Faster graph-theoretic image processing via small-world and quadtree topologies

Leo Grady and Eric L. Schwartz

***Abstract*— Numerical methods associated with graph-theoretic image processing algorithms often reduce to the solution of a large linear system. We show here that choosing a topology that yields a small graph diameter can greatly speed up the numerical solution. As a proof of concept, we examine two image graphs that preserve local connectivity of the nodes (pixels) while drastically reducing the graph diameter. The first is based on a "small-world" modification of a standard 4-connected lattice. The second is based on a quadtree graph. Using a recently described graph-theoretic image processing algorithm we show that large speed-up is achieved with a minimal perturbation of the solution when these graph topologies are utilized. We suggest that a variety of similar algorithms may also benefit from this approach.**

I. INTRODUCTION

TRADITIONAL solution methods to partial differential equations (e.g., finite differences, finite elements) often culminate in the solution of a large, sparse, symmetric system of linear equations where the sparsity pattern of the matrix corresponds directly to the topology of the sampling grid. Standard discretizations of 2D physical systems (e.g., heat flow, electrostatic fields), usually choose a topology based on a four- or eight-connected grid [1]. Graph-based image processing algorithms [2], [3], [4] typically take the pixels as the node set and connect the nodes locally with a four- or eight-connected edge set. Matrices associated with these graphs (e.g., the Laplacian, adjacency, or incidence matrix) possess a sparsity pattern defined by the graph topology [5], [6], as illustrated in Figure 1. Although a lattice is locally connected and shift-invariant (aside from the borders), there is no fundamental reason why an image should be restricted to this connectivity. We show here that alternate methods of choosing an image topology may significantly increase the speed and performance of graph-based algorithms that employ the conjugate gradients method to solve a set of linear equations. Although some algorithms explicitly require the solution to a sparse system of equations [7], it was pointed out in [8] that parabolic PDEs (e.g., the anisotropic diffusion of [9]) may be more efficiently placed in this form by using the backward Euler approximation to the time derivative rather than the forward Euler approximation.

Published in: Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Volume 2, pp. 360–365, 2004. For code and more information, see: http://www.cns.bu.edu/~lgrady

L. Grady is with the Imaging and Visualization Department of Siemens Corporate Research, Princeton, NJ 08540. This work was undertaken while the author was at the Department of Cognitive and Neural Systems, Boston University, Boston, MA 02215. E-mail: lgrady@scr.siemens.edu

E. L. Schwartz is with the Departments of Cognitive and Neural Systems and Electrical and Computer Engineering, Boston University, Boston, MA 02215, E-mail: eric@bu.edu

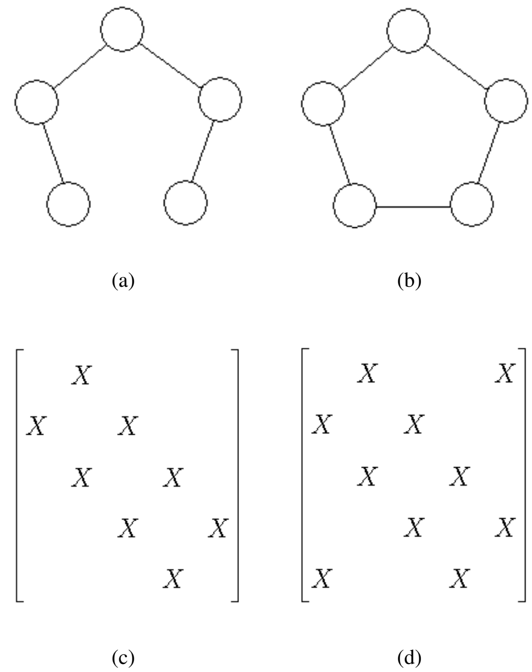


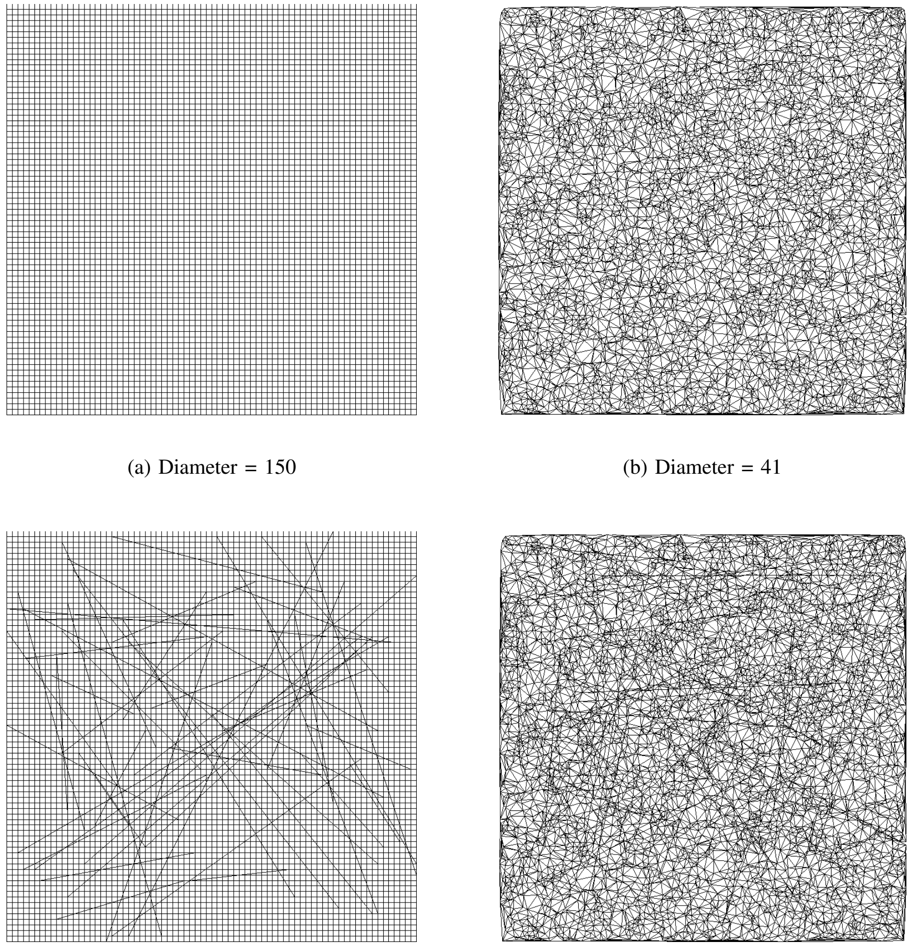
Fig. 1. (a,b) Example graphs. (c,d) Sparsity pattern of corresponding adjacency matrices.

A **graph** is a pair $G = (V, E)$ with vertices $v \in V$ and edges $e \in E \subseteq V \times V$. An edge, e, spanning two vertices, v_i and v_j, is denoted by e_{ij}. Let $n = |V|$ and $m = |E|$ where $|\cdot|$ denotes cardinality. A **weighted graph** has a value (typically nonnegative and real) assigned to each edge called a **weight**. The weight of edge e_{ij}, is denoted by $w(e_{ij})$ or w_{ij}.

Conjugate gradients is generally the algorithm of choice for solving a large, sparse, system of linear equations [10]. When applied to a matrix generated as a result of graph topology (e.g., Laplacian matrix, adjacency matrix), it has been shown [11], [12] that the rate of convergence for the conjugate gradients method is a function of the graph **diameter**. The diameter of a graph, G, is defined formally as

$$\text{diameter}(G) = \max(\min_{v_i, v_j \in V}(g(v_i, v_j))), \tag{1}$$

where $g(v_i, v_j)$ denotes the number of nodes traversed in the shortest path between two nodes (i.e., the length of the minimal *geodesic* between nodes v_i and v_j) [13]. In other words, the graph diameter is the maximum number of nodes traversed along an optimal path connecting two arbitrary nodes.



(a) Diameter = 150

(b) Diameter = 41

(c) Diameter = 46

(d) Diameter = 22

Fig. 2. (a) 75×75 Lattice substrate. (b) Delaunay triangulation substrate. (c) Small world graph built on a lattice substrate by adding 50 random edges. (d) Small world graph built on a Delaunay triangulation substrate by adding 50 random edges.

Here, we present two "small diameter" image topologies with desirable properties for graph-based image analysis algorithms:

- Small world [14]: A small number of edges (e.g., about 1% of the original number of edges) are added to E, with the nodes to be connected chosen at random.
- Multi-resolution quad-tree: A multi-resolution quad-tree is used to represent the image with explicit connections within (lattice) and across pyramid layers. Effectively, this introduces short paths via short-circuits through the higher levels of the quad-tree.

We demonstrate that both approaches significantly reduce the graph diameter and, as expected, improve the convergence rate of graph-based image processing algorithms requiring a solution by conjugate gradients.

A recently developed image segmentation algorithm, the **isoperimetric algorithm** [7], [15], is used to demonstrate the effects of the proposed topologies on the convergence of conjugate gradients and on segmentation quality. The main computational requirement of this algorithm is the solution to the system of linear equations given by

$$Lx = d, \tag{2}$$

where L is the weighted Laplacian matrix [16] defined by

$$L_{v_i v_j} = \begin{cases} d_i & \text{if } i = j, \\ -w(e_{ij}) & \text{if } e_{ij} \in E, \\ 0 & \text{otherwise,} \end{cases} \tag{3}$$

and d denotes the vector of (weighted) node **degree**. Specifically, d_i denotes the weighted degree of vertex v_i

$$d_i = \sum_{e_{ij}} w(e_{ij}) \;\; \forall \, e_{ij} \in E. \tag{4}$$

II. Convergence of the Conjugate Gradient Method

When using a graph-theoretic data structure, each iteration of conjugate gradients [10] propagates information along paths that are longer by one additional edge. For example, if x_0 represents an impulse function (i.e., $x_0 = [1, 0, \ldots, 0]^T$, a

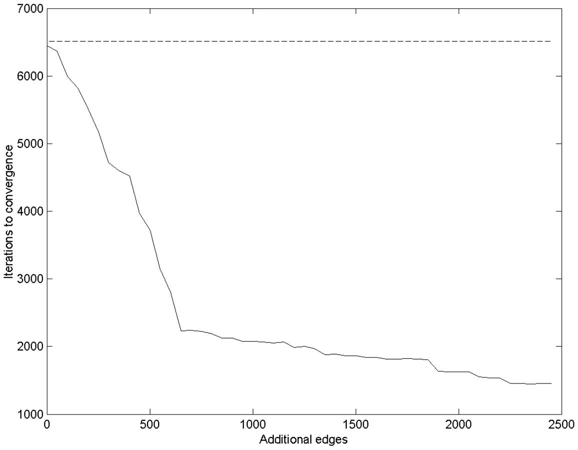


Fig. 3. Iterations required to converge on a solution for the isoperimetric algorithm with conjugate gradients as the number of random edges added increases for a 128×128 4-connected lattice, weighted to reflect the blood cells image of Figure 4. The dashed line represents the number of iterations required for convergence with the unaltered 4-connected topology.

nonzero value only at node v_0), then that impulse will have spread only k edges after k iterations. This analogy allows for the interpretation of the conjugate gradients method as a mixing process [11], [12].

This analogy can be made explicit by considering the solution to a diffusion process over a graph (e.g., discrete lattice) with discrete time steps. For the graph Laplacian matrix [16], L, and current state, x_i, the discrete diffusion equation may be written

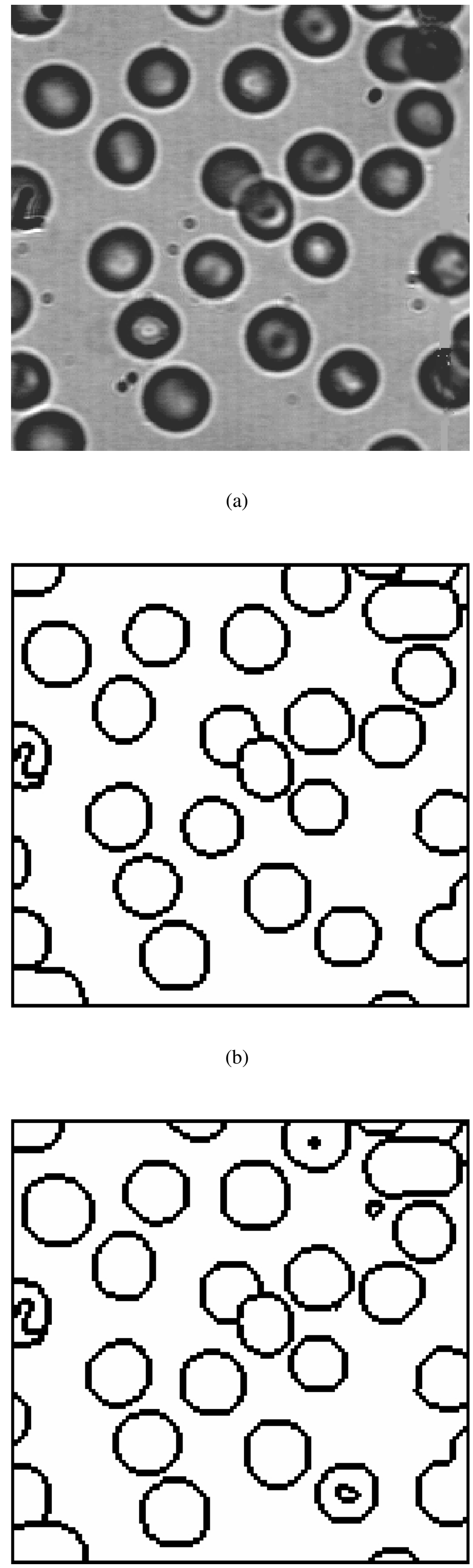
$$x_{i+1} = x_i + \Delta t L x_i. \quad (5)$$

Each iteration, x_i, is the sum of a polynomial in L multiplied by the vector representing the initial state x_0.

This analogy between the conjugate gradients method and mixing processes suggests that the rate of convergence of the conjugate gradients method will be a function of graph diameter [11]. In other words, since each iteration of the conjugate gradients method only spreads information along one additional edge with each iteration, the algorithm cannot converge, in general, until the information has spread to all nodes in the graph. Therefore, the minimum number of iterations is the length of the longest optimal path between any two nodes (i.e., the graph diameter).

We demonstrate two proposals for choosing a graph topology that increases the convergence rate of the conjugate gradient method. The first is to choose a locally connected topology (e.g., 4-lattice) and add in a small number of random edges. The second is to construct a conventional quad-tree over the image graph, allowing explicit connections between levels of the tree. We have constructed an image graph MATLAB toolbox, which is publicly available (the Graph Analysis Toolbox (`http://eslab.bu.edu:/software/graphanalysis`)). All figures in this paper are represented with the scripts (and corresponding public domain source images) that created them at this location.

To demonstrate the (heuristic) validity of the "small-world" approach, we consider the following questions for the cases of random, and regular (quad-tree) small-world topologies.

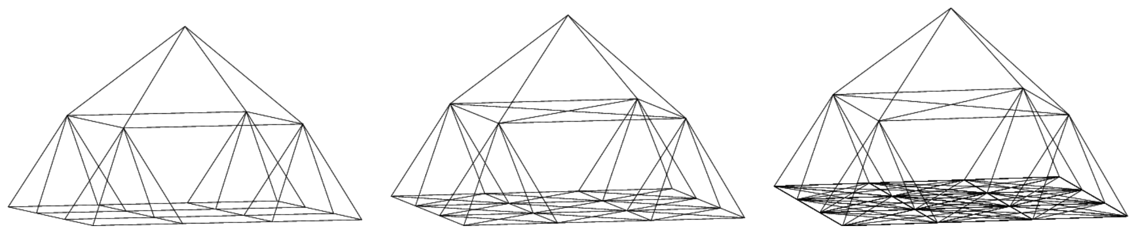


(a)

(b)

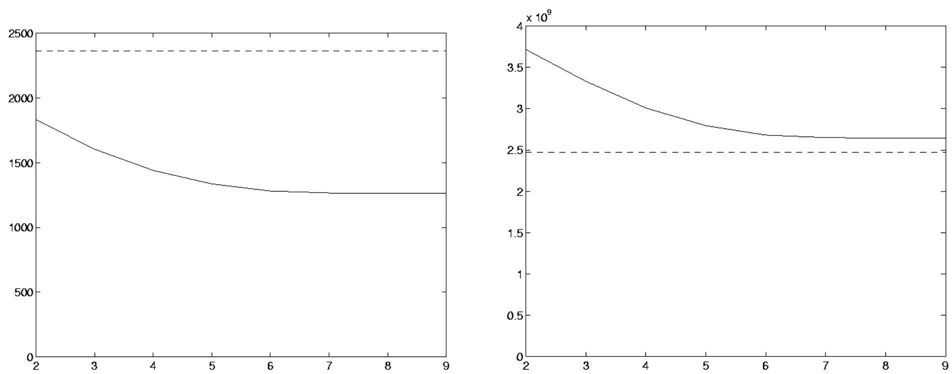
(c)

Fig. 4. (a) Original (input) image. (b) Segmentation obtained with unaltered 4-connected topology ($\beta = 95$, $\texttt{stop}= 10^{-5}$). (c) Segmentation obtained with the addition of 200 random edges ($\beta = 95$, $\texttt{stop}= 10^{-5}$). Preceding parameters refer to the weighting function in [7]



(a) (b) (c)

Fig. 5. Topology of the connected pyramid graph with 4-connected (a), 8-connected (b), and radius = 5 connected (c) within-level connections.



(a) (b)

Fig. 6. (a) Number of iterations required for convergence of conjugate gradients for equation (2) on a 512×512 unweighted lattice as the number of levels in the pyramid are added. Dashed line represents the number of iterations required to converge for a simple (non-pyramid) lattice. (b) Total number of `multiply` operations required to perform conjugate gradients as the number of levels increases. Dashed line represents the number of `multiply` operations required for a simple (non-pyramid) lattice.

1) What is the effect of the altered topology on the convergence rate of conjugate gradients?
2) What is the effect of the altered topology on the number of computations?
3) How does the topology change perturb the solution?

III. SMALL WORLDS

In their landmark paper [14], Watts and Strogatz define what they term a "small world" topology based on the six degrees of separation or small world phenomenon found in social networks. The defining property of a small world network is that it is locally connected (under suitable definition) while maintaining a small graph diameter.

Watts and Strogatz demonstrate that a graph with these properties may be obtained by "interpolating" between a typical, locally connected graph and the random graphs first defined by Erdös and Renyi [17], [18]. Most remarkably, Watts demonstrates [19] that a locally connected graph (the **substrate** graph) may be made into a small world graph (i.e., given a small diameter) with the addition of a small number of random edges. Figure 2 shows a lattice (4-connected) graph and a Delaunay triangulation after addition of a small number of random edges.

Based on the "small-worlds" intuition, the graph diameter is dramatically decreased by the addition of these new edges and the convergence rate of the iterative method should substantially increase. Furthermore, the additional computational cost due to these edges, per iteration, should be negligible since the number of new edges is small. Finally, since the number of long-range edges is "small", we conjecture that the difference between the solution to the problem using the "small-world" formulation and the solution to the original problem, is also small.

A. Results

The number of `multiply` operations per iteration in the conjugate gradients method is equal to the number of nonzero elements in the matrix. In the case of a 4-connected lattice, the number of nonzero elements, p, in the Laplacian matrix is $p \leq 5n$. Every random edge added incurs 2 additional nonzero elements (due to symmetry). Therefore, the amount of computation required (i.e., number of `multiply` operations) per iteration using a small world graph with a few extra edges is essentially the same as the computation required to process on the substrate graph.

Since the solution, x, clearly changes with a change in the underlying graph (i.e., a change in topology), it is useful to examine the effect of adding random edges on the solution. For purposes of applying the isoperimetric algorithm [7] to an image, the effect of a significant number of edges (as regards the number of iterations required for convergence) is shown in Figure 4 to have a minimal effect on the final solution. This is expected, since adding several hundred edges to an image of size 128×128 ($4n^2 = 64k$) is less than one percent.

IV. Quad tree

We first construct a pyramid of progressively coarser images and link them with the original in a typical quadtree topology. We term this a **connected pyramid** (see Figure 5).

In order to perform graph-based image processing, the connections within layers must also be made explicit. Taking the within layer topology to be the standard 4/8-connected or a radially connected topology [2] results in the three layer connected pyramids in Figure 5.

Although it is possible to define hierarchical arrangements of arbitrary graphs (e.g., through use of maximal independent sets [20]), we focus here on the standard Cartesian lattice. For purposes of simplicity, the values at each (parent) node in the higher level is taken as the average of the (child) nodes on the lower level.

The graph diameter in an $n \times n$ Cartesian lattice is $2n$, while the addition of each new level causes the graph to have half the diameter of the previous level, to a minimum diameter of $2\log_2(n)$ for a full quadtree pyramid. Therefore, despite the fact that the addition of new levels requires the solution of (2) for more nodes (to a limit of $\frac{4}{3}n \times n$), the graph diameter decreases dramatically with the addition of new levels, suggesting that conjugate gradients should converge faster. In the next section, the effect of decreasing graph diameter is shown to almost entirely compensate for the additional nodes in terms of computational efficiency.

A. Speed

In order to determine the mitigating effect of decreased graph diameter on the solution to (2), we varied the number of levels used in a 512×512 lattice with uniform weights and measured the number of iterations required for convergence of the conjugate gradients method. However, this measure can be misleading since the number of computations per iteration increases as the cardinality of the node and edge sets increases. In order to capture the computational efficiency of conjugate gradients in solving (2) on a lattice and a pyramid, the number of `multiply` operations required to solve (2) was also calculated. Figure 6 demonstrates that the number of iterations required for convergence decreases significantly as new levels are incorporated into the graph, such that the number of iterations required for convergence for a full pyramid is slightly greater than half that required for a lattice. The computational effect of reducing the number of iterations required for convergence is also displayed in Figure 6, demonstrating that the improved segmentations obtained from a pyramid architecture incur less than 7% additional computations. This result represents significant improvement over the additional computations of 33% expected by a an algorithm that is linear in the number of nodes.

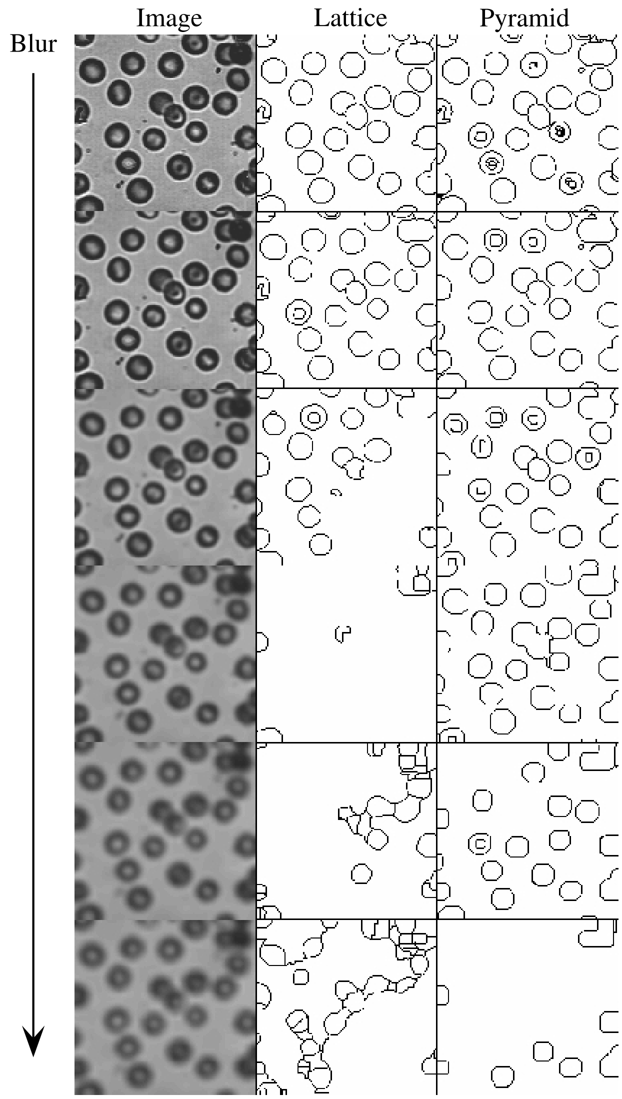


Fig. 7. Comparison of segmentations produced by lattice-based and pyramid-based isoperimetric algorithm in response to increased blur. Left: Image with increased variance Gaussian kernel (1–7 pixel variance). Middle: Lattice-based segmentation ($\beta = 95$, $\texttt{stop} = 1.0 \times 10^{-5}$). Right: Pyramid-based segmentation ($\beta = 180$, $\texttt{stop} = 2.0 \times 10^{-5}$).

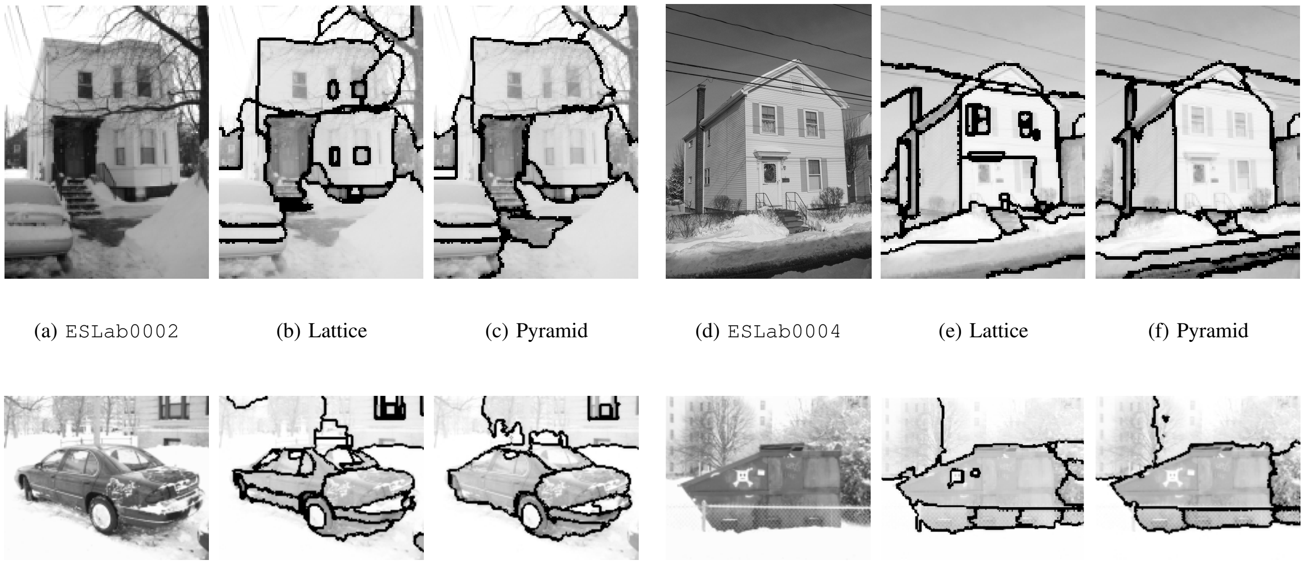
B. Segmentation quality

Due to the additional levels in a connected pyramid, more global information is used by the isoperimetric algorithm in determining good partitions. This additional global information generates improved localization of blurred boundaries, resulting in higher quality edge detection.

Since the connected pyramid based isoperimetric algorithm makes better use of blurred edges, we expect that the final segmentation on natural images will be improved. In Figure 8 the lattice-based and pyramid-based isoperimetric segmentations are compared for several natural images. One can see that difficult edges are better localized with the pyramid-based algorithm.

V. Conclusion

Our purpose in this paper was to use the connection between conjugate gradients and a mixing (i.e., diffusion) process to



(a) `ESLab0002` (b) Lattice (c) Pyramid (d) `ESLab0004` (e) Lattice (f) Pyramid

(g) `ESLab0027` (h) Lattice (i) Pyramid (j) `ESLab0031` (k) Lattice (l) Pyramid

Fig. 8. Comparison of images segmented with pyramid ($\beta = 180$, `stop=` 10^{-5}) and lattice ($\beta = 95$, `stop=` 10^{-5}) based isoperimetric algorithms. More examples of the segmentations produced by the pyramid-based algorithm may be found at `http://eslab.bu.edu/publications/grady2004faster/`.

motivate the design of image graph topologies when employing algorithms that require the solution to a system of linear equations.

Specifically, we have demonstrated that the conversion of a standard lattice to a "small world" graph through the addition of a small number of random edges results in large increases in the convergence rate with minimal effect on the final solution. We can expect the efficiency of the small world modified graph to increase as the image size grows, since the diameter of the unmodified lattice grows linearly with image size, while the diameter of the modified graph remains roughly constant, for the sizes of image graphs that we have investigated.

The connected pyramid graph introduces additional nodes and edges in an attempt to produce higher quality segmentation results by taking into account the image at multiple resolutions. We have demonstrated that the amount of computation required to process the graph that has been modified with significant additions to the node and edge set is reduced from the expected $4n^2/3$ for an $n \times n$ lattice to a much smaller amount. In the case of a 512×512 lattice, only 6% additional computations were required to compute a solution on the modified graph. We have demonstrated that the payoff for this slight increase in computation is an enhanced ability for the isoperimetric segmentation algorithm to detect blurred object boundaries and an overall increase in segmentation quality.

References

[1] Wolfgang Hackbusch, *Iterative Solution of Large Sparse Systems of Equations*, Springer-Verlag, 1994.

[2] Jianbo Shi and Jitendra Malik, "Normalized cuts and image segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 8, pp. 888–905, August 2000.

[3] Sudeep Sarkar and Padmanabhan Soundararajan, "Supervised learning of large perceptual organization: Graph spectral partitioning and learning automata," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 22, no. 5, pp. 504–525, May 2000.

[4] Yuri Boykov, Olga Veksler, and Ramin Zabih, "Fast approximate energy minimization via graph cuts," *IEEE Transacations on Pattern Analysis and Machine Intelligence*, vol. 23, no. 11, pp. 1222–1239, 2001.

[5] Norman Biggs, *Algebraic Graph Theory*, Number 67 in Cambridge Tracts in Mathematics. Cambridge University Press, 1974.

[6] John Gilbert, "Predicting structure in sparse matrix computations," *SIAM Journal of Matrix Analysis and Applications*, vol. 15, no. 1, pp. 62–79, 1994.

[7] Leo Grady and Eric Schwartz, "Isoperimetric graph partitioning for data clustering and image segmentation," Tech. Rep. CAS/CNS-TR-03-015, Boston University, Boston, MA, July 2003, Submitted to IEEE Pattern Analysis and Machine Intelligence.

[8] Mathieu Desbrun, Mark Meyer, Schröder, and Alan H. Barr, "Implicit fairing of irregular meshes using diffusion and curvature flow," in *Proceedings of the 26th annual conference on computer graphics and interactive techniques*. ACM Special Interest Group on Computer Graphics and Interactive Techniques, August 1999, pp. 317–324, ACM Press/Addison-Wesley Publishing Company.

[9] Pietro Perona and Jitendra Malik, "Scale-space and edge detection using anisotropic diffusion," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 12, no. 7, pp. 629–639, July 1990.

[10] Gene Golub and Charles Van Loan, *Matrix Computations*, The John Hopkins University Press, 3rd edition, 1996.

[11] Keith Gremban, *Combinatorial preconditioners for sparse, symmetric diagonally dominant linear systems*, Ph.D. thesis, Carnegie Mellon University, Pittsburgh, PA, October 1996.

[12] J. J. Dongarra, I. S. Duff, D. C. Sorenson, and H. A. van der Vorst, *Solving Linear Systems on Vector and Shared Memory Computers*, Society for Industrial and Applied Mathematics, Philadelphia, 1991.

[13] Frank Harary, *Graph Theory*, Addison-Wesley, 1994.

[14] Duncan Watts and Steven Strogatz, "Collective dynamics of 'small-world' networks," *Nature*, vol. 393, no. 6684, pp. 440–442, June 1998.

[15] Leo Grady, *Space-Variant Computer Vision: A Graph-Theoretic Approach*, Ph.D. thesis, Boston University, Boston, MA, 2004.

[16] Russell Merris, "Laplacian matrices of graphs: A survey," *Linear Algebra and its Applications*, vol. 197,198, pp. 143–176, 1994.

[17] Paul Erdös and Alfred Renyi, "On the evolution of random graphs," *Közlemények Publications*, vol. 5, pp. 17–61, 1960.

[18] Paul Erdös and Alfred Renyi, "On random graphs. i.," *Publicationes Mathematicae Debrecen*, vol. 6, pp. 290–297, 1959.

[19] Duncan J. Watts, *Small worlds: the dynamics of networks between order and randomness*, Princeton studies in complexity. Princeton University Press, Princeton, N.J., 1999.

[20] Stephen Barnard and Horst Simon, "A fast multilevel implementation of recursive spectral bisection for partitioning unstructured problems," Tech. Rep. RNR-92-033, NASA Ames Research Center, November 1992.