Coarse Classification of Terrain Image Information by Using Sobel Edge Detection Technique

Dr. Arshad Ali

Assistant Professor, Faculty of Computer and Information Systems, The Islamic University of Madinah, Al Madinah Al Munawarah, Saudi Arabia - 42351

Corresponding Author: Dr. Arshad Ali, Faculty of Computer and Information Systems, The Islamic University of Madinah, Prince Naif Ibn Abdulaziz Road, Al Jamiah Al Madinah Al Munawarah, Saudi Arabia – 42351

Mobile: 00966 59 964 2899

Summary
A new approach for the classification of man-made features and natural features in a terrain using a single image is presented. Prominent features that distinguish man-made objects from natural features are identified. Techniques are developed to detect image features and based on the features extracted and use decision based classification to identify regions in an image possessing man-made features. These techniques include feature extraction, edge detection, and classification. Some of these methods are novel and others present unique properties and advantages compared to previous related works. Experimental results are presented using real images, the result clearly identify the regions with man-made features. This technique for terrain classification is developed for air vehicles navigating in a terrain; this technique would assist the air vehicles in efficiently searching/navigating a terrain.

Key words:
man-made features, feature extraction , Edge detection, air vehicles navigating

1. Introduction

Edge detection is one of the most significant parts of this work, as the research work mainly involve classification of terrain images, hence makes image analysis hold great importance in the overall research. In images, edges represent object boundaries and hence are useful for identifications of objects in a terrain. The legitimate importance of edge detection in image processing can be revealed by the fact that; In order to extract the features of an object, we must detect the edges forming that object. In this paper theory on edge detection is presented followed by discussion on results gathered on performing experiments based on the theory. The future research to follow utilize the results from this on; as data information to be extracted highly depends on the nature of edges detected.

2. Related Work

Edge detection is a forefront tool used in most image processing applications. It is one of the most common and widely used operations in image analysis; the process detects outlines of an object and boundaries between objects and the background in the image. For object detection, it is primarily important to have a good knowledge of edge detection. This section presents background knowledge on edge detection and emphasis is made on selecting a technique which is most suited to the project.

Discontinuities in the physical, photometrical and geometrical properties of scene objects provide essential visual information and are known to be physical edges; and they correspond to significant variations in the reflectance, illumination, orientation, and depth of scene surface [4]. Methods of edge detection involve convolving the image with an operator (a 2-D filter), there are a number of operators available and each of the operators are designed to find out specific type of edges, the design of the operators make them to returns zero values in uniform regions while they are sensitive to large gradients in the image [5]. Generally, an Edge detection method or selection of edge detection operator can be divided in to three broad stages:

- **Noise Reduction:** Edge detection is difficult in noisy images, since both the noise and the edges contain high-frequency content. This noise reduction is usually achieved by performing a low-pass filter because the additive noise is normally a high-frequency signal. However, the edges can possibly be removed at the same time because they are also high-frequency signals. Hence, a parameter is commonly used to make the best trade-off between noise reduction and edges information preservation. In order to

Manuscript received January 5, 2018
Manuscript revised January 20, 2018

gain better performance of edge detection, image noise should be reduced as much as possible. [5] [7]

- **Edge Detection:** A high-pass filter such as a differential operator is usually employed to find the edges. The geometry of the operator determines a characteristic direction in which it is most sensitive to edges. Operators can be optimized to look for horizontal, vertical, or diagonal edges. [5] [7]

- **Edge structure:** Not all edges involve a step change in intensity. Effects such as refraction or poor focus can result in objects with boundaries defined by a gradual change in intensity. The operator needs to be chosen to be responsive to such a gradual change in those cases. Newer wavelet-based techniques actually characterize the nature of the transition for each edge in order to distinguish, for example, edges associated with hair from edges associated with a face.

- **Gradient:** In this technique edges are detected by taking the first derivative of the image, and then by looking for maximum and minimum in the resulting derivative of the image. [4]

- **Laplacian:** The Laplacian methods search for zero crossings in a second-order derivative expression computed from the image in order to find edges, such as the Laplacian or a non-linear differential expression. [7]

An edge has the one-dimensional shape of a ramp and calculating the derivative of the image can highlight its location. Suppose we have the following signal, with an edge shown by the jump in intensity below [12]:

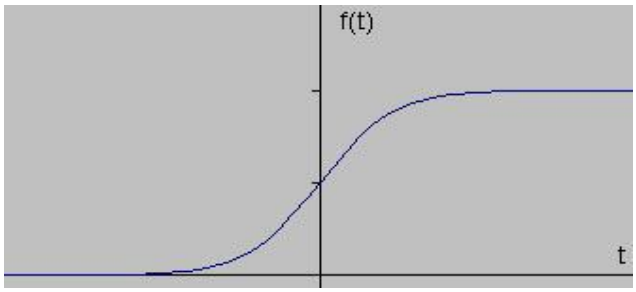

Fig. 1 One-dimensional ramp

If we take the gradient of this signal (which, in one dimension, is just the first derivative with respect to t) we get the following:

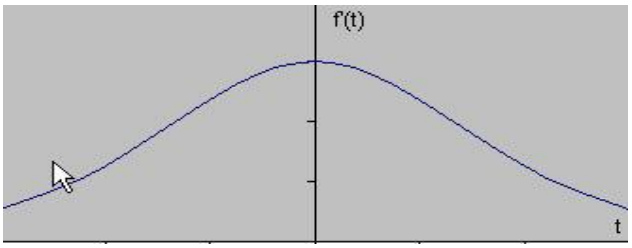

Fig. 2 Gradient of Ramp

Clearly, the derivative shows a maximum located at the center of the edge in the original signal. This method of locating an edge is characteristic of the "gradient filter" family of edge detection filters and includes the Sobel method. A pixel location is declared an edge location if the value of the gradient exceeds some threshold. As mentioned before, edges will have higher pixel intensity values than those surrounding it. So once a threshold is set, you can compare the gradient value to the threshold value and detect an edge whenever the threshold is exceeded. Furthermore, when the first derivative is at a maximum, the second derivative is zero. As a result, another alternative to finding the location of an edge is to locate the zeros in the second derivative [13]. This method is known as the Laplacian and the second derivative of the signal is shown below:

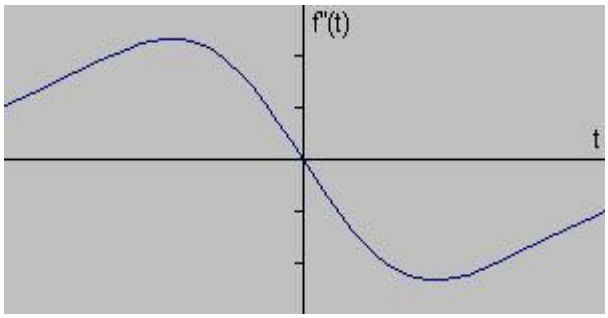

Fig. 3 Second Derivative of Ramp

A. Method of Edge Detection

Sobel operator is the method of edge detection implemented in MATLAB. The operator consists of a pair of 3×3 convolution kernels as shown below. One kernel is simply the other rotated by 90°. The 3x3 convolution kernels detects gradients in the X and Y directions.

$$S_x = \begin{matrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{matrix} \quad S_y = \begin{matrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{matrix} \qquad (1)$$

In image processing, the convolution is a general purpose filter that allows producing a range of effects by specifying a set of convolution kernels. It works by determining new value for a pixel by adding weighted values of all its

neighboring pixels together. The applied weights are determined by a 2D array called convolution kernels. [9]

The two kernels are defined as two matrices in MATLAB. These kernels are designed to respond maximally to edges running vertically and horizontally relative to the pixel grid, one kernel for each of the two perpendicular orientations. The kernels are convolved separately to the input image, to produce separate measurements of the gradient component in each orientation, these two separate measurements are then stored as two matrices named as Gx and Gy. These can then be combined together to find the edge strength of the gradient at each point and the orientation of that gradient. The edge strength is given by:

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

Typically, a faster computation of edge strengths is:

$$|G| = |G_x| + |G_y| \quad (3)$$

The orientations of the edges are computed using the gradient in the X and Y directions, the relation which is used for finding orientations of edges is:

$$\theta = \tan^{-1} {G_y}/{G_x} \quad (4)$$

3. Result and Discussion

Sobel edge detection method is implemented in MATLAB; the algorithm is implemented using the Sobel edge detection method illustrated in coming section of this paper. The implemented algorithm is tested on a number of images, in order to check its functionality and performance. This section of the chapter presents the results from one of the image which was used for edge detection.

Figure 5 shows the image that is used for edge detection, the image is an input to the MATLAB program of edge detection. The input image is carefully selected as it possesses a lot of features with edges; hence the image is a good qualifier for illustrating the results of edge detection from the implemented algorithm. It can be observed in the image that it comprises of patches; some of them are heavy edge loaded areas whereas some have very little edges in them, and the area with water has no edges at all, these are the sort of observations that are likely to be made once this image is processed for edge detection.

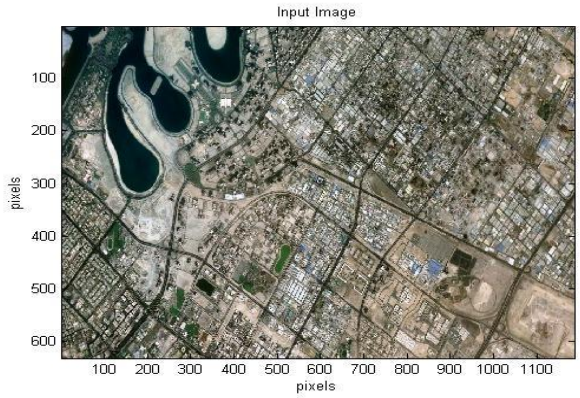

Fig. 4 Input Image used for edge detection

The input image is a colored image and must be converted into a gray-scale format in order to use MATLAB's function for edge detection. The converted gray-scale image is shown in figure 6. The conversion is done using one of MATLAB's functions; which converts RGB images to gray-scale images. The same grayscale image is also again used for edge detection using the sobel algorithm implemented in MATLAB.

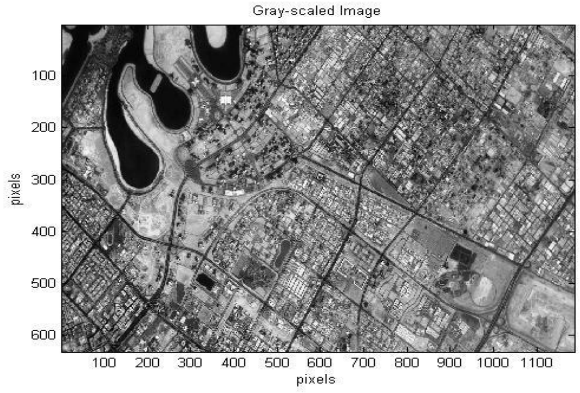

Fig. 5 Gray-scaled converted Image

Figure 7 represents the results of edge detection, when MATLAB's function is used. The MATLAB's function takes the gray-scale or intensity image as input and returns a binary black and white image of the same size. In the image shown in figure 7 and 8, 1's (represented by white) represents where the function finds edges and 0's (represented by black) is where no edge are detected by the function.

MATLAB's function for edge detection is limited to only the functionality of finding edges; is incapable of calculating edge's strengths and their orientations. The function has a sensitivity threshold; the function ignores all edges that are not stronger than the threshold. Note figure 8 has more edges detected than the edges detected in figure 7, it is due to the fact that the sensitivity threshold for the detected edges is higher than that which is for the case of figure 8. The threshold for figure 7 is selected automatically by the MATLAB function and is selected to be 0.1901 while the threshold selected for the case in figure 7 is a value which half of the threshold in figure 7's case.

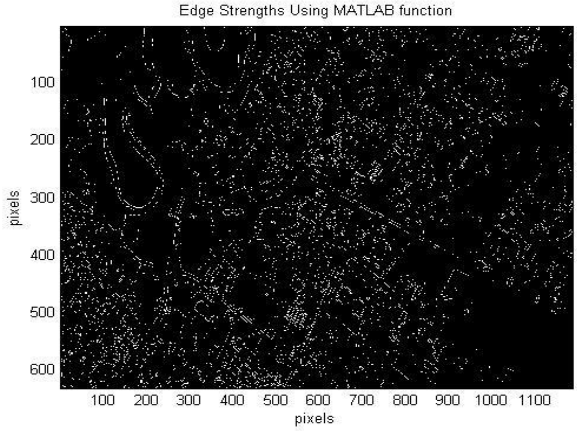

Fig. 6 Edge detection using MATLAB's function

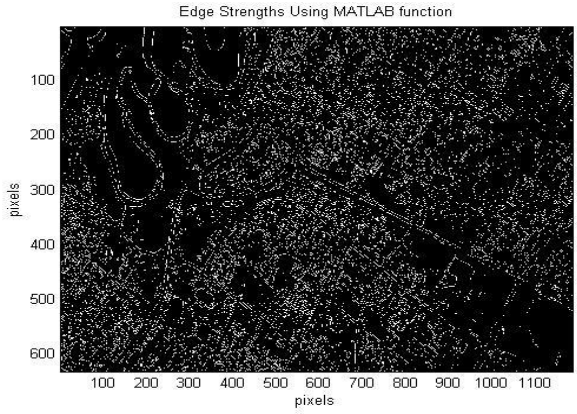

Fig. 7 Decreased threshold value for MATLAB's function

Figure 9 represents the edge strength parameter maps upon performing edge detection algorithm on figure 6. In the figure it can be observed that those areas which exhibit large values of gradients are easily identified and are represented in white, whereas those areas with insignificant signal change activity appear as dark. The figure representing all edge and their strengths information is obvious, on comparison of figure 9 and 8, it can be inferred that this the algorithm built for edge detection gives accurate results, and also gives complete profile of edges information and their strengths.

The edge strengths have values within a finite range; ranging from 0 to maximum strength of an edge in the image. The implemented edge detection program in MATLAB computes edge strengths per pixel of the input image, resulting in matrix containing strengths of the edges. It can also be observed by the intensity changes in figure 9; these intensity changes actually represent strong and weak edge strengths. Some of the pixel's edge strengths are as follows:

145	149	125	110	96	90
208	113	59	33	22	17
252	154	123	36	14	20

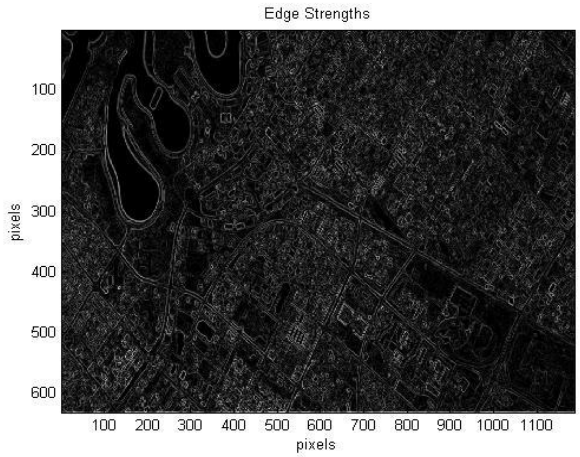

Fig. 8 Edge strengths

Performing edge detection on the input image using the sobel method implemented in MATLAB it is possible to compute edge orientations as well along with their strengths. Figure 10 shows the edge orientations; as the input signal contains a lot of information, as a result strong long orientations are observed in the figure. Randomness can be easily observed in the figure and this randomness can be noticed especially where edges are weak. The orientation parameter maps shown is a mixture of black and white, the black represents horizontal while the white represents vertical orientation parameter maps. The orientations like the strengths also have values within a finite range; i.e. from $-\pi/2$ through 0 to $\pi/2$.

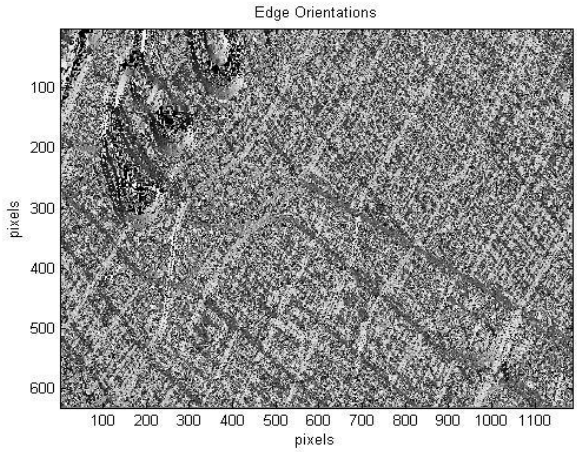

Fig. 9 Edge Orientations

The complementary nature of visual information contained in the images is evident. To illustrate edge detection performed in a better way, the most widely used picture in the world of image processing is selected and corresponding edge detection results are shown in figure 11. Observing the edge strength and orientation parameter maps give a clearer picture of the sobel edge detection algorithm that has been implemented in MATLAN

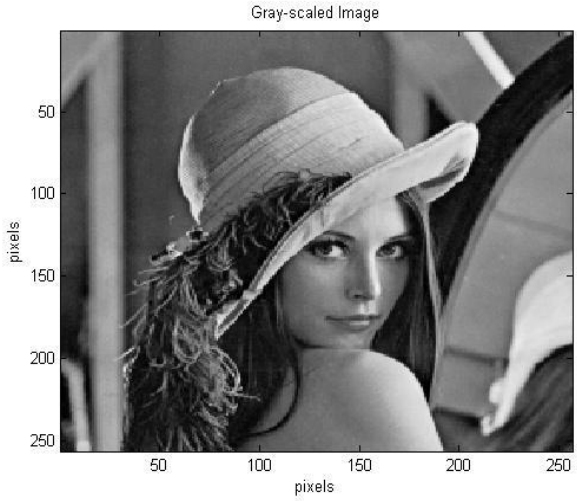

Fig. 10 Edge Orientations

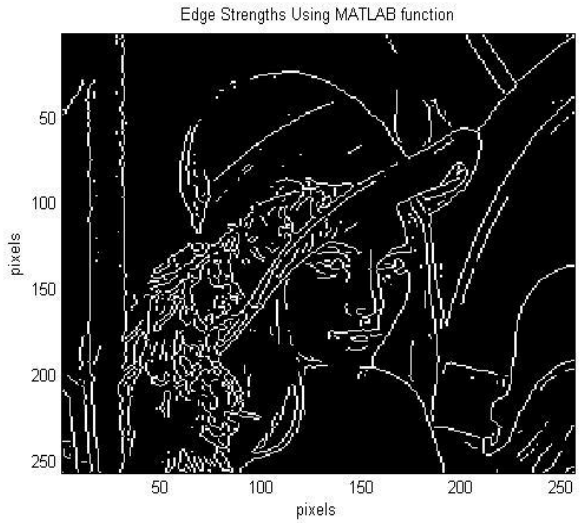

Fig. 11 Edge Orientations

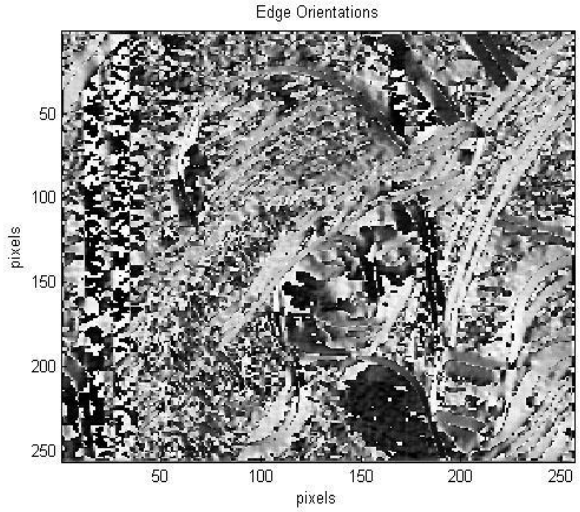

Fig. 12 Images to represent phenomena

4. Conclusion

The edge representation of an image drastically reduces the amount of data to be processed, yet it retains important information about the shapes of objects in the scene. The performance of sobel edge detection method is observed in this paper, and by judging subjectively. An important property of the edge detection method is its ability to extract the accurate edge line with good orientation. Good edge detection is noticed in this chapter, the edge lines are thin and with few speckles.

There is certainly a great deal of diversity in the applications of edge detection, but it is felt that many applications share

a common set of requirements. These requirements yield an abstract requirement. These requirements yield an abstract edge detection problem, the solution of which can be applied in any of the original problem domains. The description of an image gathered on applying edge detection, makes it easy to integrate into a large number of object recognition algorithms used in computer vision and other image processing applications. Edge detection results benefit wide range of applications such as image enhancement, recognition, morphing, restoration, registration, compression, retrieval, watermarking. The future work will be how to use this edge detection for wireless sensor node position optimization in the experimental field for the detection of different type of phenomena.

References

[1] Ian F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci. “A survey on sensor networks”. IEEE Communications Magazine, pages 102.114, 2002

[2] Chris Townsend and Steven Arms, "Wireless Sensor Networks: Principles and Applications," in Sensor Technology Handbook, Jon S. Wilson, Ed.: Elsevier, 2005, ch. 22, pp. 439-450.

[3] Research tutorial from Wireless Sensor network lab at Stanford University, http://wsnl.stanford.edu/tutorial.html (accessed on 25/07/2010)

[4] Ziou, D. and Tabbone, S., “Edge detection techniques-an overview”. Int. J. Pattern Recognit. Image Anal. v8 i4. 537-559.

[5] M. Juneja, P. S. Sandhu, “Performance Evaluation of Edge Detection Techniques for Images in spatial Domain”. International Journal of Computer Theory and Technology, Vol. 1, No. 5, December 2009, 1793-8201

[6] K. R. Rao, J. J. Hwang, "Techniques and Standards for image, Video, and Audio Coding", Prentice Hall PTR, 1996.

[7] Y. Yu, C. Chang, ―”A new edge detection approach based on image context analysis”, Elsevier Journal of Image and Vision Computing 24 (2006) 1090–1102

[8] R. C. Gonzalez, R. E. Woods, “Digital Image Processing”, Third Edition, 2008 Pearson Prentice Hall. ISBN: 0-13-505267-X

[9] Stephen E. Reichenbach, “Two-Dimensional Cubic Convolution”. IEEE TRANSACTIONS ON IMAGE PROCESSING, VOL. 12, NO. 8, AUGUST 2003

[10] J.A. Kong, S. H. Yueh, H. H. Lim, R.T. Shin, and J. J. Van Zyl. “Classification of earth terrain using polarimetric synthetic aperture radar images,” in Progress in Electromagnetic Research, Vol. 111, “Polarimetric Remote Sensing”, New York: Elsevier, 1990

[11] R. C. Gonzalez and R. E. Woods. “Digital Image Processing”. 2nd ed. Prentice Hall, 2002.

[12] V. Torre and T. A. Poggio. “On edge detection”. IEEE Trans. Pattern Anal. Machine Intell., vol. PAMI-8, no. 2, pp. 187-163, Mar. 1986.