Drawing Stressed Planar Graphs in Three Dimensions [*]

Peter Eades and Patrick Garvan

Dept. of Computer Science, University of Newcastle,
University Drive, Newcastle NSW 2308 Australia
{eades,pgarvan}@cs.newcastle.edu.au

Abstract. There is much current interest among researchers to find algorithms that will draw graphs in three dimensions. It is well known that every 3-connected planar graph can be represented as a strictly convex polyhedron. However, no practical algorithms exist to draw a general 3-connected planar graph as a convex polyhedron. In this paper we review the concept of a stressed graph and how it relates to convex polyhedra; we present a practical algorithm that uses stressed graphs to draw 3-connected planar graphs as strictly convex polyhedra; and show some examples.

Key words: graph, stressed graph, convex polyhedron, reciprocal polyhedron

1 Introduction

It is well known that 3-connected planar graphs can be drawn as convex polyhedra. However, no practical algorithms exist to draw general 3-connected planar graphs as convex polyhedra. The two-dimensional (2D) drawing in Fig.1 is 3-connected and planar, and the corresponding polyhedron is drawn in Fig.2 as three different views. The 2D drawing in Fig.1 has large differences between the lengths of its edges and in its face areas making the graph difficult to understand, while the polyhedron in Fig.2 has more nearly equal edge lengths and face areas. The ratio of the shortest distance between vertex positions to the diameter of the set of of vertex positions is 0.01 for the 2D drawing in Fig.1, and 0.09 for the three-dimensional (3D) drawing in Fig.2. We say that a planar graph is *cluttered* if this ratio is low or *easy-to-look-at* if it is high.

A method for drawing any graph in 3D using straight line edges such that no pair of edges cross is presented in [2]. This method does not restrict the drawing to the integer grid. In [3] a method is provided for drawing any graph in 3D using straight line edges such that no pair of edges cross. All vertices are located on the integer grid and, for a graph of n vertices, the required grid size is $n \times 2n \times 2n$ which is shown to be optimal to within a constant. Both [2] and [3]

[*] This paper appears as Technical Report 95-02, Department of Computer Science, University of Newcastle, Newcastle NSW 2308, Australia.

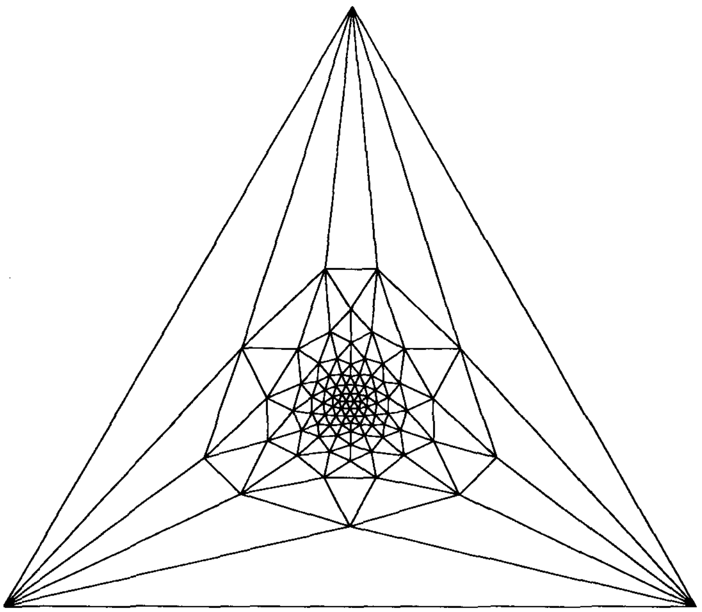


Fig. 1. Example of a cluttered 2D drawing of a planar graph.

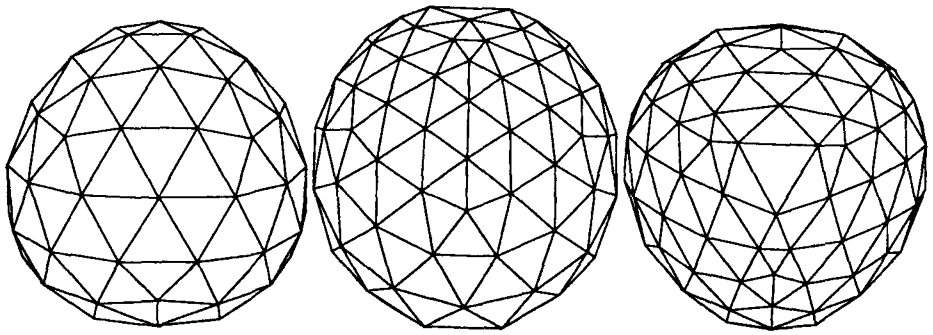


Fig. 2. Three views of the 3D polyhedron corresponding to Fig.1 as produced by Algorithm 1. The polyhedron is isomorphic to a three frequency geodesic sphere.

use the aesthetic criterion that no two edges should cross, and this is important when producing easy-to-look-at graphs. There are many other criteria that may be considered, including resolution and symmetry.

The following is a well-known theorem by Steinitz [7]:

Theorem 1 (Steinitz) *A graph G is the skeleton of a polyhedron P if and only if it is planar and 3-connected.*

This theorem guarantees that any 3-connected planar graph can be drawn as the vertices and edges of a convex polyhedron.

Some recent work [4] provides a linear-time algorithm for realizing 3-connected triangulated planar graphs as convex polyhedra. However, [4] only deals with triangulated graphs.

In this paper, we are concerned specifically with drawing 3-connected planar graphs as "strictly convex" polyhedra. A polyhedron P is *strictly convex* if the skeleton of the convex hull of the vertices of P is isomorphic with the skeleton of P. In section 2 we review the concept of "stressed graphs" and how they relate to polyhedra. We apply these concepts to drawing arbitrary 3-connected planar graphs as convex polyhedra. The time complexity of the algorithm is $O(n^{3/2})$, and the worst case "resolution" of the resulting drawings is at least exponential in n, where n is the number of vertices. In section 3 we show some examples of drawings as produced by the algorithm. Our experience with drawing polyhedra using Algorithm 1 has revealed a useful addition to our original algorithm. We conclude with some open problems in section 4. Note: some of the concepts in this paper are from [9].

2 The Algorithm

In this section we review the concept of stressed graphs (see for example [9]) and their relation to polyhedra. We give a definition of "reciprocal" polyhedra and present the main drawing algorithm. Finally, we discuss the time complexity of the algorithm and the "resolution" of the resulting drawings of polyhedra.

A *stressed graph* is a graph that is drawn in the plane such that each edge is drawn as a straight line segment and labelled with a real number which we will call a *stress*. The entire collection of stresses for a particular graph is called a *stress* on that graph. If each edge is considered to be a two-dimensional vector with x- and y-components equal to the length of the edge in the x- and y-directions multiplied by the stress on that edge, then we define an *equilibrium stress* on a graph to be a collection of edge stresses that produces at each vertex a zero vector sum of adjacent edge-vectors. Suppose every face of a graph is drawn as a convex polygon. We refer to the edges of the outer polygon as the external edges and the remaining edges as internal edges. If the internal stresses are positive while the outer stresses are negative, then an equilibrium stress on the graph is called a *convex equilibrium stress*. For convenience, a *restricted equilibrium stress* on a graph is defined as a collection of stresses where the inner stresses are positive and in equilibrium and no restrictions are placed on the outer stresses. [Note: Not every drawing of a 3-connected planar graph has a convex equilibrium stress.]

An example of a restricted equilibrium stressed graph is a Tutte [13] drawing of a 3-connected planar graph. This well known algorithm chooses a face of the graph and draws it as a convex polygon on the plane. The position of any internal vertex is then defined as the barycentre (or average) of the positions of its adjacent vertices. Fig.1 is an example of a Tutte drawing. It is clear that a collection of stresses for a such a drawing where each internal stress is $+1$ will be a restricted convex stress on the graph. [Note: Although Tutte's algorithm positions the internal vertices with the exact coordinates appropriate for a restricted equilibrium stress where each internal stress is +1, a simple variation of the algorithm will allow any collection of internal stresses to be used, as long as they are positive. Given such a collection, this variation on Tutte's algorithm will position each internal vertex so that the internal stresses are in equilibrium at each internal vertex, regardless of the shape or size of the outer polygon.]

We now describe the well known correspondence between convex stressed graphs and convex polyhedra first noted by Maxwell [11]. Consider an upturned bowl-shaped convex polyhedral surface Σ sitting on the plane H ($z = 1$) (see Fig.3(a)).

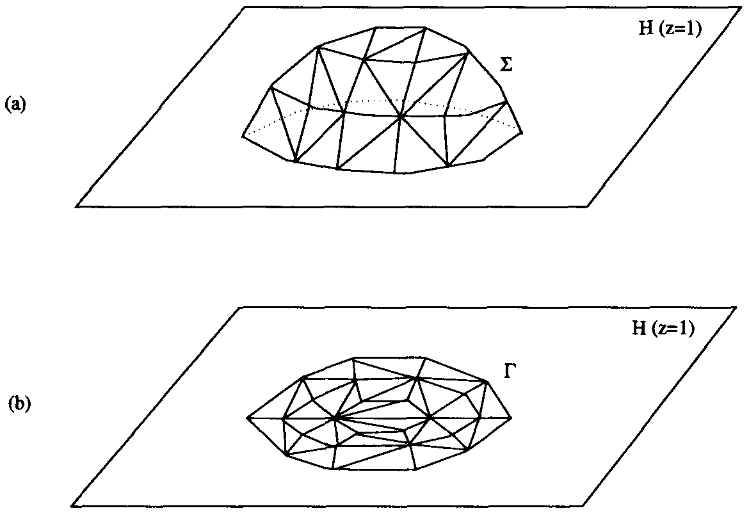


Fig. 3. (a) Upturned bowl shaped convex polyhedral surface Σ sitting on the plane H. (b) The 3-connected planar graph Γ resulting from an orthogonal projection of Σ down onto H. This projection is equivalent to flattening the bowl from above.

Let Γ be the drawing of the 3-connected planar graph resulting from an orthogonal projection of Σ down onto H (see Fig.3(b)), and ω be an edge label (or stress) for an edge of Γ defined using the following *stress equation*:

$$\omega = \frac{\varepsilon(r,s)(a_s(p_*)-a_r(p_*))}{[p_i,p_j,p_*]}$$

where:

- ω is the stress for the edge e of Γ joining vertices v_i and v_j (see Fig.4);
- $\varepsilon(r,s)$ is ± 1 according to the orientation of faces f_r and f_s whose intersection defines e – that is, $\varepsilon(r,s) = +1$ if in an anticlockwise ordering of the vertices around f_r, v_i precedes v_j, and $\varepsilon(r,s) = -1$ otherwise;

- p_i and p_j are the coordinates of v_i and v_j in Γ;
- a_r and a_s are piecewise-affine functions mapping, respectively, faces f_r and f_s of Γ up to the corresponding facets of Σ;
- p_* (a reference point) is any point on H that is not collinear with any edge of Γ;
- $[p_i, p_j, p_*]$ is the scalar triple product of p_i, p_j, and p_*, and may be calculated by the determinant $|p_i, p_j, p_*|$, where p_i, p_j, and p_* are written as column triples.

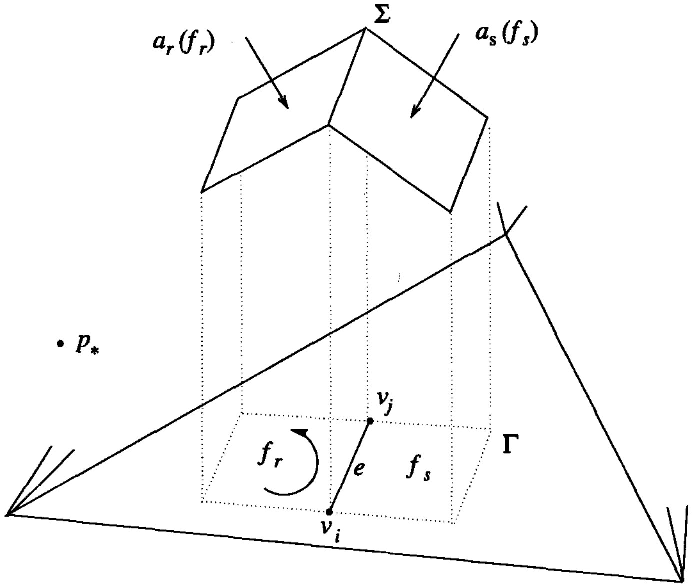


Fig. 4. Diagram illustrating the definitions used in the stress equation.

Informally, Γ records the skeletal structure of Σ, as projected onto H, and ω records the shape of Σ in the third dimension. A 3-connected plane graph Γ with an equilibrium stress ω may be written (ω, Γ).

Given the above definitions, the following theorem formally describes the bijectivity:

Theorem 2 ([9]) *Suppose (ω, Γ) is a stressed graph in H and Σ is the corresponding polyhedral surface sitting over Γ. Then Σ is the boundary of a convex polyhedron if and only if Γ and ω are convex.*

Given a polyhedron P, a *dual polyhedron* P^* is a polyhedron with a skeleton that is the graph-theoretic dual to the skeleton of P. A specific type of dual polyhedron is called a "reciprocal". Given a reference sphere S with equation $x^2 + y^2 + z^2 = r^2$, every plane $ax + by + cz = r^2$ has a reciprocal point (a, b, c)

defined with respect to S. Similarly, every point (a, b, c) (except the origin) has a reciprocal plane $ax+by+cz = r^2$ defined with respect to S. If P is a polyhedron, then a *reciprocal* polyhedron P^* with respect to S is defined as that polyhedron with vertices that are reciprocal to the facet-planes of P, and with facet-planes that are reciprocal to the vertices of P. Further, if P is convex and contains the origin, then the reciprocal polyhedron P^* will also be convex. (For more information see [1] and [8].)

Algorithm 1 *Input:* A planar embedding of a 3-connected planar graph G
Output: A strictly convex polyhedron P

1. If G does not contain a C_3 (a cycle of three edges), then replace G by its graph-theoretic dual G^*.
2. Choose any C_3 in G and draw it as a triangle on the plane $z = 1$.
3. Using Tutte's algorithm [13] draw the remaining vertices and edges of G producing a planar drawing Γ that has a restricted convex equilibrium stress. Calculate the remaining three external stresses on G [9] producing a convex equilibrium stressed graph.
4. Perform a breadth-first-search (BFS) of the face-lists of G, starting at the outerface. When visiting each face, use the stress equation to calculate the plane-equation for the corresponding facet of the polyhedron P.
5. If necessary, replace P with a convex reciprocal P^* of P.

□

Theorem 3 *Algorithm 1 draws 3-connected planar graphs as strictly convex polyhedra.*

Proof: We shall prove the correctness of the algorithm step by step.

Step 1: If a 3-connected planar graph does not contain a C_3 (a triangle), then its graph-theoretic dual will contain at least eight triangles. Therefore, if G does not contain a triangle, then its graph-theoretic dual G^* will contain a triangle.

Step 2: Step 1 guarantees that G will contain a triangle. Step 3, the Tutte drawing, requires that the outerface is a convex face. A triangular face will always be convex.

Step 3: If the outer face is convex, a Tutte drawing of a 3-connected planar graph will always be a convex plane drawing of the graph. That is, each face of the graph will be convex. Further, by definition of a Tutte drawing, the internal stresses will all be $+1$. In [9] it is shown that external equilibrium stresses for such a drawing (ie one with an outer polygon that is a triangle) can always be calculated and are unique. Thus, we have a convex equilibrium stressed graph.

Step 4: Given a convex equilibrium stressed graph, there exists a strictly convex polyhedron defined by the stress equation with a skeleton that is isomorphic with G. Using this equation, we can produce a strictly convex polyhedron.

Step 5: Every polyhedron has at least one reciprocal polyhedron. If a polyhedron is convex, then it is possible to find a reciprocal polyhedron that is also convex. Furthermore, the skeleton of a polyhedron is dual to the skeleton of a reciprocal of that polyhedron. □

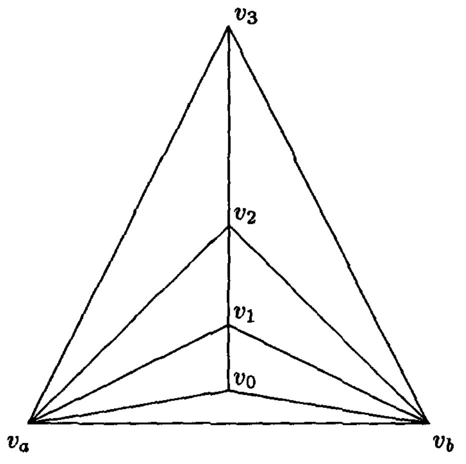
Theorem 4 *Algorithm 1 requires $O(n^{3/2})$ time.*

Proof: Step 3 requires $O(n^{3/2})$ time to perform a Tutte drawing [12]. Every other step requires at most $O(n)$ time. Therefore the overall time complexity is $O(n^{3/2})$. □

Unfortunately, Algorithm 1 does not always produce drawings with good "resolution", where *resolution* refers to the ratio of the smallest distance between vertex positions to the diameter of the set of vertex positions.

Lemma 5 *The worst case resolution of a Tutte drawing is $\Omega(k^n), k > 1$.*

Proof: Consider a Tutte drawing of the following 3-connected planar graph:



If v_a and v_b are each defined to have a y-coordinate of zero ($y_a = y_b = 0$); and v_0 is defined to have a y-coordinate of 1 ($y_0 = 1$), then, by definition of a Tutte drawing, v_1 will have a y-coordinate of $3y_0$ ($y_1 = 3y_0$). Now, $y_2 = 4y_1 - y_0$, and similarly for y_3. In this way, the maximum y-coordinate for a similar graph of n vertices can be found by solving the recurrence $y_i = 4y_{i-1} - y_{i-2}$; where $y_0 = 1$ and $y_1 = 3$. Using standard techniques [10], the solution to this recurrence equation shows that $y_n = \Theta(k^n)$, where k is a constant and $k > 1$. Therefore, the worst case resolution of a Tutte drawing is $\Omega(k^n), k > 1$. □

Theorem 6 *Algorithm 1 produces a drawing of a polyhedron with at least exponential worst case resolution.*

Proof : By Lemma 4, the resolution of a Tutte drawing is at least exponential in the worst case, therefore the resolution of the drawing produced by Algorithm 1 will be at least exponential in the worst case. □

3 Examples

In this section we will show some examples of drawings as produced by Algorithm 1, and suggest a possible extension of the existing algorithm so as to produce more sphere-like polyhedra. Unlike Fig.2, the drawings of polyhedra in this section are 2D projections of vertices and edges and do not explicitly show the facets. However, we have attempted to choose projections that clearly illustrate the shapes of the polyhedra. In some instances, we have drawn the polyhedron as a stereoscopic-pair, where the left drawing is the view as seen by the left eye and the right drawing is the view as seen by the right eye. A 3D effect may be produced using these pairs by looking at a point behind the page, thus causing the two drawings to overlap. An inverted image may also be produced by crossing one's eyes in order to overlap the drawings.

Figures 5 and 6 show drawings of the skeleta of the Platonic solids and the corresponding polyhedra as produced by Algorithm 1. The resulting polyhedra are isomorphic with the Platonic solids but are not regular. (For more information on the Platonic solids see [1].)

In Fig.6, the polyhedra are the direct result of a reciprocation operation (i.e. the calculation of a reciprocal polyhedron), since the skeleta of the cube and the dodecahedron have no C_3 faces. The polyhedron in drawing 6.II(a) has facets of nearly equal area and shape, as does the polyhedron in drawing 6.II(b). This near-regularity of facet area and shape is in most part due to the reciprocation step at the end of Algorithm 1. Applying *two* suitable reciprocation operations to each of the polyhedra in column II of Fig.5 will have the same effect. That is, it is possible to apply a double reciprocation to these polyhedra in such a way that the large facets will be shrunk and the small facets will be expanded. The polyhedra in 5.III are the result of such a double reciprocation.

The drawings in 5.III are nicer than those drawings in column 5.II. The most notable attributes are the approximate spherical shape of the polyhedra, and the relatively uniform facet areas and edge lengths.

This suggests changing the last step of Algorithm 1 to perform a suitable double reciprocation in those cases when the dual was not constructed. That is, when the input graph G contains at least one C_3.

Figure 7.II(a) shows a drawing of a polyhedron isomorphic to a twin cube. This polyhedron is strictly convex, where a more intuitive drawing may only be weakly convex.

Figure 7.II(b) shows the drawing of a polyhedron isomorphic to a pentagonal prism. This polyhedron is strictly convex, however no two adjacent facets are at right angles to each other.

Figure 8.III shows the drawing of a polyhedron as produced by Algorithm 1 with a skeleton that is isomorphic to the octagonal mesh shown in Fig.8.I. Figure 8.III is a bad drawing since there are large facets and many small facets clustered together. Figure 8.II is the polyhedron created by Algorithm 1 prior to the reciprocation step. This polyhedron has many long thin facets clustered down one side, and so reciprocation about a single internal point results in a cluster of small facets in the final polyhedron.

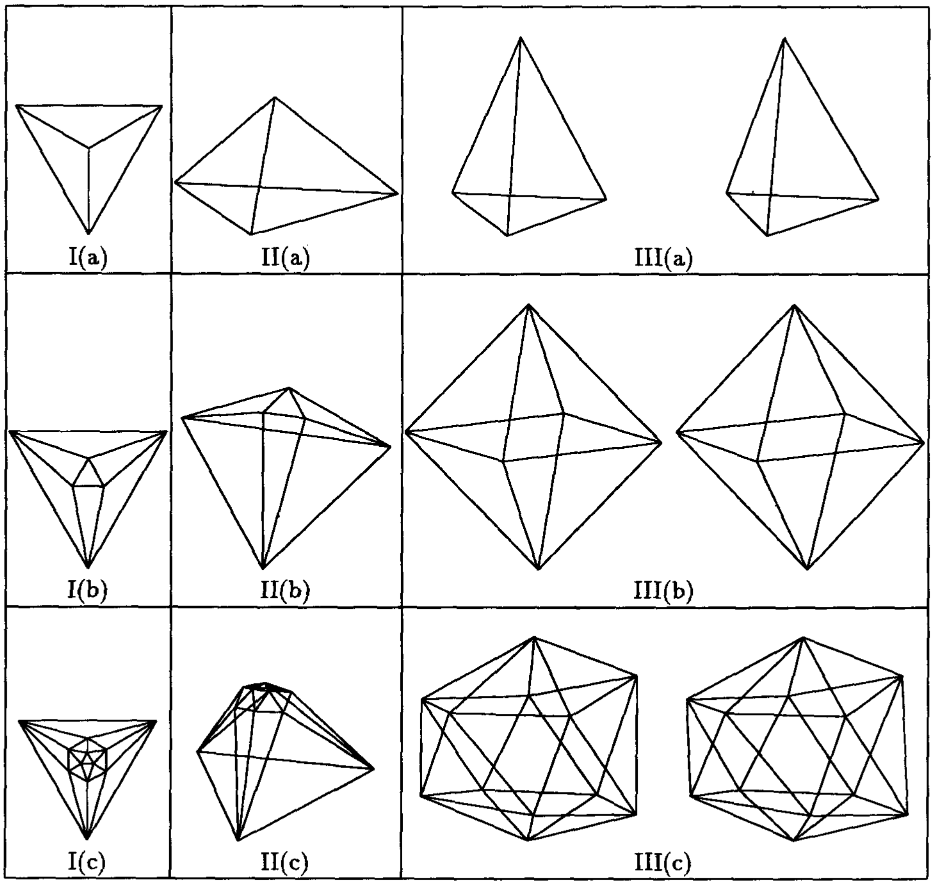


Fig. 5. (I) 2D planar drawings of the skeleta of (a) a regular tetrahedron, (b) a regular octahedron, and (c) a regular icosahedron; (II) the corresponding polyhedra as produced by Algorithm 1; and (III) stereoscopic-pair drawings of the polyhedra resulting from a double reciprocation. Note that these polyhedra consist entirely of triangular facets.

In [5] there are drawings of some 3-connected planar graphs in 2D, and in some cases the result is similar to a projection of a 3D polyhedron into 2D. In [6] there are drawings of some 3-connected planar graphs in both 2D and 3D, however the resulting drawings are not polyhedral in the sense that for any particular face of a graph the vertices of that face are not necessarily drawn so that they lie on the same plane in 3D.

As mentioned in the introduction, Fig.1 is the skeleton of a 3 frequency geodesic sphere, and Fig.2 shows three views of the corresponding polyhedron as produced by Algorithm 1. The line-hiding utilised in Fig.2 emphasises the fact that Algorithm 1 draws 3-connected planar graphs as strictly convex polyhedra, where each face of the graph is drawn in 3D as a single facet.

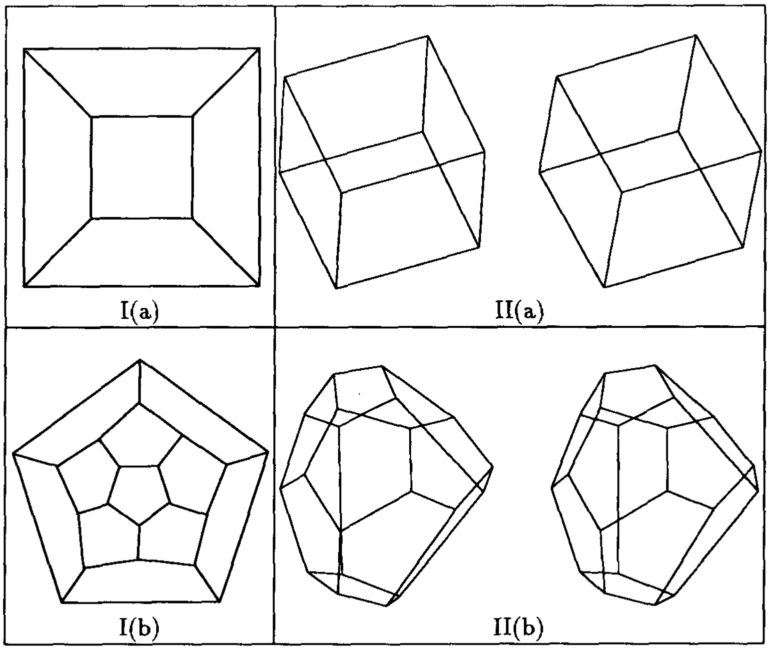


Fig. 6. (I) 2D planar drawings of the skeleta of (a) a regular hexahedron (or cube) and (b) a regular dodecahadron; and (II) stereoscopic-pair drawings of the corresponding polyhedra as produced by Algorithm 1. Note that these polyhedra consist entirely of non-triangular facets.

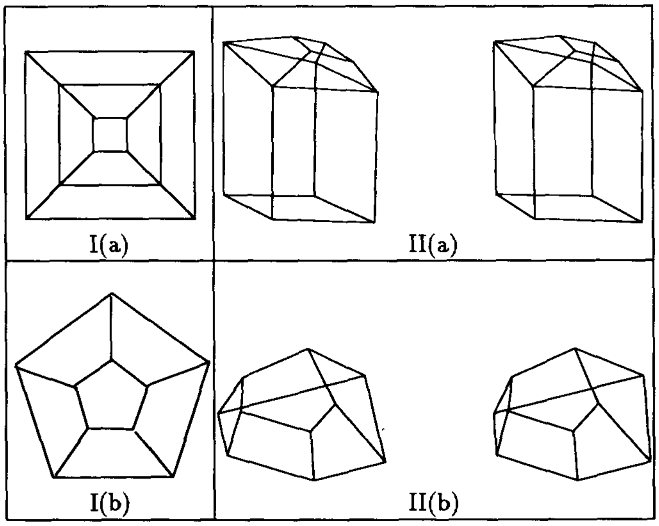


Fig. 7. (I) 2D planar drawings of the skeleta of (a) a twin cube and (b) a pentagonal prism; and (II) stereoscopic pair drawings of the corresponding polyhedra as produced by Algorithm 1.

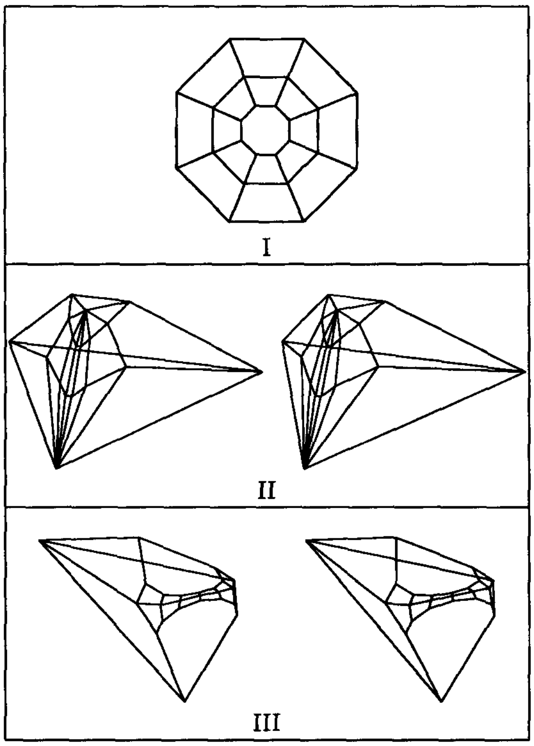


Fig. 8. (I) 2D planar drawing of an octagonal mesh; (II) the corresponding *dual* polyhedron as produced by Algorithm 1 prior to the reciprocation step; and (III) stereoscopic-pair drawing of the polyhedron produced by Algorithm 1 after the reciprocation step.

4 Conclusion and Open Problems

We have described an algorithm that will draw 3-connected planar graphs as strictly convex polyhedra. This algorithm requires $O(n^{3/2})$ time, and produces drawings of polyhedra with a resolution that is at least exponential in the worst case. We have given example drawings of polyhedra as produced by Algorithm 1, some of which were improved by a suitable double reciprocal operation.

The following are some open problems:

(i) Is there an algorithm that will draw 3-connected planar graphs in 2D as convex equilibrium stressed graphs with a resolution that is polynomial in the number of vertices?

(ii) What is the largest ratio of the diameter of the smallest spherical shell that encloses a polyhedron produced by Algorithm 1 to the diameter of the largest spherical shell enclosed by the same polyhedron, where the two spherical shells are concentric? Such sphericity of a polyhedron is a good aesthetic for identifying nice drawings of polyhedra.

(iii) Where should vertex labels be placed after drawing a 3-connected planar graph as a polyhedron? One possibility is to draw the dual of such a graph as a convex polyhedron and place labels on the facets. The facet adjacencies of such a drawing correspond exactly to the vertex adjacencies of the original graph.

(iv) What is the best way to display a three-dimensional graph? In particular, what is the best way to display a three-dimensional graph on a static two-dimensional medium such as paper? Possibilities include red/green anaglyphs and stereoscopic pairs.

Acknowledgements

We would like to thank Bob Cohen, Mike Goodrich, and Mike Houle for their useful comments on earlier versions of this paper.

References

1. H. S. M. Coxeter. *Regular Polytopes.* Dover, NY, 1973.
2. I. Cahit. Drawing the Complete Graph in 3-D with Straight Lines and Without Crossings. *Bulletin of the ICA*, Vol 12, Sep 94.
3. R. F. Cohen, P. Eades, T. Lin, and F. Ruskey. Three-Dimensional Graph Drawing. Proc. Graph Drawing 94. *Lecture Notes in Computer Science.* Volume 894, 1-11. Springer-Verlag, Berlin, 1995.
4. G. Das, M. T. Goodrich. *On the Complexity of Approximating and Illuminating Three-Dimensional Convex Polyhedra.* To appear, WADS 1995.
5. R. Davidson and D. Harel. *Drawing Graphs Nicely Using Simulated Annealing,* Technical Report CS89-13, Department of Applied Mathematics and Computer Science, The Weizmann Institute of Science, Rehovot, Israel, July 1989.
6. T. M. J. Fruchterman and E. M. Reingold. Graph Drawing by Force-directed Placement. *Software – Practice and Experience*, vol. 21(11), 1129-1164 (Nov 1991).
7. B. Grünbaum. *Convex Polytopes.* Wiley, NY, 1967.
8. B. Grünbaum and G. C. Shephard. Duality of Polyhedra. In *Shaping Space: A Polyhedral Approach.* Birkhauser, Boston, 1988.
9. J. E. Hopcroft and P. J. Kahn. A Paradigm for Robust Geometric Algorithms. *Algorithmica* (1992) 7:339-380.
10. C. L. Liu. *Introduction to Combinatorial Mathematics.* McGraw-Hill, NY, 1968.
11. J. C. Maxwell. On Reciprocal Figures and Diagrams of Forces. *Phil. Mag.* S. 4. vol. xxvii. (1864) pp.250-261
12. T. Nishizeki and N. Chiba. Planar Graphs: Theory and Algorithms. *Annals of Discrete Mathematics*, Volume 32. North-Holland, Netherlands, 1988.
13. W. T. Tutte. How to Draw a Graph. *Proc. London. Math. Soc.* (3), 13 (1963), 743-768.