Texture Segmentation by Local Fractal Dimension as Applied to Oceanic Search and Rescue

Andrew Ringler aringl@cs.mcgill.ca

April 30, 2003

Abstract

An algorithm for Texture Segmentation is presented for segmenting an image into regions of similarity based on computation of the Fractal Dimension - Fractal Dimension Segmentation (FDS)[9]. I utilize two methods of calculating the fractal dimension of binary and grayscale images: the box counting and 2D variation dimension respectively. Texture segmentation algorithms are developed and detailed based on the two mentioned dimension measures. Efficient algorithms are provided that allow for linear time computation of fractal dimension segmentation. I explore an image domain where fractal dimension is thought to be a meaningfull classifier: oceanic search and rescue.

1 Introduction

An algorithm for Texture Segmentation is presented for segmenting an image into regions of similarity based on computation of the Fractal Dimension - Fractal Dimension Segmentation (FDS). I utilize two methods of calculating the fractal dimension of binary and grayscale images: the box counting and 2D variation methods respectively. I extend both of these algorithms for use in texture segmentation. I then explore a very constrained image domain: oceanic search and rescue.

Section 2 introduces the concepts of image segmentation. Section 3 reviews set dimensions, and important definitions. Section 3.1-3.2 presents fractals and fractal dimensions. Section 4 details the box counting method for images. Section 5 details the 2D variation method for images. Section 6 presents the fractal dimension segmentation algorithm. Section 7 provides theoretical and intuitive justification of fractal dimension segmentation. Section 8 details the image domain: oceanic search and rescue. Section 9 provides results for both dimension measures. Section 10 provides a conclusion. Lastly Appendix A and B detail linear time algorithms for fractal dimension segmentation.

2 Image Segmentation

We define image segmentation as the process of translating an image into a description of the “distinct” regions contained within that image. One way to segment an image into similarity regions is through texture measurements[6]. Textures often correspond to human’s perceptual notions of region similarity[8].

We define a surface texture by the regular repitition of an element or pattern, called surface texel on a surface[6]. Given such a loose definition is is difficult to distinguish mathematically between textures that are perceptualy different.

One proposed method for texture segmentation is to use the fractal dimension of the regions. This has some validity in certain constrained image domains. The necessary background will be introduced first.

3 Dimension

We can intuitively say that the dimension of a point, a line, or the interior of a square have dimensions zero, one and two respectively. To assign a dimension to an arbitrary set is not as trivial. Any dimension measure on an arbitrary set $X \subset \Re^n$ should satisfy the following properties, (taken verbatim from [1]):

1. For the singleton set $\{p\}$, $dim(\{p\}) = 0$, for the unit interval I^1, $dim(I^1) = 1$, and in general, for the m-dimensional hypercube I^m, $dim(I^m) = m$.
2. *(Monotonicity) If* $X \subset Y$*,*
$$dim(X) \leq dim(Y)$$
3. *(*Countable stability) If $\{X_j\}$ is a sequence of closed subsets of $\Re^n$, then
$$dim(\bigcup_{j=1}^{\infty}) = \sup_{j \geq 1} dim(X_j)$$
4. *(Invariance) For an arbitrary map* $\psi belonging to some subfamily of the set of homeomorphisms of \Re^n$ *to* $\Re^n$
$$dim(\psi(X)) = dim(X)$$

Ursohn constructed a dimension satisfying the above properties know as the topological dimension (dim_T) and which takes on integer values. Haufsdorff noticed that non-integer dimensions also made sense. He defined the Haufsdorff dimension (dim_H) which satisfies the above properties and can for certain sets take on real values. We have the inequality

$$dim_T(X) \leq dim_H(X)$$

Both $dim_T(X)$ and $dim_H(X)$ agree with our intuitive notion of dimension, having dimensions zero, one and two for the point, line and square interior respectively.

3.1 Fractals

Mandelbrot recognizing the inequality between the topological and Haufsdorff dimensions stated that a set X in $\Re^n$ is "fractal" if

$$dim_T(X) < dim_H(X)$$

Or in other words if the topological dimension and the Haufsdorff dimension differ than X is a fractal. We can now easily say now define the fractal degree δ of a set X

$$\delta(X) = dim_H(X) - dim_T(X)$$

Computation of the fractal degree directly from the above formula requires the Hausfdorff and the Topological dimension. In general it is not easy to compute the Hausfdorff dimension for an arbitrary set in $\Re^n$. Mandelbrot introduced another real-valued dimension, the box counting dimension, which is very easily computable for sets in $\Re^n$.

3.2 Box Counting Dimension

Let "ϵ-ball" be a closed ball in $\Re^n$ of radius ϵ. Let X be a bounded set in $\Re^n$. Let $N(\epsilon)$ be the minimum number of ϵ-balls needed to cover X. Then

$$dim_B = \lim_{\epsilon \to 0_+} \frac{\log(N(\epsilon))}{\log(1/\epsilon)} \tag{1}$$

Where dim_B is the box counting dimension satisfying the inequality

$$dim_T(X) \leq dim_H(X) \leq dim_B(X)$$

4 Discrete Box Counting

Now we consider the discrete case for solving 1. In order to approximate the limit as the box size is decreased to zero we can calculate $N(\epsilon)$ at many discrete box sizes ϵ. We then plot $\log(1/\epsilon)$ on the x-axis vs $\log(N(\epsilon)$ on the y-axis and fit a line to the points (in the linear least squares sense). Now note that the slope of the line corresponds to $\frac{\log(N(\epsilon))}{\log(1/\epsilon)}$. Since this line intersects the Y axis the slope will approximate $\lim_{\epsilon \to 0}$.

4.1 Discrete Box Counting Procedure for Images

The discrete box counting dimension can easily be applied to binary images[2].

Divide the image into a grid of cells of size $r \cdot r$. Denote $N(r)$ as the number of cells containing at least a part of the structure (an **on** pixel). Then by the power law we have the number of cells needed to cover the structure at scale r as

$$N(r) = const \cdot r^{-D_B}$$

where D_B is the box counting dimension. The total area A covered by the cells of size r is

$$A(r) = N \cdot r^2 = const \cdot r^{-D_B} \cdot r^2 = const \cdot r^{2-D_B}$$

We can now plot $\log(r)$ vs. $\log(N \cdot r^2)$ as mentioned in discrete box counting for different scales r. Note that we are taking the dimension of a binary flat surface. A line on this surface will have dimension 1 and the interior of a square dimension 2. Thus an arbitrary binary image will have a box counting dimension between 1 and 2.

5 2D Variation Method

We can compute the fractal dimension of a given grayscale image by extending the 2D variation method[12] of binary images. I will describe the grayscale method[2] only.

For each pixel p in the image denote a box of size r around p. Now find the maximum and minimum pixel values in this box. Record these values as the values min and max of p. (Note that there is a border of size $r/2$ at the perimeter of the image for which the box is undefined, we do not consider these pixels). Now, since the intensity value of a pixel is related to the height of the photographed object we can meaningfully define volume for cells. Define difference volume of p as $r \cdot r \cdot (max - min)$. Denote $V(r)$ as the sum of all p's difference volumes for scale r. We now have volume instead of area but similarly to the box counting method we find that

$$V(r) = const \cdot r^s$$

And

$$D_{2D} = 3 - s/2$$

Where D_{2D} is the 2D variation dimension. Note that now we are measuring the dimension of a 3-Dimensional surface, since the intensity value of a given pixel corresponds to the slope of the surface at that pixel and loosely to the height of the object. Now a completely flat surface will have dimension 2 and a three dimensional one will have dimension 3. Thus 2D variation dimension will output values between 2 and 3.

6 Fractal Dimension Segmentation Algorithm

Now we can apply the fractal dimension algorithm over a small section of data $w \cdot w$ (instead of the entire image). We will call this a window of size w (w should be much smaller than the image $width$ and $height$). Now constrain the window size to an odd value so that there is a well defined pixel in the center of the window. Call this pixel the center pixel.

Now for a given pixel p of the original image let the local fractal dimension (LFD)of p be defined as the fractal dimension of the window containing p as the center pixel (the fractal dimension in the neighborhood of p).

Compute the LFD for each well defined center pixel. Pixels on the edge of the image cannot have a full window defined around them so we will ignore. We call the set of local fractal dimensions computed for each center pixel the dimension profile[9].

We can now view the dimension profile as an image by mapping the profile to a valid set of pixel values (often called histogram equalization). I.E. if we want 8-bit grayscale images as output, and we are using 2D variation method for computing our local fractal dimension, then our profile will be values in the range 2-3 which will map to the range 0-255.

7 Theoretical Motivation

I will first give a theoretical bases for fractal dimension segmentation, and will then detail some of the assumptions I had to make. Let S^* be the set of all posible segmentations of an image and s_i a given segmentation. Now a given segmentation s_i is segmented into $|s_i|$ number of regions r_k. We define the fractal dimension of a given region r_k as the fractal dimension of an image identical to r_k (note we are not talking about local fractal dimension here).

Now we can take the local fractal dimension of each pixel in r_k. Use the same window for all pixels defined by the bounds of the region r_k. Now define the error of the region $\epsilon(r_k)$ as the sum of squared differences of the LFD profile of the region r_k. We are measuring how homogenous the region is. Now define the error of the segmentation $\epsilon(s_i)$ as the sum of errors of all regions contained in s_i.

Now take the optimal segmentation as the segmentation with the minimum error. Now we have the segmentation with the most homogenous regions, this is precicely what we want. Now for each region take his fractal dimension and take this as the new dimension profile. We now have completely homogoneous regions with meaningful relative intensities between them.

The local fractal dimension measure in this paper (in some sense) skips to the very last step mentioned but is meant to be an approximation to the entire method above.

7.1 Assumptions

We make the following assumptions in our problem formulation

1. Texture is a meaningfull way to segment images into similarity regions. This is assumption is beyond the scope of this paper and is discussed in [8]

2. The fractal dimension of a texture region is different from the fractal dimension of all his neighboring texture regions. This assumption allows us to differentiate texture regions by fractal dimension. I deal with this by limiting my images to a domain where I feel this assumption is valid. This is discussed in the next section.

3. Every subset of a given region has the same fractal dimension as the entire region. In other words the region is homogenous. If this were not the case then **local fractal dimension** would not be as meaningful. This assumptions did not turn out to be as important as suspected.

8 Image Domain

The image domain is in fact the personal motivation for the presented algorithms. The U.S. Coast Guard performs many search and rescue operations every year. These operations involve sending out a combination of ships planes and helicopters to search for and rescue a lost ship or person. The vast majority of the search effort is performed by low flying planes equiped with belly-mounted cameras. A human must continually monitor an in-plane display: searching for the rescue target. The combination of long flights, tired rescuers, tired eyes, and the wide viewing area of the camera display can result in missed detections. Computer assisted detection could improve detection results.

Now assuming digital access to the in-plane display I propose to segment the image into two detectable classes: water and non-water. I believe water and non-water should have different fractal dimensions. Ocean waves are self-similar in structure. We can intuitively say that they are self-similar at differing scales. Take an overhead image of many large waves. Zoom into this image until a single large wave fills the entire screen. Notice that the new image (of the large wave) contains many smaller waves. Now zoom in on one of these smaller waves. Note that the new images yet again contains wave like structures.

We can also see that an image of waves should be somewhat invariant in fractal dimension as we move about spacially since similar forces of nature are acting on the waves all at points in the photograph. This satisfies our requirement for homogenous regions.

Now introduce an object into the water. The object (especially his edges) should have a different fractal dimension than the sourounding waves. Our algorithm should be able to distinguish between water and non-water.

9 Results

Both the box counting and 2d variation method yielded useful and interesting results. No post processing was performed as this would have distracted from the researched algorithm. Note that simple intensity thresholding could visually segment the output images into our two classes.

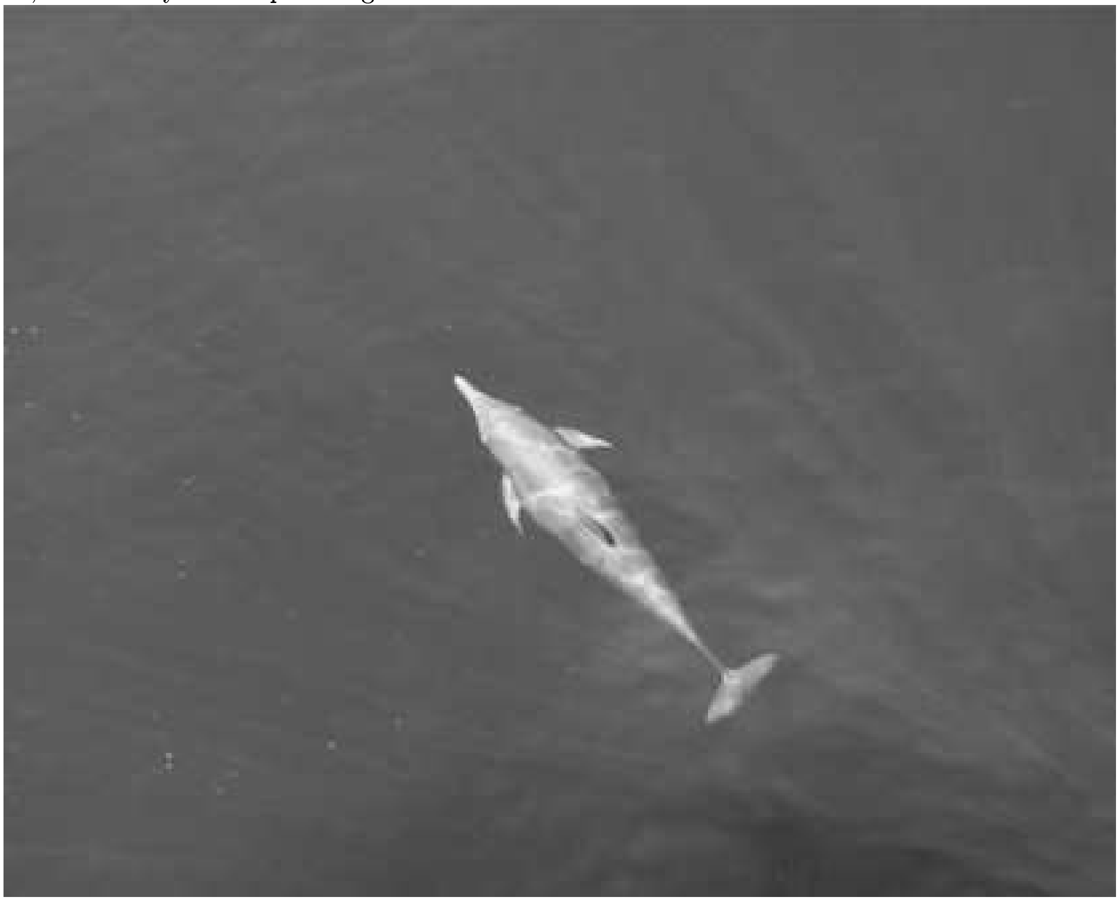
9.1 Box Counting

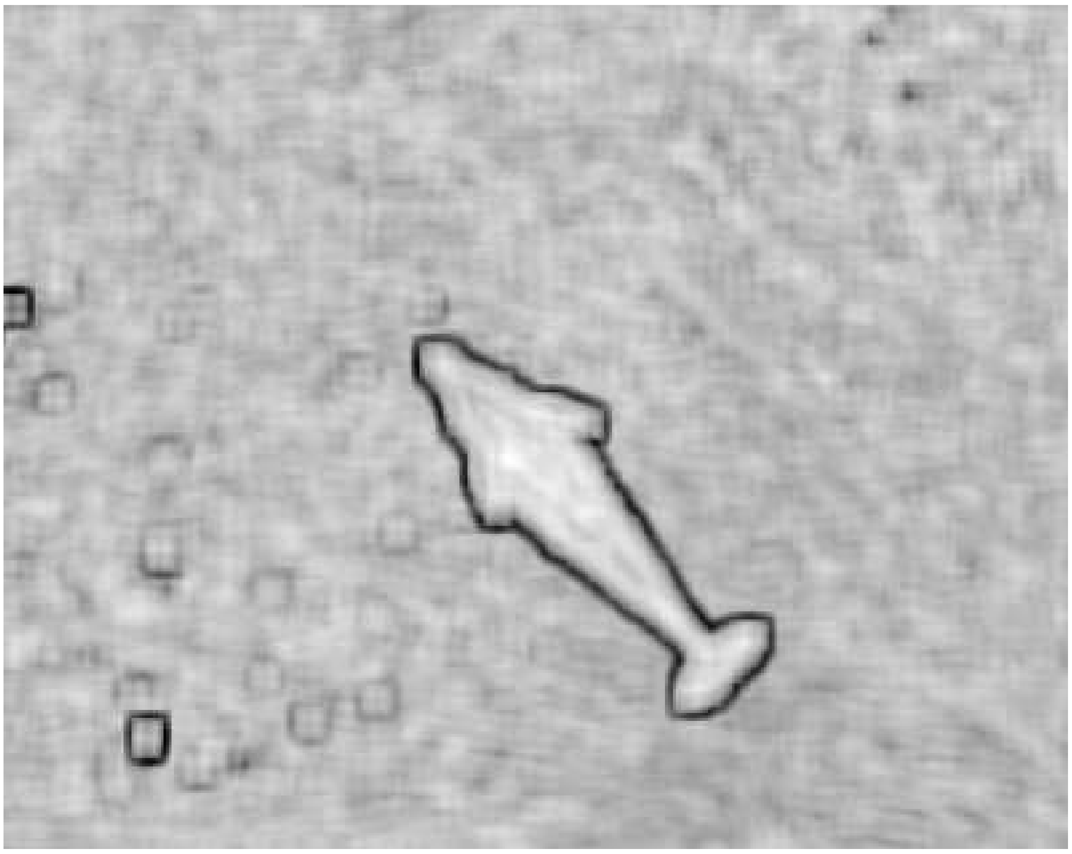
As the box counting method is the simplest of the methods this was implemented first. Grayscale images needed to be quantized first, since the box counting method takes binary images as input. Quantization was done by assigning the darkest pixels to the value 0 and the lightest pixels a value 1. The first image tested was an image of four Coast Guard swimmers, shown below, followed by the output image.

The output image is shown above. We can see in the output that the algorithm has segmented the image into two regions. We have a very dark region showing the outline of the swimmers. The rest of the image is almost solid grey. Note that the swimmers are highly two dimensional compared with the 3D ocean waves and so the former are much darker.

10 2D Variation Method

I was very impressed with the results from the 2D Variation Segmentation algorithm. The analysis of a single image will be instructive. This is an overhead photograph of a dolphin swimming in the ocean, shown below, followed by the output image.





The output yields a dark band of black, highlighting the outline of the dolphin. Now the most saturated intensity (white) corresponds to the largest local dimension while the least saturated (black) corresponds to the smallest local dimension. Since we are using histogram equalization it is not necessary that white correspond to dimension 3 and black to 2, but black will always correspond to a dimension less than or equal to white.

We notice that the curved back of the dolphin is white as well the waves. These are very close to 3-Dimensional. Where edge of the dolphin meets the ocean his outline is traced. This line is close to being 2-Dimensional and is represented in our output by a black region. In testing, the box counting method did not have enough accuracy to dilimenate the edge of the dolphin but instead blured the entire dolphin.

11 Conclusion

Initial results in search and rescue type imagery were promising. More research could be done in this area. There are a few open questions after performing this research. Firstly, why did the box counting and 2D variation methods seem to differ qualitatively? More testing could certainly be done using the software developed for this research. Extensive image testing might yield interesting discoveries. There is certainly further research that can be done in this field.

The use of color images can be explored. The use of fractal dimension segmentation coupled with other texture and segmentation measures could prove useful. The use of this method on an extremely constrained set of data images (i.e constant camera type).

This paper presented a fractal dimension segmentation methodology and applied it to two fractal dimension measures. Efficient implementations would almost certainly allow for real-time applications. These algorithms might prove to have some use in the interested domain: oceanic search and rescue.

References

[1] Yamaguchi, Masaya, 1925, Mathematics of Fractals/Masaya Yamaguti, Masayashi Hata, Jun Kigami; translated by Kiki Hudson. p. cm. - (translations of Mathematical Monographs v. 167).

[2] Kraft, Roland, Estimating the Fractal Dimension from Digitized Images. Roland Kraft, Josef Kauer, Munich University, Statistics and Data Processing Institute

[3] B.B. Mandelbrot, Fractal Geometry of Nature, Freeman Press, San Fransisco, 1982.

[4] B.B. Chaudhuri and Nirufam Sarkar, Texture Segmentation Using Fractal Dimension.

[5] An Efficient Approach to Compute Fractal Dimension in Texture Image, B.B. Chaudhuri and N. Sarkar. Electronics and Communications Sciences Unit. Indian Statistical Institute. 203 B.T. Road Calcutta, INDIA.

[6] Wilson, Roland, 1949 - Image Segmentation and Uncertainty. (Electronics Electrical Engineering Research Studies). Pattern Recognition Image Processing Series; 9).

[7] Trucco, Emanuele, Introductionary Techniques for 3-D Computer Vision, Emanuele Trucco and Allessadro Verri.

[8] J.Garding, Direct Estimation of Shape from Texture, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. PAMI-15, no. 11, pp. 1202-1207 1993.

[9] Fractal Dimension Segmentation of Synthetic Aperture Radar Images, J.M. Blackledge and E. Fowler. Cranfield Institute of Technology, U.K.

[10] Machine Vision for Detection of the Rescue Target in the Marine Casualty.

[11] Alligood, Kathleen T. Chaos - An Introduction to Dynamical Systems/ Kathleen Alligood, Tim Sauer, James A. Yorke. 1997 Springer-Verlog New York, Inc.

[12] Dubuc B., Quiniuo J.F., Roques Carmes C., Tricot C., Zucker S.W. (1989). Evaluating the Fractal Dimension of Profiles. Phys. Rev. 39: 1500-1512.

Appendix A: Efficient Box Counting Fractal Segmentation

A naive implementation of Box Counting involves counting of the boxes for each scale. Assume that an image's dimensions are a power of 2. The number of scales is k. Then if an image has n pixels we have an $O(nk)$ running time. Now applying this to segmentation we need to compute a fractal dimension for each center pixel. Thats about n pixels times w^2k computations per pixel (where w is the window size). Giving us a $O(nw^2k^2)$ algorithm. We approaches $O(n^2)$ for certain image sizes.

We will precompute new images for each scale. For the two pixel scale we will scan the original image and for each two pixel block we will generate a new image with an *on* pixel if the block contains an on, and an off if not. Then we will generate the 4-pixel scale from the 2-pixel scale image and for each scale generate an image. This takes $n + n/4 + n/16 + .. =$. Now for each local fractal dimension we will use the scale matrix for each scale and we use the previous iteration.

If we have the number of boxes from the previous iteration then the number of boxes in the current iteration is the previous count minus the previous' first column plus the current's last column (or rows if we are on the last pixel in a row). Or to compute the box counting segmentation we need $O(nk) + O(nw) = O(nk + nw) = O(n(k + w))$.

Appendix B : Efficient 2D Variation Fractal Segmentation

Now an efficient implementation of 2D variation method is almost identical to that of efficient box counting. We do not have to precompute scales this time however. Now when we want to calculate the max and min of a center pixel we can take the previous iteration's center min/max. We define the center min/max of an iteration as the min/max excluding the first column. Thus we record two sets of min/max, one for the current iteration and one for the next iteration.

In all other details the procedure remains the same as efficient box counting. Again we have the same running time as the box counting fractal segmentation.