A Novel Method for Continuity Based Edge Detection

Adam K. Schmidt

Supervised by Kenneth Dawson-Howe

A dissertation submitted to the University of Dublin, Trinity College, in partial fulfilment of the requirements for the degree of M.A.I. (St.) in Computer Engineering

Submitted to the University of Dublin, Trinity College, May, 2017

Declaration

I, Adam K. Schmidt, declare that the following dissertation, except where otherwise stated, is entirely my own work; that it has not previously been submitted as an exercise for a degree, either in Trinity College Dublin, or in any other University; and that the library may lend or copy it or any part thereof on request.

Signature: _____

Date: _____

Acknowledgements

I would like to thank my supervisor, Kenneth Dawson-Howe, for his guidance throughout this project.

I would also like to thank my parents, Valerie and Jörg, my brother Davin, and my girlfriend Hannah for their support and encouragement this year.

Abstract

The goal of this work is to use edge continuity metrics to improve the detection of salient edges in images. It is argued that continuity can act as a proxy for saliency in broad classes of images, especially those involving the extraction of human-made artefacts from natural scenes. Multiple interrelated methods of determining directional edge continuity and strength are described and evaluated. The final system is subjectively evaluated and compared to a number of state-of-the-art edge detectors. The results show that the system performs well on its stated goals of extracting continuous smooth detail from images while suppressing textural noise. Further, edge directionality is found to be a valuable metric for reducing noise in detected edges.

Contents

Chapter 1

Introduction

1.1 Motivation & Scope

This dissertation aims to investigate and evaluate the viability of a local-filter and continuity based approaches to edge detection. Though significant amounts of published work exists on the topic of suppression of textural elements by an explicit analysis of texture [18], little work has been done on large scale filter approaches to edge continuity. This work aims to develop such an approach.

The central argument in this work is that, for a broad range of images and scenes, important edges are likely to be both continuous and smooth. In building representations of a scene we are frequently interested in this sort of detection — we are interested in relatively large scale features of an image, and not in rapidly varying textural elements. It will be shown that this approach produces edge images which have broad applicability across a range of scenes.

Fundamentally, the most common purpose of edge detection is simply to reduce the number of calculations that following steps in a vision algorithm need to perform. It could be said that edge detection seeks to represent an image by identifying the likely boundaries between objects in a scene and likely points of interest within those. Martin et al. reduce the problem further by defining edge detection as concerned only with identifying the "abrupt change in some low-level image feature such as brightness or colour" [14]. However, in broadening the scope from low-level to a mid-level approach (for lack of a better word — not strictly low-level and not global) it must be accepted that the line between edge and boundary detection is blurred. Traditionally, boundary detection is concerned with finding full

and strictly connected contours around objects in scenes. In this approach, a middle ground is sought, one which is concerned with how an abrupt feature change is propagated in the environment around each pixel but does not tie itself to mandating edge pixel connectedness.

As one of the fundamental steps in Computer Vision, improvements and developments in edge detection can yield far-reaching results. Any application where edge detection is used stands to be improved, from self-driving cars to object recognition and scene reconstruction.

1.2 Summary

This work is divided into a number of chapters:

Chapter 2 is a review of the current state of the literature on edge detection, which will contextualise and inform the rest of the work. Particular attention is devoted to a number of different broad classes of approach to edge detection. The approach developed in this work fits somewhere in between these broad classes, bridging the gap between approaches that consider small regions around each pixel, as in 2.1, and approaches that attempt to integrate information from the entire image at once, as 2.2.

Chapter 3 details a number of successive developments and approaches to the problem of continuity-based edge detection. Each approach leads to some broader insight about the problem as a whole and serves to motivate the next. Particular attention is paid to the trade-off between pixel- and contour-based analysis of edges. Arguments are also made for the necessity of an easily configurable approach to allow for the multitude of potential problem domains to which the detector could be applied.

Chapter 4 describes the operation of the final algorithm, building upon the lessons learned from Chapter 3 above. In particular, it describes each stage of the algorithm:

- Sobel Detection
- Polar Representation
- Weighted Averaging
- Negative Deviation Weighting
- Break Counting

- Directional Edge Likelihood Histogram
- Edge Classification
- Hysteresis Combination

Chapter 5 seeks to describe and evaluate the results of the algorithm. It also outlines the problems inherent in attempting to objectively evaluate edge detection methods, especially those with different focuses than the extremely general datasets currently available. It highlights a number of key areas where the algorithm has advantages over some contemporary approaches, as well as some areas where it is outperformed. The issue of applicability is investigated.

Chapter 6 concludes the work, contextualising its results and applicability while highlighting areas of potential future work in the field. The main results are summarised and reiterated. The main finding of this dissertation is that edge continuity is both a valuable and under-utilised method of determining edge saliency, and that its ability to separate continuous smooth edges from discontinuous noise while maintaining fine detail is broadly applicable in the field.

Chapter 2

Literature Review

Edge detection is often used as the first step in vision operations. Its goal is to identify "salient" points in an image. Specifically, to identify the likely boundaries of objects within a scene. Edge detection is most often used to reduce the amount of information which later stages of a vision algorithm must consider. For instance, recognition algorithms can use edge detectors to attempt to recognise objects by their boundaries, or detection algorithms can find flaws in products on an automation line using their outlines.

Historically, most edge detectors suffered from problems of noise sensitivity. Because these local gradient based edge detectors focussed only on small filters (e.g. 3×3, 5×5) they were easily influenced by the presence of noisy pixels. In this context it is important to note that we take noise to mean both systematic noise (e.g. Gaussian noise, salt-and-pepper noise) which are consequences of image capture and transmission, and content noise (e.g. textural elements, small-scale image variations) which are contained in the scene as is, but are not "salient" edge points. An example of this latter type of noise is that of leaves on a tree — there are a huge number of varying points and therefore strong local edges and they correspond to real-world objects captured "correctly" by the image capture process, but they are not considered important as boundaries within the scale of the entire image.

More recently, large amounts of work investigating the concept of large-scale *ground-truth* datasets both in terms of their creation [15] and their use as the training set for complicated machine-learning approaches to edge detection [12, 5, 14]. These approaches have received some criticism for their methods of evaluating the value of human-annotated ground-truth [9]. Further, while edge detectors of this kind perform "well" on these benchmarks and are certainly much more resistant to noise than their earlier counter-

parts, they tend to be computationally expensive, both in terms of their initial processing, and of their run-time over even relatively small images.

This work aims to bridge the gap between these two approaches by using broader scale directional continuity metrics to improve upon the output of detectors like Sobel[6].

2.1 Local Filters

Local Filter approaches work by calculating some value for each pixel in an image based on the local neighbourhood of that pixel. Roberts developed one of the first edge detectors [21]. It used a pair of 2×2 filters to find an approximation of the derivative for each pixel:

$$\begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}, \quad \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}$$

Due to the fact that this approach only considered the 2×2 neighbourhood of each pixel it was highly susceptible to noise.

Sobel and Feldman[1] developed a pair of 3×3 filters which included a smoothing operation and an estimate of the derivative in both x and y directions:

$$\texttt{Y-Direction:} \begin{pmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{pmatrix}, \quad \texttt{X-Direction:} \begin{pmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{pmatrix}$$

This can be seen as the combination of a Gaussian smoothing filter and a simple derivative estimate. Their approach results in two images, one for each direction. From these images, both the edge "strength" and normal orientation can be computed.

$$|G| = \sqrt{{G_x}^2 + {G_y}^2}$$

$$\theta = \operatorname{atan}\left(\frac{G_y}{G_x}\right)$$

Both Canny [3] and Marr & Hildreth [13] sought to formalise edge detection in terms of various optimality constraints — Canny held that detected edges should have a low error rate, be spatially well-localised and respond

[1]This approach was first published in a footnote in [6]. Though apparently Sobel presented this work at a meeting at SAIL in Stanford previously [4].

only once to each edgel[2] while Marr & Hildreth sought to find the optimal balance between frequency and spatial domain localisation. Both resulted in the adoption of some form of the Gaussian filter. In Marr & Hildreth's case, they used the Gaussian:

$$G(r) = \frac{1}{2\pi\sigma^2} \exp\left(\frac{-r^2}{2\sigma^2}\right)$$

as the optimal smoothing filter to apply before edge detection.

In Canny's case, he uses the zero-crossings of the second derivative of the Gaussian to locate local maxima (i.e. edges) and a first derivative operator similar to Sobel to estimate edge strength. His method of thresholding is also interesting. He develops the concept of *hysteresis thresholding* where possible edge points are initially thresholded by some high value, and then any possible edge points which are connected to the initial edge points are thresholded by some lower value. This allows the detector to be robust against "streaking" where edges whose average value is near the threshold fluctuate above and below it and so are broken up by the thresholding step.

2.2 Snakes and Cost Minimisation

Another approach to edge detection, one which aims to be solve some of the problems of the local filter approaches above, is the framing of the problem as one of energy minimisation. Menet, Saint-Marc, and Medioni give a review of the state of the art in 1990, and add their own improvements [16]. This approach focusses on the idea of a "snake" — "an elastic curve which evolves from an initial position in the image (provided either by interactive action of the user, or by higher level process) toward features to extract, by means of energy minimization. Its energy accounts for elastic forces (smoothness constraints of the curve: tension and bending) and image forces (fitted to the features)." This model guarantees smooth continuous curves by definition, and seeks to reduce the influence of noise by valuing broad scale patterns in the image and ignoring high-frequency local variation. It assumes that there exist some edges in the image, and that the salient ones will be smooth, continuous, and will explain as much of the image data as possible. It "costs" the snake more to occupy areas of low gradient and to have sharp curves or corners. Tan, Gelfand, and Delp propose a similar but improved model which takes into account more local information about each edge — its local

[2]By analogy with pixel: an edge element

continuity, thickness etc. — and uses these as further costs to be minimised by a simulated annealing process [22].

Felzenswalb and McAllester further this approach by using classical greedy approximation to find a small number of smooth curves which explain as much of the image as possible [7]. Their approach begins with short curve segments and builds them up by selecting likely continuation candidates by their costs. They state the benefits of their algorithm as being significantly more noise resistant than local filter based approaches. By focussing on only the most salient of curves they contend that their approach provides a better approximation of saliency in the image than other detectors.

2.3 Learning-Based Approaches

2.3.1 Ground-Truth and Evaluation

Before any learning-based approaches can be described, we must first address the issue of *ground-truth* for edge detection because these algorithms necessarily must have a ground-truth labelled dataset on which to train. In framing the problem of edge detection as one that seeks to find "salient" boundaries in images, we admit that there can not be one true ground-truth edge image for a given image. "Saliency" is poorly defined, and arguably depends on the exact application of a vision algorithm — an industrial algorithm for finding small defects in label stickers on glue bottles will want to find edges at a much finer scale and with lesser continuity and smoothness constraints than an algorithm that attempts to recognise and distinguish glue bottles by their shape. From this it could be argued that detectors which allow some application-specific tuning parameters have an advantage over those that don't.

Nevertheless, a number of datasets exist that map images to ground-truth edge images. One of the first of these was the Sowerby Dataset, featuring pictures of the English countryside segmented by humans[3].

A more extensive approach was developed by Martin *et al.* in [15] — the Berkeley Segmentation Data Set (BSDS). They created a dataset (initially of 300 images, expanded to 500 in a later publication) which features a number of segmentations of each image by a number of different human subjects. They contend that by creating an aggregate segmentation in this way that they reach closer to the empirical ground-truth of the image segmentation

[3]This dataset is referenced in [10] but does not seem to be publicly available itself. It is attributed to the Sowerby Research Centre, British Aerospace.

as it would be perceived by the average human. This approach has been used to evaluate a number of the edge detectors published since (and indeed to Canny as a baseline) [12, 5, 14, 19, 7]. The benchmarking algorithm used allows for some "slop" in the detected edges — i.e. it allows for edges to be within 2 pixels or so of the dataset's ground-truth and still be counted as true positives. It also evaluates edges at the level of the individual pixel exclusively, rather than at a higher conceptual level such as continuous edge segments.

This approach has been criticised by a number of papers in the literature. In [7], Felzenswalb and McAllester maintain that since their min-cover approach attempts to find the most salient continuous smooth edges, it performs more poorly than is warranted on the BSDS benchmark, which does not take continuity or saliency[4] into account.

Hou, Yuille, and Koch published a paper critiquing the BSDS for the undue weight it places on "orphan" labels — those edge pixels which were only identified by one labeller. According to their analysis, these orphans make up ~30% of the pixels in both the BSDS300 and the BSDS500. They argue that since false positives are counted only if *all* labellers labelled a negative whereas false negatives are counted for *each* labeller that marked a positive, that the benchmark favours a liberal marking of edges. They create a new dataset built from only the "consensus" labels of the BSDS — those labels for which every test subject agrees. They found that for those edge detectors whose source code was publicly available, all suffered a significant drop in F-score when evaluated against this new dataset,[5] suggesting that no current edge detectors are successful at distinguishing "strong" edges (those marked by all human subjects) from "weak" ones.

2.3.2 Methods

pB [14] uses measures of the oriented energy and the gradient of local textures, brightness, and colour as features for a supervised learning process on the BSDS. Oriented energy in this case refers to the response of a pair of even and odd-symmetric filters to the local region around each pixel, and serves as a measure of the "linearity" of the local region. For brightness and colour gradients, they compare the histograms of two half-discs centred

[4] Arguably, they use the number of human subjects who agreed upon an edge as an implicit measure of saliency, but only for false negative calculations

[5] They went so far as to retrain the pB detector on the new dataset and test it again, finding very little improvement on the task of detecting "strong" edges, even when trained exclusively with them.

on the pixel under consideration at some orientation θ (they compute the response at 8 orientations). For textures they utilise a bank of 13 filters, associate each pixel in each half-disc with the vector of its 13 responses, and then compare the two resulting half-disc distributions. They claim that this explicit treatment of texture is necessary as other operators, like a spatially-averaged second moment matrix, tend to suppress both legitimate textured areas along with corners, junctions, and the boundaries of textured regions (all of which *should* be detected). They report an F-score of 0.67 on the colour BSDS benchmark.

The pB detector is improved upon in [12], which uses a multi-scale version of pB along with a spectral component, sPb, to create gPb, an edge detector that combines information from the local and global scales. This spectral component works by analysing the eigenvectors of an "affinity matrix" which encodes information about how similar pixels are to each other. This gPb detector has a reported F-score of 0.70 on the colour BSDS.

Inspired by [17], the authors of [19] and [11] apply sparse coding techniques to edge detection and class-specific boundary detection respectively. In [19] a sparse coding scheme is learned from an un-annotated image, and then this sparse code is used as a feature set to learn to detect edges on the BSDS. This approach has the advantage that it is easy to expand to extra channels (e.g. depth from a depth camera). Further, it outperforms gPb on the BSDS benchmark, scoring 0.74 on the colour version. (It is important to note that in this case, the sparse coding is only used for a local 5×5 patch detector, it uses sPb for global information in exactly the same manner as gPb does.)

Chapter 3

Design

The initial conception of this project was to develop a new method of edge detection which focussed on edge continuity information — as such it was taken to be a second-step in the edge detection process, i.e. one that would refine the results of a previous edge detection stage. Throughout the project a number of different approaches were developed exploring different conceptions of continuity and edge processing.

This chapter will provide a quick overview of a number of the most significant approaches developed. Their advantages and disadvantages will be discussed. Many of the techniques and motivations for the approaches below survive in the final design, while some present avenues of future work.

3.1 Initial Edge Detection

The initial edge detector should be simple and have a low false negative rate — i.e. it should detect every edge in an image, even at the consequence of falsely detecting edges where there are none. This is because the post-processing that our edge detector applies will behave like a filter, but it cannot generate information from nothing. Therefore using an edge detector which attempts to find small numbers of salient curves like [7] is infeasible.

We also want this step to be as quick and simple as possible, so that its affect on the overall run time of the algorithm is minimal. The clear choice is to use a Sobel-like edge detector, which at small filter sizes has excellent detection and localisation but has a high rate of false positives. Sobel edge detection is very simple yet produces both a greyscale gradient image and an orientation image, so that each pixel is associated with both its maximum gradient value and the angle at which that maximum gradient occurs. This

is useful in our processing, as we use the orientation of an edge as a key cue for determining continuity metrics.

3.2 Contour Following

Inspired by the extension field of [8], the edge length metrics of [24], and related approaches the initial approach was a global contour following algorithm. The idea was to build contours up from an initial strong edge point by picking the strongest neighbouring pixel and then for that pixel, picking the strongest again and so on. At each stage, the pixels to be considered would be weighted by a function which took into account both the orientation information from Sobel and information about the edge contour up to that point.

$$\begin{pmatrix} 1.3 & 1.5 & 1.3 \\ 1 & 0 & 1 \\ 1.3 & 1.5 & 1.3 \end{pmatrix}$$

This is an example of one of the weighting matrices — for a contour where the edge orientation implies the next pixel should be either above or below. This is only one half of the necessary weighting, the one that takes into account just the edge orientation from Sobel. However the real value in this approach is in its ability to utilise information from the rest of the edge contour. To do this we must also weight the next pixel based on the previous behaviour of the contour. One can conceptualise this as giving the edge a kind of pixel to pixel inertia — it wants to continue in the same direction. This concept can be extended to more than just direction: we could use information about the rate of change of the contour etc. However, these more complicated ideas were only examined, not implemented.

However, this approach suffered from a number of problems — edges were detected multiple times, edges tended to double back on themselves at their ends or at junctions, the weighting approach is distinctly asymmetric in that it is heavily dependent on which end of an edge is found first etc. There are a number of possible solutions and further developments of these ideas, for instance the contour continuity metrics from [24] could be applied to resultant edges, some form of contour joining approach could occur where nearby detected contours can pool together into thicker contour representations etc. For the purposes of this project these seemed unlikely to yield the desired results. It was decided that an approach based purely in pixel space rather than in contour space would be more likely to yield scalable results.

3.3 Neighbour Voting

We develop two key metrics for the evaluation of edge continuity in pixel space — *Directionality* and *Collinearity*. Directionality refers to the extent to which a pixel's Sobel gradient orientation is in the same direction as the gradient orientation of the central pixel. Collinearity refers to the extent that the pixel lies on the line orthogonal to the central pixel's gradient orientation. Directionality is defined as

$$D_n = 1 - \alpha \left(\theta_c - \theta_n\right)^2$$

Where α is a parameter that determines the "sharpness" of the peak — how harshly we penalise inexact matches of orientations. θ_c refers to the gradient orientation of the central pixel, while θ_n refers to the gradient orientation of the n^{th} pixel (i.e. the one currently being considered).

Similarly, collinearity was defined as

$$C_n = 1 - \alpha \left(\theta_n - \phi_n + \frac{\pi}{2}\right)^2$$

Where ϕ_n refers to the angle between the central pixel and the n^{th}. Other definitions were also investigated, but it was found that the quadratic offers the best performance. The testing, along with more information about this approach, is located in Appendix A at the end of this report.

Finally we combine these values to get a score for each pixel in the local neighbourhood, which we then sum together.

$$S_n = D_n C_n \frac{V_n}{255}$$

Where V_n is that pixel's gradient value. 255 is merely a scaling factor as we are dealing with 8-bit images.

We then sum each of these for the local neighbourhood, multiply by a scaling factor proportional to the size of the local neighbourhood, and assign the new pixel this value.

This algorithm in general had a number of problems, among them that detail was lost almost as quickly from some shorter edges as from "noise" pixels — compare the leafy regions with the small windows on the building in each of the examples. The algorithm has no concept of the distance of each pixel to the central pixel (ideally nearby pixels would be weighted more heavily).

Chapter 4

Implementation

4.1 Final Design Goals

From the previous designs, a number of key features were identified for the final edge detector:

- Based on Local Pixel Filters
- Directional Edge Magnitude
- A Notion of Continuity
- Distance Sensitivity
- Edge Strengthening
- Weakening of "Noise" Pixels
- Configurable

The notion of Directional Edge Magnitude above warrants further explanation. A common problem with previous approaches was the way they dealt with edges ending, changing direction, or crossing. In each of these cases, local information tends to get reduced — at endpoints, only half of the line exists, and so score tend to dip drastically in the region surrounding an end. Corners suffer from a similar problem, where edge pixels suddenly occupy an entirely different position to that which was expected. Many approaches in the literature suffer from similar problems — the oriented energy metric of [14], or the log-Gabor filter of [24] are examples.

This dissertation introduces the concept of directional edge magnitude to mitigate this problem. The directional edge magnitude at a pixel refers

to the edge magnitude emanating from a pixel at each direction. This can be thought of as a graph with angular direction along the x-axis and edge magnitude along the y-axis. In this model it is much more simple to differentiate between endpoints, corners, junctions, midpoints, and non-edge points. This allows this approach to avoid many of the common problems associated with linear local filters.

4.2 Algorithm Overview

The algorithm consists of a number of discrete steps. The processing pipeline of the algorithm as developed is as follows:

1. Sobel
2. Polar Representation
3. Weighted Averaging
4. Variance Weighting
5. Break Counting
6. Directional Edge Likelihood Histogram
7. Edge Classification
8. Hysteresis Combination
9. Iteration

4.3 Sobel

Throughout all of the development and design cycle, the 3×3 Sobel filter has performed well as a first stage of the algorithm, almost always providing at least some response for every edge in the image (there are some cases where no physical difference in pixels occurs but humans perceive an edge because of a semantic understanding of the scene). Therefore in this final design, the first stage will again be a Sobel detector. This means that at the beginning of our algorithm we have both a gradient magnitude image and a gradient orientation image as inputs.

4.4 Polar Representation

As described above, many of the common problems of previously developed algorithms can be mitigated by creating a local directional edge magnitude graph. A first approach based on hand-crafted pixel masks was quickly discarded as being too inflexible and time consuming to configure. Instead a modified polar representation system was created which allows for easy configuration while creating a reliable base for directional edge magnitude calculations.

First, a quadratic orientation agreement step, as in 3.3 above, is applied. This step uses the same conception of collinearity as the Neighbour Voting algorithm. This allows for pixel magnitudes to be progressively reduced as their orientations fall further from their expected ones. This step provides us with a local neighbourhood weighted by each pixel's directionality as dictated by the α parameter, which determines the range of angles differences which are positively weighted. High α values result in collinearity values which require almost exact agreement between gradient magnitude and expectation, while low values are much more permissive.

We assign each pixel in the local neighbourhood to an entry in a $r \times a$ matrix, where r is the maximum distance in pixels away form the centre that we wish to consider and s is the number of angular divisions we wish to have. Figure 4.1 shows an example where the grey value of each pixel indicates the sector it has been assigned. Multiple pixels can be assigned to the same bin under this scheme. We take the maximum of all assigned values. This is a key difference between this approach and regular polar representations, which would take some form of average of all pixels assigned to the bin. Here, however, all that is of interest is the continuation of the edge, and it is counter-productive to reward broader edges (i.e. those with many relatively high values per bin) compared to narrow edges (those which have a single high value, with the rest being much lower). This was a large problem with approaches like Neighbour Voting above, where very sharp edge lines would be punished compared to those with slightly more gradual changes. By taking only the maximum, we avoid these problems entirely.

Figure 4.2 shows an example local neighbourhood region — a section of bridge with three distinct edge lines departing from the centre. The orientation image 4.2c shows the relatively consistent orientation information along each of these lines. It is important to note however that since this orientation is only based on the 3×3 local neighbourhood around each pixel it can vary quite a bit based on small variations in an edge line. Therefore θ, our maximum permissible angle difference, must be relatively large. It is possi-

ble that using a larger filter Sobel detector to build the orientation image might produce more consistent results but this has not been investigated.

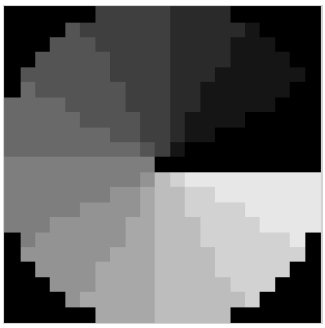


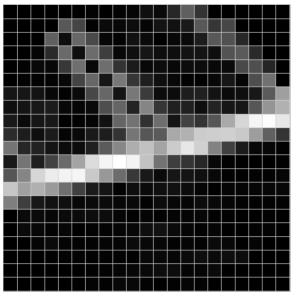
Figure 4.1: An example visualisation of the pixels considered to be in each sector. $r = 10\texttt{px}$, $s = 12$

The result of this step is a representation of the local neighbourhood with distance from the central pixel increasing downwards along the y-axis and angle increasing anticlockwise along the x-axis, as in Figure 4.3 below. It can be seen that this representation allows for the easy analysis of lines in the local neighbourhood, as lines in all directions from the centre are transformed simply into vertical lines in the polar representation.

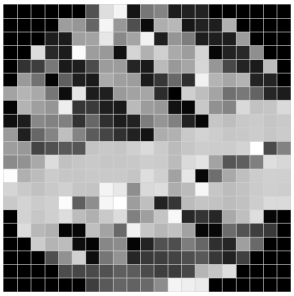
It is important to note that there are pathological cases where this method of building a representation can cause undesirable results — consider a sector where there is a strong edge pixel at each distance but that at successive distances this strong pixel occurs at opposite sides of the sector. In this model that sector is treated as equally "continuous" to one in which there is a single straight line of strong edge pixels. We rely on the narrowness of our sectors to mitigate this problem, however future work could investigate a more robust approach which considers the connectedness or angular variations between successive points in the polar representation.

There is a problem with this approach however. Hard binning results in a situation where continuous lines in the image can fall on the boundary between two adjacent sectors and are therefore split so much as to be discounted unfairly. This is mitigated by an incursion parameter, i, which allows pixels to be considered for inclusion in an adjacent bin if they are within some fraction of the sector width of a boundary.

(a) Sobel gradient image showing enlarged region



(b) Masked and enlarged gradient neighbourhood



(c) Masked and enlarged orientation neighbourhood

Figure 4.2: Example of Local Neighbourhood

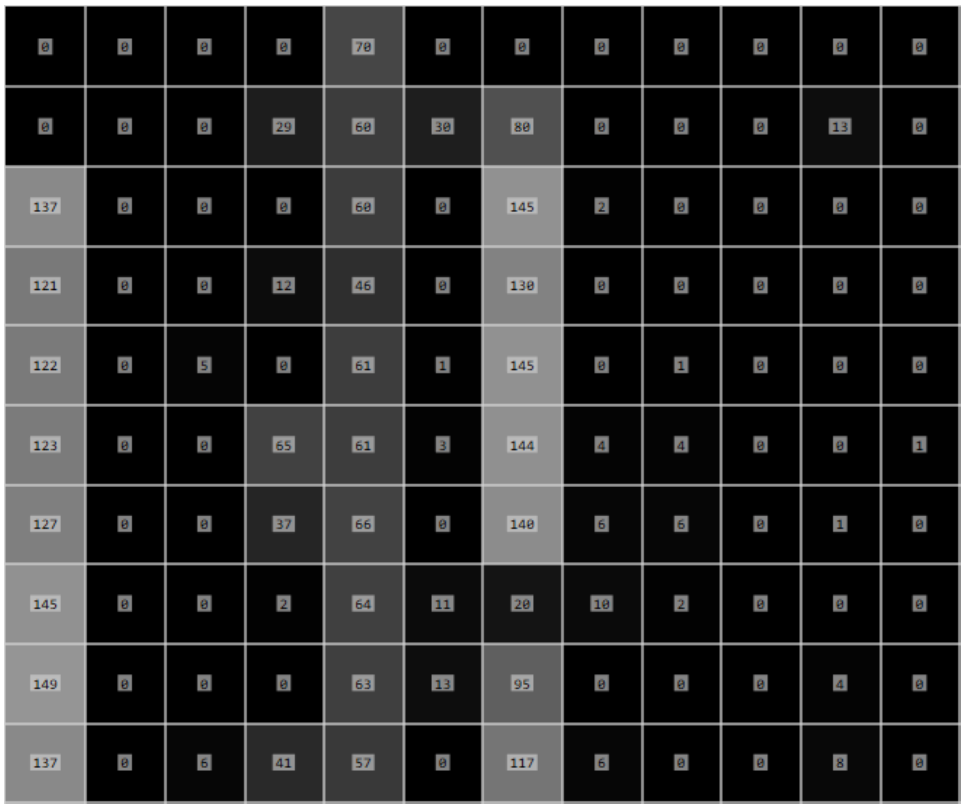


Figure 4.3: Polar representation of figures 4.2b and 4.2c

Parameters Introduced in This Section

α The coefficient of the orientation weighting quadratic — dictates the permissible deviation from expected orientation

r The maximum radius (in pixels) at which pixels are considered to be in the local neighbourhood.

s The number of sectors into which to divide the polar representation of the neighbourhood.

i The fraction of a sector which overlaps with the adjacent sector.

4.5 Directional Edge Processing

4.5.1 Weighted Averaging

We would like to have a measure of a sector's combined edge strength which is more sensitive to edgels that are closer to the central pixel than those further away. This would allow for edges which are strong for the first few pixels but tail off afterwards to be weighted more strongly than those which vary more over their length or are only strong far away from the centre. Intuitively, it makes sense that nearby pixels provide more relevant evidence about the existence of an edge at a point than those far away.

To implement this in practice we take a weighted average of each column in the local neighbourhood polar representation (as in Figure 4.3 above). We weight this average so that the first (i.e. closest) pixel counts twice as much towards the final result as the last.

$$\frac{\sum_{i=0}^{r}(2r-i)(val_i)}{\sum_{i=0}^{r}(2r-i)}$$

Where val_i is simply the value at the i^{th} distance in a sector. This reduces the polar representation to a single number per sector, as in Figure 4.4 below.

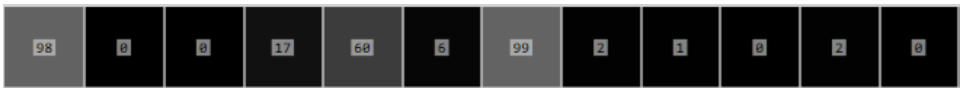


Figure 4.4: The results of weighted averaging on the polar representation in Fig. 4.3

4.5.2 Break Counting

In addition to this, we would like to have some metric which favours unbroken edge lines over those which have many breaks. We define a break to be an entry in the polar representation which has a value significantly below the average for that sector. This allows us to recognise that a pixel with a value of, say, 20 can be a break if the rest of the sector has values of 200, but that in a sector where each entry is ~20 it is part of the line. We set the threshold for break consideration as $\frac{2}{3}$ of the average value for the sector.

In practice, this system allows us to ignore a large number of noisy sectors, improving the quality of results. However, future work could be done in more robust means of weighting the relative "broken-ness" of edges, to allow for more fine control than a threshold.

Fig 5.8 in Chapter 5 displays the results of varying this parameter.

4.5.3 Negative Deviation Weighting

A key design goal is that weak but continuous edges should be strengthened. To do this, while further punishing "noisy" sectors, we introduce the concept of negative deviation weighting.

We wish to increase the strength of edges which have few points which are below the average for the sector. We do this by taking the sum of the negative deviations from the average of all points in the polar representation (i.e. we ignore all points greater than or equal to the average and sum the differences between lower than average points and the average). This provides a measure of the extent to which low-magnitude pixels are present in the sector.

We linearly weight this value based on two parameters; p, the point at which the weight is unity, and m, the slope of the line. In practice, we set these parameters so that the maximum possible weight (when the sum of negative deviations is 0) is around 2.

Parameters Introduced in This Section

b The number of permissible breaks along a sector

p The negative deviation value for which the output weight is 1

m The slope of the weighting line for negative deviation

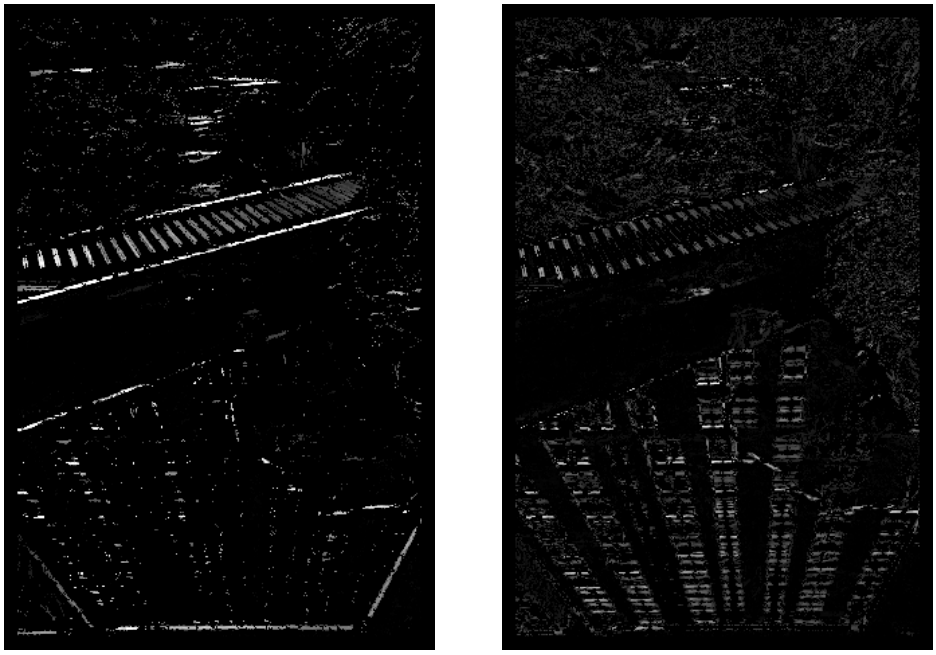
4.6 Edge Classification

The next step in the algorithm is to use the values from the previous steps to classify each pixel into one of four categories:

- Midpoints
- Endpoints
- Corners
- Junctions

The information from the per-direction edge magnitude histogram from 4.5.1 can be used to determine the class of each central pixel for the image. This allows for further processing to be done selectively on each of these classes. The decision itself is quite simple.

- The per direction weighted average from 4.5.1 is non-maxima suppressed, that is, all entries which are not larger than the previous and next entry are set to 0.
- Any entries in the resultant histogram which are below a threshold are discarded.
- Any entries with more than the specified number of breaks are also discarded.
- The remaining entries are counted and analysed:
  - If there is only one non-zero entry, the point is an endpoint
  - If there are two non-zero entries and they are within a sector of being in opposite directions, the point is a midpoint
  - If there are two non-zero entries at any other orientation, the point is a corner
  - If there are more than two non-zero entries, the point is a junction



(a) Midpoints

(b) Endpoints

(c) Corners

(d) Junctions

Figure 4.5: Sample results of the Edge Classification stage

4.7 Edge Combination

As can be seen from Figure 4.5b above, the detected endpoints of an image include large amounts of noise along with pixels which represent both actual endpoints and some smaller detail points (for example, the windows in the reflected building, the ends of the planks on the bridge etc). In an attempt to remove the noise pixels form the relevant ones, a form of hysteresis thresholding inspired by [3] is used. This method is based on the intuition that for an endpoint to be relevant it should at some point be legitimised by a connection to one of the other forms of point — an endpoint connected only to other endpoints or zero-values is very likely to simply be noise.

The notion of connected-ness used for this step is that of 8-connectedness. The region in which a pixel is considered 8-connected to another pixel is simply the 3×3 region around that pixel. (The name comes from the fact that this region includes 8 pixels, excluding the central one.) The concept of 4-connectedness also exists, which excludes the diagonals and therefore only consists of the 4 pixels above & below, and to the left & right.

1	1	1
1	0	1
1	1	1

(a) 8-connected

	1	
1	0	1
	1	

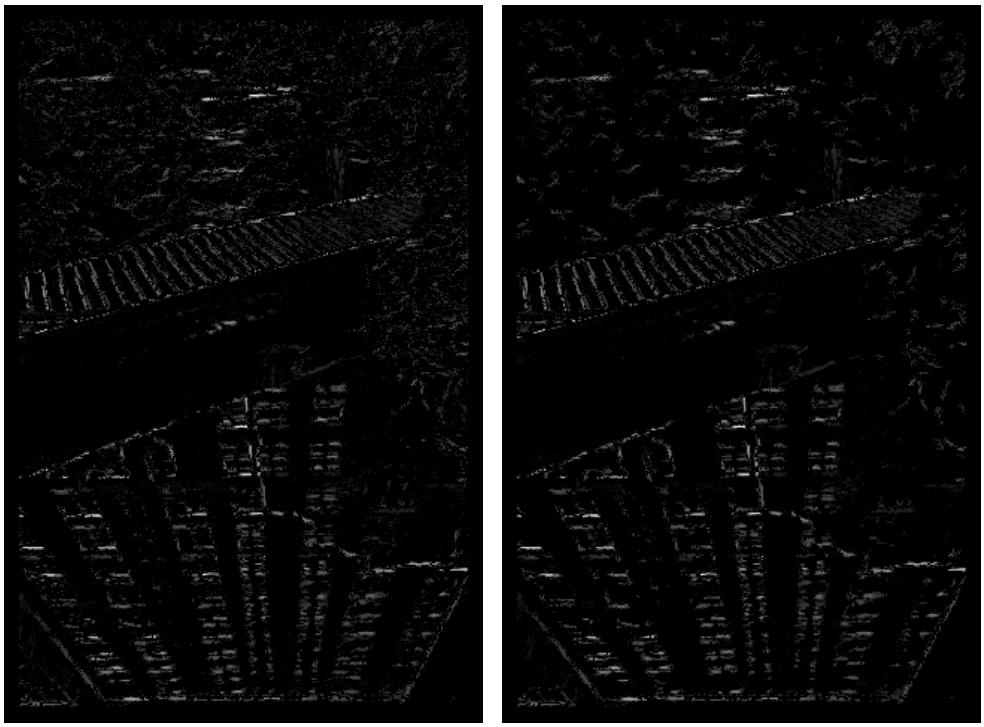
(b) 4-connected

Figure 4.6: The different kinds of connectedness

Each endpoint is checked for an midpoint, junction, or corner in its immediate neighbourhood (i.e. the 8 pixels directly bordering it). If none exists, the next endpoint is checked etc. If one is found, that endpoint and all endpoints connected to it recursively are added to a resultant image. After processing every endpoint, the resultant image consists of only those endpoints which are connected through 0 or more other endpoints to one of the other kinds of points.

The results of this process can be seen in Figure 4.7 below. Note how many of the "noisy" leaf pixels in the upper portion of the image are removed by this step, while maintaining those edge pixels which are meaningful. This step is of course not perfect — sometimes midpoints, corners, or junctions are detected in pixels which are externally considered to be noise, then all endpoints connected to those pixels are passed through. Still, the reduction

is successful enough to warrant this step's inclusion.



(a) Endpoints (b) Endpoints after hysteresis

Figure 4.7: The results of the hysteresis step

Chapter 5

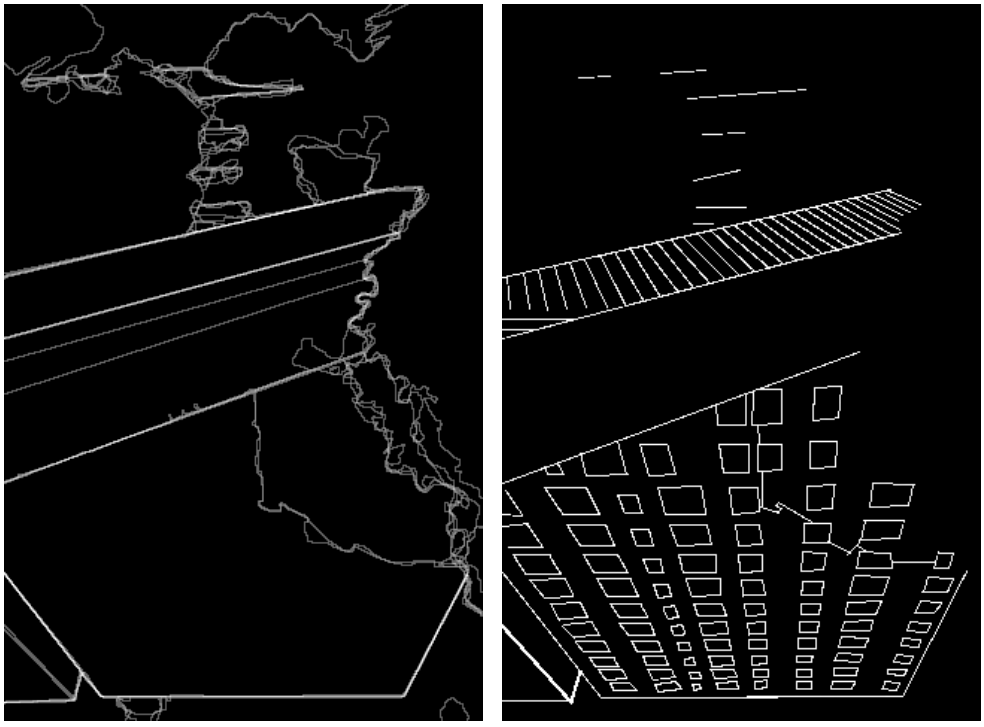
Results & Evaluation

A Note on Evaluation

As mentioned in Section 2.3.1, the issue of ground-truth in edge detection is murky at best. In the case of this approach, evaluation is made more difficult by the goals themselves — the approach seeks to seriously weaken some classes of edges (discontinuous, "noisy" ones) and strengthen others down to relatively low levels of detail. Fig 5.1 below shows a comparison between the "ground-truth" of the BSDS and a rough annotation by the author of the edges which the current detector should favour. There are a number of different things to note from this comparison.

Firstly, in the BSDS ground-truth (5.1a) a number of different human subjects' annotations have been combined into one image. The grayscale value of each pixel indicate how many of the subjects agreed on that pixel being an edge. Note how at the boundaries of leafy sections human subjects agree with each other very infrequently — lines are seemingly drawn at random for large sections of these boundaries. In contrast, note how the straight line edges 'detected' by the human subjects consistently localised. This suggests that our approach in treating continuity and smoothness as proxies for saliency holds at least as far as those lines which human annotators *agree* on.

Of course, it can be argued that while the annotators did not agree on a location for "noisy" edge lines, they did agree that there was an edge *somewhere* in that region. Our detector, as designed, would simply filter these noisy boundaries out. This is valid in a number of cases, especially those involving artificial artefacts in natural scenes, but is ill-suited to some other problem domains. Imagine for instance the problem of identifying



(a) BSDS Ground-Truth (b) Author-Annotated Ground-Truth

Figure 5.1: Comparison of different notions of ground-truth

distinct trees and their foliage in a natural scene — this detector would simply ignore the leafy regions entirely and a detector which has an explicit texture component (e.g. [2]) would perform better.

Returning to Fig 5.1, note the difference in level of detail between the two approaches. The most obvious example of this is in the windows of the building — in the BSDS version, they are simply absent. Again this is a question of focus. It is instructive to look at the instructions provided to the BSDS human annotators [15]:

> Divide each image into pieces, where each piece represents a distinguished thing in the image. It is important that all of the pieces have approximately equal importance. The number of things in each image is up to you. Something between 2 and 20 should be reasonable for any of our images.

The soft limits set by the dataset authors of "between 2 and 20" 'things'

in the image goes some way to explaining the relative lack of detail in the human annotations. This effect is not limited to this example, Fig 5.2 shows a number of other images where smaller objects are ignored in the human annotation in buildings.

In conclusion then, the BSDS is an inappropriate evaluative tool for this detector's specific use case. It would be a valuable piece of future work to create an annotated dataset of the sorts of edges that this detector is designed to weight highly but this work has not been undertaken as of yet. Therefore, for the time being, evaluation must be undertaken by subjective comparison to other edge detectors and to the original Sobel inputs.

5.1 Parameter Effects

Both because there are a relatively large numbers of parameters in this system and because of the application-dependent nature of edge detection, it is instructive to analyse the effects that each of the parameters have on the produced result. To this end, this section will use a variety of images from different contexts selected from the BSDS. Using BSDS data, even without using BSDS' evaluation framework, has the advantage that this algorithm's results can be compared against the results of a number of leading edge detectors, whose edge images are hosted on the BSDS website [1].

Note that for the sample images to follow, the following values of each of the parameters are set unless otherwise stated:

- **Radius** $r = 10$
- **Sectors** $s = 12$
- **Orientation** $o = 5$
- **Incursion** $i = 0.25$
- **Breaks** $b = 2$
- **Negative Deviation Unity Point** $p = 10$
- **Negative Deviation Slope** $m =$ -0.1
- **Threshold** $t = 5$

A simple test image (Fig 5.3 below) has been created to test and display other features of the algorithm in a more controlled setting. The image contains very fine gradations of greyscale value, multiple circles and curves

for testing the response for a variety of angles and slopes, very weak but smooth and continuous curves, "noisy" edge points, and straight lines.

5.1.1 Radius

The radius parameter controls the balance between detail and noise in short line segments. Longer radius parameters naturally favour longer edge lines while punishing shorter ones (even with the weighted averaging, a line ending midway through a sector will still have a large effect).

The radius also has a significant effect on the speed of the algorithm, as it controls the number of pixels in the local neighbourhood, and therefore the number of times that the pixel analysing algorithms must run. The number of pixels in the neighbourhood increase as $(2r + 1)^2$. It is possible that various methods could be employed in an attempt to reduce this load, such as by no longer processing an entire sector once it has a number of low enough pixels, but these approaches have not been investigated in this dissertation.

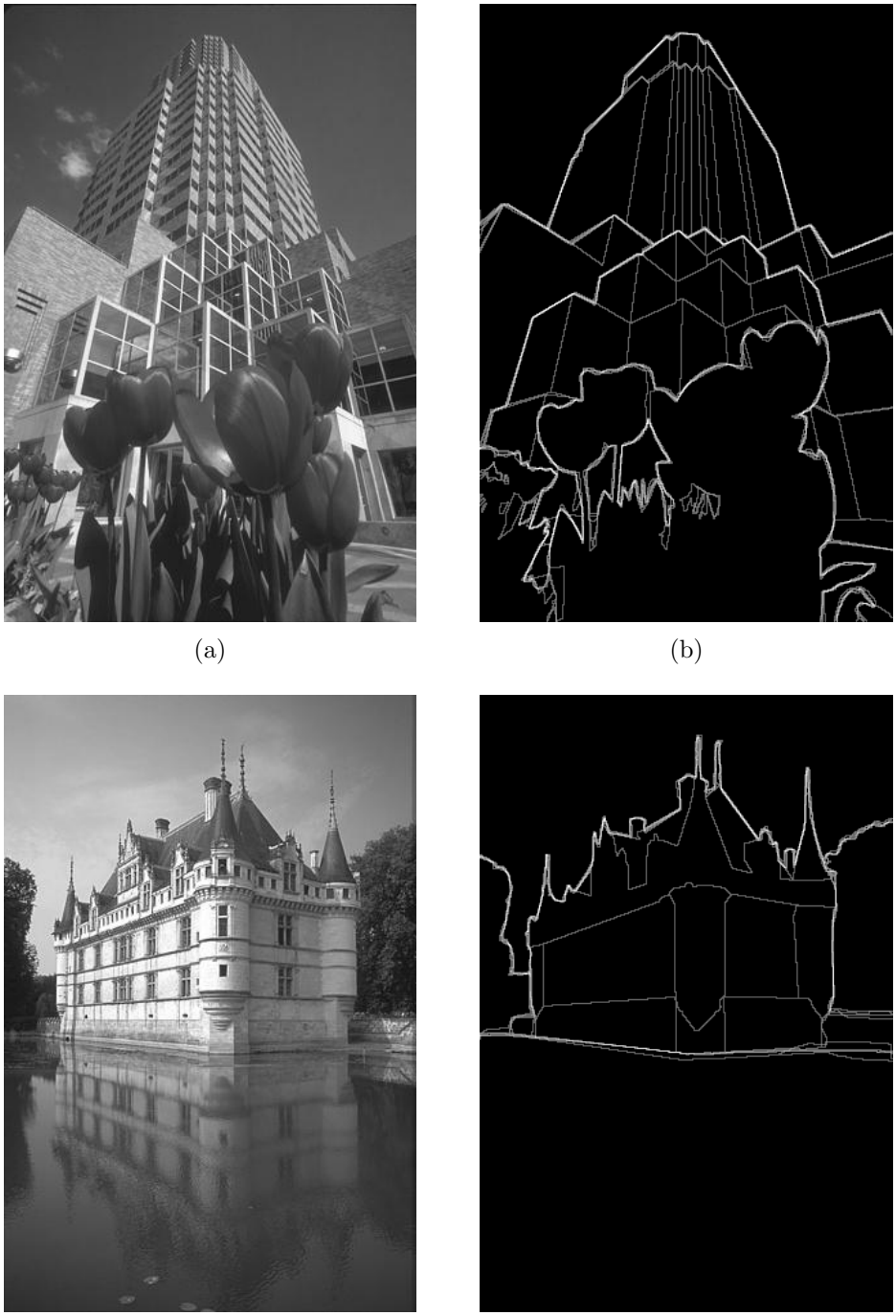
The breaks parameter is intrinsically linked to the radius — with more pixels comes more opportunity for breaks. In the examples in 5.4, the ratio between breaks and radius has been kept constant at 0.2, rather than maintaining the same number of breaks across different radii.

For low radius values as in Figure 5.4 below, noise becomes more difficult to separate from important edges. some edges are still strengthened well though — note how the straight edges of the stones in the upper middle section of BSDS 62 stand out much more clearly from the noisy leaves than in the Sobel image. Note also how the weak curves in the bottom right of the test image are strengthened relative to Sobel also, while much of the "noise" in the bottom left corner has been filtered away.

At a radius of 10 pixels, the effects are more clear — the majority of the leafy textural elements in each of the natural images has been suppressed with only small sections remaining. However, some fine detail is lost due to being shorter than 10 pixels. For example, the smallest windows of the building in BSDS 62 and the smallest concentric circle in the testing image are suppressed.

At 20 pixels, most fine detail is lost. Note how in BSDS78 the short vertical lines of the windows are absent while the long horizontal lines are maintained. Note also how even at this radius the algorithm maintains smoothly curved lines like the ones in the bottom right of the testing image, or the outline of the horse in BSDS33. This is a key advantage of this algorithm as compared to Hough-transform based approaches like [23] which

can only detect straight lines.



(a) (b)

(c) (d)

Figure 5.2: Examples of BSDS ignoring fine details in buildings. (Original image on left, human annotation on right)

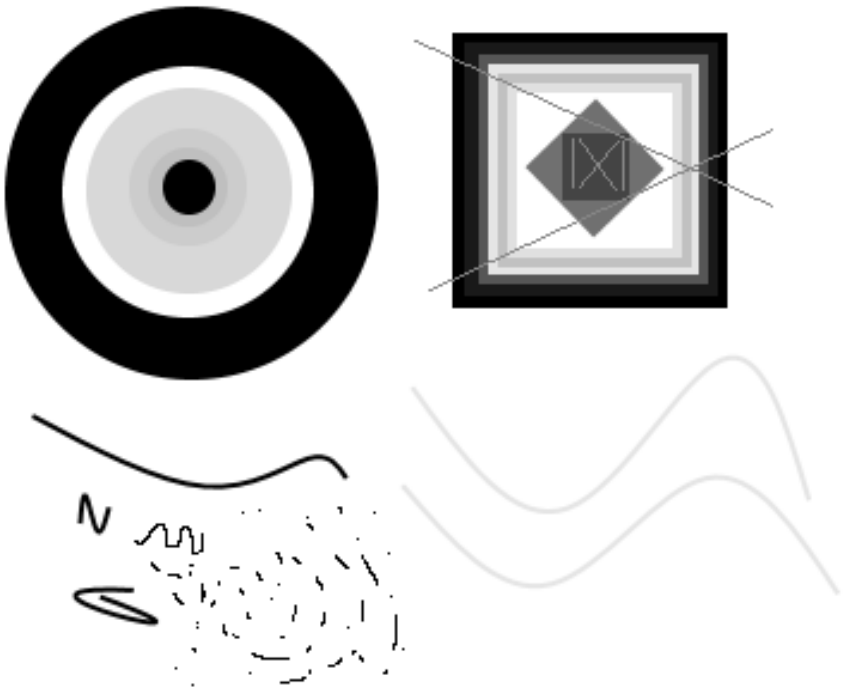


Figure 5.3: Test Image

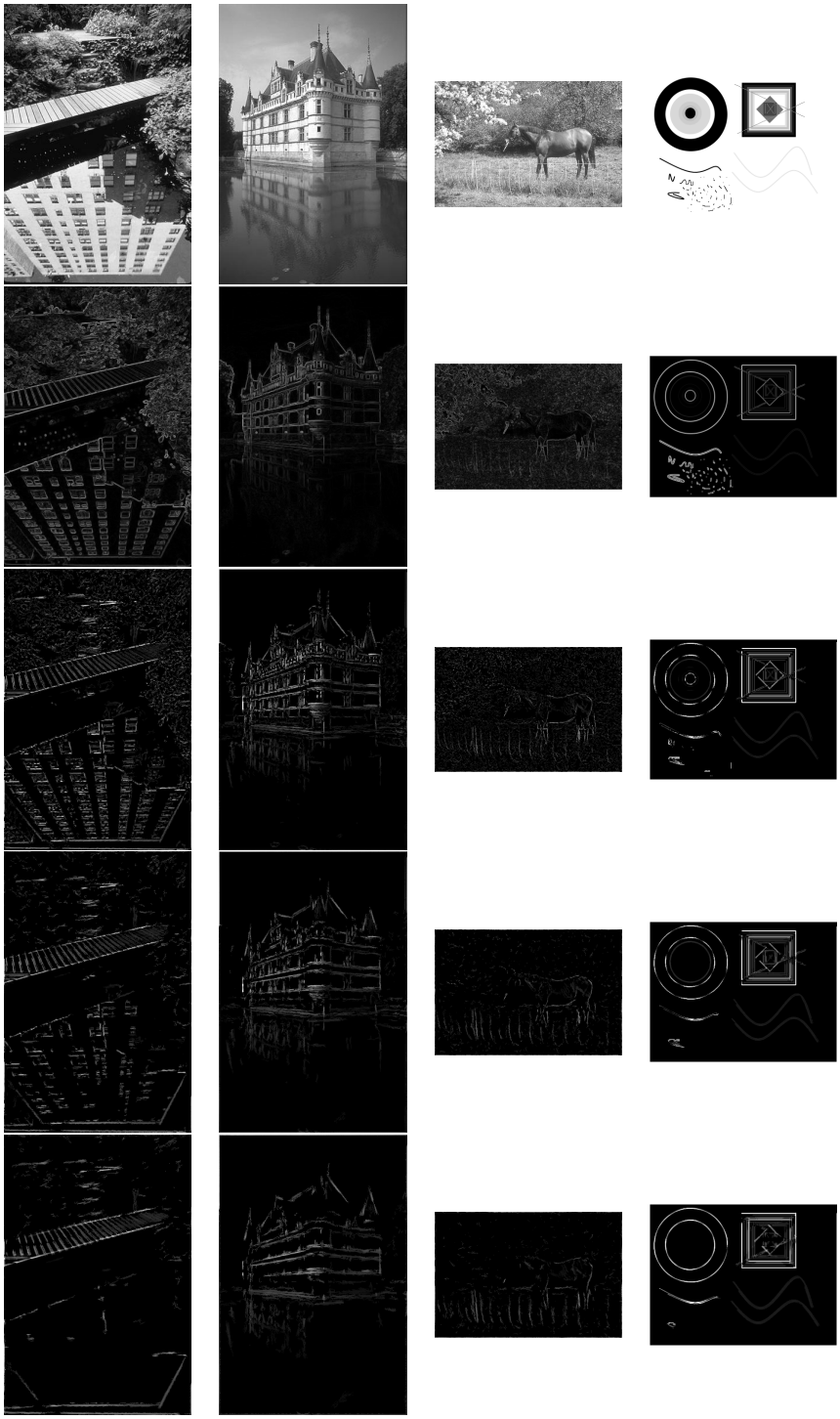


Figure 5.4: The effect of the radius parameter

5.1.2 Sectors

The sectors parameter determines the number of angular divisions each pixel's local neighbourhood is divided into. Earlier, it was pointed out that our algorithm does not have a strong sense of intra-sector continuity and that pathological cases exist where a high gradient magnitude value exists at alternating sides of a sector so that what is a broken and discontinuous region is counted as a continuous one. Altering the sectors parameter allows for the testing of the extent of this effect. With earlier examples showing the results of the algorithm applied with 12 sectors, each sector covers 30 degrees. At a radius of 10 pixels, this results in a maximum of 5 possible candidate pixels at the furthest distance level. The incursion parameter allows for slightly more pixels to be considered but suffice it to say that there is much less scope for the kind of pathological cases described above to exist with large numbers of sectors, purely as a result of their physical width. This implies that low numbers of sectors should result in an increase in the observed noise pixels, as more pixels are considered and in noisy regions the likelihood that a pixel could be in sufficient gradient orientation agreement with its direction to the central pixel increases.

Indeed, in Figure 5.5, it can be seen that this is the case. The leafy regions are much more prominent in the low sector image, and are filtered progressively more the higher the number of sectors.

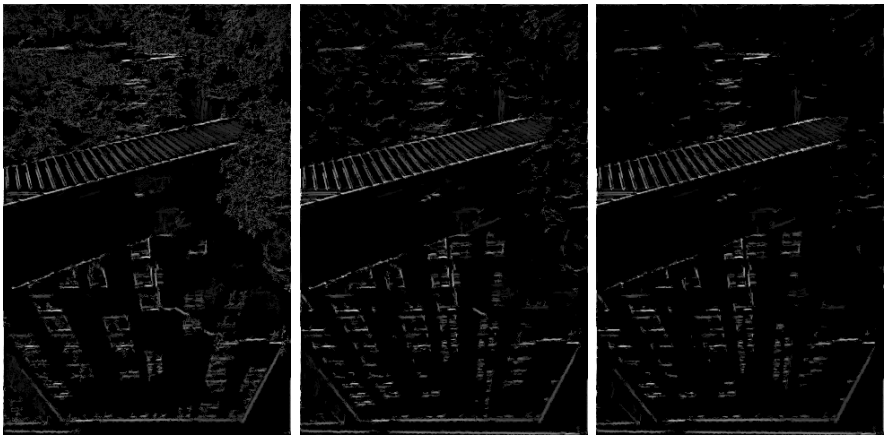


Figure 5.5: The effects of the sectors parameter, $s = 4, 12, 20$ respectively

5.1.3 Orientation

The orientation parameter controls the coefficient of the quadratic relating a pixel's deviation from its expected angle with that pixels orientation weight. It behaves in the same manner as the orientation parameter in the Neighbour Voting section (Section 3.3). Lower values of the orientation parameter result in a more permissive approach to orientation angles — angles which are further away from their expected values will be considered. This parameter controls a trade-off between filtering noisy pixels and allowing for more curves and unexpected slope changes.

Note how for the test image in Figure 5.6 the concentric circles are broken up at higher orientation values. This is a result of the orientations of the pixels within a sector being too far away from the orientations expected purely based on their location relative to the central pixel. Note also how increased permissiveness in the angle results in an increase in both noise and detail in the natural image — both the "detail" of the windows and the "noise" of the leaves are increased with lower orientation coefficient.

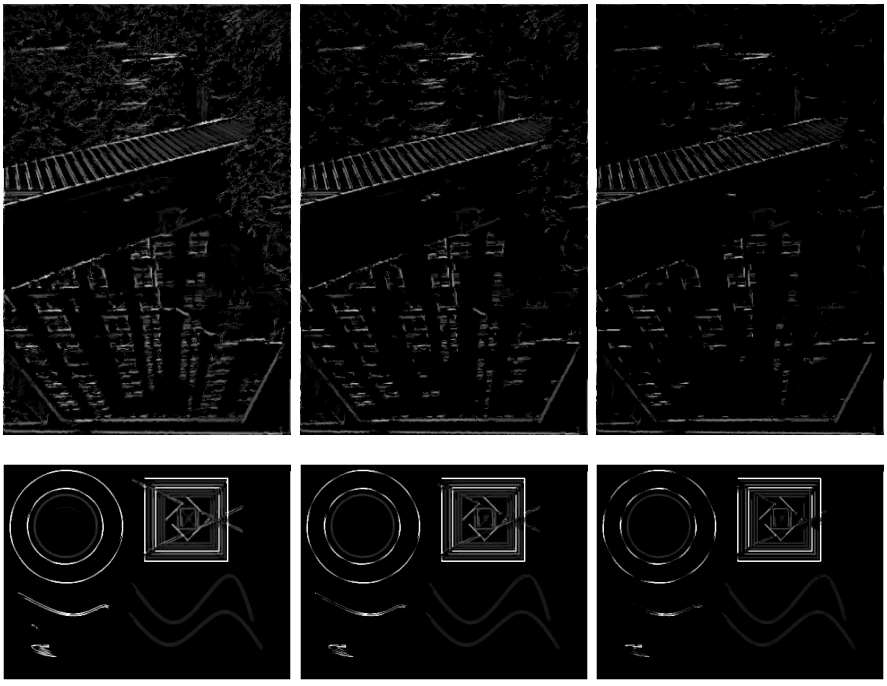


Figure 5.6: The results of the algorithm with $o = 3, 5, 8$ respectively

5.1.4 Incursion

The incursion parameter allows for sectors to overlap each other, in an attempt to mitigate problems caused by edges falling along sector boundaries, or curving so that they are split between multiple sectors. When constructing the polar representation of a pixel's local neighbourhood, the angle between each pixel under consideration and the central pixel is calculated. This angle (in radians) is then multiplied by $\frac{s}{2\pi}$ where s is the number of sectors. This results in a floating point number between 0 and s which determines to which sector each pixel belongs. The incursion parameter i specifies the range where a pixel can be determined to be in one sector but also have be considered for inclusion in another. Specifically, if the part of the calculated index after the decimal point (i.e. the fractional part) is less than i or greater than $1 - i$ then the pixel is considered for inclusion in the index below or above the current one, respectively.

This parameter has arguably less practical use than many of the other parameters involved in the algorithm — a value of 0.25 seems almost universally applicable, with lower values resulting in missed edges and higher ones resulting in untenable amounts of noise. Nevertheless, Figure 5.7 below shows the results of running the algorithm with no incursion, 0.25 incursion as recommended, and 0.5 incursion (i.e. where every pixel always maps to two sectors). As can be seen, no incursion results in very poor detection of say the circles in the testing image, while high levels of incursion don't improve performance on the circle relative to $i = 0.25$ while adding large amounts of noise.

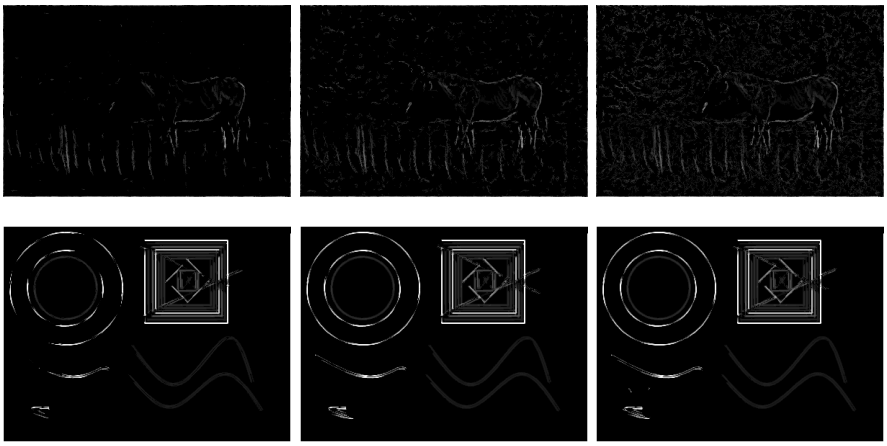


Figure 5.7: The results of the algorithm with $i = 0, 0.25, 0.5$ respectively

5.1.5 Breaks

The breaks parameter performs broadly similarly to the incursion parameter above, in that there seems to be a consistently applicable value and deviating from this either ignores many edges or adds large amounts of noise. Fig 5.8 below shows the results. It seems that a value of one fifth of the radius offers the best results.

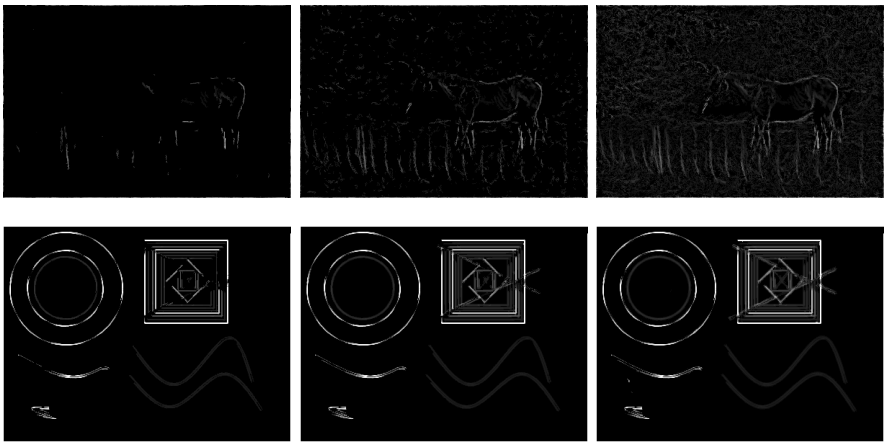


Figure 5.8: The results of the algorithm with $b = 1, 2, 3$ respectively

5.1.6 Threshold

The threshold parameter determines the cut-off point for inclusion into consideration as one of the 4 types of edge point. Any pixels where no entries in the final per-direction histogram are above the threshold are set to 0. As such, the threshold acts effectively like a regular image processing threshold. However, it also tends to push pixels from being considered midpoints, corners, and junctions to being considered endpoints. This has an effect on the hysteresis step and therefore on the final image. Further, very low value pixels can be inadvertently increased by the negative deviation weighting step, as a sector with consistent low values will have very low negative deviation and will therefore be weighted strongly. Therefore it is desirable to have a relatively low but non-zero threshold, to avoid boosting meaningless faint pixels.

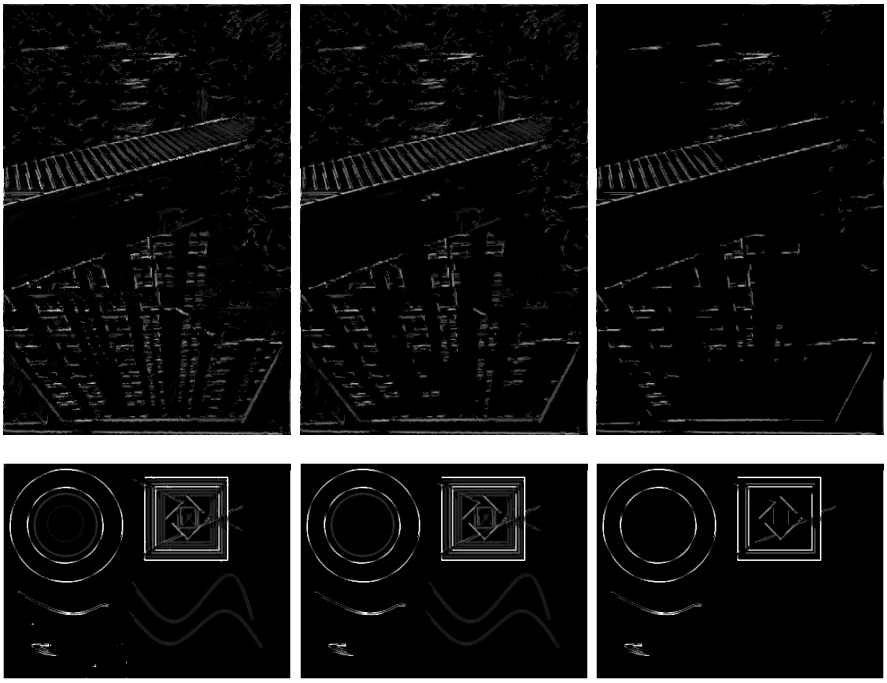


Figure 5.9: The results of the algorithm with $t = 0, 5, 40$ respectively

5.2 Strengths & Weaknesses

In this section, comparisons are made between this edge detector and a number of state-of-the-art detectors. These comparisons are intended to be illustrative rather than particularly authoritative or empirical — the state-of-the-art detectors have all been trained on the BSDS ground truth, which has already been established to be a poor ground-truth for this algorithm's design goals. Therefore, no pixel by pixel comparisons will be made. This section merely seeks to illustrate some differences in the way each detector handles things like continuity, texture, detail, etc.

One of the key strengths of this detector is its ability to extract strong smooth edges from background noise. In Figure 5.11 note how effective the algorithm is at extracting and strengthening the smooth lines of the rocks in the upper middle portion of the image. Note also how the global spectral analysis present in both 5.10d and 5.10e tends to distort and erroneously connect the straight lines in the window segments of the building.

In the natural scene of Figure 5.11 it can be seen that the detector performs comparably to state-of-the-art detectors in terms of finding the

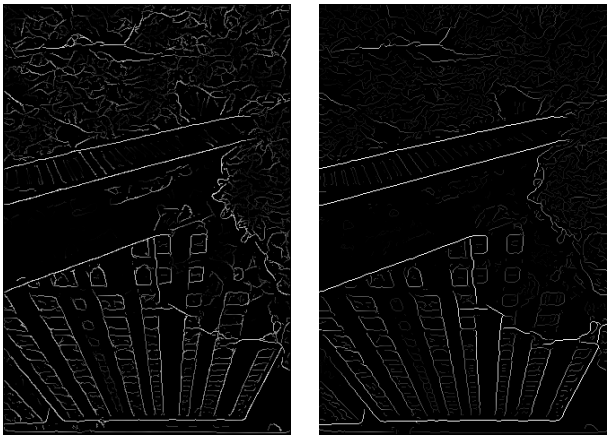
contours of the horse. Note the difference between this detector's strengthening of the straight lines of the fence in the foreground as compared to the other detectors' inclusion of the branches in the top left.

There are a number of types of images for which this algorithm is broadly unsuitable. Since the detector has no explicit notion of textures, it can be confused by strong textural components. Note how in Figure 5.12 both the xren and gPb algorithms tend to close the gaps around the birds' feathers into a single bounding contour. They also respond much more strongly to the boundaries of the birds than this detector.

5.3 Noise Resistance

One final property of the algorithm is its resistance to degradation under the influence of noise in the image. The most commonly encountered type of noise is a simple Gaussian additive noise, where each pixel in the image is subject to modification by a value picked at random from a normal distribution with mean 0. The standard deviation of this distribution determines the extent of noise added to the image. In Figure 5.13 below, Gaussian noise of standard deviation 10 and 20 has been added. Note that at the higher levels of noise, some points are lost while others are erroneously added, but that on the whole the detected edges remain the same.

(a) Original (b) Sobel (c) This Detector

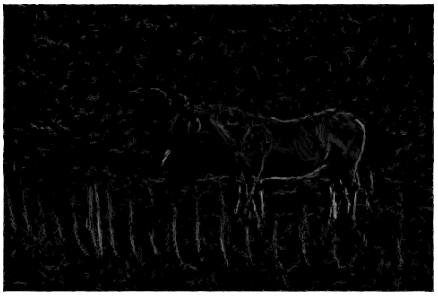


(d) xren [20] (e) gPb [2]

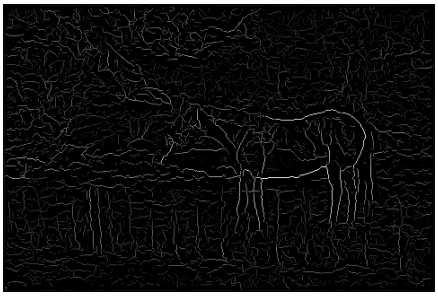
Figure 5.10: A comparison of results on BSDS 62 for a number of edge detectors

(a) Original

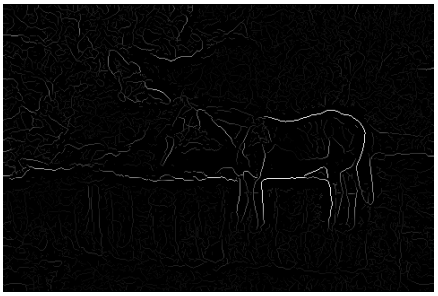
(b) Sobel



(c) This Detector



(d) xren [20]

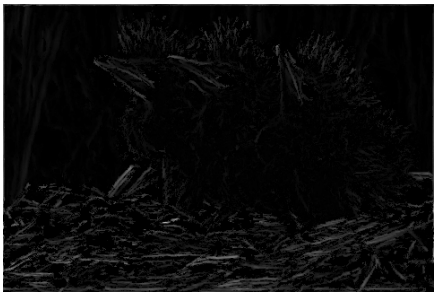


(e) gPb [2]

Figure 5.11: A comparison of results on BSDS 33 for a number of edge detectors

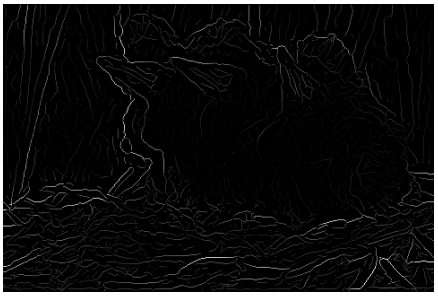
(a) Original

(b) Sobel



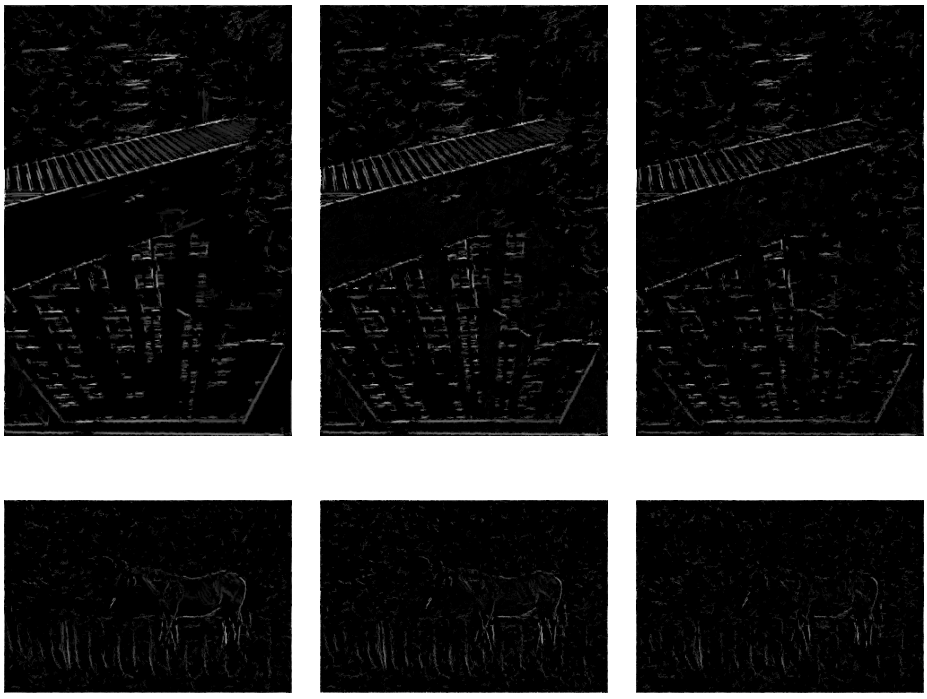
(c) This Detector

(d) xren [20]



(e) gPb [2]

Figure 5.12: A comparison of results on BSDS 4 for a number of edge detectors



(a) No noise (b) SD = 10 (c) SD = 20

Figure 5.13: The effects of noise on the algorithm

Chapter 6

Conclusion

In conclusion, the work presented has successfully developed a novel method for continuity-based edge detection. This method produces excellent results on a number of images. It has broad applicability not only to finding straight lines in buildings and other human-created artefacts, but also to finding smooth curves in certain classes of natural scenes. Current state-of-the-art edge detectors tend to operate on complex machine-learned cues based on small local filters. Few approaches have used large filter directional edge continuity as a metric, as presented in this work. This dissertation has investigated methods of calculating measures of directional continuity and has studied their effects. Based on these results, there is now the potential for machine-learning based approaches to integrate a new class of edge cue, allowing for an improvement in the detection of detailed continuous edges.

Included in this dissertation are the results of a series of alternative methods developed by the author for analysing edge continuity. These results and their evaluation provides valuable groundwork in this field. This will allow for future work to be conducted based on these findings.

Throughout the literature, the issue of edge saliency is contentious. Arguably there can be no ultimate measure of edge saliency without an understanding of later stages in a vision algorithm — an algorithm that exactly follows human perceptual hierarchies might be completely irrelevant in an industrial context etc. It is the contention of this dissertation that an approach based on continuity is likely to be useful in a broad range of practical vision algorithms, especially those based on extracting relatively smooth edges. This is substantiated by the results presented here.

As part of this study, the detector has been compared to a number of leading edge detectors and its performance has been subjectively evaluated.

Due to the aforementioned issues with saliency and ground-truth in edge detection, no empirical testing can be carried out. A valuable piece of future work would be the development of an annotated ground-truth dataset focussing on details and continuous edges in images, to allow for the empirical evaluation of edge detectors of this kind.

The main goal of this project as a whole was to develop an edge detector which:

- Evaluates directional continuity
- Strengthens smooth continuous edges
- Filters and suppresses discontinuous non-salient edges

The results outlined in Chapter 5 above show that the developed detector satisfies these criteria. This work therefore serves to prove the validity of large-filter continuity metrics as a means to improving edge detection results and, in addition, opens new avenues of further research in this field.

Bibliography

[1] P. Arbelaez, C. Fowlkes, and D. Martin. *The Berkeley Segmentation Dataset and Benchmark*. 2007. URL: https://www2.eecs.berkeley.edu/Research/Projects/CS/vision/bsds/.

[2] P. Arbelaez et al. "Contour Detection and Hierarchical Image Segmentation". In: *Ieee Transactions on Pattern Analysis and Machine Intelligence* 33.5 (2011), pp. 898–916. ISSN: 0162-8828. DOI: 10.1109/tpami.2010.161. URL: %3CGo%20to%20ISI%3E://WOS:000288677800004.

[3] J. Canny. "A Computational Approach to Edge Detection". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* PAMI-8.6 (Nov. 1986), pp. 679–698. ISSN: 0162-8828. DOI: 10.1109/TPAMI.1986.4767851.

[4] P.E Danielsson and O. Seger. "Generalized and Separable Sobel Operators". In: *Machine Vision for Three-Dimensional Scenes* (1990). Ed. by Herbert Freeman, pp. 348–75.

[5] P. Dollar, Zhuowen Tu, and S. Belongie. "Supervised Learning of Edges and Object Boundaries". In: *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*. Vol. 2. 2006, pp. 1964–1971. DOI: 10.1109/CVPR.2006.298.

[6] R. Duda and Hart P. *Pattern Classification and Scene Analysis*. John Wiley and Sons, 1973, pp. 271–2.

[7] Pedro Felzenszwalb and David McAllester. "A min-cover approach for finding salient curves". In: *Computer Vision and Pattern Recognition Workshop, 2006. CVPRW'06. Conference on*. IEEE. 2006, pp. 185–185.

[8] Gideon Guy and Gérard Medioni. "Inferring global perceptual contours from local features". In: *International Journal of Computer Vision* 20.1 (1996), pp. 113–133.

[9] X. Hou, A. Yuille, and C. Koch. "Boundary Detection Benchmarking: Beyond F-Measures". In: *2013 IEEE Conference on Computer Vision and Pattern Recognition*. June 2013, pp. 2123–2130. DOI: 10.1109/CVPR.2013.276.

[10] Scott Konishi et al. "Fundamental bounds on edge detection: An information theoretic evaluation of different edge cues". In: *Computer Vision and Pattern Recognition, 1999. IEEE Computer Society Conference on*. Vol. 1. IEEE. 1999, pp. 573–579.

[11] Julien Mairal et al. "Discriminative sparse image models for class-specific edge detection and image interpretation". In: *Computer vision–ECCV 2008* (2008), pp. 43–56.

[12] M. Maire et al. "Using contours to detect and localize junctions in natural images". In: *IEEE Conference on Computer Vision and Pattern Recognition*. Proceedings - Ieee Computer Society Conference on Computer Vision and Pattern Recognition. NEW YORK: Ieee, 2008, pp. 611–618. ISBN: 978-1-4244-2242-5. URL: %3CGo%20to%20ISI%3E://WOS:000259736800080.

[13] David Marr and Ellen Hildreth. "Theory of edge detection". In: *Proceedings of the Royal Society of London B: Biological Sciences* 207.1167 (1980), pp. 187–217.

[14] D. R. Martin, C. C. Fowlkes, and J. Malik. "Learning to detect natural image boundaries using local brightness, color, and texture cues". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 26.5 (May 2004), pp. 530–549. ISSN: 0162-8828. DOI: 10.1109/TPAMI.2004.1273918.

[15] D. Martin et al. "A Database of Human Segmented Natural Images and its Application to Evaluating Segmentation Algorithms and Measuring Ecological Statistics". In: *Proc. 8th Int'l Conf. Computer Vision*. Vol. 2. July 2001, pp. 416–423.

[16] S. Menet, P. Saint-Marc, and G. Medioni. "Active contour models: overview, implementation and applications". In: *1990 IEEE International Conference on Systems, Man, and Cybernetics Conference Proceedings*. Nov. 1990, pp. 194–199. DOI: 10.1109/ICSMC.1990.142091.

[17] Bruno A Olshausen and David J Field. "Sparse coding with an overcomplete basis set: A strategy employed by V1?" In: *Vision research* 37.23 (1997), pp. 3311–3325.

[18] Giuseppe Papari and Nicolai Petkov. "An improved model for surround suppression by steerable filters and multilevel inhibition with application to contour detection". In: *Pattern Recognition* 44.9 (2011). Computer Analysis of Images and Patterns, pp. 1999–2007. ISSN: 0031-3203. DOI: http://doi.org/10.1016/j.patcog.2010.08.013. URL: http://www.sciencedirect.com/science/article/pii/S0031320310003997.

[19] X. Ren and L. Bo. "Discriminatively trained sparse code gradients for contour detection". In: vol. 1. cited By 74. 2012, pp. 584–592. URL: https://www.scopus.com/inward/record.uri?eid=2-s2.0-84877752264&partnerID=40&md5=fb45e664544a7049346547e1e2102816.

[20] Xiaofeng Ren. "Multi-scale improves boundary detection in natural images". In: *Computer Vision–ECCV 2008* (2008), pp. 533–545.

[21] Lawrence Gilman Roberts. "Machine perception of three-dimensional soups". PhD thesis. Massachusetts Institute of Technology, 1963.

[22] H. L. Tan, S. B. Gelfand, and E. J. Delp. "A cost minimization approach to edge detection using simulated annealing". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 14.1 (Jan. 1992), pp. 3–18. ISSN: 0162-8828. DOI: 10.1109/34.107010.

[23] J. Wu, G. Xu, and D. Chen. "One Valid Framework of Integrating Intensity Discontinuity and Edge Continuity for Image Straight-Line Extraction". In: *2009 First International Conference on Information Science and Engineering*. Dec. 2009, pp. 1356–1359. DOI: 10.1109/ICISE.2009.803.

[24] Hui Zhang et al. "Orientation contrast model for boundary detection". In: *Journal of Visual Communication and Image Representation* 25.5 (2014), pp. 774–784. ISSN: 1047-3203. DOI: http://doi.org/10.1016/j.jvcir.2014.01.011. URL: http://www.sciencedirect.com/science/article/pii/S1047320314000121.

Appendix A

Neighbour Voting Details

Directionality was originally defined as

$$D_n = cos^{\alpha}(\theta_c - \theta_n)$$

where θ refers to the gradient orientation, with c denoting the central pixel and n denoting a pixel in the local neighbourhood. Further, we set any negative values of D_n to 0 as we don't want to punish pixels which have other edges in their local neighbourhood (ones that have orientations opposite the orientation of the pixel for instance).

Similarly, collinearity was defined as

$$C_n = cos^{\beta}\left(\theta_n + \frac{\pi}{2} - \phi_n\right) = sin^{\beta}\left(\theta_c - \phi_n\right)$$

where β performs similarly to α above and ϕ_n refers to the angle between the n^{th} pixel and the central pixel. The addition of $\frac{\pi}{2}$ is because the edge continues perpendicular to the gradient orientation. As above, negative values are set to 0 — the existence of an edge at a point at right angles to the expected point shouldn't punish that pixel. However, in this case we wish to treat angles π rads away from each other the same as it is irrelevant where on the line the pixel is, merely that it is on the same line. To do this we translate the difference angles so that they are always in the region $-\pi < \theta < \pi$.

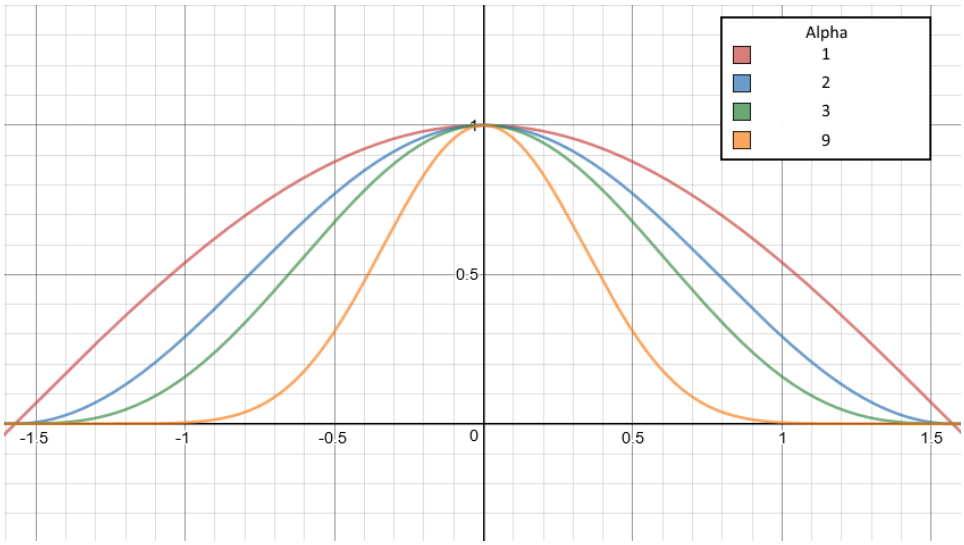
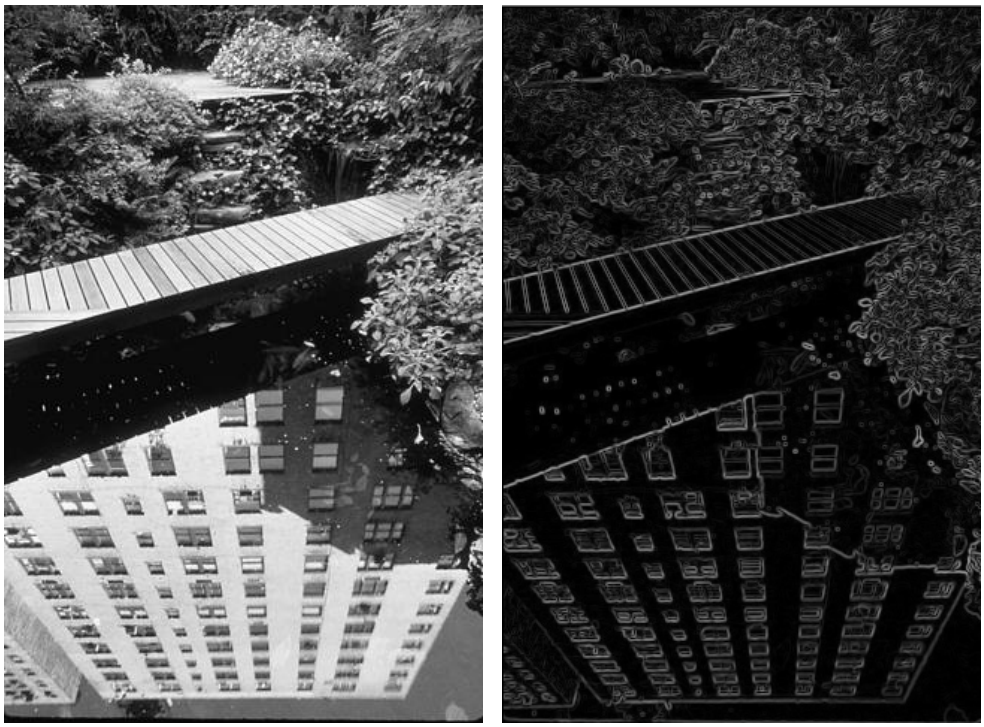
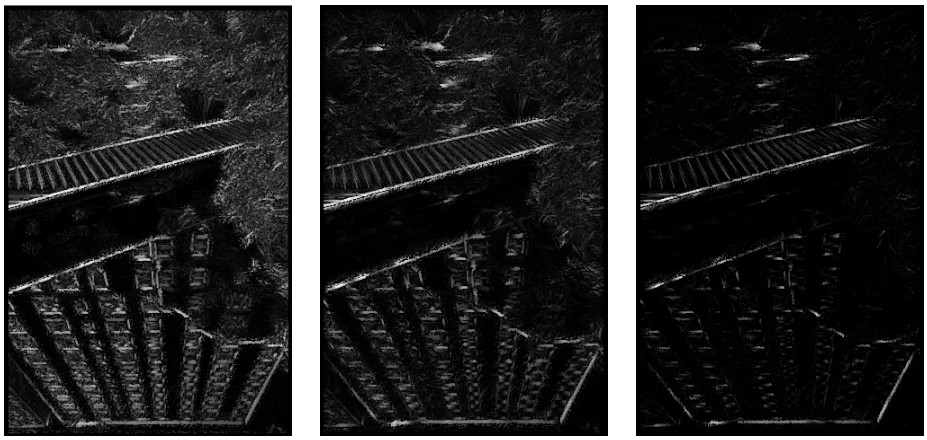


Figure A.1: The effects of the α parameter, with angle in radians along the y-axis



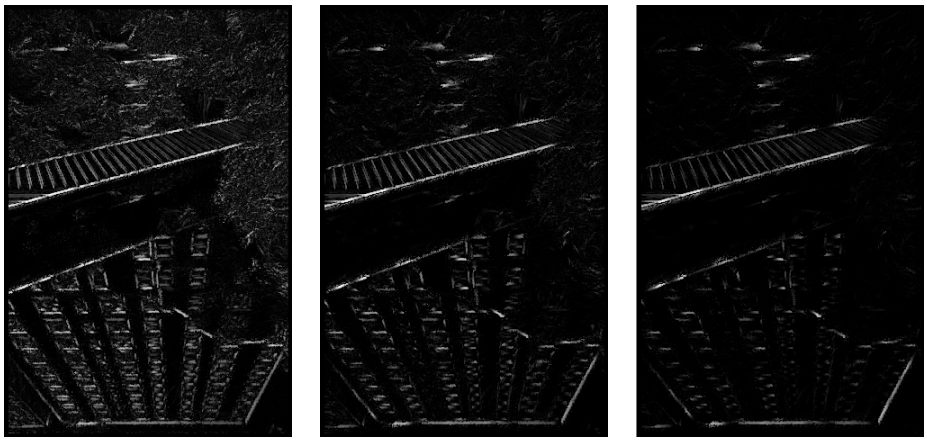
(a) (b)

Figure A.2: Image #62 from the BSDS, along with its edges from Sobel



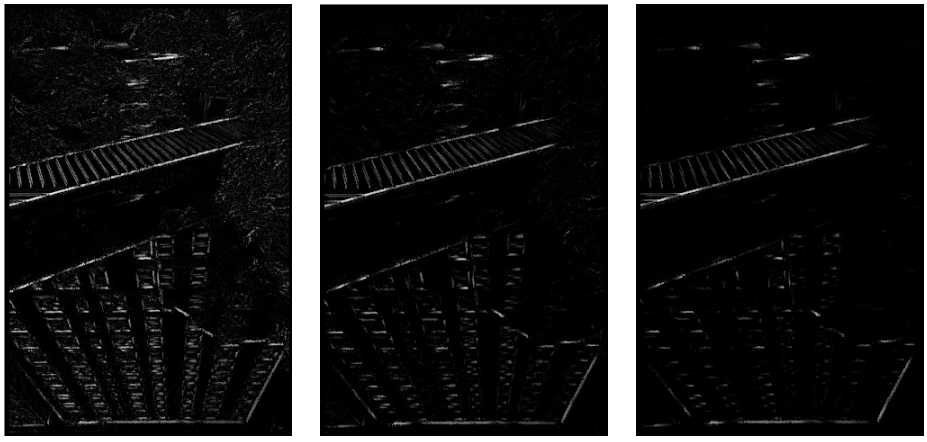
(a) (b) (c)

Figure A.3: The results of applying the algorithm once, twice, and three times over a local neighbourhood of 11×11. $\alpha = \beta = 3$



(a) (b) (c)

Figure A.4: $\alpha = \beta = 9$



(a) (b) (c)

Figure A.5: $\alpha = \beta = 21$

Note that the results seem to get better for the first iteration with higher values of α and β. This was presumably because of the long "tails" of the cosine function when raised to such a high power. Quadratic functions do not suffer from these long tails, so similarly scaled quadratics were found and tested.

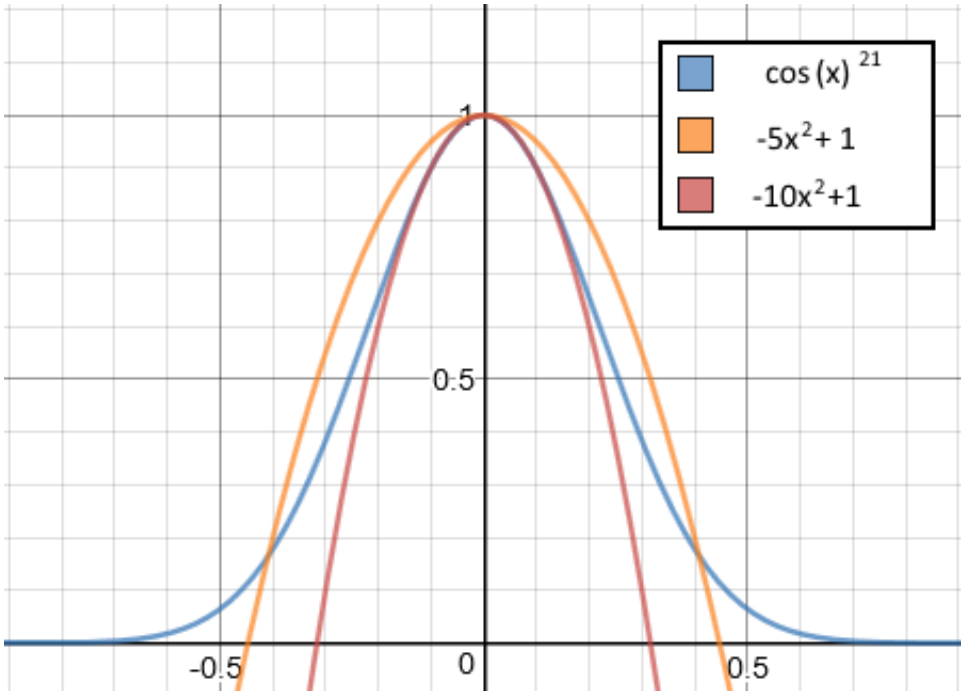
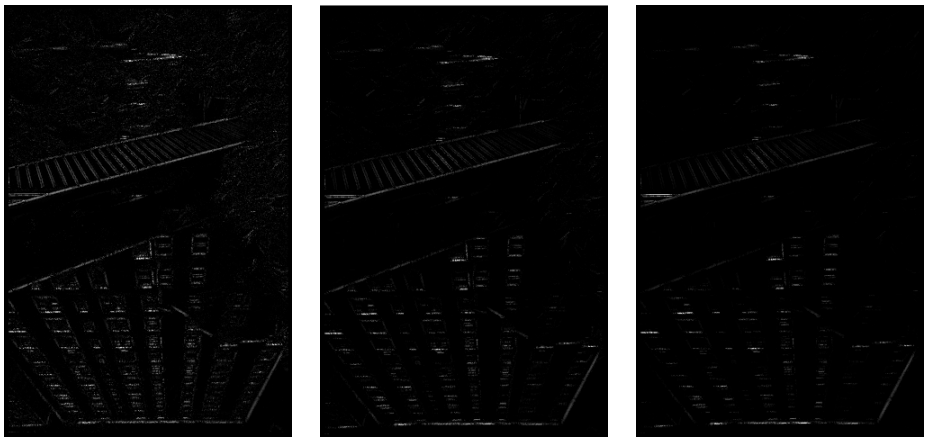
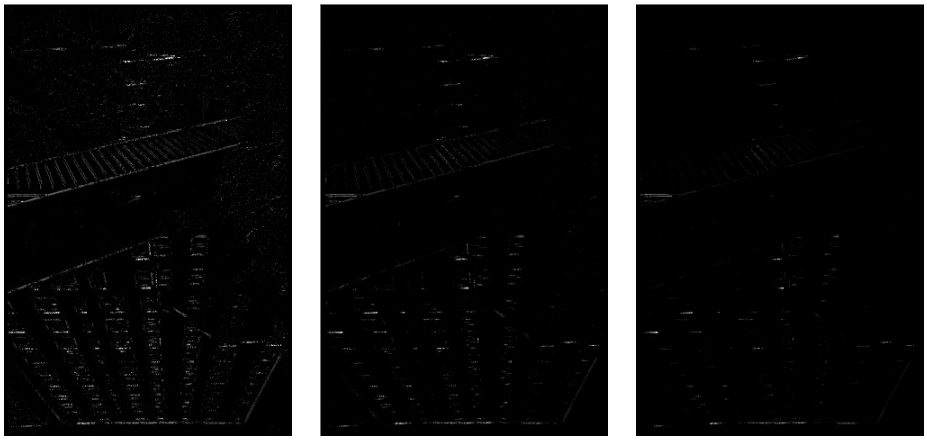


Figure A.6: A comparison between the cosine-based function and the quadratics.



(a) (b) (c)

Figure A.7: Results with the quadratic algorithm, coefficient = 5



(a) (b) (c)

Figure A.8: Results with the quadratic algorithm, coefficient = 10

For these results, iteration proves to be less useful as we tend to lose fine detail too quickly. However this has the benefit that we get better results on the first iteration.