0.6, R = 1)$ and $\phi(LR = 0.6, R = 0)$ sum to one. The probability distribution is thus complete. The beta distribution was implemented in Python by using the

scipy.stats.beta.pdf [18] function found in the SciPy library, which is an open-source library that builds on the NumPy library and is often used to solve mathematical and engineering problems.

7.3. Results Obtained from the PGM for LR Observed

After setting up the *LR* factor in the EMDW library to include the probabilities obtained from the beta distributions, all superpixels that had a probability greater than 0.5 according to the EMDW library after inference were classified as road superpixels. Superpixels classified as road according to the PGM are shown in colour in Figure 7.3, while non-road superpixels are shown in white.

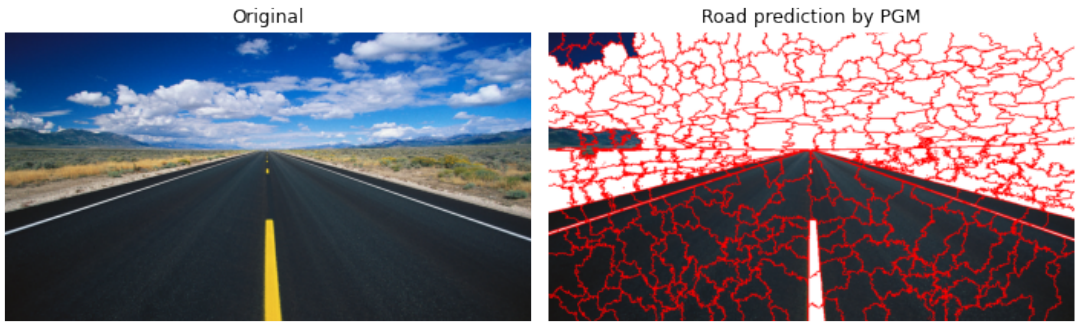


Figure 7.3: Superpixels with a probability greater than 0.5 of being road according to the PGM after observing the predictions from the logistic regression model. In the image shown on the right superpixels shown in colour are superpixels classified as road, whereas superpixels shown in white are non-road superpixels

When the results from the PGM are compared to the results from the logistic regression model as shown in Figure 7.4, the differences observed are slightly fewer superpixels being incorrectly labelled as road. This is to be expected, since the current PGM only lowers the probability that any given superpixel consists of road pixels. As a result, superpixels that previously had a probability of just above 0.5 of being a road superpixel will now have a probability of less than 0.5, causing it to be classified as a non-road superpixel, reducing the number of superpixels incorrectly classified. A comparison of the results obtained from the PGM, and the results obtained from logistic regression alone is shown in Figure 7.4.

The PGM prediction of whether a superpixel consists of road pixels can be further improved by labelling a superpixel as road only if the probability received from the PGM is greater than 0.7. This is much better since we do not want to be only 50% certain when classifying a superpixel as a road, because if multiple superpixels are incorrectly classified it could potentially cause self-driving cars to have an accident. The result after only labeling superpixels with a probability greater than 0.7 of being road as road superpixels is shown in Figure 7.5.

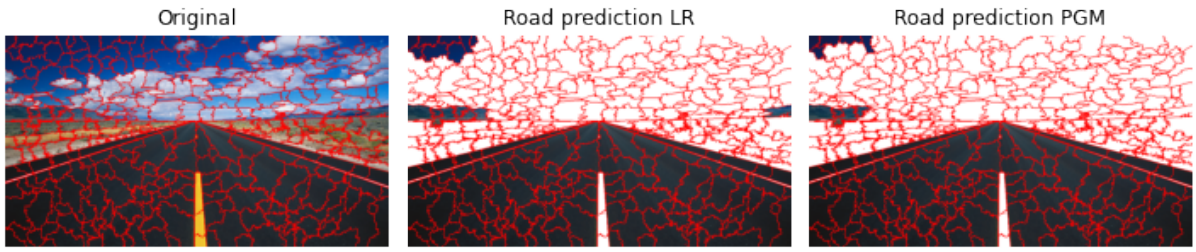


Figure 7.4: Comparison of superpixels predicted to be road shown in colour by the logistic regression model to the superpixels predicted to be road by the PGM.

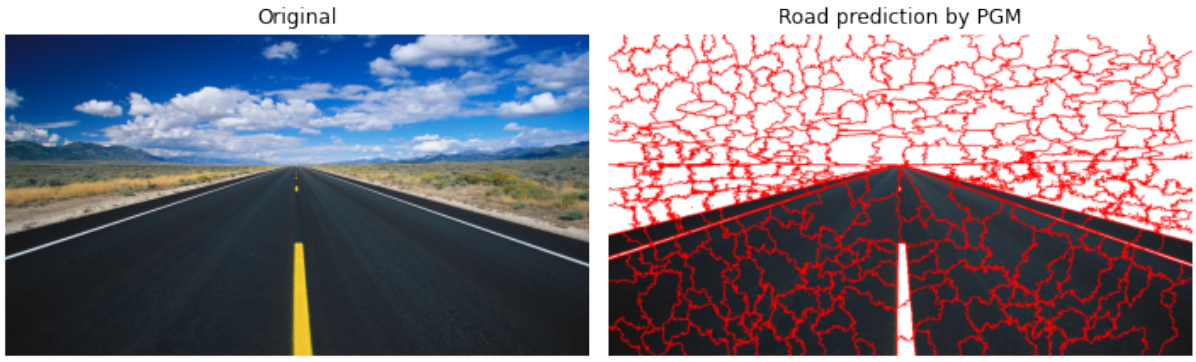


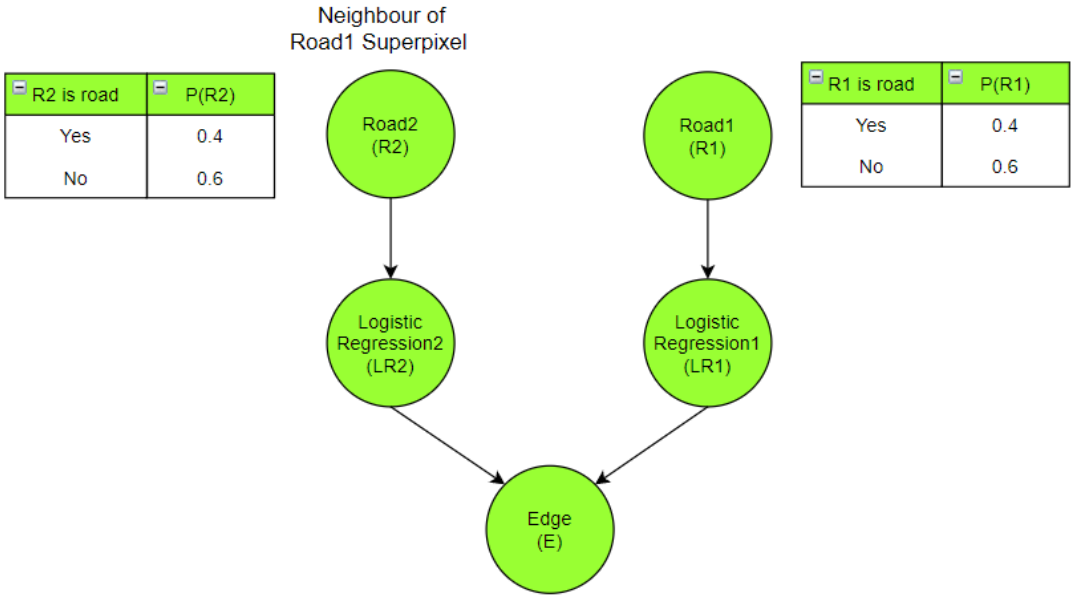
Figure 7.5: Superpixels with a probability greater than 0.7 of being road according to the PGM shown.

7.4. Extending the PGM to Include the Information about Edges Between Superpixels

Once the base PGM has been set up, it can easily be extended to include the information obtained regarding the edges between superpixels and their neighbours. The extended PGM can be observed in Figure 7.6. In Figure 7.6, the random variable $R1$ represents the superpixel that is currently being classified, while the random variable $R2$ is used to denote a neighbouring superpixel of $R1$. The random variable $LR2$ is the probability that $R2$ is a road superpixel according to logistic regression and the random variable E is the probability that there is an edge between the superpixel being classified and its neighbour $R2$.

The first step to extending the PGM in the EMDW library is to set up the three additional factors, one for the random variable $R2$, the second factor for the random variable LR, and the third for the random variable E. The factor for $R2$ is the same as the factor for $R1$, since the prior probability that any superpixel consists of road pixels remains the same. The factor $LR2$ is set up following the same process as the setting up of the factor for $LR1$. The probabilities for the edge factor E were obtained by analysing the superpixels in images and calculating the probability of there being an

edge between two road superpixels, one road and one non-road superpixel, and finally two non-road superpixels. After implementing the three new factors, the second step is to observe whether there is an edge between a superpixel and each of its neighbours. These observations are obtained from the results in Chapter 5. Since each superpixel has multiple neighbours the inferred probability of whether each superpixel is road needs to be obtained for each of its neighbours. After obtaining the inferred probabilities for each of a superpixel's neighbours these probabilities were averaged, this average was then used as the final probability that a superpixel consists of road pixels.



R2 is road	P(R2)
Yes	0.4
No	0.6

R1 is road	P(R1)
Yes	0.4
No	0.6

R1 is road	R2 is road	P(E=0|LR1,LR2)	P(E=1|LR1,LR2)
Yes	Yes	0.9	0.1
Yes	No	0.1	0.9
No	Yes	0.1	0.9
No	No	0.6	0.4

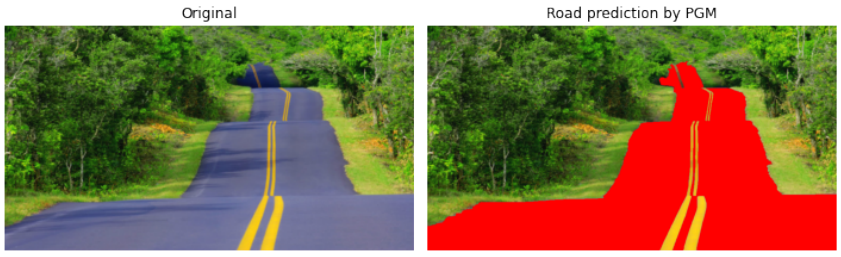
Figure 7.6: PGM extended to include the information regarding edges between superpixels.

7.5. Results Obtained From the PGM for LR and E Observed

The inferred probabilities of whether each superpixel consists of road pixels according to the extended PGM were added to a list in the EMDW library. The list was then converted to a CSV file so that superpixels with a probability greater than 0.7 of being road could be highlighted in Python. Figure 7.7 shows a comparison of the PGM prediction before and after adding the information regarding edges between superpixels the PGM predictions of

where the road is, are highlighted in red. The predictions of where the road is made by the PGM for other images are shown in Figure 7.8. As can be seen in Figure 7.8c, there are some road segments not identified as road. This is primarily due to the superpixels including parts of the brown field in the road segment. Overall, the PGM performs very well and can almost always predict where the road is in images taken by a car-mounted camera. The PGM will become more accurate as more relationships are added to it since there will be more information available to allow it to correctly infer the label of each superpixel.

Figure 7.7: Comparison of the PGM prediction of where the road is before and after adding the information regarding edges between superpixels.



(a)

(b)

(c)

Figure 7.8: PGM predictions of where the road is in images highlighted in red.

Chapter 8

Road Detection PGM Extensions

This chapter discusses potential extensions or improvements that were considered to obtain more reliable road predictions from the probabilistic graphical model but were not implemented due to time constraints. This chapter is largely based on how I would go about improving or extending the existing PGM and may not make the prediction made by the PGM more accurate in practice.

8.1. Possible Improvements to Existing PGM

The existing PGM can be improved by pre-processing the images; the images used in this report have had very minimal pre-processing. By pre-processing the images by using the HSV colour space for example, the predictions may become much more reliable. As mentioned in Chapter 3, a neural network is expected to perform better than a logistic regression model; however, it requires a large amount of training data to be accurate and demands significantly more resources and time to train. Much better results may be produced if the resources are available to train a neural network instead of using a logistic regression model. Logistic regression tries to find a point that divides two classes, this causes logistic regression to perform much worse than a neural network in cases where the two classes share similar characteristics. In Chapter 5 the method used to determine whether there is an edge between a superpixel and each of its neighbours is quite inefficient. A less computationally expensive method can be used.

8.2. Possible Extensions to Existing PGM

This section discusses the some of the other characteristics of road found in images taken by a car-mounted camera that were not used in this project. These characteristics can perhaps be used in the future to extend the existing PGM to improve the accuracy of the predictions.

8.2.1. Typical Road Texture Characteristics

In images taken by a car-mounted camera, it is expected that all of the regions in the image that are road will have a similar texture, and therefore superpixels consisting of road pixels will have a similar texture. To find the texture in an image, it is best to first convert the image to grayscale so that each pixel in the image has only one intensity value. The texture of each superpixel can then be found by creating a gray-level co-occurrence matrix (GLCM), which determines the different combination of gray levels found in an image [19]. This is done by analysing each pixel and comparing its grayscale value with the grayscale value of nearby pixels. If there is a minor difference in grayscale values between nearby pixels, the texture is smooth; however, frequent changes in grayscale values in neighbouring pixels suggest that the region of the image has a rough texture. Figure 8.1 shows how different textures appear in a grayscale image.

Figure 8.1: Example of how different textures appear in grayscale images

Once the texture of each superpixel has been obtained, it can be used to train either a logistic regression model following a similar procedure to that described in Chapter 3, or a neural network. A neural network is expected to provide better predictions since typical texture values are often relatively similar, making it difficult for a logistic regression model to find a point that separates the two classes. A neural network does, however, require much more resources and time to train. The predictions from the either the logistic regression model or the neural network can then be used to extend the PGM following the same procedure described in Chapter 7 for adding the logistic regression model results. Figure 8.2 depicts how the texture results can be used to extend the PGM.

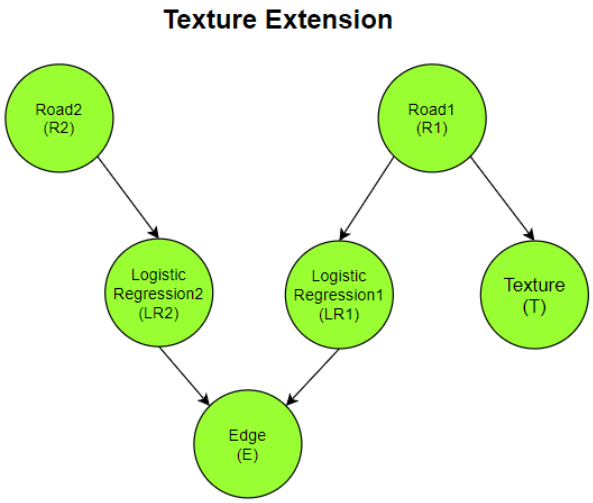


Figure 8.2: How the existing PGM can be extended to include road texture.

8.2.2. Typical Road Width Characteristics

Another characteristic typical to road found in images taken via a car-mounted is that the road appears wider at the bottom of the image and becomes narrower higher up in the image. This characteristic is very difficult to make use of as it requires a relationship between all the superpixels that have been predicted to be road. A possible solution to this would be to analyse all the superpixels that are on the road border and determine if they are getting closer as you move to the top of the image. To determine whether a superpixel is on the road border, one can check if that superpixel has been predicted to be road and has an edge between itself and any of its neighbours, if there is an edge, the road superpixel is most likely road border. This is because road superpixels in the centre of the road will be surrounded by other road superpixels and therefore will not have an edge between itself and any of its neighbours. Once all the superpixels that are on the edge of the road have been found, the centre of the pixels that are edge pixels according to the Canny edge detector can be found for each superpixel. This is indicated in green in Figure 8.3. The points indicated in green in Figure 8.3 can now be analysed to determine whether they are moving towards the centre of the image the higher up in the image they are.

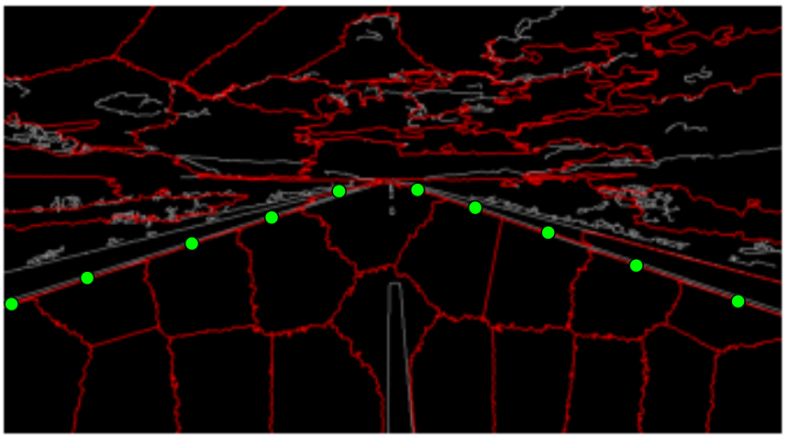


Figure 8.3: Centre of edge pixels for superpixels on the edge of the road indicated in green.

8.2.3. Expected Neighbours of a Road Superpixel

A further extension that may be made to the PGM is to is to reduce the probability that superpixels that do not have any neighbouring superpixels predicted to be road are road. It is expected that every road superpixel will have at least one neighbouring superpixel that is also road. This should be quite simple to implement since we have already obtained the neighbours of each superpixel in Chapter 5. All that is required now is to determine whether at least one of those neighbouring superpixel has been predicted to be road. If this is not the case, the superpixel will have a very low probability of being a road superpixel.

8.2.4. Expected Location of a Road Superpixel

Road superpixels are typically expected to be in the bottom half of an image taken by a car-mounted camera, whereas a superpixel in the top half of the image is typically not expected to be a road superpixel. Figure 8.4 shows an example of the regions that are typically expected to be road in images taken by a car-mounted camera.

Figure 8.4: Regions of an image taken by a car-mounted camera expected to be road highlighted in green and regions not typically expected to be road highlighted in red.

Chapter 9

Conclusion

The aim of this project was to identify which regions of images taken via a car-mounted camera are road. This report looked at the use of a probabilistic graphical model to combine the uncertain characteristics typical to road to infer where the road is in images. To achieve this, several characteristics typical to a road surface were considered. In this report the primary characteristics used were typical road colour and edges between road and non-road superpixels. A logistic regression model was trained in order to predict whether a superpixel consists of road pixels based on the RGB values of each superpixel. The results from the logistic regression model were then analysed to determine the accuracy of the model to ensure that no additional training was required. The PGM was then set up to use the results obtained from the logistic regression model to infer whether a superpixel was either a road or a non-road superpixel. Information regarding edges between superpixels and each of their neighbours was obtained. The PGM was then extended to incorporate this new information. Extending the PGM resulted in more accurate predictions, since there was more information available to allow the PGM to correctly infer the label of each superpixel. The report then discussed some of the possible improvements and extensions that can be made to the existing PGM to obtain more reliable predictions.

In conclusion the PGM was able to accurately infer where the road is in images taken via a car-mounted camera. There is still, however, a lot of room for improvement, this report only looked at identifying where tarred roads are in images taken by a car-mounted camera. The existing PGM can easily be extended to infer where the road is in images containing dirt roads.

9.1. Summary of New Concepts Learnt During this Project

This section gives a very summary of the new concepts learnt that were essential to the completion of this project. A list of the new concepts learnt is provided below.

- The data format and data structure of images, as well as how images are represented in computer vision.
- What superpixels are and how they can be used to group pixels that have similar characteristics, hence reducing the processing power necessary to identify which regions of an image share common characteristics.
- How logistic regression works and how to train a logistic regression model for binary classification.
- What an "edge" in an image is and how to obtain the edges in an image.
- What a probabilistic graphical model is and how it can be used to combine uncertain characteristics and relationships in a principled way.

Bibliography

[1] B. Widmer, "50 car accident statistics (sept. 2021) auto accident deaths," Sep 2021. [Online]. Available: https://www.thewanderingrv.com/car-accident-statistics/

[2] 08/02/2021, N. S. zealous learner aspiring to advance in the domain of AI/ML. Eager to grasp emerging techniques to get insights from data, hence explore realistic Data Science applications as well., and N. Shiledarbaxi, "Semantic vs instance vs panoptic: Which image segmentation technique to choose," Apr 2021. [Online]. Available: https://analyticsindiamag.com/semantic-vs-instance-vs-panoptic-which-image-segmentation-technique-to-choose/

[3] "Improving learning effectiveness for object detection and classification in cluttered backgrounds - scientific figure on researchgate." Sep 2021. [Online]. Available: https://www.researchgate.net/figure/Semantic-segmentation-left-and-Instance-segmentation-right-8_fig1_339328277

[4] "Module: segmentation." [Online]. Available: https://scikit-image.org/docs/dev/api/skimage.segmentation.html#skimage.segmentation.felzenszwalb

[5] "Module: segmentation." [Online]. Available: https://scikit-image.org/docs/dev/api/skimage.segmentation.html#skimage.segmentation.quickshift

[6] "Module: segmentation." [Online]. Available: https://scikit-image.org/docs/dev/api/skimage.segmentation.html#skimage.segmentation.slic

[7] "Module: segmentation." [Online]. Available: https://scikit-image.org/docs/dev/api/skimage.segmentation.html#skimage.segmentation.watershed

[8] A. Alorf, "K-means, mean shift, and slic clustering algorithms: A comparison of performance in color-based skin segmentation," Sep 2017. [Online]. Available: http://d-scholarship.pitt.edu/32379/

[9] "Scikit-image library." [Online]. Available: https://scikit-image.org/

[10] "Getting started with xlsxwriter." [Online]. Available: https://xlsxwriter.readthedocs.io/getting_started.html

[11] "Canny edge detector," Sep 2021. [Online]. Available: https://en.wikipedia.org/wiki/Canny_edge_detector

[12] "Canny edge detection." [Online]. Available: https://docs.opencv.org/3.4.15/d7/de1/tutorial_js_canny.html

[13] "numpy.unique." [Online]. Available: https://numpy.org/doc/stable/reference/generated/numpy.unique.html

[14] D. Barber, "Bayesian reasoning and machine learning," *Cambridge University Press*, 2012.

[15] D. Koller and N. Friedman, "Probabilistic graphical models : principles and techniques." *The MIT Press*, 2009.

[16] J. A. du Preez, *EMDW: A Pragmatic Approach to PGMs.*

[17] "Gamma function," Oct 2021. [Online]. Available: https://en.wikipedia.org/wiki/Gamma_function

[18] "scipy.stats.beta." [Online]. Available: https://docs.scipy.org/doc/scipy/reference/generated/scipy.stats.beta.html

[19] "Glcm texture features." [Online]. Available: https://scikit-image.org/docs/dev/auto_examples/features_detection/plot_glcm.html

Appendix A

Project Planning Schedule

Week	Starting Date	Description
1	05/07/2021	Literature study of probabilistic graphical models.
2	12/07/2021	Literature study of probabilistic graphical models.
3	19/07/2021	Literature study of probabilistic graphical models and practice using EMDW software.
4	26/07/2021	Literature study of Canny edge detector.
5	02/08/2021	Literature study superpixel algorithms.
6	09/08/2021	Implementation of superpixel algorithms.
7	16/08/2021	Implementation of superpixel algorithms.
8	23/08/2021	Literature study of logistic regression.
9	30/08/2021	Implementation of a logistic regression model to predict whether a superpixel consists of road pixels.
10	6/09/2021	Implementation of a logistic regression model to predict whether a superpixel consists of road pixels.
11	13/09/2021	Determine the neighbours of each superpixel.
12	20/09/2021	Determine the neighbours of each superpixel.
13	27/09/2021	Implementation of the Canny edge detector.
14	04/10/2021	Determine whether there is an edge between superpixels according to the Canny edge detector.
15	11/10/2021	Use a graphical structure to model the relationships found to infer whether a superpixel consists of road pixels.
16	18/10/2021	Use the EMDW library to encode the graphical model created to infer whether a superpixel consists of road pixels.
17	01/11/2021	Report writing and editing to incorporate study leader feedback (report writing is a continuous process this is for pure report writing and editing).
18	08/11/2021	Report writing and editing to incorporate study leader feedback (report writing is a continuous process this is for pure report writing and editing).
19	15/07/2021	Report finalisation for hand in

Table A.1: The breakdown of the project schedule per week.

Appendix B

Outcomes Compliance

Table B.1: ECSA Outcomes Compliance

Outcome	Description of outcome in report	Chapters
ELO 1: Problem Solving (identify. assess, formulate and solve convergent and divergent engineering problems)	Identifying the road detection problem, solving the road detection problem using superpixels, logistic regression, edges between superpixels, and a PGM to combine all the uncertain characteristics.	2, 3, 5, 7
ELO 2: Application of scientific and engineering knowledge.	Analysis, modelling of a PGM using typical road characteristics obtained.	2, 3, 5, 6, 7, 8
ELO 3: Engineering design.	Identifying the road detection problem and designing as graphical structure to model the system using a Bayesian network.	2, 3, 5, 7
ELO 4: Investigations, experiments and data analysis	Superpixel algorithms were tested to determine which would work best for the road detection problem. Logistic regression model was tested to ensure that the training was sufficient and the results were accurate. PGM was tested to ensure the predictions of where the road is in images taken by a car-mounted camera were accurate.	2, 3, 5, 7
ELO 5: Engineering methods, skills and tools, including information and technology.	Using Python and C++ to program solutions to the road detection problem. Analysing all the results to ensure the they are what is expected.	2,3,4,5,7
ELO 6: Professional and technical communication.	Structure, style and language used throughout this report. Use of figures, diagrams and equations to support explanations and make it make it easier for a reader to understand.	All chapters
ELO 8: Individual work	The work done was performed individually and only had guidance form a study leader.	All chapters
ELO 9: Independent learning ability.	Aside from the guidance from the study leader where necessary all concepts and implementations were learnt independently through extensive research.	All chapters