COMPUTING MINIMAL SURFACES USING DIFFERENTIAL FORMS

Stephanie Wang - Albert Chern

SIGGRAPH 2021

English project

Maths En Action

Author

Baptiste Genest

Supervisors

Simon Masnou

Roland Denis

Contents

As the article [1] already describes the approach in a rigorous way, this text mainly aims at facilitating its comprehension, insisting on its beauty and encouraging the use of such techniques for future works.

Illustrations

The beautiful illustrations that you will see in this text comes from Keenan Crane's introductory course on discrete differential geometry [2] and from Albert Chern and Stephanie Wang SIGGRAPH course about discrete exterior calculus [3].

1 A quick primer on differential forms and their interest in differential geometry

Key idea 1. Differential forms and the operators that acts on them allow us to express, in a single framework, all of vector calculus concepts in a way agnostic of dimension and coordinate systems. Such tools are of great use in the context of differential geometry (and physics !) since they allow to capture the variations of infinitesimal geometric elements (k-vectors) while simplifying notations. It is very important to keep in mind that, from a numerical point of view, every computation boils down to standard vector operators.

1.1 k-vectors

One could define what *differential* approaches are by saying that it is the use of derivatives to compute local polynomial approximations of a possibly complex object, for instance a surface. If you had to compute the area of a manifold, one natural idea would be to approximate your surface by a collection of small parallelograms that will fit the surface when the resolution tends to zero, and to sum up the area of each element to approximate the surface area.

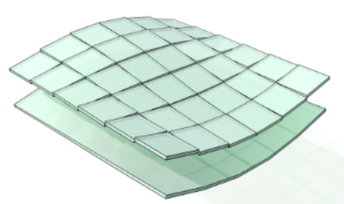
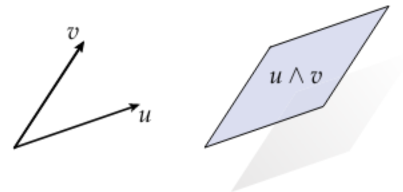


Figure 1: A collection of 2-vectors approximating a surface

Then, to give an orientation to the surface, we would like to be able to define *oriented* parallelograms. That is precisely what k-vectors are. To make things more precise, we define k-vectors by "gluing" together k elemental vectors using the *wedge* product, which is a multilinear and alternating operator :

$$u \wedge v = -v \wedge u$$



Definition 1. A k-vector $v \in \bigwedge^k V$ is an object that can be decomposed as a linear sum of *simple* k-vectors of the form:

$$v = v_1 \wedge \dots \wedge v_n, \text{ where } v_k \in V$$

which in turn can be decomposed in the k-vector basis.

For instance, in $\mathbb{R}^3$, following from the multilinear and alternating property of $\wedge$, we have :

$$\begin{pmatrix}1\\2\\3\end{pmatrix} \wedge \begin{pmatrix}4\\5\\6\end{pmatrix} = (2 \times 6 - 3 \times 5)e_2 \wedge e_3 + (1 \times 6 - 3 \times 4)e_1 \wedge e_3 + (1 \times 5 - 2 \times 4)e_1 \wedge e_2$$

1.2 k-forms

To be able to *measure* those elements, we need to define dual objects that compute quantities from k-vectors, namely k-forms. In the same way, k-forms are formed by gluing elements of the *dual* basis, defined as : $dx^i(e_j) = \delta_{ij}$.

Proposition 1.1. From the alternating property of the wedge product, it can easily be shown that the basis of k-forms in $\mathbb{R}^n$ has a size of $\binom{n}{k}$. In $\mathbb{R}^3$:

$$\begin{aligned}
&\text{0-forms : } 1\\
&\text{1-forms : } dx, dy, dz\\
&\text{2-forms : } dx \wedge dy, dx \wedge dz, dy \wedge dz\\
&\text{3-forms : } dx \wedge dy \wedge dz
\end{aligned}$$

Key idea 2. One very handy computation rule and important fact is that one should see k-forms as machines that compute determinants from projected vectors, indeed, since we saw that k-forms are linear and alternating, one can show that they must be multiple of the usual determinant ! To compute the value of a k-form applied to k vectors (or a k-vector), one can apply the following rule

$$dx^{i_1} \wedge \dots dx^{i_k}(x_1, \dots, x_k) = \det(\Pi_{i_1,\dots,i_k}(x_1, \dots, x_k))$$

Where $\Pi_{i_1,\dots,i_n}$ is the orthogonal projection over the coordinates $i_1, \dots, i_n$.
For example :

$$dx \wedge dz \left(\begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix}, \begin{pmatrix} 4 \\ 5 \\ 6 \end{pmatrix} \right) = 1 \times 6 - 3 \times 4 \tag{1}$$

1.3 Differential k-forms

Definition 2. A differential k-form is a smooth field of k-forms.
We will denote the set of differential k-forms over M as $\Omega^k(M)$.

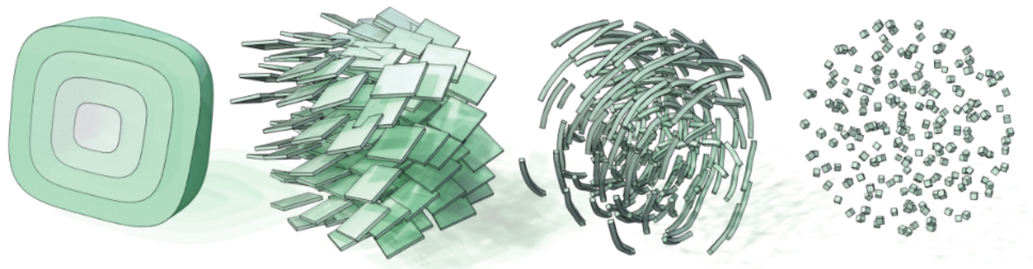
Key idea 3. In $\mathbb{R}^3$, since the dimension of the basis of k-forms is 1,3,3,1, it is very common to represent them using scalar fields for 0 and 3 forms or a 3D vector fields for 1 and 2 forms, for instance :

$$(v)_{1-\text{form}}(x) = v_1(x)dx + v_2(x)dy + v_3(x)dz$$
$$(f)_{3-\text{form}}(x) = f(x)dx \wedge dy \wedge dz$$

Key idea 4. One should see differential k-forms as fields of objects of *codimension* k ($(n-k)$-vectors). Indeed, from the multilinear-alternating property of k-forms, in $\mathbb{R}^n$, one can prove that a k-form can always be computed as the determinant of an $n \times n$ matrix :

$$\omega(x_1 \wedge \dots \wedge x_k) = \left| \begin{pmatrix} | & & | & | & & | \\ \alpha_1 & \dots & \alpha_{n-k} & x_1 & \dots & x_k \\ | & & | & | & & | \end{pmatrix} \right|$$

where the $\alpha_i \in \mathbb{R}^n$ are induced by the coefficients of ω and correspond to a basis of $\mathrm{Ker}(\omega)$. For example, evaluating the 2-form : $(cdx \wedge dy + bdx \wedge dz + ady \wedge dz)(\mathbf{x_1}, \mathbf{x_2})$ is equivalent to computing $\det((a,b,c)^t, \mathbf{x_1}, \mathbf{x_2})$, hence one can represent the 2-form by the vector field $(a,b,c)^t$. Note that one can also see differential k-forms as fields of k-vectors but this interpretation does not interacts as well with the other operators, for more details see [3].



1.4 Operators on differential forms

1.4.1 Musical isomorphisms $\sharp, \flat$

A very handy operator is $v^\flat = (v)_{1-\text{form}}$ and its inverse $((v)_{1-\text{form}})^\sharp = v$. They exploit the isomorphism between vector fields and 1-forms.

Note that one can see 1-forms as a field of $(n-1)$-vectors in accordance with Key idea4, or as a standard smooth vector field. Note that these two points of view are not contradictory since one can see a scalar product with a vector v as a determinant with an adequate basis of $v^\perp$.

1.4.2 Hodge Star

The Hodge star is also an operator that simplifies notations. Since $\binom{n}{k} = \binom{n}{n-k}$, we can associate with each k-form an $(n-k)$-form , for a orthonormal direct basis of $\mathbb{R}^n$ $(e_1, \dots, e_n)$:

$$\star(e_1 \wedge \cdots \wedge e_k) = e_{k+1} \wedge \cdots \wedge e_n$$

and $\star \star \eta = (-1)^{k(n-k)} \eta$. For instance : $\star 1 = dx \wedge dy \wedge dz$ and $\star dx = dy \wedge dz$.

The combination of the previous operators allows to build differential forms from vector fields : if f is a smooth scalar function and v a smooth 3D vector field :

If f is a smooth scalar function and v a smooth vector field

$$\begin{aligned}
(f)_{0-\text{form}} &= f \\
(v)_{1-\text{form}} &= v^\flat \\
(v)_{2-\text{form}} &= \star(v^\flat) \\
(f)_{3-\text{form}} &= \star f
\end{aligned}$$

1.4.3 Exterior derivative d

As already mentioned, the interest of differential forms is that they unify all of vector calculus in a single language. The exterior derivative is what allows to recover all formulas that involve *curl*,*div*, *gradients*, etc...

The exterior derivative d is defined as follows:

Definition 3. For a differential k-form $\omega(x) = \sum_i^k f_i(x)dx^{i_1} \wedge \cdots \wedge dx^{i_k}$, we define:

$$d\omega(x) = \sum_i \frac{f_i(x)}{\partial x_i} dx^i \wedge dx^{i_1} \wedge \cdots \wedge dx^{i_k} \tag{2}$$

One should notice that d raises the degree of a k-form by one.

It has a lot of important properties, maybe the most important one is :

Proposition 1.2. $d^2 = d \circ d = 0$

As promised, in $\mathbb{R}^3$ we can recover all standard operators :

$$\begin{aligned}
d((f)_{0-\text{form}})(x) &= (\nabla f)_{1-\text{form}}(x) \\
d((v)_{1-\text{form}})(x) &= (\nabla \times v)_{2-\text{form}}(x) \\
d((v)_{2-\text{form}})(x) &= (\nabla \cdot v)_{3-\text{form}}(x)
\end{aligned}$$

Note that from the property $d^2 = 0$, we directly obtain that $\text{curl}(\nabla f) = 0$ and $\text{div}(\text{curl}(v)) = 0$.

Let us try to define the laplacian on 0-forms : since $\Delta f = \text{div}(\nabla f)$, we can get the gradient using df, which is then a 1-form, then to obtain the divergence of the gradient, we first use the Hodge star to obtain a 2-form, $\star df$, then applying d once again yields $(\nabla \cdot \nabla f)_{3-\text{form}}$. Finally we apply the Hodge star to recover a 0-form [1]:

$$\star d \star df = \Delta f.$$

Since $\star d \star$ appears often, it is named the *codifferential* operator $\delta = \star d \star$.

Another important aspect is the link between d and the boundary operator. Such connection comes primarily from the amazing Stokes Theorem:

Theorem 1.3. Stokes Theorem

$$\int_M d\omega = \int_{\partial M} \omega$$

.

The elegance and the power of such statement is in itself enough to justify studying the theory of differential forms. It leads easily to some of the most important vector-analysis theorems: the fundamental theorem of analysis, Green-Ostrogradski formula, Gauss divergence theorem, etc...

[1] For any differential form, the laplacian is defined as: $\Delta = \delta d + d\delta$

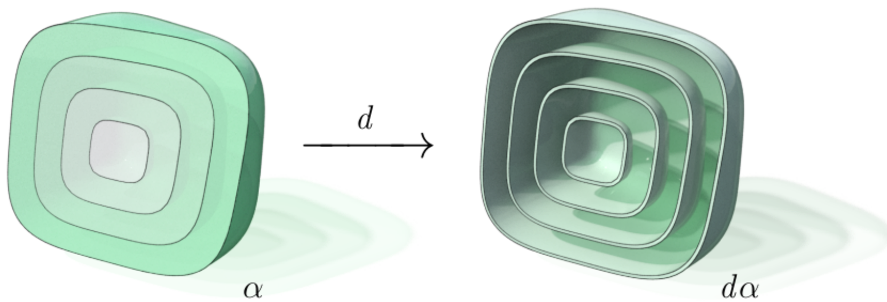


Figure 2: The exterior derivative corresponds to the boundary operator on the orthogonal representation

2 Currents

Even though the article does not explicitly use the term of *currents*, the authors define such object as a limit of a sequence of differential forms whose value tend to infinity on a set and to zero everywhere else. This of course reminds of the definition of the Dirac δ distribution, which is relevant since **currents are to differential forms what distributions are to $\mathcal{C}_c^\infty$ functions**.

The set of currents is defined as the dual of the set of differentials forms, namely :

> **Key idea 5.** Currents represent any kind of objects you can integrate a differential form on. For instance manifolds.

One might ponder, if currents are used to represent manifolds, why not directly handling manifolds ? It turns out that the answer is the same as for distributions : this dual formulation allows a variational approach to the problem which is highly relevant here since the manifold itself is the unknown.

> **Key idea 6.** In this article, in a way similar to *phase fields*, the target manifold is approached as a limit of smooth differential forms on $\mathbb{R}^3$ whose support tends to Σ and whose value tends to the normal field $n_\Sigma^\flat$.

3 Minimal surfaces

A minimal surface is a solution of the following problem :

$$\min_{\Sigma:\partial\Sigma=\Gamma} \text{Area}(\Sigma) \tag{3}$$

Which provides a mathematical formulation for soap films.

The starting point of expressing minimal surfaces as an optimization problem over dual objects comes from this expression:

$$\text{Area}(\Sigma) = \int_\Sigma 1 dA = \int_\Sigma ||n(x)||^2 dA = \sup_{v:\Sigma\mapsto\mathbb{R}^3, |v|\leq 1} \int_\Sigma v(x)\cdot n(x) dA \tag{4}$$

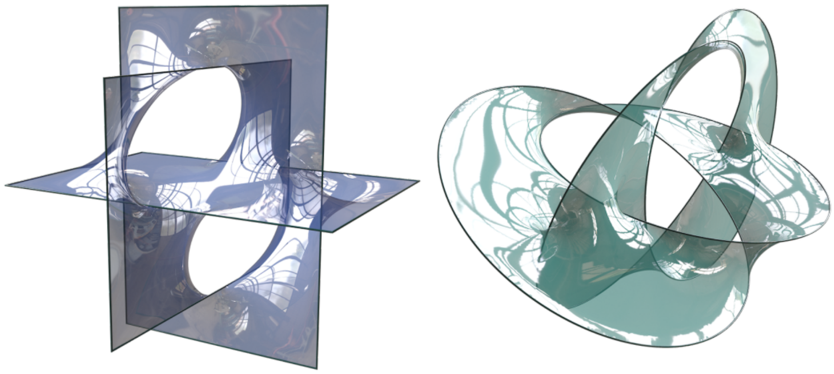


Figure 3: Given a boundary Γ, we aim at computing the surface with minimal area such that $\partial\Sigma = \Gamma$

Such rewriting expresses the area as a maximization problem where the solution is the normal field of the surface. [2]

One natural way to see the connection with currents is that one might recognize a *dual norm*, i.e., if $(X, ||.||)$ is a normed vector space (here smooth vector fields or, equivalently, $\Omega^1(\mathbb{R}^3)$ by the isomorphism $v = \omega^\sharp$), the dual norm on X^* is, for $f \in X^*$:

$$||f|| = \sup_{x \in X, ||x|| \leq 1} |f(x)|.$$

Useful analogy 1. One should recognize a generalization of standard matrix norms.

In our context, where the dual space is the one of currents, f is here δ_Σ:

$$\langle \omega, \delta_\Sigma \rangle = \int_{\mathbb{R}^3} \omega \wedge \delta_\Sigma = \int_\Sigma \omega dS \tag{5}$$

Note that in the context of currents, the dual norm is called the *mass norm*, noted $||\cdot||_{\text{mass}}$

Useful analogy 2. This expression exhibits the analogy with Dirac distributions where the duality bracket is defined as the integral over the support of the Dirac but here with the additional normal information carried by the current.

Which leads to the key equality of the approach:

$$\text{Area}(\Sigma) = ||\delta_\Sigma||_{\text{mass}} \tag{6}$$

[2]this article's approach can be summed up as : finding the normal field of the unknown surface where the variable of optimization is a smooth vector field defined over $[0, 1]^3$

Key idea 7. This equality illustrates the link between measure theory and geometry and motivates the interest of the field of *geometric measure theory* (for a comprehensive introduction, see [4]).

4 The article's approach

4.1 Shape representation and discretization

As previously mentioned, we aim at computing the surface that minimizes the Area functionnal :

$$\min_{\Sigma:\partial\Sigma=\Gamma} \text{Area}(\Sigma) \tag{7}$$

But of course, this is the continuous problem that cannot be solved exactly nor will fit into a computer, so one must discretize the shape, hence asking a fascinating question : *How to represent a shape in a computer?*

Of course there are many possible answers but one can split them into 2 major families

- Explicit : Describing the shape by a finite subset, possibly specifying topological information. examples : *Mesh, Point cloud, Templates*[3]
- Implicit : Describing the shape by providing a way to query various requests about its geometry, such as : "Is the point x in the shape?", "At what (signed) distance is the point x from the shape ?". examples : *Signed distance fields (CSG), Phase fields, level sets*

From a computational perspective, when dealing with surfaces in $\mathbb{R}^3$, computing on an explicit representation of a shape, for instance a mesh, has a complexity of the order of n^2 where n is the resolution of the discretization. But, meshes are famously un-adapted to topological changes, which is very common in the context of shape optimization problems, such as this one.

On the other hand, *implicit* representations can handle topological changes without any trouble, but the computations on implicit representation comes with a cost of the order of n^3, which is often unsatisfactory to get both precision and speed.

For example, solving a linear system for a function discretized on such grid involves classically $\mathcal{O}(n^6)$ operations[4], while solving it on a mesh would result in $\mathcal{O}(n^4)$ operations.

To speed up computations, this article uses the FFT. For operators that have a nice form in the Fourier domain, such as the Laplacian, where the solution can be expressed as a component-wise product[5] with a known kernel in the Fourier domain, we get an $\mathcal{O}(n^3\log(n^3)) = \mathcal{O}(n^3\log(n))$ complexity. Very importantly, using the FFT implies many interesting questions, covered in section4.3.

As already mentioned, explicit representations would not fit here, and the main interest of the article is to provide an effective way to represent shapes by *currents*. One might ask, why using a current and not another way?

[3] for a very clear introduction, I recommend [5]

[4] since we have n^3 variables and best linear solvers have a v^2 complexity of the number of variables v

[5] hence requiring $\mathcal{O}(n^3)$ operations on a grid, which makes it negligible in front of the cost of computing the FFT on the grid and its inverse.

Key idea 8. First, the fact that the area of a surface is equal to the mass norm of the associated current will be helpful, but most importantly, as described in section 1.3, using the exterior derivative, *one can translate topological constraints into differential ones !* (which is easier to discretize !)
Indeed, using *Stokes' theorem* : we have

$$\langle \eta, \delta_\Sigma \rangle = \int_{\partial\Sigma} \eta = \int_\Sigma d\eta = \langle d\eta, \delta_{\partial\Sigma} \rangle$$

This implies that we will be able to impose that our surface has the given boundary simply by putting

$$d\eta = \delta_\Gamma$$

4.2 Relaxing the minimization problem

Let us see how can we rewrite the problem (6) to reach something easier to minimize in practice. We first use the equivalence with the mass norm

$$\min_{\Sigma : d\delta_\Sigma = \delta_\Gamma} ||\delta_\Sigma||_{\text{mass}}$$

Of course, working with current directly will not help us since we can't discretize a shape we do not know yet. In the founding work of geometric measure theory [6], one main result is that the problem (6) can be relaxed by saying that the following problem is equivalent :

$$\min_{\eta \in \Omega^1(M) : d\eta = \delta_\Gamma} ||\eta||_1 \tag{8}$$

The difference being that the support of the current is only on the unknown surface whereas a one form is defined on the hole space and can be discretized on a grid.

Useful analogy 3. Minimizing an L^1 norm should ring the *sparsity* bell inside your head, and rightfully here ! We seek for the 1 form whose support is the smallest as possible while satisfying the border constraint (valued only on Σ and 0 elsewhere).

Now, we can describe the space our variable lives in with more ease. In particular, the Helmholtz-Hodge decomposition describes the space of 1-forms in a particularly handy way for this problem.

4.3 Using the FFT... but at what cost?

In contexts where we have to solve PDEs often, using the FFT allows to extensively reduce computation costs, but, very interestingly, this has *topological* consequences. Namely, since the FFT imposes to work on a periodic domain, the cohomology group of the ambient space isn't trivial anymore. *What does it mean and imply?*

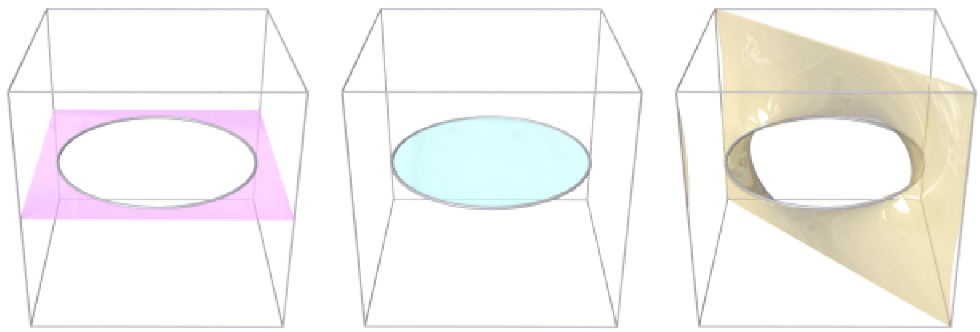


Figure 4: On a periodic domain, a new family of solution exists that don't fit the original problem statement

Helmholtz-Hodge decomposition

> **Key idea 9.** In order to use the FFT, the periodicity of the domain must be handled. Namely, periodicity adds another unwanted degree of freedom to the solution of our minimization problem. We aim at fixing this degree of freedom to recover the original solution while still benefiting of the FFT's speed up.

Regarding the L^2 scalar product[6], the set of 1 differential-forms can be written as the following orthogonal sum:

> **Theorem 4.1.**
> $$\Omega^1(M) = \mathrm{Im}(d^0) \oplus \mathrm{Ker}(d^1)^\perp \oplus H^1(M) \tag{9}$$

where $H^1(M)$ is the set of *harmonic* 1 forms on M, i.e., 1 forms ω satisfying, $d\omega = 0$ and $\delta\omega = 0$. This group is isomorphic to the cohomology of M, which is the quotient vector space defined as $H^1(M) = \mathrm{Ker}(d^1)/\mathrm{Im}(d^0)$. Even though differential forms appears to only capture local variations and geometry, a very beautiful and deep bridge exists between them and topological invariants of the spaces they live on. If $M = \mathbb{R}^3$, then H^1 is trivial, but since we now work on T^3, H^1 is 3-dimensional. That means that the solution now has an additional degree of freedom that exists only because of a practical choice, which we need to fix.

> **Useful analogy 4.** To express it in the terms of vector operators, the Helmoltz-Hodge decomposition simply tells you that any smooth vector vield can be decomposed as $v = \alpha + \beta + \gamma$ where $\mathrm{div}(\alpha) = 0, \mathrm{curl}(\beta) = 0$ and, if M has a non trivial topology (else $\gamma = 0$), $\mathrm{div}(\gamma) = \mathrm{curl}(\gamma) = 0$.

The Helmoltz-Hodge decomposition is very helpful here since we can now think about our solution in terms of each component of the decomposition, namely :

[6]The scalar product between two k-forms if defined as $\langle \alpha, \beta \rangle = \int_M \alpha \wedge \star\beta$

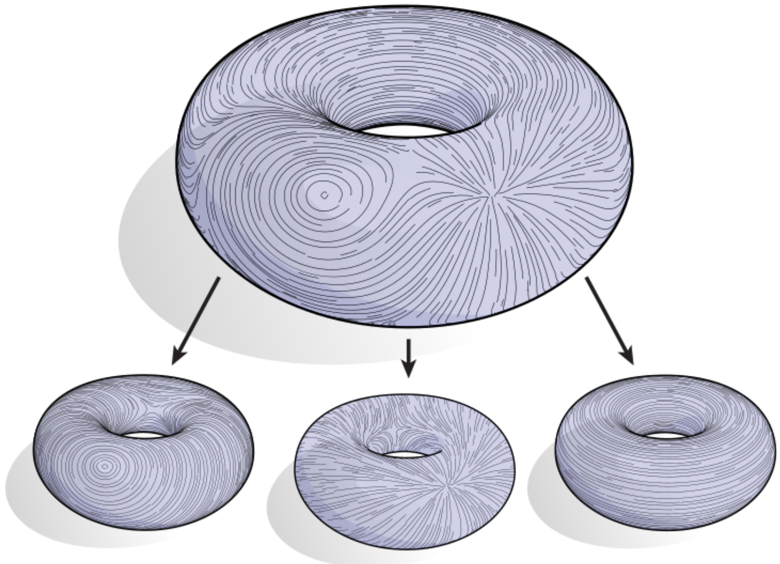


Figure 5: Helmholtz-Hodge decomposition on a periodic domain

- Since we want to impose $d\eta = \delta_\Gamma$, this implies that we fix the component of η in $\mathrm{Ker}^\perp(d^1)$.
- Since we also want to not use the periodicity of the domain, we want to fix the component of η in $H^1(T^3)$, to do so, we can define a basis of $H^1(T^3)$ to be then able to impose that the coordinates of η in $H^1(T^3)$ are the same as of δ_Σ, so that it respects our original topological setting :

  The most natural basis of $H^1(T^3)$ are the canonical basis fields, viewed as 1-forms :

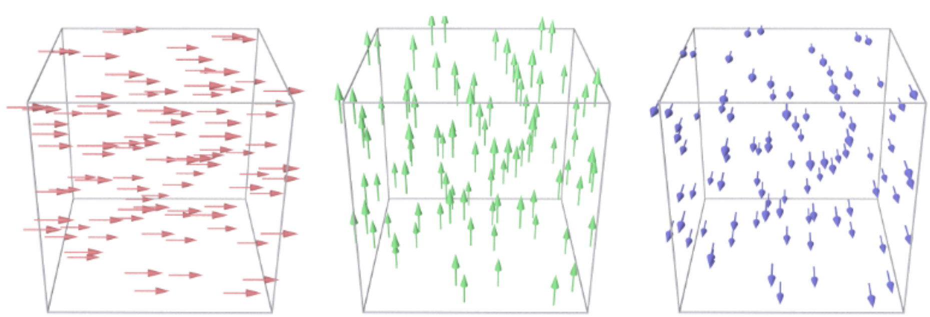


Figure 6: A basis of $H^1(T^3) = \{e_x, e_y, e_z\}$

We can then compute the coordinates of δ_Σ in $H^1(T^3)$ by taking the dot product with each basis vector, namely :

for $i \in \{1, 2, 3\}$,

$$\langle e_i, \delta_\Sigma \rangle = \int_M \langle e_i, \delta_\Sigma \rangle dx = \int_\Sigma \langle e_i, n_\Sigma \rangle dS = \langle e_i, \int_\Sigma n_\Sigma dS \rangle = A_i \tag{10}$$

One can recognize the expression $\int_\Sigma n_\Sigma dS$ since it is known as the *vector* area of a surface, $\vec{A}(\Sigma)$.

One might think, "But... You fool ! How can we determine $\vec{A}$ since we don't know Σ ?!". But all hope is not lost ! Actually, one can compute the vector area of a surface *only using its boundary* Γ, thanks to the formula :

$$\vec{A}(\Sigma) = \int_\Sigma n_\Sigma dS = \int_\Gamma \gamma \times d\gamma$$

where $\Gamma = \{\gamma(t), t \in [0,1]\}$.

Useful analogy 5. A direct discrete analogue of this formula exists, know as the *shoelace* formula, as it allows to compute the *signed* area of any polygon $P = (x_1, \ldots, x_n)$ only using its boundary:

$$\vec{A}(P) = \sum_i x_i \times \frac{x_{i+1} - x_{i-1}}{2}$$

where $\times$ is the usual cross product.

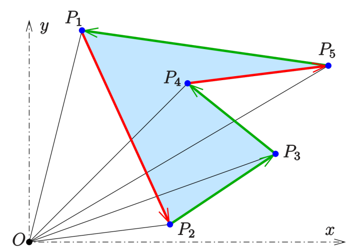


Figure 7: Since the cross product is alternating, the excess of area that is counted twice cancels out since it is counted once positively and once negatively

- This shows that the only degree of freedom we can't set directly, and that needs to be found by this optimization problem, is the $\mathrm{Im}(d^0)$ part of η.

In conclusion, the problem (8) can be stated on T^3 as :

$$\min_{\eta \in \Omega^1(T^3): d\eta = \delta_\gamma, \langle e_i, \eta \rangle = A_i} ||\eta||_1 \tag{11}$$

Even though this is a correct rewriting of the original problem, using a numerical method directly on (11) would be painful since the conditions are quite heavy to impose at each iteration. But since the conditions are orthogonal we can encapsulate them by starting the minimization from a 1-form that already satisfies these constraints...

4.4 Starting from somewhere.

4.4.1 How beautiful life would be with η_0

If we had an initial guess η_0 that already satisfied conditions (11), adding a 1-form in $\mathrm{Im}(d^0)$ would not violate the constraints ! Hence, we can rewrite (11) very neatly as :

$$\min_{\phi \in \Omega^0(T^3)} ||\eta_0 + d\phi||_1 \tag{12}$$

How beautiful !

Finally, since most efficient optimization algorithms require functionals to be convex, one can pose the following change of variable :

$$\min_{X,\phi : X = D\phi + \eta_0} ||X||_1 \tag{13}$$

In the end, we have a convex problem of two variables under linear constraints, hence we can use *ADMM*! Indeed, ADMM (Alternating Direction Method of Multipliers), first defined in [7], can be used to solve problems of the form :

$$\begin{cases} \min_{u,v} f(u) + g(v) \\ Au + Bv = c \end{cases}$$

Where f, g are usually convex and A, B are linear operators. Hence here we have $f = ||.||_1, g = 0, A = I, B = -D, c = \eta_0$, where D is a finite difference approximation of the gradient[7]. Note that the authors use an accelerated version of the algorithm defined in [8].

4.4.2 Making life beautiful

Let us now try to find a suitable η_0. As already mentionned, we wish that $\eta_0 \in \Omega^1(T^3)$ satisfies :

$$d\eta_0 = \delta_\Gamma \tag{14}$$

$$\forall i \in [[1, 3]], \langle e_i, \eta_0 \rangle = A_i \tag{15}$$

When writing equation (14) using the fact that η_0 is a 1-form and that d acts on 1-forms as the *curl* operator, on can rewrite the condition as :

$$\mathrm{curl}(\eta_0) = \delta_\Gamma$$

Where δ_Γ is interprated as a vector field. Of course, as the Helmholtz-Hodge clearly shows, the curl operator *is not* injective (two vector fields that differ only by a curl free field will have the same curl). Since this problem is under constrained, we would like to define a *pseudo*-inverse operator d^+. As in the finite dimensional setting, it can be defined here as :

$$\min_{\eta_0 \in \Omega^1(M)} ||\eta_0||^2$$
$$\text{s.t. } \mathrm{curl}(\eta_0) = \delta_\Gamma$$

[7] see section 5 for details about the discretization

The solution to that problem is :

$$(d \star d \star + \star d \star d)\omega = (d\delta + \delta d)\omega = \Delta\omega = \delta_\Gamma$$
$$d^+ \delta_\Gamma = \star d \star \omega$$

Since ω is a 1-form in $\mathbb{R}^3$, we can rewrite this more simply as :

$$\mathrm{curl}^+(v) = \mathrm{curl}(-\Delta^{-1} v)$$

where Δv is the *vector* Laplacian of v.

One can easily check that this is a pseudo inverse since, if we use the following identity :

$$\mathrm{curl}(\mathrm{curl}(v)) = \nabla(\nabla \cdot v) - \Delta v$$

we have here :

$$\begin{aligned} \mathrm{curl}(\mathrm{curl}^+(v)) = \mathrm{curl}(\mathrm{curl}(-\Delta^{-1} v)) &= -\nabla(\nabla \cdot \Delta^{-1} v) + \Delta\Delta^{-1} v \\ &= v - \nabla(\Delta^{-1} \nabla \cdot v) \end{aligned}$$

Hence, if $\nabla \cdot v = 0 \iff v \in \mathrm{Im}(\mathrm{curl})$, we have $\mathrm{curl}^+(v) = v$, which proves that is the appropriate pseudo-inverse operator.

Useful analogy 6. The term $\nabla \Delta^{-1} \nabla \cdot v$ is the orthogonal projector on $\mathrm{Im}(\nabla)$, indeed :
if $f = v + \nabla\varphi$, where $\nabla \cdot v = 0$, we have :

$$\begin{aligned} \nabla \cdot f = \nabla \cdot v + \nabla \cdot \nabla\varphi &= \Delta\varphi \\ \implies \Delta^{-1} \nabla \cdot f &= \varphi \\ \implies \nabla\Delta^{-1} \nabla \cdot f &= \nabla\varphi. \end{aligned}$$

From that we can deduce that $\mathbb{P} = \mathbb{I} - \nabla\Delta^{-1}\nabla\cdot$ is the orthogonal projector on $\mathrm{Im}(\mathrm{curl}) = \{\nabla \cdot u = 0\}$, which is very common in fluid dynamics.
The analogy with the finite dimensional case is once again striking : AA^+ is the orthogonal projection on $\mathrm{Im}(A)$.

This operation is known in physics as the *Biot-Savard law*, it describes the magnetic field induced by an electric current running through a curve (here the curve is δ_Γ).

4.4.3 Imposing the cohomology constraint

To make sure that η_0 satisfies $\vec{A}(\eta_0) = \vec{A}(\Sigma)$, we can first compute the *Biot-Savard law* to impose the boundary constraint:

$$\tilde{\eta_0} = \mathrm{curl}(-\Delta^{-1} \delta_\Gamma)$$

and then settting:

$$\eta_0 = \tilde{\eta_0} - \vec{A}(\tilde{\eta_0}) + \vec{A}(\Sigma)$$

Since $\vec{A}$ is linear and $\vec{A}(v) = v$ if v is constant.

4.5 Extracting the surface from the normal field

4.5.1 Poisson Surface Reconstruction

Once the normal field of the solution is computed, we can apply a very famous algorithm to recover the underlying surface, *Poisson Surface Reconstruction*, see [9].

The basic idea of the algorithm is very simple : if a surface Σ is represented by a signed distance function (SDF) φ, its normal field can be computed as

$$n_\Sigma(x) = \nabla\varphi(x) \tag{16}$$

$\forall x \in \Sigma$, or equivalently $\forall x$ s.t. $\varphi(x) = 0$. Hence, the idea of the surface reconstruction algorithm is to compute a reasonable guess to find a function φ such that its gradient is the normal field n_Σ. We can then extract the recovered surface as $\{\varphi(x) = 0\}$.

Key idea 10. Even though *Poisson Surface Reconstruction* was developed to recover a surface from an oriented point cloud (for exemple from a LIDAR scan), we use it here to convert our solution defined on a grid to a more convenient form (a mesh).

Hence, in a way very similar to the *Biot-Savard law*, we aim to compute ∇^+.

We start from (16):

$$n_\Sigma = \nabla\varphi$$

we apply the divergence on both sides, and then since $\nabla \cdot \nabla = \Delta$, we can then recover φ by solving[8] the following *Poisson* problem :

$$\Delta\varphi = \nabla \cdot n_\Sigma$$

4.5.2 The marching cubes

The extraction of $\{\phi = 0\}$ is done using another famous algorithm, *the marching cubes*, that builds the mesh of a level-set by running through each cube of a grid. More precisely, to extract the 0-level set of a continuous function ϕ discretized on a n^3 grid, the algorithm proceeds like this :

- look at each cell C_{ijk}
- if all the values of ϕ at the corner of the cell C_{ijk} do not have the same sign that means that the level set $\{\phi = 0\}$ must run through this cell
- add the appropriate triangles so that the set of vertices $\{v_{xyz} \in C_{ijk} | \phi_{xyz} < 0\}$ and $\{v_{xyz} \in C_{ijk} | \phi_{xyz} > 0\}$ are separated.

Up to rotations, all the possible combinations are :

for a comprehensive survey, see [10].

[8] we can use again the FFT to solve it really fast since the Green kernel of the Laplacian is known and simple, namely, in 3D, $\hat{\Delta}(\zeta) = \frac{1}{||\zeta||^2}$

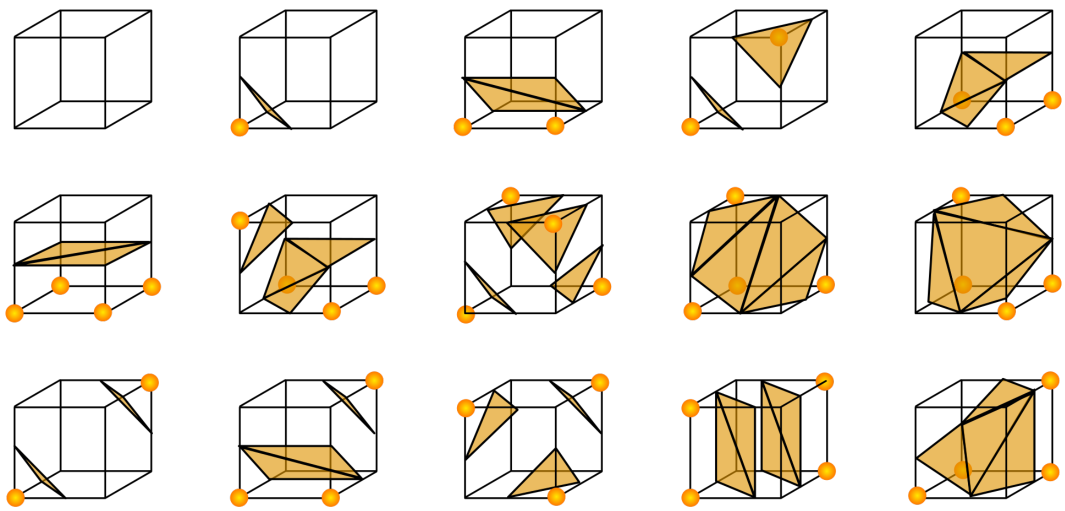


Figure 8: The marching cubes algorithm

5 Implementation details

5.1 Discretization

First, since they aim at using the FFT, the discretization space is an $n \times n \times n$ grid, we note $h = \frac{1}{n-1}$ the resolution. Following the founding work of discrete exterior calculus [11], the authors discretize k-forms by their integral over k-geometric elements, for instance, 1 forms are discretized as:

$$\eta_{i,v} = \int_{v}^{v+he_i} \eta,$$

hence stored as vectors fields defined at each vertex v. And 2-forms as:

$$\omega_{i,v} = \int\int_{[v,v+he_j]\times[v,v+he_k]} \omega, \text{where } \varepsilon_{ijk} = 1$$

where ε is the Levi-Civita symbol, which is equivalent to taking the integral of ω on the face based at vertex v with normal e_i. Note that while 1-forms should be stored on edges and 2-forms on faces, the fact that the grid is periodic induces that $E = F = V \times \{1, 2, 3\}$ hence allowing them to store everything at the vertices to simplify computations.

Furthermore, the fact that the edges are axis-aligned boils down all discretizations to standard finite difference techniques. For instance, to define the gradient operator D used in (13), they simply use:

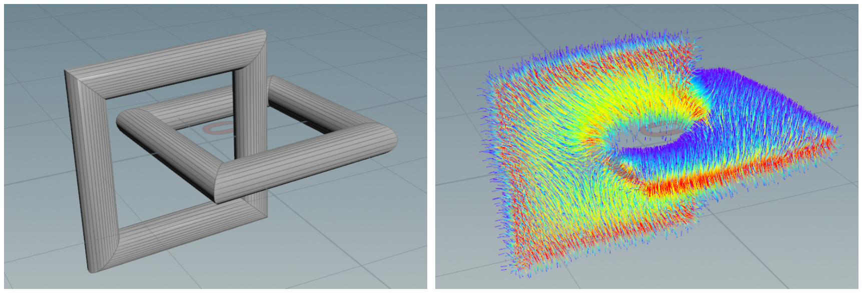
$$(D\phi)_{v,i} = \frac{1}{2h}(\phi_{v+he_i} - \phi_{v-he_i})$$

which is the standard central difference scheme, of order 2 .

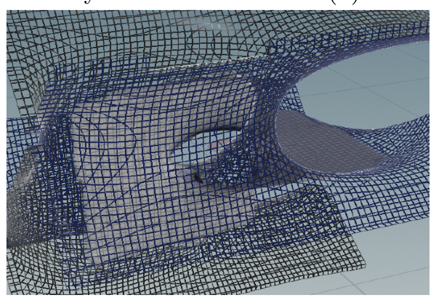
No further numerical analysis is provided since they only adopt the framework defined in [11] and only use standard operators (D, Δ,etc...).

5.2 Implementation provided

The authors provide an implementation in the Houdini software, that allows to render many geometrical objects programmatically. Houdini and the code given here works in a way following the "no-code" philosophy, where direct code is replaced with a succession of nodes that allows to chain operations in a more readable and compact way.



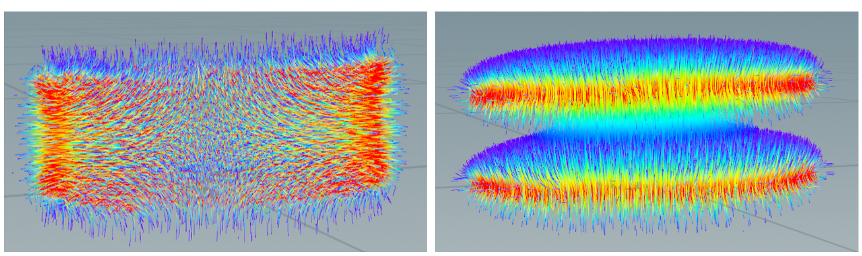
(a) Given boundary Γ (b) Solution to (13)



(c) Extracted surface

Figure 9: The algorithm's pipeline

By the way, playing with the initial boundary made me question the importance of the way the boundary is parametrized. We realized that it is a critical point since : *if the multiple boundaries are not parametrized in a "compatible" way, the algorithm does not provide the solution!* This comes primarily from the fact that the vector area changes sign when the parametrization is reversed :



(a) Correct boundary parametrization (b) Incorrect boundary parametrization

Figure 10: Importance of boundary orientation

6 On a personal note, general thoughts on the paper

6.1 Strengths

First and foremost, the aspect of the paper I appreciate the most is the fact that they bring geometric measure theory (GMT) concepts into the computer graphics community in a very elegant way. They successfully combine : mathematical rigor, computational efficiency, and clarity of exposition. On a mathematical aspect, the true contribution here, since the link between GMT and minimal surfaces is not new, is the use of the FFT to speed-up computations while handling its theoretical (topological) consequences. The derivation of the final optimization problem, using the Helmoltz-Hodge decomposition is very clever and elegant. Finally, regarding the optimization process itself, starting the search for a minimum from a pertinent initial guess while making it compatible with ADMM provides a very fast minimization.

6.2 Limitations

The major flaw of the paper is the lack of details regarding the orientation of the boundary and of the resulting surface. As demonstrated in figure 10, if the boundaries are not oriented accordingly the minimal surface cannot be reached. Even though the authors mention the fact that their algorithm only works for orientable surface, I think that they do not define precisely enough the boundary Γ for example (is it a set or a parametrized curve?). To be fair, the authors declare that their method cannot work with non-orientable surfaces but that question should have deserved more precision.

References

[1] S. Wang and A. Chern, "Computing minimal surfaces with differential forms," *ACM Transactions on Graphics (TOG)*, vol. 40, no. 4, pp. 1–14, 2021.

[2] K. Crane, "Discrete differential geometry: An applied introduction," *Notices of the AMS, Communication*, vol. 1153, 2018.

[3] S. Wang, M. S. Nabizadeh, and A. Chern, "Exterior calculus in graphics," 2023.

[4] F. Morgan, *Geometric measure theory: a beginner's guide.* Academic press, 2016.

[5] L. Younes, *Shapes and diffeomorphisms.* Springer, 2010, vol. 171.

[6] H. Federer, *Geometric measure theory.* Springer, 2014.

[7] S. Boyd, N. Parikh, E. Chu, B. Peleato, J. Eckstein *et al.*, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Foundations and Trends® in Machine learning*, vol. 3, no. 1, pp. 1–122, 2011.

[8] T. Goldstein, B. O'Donoghue, S. Setzer, and R. Baraniuk, "Fast alternating direction optimization methods," *SIAM Journal on Imaging Sciences*, vol. 7, no. 3, pp. 1588–1623, 2014.

[9] M. Kazhdan, M. Bolitho, and H. Hoppe, "Poisson surface reconstruction," in *Proceedings of the fourth Eurographics symposium on Geometry processing*, vol. 7, 2006, p. 0.

[10] T. S. Newman and H. Yi, "A survey of the marching cubes algorithm," *Computers & Graphics*, vol. 30, no. 5, pp. 854–879, 2006.

[11] A. N. Hirani, *Discrete exterior calculus.* California Institute of Technology, 2003.