MVA'94 IAPR Workshop on Machine Vision Applications Dec. 13-15, 1994, Kawasaki

Avoiding Redundant Processing in Gradient Based Edge Detection

Martin J.J. Scott, Richard W. Prager
Cambridge University Engineering Department
Trumpington Street, Cambridge, England.

ABSTRACT

Gradient based edge detection produces edge location information by convolving an image with a kernel to calculate local intensity gradients. In many images, the area of the image that contains edge location information is small relative to the area of the image as a whole. During edge detection areas all areas of an image are processed. This leads to the redundant processing of areas that contain no edge information. In this paper an algorithm is described that combines two different approaches to image segmentation. The goal of the algorithm is to identify areas of the image that do not contain edges. This information may then be used to drive selective edge detection which would avoid redundant processing.

I. INTRODUCTION

After capturing a digital representation of an image, the first stage of image analysis is segmentation. Segmentation is the task of discovering the boundaries of regions within an image, the end result being an image that may be entirely segmented by these regions. In a real time image analysis system, improvements to the segmentation stage of analysis could mean reducing processing time while still producing satisfactory results.

Many image analysis systems use gradient analysis to detect edges during segmentation. In some images edge detection can produce redundant processing, when the amount of actual edge information in the image is small. A method for selecting areas of the image to be processed for edge detection would avoid areas of the image containing no edge data. Such a method would reduce the processing time required for edge detection, but would not reduce the accuracy of the segmentation.

In this paper an algorithm for combining two quite different segmentation schemes is proposed. Firstly a low resolution technique is employed to discover areas of the image that may contain significant edges. This location information is then used to drive high resolution edge detection. This technique can produce computational savings of up to fifty percent in some instances.

II. THEORY

The underlying goal of image analysis is to determine *what is where*. This is true at all levels of image analysis, including image segmentation where one wishes to classify all the points in the image into regions. In an image the degree of certainty that can be attached to *what* region a point belongs to is called the class resolution. *Where* that point is is the spatial resolution. Unfortunately these two are incompatible.

The algorithm described in this paper eliminates redundant processing during edge detection. Two different types of segmentation are combined to do this. Firstly an initial segmentation is carried out using thresholding; this has a high class resolution. Information from this initial segmentation is used for edge detection which has a high spatial resolution. In this manner the strengths of both segmentation techniques are used.

In this section a brief overview of edge detection and threshold segmentation will be given, along with a method for improving the class resolution of an image.

A. Gradient Based Edge Detection

An image may be viewed as a discreet representation of an intensity function. Edge detection is carried out by differentiating the entire image, and then looking for points that have a high gradient. A high gradient indicates a rapid change in intensity. This rapid change would be expected at the transition from one region to another. Contiguous sets of points with high gradients form edges. These edges may then be followed and joined to create continuous boundaries surrounding regions in the image. When all the boundaries have been formed, the image will be segmented.

The gradient at each point in the image is calculated approximately by convolving the image with a kernel. Different edge detectors, such as the Sobel, Canny, and Marr Hildreth edge detectors, use

different kernels. However the important point to note is that each type uses convolution of the original image with a kernel to calculate the gradient. The operation is local. The gradient information for each point in the image is obtained using intensity information contained in an area the size of the kernel, centred on that point. The percentage of the image that contains edge information, however, is small compared to the total area. This means that a large amount of redundant processing will be carried out, convolving areas of the image which contain no useful information. In an ideal situation, only the area of the image containing the edge information would be convolved; this would be an area covering the edge, as wide as the kernel.

B. Threshold Segmentation

Thresholding is a class based approach to segmentation. That is, a number of regions are defined for an image, and every point in that image is classified as belonging to a particular region. The regions are defined by examining the frequency histogram for the grey levels that occur in the image. A distinct peak in the histogram designates a region. All grey levels that lie between the valleys to the left and right of the peak are said to belong to that region.

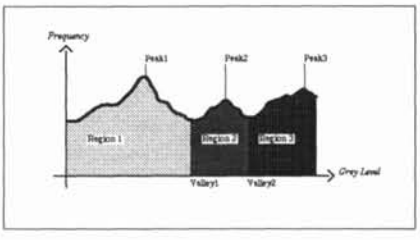

Fig. 1. An example of a grey level histogram. This example contains three distinct peaks and two valleys.

The peaks and valleys are located in the histogram as in Figure 1. The image is segmented by labelling each pixel as region one, two, or three. If the grey level of the pixel lies between the origin and valley one, the label region one is assigned, between valleys one and two, region two, and so on.

C. Quad-tree Image Compression

Threshold segmentation can mis-label at high spatial resolutions. One method for improving the performance of thresholding is to enhance the class resolution of the image. The image may be compressed using a quad-tree structure, which will raise the level of class resolution. As the class resolution increases, the definition of regions in the grey level frequency histogram also increases. Quad-tree compression is essentially a 2×2 local averaging process, see Figure 2. Due to the local averaging that causes the compression, the effects of noise are reduced with each level of the quad-tree.

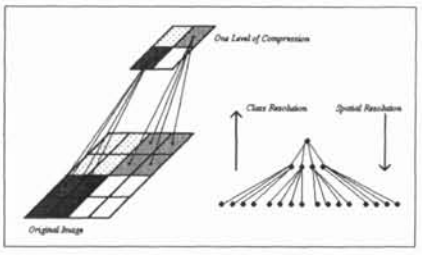

Fig. 2. Quad-tree image compression

As an image is compressed level by level, the spatial resolution lowers as the class resolution rises. This can be seen as an increase in the peakiness of the grey level histogram. A threshold segmentation of a compressed image would gave a higher degree of certainty attached to the classifications of the points in the image. A weakness of this approach is the difficulty in relating the compressed image to the original. In the compressed image the classifications are more accurate, but now the degree of certainty about *where* in the original these classifications relate to is lower.

III. Proposed Algorithm

The aim of this technique is to identify the regions of the image that definitely do not contain an edge. The remaining area of the image may be processed for edge and boundary detection. The algorithm proposed has four parts

1. Quad-tree compression
2. Threshold segmentation
3. Mapping edge areas
4. Use edge area map to drive edge detection

The purpose of the algorithm is to generate the same edge detection results as a standard implementation of a given edge detector. Both edge images will have the same edge information, but the results produced by the proposed algorithm will have been generated using less processing time.

A. Quad-tree compression

The original image is compressed using a quad-tree structure. The resulting compressed image will have a higher class resolution than the original, and the effects of noise present in the original will have been reduced.

The level of compression will be application specific. Each level of the quad-tree causes a 2×2 compression of the image. If *a priori* information about the regions in the image is available this may be used to determine the level of compression. For example take an image of dimension 256×256 pixels where a large single object is being sought. It may be possible to state that any isolated region that may be contained in an 8×8 pixel square is either irrelevant or noise. One could say in this case that it would be safe to compress the image through three levels of the quad-tree. Any more than this and one would risk merging regions that ought to be kept distinct.

Another factor to be considered when deciding upon the level of compression is the area of the original image represented by each point in the compressed image. It will become clear from the explanation of the latter stages of the algorithm why this is important. If the area represented is too large, one risks reintroducing redundant processing.

B. Threshold segmentation

The compressed image obtained from the quad-tree is segmented using thresholding. This segmentation will be class based, i.e. carried out based on a grey level histogram of the compressed image. General thresholding may be employed for this segmentation, where each distinct peak in the histogram designates a distinct region. Alternatively, where some *a priori* information on the number of regions present is available, a hill clustering thresholding algorithm may be used (Tsai and Chen 1992).

C. Mapping edge areas

Once a labelled segmentation of the compressed image has been made, the next stage is to make a map of the segmented image. This map will locate areas that definitely do not contain edges and areas that may contain edges. To explain the uncertainty about edge locations it must be remembered that the goal is to discover the location of the edges present in the original image. The segmented image is a segmentation of a representation of the original image in which the spatial resolution has been degraded. Hence the edge between two regions in the segmented compressed image refers to an area of the original in which one may be sure an edge exists, while at the same time not knowing the exact location of said edge. The task now is to determine which areas of the compressed image represent areas of the original where one is certain no edge exists.

The compressed image is examined from top left hand corner to bottom right. The examination is carried out on non overlapping sections of dimension 2×2. A binary map is constructed which will effectively be another 2×2 compression of the segmented image. Each point on the binary map representing the corresponding 2×2 section of the segmented image. The values in the map correspond to

- $1 \longrightarrow$ *Section may have an edge*
- $0 \longrightarrow$ *Section has no edge*

The determination of which value to be assigned is simple

- 0 if the 2x2 section is homogeneous *and* all of the eight neighbouring sections are homogeneous
- 1 otherwise

Some homogeneous areas of the image are included to allow for the merging that occurs at region boundaries during quad-tree compression.

D. Use edge area map to drive edge detection

The binary map of areas containing edges will now be used to drive edge detection in the full resolution original image. If the compressed image had been compressed through m levels of the quad-tree, then each point in the binary map will represent a square area of dimension $2^m \times 2^m$. The original may now be analysed in sections of this size. If the corresponding point on the edge area map has a value of 0, then no analysis is required for that section. Otherwise, analyse the section with whichever edge operator has been selected. Preprocessing of the image using Gaussian smoothing may also be carried out in this fashion. Kernel overlap problems only occur at the actual image boundary, as overlap in central sections of the image may be overcome simply by using pixel information from neighbouring sections.

IV. Experimental Results

The algorithm was implemented in C, using the GNU complier running under UNIX on a Sun Sparc 10 workstation. All images tested were of dimension 256×256 pixels. The images were grey scale, with a range of 255 different grey levels.

A. Experiment 1.

The first test of the algorithm was carried out on a simple image. The image consists of four hoops, Figure 4. No noise is present in the image. The hoops are solid, with a grey level of 0. The background is solid with a grey level of 255.

The image was compressed through two levels of the quad-tree. Threshold segmentation was carried

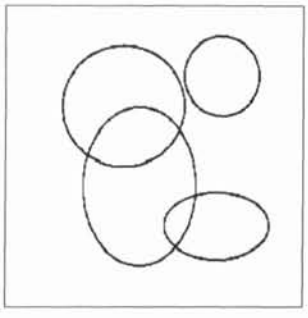

Fig. 3. Original image

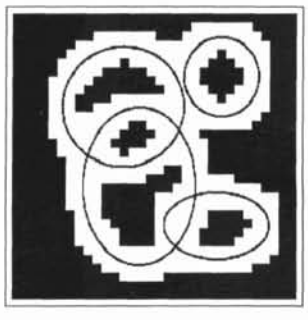

Fig. 4. The edge area map, and the edge area map overlaid on the original image

out. An edge area map is constructed from the segmentation of the compressed image. 55 percent of the original image is determined not to contain any edges.

Selective edge detection is carried out on the original image using a Canny edge detector. The original image is analysed in non overlapping sections of dimension 8×8. The white regions in the edge area map indicate locations in the original where edge detection will take place. Allowing for compression, thresholding and edge area mapping, the processing time for edge detection in the image was reduced by 53 percent using the proposed algorithm.

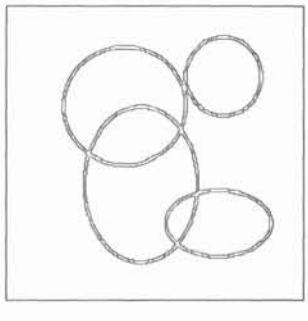

Fig. 5. The edge image produced using selective edge detection

V. Conclusions

In some images, the area of the image that contains edge information is relatively small in comparison with the area of the image as a whole. When carrying out high spatial resolution edge detection, this can lead to redundant processing. In a real time image analysis system that must segment a large number of images the time spent carrying out this redundant processing can present a problem.

In this paper an algorithm is described that uses an initial stage of high class resolution segmentation to drive selective edge detection at high spatial resolution. The aim of the algorithm is to provide a fast method for locating the areas of an image that definitely do not contain an edge. These areas may then be eliminated from the edge detection process, reducing the overall processing time for segmentation.

The experimental results have shown that the algorithm can produce satisfactory results. In a first test a simple hoops image with no noise was processed. The algorithm successfully located all the areas of the image containing edge information. The processing time for edge detection in the hoops image was reduced by 53 percent. A second test used an ultrasound heart scan which contained a large amount of noise. The algorithm successfully eliminated 58 percent of the image from edge detection, producing a total processing reduction of 56 percent. The algorithm proposed in this paper is currently being applied to real time medical imaging, producing segmentations of $2D$ ultrasound scans for $3D$ reconstruction.

References

[1] Wilson, R., Spann, M.. *Image Segmentation and Uncertainty, Research Studies Press, 1988.*

[2] Tsai, D-M, Chen, Y-H. *A Fast Histogram-clustering Approach for Multi-level Thresholding, Pattern Recognition Letters 13, 1992, 245-252.*

[3] Canny, J. *A Computational Approach to Edge Detection, IEEE PAMI-8 6, 1986, 679-688.*