Variations on Angle Based Flattening

Rhaleb Zayer, Christian Rössl, and Hans-Peter Seidel

Max-Planck-Institut für Informatik Stuhlsatzenhausweg 85, 66123 Saarbrücken
{zayer,roessl,seidel}@mpi-sb.mpg.de

1 Introduction

Surface parameterization is a fundamental problem in computer graphics. Intuitively, we can think of it as the *flattening* of a surface to a valid planar configuration, i.e. one without foldovers or self-intersections. More formally, consider a surface that is homeomorphic to a disk. Then the goal is to find a bijective mapping from a parameter domain to the surface, that fulfills certain quality constraints. For a triangulated surface this is a piecewise linear mapping between the original and an isomorphic planar mesh.

The importance of the problem makes surface parameterization a very active field of research. Numerous approaches have been proposed so far, inspired by results from different areas of research. Tutte [21] starts from graph theory and uses barycentric maps for embedding a planar graph. Floater's shape-preserving weights [6] improve the conformality of the mapping while still guaranteeing bijectivity. Eck et al. [5] use discrete harmonic maps to minimize angular distortion. Sander et al. [17] introduce a stretch metric to reduce the distortion induced by the parameterization. All the above methods require a predefined convex boundary in the parameter domain. Hormann and Greiner construct a most-isometric parameterization [12] by minimizing a nonlinear deformation functional without need to fix the boundary. Desbrun et al. [3] and Levy et al. [13] achieve quasi-conformal mappings with an evolving boundary by solving linear systems based on Cauchy-Riemann equation and harmonic energy minimization respectively. Other recent approaches apply multi-dimensional scaling [22] or an iterative algorithm that locally flattens the triangulation until a prescribed distortion bound is reached [20].

While quasi-conformal parameterizations such as [3, 5, 11, 13, 16] propose several schemes to minimize angular distortion, it seems natural to formulate the problem in terms of interior angles of the flat mesh. This leads to the *Angle Based Flattening*(ABF) method introduced by Sheffer and de Sturler [18]. The ABF algorithm constructs such a parameterization by minimizing a functional that punishes the angular distortion of the planar mesh w.r.t. the an-

gles of the original mesh. A set of linear and non-linear equality constraints on the planar angles guarantees the validity of the parameterization. These constraints however do not prevent the boundary from self-intersection. Hence a post-processing of the flat mesh is needed to handle edge crossings at the boundary. Each post-processing step first identifies the nodes causing intersections in the flat mesh, then it adds constraints on the local configurations in order to avoid intersections. The flat mesh is recomputed as a solution of the updated nonlinear system. The post-processing algorithm is repeated until no more intersections are found.

2 Overview

In this paper, we discuss several approaches to effectively reduce the computational effort involved in *Angle Based Flattening* and discuss algorithms to effectively solve the parameterization problem. The complexity of the constrained optimization problem raised in the ABF method makes finding a solution in reasonable time a very challenging problem. Several numerical schemes have been proposed to speed up the convergence of the original algorithm [18] by using preconditioning [14] and smoothing [19]. We take a completely different approach by identifying the main reasons that hinder convergence within the setting of the constrained problem itself. In fact, the post-processing might be very expensive as it tries to find intersections and then solve the whole non linear system as many times as needed.We take advantage of a characterization of convex planar drawings of tri-connected graphs to eliminate boundary intersections in the first place. This way we can steer or even avoid post-processing. Having this characterization in hand, it can be used in association with different objective functions that reflect the criteria we would like to minimize. Such functions can be described as the angular distortion [18] or the MIPS energy introduced by Hormann and Greiner [12] as both can be expressed completely in terms of angles.

The non-linear equations in the ABF method lead to a dense sparsity pattern of the Hessian matrix of the system which increases the computational cost. We show how the convergence can be improved alternatively by a simple yet effective transformation of the problem that relaxes the non-linear equality constraints. In fact, the Hessian becomes diagonal and its sparsity pattern becomes independent of the valences of the vertices of the input mesh. Since, the system of equations is symmetric we opt for the more appropriate symmetric numerical solvers instead of the non-symmetric ones proposed in [18, 19, 14]. We propose a practical approach that achieves fast convergence by finding approximate solutions which yield a low angular distortion.

3 Conventions

Throughout the paper, we try to restrict ourselves to the essential amount of formalism only, where the following notations are used:

- N is the total number of interior mesh angles.
- α_i^* $(i = 1, \ldots, N)$ denote the angles of the *original* mesh,
- α_i are the corresponding angles of the *flat* mesh. As these are the variables of the optimization problem, then in this context, the more usual notation x_i is used as an alternative.
- v denotes the central vertex in a centered drawing of a *wheel*, i.e. of its 1-neighborhood. d is the number of direct neighbors of v or its *valence*. α_j $(j = 1, \ldots, d)$ refer to the angles at v, while β_j and γ_j denote the opposite left and right angles of a face with central angle a_j, respectively. All faces are oriented counter-clockwise.
- Variables and functions without subscripts may refer to multivariate vectors as explained by the context.

4 Characterization of drawings of planar graphs

Sheffer and de Sturler [18] addressed the problem of the validity of the planar embedding by requiring the following consistency condition on the set of positive angles of the planar mesh:

- *Vertex consistency*
  For each internal vertex v, with central angles $\alpha_1, \ldots, \alpha_d$:

$$\sum_{i=1}^{d} \alpha_i - 2\pi = 0 \tag{1}$$

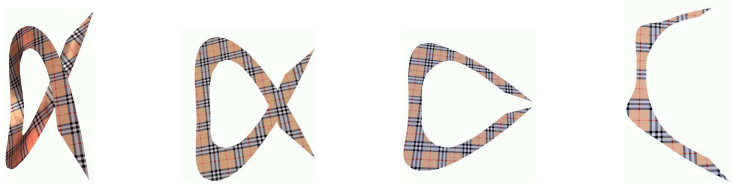
- *Triangle consistency*
  For each triangular face with angles α, β, γ the face consistency:

$$\alpha + \beta + \gamma - \pi = 0 \tag{2}$$

- *Wheel consistency*
  For each internal vertex v with left angles β_1,..,β_d and right angles $\gamma_1, \ldots, \gamma_d$:

$$\prod_{i=1}^{d} \frac{\sin(\beta_i)}{\sin(\gamma_i)} = 1 \tag{3}$$

  These conditions guarantee the centered embedding of internal vertices without overlapping of interior edges. However they do not prevent the overlapping of boundary edges. This issue is a well-studied problem in graph theory [4, 9]. Di Battista and Vismara provide a characterization



(a) (b) (c) (d)

Fig. 1. Flattening an α-shaped mode: (a) Original mesh. (b) The flattened mesh with boundary control coefficient $t = 2$). (c) $t = 1.1$.(d) $t = 1.03$. (The views are scaled differently.)

of the convex planar straight line drawing of a tri-connected graph for a given set of positive angles [4]. Their minimal constraints for the planarity of the graph impose in addition to (1),(2),(3) the following condition:

- *Convex external face condition*
  For each external vertex v, with internal angles $\alpha_1, \ldots, \alpha_d$:

$$\sum_{i=1}^{d} \alpha_i \leq \pi \tag{4}$$

  Condition (4) guarantees the *convexity of the boundary* and hence prevents boundary overlapping. Note that the inequality (4) prevents local and global self-intersection simultaneously. So it does not only prevents adjacent boundary edges from overlapping, but it also guarantees that the boundary loop as a whole does not cross itself. For the local configuration it would in fact be sufficient to require the following weakened condition to hold:
- *Adjacent boundary edges consistency*
  For each external vertex v, with internal angles $\alpha_1, \ldots, \alpha_d$:

$$\sum_{i=1}^{d} \alpha_i \leq 2\pi. \tag{5}$$

This prevents adjacent boundary triangles from crossing each other. But, condition (5) is not strong enough to globally enforce a valid mesh with no boundary intersections as shown in Fig. 1.b.

To get a better understanding and better control of the boundary behavior, we propose to multiply the left hand side by in (4) by a positive scalar t, formally

$$\sum_{i=1}^{d} \alpha_i \leq t\pi. \tag{6}$$

The scalar t can be interpreted as *boundary control coefficient* that steers the convexity of the boundary. A lower bound for this factor can be derived using discrete curvature measure. Consider the angular defect of the flat mesh can be expressed as:

$$\sum_{v=1}^{n} (\pi - A_v) = 2\pi, \tag{7}$$

where A_v is the sum of angles at vertex v and n is the number of boundary vertices. By a simple calculation, we establish the lower bound

$$t_0 = 1 - 2/n.$$

The trivial case is a single triangle, its angles cannot be all smaller than $\pi/3$.

We experimented with different values for t, and summarize the following interpretations that can be used as reference for choosing appropriate values for t:

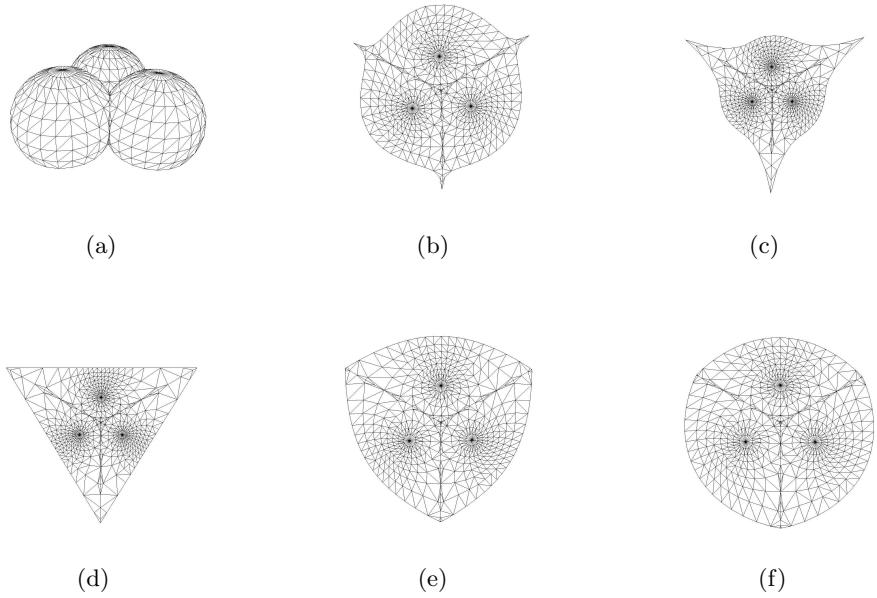


Fig. 2. Effect of the boundary control coefficient t on the *3-balls* model. (a) Original mesh. (b) Flat mesh for $t \geq 2$ (ABF). (c) $t = 1.05$. (d) $t = 1$ (convex boundary ABF). (e) $t = 0.98$. (f) $t = 0.968$

- $t > 2$ results in the classic ABF method without preconditioning. No adjacent edge overlapping or boundary self crossing is taken into consideration.

- $1 < t \leq 2$ prevents adjacent edges from overlapping, but does not necessarily prevent global self-intersections of the boundary loop. We experienced such cases only for "boundary-heavy" (w.r.t. ratio of boundary to inner vertices, e.g. Fig. 1) surfaces with non-trivial geometry.
- $t = 1$ globally prevents the boundary loop from self-intersection for any valid input mesh, note that this suffices to induce a convex boundary .
- $t_0 < t < 1$ forces the boundary to become concave.

Figures 1 and 2 illustrate the behavior of the boundary for different values of t. We can take advantage of these facts in order to avoid an iterative post-processing and thus have better control over the convergence of the constrained optimization problem. In the next sections we show how this problem with the additional inequalities included can be solved efficiently.

5 Constrained optimization problem

A general approach to establish a surface parameterization consists of minimizing an objective function $f(x)$ that quantifies distortion w.r.t. certain quality criteria. As the validity of the flat mesh is guaranteed by the angle constraints of section 4. A typical choice of such function consists of establishing an angle based objective function. Examples of such function are the angular distortion [18]

$$f(x) = \sum_{i=1}^{N} w_i (x_i - a_i)^2$$

with the weights $w_i = \frac{1}{a_i^2}$. The variables a_i represent the *optimal angles* of the flat mesh, which are

$$a_i = \begin{cases} \alpha_i^* \frac{2\pi}{\sum_{i=1}^{d} \alpha_i^*} & \text{around an interior vertex} \\ \alpha_i^* & \text{around a boundary vertex} \end{cases}$$

or the MIPS energy function, that can be written completely in terms of angles [7]

$$f(x) = \sum_{i=1}^{\#triangles} \cot \alpha_i \cot \alpha_i^*$$

We can now formulate the optimization problem as

$$\begin{array}{lrcl} \text{minimize} & f(x) & & \\ \text{subject to} & h(x) & = & 0 \\ & g(x) & \leq & 0, \end{array} \tag{8}$$

where g and h are multivariate functions of the equality (1),(2),(10) and the inequality constraints (6) respectively.

6 Solving the optimization problem

Large constrained optimization systems of the form (8) are still open problems in the field of non-linear optimization [2]. The adequacy of a minimization method depends on the properties of the objective function as well as on the constraints.

In order to solve the optimization problem we use the method of Lagrange multipliers as it guarantees the exact satisfaction of constraints. We handle the inequality constraints by means of the so called *active set* approach, a variant of Newton-like methods. It transforms inequalities to equalities which are generally easier to handle.

The active set is defined as the set of indices for which the inequality constraint (4) is active. Formally

$$A(x, \mu) = \{i | g_i \geq -\frac{\mu_i}{c}, i = 1, ..., r\}$$

where μ_i is the Lagrange multiplier associated with g_i, and c is a fixed positive scalar.

The active set approach converts inequality constraints to equality constraints by altering the Lagrange multipliers associated with them. If a constraint does not figure in the active set, its associated multipliers are set to zero. Otherwise it is treated as an equality constraint. The numerical advantage of this method is that as the iterates get closer to the solution, the active set becomes more and more stable. A detailed description of the active set method can be found in [1].

In every Newton iteration the following system is solved

$$\begin{bmatrix} \nabla^2_{xx} L & J_h^T & J_g^T \\ J_h & 0 & 0 \\ J_g & 0 & 0 \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta \mu_h \\ \Delta \mu_g \end{bmatrix} = - \begin{bmatrix} \nabla_x L \\ h \\ g \end{bmatrix} \tag{9}$$

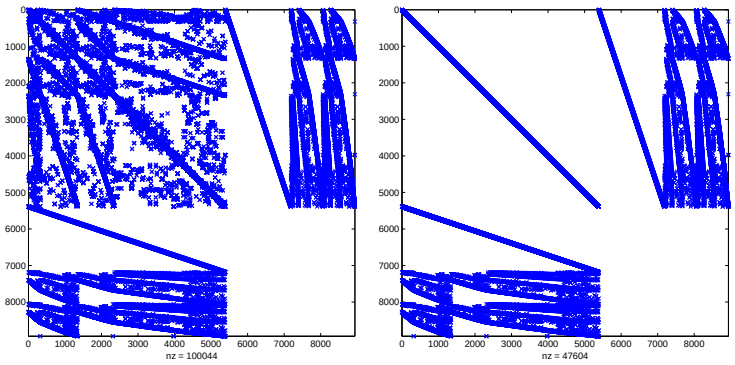
where the Lagrangian L is given by

$$L = f(x) + \mu_h^T h(x) + \mu_g^T g(x).$$

In the classic ABF algorithm, the computation of the Hessian matrix $\nabla^2_{xx} L$ involves finding the second derivatives of the products involved in condition (3). The resulting matrix is sparse, but it still contains a considerable number of non-zero elements (cf. Fig. 3(a)). This number depends largely on the valences of the input mesh vertices.

Instead, we propose to use a *modified wheel condition* (10). Since the angles are strictly positive we can safely rewrite condition (3) as

$$\sum_{i=1}^{d} \log(\sin \beta_i) - \log(\sin \gamma_i) = 0. \tag{10}$$



(a) (b)

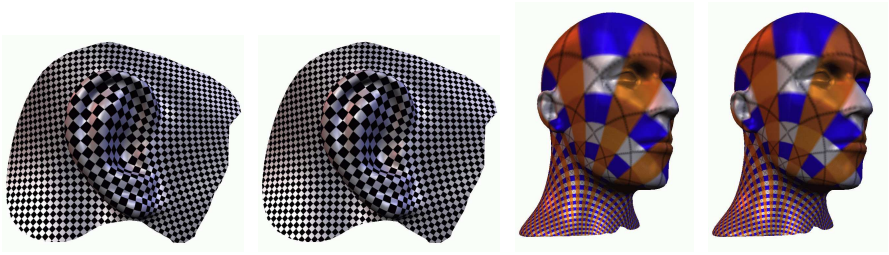
Fig. 3. System matrices of equation (9) generated from the *ear* model using the (a) original wheel condition. (b) simplified wheel condition. The Diagonal Hessian brought the number of nonzero elements from 100044 down to 47607.

The virtue of this modification resides in the fact that it yields a diagonal Hessian matrix.

$$\nabla^2_{xx} L = \mathrm{diag}(f''(x_i) + m_i \frac{-1}{\sin^2(x_i)})$$

where m_i is the linear combination of the Lagrange multipliers involved with x_i in condition (10). The amount of computation and effort by the iterative solvers is hence reduced considerably. Fig. 3 illustrates the structure of a typical system matrix and the improvement induced by the *modified wheel condition*.

The system matrix is symmetric although not necessarily positive-definite, with the additional advantage of having a diagonal Hessian. We can exploit this structure by using adequate iterative solvers such as MINRES or SYMMLQ both developed by Paige and Saunders [15] for symmetric matrices, instead of the non-symmetric GMRES and BiCGStab that were used in [18, 14]. The latter solvers have higher cost per iteration and may suffer form breakdowns or simply stagnate while MINRES and SYMMLQ have a relatively cheap cost per iteration, which is just 4 axpys(The term axpy denotes the addition of a scalar multiple of a vector to a vector i.e vector1+=scalar*vector2) higher than the iteration cost of the conjugate gradient method. Another alternative, which is relatively inexpensive, is the CGNR algorithm introduced by Hestens and Steifel [8]. In our case, the benefit is that there is no need to transpose the system matrix as it is symmetric. The cost per iteration then, is just one matrix-vector multiplication higher than the cost of the conjugate gradient method. A comparison of the convergence of these iterative solvers for typical meshes is given in section 8.



(a) (b) (c) (d)

Fig. 4. Comparison of underdetermined (a, c) and minimization (b, d) solutions.

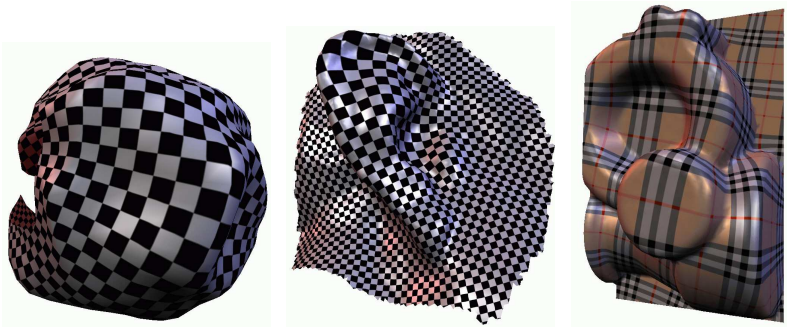
Note that the initial guess for the x is the set of the optimal angles $\{a_i | i = 1..n\}$. Consequently, at every Newton iteration the solution stays within the positive domain. In order to guarantee that our algorithm does not step into the negative domain, we can apply a similar technique as in [18] that rejects negative iterates and appends increased weights to the corresponding angles. However, our experiments with different meshes show that we hardly ever run into this situation.

7 Practical Approach

In general, the method of Lagrange multipliers we use is a local optimization method. This means that the solution it provides is a local minimum that is largely dependent on the initial guess provided by the user. In other words it is the closest minimum to the initial guess. Since the initial guess we provide is very close to the solution as only few newton iterations are needed for convergence. We can assume that any feasible point that is close to the initial guess mentioned above gives a good estimate for the solution and would yield a low angular distortion.

With this consideration in mind, The problem can be restated as how to get a feasible point. The idea is to have a null objective function, i.e set $f = 0$. This means that we reduce the problem to solving the under-determined system of equality and inequality constraints. This change leads to considerable speed up of convergence as there is no extra load form the objective function. In the following, we call this solution the underdetermined solution and the one using the angular distortion functional the minimization solution (literally speaking both solution are just approximate solutions).

In general, the difference between the minimization and the underdetermined solution is hardly noticeable. Fig. 4 shows a comparison between such solutions. Table 2 shows the numerical difference w.r.t. angular distortion be-



(a) (b) (c)

Fig. 5. Textured models: (a) Clumpy (c) Large ear(b) and (d) Mechanical part. Notice the quasi-conformality of the parameterization.

tween the two methods. The latter method seems to outperform the minimization method as it converges much faster. Table 1 summarizes the performance of both methods.

If we are looking only for a topological mapping of the mesh to the plane, we can get a very fast feasible solution by setting the initial guess to zero and the objective function to zero. This solution does not reflect the geometry of the mesh and might not suited for texture mapping.

Another alternative method for finding a feasible point would be to use least squares methods for solving underdetermined nonlinear problems. However, as these methods do not guarantee the exact satisfaction of the constraints for large problems, they failed in general to produce valid parameterizations.

8 Results and Discussion

We applied our algorithm to a set of different triangular meshes. Table 1 summarizes the numerical results of our method, all timings were measured on a 1.7 GHz Intel Xeon CPU. The parameterization time depends on the number of triangles, on the geometry as well as on the connectivity of the input mesh. For consistency with the original ABF we use the same metrics as in [19] to measure angular distortion. The numerical data suggests that the underdetermined method in association with the CGNR algorithm delivers high quality quasi conformal parameterizations in very competitive time.

model	CGNR		SYMMLQ		MINRES	
	minim.	*under.*	*minim.*	*under.*	*minim.*	*under.*
3 Balls (1032△)	66	1	21	2	7	2
Ear (1796△)	159	3	44	9	11	7
Man head (5420△)	> 900	26	292	69	126	56
Mech. part (7938△)	> 999	86	> 999	245	> 999	192
Large ear (24914△)	> 999	237	> 999	654	> 999	522

Table 1. Comparison of runtime (in seconds) of the minimization and underdetermined method using different iterative solvers.

model	CGNR		SYMMLQ		MINRES	
	minim.	*under.*	*minim.*	*under.*	*minim.*	*under.*
3 balls	0.106	0.137	0.106	0.137	0.106	0.137
ear	0.001	0.001	0.001	0.001	0.001	0.001
man head	0.002	0.002	0.002	0.002	0.002	0.002
Mech. part	0.002	0.002	0.002	0.002	0.002	0.002
Large ear	0.0001	0.0001	0.0001	0.0001	0.0001	0.0001

Table 2. Comparison of angular distortion induced by the minimization and the underdetermined method using different iterative solvers.

9 Conclusion

We presented and discussed several extensions to angle based flattening. With additional inequality constraints we can eliminate global and/or local boundary self-intersections. This leads to a nice interpretation of boundary behavior through the introduction of the new boundary control coefficient. While its use initially targets on the avoidance of an iterative post-processing, we see potential use for optimizing the parameterization w.r.t. this coefficient variable.

The arising non-linear constrained optimization problem can be solved efficiently. With a simple and intuitive transformation we take advantage of a simply structured symmetric system matrix, enabling the application of robust iterative solvers. The use of our underdetermined system solution leads to a relatively fast method for generating angle based parameterizations.

References

1. D.P. Bertsekas. *Constrained Optimization and Lagrange Multiplier Methods*. Athena Scientific, 1996.
2. R.H. Byrd, J. Nocedal Active Set and Interior Methods for Nonlinear Optimization *Doc. MATH*, Extra Volume ICM III, 1998, pp. 667–676
3. Mathieu Desbrun, Mark Meyer, and Pierre Alliez. Intrinsic Parameterizations of Triangle Meshes. *Proc. Eurographics 2002*, pp. 209-218

4. G. Di Battista and L. Vismara. Angles of Planar Triangular Graphs. *SIAM Journal on Discrete Mathematics*, 9 (3), 1996, pp. 349–359
5. M. Eck, T. DeRose, T. Duchamp, H. Hoppe, M. Lounsbery, and W. Stuetzle. Multiresolution Analysis of Arbitrary Meshes. *Proc. SIGGRAPH 95*, pp. 173–182
6. M.S. Floater. Parametrization and smooth approximation of surface triangulations. *Comp. Aided Geom. Design*, (14), 1997, pp. 231-250
7. M. S. Floater, K. Hormann. Parameterization of triangulations and unorganized points. *Tutorials on Multiresolution in Geometric Modelling* Springer-Verlag, Heidelberg (2002), 287-315.
8. M. R. Hestenes and E. Stiefel. Method of Conjugate Gradients for Solving Linear Systems. *J. Res. Nat. Bur. Stand.* 49:409–436, 1952.
9. A. Garg. New results on drawing angle graphs. *Computational Geometry* (9), (1-2), 1998, pp. 43–82
10. A. Greenbaum. *Iterative Methods for Solving Linear Systems* SIAM, Philadelphia, 1997
11. S. Haker, S. Angenent, A. Tannenbaum, R. Kikinis, G. Sapiro, and M. Halle. Conformal surface parameterization for texture mapping. *IEEE Transactions on Visualization and Computer Graphics*, 6(2), 2000, pp. 181–189
12. K. Hormann, G. Greiner. MIPS: an efficient global parametrization method. *Curve and Surface Design: Saint-Malo 1999*, 2000, pp. 153–162
13. Bruno Levy, Sylvain Petitjean, Nicolas Ray, and Jerome Maillot. Least Squares Conformal Maps for Automatic Texture Atlas Generation. *Proc. SIGGRAPH 2002*, pp. 362–371
14. J. Liesen, E. de Sturler, A. Sheffer, Y. Aydin, and C. Siefert. Preconditioners for Indefinite Linear Systems Arising in Surface Parameterization. *Proceedings of the 10th International Meshing Round Table*, 2001, pp. 71–81
15. C. Paige and M. Saunders. Solution of sparse indefinite systems of linear equations. *SIAM J. Numer. Anal*, 12, 1975, pp. 617–629
16. U. Pinkall and K. Polthier. Computing discrete minimal surfaces and their conjugates. *Experimental Mathematics*, 2(15), 1993, pp. 15-36
17. P. Sander, J. Snyder, S. Gortler, and H. Hoppe. Texture mapping progressive meshes. *Proc. SIGGRAPH 2001*, pp. 409–416
18. A. Sheffer and E. de Sturler. Parameterization of Faceted Surfaces for Meshing Using Angle Based Flattening. *Engineering with Computers*, 17 (3), 2001, pp. 326–337
19. A. Sheffer, E. de Sturler. Smoothing an Overlay Grid to Minimize Linear Distortion in Texture Mapping. *ACM Transactions on Graphics*, 21 (4), 2002
20. O. Sorkine, D. Cohen-Or, R. Goldenthal, D. Lischinski. Bounded-distortion piecewise mesh parameterization. *Proc. IEEE Visualization 2002*, pp. 355–362
21. W. T. Tutte. How to draw a graph. *Proceedings of the London Mathematical Society*, 13 (3), 1963, pp. 743–768
22. G. Zigelmann, R. Kimmel, and N. Kiryati. Texture mapping using surface flattening via multi-dimensional scaling. *IEEE Transactions on Visualization and Computer Graphics*, 8(2), 2002