# AN IMPROVED CONTENT-ADAPTIVE MESH-GENERATION METHOD FOR IMAGE REPRESENTATION

*Michael D. Adams*

Dept. of Elec. and Comp. Eng., University of Victoria, Victoria, BC, V8W 3P6, Canada

## ABSTRACT

A new content-adaptive mesh-generation method for image representation, based on the greedy point-removal scheme of Demaret and Iske, is proposed. The proposed method is shown to be capable of producing meshes of higher quality than those generated by the scheme of Demaret and Iske, while requiring substantially less computation and memory. Furthermore, with the proposed method, one can easily tradeoff between mesh quality and computational/memory complexity as needed. Since a mesh-generation scheme proposed by Yang et al. is a key component of the proposed method, some factors affecting the performance of this scheme are also explored.

***Index Terms*—** Image representations, mesh generation, triangle meshes, greedy point removal, error diffusion.

## 1. INTRODUCTION

In recent years, there has been a growing interest in image representations that are based on content-adaptive (i.e., nonuniform) sampling. In particular, triangle meshes (especially those based on Delaunay triangulations) have received considerable attention as a means for representing images [1, 2]. Such representations are useful in a number of applications, including image coding. Since images are usually uniformly sampled on a (truncated) lattice, if we wish to employ a mesh-based image representation in a particular application, a means is needed for selecting a good subset of the original sample points from which to generate a (nonuniformly-sampled) mesh approximation. For this reason, mesh generation methods are of great interest. Two highly effective mesh-generation methods that have been proposed to date are the **greedy point-removal (GPR)** scheme of Demaret and Iske [2] and **error-diffusion (ED)** scheme of Yang et al. [1]. In this paper, we propose a new mesh-generation method that is based on the GPR scheme, but exploits ideas from the ED approach in order to greatly reduce computational/memory complexity at essentially no cost to mesh quality.

The remainder of this paper is structured as follows. First, Section 2 introduces triangle meshes as a tool for image representation. Then, we turn our attention to mesh generation schemes. Section 3 describes the ED method and investigates factors affecting its performance. Next, Section 4 introduces the GPR method and its shortcomings. Our proposed mesh-generation method is then introduced in Section 5. In Section 6, some experimental results produced by our proposed method are presented and its performance is shown to be very good relative to other methods. Finally, we close in Section 7 with a summary of our work.

## 2. MESH-BASED IMAGE REPRESENTATION

Consider an image $f$ defined on the domain $I = [[0,W) \times [0,H)] \cap \mathbb{Z}^2$, where $\mathbb{Z}$ denotes the set of integers (i.e., $f$ is an image uniformly sampled on a truncated rectangular lattice of width $W$ and height $H$). To construct an approximation of the image $f$ using a triangle mesh, we proceed as follows: 1) Select a subset $S$ of the sample points $I$. 2) Construct a Delaunay triangulation of $S$. 3) For each face in the triangulation with vertices $(x_0, y_0)$, $(x_1, y_1)$, and $(x_2, y_2)$ (which correspond to sample points), and their respective sample values $z_0$, $z_1$, and $z_2$, form the unique planar interpolant through $(x_0, y_0, z_0)$, $(x_1, y_1, z_1)$, and $(x_2, y_2, z_2)$. 4) Combine all of the interpolants from step 3 to form an approximation of the image $f$ over its entire domain $I$. In some applications, such as image coding, the uniqueness of the triangulation can be quite important, as this eliminates the need for any information other than $S$ in order to determine the topology of the triangulation. For this reason, we ensure a unique triangulation by employing the preferred-directions scheme of Dyken and Floater [3]. The challenging part of the above mesh-generation process is selecting the set $S$ such that it has a certain prescribed size and also minimizes the squared error of the mesh approximation. It is this part of the mesh-generation problem that we address in this paper.

Before proceeding, a brief comment is in order concerning the image data employed in our work. Although several test images were used, the two for which we later present results, namely `lena` and `peppers`, are grayscale versions of the 512×512 Lena and Peppers images from the well-known USC Image Database.

## 3. ERROR-DIFFUSION (ED) METHOD

Before presenting our proposed method, we first introduce two other mesh-generation schemes related to our method. The first of these other schemes is the **error-diffusion (ED)** method of Yang et al. [1]. Given an image $f$ (defined on $I$) and a desired number $N$ of sample points, the ED method uses Floyd-Steinberg error diffusion [4] to generate a set $S$ of $N$ sample points, distributed such that the local density of sample points at each point $(x, y) \in I$ is proportional to the largest magnitude second-order directional derivative of $f$ at $(x, y)$. In more detail, the ED method consists of the following steps: 1) From $f$, compute the sample-point density function $d$ defined on $I$ given by

$$d(x,y) = \tilde{d}(x,y)/\tilde{d}_{\mathsf{max}},$$

where $\tilde{d}_{\mathsf{max}} = \max_{(x,y)\in I} \tilde{d}(x,y)$, and $\tilde{d}(x,y)$ is the maximum magnitude second-order directional derivative of $f$ at $(x,y)$. 2) Set the threshold $\tau$ to use for Floyd-Steinberg error diffusion to be $\tau_0 = \frac{1}{2N} \sum_{(x,y)\in I} d(x,y)$. 3) Convert the function $d$ (which can be viewed as a continuous-tone image) to a binary image $b$ (defined on $I$) using Floyd-Steinberg error diffusion, as described in [1], with threshold $\tau$ and a serpentine scan order. 4) Set $S'$ to the set of all

This work was supported by the Natural Sciences and Engineering Research Council of Canada.

points $(x, y)$ for which $b(x, y) \neq 0$. Set $N'$ to the size of $S'$. 5) If $N'$ is close enough to $N$, set $S$ to $S'$ and stop; otherwise, adjust $\tau$ appropriately and go to step 3. Note that, in step 1, $\tilde{d}$ is calculated as

$$\tilde{d}(x, y) = \max\{|\alpha(x, y) + \beta(x, y)|, |\alpha(x, y) - \beta(x, y)|\},$$

$$\text{where } \alpha(x, y) = \tfrac{1}{2}[\tfrac{\partial^2}{\partial x^2} f(x, y) + \tfrac{\partial^2}{\partial y^2} f(x, y)],$$

$$\beta(x, y) = \sqrt{\tfrac{1}{4}[\tfrac{\partial^2}{\partial x^2} f(x, y) - \tfrac{\partial^2}{\partial y^2} f(x, y)]^2 + [\tfrac{\partial^2}{\partial x \partial y} f(x, y)]^2},$$

and the one-dimensional convolution masks of $\left[\, \frac{1}{2} \;\; 0 \; -\frac{1}{2} \,\right]$ and $\left[\, 1 \; -2 \;\; 1 \,\right]$ are used as discrete-time approximations for the first- and second-order partial derivative operators, respectively.

Since the ED method must perform convolutions to compute partial derivatives, this raises the question of how to handle filtering at the image boundaries. In [1], Yang et al. did not clearly specify how image boundaries should be treated. Also, the authors did not quantify the benefits (or lack thereof) of including a smoothing (i.e., lowpass-filtering) operator in the convolution kernels used for derivative computation. In what follows, we examine the preceding issues more carefully. In particular, we consider three popular signal extension strategies for handling image boundaries: 1) zero extension (i.e., zero padding), 2) constant extension (i.e., repeating the first and last samples), and 3) symmetric extension (i.e., mirroring about the first and last samples). Also, we consider two smoothing strategies: 1) no smoothing is employed, and 2) smoothing is performed using a Gaussian lowpass filter with a standard deviation of one.

For several images and sampling densities, we generated a mesh using the ED method in conjunction with each combination of signal extension method and smoothing strategy, and measured the resulting mesh approximation error in terms of peak-signal-to-noise ratio (PSNR). A representative subset of the results (for the lena image) is given in Table 1. From these results, it is clear that, for the cases of both smoothing and no smoothing, zero extension performs best. A more careful analysis shows that the better performance of zero extension is due to the fact that it typically yields much larger magnitude second-order derivatives at the image boundaries, resulting in significantly more points being placed on the image-domain boundary. Constant and symmetric extension perform relatively poorly due to their inability to place a sufficient number of points on the image-domain boundary. This behavior is illustrated in Fig. 1 for the lena image. Figs. 1(a) and (b) show, respectively, the points selected when zero and symmetric extension (with smoothing) are employed. From these figures, we can see that, in the case of zero extension, many more points are chosen along the image boundary as compared to symmetric extension. The results in Table 1 also show that, for each extension method, the use of smoothing leads to better results than no smoothing. As it turns out, not using smoothing causes points to be somewhat more randomly distributed due to phantom large-magnitude derivatives caused by noise. This behavior can be seen by comparing Figs. 1(a) and (c), which show results obtained using zero extension with and without smoothing, respectively. Clearly, the points in Fig. 1(a) (which were obtained with smoothing) better reflect the underlying structure of the image (especially fine detail) than those in Fig. 1(c) (which were obtained without smoothing). Since zero extension with smoothing was found to perform best, this combination is always used for the ED method in the remainder of this paper.

## 4. GREEDY POINT-REMOVAL (GPR) METHOD

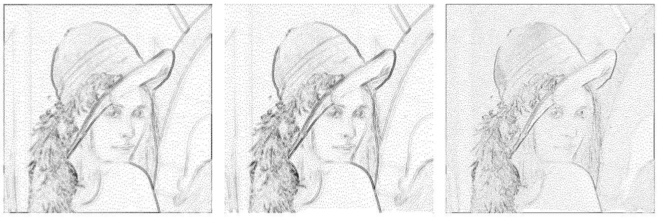
The second mesh-generation method of interest herein is the **greedy point-removal (GPR)** scheme of Demaret and Iske [2]. In short, this method first constructs a Delaunay triangulation of all of the sample points of the image, and then repeatedly removes the point that yields the smallest increase in the squared error of the mesh approximation. More specifically, for a given image (of width $W$ and height $H$) and a desired number $N$ of sample points, the method consists of the following steps: 1) Insert all $WH$ of the sample points in the triangulation. 2) For each point $p$ remaining in the triangulation, compute the increase in the squared error of the mesh approximation that is incurred if $p$ is removed. 3) Remove the point that results in the smallest error increase as determined by the calculations in step 2. 4) If the number of points remaining in the triangulation is greater than $N$, go to step 2. Incidentally, since the deletion of a vertex from a Delaunay triangulation is guaranteed only to affect the faces incident on the vertex to be deleted, step 2 can be performed quite efficiently in practice. That is, in each iteration (with the exception of the first), step 2 only needs to recompute the error increase for a very small number of points (i.e., the immediate neighbours of the point deleted in the previous iteration).

Although the GPR method has been shown to yield excellent quality meshes, it has one major weakness, namely its very high computational and memory costs. Since the GPR method starts with a triangulation containing all of the sample points of the image (i.e., $WH$ points), the mesh size at the beginning of the algorithm can be extremely large. With today's digital cameras, a value for $WH$ on the order of $10^7$ is not unreasonable. The large mesh size leads to an algorithm that can require very high computation times and very large amounts of memory.

Lastly, we note that, due to the greedy nature of the GPR method, it is extremely unlikely to yield a globally optimal solution. This suboptimality is a direct consequence of the short-sightedness of the greedy strategy. That is, when a point is removed, the algorithm fails to consider how this point's removal affects the evolution of the algorithm in *all subsequent* iterations. In short, trying to minimize the increase in error in the current iteration may cause the error-increment values of later iterations to become much larger.

**Table 1**. Comparison of several variants of the ED method for the lena image

| Samp. Density (%) | PSNR (dB) | | | | | |
|---|---|---|---|---|---|---|
| | No Smoothing | | | Smoothing | | |
| | Zero Ext. | Const. Ext. | Sym. Ext. | Zero Ext. | Const. Ext. | Sym. Ext. |
| 1 | 20.12 | 19.63 | 19.84 | **22.24** | 21.36 | 21.20 |
| 2 | 23.71 | 23.21 | 23.06 | **26.32** | 25.81 | 25.90 |
| 4 | 27.85 | 26.92 | 27.04 | **29.43** | 28.96 | 28.93 |
| 8 | 31.62 | 30.61 | 30.63 | **32.35** | 31.62 | 31.66 |



(a) (b) (c)

**Fig. 1**. Selected sample points obtained for the lena image at a sampling density of 8% using (a) zero extension with smoothing, (b) symmetric extension with smoothing, and (c) zero extension without smoothing.

In light of the above suboptimality, it would seem plausible that solutions of quality comparable to (or better than) those obtained with the GPR method could be achieved without the need to consider all $WH$ sample points of the original image. This hypothesis motivated us to propose a modified version of the GPR method, which we present next.

## 5. PROPOSED METHOD: GPR FROM SUBSET (GPRFS)

To improve upon the shortcomings of the GPR scheme, we propose a slightly modified version of this scheme, which we henceforth refer to as **GPR from subset (GPRFS)**. Given an image (of width $W$ and height $H$) and a desired number $N$ of sample points for the mesh, our GPRFS method is identical to the GPR scheme, except for step 1 (from the GPR scheme), which is amended to read: "1) Insert a subset $S_0$ of the sample points of size $N_0$ into the triangulation, where $N_0 \in [N, WH]$." In other words, our GPRFS method inserts only a subset of the sample points, instead of all of them. Clearly, by choosing $N_0$ to be much less than $WH$, we can achieve lower computational and memory complexities than the GPR scheme. Of course, for our proposed approach to be useful, we need an effective means for choosing $S_0$. Although many schemes could be formulated for selecting $S_0$, we consider only two herein, with each leading to a different variant of the GPRFS method. The first scheme, which is the one whose use is advocated by this paper, employs the ED method (of Section 3) to select $S_0$, and yields the GPRFS variant known as **GPRFS-ED**. The second scheme, which is only used later for benchmarking purposes, simply chooses $S_0$ randomly, and yields the GPRFS variant known as **GPRFS-Random**.

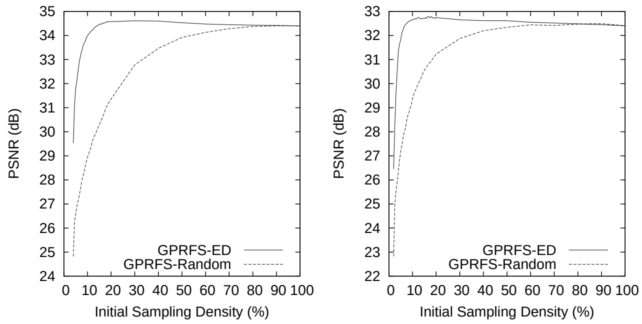
Note that the proposed GPRFS-ED method includes the GPR and ED methods as special cases. That is, if $N_0 = N$, the ED method is obtained, whereas if $N_0 = WH$, the GPR method is obtained. Moreover, the proposed method can tradeoff between mesh quality and computational/memory cost by varying $N_0$ in the range $[N, WH]$.

Since $N_0$ can assume any value in the range $[N, WH]$, clearly we need a means for choosing $N_0$, or equivalently, the initial sampling density $D_0$ (where $N_0$ and $D_0$ are related by $N_0 = D_0 WH$). To help in determining an appropriate strategy for choosing $D_0$, the following experiment was conducted for several images and values of the desired sampling density $D$ (where $D = \frac{N}{WH}$). For the given image and value of $D$, we measured the mesh quality (in PSNR) as a function of $D_0$ while keeping $D$ fixed. Fig. 2 shows the results obtained for two such experiments, with each graph having two lines, one for each of the two GPRFS variants. (For each graph, the horizontal axis corresponds to values of $D_0$.)

From the preceding results, we can make a number of observations. The first is that the GPRFS-ED variant is vastly superior to the GPRFS-Random variant. That is, higher quality meshes (i.e., higher PSNR) can be obtained with the GPRFS-ED variant than the GPRFS-Random variant for nearly all values of $D_0$ (and certainly all values of practical interest). In other words, using the ED scheme to wisely select $S_0$ in the GPRFS method is highly effective, much more so than choosing $S_0$ randomly. This shows the effectiveness of the subset selection method in our proposed GPRFS-ED method. Since the GPRFS-ED variant is vastly superior to the GPRFS-Random variant, we only consider the first of these variants in the remainder of our work. With this in mind, let us continue to examine Fig. 2, focusing our attention only on the results obtained with the GPRFS-ED variant. A careful examination of these results leads to the following interesting observation: In both graphs, the maximum PSNR is not obtained when $D_0$ is 100%, the point at which the GPRFS-ED method becomes equivalent to the GPR scheme. More specifically, as $D_0$ is decreased from 100%, the PSNR climbs very slowly to a maximum value and then drops relatively rapidly thereafter. Due to this behavior, the GPRFS-ED method can be made to produce a higher quality mesh than the GPR method, provided that an appropriate choice of $D_0$ is made. Based on further experimentation, we found that, for $D$ values of practical interest (e.g., $D < \frac{1}{5} = 20\%$), the GPRFS-ED method usually achieves a PSNR very close to the GPR method if we choose $D_0$ to be about 4 to 5 times the value of $D$, with the best choice being closer to $5D$ when $D$ is very small. In the interest of minimizing computational/memory complexity, however, we elect to choose $D_0 = 4D$, incurring a small mesh-quality penalty relative to the GPR scheme when $D$ is very small. Thus, in our GPRFS-ED method, we recommend the following choice for $N_0$ in terms of $N$:

$$N_0 = \min\{4N, WH\}. \tag{1}$$

In the remainder our paper, all experiments involving the GPRFS-ED method choose $N_0$ in accordance with (1).



(a) (b)

**Fig. 2**. Effect of varying the initial sampling density $D_0$ on mesh quality for the GPRFS method. (a) `lena` with a desired sampling density $D$ of 4%; and (b) `peppers` with a desired sampling density $D$ of 2%.

## 6. RESULTS

Having introduced our proposed GPRFS-ED method, we now compare its performance to the GPR scheme in terms of mesh quality as well as computational and memory complexities.

MESH QUALITY. For several combinations of images and desired sampling densities, the GPRFS-ED and GPR methods were used to generate meshes, and the mesh approximation error in PSNR was measured. A representative subset of the results is given in Table 2. For further comparison purposes, we have also included the results obtained with the ED method (of Section 3). As can be seen from the results, our GPRFS-ED method outperforms the GPR scheme, except at very low sampling densities of about 1% or less where the GPR scheme yields slightly better results. As we will show later, however, at very low sampling densities, our GPRFS-ED method requires about 16.8 times less computation time and about 25 times less memory than the GPR scheme. So the small difference in mesh quality at the sampling density of 1% is arguably a small price to pay considering the savings in computational/memory cost. From the above results, we can also see that the GPRFS-ED method outperforms the ED scheme by a very large margin, demonstrating that the excellent performance of the GPRFS-ED method is not simply due to its use of the ED scheme alone. Lastly, we note

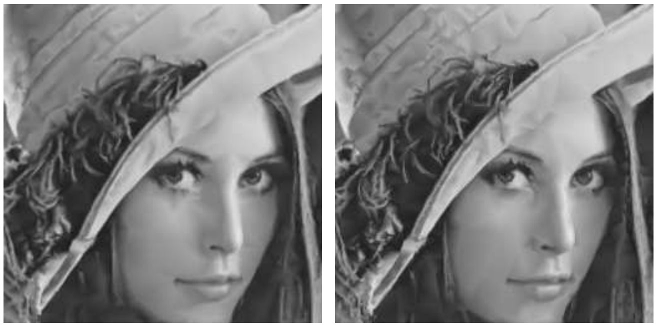
**Table 2**. Comparison of the mesh quality obtained with various methods for the (a) `lena` and (b) `peppers` images

(a)

| Samp.<br>Density<br>(%) | PSNR (dB) | | |
|---|---|---|---|
| | GPRFS-<br>ED | GPR | ED |
| 1.0 | 28.85 | **29.11** | 22.24 |
| 1.5 | **30.68** | 30.68 | 24.75 |
| 2.0 | **31.95** | 31.78 | 26.32 |
| 4.0 | **34.50** | 34.40 | 29.43 |
| 8.0 | **37.11** | 37.00 | 32.35 |

(b)

| Samp.<br>Density<br>(%) | PSNR (dB) | | |
|---|---|---|---|
| | GPRFS-<br>ED | GPR | ED |
| 1.0 | 29.85 | **30.05** | 22.23 |
| 1.5 | **31.57** | 31.55 | 24.84 |
| 2.0 | **32.55** | 32.40 | 26.33 |
| 4.0 | **34.43** | 34.20 | 29.78 |
| 8.0 | **36.11** | 35.76 | 32.04 |



(a) (b)

**Fig. 3**. Part of the image approximations obtained with the (a) GPRFS-ED (31.95 dB) and (b) GPR (31.78 dB) methods for the `lena` image at a sampling density of 2%.

that, in terms of subjective quality, the GPRFS-ED and GPR methods are quite comparable, with the former having a slight edge in some cases. An example illustrating the subjective quality achieved by the GPRFS-ED and GPR methods is provided in Fig. 3, where a small part of each mesh approximation is shown under magnification. In this example, the two methods clearly yield quite comparable results.

Time and Memory Complexities. Earlier, we claimed that our proposed GPRFS-ED method has significantly lower computational and memory complexities than the GPR scheme. We now present some results to substantiate our claim. First, we consider the time complexities of the methods. For several test images and desired sampling densities, we measured the time required to generate a mesh using each of the GPRFS-ED and GPR methods. A representative subset of these results (for the `lena` image) is shown in Table 3. These results show that our GPRFS-ED method requires anywhere from about 3 to 17 times less computation time than the GPR scheme, with the difference being most pronounced at low sampling densities. Clearly, our method offers a very substantial savings in computation time.

Next, we compare the memory complexities of the GPRFS-ED and GPR methods. In both methods, the memory usage is dominated by the mesh data structure and (to a lesser extent) a priority queue that contains one entry for each vertex in the mesh. Due to the similarities between the two methods, both employ identical data structures for representing the mesh and priority queue. Consequently, the peak memory usage for each method is approximately proportional to the peak number of vertices in mesh. In the GPR method, the peak mesh size is always $M_{\mathsf{GPR}} = WH$, while in the GPRFS-ED method, the peak mesh size is $M_{\mathsf{GPRFS}} = \min\{4N, WH\}$. Expressing the ratio $M_{\mathsf{GPR}}/M_{\mathsf{GPRFS}}$ in terms of the sampling density $D$, we obtain $M_{\mathsf{GPR}}/M_{\mathsf{GPRFS}} = 1/\min\{4D, 1\}$. Thus, for sampling densities from 1% to 10%, our proposed GPRFS-ED method requires from 25 to 2.5 times (respectively) less memory than the GPR scheme. Clearly, our method offers a very substantial memory savings.

**Table 3**. Comparison of the time complexities of the GPRFS-ED and GPR methods for the `lena` image

| Samp.<br>Density<br>(%) | Time (s) | | |
|---|---|---|---|
| | GPRFS-<br>ED | GPR | Ratio$^*$ |
| 1 | 3.47 | 58.41 | 16.8 |
| 2 | 5.39 | 57.39 | 10.6 |
| 4 | 9.26 | 56.30 | 6.0 |
| 8 | 17.37 | 54.02 | 3.1 |

$^*$ratio of the time for the GPR method to the time for the GPRFS-ED method

## 7. CONCLUSIONS

In this paper, we have proposed a new content-adaptive mesh-generation method, known as GPRFS-ED. Our GPRFS-ED method was shown to yield better quality meshes in terms of squared error than the highly-effective GPR method, at only a very small fraction of the computational and memory costs. The subjective quality of the meshes obtained with our GPRFS-ED method are also comparable to (if not slightly better than) those generated by the GPR method. Furthermore, with our proposed GPRFS-ED method, one can easily tradeoff between mesh quality and computational/memory complexity as needed. By improving upon the state of the art in mesh-generation methods, we help the numerous applications that can benefit from the use of content-adaptive mesh representations of images, with image coding being one such application.

## 8. REFERENCES

[1] Y. Yang, M. N. Wernick, and J. G. Brankov, “A fast approach for accurate content-adaptive mesh generation,” *IEEE Trans. on Image Processing*, vol. 12, no. 8, pp. 866–881, Aug. 2003.

[2] L. Demaret and A. Iske, “Advances in digital image compression by adaptive thinning,” in *Annals of the Marie-Curie Fellowship Association*, vol. 3, pp. 105–109. Marie Curie Fellowship Association, Feb. 2004.

[3] C. Dyken and M. S. Floater, “Preferred directions for resolving the non-uniqueness of Delaunay triangulations,” *Computational Geometry—Theory and Applications*, vol. 34, pp. 96–101, 2006.

[4] R. W. Floyd and L. Steinberg, “An adaptive algorithm for spatial greyscale,” *Proc. of the Society for Information Display*, vol. 17, no. 2, pp. 75–77, 1976.