5. Acknowledgments

The authors thank Francisco Gomez, Fady Habra, Sunil Shende and Xun Xue for interesting discussions.

6. References

[AHMS94] Arkin, E., M. Held, J. Mitchell, and S. Skiena, "Hamiltonian triangulations for fast rendering," *Algorithms-ESA'94*, J. van Leeuwen, ed., Utrecht, NL, LNCS 855, pp. 36-47, September 1994.

[BE92] Bern, M. and Eppstein, D., "Mesh generation and optimal triangulation," in *Computing in Euclidean Geometry*, F. K. Hwang and D.-Z. Du, eds., World Scientific 1992.

[Bo79] Bown, J., *Injection Moulding of Plastic Components*, McGraw-Hill, New York, 1979.

[BT95] Bose, P. and Toussaint, G. T., "No quadrangulation is extremely odd," Tech. Report 95-03, Department of Computer Science, University of British Columbia, January 1995.

[Ch91] B. Chazelle, "Triangulating a simple polygon in linear time," *Discrete Comput. Geom.*, vol. 6, 1991, pp. 485-524.

[Chv75] V. Chvátal, "A combinatorial theorem in plane geometry," *J. Combinatorial. Theory Ser. B*, vol. 18, 1975, pp. 39-41.

[DFP85] De Floriani, L., Falcidieno, B. and Pienovi, C., "Delaunay-based representation of surfaces defined over arbitrarily shaped domains," *Computer Vision, Graphics & Image Processing*, vol. 32, 1985, pp. 127-140.

[ELOSU92] H. Everett, W. Lenhart, M. Overmars, T. Shermer and J. Urrutia, "Strictly convex quadrilateralizations of polygons," *Proc. of the 4th Canadian Conference on Computational Geometry*, St. Johns, Newfoundland, 1992, pp. 77-82.

[Fi78] S. Fisk, "A short proof of Chvátal's watchman theorem," *J. Combinatorial. Theory Ser. B*, vol. 24, 1978, pp. 374.

[He83] Heighway, E., "A mesh generator for automatically subdividing irregular polygons into quadrilaterals," *IEEE Transactions on Magnetic*s, 19, 6, pp. 2535-2538, 1983.

[Ho88] Ho-Le, K., "Finite element mesh generation methods: A review and classification," *Computer Aided Design*, vol. 20, 1988, pp. 27-38.

[JSK91] B. P. Johnston, J. M. Sullivan and A. Kwasnik, "Automatic conversion of triangular finite meshes to quadrilateral elements," *International Journal of Numerical Methods in Engineering*, vol. 31, No. 1, 1991, pp. 67-84.

[KM91] A. A. Kooshesh and B. M. E. Moret, "Three-coloring the vertices of a triangulated simple polygon," *Pattern Recognition*, vol. 25, 1992.

[La94] M. J. Lai, "Scattered data interpolation and approximation by using C^1 piecewise cubic polynomials," submitted for publication.

[LS94] M. J. Lai and L. L. Schumaker, "Scattered data interpolation using C^2 piecewise polynomials of degree six," *Third Workshop on Proximity Graphs*, Mississippi State University, Starkville, Mississippi, December 1-3, 1994.

[Lu85] A. Lubiw, "Decomposing polygonal regions into convex quadrilaterals," *Proc. Symposium on Computational Geometry*, 1985, pp. 97-106.

[O'R94] J. O'Rourke, *Computational Geometry in C*, Cambridge University Press, 1994.

[PS85] F. P. Preparata and M. I. Shamos, *Computational Geometry: An Introduction*, Springer-Verlag, New York, 1985.

[PT89] F. Preparata and R. Tamassia, "Fully dynamic point location in a monotone subdivision," *SIAM Journal on Computing*, vol. 18, 1989, pp. 811-830.

[Sc79] Schumaker, L. L., "On the dimension of spaces of piecewise polynomials in two variables," in *Multivariate Approximation Theory*, W. Schempp and K. Zeller, (eds.), Birkhauser Verlag, 1979, pp. 396-411.

[SNTM] V. Srinivasan, L. R. Nackman, J-M Tang and S. N. Meshkat, "Automatic mesh generation using the symmetric axis transformation of polygonal domains", *Proceedings of the IEEE (Special Issue on Computational Geometry)*, G. T. Toussaint, Guest Editor, vol. 80, No. 9, 1992, pp. 1485-1501.

[ST81] J.-R. Sack and G. T. Toussaint, "A linear-time algorithm for decomposing rectilinear star-shaped polygons into convex quadrilaterals," *Proc. 19th Annual Conf. on Communications, Control and Computing*, Allerton, 1981, pp. 21-30.

[ST85] J.-R. Sack and G. T. Toussaint, "Guard placement in rectilinear polygons," in *Computational Morphology*, Ed., G. T. Toussaint, North-Holland, 1988, pp. 153-175.

.

Theorem 3.3: *The percolate-up-and-down algorithm gives a quadrangulation of a simple n-gon with the minimum number of outer Steiner points required to quadrangulate the given triangulation. In particular, at most* $\lfloor n/3 \rfloor$ *outer Steiner points are used. Furthermore, this algorithm runs in O(n) time.*

We also introduce another variant of this algorithm, called the *Q-percolation* algorithm, which converts a triangulation to a quadrangulation while adding Steiner points *inside* the polygon (we call these *inner Steiner points*), with at most one outer Steiner point. Inner Steiner points are an important consideration when the goal is to quadrangulate a simple polygon without modifying the boundary of the polygon too much. We prove the following result.

Theorem 3.4: *The Q-percolation algorithm computes a quadrangulation of a simple n-gon with at most* $\lfloor n/2 \rfloor$ *inner Steiner points and at most one outer Steiner point in O(n) time.*

We conclude this section by mentioning an important feature of the Q-percolation algorithm, which is that it can be used to obtain quadrangulations of any triangulation (that is, not necessarily triangulations of simple polygons). Let *T* be any triangulation, as in the definition given at the beginning of this section. We can quadrangulate *T* by constructing a spanning tree of the dual graph of *T*, and then applying the Q-percolation algorithm to the resulting tree. Observe that the method used in the percolate-up-and-down algorithm is not particularly useful for the spanning tree of the dual graph of *T*. This is because the leaves of the spanning tree do not necessarily correspond to boundary triangles and hence unmatched leaves cannot be dealt with in the straightforward manner of the percolate-up-and-down algorithm.

It follows therefore that we can quadrangulate triangulated polygons with holes as well as triangulated line segments. The number of Steiner points required to quadrangulate these triangulations is $\lfloor t/2 \rfloor$, where *t* is the number of triangles in the triangulation.

4. Triangulated Sets of Points

Given a triangulation of a set of *n* points, the Q-percolation algorithm will find a quadrangulation using at most *n* - 2 Steiner points (since the number of triangles in a triangulation of *n* points is $\leq 2n$ - 4). Note that a set of points can always be quadrangulated with at most one Steiner point, since it is always possible to find a Hamiltonian triangulation of a set of points [BT95], [AHMS94]. On the other hand, here we are interested in converting a *given* triangulation to a quadrangulation.

In the remainder of this section we address the problem of obtaining a *strictly convex* quadrangulation of a set of points by adding a number of Steiner points. A solution for the problem of obtaining a strictly convex quadrangulation for a polygon by adding Steiner points is given in [ELOSU92], where it is shown that any *n*-gon can be decomposed into at most $5(n-2)/3$ strictly convex quadrilaterals. In the same paper, the following Lemma is proven:

Lemma 4.1: [ELOSU92] *Let F be a pentagon with one Steiner point s placed on its boundary. F can be divided into at most five strictly convex quadrilaterals by adding at most three Steiner points in the interior of F and no other point on the boundary.*

Let *T* be a hamiltonian triangulation of a set of points *S* (see [AHMS94, BT95]). If we consider six consecutive triangles in the path, we can use Lemma 4.1 to show that these triangles can be divided into at most ten strictly convex quadrilaterals by adding at most seven Steiner points in the interior. We thus obtain the following result, which we state without proof.

Theorem 4.2: *Let S be a set of n points with h vertices on the convex hull. A strictly convex quadrangulation of S can be obtained by adding at most* $7\lceil (2(n-1)-h)/6 \rceil$ *i.e., less than* 7*n*/3 *Steiner points and obtaining at most* $10\lceil (2(n-1)-h)/6 \rceil$ *i.e., less than* 10*n*/3 *quadrilaterals.*

We remark that *n*/2 Steiner points are sometimes necessary in order to obtain a convex quadrangulation of a set of *n* points.

matching of the dual graph. Each edge in M gives us a quadrangle and since M is perfect, there are no left-over triangles in the triangulation. It follows that we can obtain a quadrangulation of T without using Steiner points. This implies the following proposition.

Proposition: *A triangulation can be quadrangulated without Steiner points if and only if the dual graph of the triangulation admits a perfect matching.*

As we will see in the remainder of this paper, this relation between quadrangulations and matchings gives us a unified approach to handle the problem of quadrangulating (while possibly adding Steiner points) triangulations. In the rest of this section, we give algorithms which allow us to obtain a quadrangulation from any given triangulation in linear time. We will also give bounds on the number of Steiner points that may be necessary for the quadrangulation.

We now give the description of simple linear-time algorithms, which we call the *percolation algorithms*, that give us maximum matchings for binary trees. Note that we focus on binary trees because the graphs that are of interest to us are either duals of triangulations of polygons or the spanning trees of dual graphs of more general triangulations. The nodes in dual graphs have degree at most 3. (Some of these techniques will generalize to general trees, but we will not go into that here). Interestingly, the methods described here provide us with an alternative proof of Lemma 2.1. More importantly, the technique behind the percolation algorithms can be used to obtain quadrangulations of triangulations. In particular, we will be able to give upper bounds on the number of Steiner points for obtaining quadrangulations from triangulated polygons with holes and a triangulated set of line segments.

Let $T = (V, E)$ be a binary tree, and without loss of generality, we assume that T is rooted at a degree 1 node (this makes no difference to our algorithm, but makes the discussion simpler). Let h be the number of levels in T, with the root being at level 1. Consider now the following matching algorithm, which we call the *percolate-up algorithm*. Let V_h be the set of nodes at level h of T. Clearly all these nodes are leaves (note that not all leaves of T are in V_h). Let $v \in V_h$ and let $par(v)$ represent v's parent. We have the following cases:

Case 0: If $par(v)$ is a node of degree 1, then T consists of two nodes joined by an edge. In this case, match v and $par(v)$. Note that if $par(v)$ is NIL (i.e. v does not have a parent) then T consists of just a single node and we leave it unmatched.

Case 1: $par(v)$ is a node of degree 2. In this case, match v and $par(v)$.

Case 2: $par(v)$ is a node of degree 3 and v is the left child of $par(v)$. In this case, match v and $par(v)$.

Case 3: $par(v)$ is a node of degree 3 and v is the right child of $par(v)$. In this case, leave v unmatched.

For each $v \in V_h$, perform the above matching step and then prune T in the following way: If Case 0 applies, delete v and $par(v)$ from T. If Case 1 applies, delete v and $par(v)$ from T. Note that for every Case 2, there must be a Case 3. Hence if Case 2 applies, we delete v, $par(v)$ and v's sibling (v and $par(v)$ are matched, and v's sibling remains unmatched). After the matching and pruning steps have been carried out for all $v \in V_h$, we have a new tree whose height is either h - 1 or h - 2. Let $T^{(1)}$ denote this pruned version of T. Repeat the above matching and pruning step on all nodes at the lowermost level of $T^{(1)}$, and obtain a new pruned tree denoted by $T^{(2)}$.Continue this step with successively pruned trees $T^{(3)}$, $T^{(4)}$ and so on until we obtain $T^{(k)}$, where $T^{(k)}$ is the empty tree. Note that $k \leq h$. By using properties of maximum matchings we prove the following theorem.

Theorem 3.1: *The percolate-up algorithm finds a maximum matching for the tree T.*

The percolate-up algorithm gives a maximum matching in which some of the unmatched nodes are internal. However, we are interested in a maximum matching in which the unmatched nodes are at the leaves. This is because, for simple polygons, the quadrangulation can then be obtained immediately by adding a Steiner point for each unmatched node (which corresponds to a triangle in the triangulation). We can show that the maximum matching given by the percolate-up algorithm can be modified appropriately so that all unmatched nodes are now at the leaves.

Theorem 3.2: *There exists a maximum matching for a tree T such that all the unmatched nodes are leaves.*

Furthermore, we modify the percolate-up algorithm to give a linear-time algorithm for finding a maximum matching with all unmatched nodes at the leaves. We call this the *percolate-up-and-down algorithm*. Observe that this method gives the minimum number of outer Steiner points that are required to quadrangulate the given triangulation, since percolate-up-and-down finds a maximum matching for the dual tree. We thus have the following result.

gon.

To see that these Steiner points can be located in O(n) time, consider the following: P can be triangulated in O(n) time by Chazelle's algorithm [Ch91]. The triangulated polygon can then be three-colored in linear time with the algorithm of Kooshesh and Moret [KS92]. The edge on which a guard is placed gives us the fan-arm e outside which we place the Steiner point.To find an appropriate placement of the Steiner point, we may triangulate the simple polygon(s) that lie outside P and within the convex hull of P, which can also be done in linear time. The Steiner point for e can be placed anywhere inside the triangle incident on e (and outside P). If e is an edge of the convex hull, then it is not difficult to find a placement for the Steiner point. It follows therefore that all Steiner points can be located in O(n) time.

■

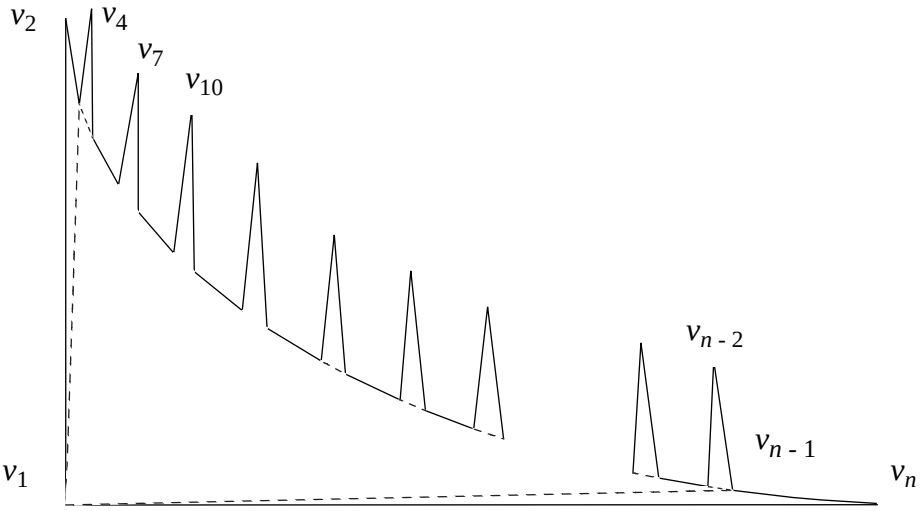

Figure 2.1: A quadrangulation of this polygon requires $\lfloor n/3 \rfloor$ outer Steiner points. This polygon admits only one triangulation (as shown).

3. Quadrangulations and Matchings

Consider a planar subdivision which has the property that every face is classified in one of three ways: an *outer face*, an *object face* or a *hole*. The outer face is the only unbounded face. Bounded faces that do not belong to the object are called holes. By a *triangulation*, we mean a planar subdivision in which every object face is a triangle and every edge of the subdivision belongs to at least one object face. From now on, when we use the phrase "triangle of the triangulation", we refer exclusively to an object face of the triangulation. The *dual graph* of a triangulation is the graph in which there is a node for every triangle of the triangulation, and an edge between two nodes if the corresponding two triangles share a side.

Given a graph $G = (V, E)$ (possibly weighted), a *matching* M on G is a set of edges such that no two of them have a common vertex. The *maximum cardinality* (weight) *matching* problem is that of finding a matching of maximum size (weight). A maximum cardinality matching is also referred to simply as a maximum matching. A *perfect matching* is a matching such that the cardinality of M is $|V|/2$.

When we obtain a quadrangulation from a triangulation, we would like to add as few Steiner points as possible. Consequently, the idea of pairing up triangles in a triangulation to form quadrangles immediately implies that our goal is to find the maximum possible number of such pairings. This corresponds exactly to the maximum cardinality matching problem for the dual graph of the triangulation.

If a triangulation T can be quadrangulated without Steiner points, it means that we can eliminate some of the edges of the triangulation so that the resulting set of object faces are quadrangles. In other words, all the quadrangles are formed by pairs of triangles that share a side. In the dual graph, consider the set M of edges defined by these pairs of triangles. M is a matching and is perfect (since T can be quadrangulated). Conversely, let M be a perfect

triangulation, we point out a powerful connection between quadrangulations and *perfect matchings* of the dual graphs of the triangulations in question. We obtain a variety of characterizations for when a triangulation (of some structure such as a polygon, set of points, line segments or planar subdivision) admits a quadrangulation without using Steiner points (or with a bounded number of Steiner points). We also investigate the effect of demanding that the Steiner points be added in the interior or exterior of a triangulated simple polygon. Furthermore, we propose efficient algorithms for accomplishing these tasks. For example, we show that a triangulated polygon may be quadrangulated in O(n) time using at most $\lfloor n/3 \rfloor$ *outer* Steiner points and that there exist polygons that require this many outer Steiner points. We give a method that quadrangulates a triangulated simple polygon with the minimum number of outer Steiner points required for that triangulation. We also show that a triangulated simple polygon may be quadrangulated with at most $\lfloor n/2 \rfloor$ Steiner points *inside* the polygon and at most one outside. We are also able to optimize certain properties of quadrangulations (such as convexity) by applying *maximum weighted* matching algorithms on the dual graphs of the given triangulations and associating suitable weights with each edge in the dual graph. In this extended abstract, due to space limitations, we illustrate a portion of our results and either sketch proofs or omit them altogether.

2. Triangulated Polygons

As pointed out in the previous section, not all polygons admit a quadrangulation. In such cases, it is necessary to add "Steiner points" (i.e. points that are not vertices of the original polygon) in order to quadrangulate the polygon. We define *outer Steiner points* to be Steiner points that are added outside the simple polygon. Each outer Steiner point p is affiliated with an edge e of the original polygon and modifies the boundary of the original polygon in the following way: the edge e is deleted and two new edges are created by connecting p to the two endpoints of e. Notice that we do not allow deletion of vertices of the original polygon. In this and the following section, we address the question of obtaining a quadrangulation of a simple polygon after it has been triangulated. The following theorem gives us bounds on the number of outer Steiner points that are required to quadrangulate a polygon.

Theorem 2.1: *$\lfloor n/3 \rfloor$ outer Steiner points are always sufficient, and sometimes necessary, to quadrangulate a simple polygon of n vertices. Furthermore, these Steiner points may be located in O(n) time.*

Proof: This observation follows immediately from Fisk's proof [Fi78] for Chvátal's art gallery theorem [Chv75]. In his proof, Fisk showed that every triangulation of an n-gon P can be partitioned into at most $\lfloor n/3 \rfloor$ *fans* (a *fan* is a triangulation where one vertex, called the *fan center*, is shared by all the triangles). Observe that there is always a decomposition such that these fans start and end at edges of the polygon; we will refer to such edges of P as *fan-arms*. It follows that a fan-arm can appear in only one fan.

Consider now a vertex v of P that is a fan center. Vertex v defines a sequence of triangles in the triangulation. These triangles can be paired up to form quadrilaterals. If the number of such quadrilaterals is odd, we will be left with one triangle, one of whose edges is a fan-arm e. One of the endpoints of e is v; let the other be v'. We can convert this to a quadrilateral by adding a Steiner point p outside e, deleting the edge e and connecting p to the two vertices v and v'.

Thus we need to add at most one Steiner point per fan. Since P can be partitioned into at most $\lfloor n/3 \rfloor$ fans, it follows that $\lfloor n/3 \rfloor$ outer Steiner points are always sufficient to quadrangulate a simple polygon.

In order to see that $\lfloor n/3 \rfloor$ outer Steiner points are sometimes necessary to quadrangulate a polygon, consider the polygon in Figure 2.1 (this is similar to an example of a polygon that requires $\lfloor n/3 \rfloor$ guards [Chv75]). There is only one way to triangulate this polygon, as shown in the figure. There are only three ways in which fans may be chosen:

- If v_1 is chosen as one of the fan centers, then the other fan centers must be the vertices v_4, v_7, v_{10},..... v_{n-2}. These fans consist of single triangles and hence they will each need one outer Steiner point for the quadrangulation.
- If v_3, v_6, v_9,.....v_{n-3}, v_n are chosen as the fan centers, each of the fans has an odd number of triangles, and hence each of them will need one outer Steiner point for the quadrangulation.
- If v_2, v_5, v_8,....,v_{n-1} are chosen as the fan centers, we have a case similar to the above.

We see that in each of the above cases, $\lfloor n/3 \rfloor$ outer Steiner points are necessary in order to quadrangulate the poly-

CONVERTING TRIANGULATIONS TO QUADRANGULATIONS*

Suneeta Ramaswami [1], *Pedro Ramos* [2], *Godfried Toussaint* [1]

1. Introduction

A central problem in the manufacturing industry concerns the simulation of a wide variety of processes, such as fluid flow in injection molding, by solving complicated systems of partial differential equations [Bo79]. To make this task easier, the method of *finite elements* is usually employed [Ho88]. In this approach a solid model of the object under study (or its bounding surface) is divided up into small pieces determined by data points sampled on the object's surface.

In scattered bivariate data interpolation one is required to construct a bivariate function (or surface) that fits data that has been collected at sampled points on the plane [Sc79]. One application of such a problem in the area of computer cartography is the construction of approximate models of terrains from data consisting of the elevation at a given finite set of sampled points [FFP85]. To facilitate this process the data points in the plane are used to divide it into small pieces. Each such piece then gives rise to a surface patch and these surface patches are finally "stitched" together to form the desired approximation to the surface.

One fundamental geometric problem in applications such as those mentioned above is the construction of a *mesh* from the given set of data points. For several decades the favored mesh used in such applications has been the *triangular mesh* or triangulation of the data points [DFP85]. In a triangular mesh the finite elements are, as the name implies, triangles. As a result, triangulations of sets (such as sets of points, line segments, polygons, etc.) have been studied in depth and much is known about them [BE92]. However, in some situations it is preferable that the finite elements be *quadrangles* (quadrilaterals) instead of triangles. For example, it has recently been shown that quadrangulations have several advantages over triangulations for the problem of scattered data interpolation [La94], [LS94]. Unfortunately, not much is known about quadrangulations of point sets. In fact, if edges are allowed to be inserted only between the given data points (i.e., no extra points called Steiner points are permitted) then not all sets of points admit a quadrangulation. The characterization of quadrangulations of point sets and the design of algorithms for their computation has only just begun [BT95].

Since little is known about computing quadrangulations, whereas triangulations have been well studied for a long time [Be92], engineers have devoted some attention to the problem of *converting* triangulations to quadrangulations [He83], [JSK91]. These methods however are heuristic, conceptually rather cumbersome and may require many Steiner points. For example, Johnston et al. [JSK91] integrate several heuristics into a system that automatically converts a triangular mesh into a quadrangular mesh which runs in $O(n^2)$ time and may add more than n Steiner points in the process. No attempts appear to have been made to optimize either the number of Steiner points or the complexity of the corresponding algorithms.

We remark that quadrangulations of polygons, on the other hand, have been investigated in the computational geometry literature for some time in the different context of guarding or illumination problems. First we note that, as with points, arbitrary simple polygons do not always admit a quadrangulation. In fact it is not difficult to construct polygons that require $\Omega(n)$ Steiner points in order to complete a quadrangulation. On the other hand *orthogonal* polygons (also isothetic or rectilinear) always admit a quadrangulation *without* Steiner points. In fact such polygons always admit quadrangulations in which every quadrangle is *convex*. This was first proved by Sack & Toussaint [ST81] for star-shaped polygons and subsequently generalized to arbitrary simple orthogonal polygons by Lubiw [Le85] and Sack & Toussaint [ST85] among others.

In this paper we study the problem of converting triangulated domains to quadrangulations, under a variety of constraints, from a formal perspective. For the simple case when no Steiner points are allowed, i.e., when it is asked whether a quadrangulation can be obtained simply by removing a carefully selected subset of edges of the

* Research of the first author was supported by NSERC Grant no. OGP0046218. Research of the second author was supported by NSERC Grant no. OGP0009293 and FCAR Grant no. 93-ER-0291. (1) School of Computer Science, McGill University, Montreal, Canada, (2) Dept. of Applied Mathematics, Universidad Politecnica de Madrid, Spain.