Eurographics Symposium on Geometry Processing 2015
Mirela Ben-Chen and Ligang Liu
(Guest Editors)

Volume 34 (2015), Number 5

Topology Optimization with Levelsets and Heightmaps

M. Tao[1]

[1]University of Toronto

Abstract

We propose a topology optimization method that utilizes a levelsets and heightfields in order to generate geometry that can be used to modela 3-axis CNC milling device.

Categories and Subject Descriptors (according to ACM CCS): I.3.3 [Computer Graphics]: Topology Optimization—Heightmaps,Levelsets

1. Introduction

Despite the current additive manufacturing craze, traditional manufacturing techniques are still the prevalent manufcaturing method industrial applications. This is likely because these traditional methods are able to produce artifacts from a wider variety of materials, with more consistent quality, and with significantly chaper cost. However, the modern trend of topology optimization pays little attention to the manufacturability, to the degree that the geometries designed are not manufacturable by any technique. We are interested in looking at one traditional manufacturing technique, three-axis CNC milling, which can only represent a fairly small class of geometries.

Different manufacturing methods have distinctly different costs and challenges associated with them. Additive manufacturing has become increasingly popular because the class of geometries its artifacts can generate are probably the largest among the manufacturing methods available. However, this comes at the cost of manufacturing speed and reliability. We will look at the task of 3-axis CNC milling, though there somewhat natural extensions that could be taken for more general milling apparatuses. Although this class of machining problems has a significant limitations on the sorts of geometry it can use, it is a relatively fast method for generating a fairly large class of geometries.

1.1. Design Through Optimization

Traditional design has required direct intervention from a human to direct the geometry and constitutive material of that geometry. As an alternative to this traditional perspective, by taking into account modern advances in computing power, one may choose to pose design is a complex multi-objective optimization task. The task of designing an artifact can be seen as the balancing of various criteria such as aesthetic quality, monetary cost, manufacturing cost, and functionality. Some criteria like costs or functional ability are easier to represent with a computer, while aesthetic quality can be ambiguous and challenging to represent programmatically.

For now we will ignore the aesthetic qualities and work on optimizing for quantities that are directly measurable. We choose to focus exclusively on functionality and material used. Because we are using homogeneous materials the financial cost will be taken to be approximately the amount of material so in a weak sense financial cost is also being considered.

Although we will only look at these objectives we will describe and discuss a general framework theoretically capable of handling multiple criteria simultaneously.

2. Heatsink Design

The problem we choose to tackle is the problem of designing heatsinks that can be milled by a 3-axis CNC milling device. These devices are made of materials with high thermal conductivity and attached to hot objects to spread the heat energy to a cool interface (such as air or water). The effectiveness of a heatsink is measured by its ability to transfer heat heat away from the system.

There's two boundaries: one that interfaces with the hot object and one that touches the cool interface. The objective is to create an artifact with high thermal conductivity such that heat can easily move from one interface to the other. The performance of a heatsink can therefore be measured by the amount of heat that can be transfered from one end to another:

$$\mathcal{E}_{\mathcal{T}}(\Omega) = \int_{\Gamma_{out}} \frac{\partial v}{\partial n} d\sigma = -\int_{\Gamma_{in}} \frac{\partial v}{\partial n} d\sigma$$

where v is the heat in the heatsink of geometry Ω and the cool interface is Γ_{out}.

A secondary objective is to create a light object by minimizing the contained energy. This is of course implemented by the integral

$$\mathcal{E}_{\ell}(\Omega) = \int_{\Omega} u^2 dx.$$

It would probably be better to minimize some other norm, such as ℓ_∞, but for now we will use this as an easy relaxation.

2.1. Thermodynamic Model

We want to model an equilibrium solution for heat transfer problem where one end has some hot temperature and the other end is cooler. We could say the cool end is air that is being replaced at some sufficient rate so its temperature can be treated as constant. We model the heat transfer within Ω as a standard Poisson problem, treat the hot component as a Dirichlet-boundary constrained component and a Robin-boundary constrained outer interface. The Robin boundary constraint represents energy being constantly removed from the system at some rate according to the temperature gradient. We choose a Dirichlet boundary constraint for simplicity and to clamp one side. This turns into equations of the form

$$\begin{aligned} \Delta u &= 0 \text{ in } \Omega \\ \frac{\partial u}{\partial n} &= \alpha(u - T_{out}) \text{ on } \Gamma_{out} \\ u &= T_{in} \text{ on } \Gamma_{in} \end{aligned}$$

which can be rewritten as

$$\begin{aligned} \Delta v &= 0 \text{ in } \Omega \\ \frac{\partial v}{\partial n} - \alpha v &= 0 \text{ on } \Gamma_{out} \\ v &= T_{in} - T_{out} \text{ on } \Gamma_{in} \end{aligned}$$

by $v = u - T_{out}$.

3. Representation

For a 3-axis CNC mill, the natural representation of a constructible object seems to be a heightfield sitting in a build volume. The intuition comes from the following:

When milling an artifact one usually starts with a solid rectangle of material and with a 3-axis mill one has a drill bit aimed in the $-\hat{k}$ direction and freedom to move in any axis. Because this drill bit cannot change its direction if it reaches a point (x,y,z) one immediately knows that $\forall \bar{z} > z$, material at $(x,y,\bar{z})$ has been removed as well. Therefore the geometry carved from this type of milling device can be seen as a function from (x,y) to z where z is the highest point where material remains.

Of course the paths made by a drill bit have some continuity as the bit is usually not infinitely thin. If one assumes that the drill bit forms a cylinder with a spherical cap of the same radius r when spun up, the geometry can be prescribed *exactly* by looking at the r-offset surface of the set of points the drill bit passes through. We do not incorporate this exactness in our method at this time, but it would be a trivial extension.

4. Topology Optimization

We utilize topology optimization via a methodology of computing *shape derivative*. This allows for us to derive vector fields that, when our geometry is advected by them, produces new, more optimal geometry. The shape derivative is computed by looking at the evaluation of a function through the space of small perturbations of a geometry. Small perturbations are generated by vector fields $\Theta \in \mathcal{T}_p^{\Omega}$ (in the sense that diffeomorphisms of the embedded space take vector fields as their lie algebra). As with diffeomorphisms,Θ can be computed by considering arbitrary transformations of the form

$$T_t = T(t,x) : [0,\varepsilon] \times \Omega \to \mathcal{B}$$

where Ω is the geometry and $\mathcal{B}$ is the embedding space. We also have $T_0 = Id$ and ε as some small positive value. Θ is therefore defined by

$$\Theta(p) = \lim_{t \to 0} \frac{T_t - Id}{t}(p).$$

This identifies vector fields with infinitessimal deformations of our geometry.

4.1. Optimizing Variations

Our approach is to improve a given geometry by finding an optimal variation of a given geometry, where each variation is represented by vector fields. The first step is to compute a function that expresses the sensitivity our objectives are to small perturbations of a particular geometry. First note that the behavior of the vector field on the interior does not matter in the case of homogeneous materials because this is simply replacing material with more of the same material.

We therefore only care about a vector field on the boundary of the surface. For a given perturbation/vector field we therefore need to compute some function Q that gets applied as

$$\frac{d}{d\varepsilon}\mathcal{E}(\Omega_\varepsilon)\Big|_{\varepsilon=0} = \Theta(\mathcal{E}(\Omega_\varepsilon)) = \int_{\partial\Omega} Q\Theta \cdot N dx.$$

Here Q called the speed function, and N the normal on $\partial\Omega$ lets us know that only Θ in the normal direction matters for the evolution of the geometry.

4.1.1. Some Necessary Conditions

First, we must find some necessary characteristics for optimizing our system, which we write out individually for each objective. To do this we will first describe an augmented Lagrangian for our problems that incorporates the geometry iteslf (Ω), and some slack variables (λ, μ).

For our thermal problem we see the following Lagrangian: We write out our entire system as the following Lagrangian:

$$\begin{aligned}\mathcal{L}_{\mathcal{T}}(v,\Omega,\lambda,\mu) = &-\int_{\Gamma_{out}} \alpha v d\sigma \\ &+\int_\Omega \langle \nabla v, \nabla\lambda\rangle dx - \int_{\Gamma_{out}} \alpha v\lambda d\sigma \\ &-\int_{\Gamma_{out}} \frac{\partial v}{\partial n}\lambda d\sigma + \int_{\Gamma_{in}} (v - T_{in} + T_{out})\mu d\sigma.\end{aligned}$$

The first term is the negative of what we would like to maximize, flow outward $\frac{\partial v}{\partial n}$, and the latter terms are the thermal proble definition.

From looking at variations in v and some term rearrangement one obtains

$$\begin{aligned}\frac{\delta\mathcal{L}_{\mathcal{T}}(v,\Omega,\lambda,\mu)}{\delta v}\delta v = &-\int_{\Gamma_{out}} \alpha(1-\lambda)\delta v d\sigma \\ &+\int_\Omega \Delta\lambda\delta v dx \\ &-\int_{\Gamma_{out}} \frac{\partial\lambda}{\partial n}\delta v d\sigma + \int_{\Gamma_{in}} \mu\delta v d\sigma.\end{aligned}$$

these computations imply the following values for the adjoint state of our problem, λ and μ:

$$\begin{aligned}\Delta\lambda &= 0 \text{ in } \Omega \\ \frac{\partial\lambda}{\partial n} - \alpha v &= -\alpha \text{ on } \Gamma_{out} \\ \frac{\partial\lambda}{\partial n} + \mu &= 0 \text{ on } \Gamma_{in} \\ \lambda &= 0 \text{ on } \Gamma_{in}.\end{aligned}$$

by using existing tools for computing shape derivatives we obtain from these conditions

$$\frac{d}{d\varepsilon}\mathcal{E}_{\mathcal{T}}(\Omega_\varepsilon)\Big|_{\varepsilon=0} = \int_{\Gamma_{out}} \left(\alpha^2 v(1-\lambda) + v\frac{\partial^2\lambda}{\partial n^2}\right)\Theta \cdot N dx.$$

Doing the same for the volume minimization results in

$$\frac{d}{d\varepsilon}\mathcal{E}_\ell(\Omega_\varepsilon)\Big|_{\varepsilon=0} = \int_{\Gamma_{out}} \left(u^2 - \frac{dp}{dn}\frac{du}{dn}\right)\Theta \cdot N dx$$

where p is an adjoint state defined by

$$\begin{aligned}\Delta p &= 2u \text{ in } \Omega \\ p &= 0 \text{ on } \partial\Omega.\end{aligned}$$

The repeated application of vector fields that minimize these energies produces an evolution of some initial geometry Ω_0 over a series of Ω_t, which one hopes to converge to a decent local minima. This convergence is defined by the the strength of Θ, which should become the 0 vector at a minima.

4.2. Optimizing Parameters

The behavior of the vector field on the interior does not matter because for a homogeneous material this is simply replacing material with more of the same material. We therefore only have to compute an objective function defined on, or near, the boundary of a geometry. If we assume only a certain amount of energy to advect our shape Ω_t, the optimal vector field comes in the form:

$$\text{argmax}_{\Theta\in T_p\Omega, \|\Theta\|=1} \int_{\partial\Omega} Q\Theta \cdot N dx$$

for a function Q called the speed function and N being the normal on $\partial\Omega$ at the point of evaluation. We only care about the normal direction because in an infinitessimal sense moving geometry in the tangential direction is just advecting material along the surface of the geomtry.

When our geometry is free to deform in any way it wishes, the optimal vector field is, up to a scaling, $\Theta = QN$. For instance, Θ is unconstrained when the geometry is represented by an implicit function. In this case the levelset ϕ would be advected by the vector field according to this Hamilton-Jacobi type equation:

$$\nabla\phi \cdot \Theta + \frac{\partial\phi}{\partial t} = 0.$$

However, in the constrained case this is not as obvious. In the constrained case there is a pullback required to compute a gradient with respect to the actual parameters. If we let γ be our map from parameters to geometry, the chain rule, tells us that

$$\Theta(x) = \frac{\partial\gamma}{\partial p}\theta$$

for some vector field θ over the parameters p. Under this notation we see that $\frac{\partial\gamma}{\partial p}$

$$\int_{\partial\Omega} QN \cdot \left(\frac{\partial\gamma}{\partial p}\theta\right) dx \qquad = \int_{\partial\Omega} Q\left(N \cdot \frac{\partial\gamma}{\partial p}\right)\theta dx.$$

Hence, we have a linear function that dictates how to our objectives change due to perturbations of the parameters p.

4.2.1. Optimizing a heightfield

Recall that we use a heightfield to parameterize our geometry. If we look at the heightfield as piecwise-constant we obtain a map from height values to geomtry as the union of rectangular prisms:

$$\bigcup_{i,j}[x_i-\frac{dx}{2},x_i+\frac{dx}{2}]\times[y_j-\frac{dy}{2},y_j+\frac{dy}{2}]\times[0,h(i,j)].$$

At our "sample" locations $(x_i,y_j,h(i,j))$ we see that the normal is certainly only nonzero in the z axis. We therefore utilize the following definition for Θ along the top of the heightfield:

$$\frac{\partial\gamma}{\partial p}\theta=\begin{bmatrix}0\\0\\1\end{bmatrix}\theta=N_z\theta.$$

If we plug this into our optimization procedure the integral turns into

$$\int_{\partial\Omega}QN_z\theta dx$$

which is mazimied by $\theta=Q$.

We would choose piecewise linear, but we did not have time to implement that much.

4.2.2. Splitting Parameters

Naively optimizing a heightfield is too restrictive. In regions where the heightfield has value of 0, because nothing is simulated there, there is no gradient for material to grow. Furthermore, as we saw above, the gradient it is only in the z axis. This implies that without some additional trick the support of the heightfield could only contract over time, which is undesirable (these methods tend to both shrink and grow in the unconstrained case).

We therefore choose to add a second representation to allow for the geometry to extend itself in the xy plane. For this we choose a levelset represntation for the boundary of the geometry on the xy plane. We imagine that the heightfield is only explicitly defined within the domain and use the z-axis gradients from within that domain.

When this levelset contracts the behavior is quite clear as material would simply disappear. When the levelset expands though, it is ambiguous how new material should be added. The easiest solution would be to, for a point (x,y) that is now inside the levelset, to let $h(x,y)$ be the value of h when (x,y) are projected onto the boundary of levelset for the previous iteration.

4.2.3. Optimizing the base

Due to time constraints this is unfinished. The concept was to use the the speed function to compute a vector field along x,y and use something like semilagrangian advection to advect height values. This would presumably result in something like

$$\frac{\partial\gamma}{\partial p}=\frac{1}{2}\begin{bmatrix}h(x_{i+1},y_j)-h(x_{i-1},y_j) & 0\\ 0 & h(x_i,y_{j+1})-h(x_i,y_{j-1})\\ 0 & 0\end{bmatrix}.$$

we didn't have time to look into this further though.

5. Implementation

So far we have posed the equations that we must solve, namely the pertinent speed functions and how to convert those into vector fields from which to advect our geometry. What remains is how to actually discretize and solve those equations as well as how to actually deform the geometry.

We implemented our computation of the speed function with piecewise-linear finite elements for simplicies (triangles/tetrahedra) but apply triangulations on cubes (in 3D one uses the Freudenthal subdivision of a cube into tetrahedra). The choice to stick to simplicial finite elements is purely for the simplicity of their shape functions. For the most part implementation of the speed function was straightforward except for the Robin boundary conditions.

In order to generate a voxelized representation of our heightfield we mapped the heightfield to a signed distance function and applied some of our existing infrastructure for computing stiffness matrices from levelset data. This allowed for us to first test our code on the "unconstrained" case of levelset data before projecting it to the heightfield representation. With this implementation we effectively treated the heightfield as a wrapper around a standard levelset-based method.

5.1. Robin Boundaries

The boundary required a bit of extra treament due to the Robin boundary constraints, which of course only exist on the boundary. We add a layer of ghost cells in order to satisfy the Dirichlet-esque part of hte Robin boundary constraint. This implies that we expanded our domain to being one cell thicker everywhere, where we solved for the original Robin boundary condition on the original boundary and applied a Dirichlet boundary constraint to the outside. Applying this allowed for us to weakly assert the external temperature T_{out} from an ambient temperature field rather than as a single value, though we never got to apply this.

We then added the use a smoothened boundary by the following smoothened dirac on a signed-distance function on our domain:

$$\frac{1}{\sqrt{\phi^2+\varepsilon^2}}-\frac{\phi^2}{\sqrt{\phi^2+\varepsilon^2}^3}.$$

The actual boundary constraint was computed from premade stiffness matrices and then restricted to the boundary using this dirac.

5.1.1. Derivatives on the boundary

Numerically computing $\frac{\partial^2 \lambda}{\partial n^2}$ with finite differences was initially a noisy part of our solution. However, we realized that we can directly incorporate our Robin boundary constraints to replace derivatives in the normal direction:

$$\frac{d\cdot}{dn} = \alpha(\cdot + C)$$

for some C. This of course applies to multiple derivatives in the normal direction, which appear as products by α. We therefore never have to explicitly compute derivatives in the normal direction due to our Robin boundary constraint..

5.2. Gradient Descent

We experimented with several different gradient descent-based algorithms. We initially simply did gradient descent

$$h_t = h_{t-1} + \alpha\theta_t,$$

but due to the noisiness of the results we were getting from a finite difference $\frac{\partial \cdot}{\partial N}$ implementation we sought out methods that could deal with noise. Our first attempt was to simply do grdient descent with momentum, which is simply

$$h_t = h_{t-1} + \alpha\mathbb{E}[\theta_t],$$

for $\mathbb{E}[x_t] = .9x_t + .1\mathbb{E}[x_{t-1}]$ a weighted average operator (of course .9 and .1 could be set to other values).

Furthermore we experimented with using an optimization technique called ADAM from the machine learning community. It has recently shown some good results and has its roots in proximal algorithms.

$$h_t^i = h_{t-1}^i + \alpha \frac{\mathbb{E}[\theta_t^i]}{\sqrt{\mathbb{E}[\left(\theta_t^i\right)^2] + \varepsilon}},$$

for some ε that prevents the denominator from being 0. There's some rescaling usually applied to remove bias, but they are not too important.

ADAM certainly seemed to converge faster, but its results were also notably different from SGD. It was also much easier to control than SGD because, the fractional part of ADAM naturally sticks to something near ± 1, so it was easy to bound the change in h.

5.3. Issues

Piecewise-linear finite elements has issues with boundary discontinuities such as notable corners in geometry or changes in boundary conditions. We observed this issue at the boundary of the heating element, so the shape derivatives, and therefore velocity fields, had spuriously high values at the pertinent interfaces. The magnitude of these high values increased with resolution in a nonconvergent fashion and were difficult to automatically remove through different clamping schemes. We wound up zeroing the shape derivative at locations such as at heating element boundaries or at thresholded values with an exponential decay at a sufficient distance away from those thresholded boundaries. Even though it was effective, training at different resolutions always required a bit of tuning because we had to clamp at different thresholds and change the evolution speed to compensate for larger values.

These Interestingly though, ADAM was effective ontop of hard clamping of the speed function because it applies normalization in a coefficient-wise fashion.

6. Conclusion

We have presented a topology optimization method that will only produce geometries manufacturable using a 3-axis CNC mills. This is performed by applying the theory of shape derivatives onto a novel parameterization, heightfields. We test our framework by deriving and implementing shape derivatives for our thermal problem through piecewise-linear finite elements. Finally, we performed some simple experimentation with different integrators that seemed to find phenomonologically different results.