A SIMPLE AND EFFICIENT METHOD FOR MODELING CONSTANT MEAN CURVATURE SURFACES

ROBERT J. RENKA*

Abstract. A constant mean curvature (CMC) surface is a critical point of surface area with respect to variations that preserve the volume bounded by the surface. We present a simple and elegant method for constructing a triangle mesh approximation to a CMC surface by minimizing the discretized surface area subject to a fixed volume and other constraints such as fixed boundary curves. The key idea is an optimization procedure in which the surface area functional is replaced by a least squares energy functional that is easier to minimize and whose critical points are uniformly parameterized surfaces or, more generally, surfaces parameterized in accord with an arbitrary density function. A good initial solution estimate is not required. In fact, the input mesh quality may be almost arbitrarily poor. The user-specified volume is taken to be a shape-control parameter, and since the volume constraint need not be satisfied precisely, it is simply treated as an additional equation in the least squares system. The nonlinear least squares problem is treated by a trust region method. Test results demonstrate the effectiveness of the procedure.

Key words. constant mean curvature, minimal surface, surface modeling, triangle mesh, trust region, uniform parameterization

AMS 49M15, 49M20, 49M25, 65N30

1. Introduction. A solution to the problem of minimizing surface area subject to a prespecified volume and other constraints, such as one or more fixed closed boundary curves, is a surface with constant mean curvature (CMC). Without the volume constraint the mean curvature is zero at every point, and the result is termed a minimal surface. Physical examples include soap bubbles and soap films. Such surfaces have been of interest to mathematicians since the work of Lagrange in 1762, and have more recently found application in architecture and shape modeling [2].

The most common means of modeling CMC surfaces is to discretize the surface with a triangle mesh and solve a constrained optimization problem for the free vertices. This is the method used by [7], [1], [13], [14], [8], [3], [5], and [12]. A major issue with this approach is maintaining good mesh quality characterized by uniformly distributed vertices and good triangle shapes. Poorly shaped triangles reduce the accuracy of estimated surface properties such as curvature, make the mesh unsuitable for simulation, and can cause the optimization method to fail with a nearly degenerate triangle. Most of the methods address this problem by allowing the user to alternate iterations of surface area minimization with mesh improvement techniques such as Laplacian smoothing. The method of [5] circumvents the problem by minimizing Willmore energy using a curvature-based representation of geometry (integrated twice to recover positions), and employing conformal transformations that preserve triangle shape.

The approach taken by [12] is to replace the surface area functional by an energy functional associated with an extended centroidal Voronoi tessellation. This enables the optimization of both surface area and mesh quality in a unified manner. We take a similar approach but use a simpler energy functional which is cheaper to minimize and requires no mesh updates. We extend the squared area functional employed in [15] and [18] to include a density function and an additional term associated with the volume constraint. The result is an elegant theory and a powerful method that is efficient, easy to implement, and produces a high-quality variable-density mesh

*University of North Texas, Denton TX 76203-1366. (email: robert.renka@unt.edu)
without requiring a good initial mesh.

We formulate the continuous problem in section 2, describe the discretized problem and the solution method in section 3, and present test results in section 4. Section 5 concludes the paper.

2. Function Space Formulation. To understand the numerical method for constructing triangle mesh CMC surfaces (the discretized optimization problem) it is helpful to consider the function space setting first. We define the energy functionals and their derivatives in the first and second subsection, respectively, we show that a critical point of the least squares objective function is a uniformly parameterized CMC surface in the third subsection, and we discuss the analogous problem of constructing a constant curvature curve in the last subsection. The central idea in our method is quite transparent in the curve case.

2.1. Energy functionals. We will define four energy functionals: surface area $\psi(f)$, signed volume $v(f)$, density-weighted squared surface area $\alpha(f)$, and the least squares objective function $\phi(f)$. In order to simplify the exposition we will use a simply-connected parameter space, but this is easily generalized. Denote the unit square by $\Omega = [0,1]^2$, and define parametric representations of regular surfaces by

$$S = \{ f \in C^2(\Omega, \mathbb{R}^3) : f_1 \times f_2 \neq 0 \},$$

where $f_1$ and $f_2$ denote first partial derivatives. A parametric minimal surface in $\mathbb{R}^3$ is the image of a critical point of surface area with respect to variations that fix the boundary. The surface area functional $\psi : S \rightarrow \mathbb{R}$ is given by

$$\psi(f) = \int_{\Omega} ||f_1 \times f_2||,$$

where $||\cdot||$ denotes Euclidean norm in $\mathbb{R}^3$. The signed volume bounded by the surface and the cone connecting the boundary with the origin is

$$v(f) = \frac{1}{3} \int_{\Omega} \langle f, f_1 \times f_2 \rangle$$

with the angular brackets denoting inner product in $\mathbb{R}^3$. For a constant prespecified signed volume $c$, a CMC surface is a critical point of $\psi$ constrained by $v(f) = c$ or, equivalently, a critical point of $\psi + \lambda(v - c)$ for Lagrange multiplier $\lambda$ which corresponds to the (constant) mean curvature of the surface as shown by expressions for the gradients of $\psi$ and $v$ in Eq. (2.7) and Eq. (2.8).

The problem with minimizing the surface area functional directly is a nonuniformly parameterized surface characterized by widely varying distances between surface points corresponding to points that differ by a fixed distance in the parameter space. For a triangle mesh discretization this manifests itself as poorly shaped triangles. In our treatment of the minimal surface problem by minimizing a discretization of $\psi$ ([15]) we found that convergence was slowed by the freedom to change the parameterization without changing the surface area. Rather than increasing the complexity of the problem by introducing additional constraints, we found that we could remove the extraneous freedom and actually reduce the complexity by simply replacing the surface area functional with the related functional

$$\int_{\Omega} ||f_1 \times f_2||^2.$$
We showed that this functional has the same critical points as $\psi$, and they are uniformly parameterized with $\|f_1 \times f_2\|$ constant.

In order to allow variable vertex density in the discretization we define an inner product on $S$ by

$$\langle g, h \rangle_\rho = \int_{\Omega} \rho \langle g, h \rangle$$

for positive density function $\rho \in C^1(\Omega, \mathbb{R})$, and we generalize the functional to

\begin{equation}
\alpha(f) = \frac{1}{2} \|f_1 \times f_2\|^2 = \frac{1}{2} \int_{\Omega} \rho \|f_1 \times f_2\|^2.
\end{equation}

Using $\alpha$ as an alternative to the surface area functional not only eliminates the problem of poorly parameterized surfaces, but also has the numerical advantages of a least squares functional, including reliable and efficient Gauss-Newton and Levenberg-Marquardt optimization methods, and the flexibility of treating nonlinear equality constraints by simply adding them to the least squares system. We take advantage of this in defining the following least squares energy functional:

\begin{equation}
\phi(f) = \alpha(f) + \frac{w}{2} (v(f) - c)^2
\end{equation}

for nonnegative weight $w$ and target volume $c$ which serves as a shape control parameter. Note that, in order to accurately satisfy the volume constraint in the discretized problem, we could use a large value of $w$ treated as a penalty parameter. This would require that we solve a sequence of problems with increasing penalty weights because, as shown in section 3.2, Eq. (3.12), the penalty term adds a rank-1 matrix to the Hessian of $\phi$, thus increasing the condition number and decreasing the efficiency of an iterative linear solver. Since $c$ is merely a shape control parameter, however, it is not necessary to satisfy the volume constraint accurately, and we use $w = 10^4$ for computing CMC surfaces. The computed volumes of the CMC surfaces in Figure 1 differed from their target values by less than 0.4%.

2.2. Gradients and Hessians. An efficient and robust method for computing critical points of $\phi$ uses a Newton-like iteration, along with a globalization strategy based on a line search or trust region, and either a preconditioned conjugate gradient method or an efficient direct solver that takes advantage of sparsity for solving the symmetric positive definite linear system at each outer iteration. Since the conjugate gradient method requires only matrix-vector products, the Hessians or Hessian approximations need not be computed and stored; rather it is sufficient to compute a product of a Hessian with a gradient vector. We derive an expression for the latter in both the continuous and the discrete case. Having both serves as a check on the algebra: a discretization of the derivatives of the continuous functional should agree with the derivatives of the discretized functional.

We define the linear space of compactly supported variations as

$$S_0 = \{ h \in C^2(\Omega, \mathbb{R}^3) : h\vert_{\partial\Omega} = 0 \},$$

where $\partial\Omega$ is the boundary of $\Omega$ on which function values are to be fixed. In order to simplify the derivation of expressions for gradients and Hessians of $\phi$, define nonlinear differential operators $A$ from $S$ to $C^1(\Omega, \mathbb{R}^3)$ and $V$ from $S$ to $C^1(\Omega, \mathbb{R})$ by

$$A(f) = f_1 \times f_2$$

for any $f = (f_1, f_2) \in S$. The operator $V$ yields the volume functional.

MODELING CONSTANT MEAN CURVATURE SURFACES

3
Fig. 1. Discrete CMC surfaces with the same boundary but increasing volume: 0.5, 2.4, 4.0, 8.0, and 15.0. The second image (volume 2.4) is a minimal surface. Colors correspond to mean curvature.

and

\[ V(f) = \langle f, A(f) \rangle = \det(f, f_1, f_2), \]

where \( \det(f, g, h) \) denotes the determinant of the order-3 matrix with rows (or columns) \( f, g, \) and \( h \). We then have

\[ \alpha(f) = \frac{1}{2} \langle A(f), A(f) \rangle, \]

and

\[ v(f) = \frac{1}{3} \int_{\Omega} V(f). \]

The Fréchet derivatives are

\[ A'(f)h = f_1 \times h_2 + h_1 \times f_2, \]

\[ A''(f)kh = A'(k)h = k_1 \times h_2 + h_1 \times k_2, \]
\[ V'(f)h = \langle f, A'(f)h \rangle + \langle h, A(f) \rangle \]
\[ = \det(f, f_1, h_2) + \det(f, h_1, f_2) + \det(h, f_1, f_2), \]

and
\[ V''(f)kh = \langle f, A''(f)kh \rangle + \langle k, A'(f)h \rangle + \langle h, A'(f)k \rangle \]
\[ = \det(f, k_1, h_2) + \det(f, h_1, k_2) + \det(k, f_1, h_2) \]
\[ + \det(k, h_1, f_2) + \det(h, f_1, k_2) + \det(h, k_1, f_2) \]

for \( h, k \in S_0 \).

The adjoint of \( A'(f) \) in the Hilbert space consisting of \( S_0 \) with the \( \rho \) inner product is denoted \( A'(f)^* \) and defined by
\[ \langle h, A'(f)^*k \rangle \rho = \langle A'(f)h, k \rangle \rho = \int_{\Omega} \rho(f_1 \times h_2 + h_1 \times f_2, k) \]
\[ = \int_{\Omega} \langle \rho k \times f_1, h_2 \rangle + \langle h_1, f_2 \times \rho k \rangle \]
\[ = \int_{\Omega} \langle D_2(\rho k \times f_1), h \rangle + \langle h, D_1(f_2 \times \rho k) \rangle \]
\[ = \int_{\Omega} \langle h, D_2(\rho k) \times f_1 + \rho k \times f_1 + f_2 \times \rho k + f_2 \times D_1(\rho k) \rangle \]
\[ = \int_{\Omega} \langle h, f_1 \times D_2(\rho k) + D_1(\rho k) \times f_2 \rangle, \]

for all \( h, k \in S_0 \), where \( D_1 \) and \( D_2 \) denote first partial derivative operators, and \( f_{12} = f_{21} \) for \( f \in C^2(\Omega, \mathbb{R}^3) \). We thus obtain
\[ A'(f)^*k = \frac{1}{\rho} A'(f)(\rho k). \]

Although \( A(f) \) is not an element of \( S_0 \), we can formally apply \( A'(f)^* \) to \( A(f) \) to obtain expressions for Fréchet derivatives as follows:
\[ \alpha'(f)h = \langle A'(f)h, A(f) \rangle \rho = \langle h, A'(f)^*A(f) \rangle \rho \]
\[ = \langle h, \frac{1}{\rho} A'(f)[\rho(f_1 \times f_2)] \rangle \rho \]
\[ = \langle h, f_1 \times D_2[\rho(f_1 \times f_2)] + D_1[\rho(f_1 \times f_2)] \times f_2 \rangle_{L^2}, \]

where \( \langle \cdot, \cdot \rangle_{L^2} \) denotes the standard inner product associated with \( L^2(\Omega, \mathbb{R}^3) \). The \( L^2 \) gradient of \( \alpha \) is defined by \( \alpha'(f)h = \langle h, \nabla \alpha(f) \rangle \rangle_{L^2} \) so that
\[ \nabla \alpha(f) = f_1 \times D_2[\rho(f_1 \times f_2)] + D_1[\rho(f_1 \times f_2)] \times f_2 \]
\[ = -D_1[\rho(f_2 \times f_1 \times f_2)] - D_2[\rho(f_1 \times f_2 \times f_1)]. \]

The second derivative is
\[ \alpha''(f)kh = \langle A'(f)h, A'(f)k \rangle \rho + \langle A'(k)h, A(f) \rangle \rho \]
\[ = \langle h, A'(f)^*A'(f)k \rangle \rho + \langle h, A'(k)^*A(f) \rangle \rho, \]
and the Hessian of $\alpha$ at $f$ applied to $k \in S_0$ is

$$\nabla^2 \alpha(f)k = \rho(A'(f)^* A'(f)k + A'(k)^* A(f))$$

(2.6)

$$= f_1 \times D_2 [\rho(f_1 \times k_2 + k_1 \times f_2)] + D_1 [\rho(f_1 \times k_2 + k_1 \times f_2)] \times f_2 + k_1 \times D_2 [\rho(f_1 \times f_2)] + D_1 [\rho(f_1 \times f_2)] \times k_2.$$  

The Gauss-Newton approximation to $\nabla^2 \alpha(f)$ is $\rho A'(f)^* A'(f)$.

For comparison, the surface area functional has gradient

(2.7) $\nabla \psi(f) = f_1 \times D_2 (N(f)) + D_1 (N(f)) \times f_2$

$$= -D_1 [(f_2 \times f_1 + f_2)/\|f_1 \times f_2\|] - D_2 [(f_1 \times f_2 \times f_1)/\|f_1 \times f_2\|]$$

$$= -2H(f_1 \times f_2)$$

for unit normal $N(f) = A(f)/\|A(f)\|$ and curvature

$$H = \frac{(f_1 \times f_2, f_2 \times D_1 (N(f)) + D_2 (N(f)) \times f_1)}{2\|f_1 \times f_2\|^2}$$

so that a critical point of surface area is characterized by zero curvature. Note also that unlike $\nabla \alpha$, $\nabla \psi$ has no tangential component and, consequently, a gradient descent method using $\nabla \psi$ does not allow a change in the parameterization.

We use integration by parts to obtain

$$v'(f) h = \frac{1}{3} \int_{\Omega} V'(f) h = \frac{1}{3} \int_{\Omega} \langle f, f_1 \times h_2 \rangle + \langle f, h_1 \times f_2 \rangle + \langle h, f_1 \times f_2 \rangle$$

$$= \frac{1}{3} \int_{\Omega} \langle h, f_1 \times f_2 - D_1 (f_2 \times f) - D_2 (f \times f_1) \rangle$$

$$= \frac{1}{3} \int_{\Omega} \langle h, 3(f_1 \times f_2) \rangle = \langle h, A(f) \rangle_{L^2}$$

and

$$v''(f) k h = \langle h, A'(f) k \rangle_{L^2}$$

so that the gradient of $v$ is

$$\nabla v(f) = f_1 \times f_2,$$

and the Hessian is defined by

$$\nabla^2 v(f) k = f_1 \times k_2 + k_1 \times f_2.$$  

(2.8)

The gradient of $\phi$ is

$$\nabla \phi(f) = \nabla \alpha(f) + w(v(f) - c)(f_1 \times f_2),$$

(2.10)

and the Hessian is defined by

$$\nabla^2 \phi(f) k = \nabla^2 \alpha(f) k + w(v(f) - c) \nabla^2 v(f) k + w \nabla v(f)v'(f) k.$$  

(2.11)

The second term is omitted in the Gauss-Newton approximation.
2.3. CMC surface. Theorem 2.1. A critical point of \( \phi \) is a uniformly parameterized CMC surface.

Proof. At a critical point \( f \) where \( \nabla \phi(f) = 0 \), we have \( \nabla \alpha(f) = -w(v(f) - c)(f_1 \times f_2) \) in the normal direction orthogonal to \( f_1 \) and \( f_2 \) so that \( \langle f_1, \nabla \alpha(f) \rangle = 0 \) and \( \langle f_2, \nabla \alpha(f) \rangle = 0 \). Also, from (2.5) we have

\[
D_1(\rho\|f_1 \times f_2\|) = \frac{\langle f_1 \times f_2, D_1[\rho(f_1 \times f_2)] \rangle}{\|f_1 \times f_2\|} = -\frac{\langle f_1, \nabla \alpha(f) \rangle}{\|f_1 \times f_2\|} = 0
\]

and

\[
D_2(\rho\|f_1 \times f_2\|) = \frac{\langle f_1 \times f_2, D_2[\rho(f_1 \times f_2)] \rangle}{\|f_1 \times f_2\|} = -\frac{\langle f_2, \nabla \alpha(f) \rangle}{\|f_1 \times f_2\|} = 0.
\]

Hence \( \rho\|f_1 \times f_2\| \) is constant. From (2.8) we then have

\[
\nabla \alpha(f) = -D_1[\rho(f_2 \times f_1 \times f_2)] - D_2[\rho(f_1 \times f_2 \times f_1)] \\
= \rho\|f_1 \times f_2\|\nabla \psi(f) \\
= -2H \rho\|f_1 \times f_2\|(f_1 \times f_2) \\
= -w(v(f) - c)(f_1 \times f_2)
\]

implying that the curvature \( H = w(v(f) - c)/(2\rho\|f_1 \times f_2\|) \) is constant at the critical point. \( \Box \)

Corollary 2.2. At a critical point \( f \) of \( \phi \) or \( \alpha \), \( \alpha(f) \) is proportional to the squared surface area.

Proof. By the Cauchy-Schwarz inequality

\[
\psi(f) = \langle \|f_1 \times f_2\|, 1/\rho \rangle \leq \left( \int_\Omega \rho\|f_1 \times f_2\|^2 \right)^{\frac{1}{2}} \left( \int_\Omega \frac{1}{\rho} \right)^{\frac{1}{2}}
\]

so that, squaring both sides, \( 2\alpha(\int_\Omega 1/\rho) \geq \psi(f)^2 \) with equality when functions \( \|f_1 \times f_2\| \) and \( 1/\rho \) are linearly dependent; i.e., when \( \rho\|f_1 \times f_2\| \) is constant. \( \Box \)

2.4. Constant curvature curve. It is instructive to consider the analogous problem of constructing a \( C^2 \) constant curvature planar curve. We parameterize the curve by time \( t \in [0, 1] \) and have the energy functional

\[
\phi(f) = \frac{1}{2} \int_0^1 \rho\|f'\|^2 + \frac{w}{2}(v(f) - c)^2
\]

for signed area

\[
v(f) = \frac{1}{2} \int_0^1 \langle f, Rf' \rangle,
\]

where \( R \) is a 90-degree clockwise rotation operator. The gradient is

\[
\nabla \phi(f) = -(\rho f')' + w(v(f) - c)Rf'.
\]

A straightforward exercise then shows that a critical point of \( \phi \) has acceleration \( (\rho f')' \) in the normal direction \( Rf' \), constant speed \( \rho\|f'\| \), and constant curvature \( \kappa = w(v(f) - c)/(\rho\|f'\|) \).
Consider a gradient descent method applied to the minimum curve length problem with $\rho = 1$ and $w = 0$. The acceleration direction $f''$ may have a nonzero tangential component which allows for reparameterization toward constant speed (uniform parameterization) in addition to evolution toward zero curvature. The gradient of curve length $\int_0^1 \| f' \|$, on the other hand, is proportional to the curvature and normal to the curve at every point. This is a poor search direction which, in a discretized problem, leads to failure to converge unless the initial estimate is close to the solution. These ideas are further discussed in [16].

3. Discretized Problem: Triangle Mesh Surface. We define the discretized energy functionals in the first subsection, describe their derivatives in the second subsection, and discuss the computational procedure in the third subsection.

3.1. Discrete energy functionals. In order to simplify the notation, we retain many of the symbols used in the previous section but with revised definitions. A triangle mesh surface $T$ may be represented by a set of vertex indices $I = \{1, 2, \ldots, n_v\}$, a mapping $f : I \rightarrow \mathbb{R}^3$ defining the vertex locations, and a set of cyclically ordered index triples $Q$ with $\tau = [a, b, c] \in Q$ corresponding to a triangle with vertices $f_a, f_b, f_c$. We require an initial triangle mesh surface $T$ which serves both as an initial solution estimate and to define the topology. We discuss edge swaps and mesh refinement in section 3.3, but in this section we assume that the connection topology is fixed and consider only the effect of changes in vertex locations on the energy functional.

The surface may be closed or it may include any number of boundary curves whose vertices are usually fixed. An arbitrary subset of the interior vertices may also be designated as fixed, thus allowing for peaks and creases in the surface. Each triangle is assigned a constant density value $\rho_\tau$ computed by averaging vertex values taken from a smooth function so that the mesh is smoothly graded. To the extent allowed by the fixed vertices, critical points will have a uniform distribution of density-weighted triangles areas so that high vertex density corresponds to small triangle area.

We define $A_\tau$ and $V_\tau$ by analogy with $A$ and $V$ in section 2.2. The normal to triangle $[f_a, f_b, f_c]$ is

$$A_\tau(f) = (f_b - f_a) \times (f_c - f_a)$$

for $\tau = [a, b, c]$. The signed volume of the tetrahedron with vertices $f_a, f_b, f_c, \text{ and the origin}$ is proportional to

$$V_\tau(f) = \text{det}(f_a, f_b, f_c).$$

Then the discrete surface area and volume functionals are

$$\psi(f) = \frac{1}{2} \sum_{\tau \in Q} \| A_\tau(f) \|,$$

and

$$v(f) = \frac{1}{6} \sum_{\tau \in Q} V_\tau(f).$$

The discretized least squares functionals are

$$\alpha(f) = \frac{n_t}{2} \sum_{\tau \in Q} \rho_\tau \| A_\tau(f) \|^2,$$
where \( n_t \) is the number of triangles in \( T \), and

\[
\phi(f) = \alpha(f) + \frac{w}{2} (v(f) - c)^2
\]

for nonnegative weight \( w \) and target volume \( c \). We compute discrete approximations to CMC surfaces as local minima of \( \phi \). The scaling of \( \alpha(f) \) by \( n_t \) is a normalization that makes \( \alpha(f) \) asymptotically proportional to the squared surface area, so that \( w \), which defines the relative weighting of the terms in \( \phi \), is independent of \( n_t \) and the level of refinement. To set up an analogy with the continuous problem, we define an inner product on \( \mathbb{R}^{n_t} \) by

\[
\langle a, b \rangle = \sum_{\tau \in Q} \rho_\tau a_\tau b_\tau
\]

and apply the Cauchy-Schwarz inequality to the vectors of triangle areas \( \|A_\tau(f)\| \) and weight reciprocals \( 1/\rho_\tau \), giving

\[
\sum_{\tau \in Q} \|A_\tau(f)\| \leq \left( \sum_{\tau \in Q} \rho_\tau \|A_\tau(f)\|^2 \right)^{\frac{1}{2}} \left( \sum_{\tau \in Q} \frac{1}{\rho_\tau} \right)^{\frac{1}{2}}
\]

so that, squaring both sides and scaling by \( n_t \),

\[
n_t (2\psi(f))^2 = n_t \left( \sum_{\tau \in Q} \|A_\tau(f)\| \right)^2 \leq n_t \left( \sum_{\tau \in Q} \rho_\tau \|A_\tau(f)\|^2 \right) \left( \sum_{\tau \in Q} \frac{1}{\rho_\tau} \right)
\]

with equality if and only if all of the density-weighted triangles areas are identical. For an arbitrary choice of fixed vertices, it may not be possible for the minimizer of \( \alpha \) to have a perfectly uniform distribution of weighted triangle area, but in the limit as \( n_t \to \infty \), \( \alpha(f) \) is proportional to the squared surface area (assuming \( n_t/\sum_{\tau \in Q}(1/\rho_\tau) \) is bounded above).

Note that if a free vertex and its neighbors are coplanar, perturbing the vertex in that plane has no effect on the surface area \( \psi \), and the solution to the problem of minimizing \( \psi \) is therefore not unique even with fixed connection topology. By replacing \( \psi \) by \( \alpha \) we have removed the extraneous freedom and used it to enforce uniform weighted triangle areas.

3.2. Gradients and Hessians of discrete functionals. Variations of \( f \) are taken from the linear space \( S_0 \) of functions \( h : I \to \mathbb{R}^3 \) such that \( h_p = 0 \) if \( p \) indexes a fixed vertex. The following expressions for gradients and Hessians are derived by analogy with the continuous case, but can be obtained more directly by differentiating the functionals. For \( \tau = [a, b, c] \), derivatives of \( A_\tau \) and \( V_\tau \) at \( f \) in the directions \( h, k \in S_0 \) are

\[
A'_\tau(f)h = (f_b - f_a) \times (h_c - h_a) + (h_b - h_a) \times (f_c - f_a),
\]

\[
A''_\tau(f)kh = A'_\tau(k)h = (k_b - k_a) \times (h_c - h_a) + (h_b - h_a) \times (k_c - k_a),
\]
\[ V'(f)h = \det(f_a, f_b, h_c) + \det(f_a, h_b, f_c) + \det(h_a, f_b, f_c), \]

and

\[ V''(f)kh = \det(f_a, k_b, h_c) + \det(k_a, f_b, h_c) + \det(f_a, h_b, k_c) + \det(k_a, h_b, f_c) + \]
\[ \det(h_a, f_b, k_c) + \det(h_a, k_b, f_c). \]

Now denote by \( T_p = \{ \tau \in Q : p \in \tau \} \) the index triples that include \( p \), corresponding to the patch of triangles that contain vertex \( f_p \).

**Gradients.** Euclidean gradients are computed from the following.

\[
\psi'(f)h = \frac{1}{2} \sum_{\tau \in Q} \frac{\langle A_\tau(f), A'_\tau(f)h \rangle}{\|A_\tau(f)\|} = \frac{1}{2} \sum_{\tau = [p,b,c] \in Q} \left[ \langle h_p, \frac{A_\tau(f)}{\|A_\tau(f)\|} \times (f_c - f_b) \rangle + \langle h_b, \frac{A_\tau(f)}{\|A_\tau(f)\|} \times (f_p - f_c) \rangle + \langle h_c, \frac{A_\tau(f)}{\|A_\tau(f)\|} \times (f_b - f_p) \rangle \right]
\]

\[
= \frac{1}{2} \sum_{p \in I} \left( h_p, \sum_{\tau = [p,b,c] \in T_p} \frac{A_\tau(f)}{\|A_\tau(f)\|} \times (f_c - f_b) \right),
\]

\[
\psi'(f)h = \frac{1}{6} \sum_{\tau \in Q} V''(f)h
\]

\[
= \frac{1}{6} \sum_{\tau = [p,b,c] \in Q} [\det(f_p, f_b, h_c) + \det(f_p, h_b, f_c) + \det(h_p, f_b, f_c)]
\]

\[
= \frac{1}{6} \sum_{p \in I} \left( h_p, \sum_{\tau = [p,b,c] \in T_p} f_b \times f_c \right),
\]

and

\[
\alpha'(f)h = n_t \sum_{\tau \in Q} \rho_\tau \langle A_\tau(f), A'_\tau(f)h \rangle
\]

\[
= n_t \sum_{p \in I} \left( h_p, \sum_{\tau = [p,b,c] \in T_p} \rho_\tau A_\tau(f) \times (f_c - f_b) \right)
\]

for all \( h \in S_0 \). Hence the gradient components associated with a free vertex \( f_p \) are

\[ [\nabla \psi(f)]_p = \frac{1}{2} \sum_{\tau = [p,b,c] \in T_p} \frac{A_\tau(f)}{\|A_\tau(f)\|} \times (f_c - f_b), \]

\[ [\nabla v(f)]_p = \frac{1}{6} \sum_{\tau = [p,b,c] \in T_p} f_b \times f_c = \frac{1}{6} \sum_{\tau \in T_p} A_\tau(f) \quad \text{if } p \text{ is interior}, \]

(3.5)
and

\[ [\nabla \alpha(f)]_p = n_t \sum_{\tau = [p, b, c] \in T^p} \rho_\tau A_\tau(f) \times (f_c - f_b), \]  

where the expression for the gradient of \( v \) in Eq. (3.6) follows from

\[
\sum_{\tau = [p, b, c] \in T^p} (f_b - f_p) \times (f_c - f_p) = \\
\sum_{\tau = [p, b, c] \in T^p} [f_b \times f_c + f_p \times (f_b - f_c)] = \\
\left( \sum_{\tau = [p, b, c] \in T^p} f_b \times f_c \right) + f_p \times \sum_{\tau = [p, b, c] \in T^p} (f_b - f_c)
\]

with the last sum equal to zero for an interior vertex \( p \). The Euclidean gradient of \( \phi \) at \( f \) is the element of \( S_0 \) with free vertex components

\[ [\nabla \phi(f)]_p = [\nabla \alpha(f)]_p + w(v(f) - c)[\nabla v(f)]_p \]

\[
= n_t \sum_{\tau = [p, b, c] \in T^p} [\rho_\tau A_\tau(f) \times (f_c - f_b)] \\
+ \frac{w}{6} (v(f) - c) \sum_{\tau \in T^p} A_\tau(f).
\]

Note that, while the gradient of \( \psi \) at \( f \) is well-defined only for \( A_\tau(f) \) nonzero (non-degenerate triangles), there is no such restriction on the gradients of \( \alpha \) or \( \phi \).

**Hessians.** Second variations are as follows:

\[
v''(f)kh = \frac{1}{6} \sum_{\tau \in Q} V_{\tau}''(f)kh = \\
= \frac{1}{6} \sum_{\tau = [p, b, c] \in Q} [\det(f_p, k_b, h_c) + \det(k_p, f_b, h_c) + \\
\det(f_p, h_b, k_c) + \det(k_p, h_b, f_c) + \\
\det(h_p, f_b, k_c) + \det(h_p, k_b, f_c)]
\]

\[
= \frac{1}{6} \sum_{p \in I} \left( h_p, \sum_{\tau = [p, b, c] \in T^p} [f_b \times k_c + k_b \times f_c] \right)
\]

and

\[
\alpha''(f)kh = n_t \sum_{\tau \in Q} \rho_\tau [(A'_\tau(f)h, A'_\tau(f)k) + \langle A'_\tau(k)h, A_\tau(f) \rangle]
\]

\[
= n_t \sum_{\tau = [p, b, c] \in Q} \rho_\tau [(f_b - f_p) \times (h_c - h_p) + (h_b - h_p) \times (f_c - f_p), A'_\tau(f)k] \\
+ \langle (k_b - k_p) \times (h_c - h_p) + (h_b - h_p) \times (k_c - k_p), A_\tau(f) \rangle]
\]

\[
= n_t \sum_{\tau = [p, b, c] \in Q} \rho_\tau [(f_c - f_b) \times h_p + (f_p - f_c) \times h_b]
\]

...
The Gauss-Newton approximation is

\[
\nabla c v(f) + (f_c - f_b) \times h_b, A_\tau(f) k + ((k_c - k_b) \times h_p
\]

\[
+ (k_p - k_c) \times h_b + (k_b - k_p) \times h_c, A_\tau(f))
\]

\[
= n_t \sum_{p \in T} \sum_{\tau = [p, b, c]} \rho_\tau [A_\tau(f) k \times (f_c - f_b) + A_\tau(f) \times (k_c - k_b)]
\]

for \( k, h \in S_0 \) so that the Hessians are defined by

\[
[\nabla^2 v(f)_p] = \frac{1}{6} \sum_{\tau = [p, b, c] \in T^p} [f_b \times k_c + k_b \times f_c],
\]

and

\[
[\nabla^2 \alpha(f)_p] = n_t \sum_{\tau = [p, b, c] \in T^p} \rho_\tau [A_\tau(f) k \times (f_c - f_b) + A_\tau(f) \times (k_c - k_b)]
\]

\[
= n_t \sum_{\tau = [p, b, c] \in T^p} \rho_\tau \{(f_c - f_b) \times k_p \times (f_c - f_b)
\]

\[
- (f_c - f_b) \times [k_b \times (f_c - f_p) - k_c \times (f_b - f_p)]
\]

\[
+ [(f_b - f_p) \times (f_c - f_p)] \times (k_c - k_b)\}.
\]

The Gauss-Newton approximation is

\[
[\nabla^2 \alpha(f)_p] = n_t \sum_{\tau = [p, b, c] \in T^p} \rho_\tau [A_\tau(f) k \times (f_c - f_b)]
\]

\[
= n_t \sum_{\tau = [p, b, c] \in T^p} \rho_\tau \{(f_c - f_b) \times k_p \times (f_c - f_b)
\]

\[
- (f_c - f_b) \times [k_b \times (f_c - f_p) - k_c \times (f_b - f_p)]\}.
\]

The Hessian of \( \phi \) is

\[
\nabla^2 \phi(f) = \nabla^2 \alpha(f) + w (v(f) - c) \nabla^2 v(f) + w \nabla v(f) \nabla v(f)^T
\]

and the Gauss-Newton approximation is

\[
G(f) = G_\alpha(f) + w \nabla v(f) \nabla v(f)^T,
\]

where \( \nabla v(f)^T \) denotes the transpose of \( \nabla v(f) \).

The Hessian \( \nabla^2 \phi(f) \) is not positive definite unless \( f \) is close to a local minimum of \( \phi \), but the Gauss-Newton approximation \( G(f) \) is positive definite for all \( f \) by Theorem 3.1 in [15]. If a conjugate gradient method is used to solve the linear systems involving \( G \) and \( \nabla^2 \phi \), it is not necessary to compute and store the matrices. The above expressions are used to compute matrix-vector products.

### 3.3. Computational method. Least squares problem.

For fixed connection topology defined by an initial triangulation, vertex density values, target signed volume \( c \), and weight \( w \), we have the nonlinear least squares problem of minimizing \( \phi \) as defined in section 3.1. In [17] we introduced a trust region method designed for large-scale problems of this type. The trust region radius is measured in a Sobolev metric, and the method was shown to be equivalent to a method which blends a
Newton-like iteration with a steepest descent iteration using a Sobolev gradient. The underlying theory is developed in [10], and the method is applied to the problem of approximating minimal surfaces in [18].

Solutions to trust region subproblems are approximated by a dogleg method which searches a two-dimensional subspace defined by the Newton or Gauss-Newton direction and the gradient descent direction, and linear systems are solved by the conjugate gradient method with Jacobi preconditioning. In the case of an indefinite Hessian the linear solver returns a direction of negative curvature which is used in place of the Newton search direction. Refer to [11] for background on optimization methods.

The preconditioner for the conjugate gradient method is comprised of the symmetric positive semi-definite order-3 diagonal blocks of the Hessian, which are identical to those of the Gauss-Newton approximation to the Hessian. If all of the triangles containing a vertex are degenerate, the corresponding diagonal block of the Hessian is singular. In this case we simply replace it by the identity matrix.

Note that the rank-1 outer product in the expression for $G(f)$, Eq. (3.13), is a dense matrix of order $3n_v$, where $n_v$ is the number of vertices. The operation count associated with storing the matrix would be quadratic in $n_v$, while the cost of computing matrix-vector products is linear. We can eliminate the quadratic cost by using a sparse direct method with a Cholesky factorization of $G_\alpha(f)$ along with the Sherman-Morrison formula for the inverse of $G(f)$ applied to a vector. A simpler algorithm is to omit the term altogether and simply approximate $G(f)$ by $G_\alpha(f)$, and we found this to be quite effective in our testing. As with the preconditioner, a singular block on the diagonal of $G_\alpha(f)$ is replaced by the order-3 identity matrix.

Our test results in section 4 are based on two methods for treating the least squares problem, referred to as PCG and Cholesky. In both cases we use a standard trust region dogleg method that blends a Euclidean gradient direction with a Gauss-Newton direction. The PCG method solves linear systems involving $G(f)$ using the preconditioned conjugate gradient method, while the sparse Cholesky method uses a direct solver with the Hessian approximated by $G_\alpha(f)$. Convergence of the iterative linear solver is defined by an upper bound of $5 \times 10^{-6}$ on the Euclidean norm of the residual relative to that of the right hand side (the gradient). Convergence of the outer trust region iteration is defined by an upper bound of $5 \times 10^{-5}$ on the relative change in the energy functional $\phi$.

**Edge swaps.** The mesh connection topology can be altered by swapping a shared triangle edge with the line segment connecting the two opposite vertices (assuming they are not already connected), thus replacing a pair of adjacent triangles by another pair. As an example of unswappable edges, consider connecting a new vertex to the three vertices of an existing triangle. The three new edges are unswappable. If the four vertices involved in an edge swap are not coplanar, the mesh geometry and surface area are changed. An edge swapping algorithm consists of one or more sweeps through the set of swappable edges in which each edge is tested and possibly swapped according to some criterion. The algorithm is terminated when a sweep results in no edge swap. Efficiency is improved by only testing the edges that were affected by a swap in the previous sweep. We consider two edge-swapping criteria here.

The Delaunay criterion requires an edge swap if the angles opposite the edge sum to more than $\pi$. It is easy to see that, following a swap, the angles opposite the new edge will sum to less than $\pi$. It is shown in [6] that a Delaunay edge swap, in the non-coplanar case, reduces the sum of triangle areas, and consequently the edge swapping algorithm must terminate.
We introduce a new swap criterion designed to produce uniform weighted triangle area. The \textit{squared-area} criterion dictates that the swap should occur if and only if it reduces the weighted sum of squared triangle areas. Termination of the algorithm follows from an argument similar to that used in the Delaunay case. However, despite the fact that for fixed connection topology we minimize surface area by minimizing the weighted sum of squared triangle areas, an edge swap based on the squared-area criterion may actually increase the surface area (sum of triangle areas). On the other hand, the increase in surface area relative to that resulting from Delaunay edge swaps is not significant unless the mesh is very coarse. Also, we found that following each iteration of the trust region method with the edge swapping algorithm can be advantageous for the minimal surface problem.

Delaunay edge swaps, while not significantly reducing surface area or making triangle sizes more uniform, improve mesh quality by making triangles more nearly equilateral. The edge swaps cannot be intermixed with trust region iterations because they could prevent convergence, but it is sufficient and often beneficial to apply the Delaunay edge swapping algorithm after the least squares algorithm, with or without squared-area swaps, has converged.

Note that a straightforward implementation of the Delaunay swap test can result in an incorrect decision and a corrupted triangle mesh when the four vertices are nearly collinear. Refer to [4] for a numerically stable algorithm that eliminates this potential problem.

\textbf{Mesh refinement.} A subset $\mathcal{T}$ of the triangles may be uniformly refined by connecting the edge midpoints to each other, thus partitioning each triangle of $\mathcal{T}$ into four similar equal-area subtriangles. A triangle not in $\mathcal{T}$ that shares an edge with a triangle of $\mathcal{T}$ is partitioned into two equal-area triangles by a new edge from the midpoint of the shared edge to the opposite vertex.

Our algorithm consists of a loop on the edges of triangles of $\mathcal{T}$ in which a vertex is added at the midpoint of each edge, and the new vertex is joined to the vertex opposite the edge in each of the one or two triangles that share the edge, thus replacing each of the triangles by two new triangles. The result of splitting all the edges is to add three interior edges that partition each triangle of $\mathcal{T}$ into four equal-area subtriangles, and to add one interior edge that partitions each neighboring triangle (a triangle not in $\mathcal{T}$ that shares an edge with a triangle of $\mathcal{T}$) into two equal-area subtriangles. In a second loop following the splits, an edge swap is applied to the first interior edge that was added to each triangle of $\mathcal{T}$ so that the four subtriangles are similar, both to each other and to the parent triangle.

As an option, the midpoint of a boundary edge may be moved to a point computed by circular arc interpolation. The endpoints of the boundary edge, along with the two neighboring boundary vertices, one preceding the edge and the other following the edge, define a pair of circles. The new vertex is the midpoint between the intersections of the circles with the perpendicular bisecting plane of the edge. If either circle has infinite radius, the midpoint of the boundary edge is retained. The surfaces in Figure 1 were constructed using this option and three refinements of an origin-centered cube with side-length 2 and square holes centered in each side.

Note that, while our code is designed to enable adaptive mesh refinement with arbitrary subsets of the triangles, all of the test results presented in section 4 employ uniform refinement, taking $\mathcal{T}$ to be the entire set of triangles.

\textbf{Generalizations.} Our implementation and test results are limited to triangle mesh surfaces, but the method could be extended to more general settings such as
a nonmanifold mesh with boundary curves shared by more than two surface sheets. Also, as described in [12], we could allow a surface to be split into two components if an attempted Delaunay swap encounters an unswappable edge. A boundary curve, instead of consisting of fixed vertices could be comprised of vertices constrained to lie on a fixed surface but otherwise free to vary, or it could be constrained by a fixed length. These equality constraints are easily satisfied in a weak sense by simply including them in the least squares system. It is not necessary to solve a constrained optimization problem.

An unstable minimal surface is a saddle point, rather than a local minimum, of the surface area functional. Given an initial approximation \( f \), a nearby critical point of \( \alpha \) (which could be a saddle point of the surface area functional) can be computed by using a gradient descent method to minimize

\[
\beta(f) = \frac{1}{2} \| \nabla \alpha(f) \|^2.
\]

A Euclidean gradient of \( \beta \) can be approximated by finite differencing using just two evaluations of the gradient of \( \alpha \):

\[
\nabla \beta(f) = (\nabla \alpha)'(f) \nabla \alpha(f) \approx \frac{1}{\delta} [\nabla \alpha(f + \delta \nabla \alpha(f)) - \nabla \alpha(f)]
\]

for a small positive constant \( \delta \). A discretized \( H^1 \) Sobolev gradient of \( \beta \) can be computed by solving a symmetric positive definite linear system with a discrete Dirichlet Laplacian and \( \nabla \beta(f) \) as the right hand side. The critical point is then computed by a steepest descent iteration using the Sobolev gradient.

4. Test Results. Discretization error. In order to examine the asymptotic rate of convergence as the mesh width approaches zero, we experimented with simple minimal surfaces for which the surface area is known. The Riemann-Schwarz surface defined by vertices \((1/2, 0, 1/\sqrt{8})\), \((0, -1/2, -1/\sqrt{8})\), \((-1/2, 0, 1/\sqrt{8})\), and \((0, 1/2, -1/\sqrt{8})\) has surface area approximately 0.63963. We added the origin as a free vertex in a mesh with four triangles, and we employed five mesh refinements. No edge swaps were employed. Table 1 displays the number of triangles, the computed surface area, and the absolute error in the surface area for each grid. As expected the error is reduced to about a fourth of its previous value each time the mesh width is halved.

<table>
<thead>
<tr>
<th>Mesh</th>
<th>( n_t )</th>
<th>( \psi )</th>
<th>Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>4</td>
<td>0.707107</td>
<td>0.06748</td>
</tr>
<tr>
<td>1</td>
<td>16</td>
<td>0.654128</td>
<td>0.01450</td>
</tr>
<tr>
<td>2</td>
<td>64</td>
<td>0.643301</td>
<td>0.00367</td>
</tr>
<tr>
<td>3</td>
<td>256</td>
<td>0.640554</td>
<td>0.00092</td>
</tr>
<tr>
<td>4</td>
<td>1024</td>
<td>0.639869</td>
<td>0.00024</td>
</tr>
<tr>
<td>5</td>
<td>4096</td>
<td>0.639691</td>
<td>0.00006</td>
</tr>
</tbody>
</table>

The second surface is a catenoid defined by two parallel coaxial circular rings with radius \( \sqrt{2} \), separation distance 1.8, and surface area 14.6533. Starting with a square cylinder we employed five refinements using circular arc interpolation as described
in section 3.3. The mesh-4 approximation is depicted in the upper left quadrant of Figure 5, and the results are in Table 2. The helicoid is another well-known minimal surface which can be represented analytically. We generated an initial estimate with the appropriate boundary values by discretizing the ruled surface defined by two space curves:

\[ f(u, v) = (1 - v)c_0(u) + vc_1(u) \]

for \( u, v \in [0, 1] \), \( c_0(u) = (0, 0, 2u) \), and \( c_1(u) = (\cos(10u), \sin(10u), 2u) \). We partitioned the unit square into a 50-cell by 10-cell grid corresponding to 1000 triangles. The surface area of the helicoid with boundary defined by \( f \) is approximately 5.5615, while the computed surface area was 5.5609 corresponding to a relative error of \( 1.2 \times 10^{-4} \). The discrete surface is depicted in Figure 2.

As a simple example of a CMC surface we approximated a cylinder of radius 1 and length \( \pi \) for which the surface area is \( 2\pi^2 = 19.7392 \) to six digits. Starting from a square cylinder with six triangles on each side, we used four refinements with circular arc interpolation. The results are in Table 3. The relative error in the computed volume is less than 0.002 in all cases. Similar results were obtained for a sphere.
For our final test we approximated 1.5 periods of an unduloid with radii between 1 and 2 for which the surface area is approximately 136.967 (computed from a formula in [9]). The initial mesh is a square cylinder with 18 triangles on each side, and we employed three refinements with circular arc interpolation. The surface is depicted in Figure 3, and the surface area approximations are in Table 4. Relative errors in computed volume are less than 0.001 in all cases.

**Table 3**

*Errors in surface area for cylinder approximations.*

<table>
<thead>
<tr>
<th>Mesh</th>
<th>$n_t$</th>
<th>$\psi$</th>
<th>Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>24</td>
<td>21.0092</td>
<td>0.0643</td>
</tr>
<tr>
<td>1</td>
<td>96</td>
<td>19.9673</td>
<td>0.0114</td>
</tr>
<tr>
<td>2</td>
<td>384</td>
<td>19.7898</td>
<td>0.0026</td>
</tr>
<tr>
<td>3</td>
<td>1536</td>
<td>19.7575</td>
<td>0.0009</td>
</tr>
<tr>
<td>4</td>
<td>6144</td>
<td>19.7444</td>
<td>0.0003</td>
</tr>
</tbody>
</table>

**Table 4**

*Errors in surface area for unduloid approximations.*

<table>
<thead>
<tr>
<th>Mesh</th>
<th>$n_t$</th>
<th>$\psi$</th>
<th>Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>72</td>
<td>142.122</td>
<td>0.0376</td>
</tr>
<tr>
<td>1</td>
<td>288</td>
<td>138.191</td>
<td>0.0089</td>
</tr>
<tr>
<td>2</td>
<td>1152</td>
<td>137.135</td>
<td>0.0012</td>
</tr>
<tr>
<td>3</td>
<td>4608</td>
<td>137.045</td>
<td>0.0006</td>
</tr>
</tbody>
</table>

**Fig. 3.** Unduloid with radii 1 and 2.
Variable density test. In order to test the effect of variable density we approximated an unduloid, first with constant density $\rho = 1$, and then with density based on the local feature size: the reciprocal of the squared distance to the central axis. We chose a single period of an unduloid with radii 0.1 and 1.0, and employed three refinements of a 48-triangle square cylinder using circular arc interpolation. The refined mesh results are depicted in Figure 4 where the advantage of variable density is quite evident both in terms of improved triangle shape and in higher accuracy due to higher density near the ends. The quadratic convergence rate was observed for both constant and variable density with smaller relative error in surface area (by about 1/3) for the case of variable density.

Choice of linear solver. The surfaces in Figure 1 were computed using three refinements of a 48-triangle mesh forming a cube with square holes centered in each side. The refined meshes contain 3072 triangles and 1436 free vertices, corresponding to 4308 degrees of freedom. Table 5 displays iteration counts and execution times for each of the five surfaces and the two linear solver methods. The columns correspond to signed volume $v(f)$, linear solver method, trust region iteration count, preconditioned conjugate gradient iteration count, and execution time in minutes on a MacBook Air (1.8 GHz Intel Core i7 processor) running Matlab R2013b. The iteration counts are those associated with the finest grid only, while the execution time is for the entire run. The iterative solver required 78 to 140 PCG iterations per outer iteration. Since the iterative method uses more accurate Hessian approximations for the CMC surfaces, the trust region iteration counts are much lower, but the reduction in iteration count was in no case sufficient to offset the higher cost of the iterative solution method. In the case of the minimal surface ($v(f) = 2.4$) the Hessian approximations are the same, and the direct method is much more efficient than the iterative method.

Effect of edge swaps. We tested the effect of edge swaps on mesh quality and execution time using two discrete minimal surfaces and a discrete CMC surface. In all three cases we employed four refinements of an 8-triangle initial mesh, resulting in 2048 triangles and 960 or 961 free vertices (depending on the number of boundary curves). A catenoid with height 1.8 and a barrel-shaped CMC surface with height 4 and volume 30 were computed by refining square cylinders, and a saddle-shaped minimal surface was obtained from a tent-shaped initial mesh. The results are depicted in Figures 5-7 and Tables 6-8. The first two columns in the tables are the numbers of squared-area swaps and Delaunay swaps, respectively. The column labeled Radius Ratio is the minimum of the commonly used ratio of triangle inradius to circumradius (scaled by
2 so that an equilateral triangle has measure 1). We also tabulated the ratios of the smallest to largest triangle areas under heading Area Ratio, and the outer (trust region) iteration counts under TR Iter. As in Table 5, the execution time is for an entire run, while the other table entries are associated with only the finest mesh. Linear systems were solved by the Cholesky method.

**Table 5**
Iteration counts and execution times for the surfaces in Figure 1.

<table>
<thead>
<tr>
<th>$v(f)$</th>
<th>Method</th>
<th>TR Iter</th>
<th>PCG Iter</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.5</td>
<td>PCG</td>
<td>4</td>
<td>335</td>
<td>2.39</td>
</tr>
<tr>
<td>0.5</td>
<td>Cholesky</td>
<td>21</td>
<td>0</td>
<td>1.41</td>
</tr>
<tr>
<td>2.4</td>
<td>PCG</td>
<td>3</td>
<td>319</td>
<td>2.29</td>
</tr>
<tr>
<td>2.4</td>
<td>Cholesky</td>
<td>4</td>
<td>0</td>
<td>0.18</td>
</tr>
<tr>
<td>4.0</td>
<td>PCG</td>
<td>3</td>
<td>367</td>
<td>2.67</td>
</tr>
<tr>
<td>4.0</td>
<td>Cholesky</td>
<td>11</td>
<td>0</td>
<td>1.20</td>
</tr>
<tr>
<td>8.0</td>
<td>PCG</td>
<td>4</td>
<td>521</td>
<td>5.15</td>
</tr>
<tr>
<td>8.0</td>
<td>Cholesky</td>
<td>29</td>
<td>0</td>
<td>2.06</td>
</tr>
<tr>
<td>15.0</td>
<td>PCG</td>
<td>6</td>
<td>764</td>
<td>5.40</td>
</tr>
<tr>
<td>15.0</td>
<td>Cholesky</td>
<td>67</td>
<td>0</td>
<td>4.33</td>
</tr>
</tbody>
</table>

**Table 6**
Mesh quality measures and execution times for catenoid.

<table>
<thead>
<tr>
<th>Area Swaps</th>
<th>Del. Swaps</th>
<th>Surface Area</th>
<th>Surface Radius</th>
<th>Area Ratio</th>
<th>Ratio</th>
<th>TR Iter</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>14.6510</td>
<td>0.6802</td>
<td>0.95</td>
<td>2</td>
<td>0.11</td>
<td></td>
</tr>
<tr>
<td>64</td>
<td>0</td>
<td>14.6507</td>
<td>0.7098</td>
<td>0.98</td>
<td>2</td>
<td>0.12</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>14.6510</td>
<td>0.7754</td>
<td>0.95</td>
<td>2</td>
<td>0.13</td>
<td></td>
</tr>
<tr>
<td>528</td>
<td>432</td>
<td>14.6503</td>
<td>0.7523</td>
<td>0.85</td>
<td>3</td>
<td>0.15</td>
<td></td>
</tr>
</tbody>
</table>

**Table 7**
Mesh quality measures and execution times for CMC surface.

<table>
<thead>
<tr>
<th>Area Swaps</th>
<th>Del. Swaps</th>
<th>Surface Area</th>
<th>Surface Radius</th>
<th>Area Ratio</th>
<th>Ratio</th>
<th>TR Iter</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>45.3908</td>
<td>0.6124</td>
<td>0.81</td>
<td>26</td>
<td>1.46</td>
<td></td>
</tr>
<tr>
<td>609</td>
<td>0</td>
<td>45.4072</td>
<td>0.3718</td>
<td>0.80</td>
<td>9</td>
<td>1.05</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>160</td>
<td>45.3919</td>
<td>0.6771</td>
<td>0.76</td>
<td>15</td>
<td>1.12</td>
<td></td>
</tr>
<tr>
<td>1298</td>
<td>730</td>
<td>45.4002</td>
<td>0.6640</td>
<td>0.72</td>
<td>29</td>
<td>1.82</td>
<td></td>
</tr>
</tbody>
</table>

Recall that, while Delaunay swaps are applied only after the trust region iteration has converged, squared-area swaps are applied at every iteration and therefore generally have higher counts. The Delaunay swaps do not significantly reduce surface area, and they have a negative affect on uniformity as measured by the area ratios. For the catenoid and CMC surface the regularity of the initial mesh is preserved by the absence of edge swaps, and in fact, edge swaps are of limited utility. For the saddle-
Fig. 5. Discrete catenoids with 2048 triangles: no swaps (upper left), squared-area swaps only (upper right), Delaunay swaps only (lower left), both types of swaps (lower right).

Table 8
Mesh quality measures and execution times for minimal surface with saddle-shaped boundary.

<table>
<thead>
<tr>
<th>Area Swaps</th>
<th>Del. Swaps</th>
<th>Surface Area</th>
<th>Radius Ratio</th>
<th>Area Ratio</th>
<th>Iter</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>9.91458</td>
<td>0.62</td>
<td>4</td>
<td>0.19</td>
<td></td>
</tr>
<tr>
<td>177</td>
<td>0</td>
<td>9.91252</td>
<td>0.88</td>
<td>4</td>
<td>0.20</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>268</td>
<td>9.91130</td>
<td>0.26</td>
<td>7</td>
<td>0.30</td>
<td></td>
</tr>
<tr>
<td>354</td>
<td>287</td>
<td>9.91019</td>
<td>0.28</td>
<td>6</td>
<td>0.28</td>
<td></td>
</tr>
</tbody>
</table>

shaped surface, however, the mesh quality with no swaps is poor and is worsened by the squared-area swaps, making Delaunay swaps essential.

Poor initial mesh. In order to test the effectiveness of our method for un-tangling a highly corrupted initial mesh we created a mesh consisting of a Delaunay triangulation of 100 vertices randomly distributed on the unit sphere, and then perturbed the vertex components by random displacements in $[-1, 1]$. In order to remove the freedom associated with a translation of the surface we designated the first vertex as fixed. We employed two refinement steps, resulting in 3136 triangles. The initial
mesh and computed surface are depicted in Figure 8. Relative errors in the computed volume and mean curvature are approximately 0.002, and the relative error in the surface area is approximately 0.001. When the same initial mesh was input to Surface Evolver ([1]) that code failed with a degenerate triangle. The new method presented here can not only deal with degenerate triangles, but can treat the case that a vertex and all its neighbors coincide, corresponding to an entire patch of degenerate triangles.

We conclude with two more sample surfaces in Figures 9 and 10: a CMC surface and a genus-1 minimal surface.
Fig. 7. Discrete minimal surfaces with 2048 triangles: no swaps (upper left), squared-area swaps only (upper right), Delaunay swaps only (lower left), both types of swaps (lower right).

5. Conclusion. We have presented a new method for constructing CMC surfaces by minimizing a least squares functional designed to reduce surface area while concurrently improving mesh quality. The method is simple, efficient, and easy to implement, as well as robust and flexible. Test results demonstrated the effectiveness of the method for constructing minimal surfaces and more general CMC surfaces. The advantage of providing for variable density was demonstrated, and it was shown that poor input mesh quality is not a problem for the method.

REFERENCES

[8] G. Dziuk and J. E. Hutchinson, Finite element approximations to surfaces of prescribed
Fig. 8. Tangled mesh test: initial mesh with 100 randomly displaced vertices on the unit sphere (left), computed surface using two uniform refinement steps (right).

Fig. 9. *CMC surface with volume* \( v(f) = 0.5 \).
Fig. 10. Minimal surface: torus with four catenoid ends