USGS
science for a changing world

A New Method of Edge Detection for Object Recognition

By Brian G. Maddox and Benjamin Rhew

Open-File Report 2004-1325

CONTENTS

KEY WORDS

Directed Edge Detection Object Recognition

ABSTRACT

Traditional edge detection systems function by returning every edge in an input image. This can result in a large amount of clutter and make certain vectorization algorithms less accurate. Accuracy problems can then have a large impact on automated object recognition systems that depend on edge information. A new method of directed edge detection can be used to limit the number of edges returned based on a particular feature. This results in a cleaner image that is easier for vectorization. Vectorized edges from this process could then feed an object recognition system where the edge data would also contain information as to what type of feature it bordered.

INTRODUCTION

Edge detection is the first step taken in many object recognition applications. It basically examines an image and produces a pixel at the boundary of two "colors". The output image contains all of the pixels that were created during the detection. There are many different algorithms that can perform edge detection, each with their own strengths and weaknesses.

The downside to traditional edge detection is that it can sometimes work too well for a given application. Depending on how "busy" an image is, an extremely large number of edges might be returned. In addition, some of the edges can actually be representative of noise in the original image. This large number of edge pixels can then make certain algorithms and analysis difficult as it becomes hard to identify any particular feature based on the edge data.

A new method of edge detection that reduces the number of edges so that only the pixels that border certain features will be returned may be possible. This technique is initially called directed edge detection in that it specifies in advance what types of object edges are of interest. While a small amount of noise would still be present, applications such as vectorization would be easier as there is not as much extraneous data present outside of the current search parameters.

BACKGROUND

Edge Detection

Edge detection has been studied for many years and applied to many different activities. It is usually a raster operation that creates an output image that contains the various edges in an input image. Edge detection techniques "focus on identifying continuous adjacent pixels which differ greatly in intensity or colour, because these are likely to mark boundaries, between objects, or an object and the background, and hence form an edge" (Staff 2002).

Edges themselves are boundaries of object surfaces in a scene that often lead to oriented, localized changes in intensity in an image (Heath 1996). Edge detectors come in a variety of forms that perform searches based on various orientations: horizontal, vertical, or some combination of the two. A good edge detector is one that searches all orientations to determine if there is a localized change in intensity.

There are two methods to performing an edge detection. The first, and less common,

is based on the Fourier Transform and involves a high-pass frequency filter that is run in the Fourier Domain. The Fourier Domain is a representation of temporal and spatial signals that is created by decomposing them into sinusoidal signals. The signal is a complex-valued function of the frequency or wave number including the amplitude and the phase of the sinusoidal signal at the corresponding frequency or wave number. The representation of a signal is equivalent to its spatial representation (Jähne 1997).

The second and most common method of performing an edge detection, however, involves processing the image in the spatial domain. As edges are strong localized changes in intensity, they can be found by calculating the derivatives of the image at any given point to find the maximum of the first derivatives or the zero-crossing location of the second derivatives. The second derivative is the more accurate of the two. However, it is not as commonly used as the first derivative method due to its complexity.

In image processing, the derivatives are found by applying a convolution kernel to the the image pixels. Convolution is a mathematical technique of combining two arrays of different sizes but with the same dimensionality to produce a third array with the same dimensionality as the first two. A full description is beyond the scope of this paper, but can be found in many matrix algebra texts. Both derivatives must be estimated as there is no "function" to differentiate. The first derivative is usually estimated by convolving the original function (or point in this case) with [-1 1]. The second derivative is then estimated by convolving with [1 -2 1] (*Edge Detectors* 2000). These derivatives are applied by turning the pixels into a vector array of values and convolving them into some output array of values. Different edge detectors then use this technique to calculate the derivatives of each point in an image. This is done by applying a specific kernel to the pixel, where a kernel is a matrix (generally 3x3) that is multiplied against a vector array of, say, Red, Green, and Blue (RGB) values for the pixel.

There are two main classifications that edge detectors can fall into based on the method they use to find the derivatives: Prewitt compass and gradient (*Edge Detectors* 2000). Prewitt compass edge detectors convolve the image with a set of eight kernels. Each kernel is created to detect a specific edge orientation. The kernel that produces the highest values then determines the orientation of the edge. Gradient-based detectors convolve the image only horizontally and vertically. The absolute magnitude of the gradient is then calculated based on these values. Gradient is the more widely used technique of the two, and is the basis for commonly known edge detectors such as the Sobel and Canny methods. The gradient method is discussed here as it is also the basis for the directed edge detection technique.

With the gradient technique, once the magnitude of the first derivative is calculated, the pixels that make up the edge must still be identified. A common method to do this is to

create an image comprised of the gradient values and then threshold that image. Thresholding is used to find areas where the gradient is greater than or equal to a specific value that is assumed to represent an edge. This method usually generates edges that are several pixels wide. Another method is to find the local maxima of the gradient image. This technique generally produces smaller lines (usually a single pixel in width) but can sometimes overlook edges.

Other edge detectors use the second derivative of the image in order to calculate the gradients. These detectors are also called *zero crossing* as they locate the point where the second derivative of the edge would cross over the zero value line (*Edge Detectors* 2000). These derivatives are usually calculated by a Laplacian filter. This provides an advantage in that the edge magnitude is calculated independently of the edge orientation through the use of a single kernel.

The next two figures illustrate the output from a gradient-based edge detection algorithm. Figure 1 is a portion of a digital orthophoto quadrangle (DOQ), while Figure 2 is the edge image after the Canny detection method was applied. An interesting side note about the Canny edge detector is that it is a bit more sophisticated in finding the local maximum values of the gradient by using non-maximal suppression and hysteresis tracking. These techniques allow Canny to better track edges and to alleviate the problem of noisy edges being broken up into multiple segments.

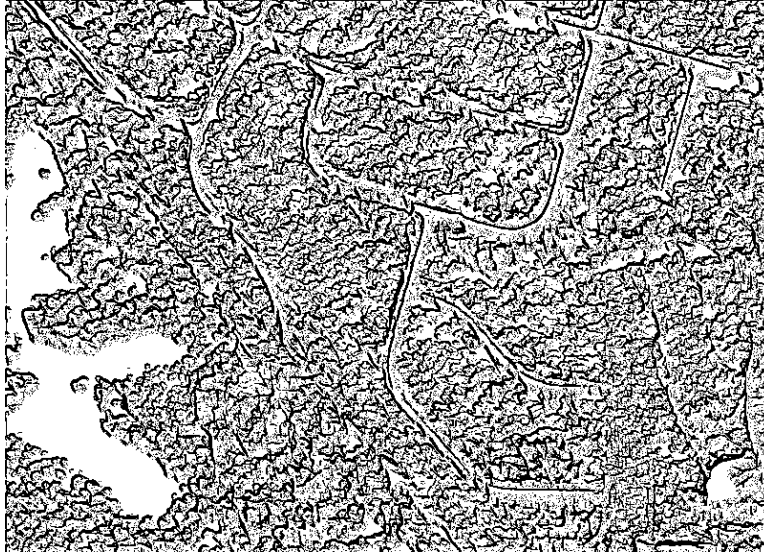Figure 1. Original Input Image

Figure 2. Canny Edge Detection

Figure 2 illustrates some potential problems that can be found with edge detection. Noise sensitivity is a big issue due to the mathematics involved. Calculations of the derivatives in the spatial domain tend to attenuate high frequency information. This attenuation magnifies noise that is present in a normal image. Some edge detectors such as Canny try to address this problem by applying a Gaussian smoothing first.

There is also a problem with the large numbers of edges returned. This can be mitigated to a certain degree by the choice of the threshold, but large numbers of edges due to noise as well as actual edges will always be present. Detection of certain features is difficult in this case as there are so many edges that some tend to run together instead of remaining separate.

Hough Transform

The Hough Transform is a technique that is used to isolate certain features of a particular shape in an image (*Hough Transform* 2000). For vectorization purposes, the Hough Transform is usually run. The classic algorithm fits a line through discreet sets of image points by using a parametric line equation and transformation to Hough parameter space. For detection, it uses edge information to fit the points to a straight line that would pass through the edges. A more detailed description can be found in the Hough Transformation web page cited in this document. Once the Hough Transform is complete, these lines could then be separated and vectorized.

The classic Hough Transform is generally only practical for simple curves or lines. Noise and large numbers of edges in an image can interfere with the classic algorithm. This can be seen in Figure 3, which is the output of the Canny edge detector in the above image with a Hough Transform applied.

Figure 3. Canny Detector and Hough Transform

The large number of edges in the image interfere with the actual Hough Transform. In this case, straight lines are fit to edges that do not exist. The actual features with straight lines (the roads) do not even have lines fit through them. Figure 4 contains just the Hough Transform drawn lines.
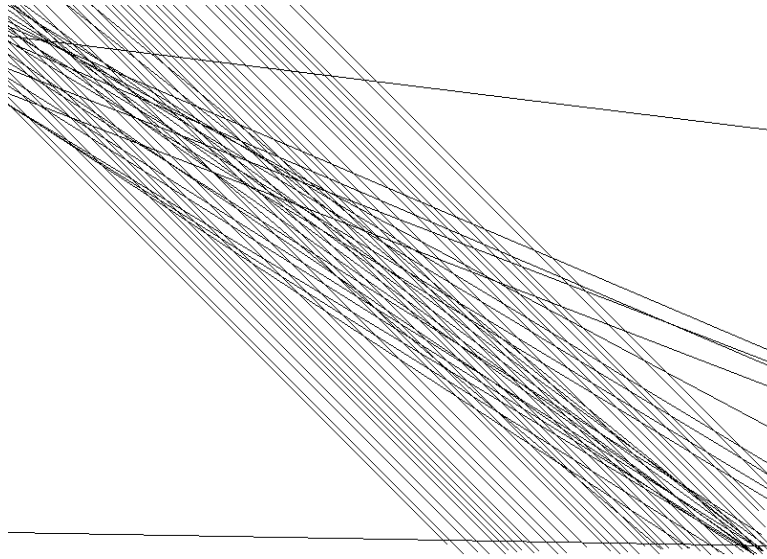
Figure 4. Separated Hough Transform Lines

When compared to Figure 1, the Hough Transform lines do not conform to any known features in the image. This is useless for vectorization purposes, as it does not follow the roads that do contain actual lines. Standard edge detection techniques fail when presented with such a "busy" image.

METHODS AND TESTING

A different method can be used in performing the edge detection that would alleviate this problem. By directing the edge detection, only the edges that border specific features would be returned. This would greatly reduce the image noise and other extraneous edges in the image, and would benefit edge detection. Work was done at the Mid Continent Mapping Center in fiscal year 2003 to develop and test this idea.

The first step to this new method would be to examine the output from the specific sensor that was used to create the input image. This would be a data collection activity that would manually examine the spectral responses of specific features from various images. For example, in RGB space this would involve recording all of the RGB values that were found that make up a certain type of road in an image taken from a specific type of sensor. A database would then be created that would contain the responses gathered from the data collection. This database could be used to gather statistics such as the mean value that represents a type of road, and distribution information such as the various standard deviations from that mean and how many values they contain.

Once this data is collected, it could then be used to modify the traditional method of performing a gradient edge detection. As previously mentioned, calculating the overall

9

gradient of a pixel involves first finding the horizontal and vertical components of the gradient. The feature-specific spectral response information collected could be used to create another threshold at the beginning of the gradient detection. In this case, the mean value and standard deviations from this value could be used to create a subset in the image color space that would contain the pixels that represent a specific type of feature. This step would have to be done at the beginning of the gradient calculation, as the gradient image that is used to calculate the local maxima does not contain any of the original color information.

For example, consider the RGB image case with a road feature being the focus of the edge detection. RGB space can be thought of as a cube structure, where one corner represents the value (0,0,0) and the the opposite represents (255,255,255). The mean value would represent the average color value for that road, and the standard deviations could be used calculate the distance from the mean in RGB color space (find which standard deviation contains 75 percent or so of the values). This would then create a sphere where every value in that sphere would be present in the road portion of the image. When the gradient is being calculated, the colors would first be checked to see that they fall within the RGB road subset. If so, the gradient would be calculated for that pixel. If not, the gradient would be skipped and the edge along that pixel ignored.

Figure 5 illustrates the output of a modified Canny edge detector that was directed to output the edges that bordered roads in the sample image. This step did not have an exhaustive statistical analysis of the spectral response values that made up the road. Instead, a simplistic method of finding and averaging around twenty pixels was used.

Figure 5. Directed Canny Detection for sample roads

As seen in Figure 5, the number of edges in the image has been drastically reduced. The road outlines are clearly visible in the image. Some noise has also been introduced into the image, partly due to the standard operation of the Canny method and also due to the simplistic method used to create the spectral response thresholds. The Hough Transform was then applied to this image with the results in Figure 6.
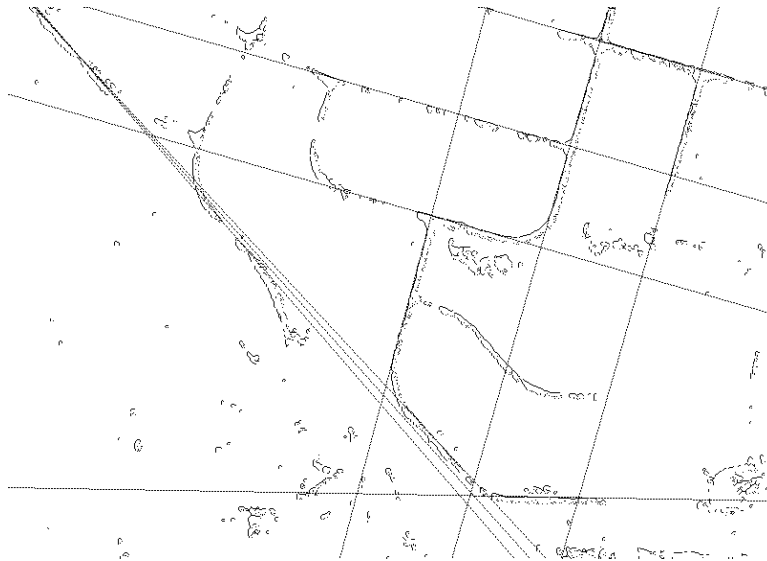

Figure 6. Directed edge with Hough Transform

The Hough Transform did indeed perform better with the directed edge detection. However, the reduction in noise and edges produced different defects in the image than before. The Hough Transform algorithm tends to fit the longest line possible through edges that it finds. It will typically go to the boundaries of an image when it is creating a line. While many of the lines now fit the roads that are in the image, they extend to the edge because the algorithm found pixels between the true end of the road segment and the image boundary.

Some work was then done in modifying the Hough algorithm to better fit the reduced set of edges. This was needed because sometimes two or more edges would contribute to one line, and that line would then be drawn across the entire image. Instead of one line, multiple lines corresponding to each of the edges should have been drawn. To accomplish this, the Hough algorithm's line-drawing function was modified so that the lines would be drawn only where edges already existed in the image. This was done by making the line-drawer check to see if there were any edge pixels a certain distance

11

ahead of the current line-drawing point. A threshold parameter was also added, so that the distance the line-drawer looked ahead could be controlled. As Figure 7 shows, the result is a much cleaner looking image.

Figure 7. Canny with some Hough modifications

The other major modification to the Hough algorithm was to make it more of a "fuzzy" algorithm, to improve the quality of the lines it detects. The basis of the idea is that sometimes the edges produced by the edge detection algorithm are not distinct as the edge pixels are not quite in a straight line. These are valid edges, but the normal Hough Transform will not assign a high enough value to the lines produced from the edges. To alleviate this problem, the algorithm was modified so that when it is considering a line, it looks at the four pixels at right angles to the line, as well as the default center pixel. For each edge pixel, one is added to that line's counter, with the result that more edge pixels are contributing to the line, giving it a higher value. There is at least one minor drawback. If an image has a lot of parallel lines close together, the detector, as it currently stands, will detect an additional line at a right angle to the parallel lines. Figure 8 is an image with the lines drawn by this modified Hough Transform where the edges are omitted.
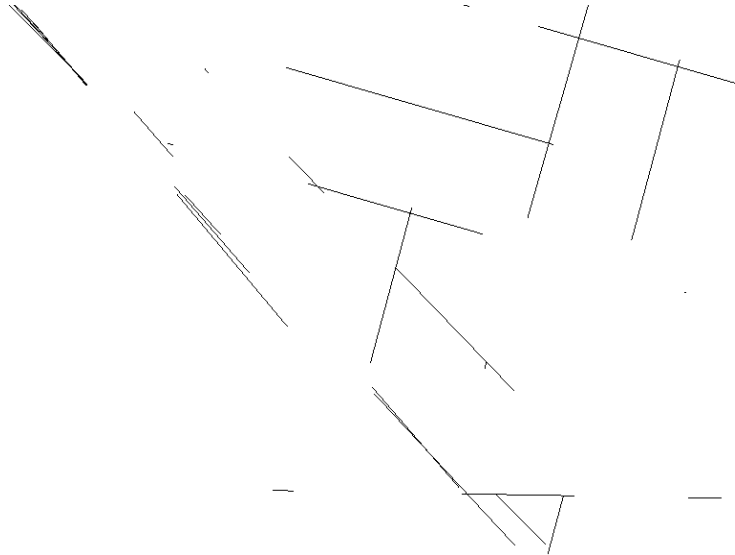
12

Figure 8. Hough Transform Lines from Directed Canny

The lines drawn by the experimental modifications to the Hough Transform in Figure 8 are much cleaner and more accurate than those in Figure 4. One can see that even though the Hough modifications are still a work in progress, vectorization would be much cleaner in this case, and result in a more accurate fit with the actual road objects. The line following algorithms that are used in vectorization routines would be able to easily convert Figure 8 to a series of vector endpoints.

DISCUSSION

As can be seen from the above figures, the technique of directed edge detection can be applied to current edge detection methods in order to improve their usability for extraction purposes. An alternative way of implementing the RGB directed edge detection starts with computing the gradient image as normal. The difference comes during the edge detection process. Before the detector checks the gradient image to determine if a pixel is on an edge, it checks that pixel and the eight surrounding pixels in the original image. If one of those pixels matches the RGB color sphere, then the gradient check goes on as normal. If not, then there is no edge pixel returned, even if the gradient image shows that an edge exists. This detection method does not directly return edges based on a transition from inside to outside the color sphere, but the effect is almost the same.

These techniques have many advantages that can be used with object recognition systems. Many of these systems require an image to be segmented in some form before various algorithms or recognition techniques can be run against the image.

13

Traditional edge detection produces large numbers of edge segments, some of which are neither discreet nor are they continuous. Vectorization processes such as the Hough Transform do not work well with these large numbers of edges. Plus, the object recognition algorithms must be made to continually sort though the edge segments in order to try to determine the identity of an object. This places a large burden on the recognition algorithms and techniques currently in use.

The benefit to this edge detection process is that vectors returned would be vectors that are known to border specific objects. This would help object recognition processes as, instead of dealing with raw vectors, the recognition would already have some data about what types of edges they were processing. Additionally, instead of vectors from each object in the image, this method would only return the ones the recognition system was specifically targeting. Consider a recognition system where all of the edges returned had some form of a pre-classification. Vectors returned from Figure 8 would already have a tag that they bordered a road. Such a system could then immediately switch to a set of heuristics to determine what type of road was present (gravel, paved, etc). Without this information, the first step of the system would be to arrange the edges and try to deduce what type of object they bordered. Pre-classification eliminates several steps from the general recognition process, and can improve accuracy as the initial edge determination step is very complex and prone to error.

Modifications to existing vectorization techniques can also be used to improve the accuracy of recognition systems. The Hough Transform, for example, can perform well at drawing a line through an object edge. However, as has been demonstrated previously, it is currently not well suited for good vectorization purposes as it cannot follow small line segments well. A change in algorithm design is required so that numerous smaller line vectors rather than larger single vectors are produced. The smaller vectors can better follow the outline of an object and can be easily merged together mathematically in order to produce the full boundary of an object in an image.

FUTURE WORK

A more in-depth study of spectral response patterns should be performed in order to generate the object classifications for the gradient detection. Time allotted in fiscal year 2003 only allowed a simplistic averaging and small data gathering scheme on an RGB image. A full analysis is necessary to determine the parameters for acceptance and rejection in the directed edge detection technique.

Further modifications to the Hough Transformation are necessary to better generate lines that could be vectorized. One possible modification could be to turn the algorithm

into a two-pass process. The first pass would be to run a modified algorithm that would only generate small line segments instead of trying to fit the longest line possible. Reducing the segment size could help to better fit edges without overshoot error. In the example of roads, smaller segments could better follow turns in a road. The output of this pass would be placed into a temporary storage area that only contained the line segments and none of the original edge data. The second step of the modified algorithm would be to then connect the various line segments into longer lines. Line-following techniques could be used to determine whether or not a segment actually exists on the same plane as the others. Other mathematical techniques could then be used to connect the endpoints of the various edges together.

CONCLUSION

Current methods of edge detection produce so many edges that they are not the best choices for input into a recognition system. When combined with noise, large numbers of edges can interfere with line-fitting algorithms that are intended to generate output for vectorization purposes. The information gathered using traditional techniques can lead to both false and missed identifications.

A new method exists that can return edges for feature-specific data. This technique requires a statistical database to be created ahead of time for object spectral responses from specific types of remote sensors. This database is then used during the edge detection process to accept or reject edges based on whether or not they fall within the range of spectral responses for a feature of interest. The final output would contain far fewer extraneous edges and noise than a traditional method.

By returning specific edges and less noise, techniques such as the Hough Transform can be used to plot curve-fit lines through the various edges. These lines would be much more accurate and cleaner than a classic Hough Transform on data with large numbers of edges. Some deficiencies in the classical algorithm must be corrected, however, in order to more accurately follow the edges in an image. Specifically, the algorithm must be modified to limit how far it will go when trying to fit a line through the edges.

These techniques can then be fed into an object recognition system. The edges returned would already be encoded with the type of object they border. Vectorization routines could use the output of a modified Hough Transform to vectorize line segments drawn through the edges. A recognition system could then use this information to determine the identity of specific features in an image. Pre-classification could benefit object recognition systems as they would not have to assemble various edges ahead of

time and then try to determine identity based on shape alone.

REFERENCES

Heath, Michael D. *A Robust Visual Method for Assessing the Relative Performance of Edge Detection Algorithms*. University of South Florida Department of Computer Science and Engineering. 1996.

Jähne, Bernd, 1997, *Image Processing for Scientific Applications*. CRC Press,

Fisher, Robert, Simon Perkins, Ashley Walker, and Erik Wolfhart. *Glossary – Edge Detectors*. <http://www.dai.ed.ac.uk/HIPR2/edgdetct.htm>. Hypermedia Image Processing Reference. 2000.

Fisher, Robert, Simon Perkins, Ashley Walker, and Erik Wolfhart. *Image Transforms – Hough Transform*. <http://www.dai.ed.ac.uk/HIPR2/hough.htm>. Hypermedia Image Processing Reference. 2000.

Staff, Chris. *CSA3020 Lecture 5 – Images and Graphics*. <http://www.cs.um.edu.mt/~cstaff/courses/lectures/csa3020/mm5.html>. University of Malta. 2002.