A Subtractive Manufacturing Constraint for Level Set Topology Optimization

Nigel Morris, Adrian Butscher, Francesco Iorio

3 July 2019

We present a method for enforcing manufacturability constraints in generated parts such that they will be automatically ready for fabrication using a subtractive approach. We primarily target multi-axis CNC milling approaches but the method should generalize to other subtractive methods as well. To this end we take as user input: the radius of curvature of the tool bit, a coarse model of the tool head and optionally a set of milling directions. This allows us to enforce the following manufacturability conditions: 1) surface smoothness such that the radius of curvature of the part does not exceed the milling bit radius, 2) orientation such that every part of the surface to be milled is visible from at least one milling direction, 3) accessibility such that every surface patch can be reached by the tool bit without interference with the tool or head mount. We will show how to efficiently enforce the constraint during level set based topology optimization modifying the advection velocity such that at each iteration the topology optimization maintains a descent optimization direction and does not violate any of the manufacturability conditions. This approach models the actual subtractive process by carving away material accessible to the machine at each iteration until a local optimum is achieved.

1 Introduction

The aim of this work is to optimally synthesize the geometry of a mechanical part under a specified set of loading conditions and constraints such that the part can be successfully and accurately manufactured using a subtractive process. This process typically begins with a solid block of stock material and gradually removes material from the block until the remaining material has the shape of the designed part. One of the most common subtractive approaches is called "milling" and uses rotary cutters otherwise known as "tools," "end-mills" or "bits" to remove the material [4]. This process inherently limits the types of shapes that can be manufactured, since the milling machine must be able to hold the part rigidly and the rotary bit must be able to access the material surface without interference. Other important considerations must also be taken into account such as vibration of the part during material removal and stresses on the bit itself by the milling process. We focus the scope of this work on the problem of synthesizing optimal shapes whose surface is entirely accessible by a user-specified milling machine and tool.

The process of synthesizing optimal shapes given specified boundary conditions including loads and restraints is generally called *structural topology optimization* and there are two primary categories, density-based and boundary-based [13]. Density-based approaches discretize the volume of the part and assign a density to each discrete cell, then the densities are driven toward solid (1) and empty (0) while supporting the specified boundary conditions [3]. Boundary-based approaches instead track the shape of the external interface of the solid part and move this boundary incrementally towards optimality. In order to enforce the manufacturability constraint, having a precise knowledge of the boundary is advantageous so we make use of the latter approach. In addition, the method we employ uses an implicit representation known as a level set [11, 2] to track the boundary.

1.1 Related Work

In [9], the problem of manufacturability is tackled by a different approach. During topology optimization, a set of millable extrusions are fit to the geometry generated by the level set method and this approximate geometry is readily manufacturable. However the final optimized geometry without constraints in general cannot be represented by a set of extrusions and thus some level of optimality is lost in this approximation process.

In [14, 5, 7], topology optimization using a density-based method and a so-called 'projection' or 'visibility map' is used to constrain output to manufacturable parts. Manufacturable here is simply defined as every element in the density volume must have no occlusion in the milling direction. Occlusion means that there must be no density higher than the element's density in elements along the specified milling direction. Depending on the manufacturability technique, elements violating the constraint are either filled or removed. While this approach is convenient, unfortunately density-based methods do not converge to solid and empty elements until the later stages of the algorithm and so the constraint will not be physically accurate until this point of the optimization. In [14, 5] there is no explicit modeling of the subtractive bit or holder for milling manufacturability. In [7] the subtractive bit is modeled as a cylinder with hemi-spherical cap, but there is no constraint on the bit length or model of the holder. In [8] the 'projection' method was extended to include multi-axis milling constraints by using affine transformations to rotate the densities to align with the milling directions before performing an aggregated 'projection.' They also include a method for modeling the bit and holder, and model the effect of these on the optimized geometry. In this method the user must specify the set of possible milling directions from which material can be removed.

Our method works in a similar way to [17, 15] where a level set-based topology optimization is performed and the level set velocity function is constrained in order to enforce the manufacturing property. In contrast to [17, 15] which constrain the part to be manufactured with a two-sided cast, our manufacturing method is a more general subtractive constraint with a user defined tool from either a fixed set of directions or automatically determined 5-axis subtraction.

1.2 Contributions

Mill geometry. To our knowledge this is the first level set-based method to incorporate accessibility of a user defined physical tool bit and mill head.

Multi-axis constraint. We find the most accessible milling direction from a user supplied set or automatically choose the best milling direction to simulate 5-axis milling.

Compatible with a shape gradient-based formulation. We impose our milling constraint by modifying the shape gradients used in the level set-based topology optimization algorithm in such a way that a descent direction is maintained in every optimization iteration.

2 Definition of the Milling Constraint

We consider a tool bit (or end mill) and head that is defined using the following parameters: bit radius r and bit length b define the shape of the tool bit represented by a cylinder capped with a hemisphere oriented in the milling direction $\mathbf{m}$, and the head radius h defines another such capped cylinder for the tool holder and head. We assume that the end of the head cylinder extends

infinitely far away from the surface. Additionally, for 3-axis milling, we allow the user to specify a set of *milling directions* $\mathcal{M}$ from which the mill can approach the surface of the part. Whereas for 5-axis milling, we consider that the milling tool can be oriented to approach the surface from an arbitrary direction and our algorithm automatically chooses the milling direction. See Figure 1 for an illustration of the quantities defined above.

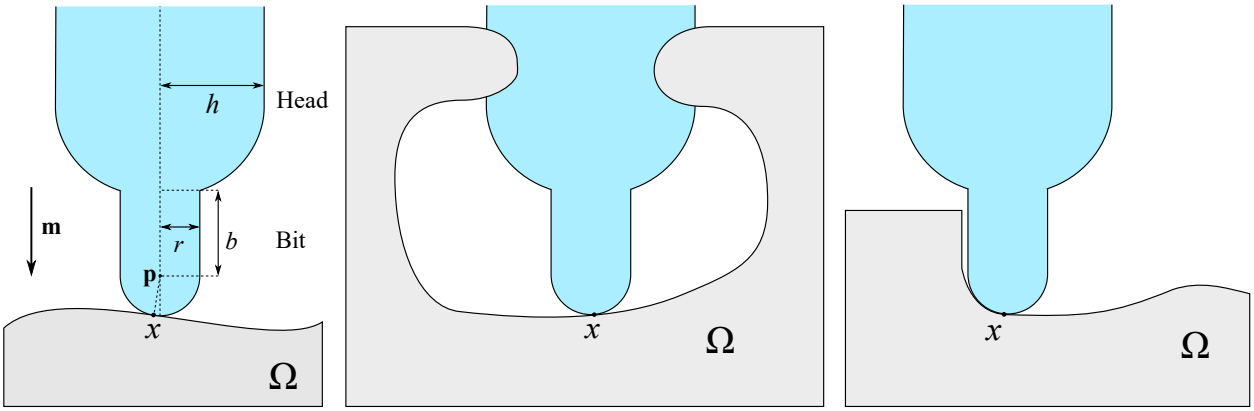


Figure 1: **Left:** Milling geometry definitions. **Middle:** Example of inaccessible partial cavity. **Right:** Example of a millable surface point.

For the purpose of this work, a part is millable if all its surface points are accessible in the sense that a tool bit as described above can be brought in from infinity along an allowed milling direction until it touches the surface point, making no intersection with the interior of the part.

Definition 1. Let Ω be a shape with boundary surface $\partial\Omega$. A point $x \in \partial\Omega$ is *millable* if there exists at least one milling direction for which the tool bit touches x and the tool bit and head as defined in the above text do not intersect with the interior of Ω. The boundary surface as a whole is millable if all its points are millable.

See Figure 1 for examples of millable and non-millable points.

3 Topology Optimization with Millability Constraints

3.1 Topology Optimization

Topology optimization problems can be formulated abstractly as the search for an optimal shape belonging to a class of *admissible shapes*, denoted herein by Adm. The specific notion of admissibility that we will use in this paper will be defined below (see Definition 2). The sought-after shape Ω solves the constrained optimization problem

$$\begin{aligned} \min_{\Omega \in Adm} \quad & \mathcal{F}(\Omega) \\ \text{s.t.} \quad & \mathcal{G}_i(\Omega) \leq 0 \quad \forall\, i = 1, \ldots, k \end{aligned} \tag{1}$$

in a suitable sense — namely, we content ourselves with a *feasible* (i.e. constraint-satisfying) *local minimum* of (1). Here $\mathcal{F}, \mathcal{G}_1, \ldots, \mathcal{G}_k : Adm \to \mathbb{R}$ are the objective and one or more inequality constraint functions. In structural topology optimization problems, at least one of these functions is formulated in terms of the solutions of linear elasto-static partial differential equations with

respect to one or more load cases. Each load case consists of a surface traction (a.k.a. Neumann boundary condition) applied to a subset of $\partial\Omega$ and a prescribed displacement (a.k.a. Dirichlet boundary condition) applied to another subset of $\partial\Omega$.

We will use level set-based topology optimization [2, 11] with the *augmented Lagrangian algorithm* [12] for solving the problem (1). This is an iterative strategy, whereby shapes are represented as the interior regions delimited by the zero-contour of piecewise smooth *level set functions* defined on an ambient design domain. The level set function for the shape at each iteration is updated in a manner that improves the optimization objective and reduces the constraint violation until a feasible, locally optimal admissible shape is achieved. Specifically, the update is performed by integrating a well-chosen *boundary normal speed function* for a small pseudo-time in each iteration by means of the standard level set Hamilton-Jacobi equation. (This update procedure is known as *advection* with respect to the chosen boundary normal speed.) In our case, the speed function on the boundary of the shape in the current iteration equals the negative *shape gradient* of the *augmented Lagrangian* which is a well-chosen algebraic combination of the objective and constraint functions. The speed function is then extended to a narrow band of the boundary by insisting that the extension be constant along the normal direction (to achieve this we solve the normal extension Hamilton-Jacobi equation for a time proportional to the width of the band). If $\mathcal{L}$ denotes the augmented Lagrangian and $d\mathcal{L}$ denotes its shape gradient at Ω, then we now have a speed function $v : \mathbb{R}^3 \to \mathbb{R}$ such that

$$v(x) = -d\mathcal{L}(x) \tag{2}$$

for all x belonging to the boundary $\partial\Omega$ of the geometry Ω. Additionally, we set v to zero on any part of the boundary of Ω we wish to hold fixed, such as the surface patches where the non-zero surface loads and prescribed displacements of each load case are applied. The chosen speed function is a *descent direction* for the augmented Lagrangian since it can be shown [6] that

$$\mathcal{L}(\Omega_\varepsilon) = \mathcal{L}(\Omega) + \varepsilon \int_{\partial\Omega} v \, d\mathcal{L} + \mathcal{O}(\varepsilon^2) = \mathcal{L}(\Omega) - \varepsilon \int_\Gamma \left(d\mathcal{L}\right)^2 + \mathcal{O}(\varepsilon^2)\,, \tag{3}$$

where Ω_ε is the advected geometry at the pseudo-time ε and Γ is the subset of $\partial\Omega$ that is allowed to move. This formula implies that $\mathcal{L}(\Omega_\varepsilon) < \mathcal{L}(\Omega)$ for sufficiently small ε. After iteratively applying this procedure and also updating the penalty coefficients and the Lagrange multipliers used to define the augmented Lagrangian, the shape converges to a local solution of (1).

3.2 Admissible Shapes

We wish to perform level set-based topology optimization, but we also wish to ensure that the resulting shape is millable as defined in Section 2. We achieve this by incorporating millability into the definition of admissible shapes. Thus in our setting, a shape is deemed admissible if the following definition holds.

Definition 2. A shape Ω is *admissible* if these requirements are met:

- Ω is the interior region delimited by the zero contour of a piecewise smooth level set function;
- $\partial\Omega$ contains all subsets where Dirichlet and Neumann boundary conditions from structural load cases are applied;
- Ω wholly contained in a specified design domain;
- $\partial\Omega$ is *millable* according to Definition 1.

3.3 The Archetypical Topology Optimization Problem for Millable Shapes

The archetypical structural topology optimization problem, namely *volume-constrained compliance minimization*, defines the objective function $\mathcal{F}$ as the average *compliance* of Ω with respect to at least one load case, and imposes exactly one inequality constraint: namely that the volume of Ω must be less than or equal to some fraction of the volume of the design domain. We will tackle this topology optimization problem in the present work, but with an additional millability constraint. That is, we require the optimized shape to belong to the space of admissible shapes defined in Definition 2 above, ensuring that the optimized shape is millable as described in Section 3.4. In other words, we solve the problem:

$$\begin{aligned} \min_{\Omega \in Adm} \quad & MeanCompliance(\Omega) \\ \text{s.t.} \quad & Volume(\Omega) \leq V_0 \end{aligned} \tag{4}$$

where $\Omega \in Adm$ if and only if Ω satisfies Definition 2. Here

$$MeanCompliance(\Omega) := \sum_{\ell=1}^{L} \frac{1}{L} \int_{\Omega} \sigma_{\Omega}^{(\ell)} : e_{\Omega}^{(\ell)} \, .$$

where $\sigma_{\Omega}^{(\ell)}$ and $e_{\Omega}^{(\ell)}$ are the stress and strain tensors in Ω induced by the ℓ^{th} structural load case.

3.4 Maintaining Admissibility in Level-Set Based Topology Optimization

Thanks to the admissibility criteria of Definition 2, the algorithm outlined in the Section 3.1 must be modified slightly before it can be applied to solve the optimization problem (4). This is because the choice of speed function (2) is no longer suitable as it can violate admissibility. For example, suppose that advection with respect to $-d\mathcal{L}$ causes a part of the boundary of the shape at the next iteration to become inaccessible with respect to the specified milling direction through occlusion with another part of the boundary of the shape. To rectify this issue, we propose in this paper a strategy for modifying the speed function (2) so that the shape remains millable from one iteration to the next while still ensuring that the Lagrangian decreases.

We are inspired by the classical *gradient projection method* [12] in which the outward-pointing components of the descent direction orthogonal to the active constraint boundary are set to zero in order to prevent violation of the constraints in those components. We adapt this method to level set-based shape optimization as follows. We first assume that our initial shape does not violate the millability condition, which is easily achieved by initializing with a convex hull of the input geometry for example. Next, in each iteration we modify the speed function at each point of the boundary of the shape by smoothly transitioning it to zero when a violation of millability is predicted in the next iteration. The goal is to force the boundary velocity to vanish where a violation of millability is predicted, thereby halting any movement of the boundary there.

To be precise, we introduce a "filter" function η (defined precisely in Section 3.5 below) that assigns a value in $[0, 1]$ to each point of the boundary. It assigns the value zero to any point belonging to the subset $\Gamma \subseteq \partial\Omega$ where a violation of millability is detected. Its values then transition smoothly from 0 to 1 in a small "collar" around the periphery of Γ, and it assigns the value one to all other points of $\partial\Omega$. To obtain the modified speed function, we *multiply* the original speed function by η. In other words,

$$v_{modified} := -\eta d\mathcal{L} \, .$$

It is important to realize that, since η is everywhere positive, then the modified boundary speed remains a descent direction for the Lagrangian. This is easily seen by substituting $v_{modified}$ for v in formula (3). Therefore we reduce the value of the Lagrangian during each iteration of topology optimization while only allowing the shape to be modified from directions accessible by the milling tool during that iteration.

3.5 Definition of the Filter

We now define the function η precisely. Note that the velocity of a point $x \in \partial\Omega$ has the form $\mathbf{v}(x) = v(x)\mathbf{n}(x)$, where $\mathbf{n}(x)$ is the outward-pointing unit normal vector field at x. If $v(x) > 0$ then the boundary of the shape grows near x; if $v(x) < 0$ then the boundary of the our first requirement for the definition of η is that $\eta(x)v(x) \leq 0$ for all x on the boundary of the shape, i.e. preventing growth, as this is a means to avoid occlusion of previously accessible regions of the surface of the shape. Next we choose the "best" milling direction from amongst all "accessible" ones, where a direction is accessible if there is no intersection between the shape and the tool bit or head (see Figure 1), and best means closest to the surface normal.

To state the above considerations rigorously, let $V_i(x)$ denote the volume occupied by the union of the milling tool and head when it is oriented along the milling direction $\mathbf{m}_i$ and the tip of the head is located at x. Then we define η as follows:

$$\eta(x) = \begin{cases} 0 & \text{if } v(x) \geq 0 \\ \max\{|\mathbf{m}_i \cdot \mathbf{n}(x)| : \text{all } \mathbf{m}_i \in \mathcal{M} \text{ s.t. } V_i(x) \cap \Omega = \varnothing\} & \text{otherwise.} \end{cases} \tag{5}$$

The initial set of milling directions $\mathcal{M}$ is either specified by the user depending on their desired machine configuration; or for 5-axis milling, we explore several methods in the following section.

3.6 Convergence Properties

The fact that our algorithm chooses a descent direction for the augmented Lagrangian means we can ensure that the value of the augmented Lagrangian decreases in every iteration of each inner loop of the augmented Lagrangian algorithm (assuming the advection time is chosen with a backtracking line search, as we do). This fact alone certainly does not guarantee the convergence of the sequence of shapes to a locally optimal shape satisfying the constraints; and a rigorous analysis of the convergence properties of our algorithm is beyond the scope of this paper (and would undoubtedly be quite difficult). However, we do observe in all our examples that the norm of the filtered shape gradient of the augmented Lagrangian decreases to zero within acceptable tolerances in each inner loop of the augmented Lagrangian algorithm. We also observe that the volume fraction constraint violation decreases to zero within acceptable tolerances over the course of the algorithm, and the millability constraint is always satisfied by construction. See Figure 2 for a graph of the relevant data for the augmented Lagrangian algorithm as a function of iteration (i.e. augmented Lagrangian value, volume fraction, compliance, Lagrange multiplier, penalty ceofficient) typical of our algorithm.

It is well-known that gradient-based optimization algorithms applied to highly non-convex problems such as topology optimization have a tendency to get stuck in local minima. Our algorithm is no exception. In fact, it may perform less well than conventional level set-based topology optimization from this point of view because it is likely that the "projected" nature of our algorithm

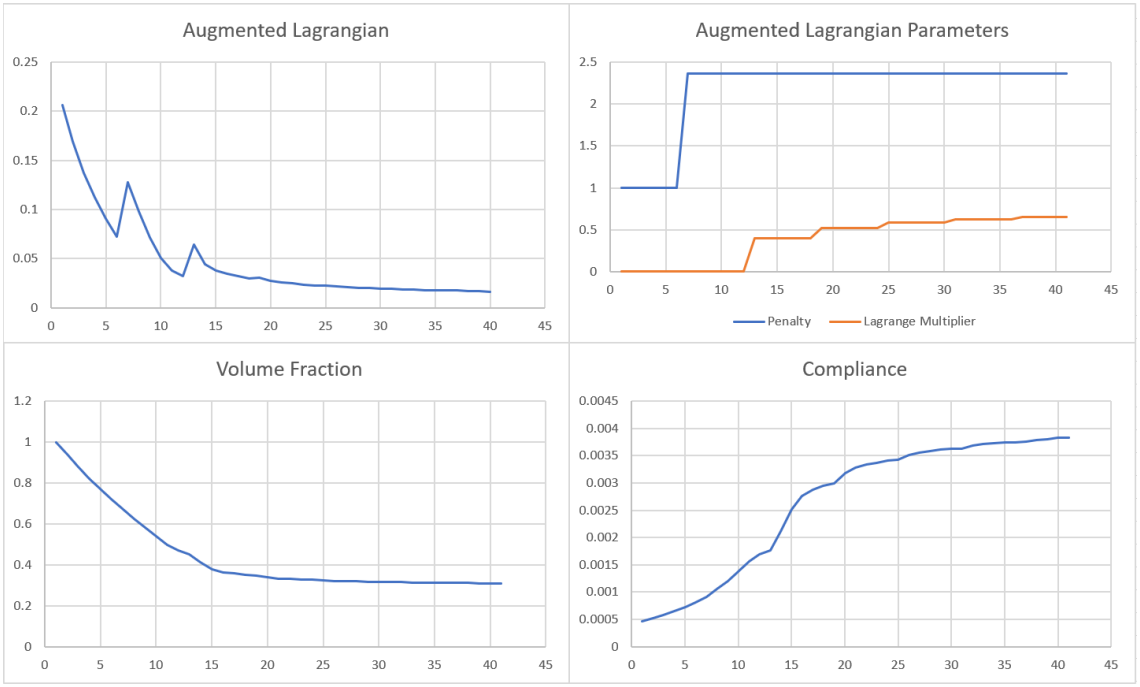


Figure 2: Optimization algorithm data as a function of iteration for the **TorqueStruct** example (see Section 5.1 for problem definition). The relative rate of change of the augmented Lagrangian and the volume fraction constraint violation in the final iteration are approaching 1%.

may cause it to stop prematurely at an undesirable locally optimal shape for which there exists a nearby millable shape with better performance that could easily be reached using an un-filtered descent step but not a filtered descent step. This is a limitation of our algorithm. However, the shapes that we can reach using our algorithm are nevertheless quite acceptable.

4 Implementation

The interesting implementation challenge is to compute for any choice of i and x the sets $V_i(x) \cap \Omega$ appearing in the milling constraint defined in Section 3. Conveniently, we can use a level set representation for $V_i(x)$ and then take advantage of a collection of level set operations that together lead to a very efficient algorithm. In addition this algorithm can be run in parallel to take advantage of modern multi-core architectures.

We implemented the volume-constrained compliance minimization problem using the level set-based topology optimization method outlined in [2] but with the modification of the boundary speed function described in the previous section. We used the level set library from [10] that creates signed distance functions and has implementations for the following functions: ray-casting, computing offsets, advection, and morphological closure. For completeness, these are given as Algorithms 1 through 4 in the Appendix.

4.1 The Millability Filter in the Three-Axis Case

The algorithm used to compute η for 3-axis machines is as follows: We find a set of points x of the boundary of Ω by projecting grid points in the narrow-band of ϕ onto the zero level set along $\nabla\phi$.

Then for each x, we loop over each milling direction and test for accessibility. The accessibility test combines two ray casting operations. The first tests against the bit intersection with the part and the second tests against mill head intersection with the part. One can see from Figure 3 that a ray originating at $\mathbf{p}$ in direction $-\mathbf{m}$ will only intersect the iso-surface of Ω offset by r if the bit will intersect with the part (i.e. $V_{bit} \cap \Omega \neq \varnothing$ where V_{bit} denotes the volume occupied by the bit). A similar test is also performed for the mill head except the iso-surface at h is used for the ray cast instead and the ray origin is $\mathbf{p} - \mathbf{m}_i(b + h)$. Combining these two tests we can easily determine $V_i(x) \cap \Omega$. Note that when using a narrow-band representation for Ω, the narrow-band must be extended to include iso-surfaces offset up to h. This process can be performed once during initialization using a standard level set offset function. Pseudo code for the above can be found in the appendix as Algorithm 6 and Algorithm 5.

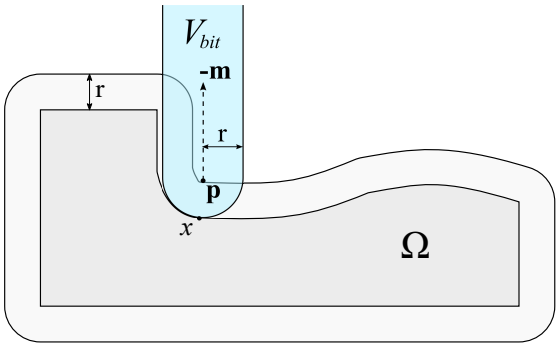


Figure 3: Example showing that a ray casting test from $\mathbf{p}$ in $-\mathbf{m}$ against Ω offset by r is equivalent to testing whether $V_{bit} \cap \Omega \neq \varnothing$

4.2 The Millability Filter in the Five-Axis Case

In the following sections we explore several different methods for obtaining the most accessible milling direction for a 5-axis machine. All of the following methods begin by testing the milling direction opposite to the surface normal $\mathbf{m}_0 = -\mathbf{n}$. This direction would provide the maximal value for $\eta(x)$. But it is possible for the surface point x being tested to be inaccessible from this direction, yet accessible from another direction. To find another direction, we propose several methods. All our methods are essentially searches on the hemisphere above x, with ever-more refined search strategies.

Hemisphere sampling. In this method we subsample the hemisphere to obtain a discrete set of directions $\mathcal{M}$ and test each to find the most accessible direction. In our implementation we used a fixed set of 26 samples on the sphere to form $\mathcal{M}$. If we consider the center of a cube, the milling directions point from the cube center to the center of every face, the midpoints of all the edges and the cube corners (See Figure 4). These directions were all normalized. We used the 3-axis algorithm (Algorithm 6) with this $\mathcal{M}$ to approximate a 5-axis procedure, and we note that in this procedure the accessibility test automatically filters $\mathcal{M}$ to those in the hemisphere such that $\mathbf{n}(x) \cdot \mathbf{m}_i < 0$ (Algorithm 5).

This method has several drawbacks: first, all directions in $\mathcal{M}$ must be tested to determine the most accessible, second it is possible that all of the directions are inaccessible but a small

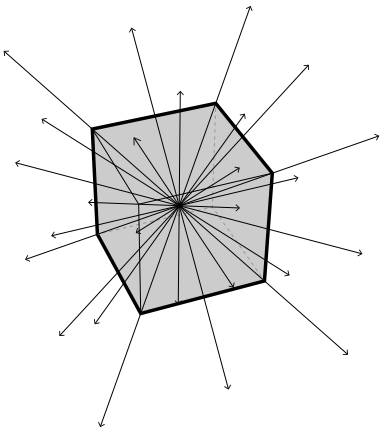


Figure 4: The set of milling directions $\mathcal{M}$ used in our implementation of 'Hemisphere sampling'.

perturbation of one of the directions could be actually accessible. The advantage of this method is that it does not just perform a local search of part of the sphere and will not be trapped by concavities.

Normal search. A dense sampling of the hemisphere is not practical so in this method we rely on the signed distance field ϕ around Ω to guide a local search of the hemisphere. If a milling direction is tested and fails, the intersection hit-point x_{hit} from the ray-trace, along with an offset is used to determine a new milling direction. This is done by moving from x_{hit} in the direction of $\nabla\phi(x_{hit})$ until either the medial axis of $\partial\Omega$ or the periphery of the level set narrow band is reached — determined by finding either a peak or a plateau along the line emanating from x_{hit} in the direction $\nabla\phi(x_{hit})$, respectively. A peak is a point y where the signed distance ceases increasing beyond y in the search direction, while a plateau consists of points y such that ϕ is constant near y. This new direction is in turn tested and updated until an accessible direction is found or an iteration limit is reached (see Figure 5). The advantage of this method is that it is a local search and often will quickly find an accessible direction in very few iterations (typically fewer than a coarse sampling of the whole hemisphere). The disadvantage of this method is that it is not guaranteed to find a solution even if one exists. It is possible for the method to move away from the accessible direction in some corner cases (see Figure 6). Details are shown in Algorithm 7.

Heat search. This method is an extension to the 'Normal search' method that handles the drawback of moving away from accessible directions. Instead of following the surface normal at the intersection point, we follow the direction of heat diffusion away from the surface toward the bounding box of the geometry. This requires an initial step of solving the transient heat equation to an approximate steady-state in the volume between the surface of Ω^+ (Ω offset by r) and the bounding box [16]. For this, we voxelize this domain and use a finite difference method to discretize the spatial derivatives as well as a first-order Euler method with sufficiently small time-step ε to discretize the time derivatives. We set two Dirichlet boundary conditions: on the vertices of the boundary of the voxel grid adjacent to Ω^+ we set the temperature to zero; and on the vertices of the bounding box we set the temperature to one. Once we have the temperature field T on the voxel grid, we integrate trajectories tangent to ∇T using sub-voxel multi-linear interpolation of T

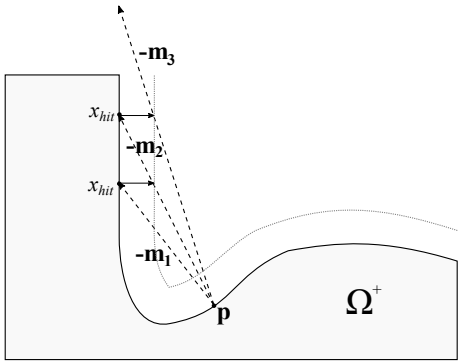


Figure 5: Example showing a successful result from the 'Normal Search' method for determining an accessible milling direction. The search algorithm begins with $\mathbf{m}_1 = -\mathbf{n}$ and performs a ray-trace on Ω^+, which is Ω offset by the bit or head radius. The intersection x_{hit} is found and a new milling direction $\mathbf{m}_2$ is found by offsetting from Ω^+ in the normal direction up to the periphery of the narrow-band (shown as dotted gray line). Direction $\mathbf{m}_2$ is then tested with another ray-trace, and in the same way leads to $\mathbf{m}_3$ which finally results in no intersection with Ω^+ and this direction is returned as a candidate milling direction.

together with a first-oder Euler method. Thus the gradient of the temperature field guides the search for the milling direction (See Figure 7). Note that it is important to solve the heat equation between the offset surface Ω^+ and the bounding box and not Ω and the bounding box since the offset surface will close off some exits from cavities that are narrower than the bit diameter.

4.3 Finite Element Analysis

In order to compute the mean compliance of a shape Ω and its shape gradient, we must solve the linear elasto-static equations in Ω with respect to one or more load cases. To this end, we use a finite element-based structural solver (Autodesk NASTRAN) in a discretization of Ω into elements according to the following procedure.

The level set function defining the shape at each iteration is defined on a uniform, background Cartesian grid. We do not use this grid for the the structural solver. Instead, in the first iteration we define a co-located conforming tetrahedral mesh using the meshing capabilities of the solver (essentially this performs a standard level set to mesh conversion on the level set function defining the initial shape). In the first iteration, we use this mesh to compute the finite element solution. In subsequent iterations, when the shape no longer conforms to this mesh, we nevertheless use an "active" subset of this mesh for the finite element solution. To be precise, we use the values of the level set function to detect which tetrahedra are fully inside the shape, which are cut by the boundary of the shape, and which are fully outside the shape. We then solve the discrete linear elasto-static equations only in the interior and cut tetrahedra. The stiffness matrix for the union of these tetrahedra is assembled from the interior tetrahedra in the usual way, and by multiplying the contribution from each cut tetrahedron with its volume fraction of intersection with the shape (and thresholded by a small value to avoid ill-conditioning). We use linear Lagrange shape functions inside these elements.

The load vectors corresponding to each load case can always be computed because the surface

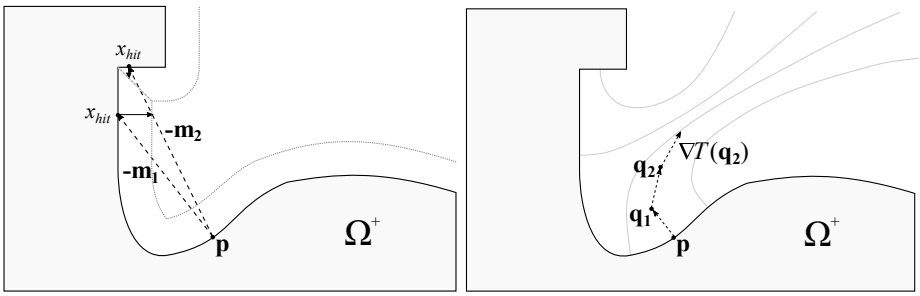


Figure 6: **Left:** Example showing an unsuccessful result from the 'Normal Search' method for determining an accessible milling direction and **Right:** a successful result from 'Heat Search'. The 'Normal Search' follows milling directions $-\mathbf{m}_1$, intersects at x_{hit} and offsets to the narrow-band to find $-\mathbf{m}_2$. When following $-\mathbf{m}_2$ a new intersection is found but the new offset does not escape from the overhanging geometry and is unable to find an accessible direction even though one exists. The 'Heat Search' follows a trajectory through the gradient of the heat $\nabla T(\mathbf{q})$ which quickly finds an accessible direction. Other trajectories are shown in light gray illustrate ∇T.

patches where the surface loads and prescribed displacements of each load case are applied belong to the evolving shape at every iteration. This is because we set the speed function governing the shape updates to zero on these surface patches. Therefore, since the initial mesh contains a triangulation of these surface patches, the subset of the mesh used for the finite element solution at every subsequent iteration does as well.

Once the solutions to the discrete linear elasto-static equations have been computed, we construct their per-element strain energy densities (the negative shape gradient of the compliance is the strain energy density) in the boundary tetrahedra of the active subset of the mesh. The negative shape gradient of the augmented Lagrangian at the current iteration is a linear combination of the sum of the strain energy densities and the negative shape gradient of the volume (equal to -1). We assign to the grid nodes nearest the boundary of the shape at the current iteration the value of the negative shape gradient in the nearest boundary tetrahedron. We then extend these values to a narrow band of the boundary of the shape using the Hamilton-Jacobi equation for normal extension described in Section 3.

4.4 Shape Update Strategies

Finally we conclude with some implementation details concerning the way in which the shape is updated and how the milling constraint fits into it. We have two variants: an algorithm that assumes initial millability and ensures millability in each subsequent iteration as described in Section 3; and an algorithm that initially relaxes the millability constraint and enforces it only at convergence.

Strict algorithm. At each iteration the negative shape gradient of the augmented Lagrangian is evaluated with finite element analysis and extended to a narrow band of the shape boundary as described in the previous section. We obtain the advection speed function by multiplying the extended negative shape gradient with the milling filter η as described in Section 3. Positive speeds that could allow for shape growth are removed in this way. The shape Ω is then updated by level set advection with this speed function. This algorithm ensures that we only remove accessible

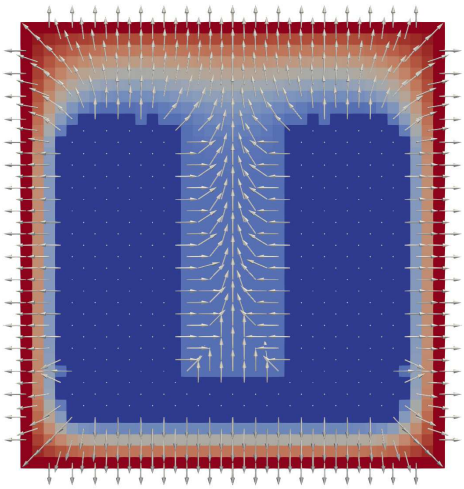


Figure 7: Example showing the heat field T for a slice through the center of **TorqueStruct** (see Figure 8 right). An arrow plot illustrates the gradient of the heat field ∇T and shows how trajectories in this field lead from the surface of Ω^+ (blue) to the accessible boundary (red).

material at each step and it requires that the initial shape of Ω be fully accessible (a feasible start). Convergence can be tested when either the speed approaches zero or the change in objective function is below a threshold. The final step performs the morphological closing of Ω, which explicitly enforces fillets and fills any cavities smaller than the tool radius r that have been created. The result of this operation is a very minor cleaning of the level set since the fillets and curvature of cavities are already implicitly controlled by the filter function η. See the details in Algorithm 9.

Relaxed algorithm. A more relaxed version of the above algorithm allows for positive values in the speed function so that the algorithm can begin with an infeasible starting condition before eventually converging to a feasible solution. In order to converge to a feasible solution we apply a positive velocity whenever we detect inaccessible regions (i.e. points where $\eta = 0$) thus filling in cavities or otherwise inaccessible regions of Ω. Values between $\alpha = 0.1$ and $\alpha = 0.25$ work well although there is a trade-off between speed of correcting inaccessible regions with interfering with the subtractive milling. When α is too large, surface regions that were inaccessible may grow too much and this overshooting will need to be compensated with material removal in subsequent iterations. See Algorithm 10 for details.

5 Results

5.1 Problem setups

SupportStruct. This problem setup consists of a plate supported from four foot plates (See Figure 8). We used a single load case with a vertical mechanical load of 3KN applied to the top surface of the top plate, while the four lower plates have fixed Dirichlet boundary conditions on their bottom surfaces. The plates and material are modelled as uniform in density with a Poisson ratio of 0.29 and Young's modulus of 17 GPa. In this problem we also define a pair of symmetry planes and enforce mirror symmetry in the output geometry across the planes. The dimensions of

the bounding box of the geometry are 50mm × 50mm × 50mm.

TorqueStruct. This problem setup consists of a set of four plates supported with a set of four loads creating a twisting load and supported by a fixed plate below (See Figure 8). We also use a single load case and each loaded plate receives a mechanical load of 50N applied to its top surface and the lower plate has a fixed Dirichlet boundary condition on its bottom surface. The plates and material are modelled as uniform in density with a Poisson ratio of 0.3 and Young's modulus of 210 GPa. The dimensions of the bounding box of the geometry is 60mm × 63mm × 60mm.

SkateTruck. This problem setup consists of the support structure for the axle and connecting to the body of a skateboard. This case is provided as an illustration for the subtractive constraint applied to a more complex real-world problem. It consists of 6 load cases, one is illustrated in Figure 9. The material is Aluminum, modelled as uniform in density with a Poisson ratio of 0.33 and Young's modulus of 68.9 GPa. The dimensions of the bounding box of the geometry are 188mm × 44mm × 151mm.

Upright. This problem setup consists of the upright in a suspension system of a performance automobile. This case is also provided as an illustration of the constraint applied to a more complex real-world problem. One of the 5 load cases is illustrated in Figure 9. The material is Aluminum, modelled as uniform in density with a Poisson ratio of 0.33 and Young's modulus of 68.9 GPa. The dimensions of the bounding box of the geometry are 188mm × 44mm × 151mm.

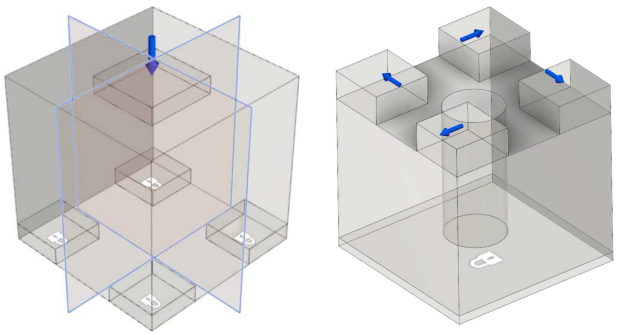


Figure 8: **Left**: The problem setup for the **SupportStruct**: The design space is a box enclosing the top plate that is loaded from above and supported by four plates below with Dirichlet boundary conditions on their bottom faces (indicated by the lock symbols). Two symmetry planes cut through the design space. **Right**: The problem setup for the **TorqueStruct**: The design space is the union of the shown boxes with a central cylindrical column subtracted away. The single load case is shown where the top plates are loaded with a twisting type load. The bottom plate is fixed with a Dirichlet boundary conditions on its bottom face.

5.2 Optimization algorithm

In our experiments on the **SupportStruct** and **TorqueStruct** we ran level set topology optimization using the Augmented Lagrangian method described in Section 3 but with boundary speed

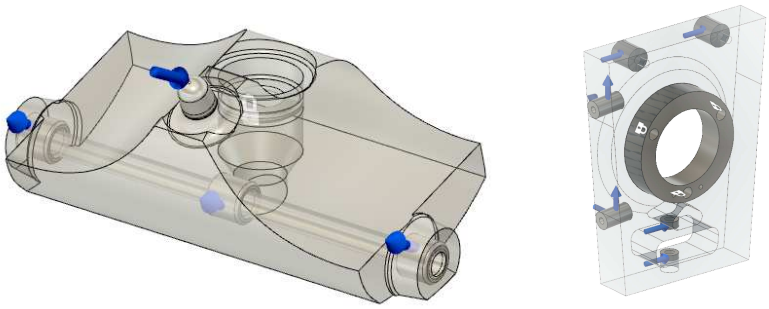


Figure 9: One of the load cases used to define the problem along with the design space for **Left**: **SkateTruck**. **Right**: **Upright**.

modification as described in Section 3. We minimize compliance subject to a volume fraction constraint of 20 and 30 percent of the initial design space respectively.

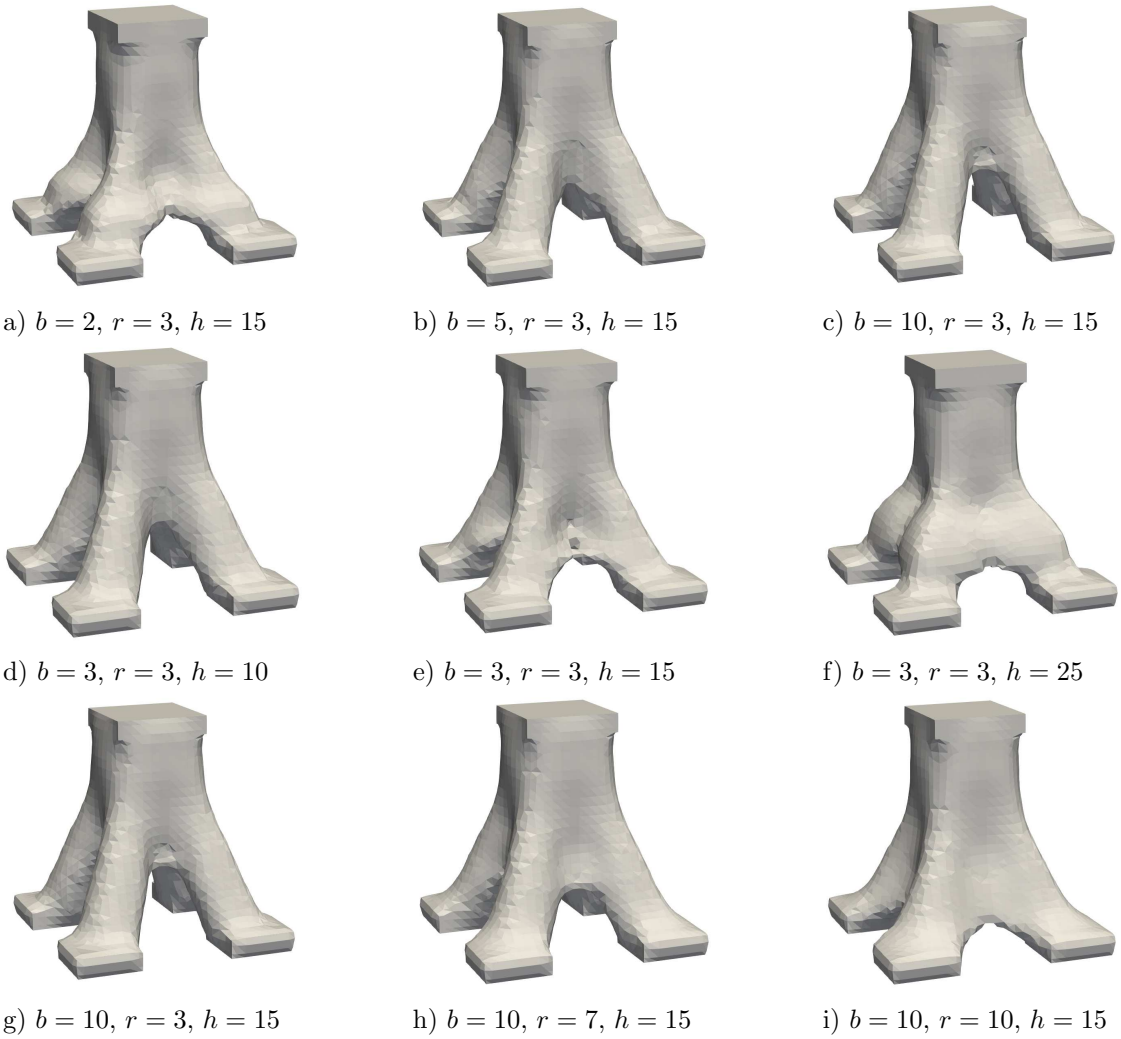
In our experiments on the **SkateTruck** and **Upright** we ran level set topology optimization using a proprietary method, with the objective of minimizing compliance subject to a volume fraction constraint of 30 percent of the initial design space, but now including a stress constraint. The proprietary aspect of the algorithm is how the stress constraints are handled. Our intention with this experiment is to highlight the versatility of our millability filtering approach — it can be applied to any shape update scheme to produce millable results. Indeed, the proprietary method assembles the boundary speed from the shape gradients of the objective and constraint functions in a different manner than the "standard" method described in Section 3. Nevertheless, when we modify this algorithm's boundary speed exactly as described in Section 3, we obtain the expected outcome that the Lagrangian still decreases for sufficiently small shape updates while the millability of the updated shape is maintained.

In all the experiments, the geometry with loads and fixity applied are all considered preserved regions and at each step in the optimization forced to remain part of the optimized geometry via boolean union operations. In all the examples, the relaxed algorithm was used to allow for recovery from infeasible states with $\alpha = 0.25$.

5.3 Parameter experiments

We ran experiments to test the effect of the various milling tool parameters on the **SupportStruct** problem. Figure 10 (a-c) shows how the output varies as we modify the bit length. The access to the bottom cavity becomes reduced with shorter bits resulting in a squatter result (a). Figure 10 (d-f) shows how changing the head radius affects the result. In f) the effect is most noticeable where a large head prevents access to the geometry above the fixed plates and again squat result is obtained. In Figure 10 (g-i) the tool radius is varied and so the curvature of the resulting geometry is reduced as the tool increases in radius, giving a smoother result with fewer divots.

We tested some variations of milling directions for 3-axis setups as shown in Figure 11. These illustrate how the preserved regions prevent removal of material that is inaccessible from the specified milling directions. We found that in general the milling constraint did not hinder the volume constraint satisfaction as long as the parameters of the bit allowed for enough material to be removed. For instance, the example in Figure 11 right could not remove enough material to meet the



a) $b = 2$, $r = 3$, $h = 15$ b) $b = 5$, $r = 3$, $h = 15$ c) $b = 10$, $r = 3$, $h = 15$

d) $b = 3$, $r = 3$, $h = 10$ e) $b = 3$, $r = 3$, $h = 15$ f) $b = 3$, $r = 3$, $h = 25$

g) $b = 10$, $r = 3$, $h = 15$ h) $b = 10$, $r = 7$, $h = 15$ i) $b = 10$, $r = 10$, $h = 15$

Figure 10: Optimized parts with 3-axis milling using $\mathcal{M}$: (+X,-X,+Y,-Y,+Z,-Z) and varying tool and head parameters.

volume target unless we specified a long enough bit length.

5.4 Algorithm experiments

We evaluated the performance of the 5-axis search algorithms on the **TorqueStruct** example since the design space has a deep cavity, it serves as an illustration of the different η search algorithms (see Table 5.4). Figure 12 shows the output after 40 iterations for each of the 5-axis search algorithms in section 4 as well as an unconstrained topology optimization result. Each method converged to the target volume fraction, however the relative compliance and the geometry varied. As expected, the unconstrained result had the lowest compliance, but of the constrained versions the 'Heat search' was able to identify surface elements deep inside the cylindrical cavity as accessible, while the 'Normal search' was only able to remove material from the top and gradually opened the central cavity. The 'Hemisphere search' also was not able to widen the cavity and resulted in the highest compliance.

In terms of performance, the 'Normal search' had least impact on computation time as it does

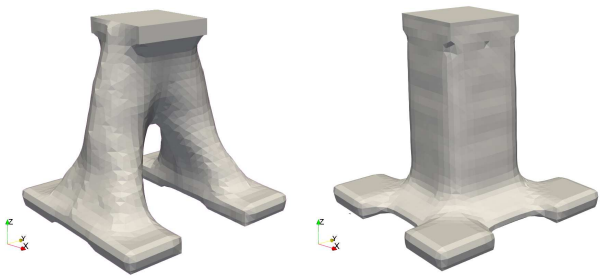


Figure 11: Optimized parts with 3-axis milling with varying milling directions. **Left:** $b = 20$, $r = 3$, $h = 15$, $\mathcal{M}$: (+X,-X). **Right:** $b = 50$, $r = 3$, $h = 15$, $\mathcal{M}$: (-Z)

not need the heat equation solve and the 'Hemisphere' method samples all 26 directions for each point.

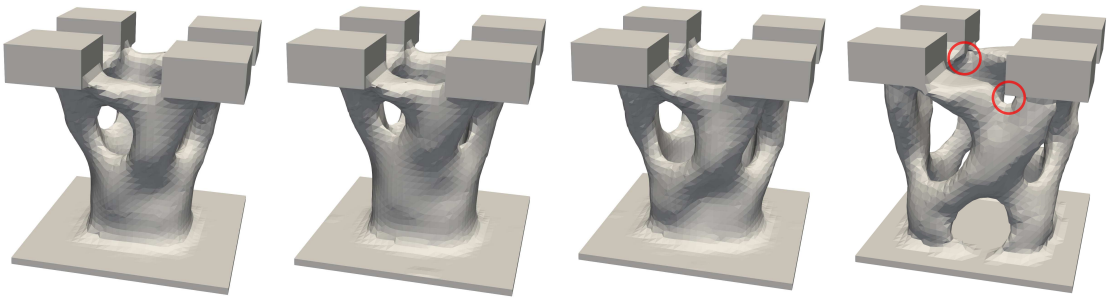


Figure 12: Search algorithm comparison. From left to right, Hemisphere search, normal search, heat search and no subtractive constraint. In the final image there are two areas highlighted where the output would not be manufacturable due to inaccessibility caused by overhangs and features that are too small for the bit to mill.

Algorithm	**Relative compliance**	**Volume**	**Avg time(s)/iter**
Hemisphere	1.000	0.302	8.641
Normal	0.968	0.301	8.333
Heat	0.901	0.301	14.179
No subtractive	0.659	0.302	7.359

Table 1: Comparison of the three accessibility search algorithms as well as the baseline optimization unconstrained by any subtractive constraint.

Figure 13 shows a comparison between unconstrained topology optimization on a skateboard truck (above) and the result of topology optimization with the 5-axis CNC constraint described above (using 'Heat Search'). Note that the unconstrained version has a number of features that would be impossible to mill, such as small holes, thin features and cage structures that would not be accessible by the tool. The constrained version avoids all of these areas and results in a much more 'compressed' result with fewer struts and all the cavities are open to attack from the tool. The constrained result is just 8% heavier and 2% stiffer than the unconstrained version. To validate

this result we manufactured the design using a 5-axis Matsuura MX-330 CNC milling machine. We used a $\frac{1}{4}$ inch bull nose end mill for roughing and a $\frac{1}{8}$ inch ball nose end mill for finishing.

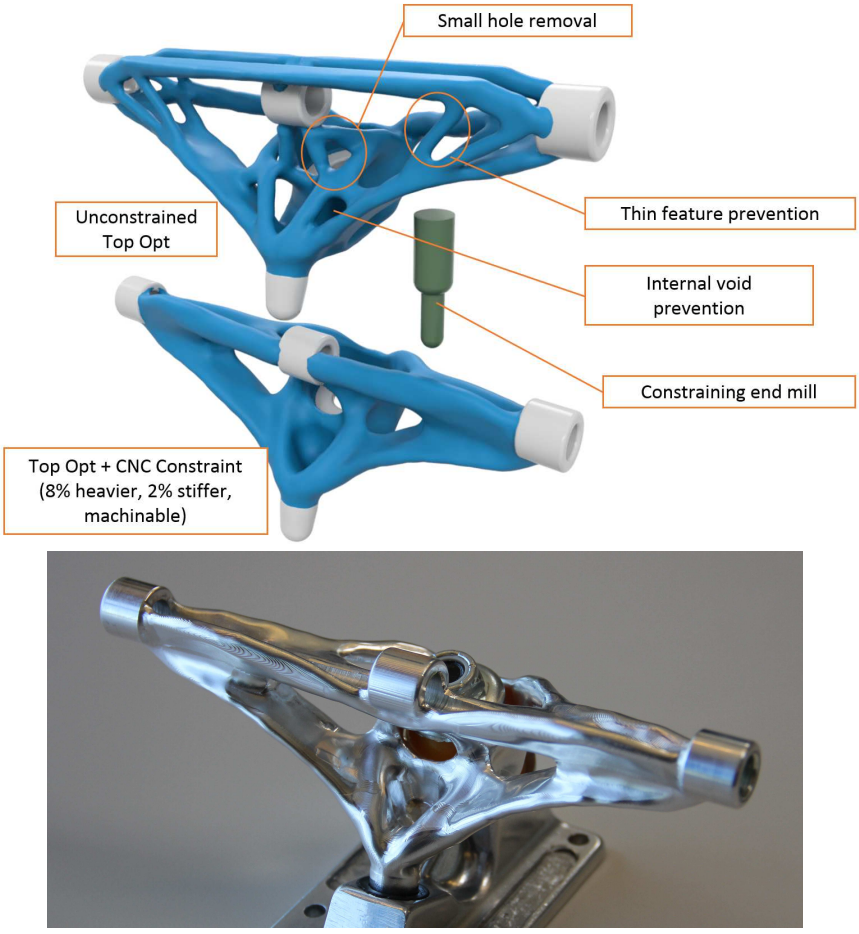


Figure 13: **Top:** Comparison between unconstrained topology optimization and 5-axis CNC milling constrained topology optimization. **Bottom:** Manufactured result using a 5-axis CNC milling procedure.

We demonstrated the algorithm with another case study for a racing car upright. Figure 14 shows the unconstrained topology optimization results along highlighted regions where the curvature is too high or the tool bit cannot access the surface. Alongside this is the constrained 5-axis result that again is more compact without inaccessible undercuts and larger features. We also validated the output by generating a fully machined version of the result shown at the bottom of Figure 14.

6 Conclusion

We have demonstrated a novel method for topology optimization with a manufacturing constraint suitable for subtractive milling machines. Our constraint allows the user to define a milling strategy for 3-axis or 5-axis machines as well as the shape of the tool bits and machine heads that will be used to manufacture the part. We have shown results for variations of the constraint parameters.

6.1 Extensions

Here we explore several potential extensions to the method as well as future work.

It is possible to consider an even wider range of tool bits and heads by adding additional head layers (approximating the head with two or more capped cylinders), or additionally by modeling a tapered frustum instead of a cylinder. Frustum intersection can be achieved by modifying the ray casting function. Another extension would be to allow the user to specify a set of potential tools at their disposal and then the optimization can do accessibility testing for each tool and choose the tool with highest η value.

One of the limitations of the strict algorithm is that the velocity function is clamped such that no growth can occur in the part. Often a topology optimization will require the shifting a subset of the part region or the growth of a subset of the part while another region shrinks. This limits the effectiveness of the topology optimization. We presented the relaxed algorithm as one method to alleviate this restriction. Another potential method is to detect regions with positive velocity and then to backtrack to the previous iteration and set $\eta = 0$ in that region so no volume is removed, thus anticipating the growth.

The algorithms presented here enforce accessibility for the specified machine parameters, this should ensure that the part is millable however it may be very slow to manufacture. Optimizing for milling time is a very interesting extension for future work.

7 Conflict of interest statement

The authors are employed at Autodesk Inc.

8 Replication of Results

The results were obtained using optimization algorithms defined in section 5.2 using the level set framework from [10] and the pseudo code algorithms are provided in the appendix for ease of replication.

9 Appendix

Algorithm 1: LevelSetRayCast

Input: Level set, rayStart, rayDirection, levelSetValue, hitPoint

Result: Returns true if the ray crosses the level set at levelSetValue and stores the intersection point in hitPoint

Algorithm 2: LevelSetOffset

Input: Level set Ω, offset value o

Result: Applies an offset of o to the signed distance function for Ω such that the zero-levelset is uniformly advanced outwardly in the direction of $o\mathbf{n}$, where $\mathbf{n}$ is the surface normal. If the level set is narrow band then the narrow band must be updated such that it centers on the new zero-level set.

Algorithm 3: LevelSetAdvection

Input: Level set Ω, scalar field v, time t

Result: Solves the Hamilton-Jacobi equation on the signed distance function with the speed defined by the scalar field v over a duration t. Note that a velocity extension method improves the stability of advection [1].

Algorithm 4: LevelSetClose

Input: Level set Ω, close value o

Result: Morphological operation that performs an offset by o followed by an inset or negative offset of o. The result is a removal of holes of radius o.

Algorithm 5: MillingTest: Level set evaluation for $V_i(x) \cap \Omega$

Input: Level set Ω, surface normal $\mathbf{n}(x)$, offset point $\mathbf{p}$, milling direction $\mathbf{m}_i$, filter value $\eta(x)$

Result: Returns accessibility boolean, the updated filter $\eta(x)$ for this $\mathbf{m}_i$ and the intersection point x_{hit}

```
if n(x) · m_i < 0 then
    hit, x_hit = LevelSetRayCast(Ω,p,−m_i,r,x_hit);
    if !hit then
        hit, x_hit = LevelSetRayCast(Ω,p − m_i(b + h),−m_i,h,x_hit);
        if !hit then
            η(x) = max (η(x), −m_i · n(x));
            return (accessible = true, η(x), x_hit);
        end
    else
        return (accessible = false, η(x), x_hit);
    end
end
return (accessible = false, η(x), x_hit);
```

Algorithm 6: MillingConstraint for 3-axis

Input: Level set of Ω, $\mathbf{n}$, set of milling directions $\mathcal{M}$
Result: Evaluation of η
Extend narrow-band of Ω up to h;
for *x on boundary of Ω* **do**
  $\mathbf{p} = x + \mathbf{n} \cdot r$;
  $\eta(x) = 0$;
  for *$\mathbf{m}_i$ in $\mathcal{M}$* **do**
    accessible, $\eta(x)$, x_{hit} = MillingTest(Ω,$\mathbf{n}(x)$,$\mathbf{p}$,$\mathbf{m}_i$,$\eta(x)$);
  end
end
return η;

Algorithm 7: MillingConstraint for 5-axis with NormalSearch

Input: Level set of Ω, $\mathbf{n}$
Result: Evaluation of η
Extend narrow-band of Ω up to h;
for *x on boundary of Ω* **do**
  $\mathbf{p} = x + \mathbf{n} \cdot r$;
  accessible = false;
  $\eta(x) = 0$;
  $\mathbf{m} = -\mathbf{n}$;
  while *!accessible* **do**
    accessible, $\eta(x)$, x_{hit} = MillingTest(Ω,$\mathbf{n}(x)$,$\mathbf{p}$,$\mathbf{m}$,$\eta(x)$);
    $x'_{hit} = x_{hit}$;
    do
      $x'_{hit} = x'_{hit} + \mathbf{n}(x_{hit})$;
    while $\phi(x'_{hit}) - \phi(x_{hit}) > 0$;
    $\mathbf{m} = \frac{x'_{hit} - \mathbf{p}}{||x'_{hit} - \mathbf{p}||}$;
  end
end
return η;

Algorithm 8: MillingConstraint for 5-axis with HeatSearch

Input: Level set of Ω, $\mathbf{n}$
Result: Evaluation of η
Extend narrow-band of Ω up to h;
T = SolveHeatEquation(Ω^+);
for *x on boundary of Ω* **do**
  $\mathbf{p} = x + \mathbf{n} \cdot r$;
  $\mathbf{q} = \mathbf{p}$;
  accessible = false;
  $\eta(x) = 0$;
  $\mathbf{m} = -\mathbf{n}$;
  while *!accessible* **do**
    accessible, $\eta(x)$, x_{hit} = MillingTest(Ω,$\mathbf{n}(x)$,$\mathbf{p}$,$\mathbf{m}$,$\eta(x)$);
    $\mathbf{q} = \mathbf{q} + \varepsilon \frac{\nabla T(\mathbf{q})}{||\nabla T(\mathbf{q})||}$;
    $\mathbf{m} = \frac{\nabla T(\mathbf{q})}{||\nabla T(\mathbf{q})||}$;
  end
end
return η;

Algorithm 9: Inner Loop of the Augmented Lagrangian Algorithm for Level Set Topology Optimization — CNC Constraint Strict Algorithm

Input: Initial part domain Ω, Boundary conditions, Objective function, Milling parameters r,b and h, set of milling directions $\mathcal{M}$
Result: Optimized version of Ω
while *!converged* **do**
  EvaluateSensitivities(Ω,Boundary conditions,Objective function);
  v = ComputeShapeDerivative();
  η = MillingConstraint(Ω,$\mathbf{n}$,$\mathcal{M}$,maxIters);
  for *x on boundary of Ω* **do**
    if $v(x) > 0$ **then**
      $\eta(x) = 0$
    end
  end
  ε = LineSearch(Ω,v);
  LevelSetAdvection(Ω,ηv,ε);
end
LevelSetClose(Ω,r);

Algorithm 10: Inner Loop of the Augmented Lagrangian Algorithm for Level Set Topology Optimization — CNC Constraint Relaxed Algorithm

Input: Initial part domain Ω, Boundary conditions, Objective function, Milling parameters r,b and h, set of milling directions $\mathcal{M}$
Result: Optimized version of Ω
while *!converged* **do**
  EvaluateSensitivities(Ω,Boundary conditions);
  v = ComputeShapeDerivative();
  v_{max} = max(v);
  η = MillingConstraint(Ω,$\mathbf{n}$,$\mathcal{M}$,maxIters);
  for x *on boundary of* Ω **do**
    if $\eta(x) == 0$ **then**
      $v(x) = \alpha v_{max}$
    else
      $v(x) = \eta(x)v(x)$
    end
  end
  ε = LineSearch(Ω,v);
  LevelSetAdvection(Ω,ηv,ε);
end
LevelSetClose(Ω,r);

References

[1] D Adalsteinsson and J.A Sethian. The fast construction of extension velocities in level set methods. *J. Comput. Phys.*, 148(1):2–22, January 1999.

[2] Grégoire Allaire, François Jouve, and Anca-Maria Toader. Structural optimization using sensitivity analysis and a level-set method. *J. Comput. Phys.*, 194(1):363–393, February 2004.

[3] Martin Philip Bendsoe and Noboru Kikuchi. Generating optimal topologies in structural design using a homogenization method. November 1988.

[4] Brown and Sharpe Manufacturing Company. *Practical Treatise on Milling and Milling Machines*. Applied mathematical science. Brown & Sharpe Mfg. Co., Providence, R.I., U.S.A., 1914.

[5] Yonghua Chen, Jianan Lu, and Ying Wei. Topology optimization for manufacturability based on the visibility map. *Computer-Aided Design and Applications*, 13(1):86–94, 2016.

[6] Michel C Delfour and J-P Zolâsio. *Shapes and geometries: metrics, analysis, differential calculus, and optimization*, volume 22. Siam, 2011.

[7] James Guest and Mu Zhu. Casting and milling restrictions in topology optimization via projection-based algorithms. *Proc of the ASME Design Engineering Technical Conference*, 3, August 2012.

[8] Matthijs Langelaar. Topology optimization for multi-axis machining. *Computer Methods in Applied Mechanics and Engineering*, 351, 03 2019.

[9] Jikai Liu and Y.-S. Ma. 3d level-set topology optimization: a machining feature-based approach. *Structural and Multidisciplinary Optimization*, 52(3):563–582, September 2015.

[10] Ken Museth, Jeff Lait, John Johanson, Jeff Budsberg, Ron Henderson, Mihai Alden, Peter Cucka, David Hill, and Andrew Pearce. Openvdb: An open-source data structure and toolkit for high-resolution volumes. In *ACM SIGGRAPH 2013 Courses*, SIGGRAPH '13, pages 19:1–19:1, New York, NY, USA, 2013. ACM.

[11] X.M. Wang M.Y. Wang and D.M. Guo. A level set method for structural topology optimization. *Comput Methods Appl Mech Eng*, 192:227–246, 2003.

[12] Jorge Nocedal and Stephen Wright. *Numerical optimization*. Springer Science & Business Media, 2006.

[13] Ole Sigmund and Kurt Maute. Topology optimization approaches: A comparative review. *Structural and Multidisciplinary Optimization*, 48(6):1031–1055, 2013.

[14] Sandro L. Vatanabe, Tiago N. Lippi, Cícero Ribeiro de Lima, Glaucio H. Paulino, and Emílio Carlos Nelli Silva. Topology optimization with manufacturing constraints: A unified projection-based approach. *Advances in Engineering Software*, 100:97–112, 2016.

[15] Yaguang Wang and Zhan Kang. Structural shape and topology optimization of cast parts using level set method. *International Journal for Numerical Methods in Engineering*, 111(13):1252–1273, 2017.

[16] D. V. Widder. *The heat equation*, volume 1. Academic Press, 1975.

[17] Qi Xia, Tielin Shi, Michael Yu Wang, and Shiyuan Liu. A level set based method for the optimization of cast part. *Structural and Multidisciplinary Optimization*, 41(5):735–747, 2010.

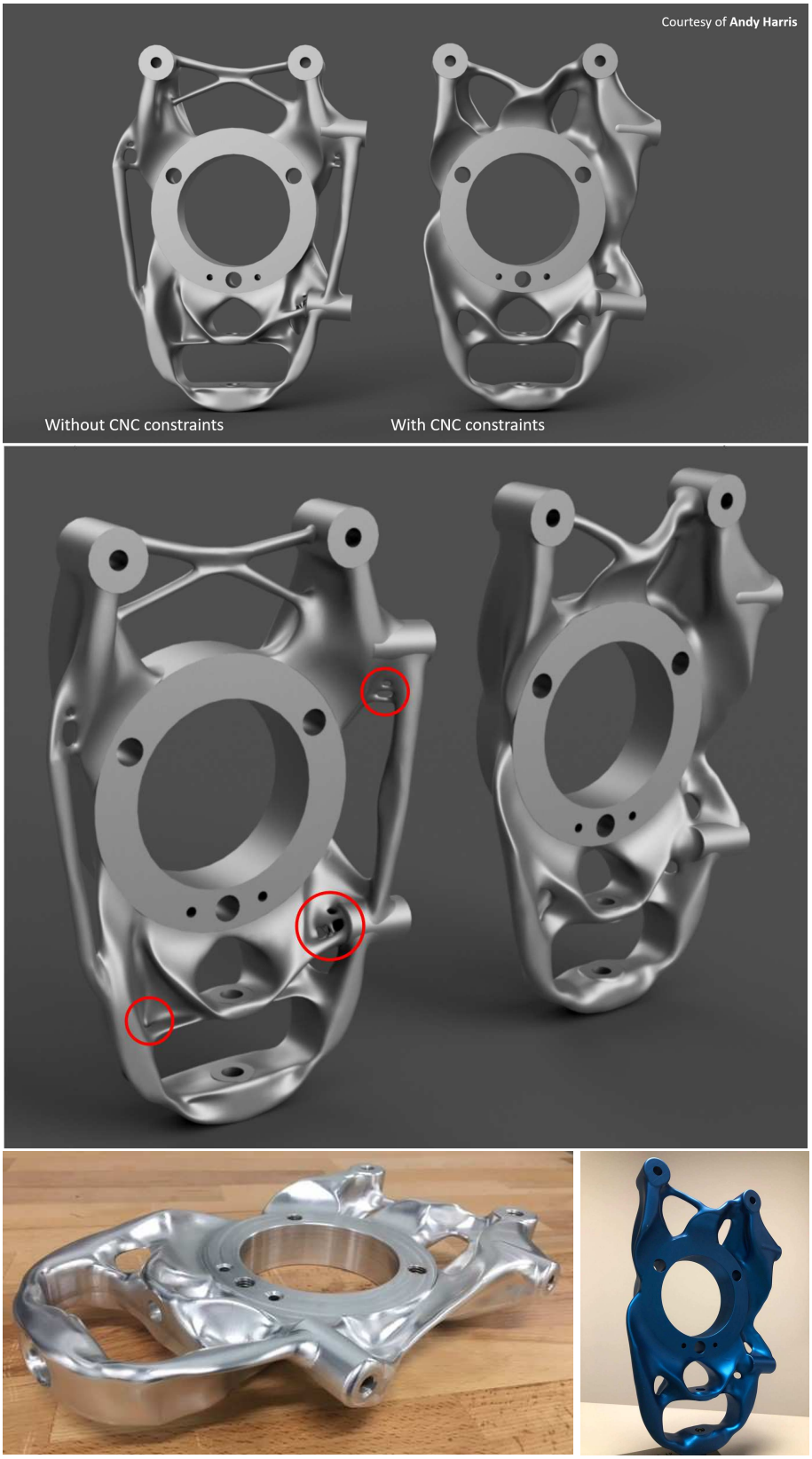


Figure 14: **Left:** Unconstrained topology optimization of race car upright with highlighted non-machinable regions, **Right:** 5-axis CNC milling constrained topology optimization. **Bottom:** Manufactured result using a 5-axis CNC milling procedure.