DOI: 10.1111/cgf.15032
EUROGRAPHICS 2024 / A. Bermano and E. Kalogerakis
(Guest Editors)

COMPUTER GRAPHICS *forum*
Volume 43 (2024), Number 2

Freeform Shape Fabrication by Kerfing Stiff Materials

N. Speetzen[1] L. Kobbelt[1]

[1]RWTH Aachen University, Germany

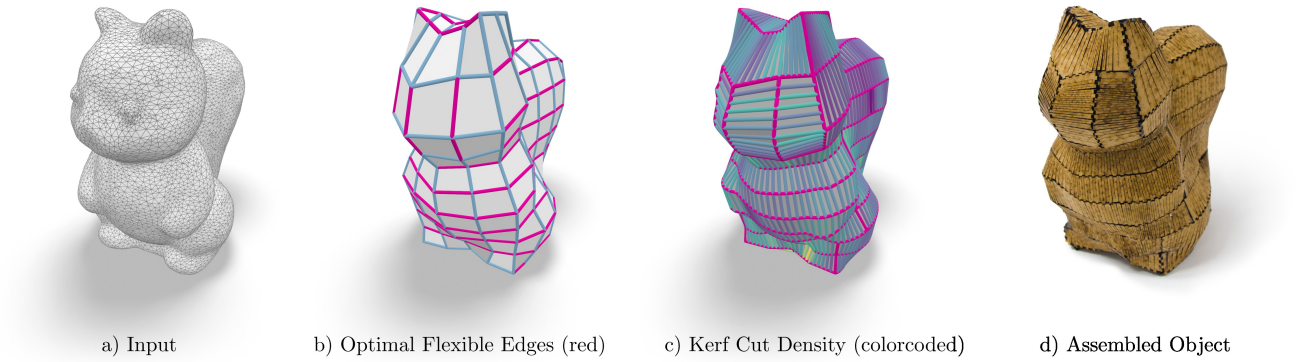


Figure 1: *We present an automatic end-to-end pipeline for freeform shape fabrication by kerfing stiff materials. Given a target shape (a), we first formulate the choice of optimal bending directions as an integer linear program on the edges of a coarse quadmesh generated from the input (b). The resulting quad strips are further subdivided and the border connections are handled via box joints. To achieve the required bending flexibility, we compute adaptive kerf cut densities (c). The kerf pattern is applied to the flattened 2D patches, which results in layouts suitable for laser cutting and assembly into a 3D object (d).* Squirrel *model from the AIM@SHAPE Shape Repository.*

Abstract

Fast, flexible, and cost efficient production of 3D models from 2D material sheets is a key component in digital fabrication and prototyping. In order to achieve high quality approximations of freeform shapes, a common set of methods aim to produce bendable 2D cutouts that are then assembled. So far bent surfaces are achieved automatically by computing developable patches of the input surface, e.g. in the context of papercraft. For stiff materials such as medium-density fibreboard (MDF) or plywood, the 2D cutouts require the application of additional cutting patterns ("kerfing") to make them bendable. Such kerf patterns are commonly constructed with considerable user input, e.g. in architectural design. We propose a fully automatic method that produces kerfed cutouts suitable for the assembly of freeform shapes from stiff material sheets. By exploring the degrees of freedom emerging from the choice of bending directions, the creation of box joints at the patch boundaries as well as the application of kerf cuts with adaptive density, our method is able to achieve a high quality approximation of the input.

CCS Concepts

• ***Computing methodologies*** → ***Shape modeling;*** • ***Applied computing*** → *Engineering;*

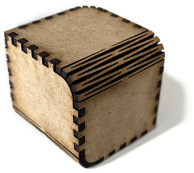
1. Introduction

The fabrication of mid-sized 3D models has use in the context of prototyping, display and exhibition. For cost-efficiency and to allow for manual assembly, bulk 2D material is used, common choices for rigid structures are plywood or medium density fibreboard (MDF), while cardboard and paper are popular for curved surfaces. Developable parts are cut from 2D sheets and afterwards assembled into an approximation of the 3D target shape.

For rigid objects created from stiff materials like MDF, box joints have emerged as a popular way of handling the connection between different planar parts. They provide guidance for the part placement relative to each other, rigidity in one direction, as well as an overlapping surface area to aid in the application of adhesives. As such,

the resulting objects are often quite sturdy and easy to assemble. The planarity of the pieces is, however, a limitation when it comes to freeform shapes, because faithful approximations can only be achieved with excessively small pieces.

Papercraft on the other hand is able to utilize the flexibility of paper to allow for bent (developable) surfaces in the resulting objects. Although papercraft is highly accessible to private users, its main drawback is the fragile result.



To achieve high curvatures using stiff materials, kerf cuts can be applied to the material. These internal cuts provide additional flexibility in directions related to their own orientation and are mostly used in architectural design and as design elements in smaller projects like beveled boxes. The interest in kerfing stiff materials has risen in recent years, triggered by the accessibility of laser cutting devices, although its use is still mostly experimental due to the difficulty of manually applying suitable kerf cuts.

Our method aims to combine the high curvature surface approximation of papercraft and the rigidity that stiffer materials provide, by proposing an automatic pipeline for the creation of kerfed cutouts. While doing so, we take material limitations as well as the creation of box joints at the patch borders into consideration.

We present a full end-to-end pipeline that takes a 3D mesh as input and produces a set of kerfed 2D patches suitable for laser cutting. The main contributions contain:

- The formulation of the choice of local bending directions as a global integer linear program.
- The creation of box joints along bent borders.
- An adaptive kerf cut density to minimize cutting time while retaining the required flexibility.

2. Related Work

Since our work touches on different domains, we show related work from the areas of papercraft and developabilty methods, as well as architectural approaches to surface approximation and works regarding the topic of kerf cutting.

A popular area of research is the automatic cutout generation for building papercraft models. [MS04] propose to use triangle strips, which are inherently developable, to approximate a target surface. They first segment the mesh into parts, adopting the least squares conformal map texture atlas generation by [LPRM02]. After further splitting these parts into zones, they create smooth cut-lines to use as the borders of newly generated triangle strips. A related method to subdivide a user-guided coarse mesh parametrization after splitting it into strips was later proposed by [Mit06]. [STL06] provide another method for creating paper craft models by approximating parts of the mesh via conic planes and resolving their boundaries analytically. The work of [MGE07] uses an approach closer to general developability methods, by approximating an input mesh with generalized cylinders, including error control.

Although not all works towards developable approximations specifically focus on papercraft, many still validate their results by building objects from paper. Examples for these are D-Charts by [JKS05], a method to segment a mesh into quasi-developable charts in an iterative manner, and in recent years methods based on local gauss image thinning [SGC18, BVHSH21, ZFO*22] and developable wrapping [IRHSH20]. [VVHSH22] discuss quadrilateral strip remeshing of already developable surfaces. For an extensive overview over a multitude of methods regarding developable approximation we refer to [YCS23].

There exist a significant amount of methods working on the developability of quad meshes in the architectural domain. [LPW*06] show the optimization of quad meshes towards quad planarity and [PSB*08] approximate large-scale freeform surfaces using planar quad strips and semi-discrete surface representations. A lot of further work towards architectural paneling using individual quadrilateral parts has been done to optimize the developability and approximation quality of the surface while minimizing costs [EKS*10, GSP19, PKW*20, JWR*20, WJW*22, IRWP22].

Relevant techniques that target the fabrication of medium to large sized objects are CofiFab [SDW*16], a method that creates a low resolution frame onto which higher detail panels are attached and WireWarping [Wan08, ZW11], a flattening method with applications in clothing manufacturing, due to its length preserving properties.

Using box joints with stiff material sheets for personal fabrication is explored by the creators of easy-to-use tools like Platener [BGM*15] and kyub [BSK*19], which creates shapes from voxel-like cells. Their follow-up works address topics like reinforcement of weak structures (fastForce [ATK*21]), intuitive layouts of the flattened pieces for easier assembly (Roadkill [ASS*21]) and automatic adjustments corresponding to the width of the kerf cuts [RAS*20, KTA*23].

Most of the works regarding kerfing stiffer materials stem from recent years and [LS21] provide a review of many different bending techniques, including kerf cuts. One of the first applicants of two dimensional kerf patterns, which provide smaller but bidirectional flexibility, were [Iva15], who use interlocking spirals. Iterating on this approach, [ZEK*17] present a method to generate such interlocking spirals from 2D meshes and show how the spirals can be modified to achieve varying levels of stiffness. [CL18] develop kerfed design elements by flattening simple 3D structures and applying a warped version of the kerf pattern to the 2D material. In a follow up work [CL19] they show different approaches to creating surfaces by kerfing in different ways, to, among other things achieve different types of curvature. Note these methods do not discuss the fabrication of entire objects, but the creation or bending behavior of single surfaces.

In the same context of working with plywood or MDF, [CTJ*20] perform stress tests on bidirectionally kerfed MDF planes via uniaxial and biaxial stretching and out of plane bending, while [LS22] analyze the accuracy of arcs created by one-directionally kerfed rectangular MDF strips of different dimensions.

3. Pipeline

The fabrication of 3D models from bent 2D panels is, of course, primarily a *geometric* problem of approximating a given freeform shape by a piecewise representation. However, the physical realization of the object also implies *physical* requirements such as static integrity of the structure or the bendability of the stiff material within patches. Last but not least, there are *aesthetic* criteria to be taken into account since the decomposition of the input shape into patches is a gestalt abstraction process and should hence reflect the underlying „semantic“ structure, like symmetries or constituting parts (primitives).

The idea of our method relies on taking a quad mesh representation of the input target shape and defining the patches of the decomposition by merging quads, which effectively turns the continuous segmentation problem of the surface into a discrete labelling problem of the quads (or edges). For this quad mesh open boundaries and any genus are allowed. There are very powerful quad mesh generation methods available [BLP*13] that can take a multitude of other requirements like a target edgelength and curvature-alignment into account. For our method we assume that the aesthetic (e.g., orientation and alignment) criteria have already been taken into account in the quad mesh generation step and hence are automatically satisfied when merging quads to form patches. Geometric objectives and physical constraints are then considered by our method.

In the following sections, we will walk through the entire pipeline, starting with the choice of bending directions per quad via the formulation as an integer linear program in Section 3.1. The result of this optimization can be used to segment the quad mesh into quad-strips, which are subdivided further corresponding to their their bending directions. This step is followed by the flattening of the strips in Section 3.2, for which we extend the method of angle based flattening [ZLS07]. On the flattened patches, we afterwards compute the dimensions and application areas of the box joints to generate the 2D contours, which is shown in Section 3.3. The last step of the pipeline in Section 3.4 details how these cutouts are made flexible using adaptive kerf cuts that consider the material stiffness. Finally, we show where user guidance can be applied to modify the output of the pipeline in Section 3.5.

3.1. Choosing Bending Directions

The individual quads of the input mesh serve as a basis for further computation. For simplicity we make the restriction that each quad is kerfed in one of its two principal directions, with the two resulting bendable edges acting analogously to generatrices of a ruled surface. This simplification is justifiable, since the quads are often curvature aligned and remaining Gauss curvature can be achieved by twisting the surface. The selection of the direction in which quads are kerfed also depends on their neighborhood. Stiff edges of quads prohibit the bending of both adjacent quads in their direction, even if the bending direction of an individual quad would allow it, see Figure 2. These situations restrict the flexibility of the result, with a perfectly alternating grid resulting in a stiff structure with no bent edges.

For this reason it is often beneficial to bend quads in directions

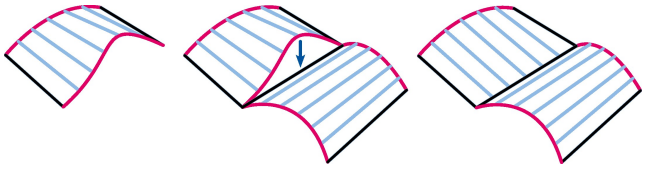


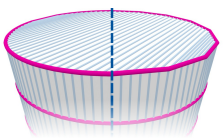
Figure 2: *Bendable edges of quads marked red. Edges between neighboring quads with different bending directions have to be restricted in their bending behavior to avoid the creation of geometric holes.*

that allow neighboring quads to be bent together. We formulate this optimal edge selection problem as an integer linear program:

$$\min_X \sum_E -w_e \cdot x_e \quad \text{s.t.} \quad x_{e_1} + x_{e_2} \leq 1 \quad \forall e_1, e_2 \in E_{\text{adjacency}}. \tag{1}$$

The benefit of choosing an edge to be bendable is formulated as a positive weight $w_e \in \mathbf{R}^{0+}$. We choose this weight to be the variance of the distance to the target shape along the edge, as it is a good indicator whether the target shape has curvature along this edge and is also oblivious to constant offsets. If a (u,v) map between the quad mesh and the input surface is known, the distance of the points on the edge to their mapped image can be chosen instead of the closest distance to the input mesh. Binary decision variables $x_e \in \{0,1\}$ describe whether edge e is selected to be bendable or not. The constraint that only opposite edges of quads can be bent is equivalent to the constraint that no adjacent edges within a quad face can be bent and is formulated as $x_{e_1} + x_{e_2} \leq 1$, for all adjacent edges e_1, e_2.

The results can be used to determine the bending direction of the quad, with two stiff opposing edges dictating the stiff direction, the other one is flexible. If only one of the other two edges is bendable, the quad relies on twisting to support curvature at that edge, while having otherwise stiff edges. To emulate the bending, we subdivide the quads equally in their bending direction. The number of subdivisions can be chosen freely, although a higher subdivision count improves the quality of the approximation. The result of this subdivision step is a closed but non-conforming quad mesh with T-junctions at unaligned patch borders, the vertices of which are projected onto the target shape. In general this is done via normal projection onto the target surface, but similarly to the edge weight computation, if a (u,v) map between quad mesh and target surface exists, it can be used instead. During this step, the vertices at T-junctions are fixed such that they form a straight line identical to their opposite halfedge. This avoids the creation of geometric holes and corresponds to the constraint shown in Figure 2.



To preserve sharp features during the projection without a map, one can enforce that vertices which lie on feature edges in the refined mesh are projected onto features in the target shape. This is beneficial for mechanical or geometric shapes like cylinders, where simple normal projection fails to reproduce the curvature at the circular face borders, but often results in undesirable

deformations when used on organic shapes. Thus, this step is optional.

After determining how individual quads should be bent, this information can be used to infer strips of neighboring quads, which are bent and kerfed in the same direction. The bent edges and their opposites act as boundaries of these strips, as seen in Figure 3. To make this strip creation more effective, we enforce that strips are formed even in flat regions with $w_e \to 0$, by adding a small constant bias w_0 to all edge weights. Increasing this constant can also make the individual edge weights more uniform by reducing their relative differences. The larger the constant, the higher the incentive to align the bending direction of adjacent quads, as this avoids the restricting scenario from Figure 2 and the total number of bendable edges increases. In contrast to setting all edge weights to the same constant value, our formulation maintains the secondary objective of following the curvature as well as possible, should the total number of bent edges stay the same. Figure 4 shows the effect of the patch alignment bias w_0. We can see that the number of patch borders across which the chosen curvature direction is aligned rises with an increase in w_0, at the cost of approximation quality. This effect is evaluated in Section 4.

In addition to the quad strip borders which stem from the chosen bending directions, further ones can be inserted between quads that are bent in the same direction. Their purpose is to retain sharp features or avoid very high curvatures whithin a strip, as they are easier to deal with using box joints at their borders. To automatically insert these borders, we threshold the angle between the opposing sector normals at the ends of the edges. If one of the ends exceeds the chosen angle threshold, the border is created. Additionally, we ensure disk topology for all strips by splitting those that form loops.

Merging quad strips sideways to form wider patches is possible, but since this decreases the approximation quality significantly, we only perform this step in a user-guided manner, which we will later discuss in Section 3.5.

3.2. Patch Flattening

The flattening of the quad strips could be done naively by triangulating the quads and unfolding the strip along the creases. This approach, however, only ever allows the use of one quad wide patches and is prone to propagating the distortion induced by the artificial diagonals in the triangulation. We instead use an adapted version of angle based flattening (ABF) [ZLS07], by initializing with a temporary triangulation of the patch and afterwards optimizing the parametrization using only the flattened quads. Vertices at the patch borders are duplicated during flattening, such that they have one 2D counterpart per patch. During the initial flattening, one arbitrary edge is fixed for every patch, since their parametrizations are independent and the linear system would otherwise be underdetermined. Afterwards, the modified ABF objective consists of two terms, the squared difference in edge lengths l_e to their original 3D lengths $l_e^\star$ along the patch borders and the squared change of the interior angles of the quads:

$$\min \quad \lambda \cdot \sum_{E_{\text{border}}} (l_e^\star - l_e)^2 + (1-\lambda) \cdot \sum_i (\alpha_i^\star - \alpha_i)^2. \qquad (2)$$

The α_i are the 2D angles from the original ABF formulation, with $\alpha_i^\star$ denoting the original angle. The new term, which similarly penalizes the change in edge lengths at the patch borders ensures consistent lengths of the borders of adjacent patches. This is especially important since the box joints along opposite patch borders need to fit together. The weighting parameter $\lambda \in [0,1]$ only has a minor impact on the solution, since the energies do not have strictly contradicting goals, and the border term vanishes in most cases. Our implementation with TinyAD [SBB*22], which uses the projected Newton solver with the Newton decrement as the termination criterion, usually terminates after few iterations, since the initial solution for one quad wide patches is already very close to the optimum.

We optimize the vertex positions of the 3D representation to mimic the distortion introduced by the flattening step, which improves the accuracy of our evaluations. Since the subdivided strips are close to developable, the flattening distortion and thus the required deformations are very small, with the Haussdorff distance

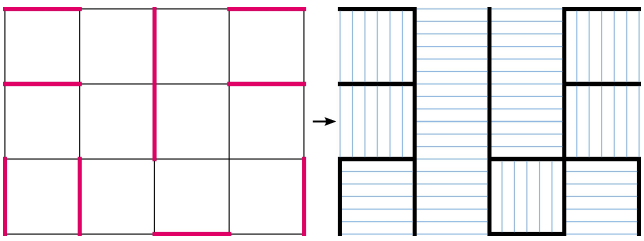


Figure 3: *Left: A possible result of the optimization, the selected bent edges are highlighted in red. Right: The black boundaries correspond to bent edges and their opposites, the hereby defined quad strips have been further subdivided in their bending direction.*

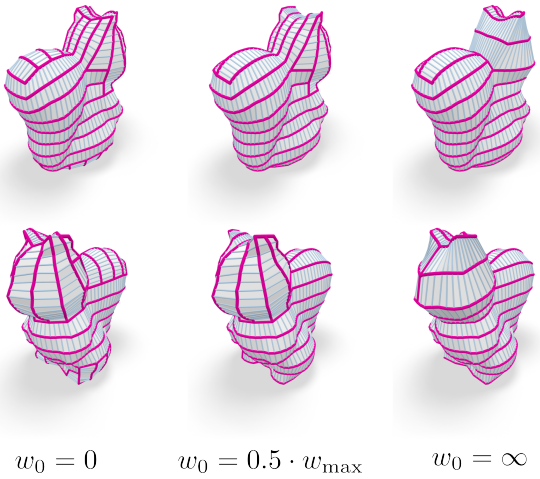


Figure 4: *Adding a constant scalar w_0 to all edge weights w_e aligns the bending directions more consistently, at the cost of a small decrease in approximation quality. The borders of the quad strips resulting from the optimization are highlighted in red. For $w_0 = \infty$ almost all patches are aligned and wrap wround the entire model, disregarding local bending behavior.*

Figure 5: *Left: Crosssection of the boundary between two patches, where the box joint offsets o_a and o_b depend on the normal angle ψ. Right: This normal angle can change along a patchborder when patches are twisted, with $\psi_1 > \psi_2$ in this example.*

between the original and deformed 3D strips staying below 0.1% in all our test cases.

3.3. Box Joints

In the following we discuss the creation of box joints along the patch border, which serve two purposes. First, the box joints help with the assembly of the object by prohibiting tangential sliding along the touching patches, which implies that they do not leave empty space if possible. Secondly, they should provide guidance for the normal angle between the patches through their depth. This can be achieved by adjusting the depth of the box joints such that they end in one exact line, beyond which they would also pierce the hull obtained by offsetting the surface by half of the material thickness outwards and inwards.

Since the computation of the regions in which box joints are applied depends on their depth, we first show how to compute their depth in Section 3.3.1 and afterwards discuss the calculation of the box joint regions in Section 3.3.2.

3.3.1. Box Joint Depths

The box joint depths that fulfill the previously stated purposes are shown in Figure 5. Here, the inner and outer tooth offsets o_a and o_b make the teeth end in one line and create as much overlap as possible. These optimal offsets only depend on the normal angle ψ at which two patches meet and are computed differently, depending on whether the angle is obtuse or acute:

$$\text{if } \psi > \frac{\pi}{2}: \quad o_a = \frac{-h}{2\tan(0.5(\pi-\psi))} \qquad o_b = o_a + \frac{h}{\sin(\psi)} \tag{3}$$

$$\text{else} \qquad o_b = \frac{h}{2\tan(0.5\psi)} \qquad o_a = o_b - h\cdot\sin(\psi) \tag{4}$$

Since the mesh lies at the middle of the material thickness, no distinction between convex and concave angles is necessary. The angle computation is done via sector vertex normals and to compute the offsets at T-junctions the normals on the side of the continuous patch borders are spherically interpolated.

3.3.2. Box Joint Regions

If box joints are applied along the entire length of the patch boundary edges, their cutins and teeth overlap, depending on their inner and outer offsets. The overlaps can happen between different box joints within a patch, but also globally between box joints and other nearby surfaces. To avoid these collisions, we compute safe box joint regions. These are defined as a set of disjoint segments for each of the border edges and describe the regions of the patch border that should be filled with box joints, and regions in which they might overlap. Finding and resolving all these intersections globally via simultaneous restriction of the box joint regions is very difficult, and would require mesh offsetting and mesh booleans.

We instead approximate the feasible box joint regions by working with the flattened patches individually. Their boundary edges are offset outwards and inwards, using the previously computed box joint offsets o_a and o_b from Section 3.3.1. The area between the two offsets represents the entire area the box joints could fill, while intersections between multiple of these areas show potential overlaps. To determine where teeth must be avoided along the edges, we project these intersections onto the edges and subtract them from the valid box joint regions. Including linear transitions between the offset areas into the intersection calculation additionally adds collision to the linear transitions between box joints for adjacent edges. Figure 6 shows an example for how the box joint regions are restricted for two adjacent edges of a patch, based on the overlap of their inner offsets o_a.

Figure 6: *Top view of a patch boundary. Left: The potential overlap between box joints restricts their application regions. The intersection area shaded red is projected onto the boundary edges, to prohibit the creation of box joints. Right: Possible box joint configuration. Note that the teeth stop before the critical region.*

Note that by using this approach instead of only considering the angle at which successive border edges meet, all edges of a patch can affect each others box joint regions. This method can detect the most relevant self-intersections between box joints and is robust towards some degeneracies, such as patches with very thin regions or vanishing surface areas, as the resulting valid box joint regions are empty as well, see Figure 7. Since inputs that contain global self-intersections, or for which adding the material thickness leads to surface penetration, have no well defined solution, detecting these situations is not necessary.

3.4. Adaptive Kerf Cuts

Applying kerf cuts to a material makes it more flexible and adapting the density of these cuts generally yields control over the possible bending radius. The pattern in which these cuts are applied greatly impacts the result, with many kerf patterns inducing one-directional flexibility, while some patterns like interlocking spirals [Iva15, ZEK*17] allow bidirectional bending at the cost of lower one-directional flexibility. Since our goal is to achieve high curvatures, we rely on the simple kerf pattern of alternating parallel cuts.

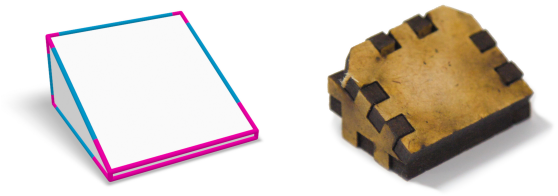
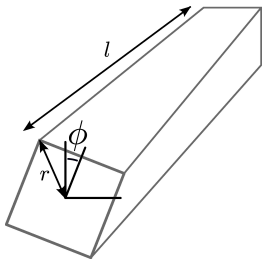


Figure 7: *Even if faces of the quad mesh become very small (left), the box region computation allows us to avoid the critical areas marked in red, where the thickness of the material would otherwise lead to self-intersections. The result (right) has no cutout for the degenerate front face.*

We are restricted to ruled surfaces, but negative Gauss curvature can be obtained by twisting the resulting strips.

While the maximum density of cuts is only restricted by the fabrication method, an incentive to use the minimum number of kerf cuts is the production time, which can be lowered drastically by reducing the cut density based on the locally required flexibility. Additionally, using too many cuts may result in a very unstable result, while the opposite can lead to the material breaking under the bending stresses, as shown by [LS22]. Since we are working with a laser cutter, the fabrication constraint translates to a minimum distance between cuts, which stems from the risk of material being burnt away entirely. To save cutting time and increase stability, we use an adaptive pattern density based on the material properties.

The alternating kerf cuts transform the strip into a series of concatenated beams, the twist of which creates the curvature of the strip. As the basis for our further computations we thus use the torsion formula from beam theory:



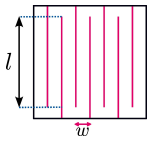
$$T = \frac{J_T \cdot \tau_{max}}{r} = \frac{J_T \cdot G \cdot \phi}{l} \quad (5)$$

The torque T applied at the end of a beam with length l is directly proportional to the twisting angle ϕ, as well as the maximum resulting shear stress τ_{max}, which appears at the maximum distance of the crosssection from the axis r. G and J_T are the shear modulus, a material constant, and the torsion constant, which is related to the shape of the crosssection of the beam. Reordering the terms of Equation 5 lets us eliminate J_T and gives us direct relations between the remaining values:

$$1 = \frac{\phi \cdot r}{l} \cdot \frac{G}{\tau_{max}} \quad (6)$$

Inserting the yield strength τ_{yield} for τ_{max} lets us estimate which angle can be reached at most for any given r and l. Here we implicitly assume that the material behaves isotropically. For high quality MDF this assumption is reasonable, while some types of MDF show significant differences in behavior depending on the direction of the applied forces due to anisotropic fibre orientations. MDF is very brittle which lets us assume that its yield strength is very similar to its ultimate strength, the stress at which it breaks. As reference for these values we refer to [CTJ*20], who perform stress tests on kerfed MDF panels. They also assume isotropic behavior and provide the values $\tau_{yield} = 18\text{MPa}$ and $G = 1.6\text{GPa}$. We use $\frac{G}{\tau_{yield}} = 100$ from now on, which incorporates a safety margin to allow for easier assembly without accidentally breaking the material.

Applying kerf cuts splits the patch into multiple beams, essentially multiplying the total length over which the torsion is distributed. Additionally, increasing the number of cuts also decreases the width of the cross-section of these rectangular beams, see inset.



Thus, both l and r are affected by the number of kerf cuts. Since our goal is to vary the cut density along the patches, we split each patch into short segments and compute the needed number of beams $n_e \in \mathbf{R}^+$ for each segment around a patch-internal edge e individually. The total width of the segment w_e around edge e is measured conservatively as half of the minimal width of both adjacent faces. Per segment, the individual beams now have width $w = w_e/n_e$ and the constant height h that is given by the material thickness. To simplify computations, we assume that the length of the beams, i.e. the width of the strip is constant in the area around the edge. This leads to the total beam length $l_e = l \cdot n_e$. The target twisting angle ϕ_e per segment is given by the maximum normal angle between the two ends of the segment. Inserting all these quantities into Equation 6 and solving for n_e (Appendix A) yields

$$n_e = \sqrt{\frac{Ah^2}{8} + \sqrt{-\frac{A^2h^4}{64} + \frac{Aw_e^2}{4}}} \quad (7)$$

$$\text{with} \quad A = \left(\frac{\phi \cdot G}{l \cdot \tau_{yield}}\right)^2. \quad (8)$$

To use this local information about how many beams are needed in the regions around each edge along the strip, the n_e need to be converted into a discrete set of cut positions. We do this by using the crossection widths w_e to define a 1D coordinate for each edge e, similarly to unrolling the patch. Each n_e can be interpreted as an integral value over the 1D domain around e, such that the total integral over the entire strip indicates the total number of beams the strip needs. We scale the total integral to the next integer value and split the area into even sections of size 1, see Figure 8. This implicitly weights the position of the cuts by n_e and is equivalent to the equilibrium of a 1D mass-spring system (or Tutte embedding) where the n_e act as edge weights and the cut positions as vertices. After the per edge computation of the n_e, we smooth them in a small neighborhood using a linear, 1cm wide kernel, which avoids rapid changes in n_e. Smoothing helps in distributing the cuts more evenly to avoid spikes in cut density and improves the visual appearance, while the small kernel size keeps the result accurate.

An example for the resulting n_e can be seen in Figure 9. The distance between the computed cut positions can be used to easily identify where the cuts are placed closer to each other than a threshold allows. The top part of Figure 9 shows an example for such critical regions, where in thin areas with high curvature cuts have to be placed closer than a custom limit of 0.9mm.

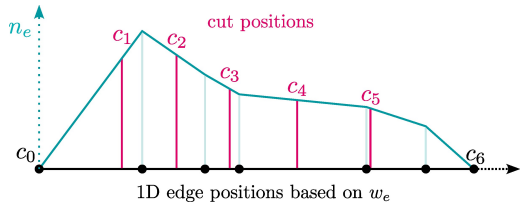


Figure 8: *Computation of the cut positions c_i depending on the locally required number of beams n_e. The distance between the black dots representing the edges depends on the width w_e of the segments assigned to the edges. The integral of n_e over all intervals $[c_i, c_{i+1}]$ is equal to one.*

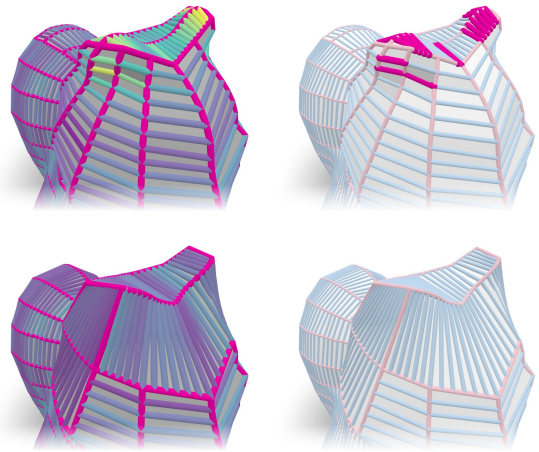


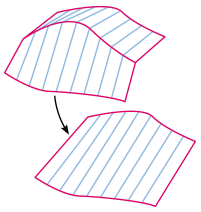
Figure 9: *Top: Cut density based on the locally required number of beams per edge n_e, high densities are shown in brighter colors (viridis colormap). Areas where the necessary cut density is too high are highlighted red on the right. Bottom: Increasing the patch alignment constant and merging two thin strips on top resolves these issues.*

3.5. User Guidance

Although the described method works without any user interaction, some automatic decisions can optionally be overridden by the user. This is helpful if a solution which yields suboptimal energy values is more aesthetically pleasing. During the bending direction optimization, one can override the fully automatic choices by enforcing the behavior of specific edges. This only adds one additional constraint per selected edge in the optimization.

The other step in which user guidance can be applied is the merging of quads into patches. Introducing additional borders to split patches helps highlight specific creases, while the merging of patches is appropriate in planar regions to reduce the number of resulting patches.

We can specifically allow the user to merge aligned patches "sideways", i.e. perpendicular to the bending direction, which increases the achievable curvature due to longer kerf cuts. Doing so has the side-effect of forcing all interior vertices onto the new ruling lines of the patch and thus losing approximation quality (see inset). The creation of new T-junctions poses no problem, since we already work with a non-conforming layout. An important restriction for the creation of wider patches is that no singularity of the quad mesh can lie within a patch. This scenario would lead to a change in bending and kerfing direction within a strip, which would severely affect the flexibility of the patch and is better handled by multiple patches of different bending directions.



4. Discussion and Results

To show the feasibility of our method, we apply the pipeline to different inputs.

A target shape with an open boundary is the *Face* [car19], for which the quad mesh was generated using integer-grid maps (IGM) [BCE*13]. It features rapid changes in curvature, noticeable in the high changes in cut density around the eyes, as seen in Figure 10. To test the effectivity of the adaptive kerf cut densities, we compare to a version of the face for which we instead apply a constant cut density to the faces of the generated quad mesh. While this version naturally looks quite uniform, its surplus in flexibility only marginally simplifies the assembly. More notably, laser cutting the constant version takes 4h instead of 1h40 for the version with adaptive cut density, although both of the versions have the same size. This is due to the constant cut density being chosen high enough to support the necessary bending in all areas of the model. The constant cut density also leads to material burn around the eyes and at the lower boundary of the model, since the quads there are both distorted and small.

For our adaptive cut density we leave out the safety margin mentioned in Section 3.4, to try approaching the material limits as close as possible. When computing the cut density, we see that only the eyes are critical regions, while the quads at the boundary pose no problem, due to their lower bending radius, even if they remain distorted. After detecting the issue at the eyes, we simply adapt the quad mesh manually and obtain a feasible solution.

The model *Bob* [Cra] shown in Figure 11 is an example of a genus 1 object with very intuitive optimal bending directions in the torus region. Additionally, a catmull-clark control mesh is given besides the target triangle mesh, which allows us to test our method with a different quad mesh initialization, for which we simply project the control mesh onto the target shape. To reduce the number of patches and avoid too thin regions inside the torus, we manually merge pairs of strips around the torus, which results in a uniform looking and easy to assemble output. The oval cutouts that form the torus region show the seamless change in cut density, depending on the width of the strip.

As a more complex test case we use the *Squirrel* model from the AIM@SHAPE Shape Repository, see Figure 1, which features

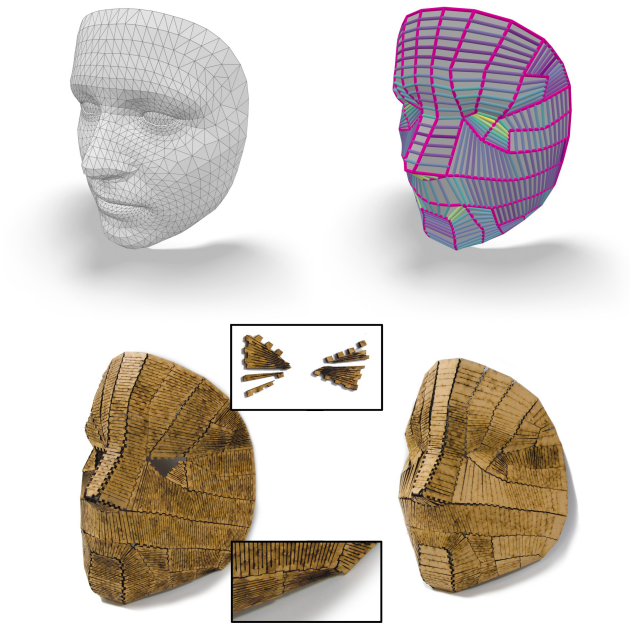


Figure 10: *Fabrication of the Face model: Input (top left) and the refined quad mesh with colorcoded adaptive cut densities (top right). The two variants shown at the bottom are produced using a constant number of kerf cuts per quad (bottom left) and our adaptive density without safety margin (bottom right). The constant cut density leads to material burn around the eyes and face border, while we can detect the critical region around the eyes beforehand and modify the quad mesh accordingly, even before fabricating the model.*

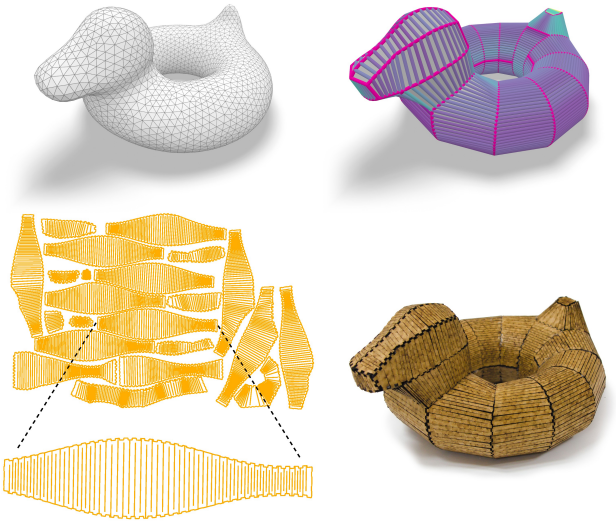


Figure 11: *Fabrication of the Bob model using our method: Input (top left), the refined quad mesh with colorcoded adaptive cut densities (top right), resulting cutouts (bottom left) and the assembled object (bottom right). The chosen bending directions capture the roundness of the torus and the result has a uniform appearance.*

$w_0/w_{\max}$	$d_{\mathrm{AHD}}(\text{input}, \text{sub QM})$	$n_{\text{aligned}}/\lvert E\rvert$	n_{strips}
0.0	0.310%	75.5%	53
0.1	0.310%	81.6%	44
0.2	0.323%	85.8%	34
0.3 - 0.4	0.326%	86.8%	32
0.5	0.328%	87.7%	26
0.6 - 0.8	0.333%	89.4%	23
≥ 0.9	0.336%	92.2%	17

Table 1: *For the Squirrel model: Relation between the relative patch alignment constant and the resulting average Haussdorff distance between the target shape and the subdivided quad mesh, the number of aligned quad borders and the resulting number of quad strips. (see Figure 4).*

regions with significant curvature and high detail. We generated the initial quad mesh using IGM [BCE*13], for which a fully automatic run of our method predicted cut density issues in the region around the ears of the Squirrel, shown in Figure 9. We resolved these by increasing the patch alignment constant from Section 3.1 to $w_0 = 0.3 w_{\max}$ (Section 3.1, Figure 4) and merging two thin patches between the ears. Afterwards we were able to successfully laser cut and assemble the 3D object. Table 1 shows the effect of the patch alignment on the average Haussdorff distance between the target shape and the subdivided quad mesh, as well as the number of resulting quad strips. Since this distance is only marginally affected, we simply increase the patch alignment to reduce the number of strips as desired. To note is that some patches were split, such that the cutouts fit into the bed of our laser cutter.

Further objects we used for our evaluation are the Stanford Bunny (Figure 12) and the top of the Lilium tower (Figure 13) as an architectural model. The fabrication of the Lilium model did not require any user guidance, for the Bunny model we changed the bending direction for the right ear, to avoid too high curvatures.

During our testing we found the limiting factor for the speed of assembly to be the rate at which the used adhesive hardens enough to withstand the torque it is subjected to at the borders between highly bent patches. We used an Epilog Helix Mini 24 laser cutter for the cutout fabrication and a simple hot glue gun for assembly. Further information and parameters for the shown test cases are given in Table 2.

Beyond its intended use case, our method can emulate simple papercraft behavior by setting the material thickness $h = 0$ and its yield strength $\tau_{\text{yield}} = \infty$. Since we are constrained to ruled surfaces by design, the solution space that is extended due to the flexibility of paper cannot be fully explored. The papercraft method by [MS04] also produces ruled surfaces and is based on creating and unfolding triangle strips. The qualitative comparison in Figure 14 shows that our results on the *Bunny* [Sta] are generally comparable to theirs, while our method produces strips with straighter

	Squirrel	Face	Bob	Lilium	Bunny
Alignment Constant c	0.3	0.01	0.01	0.3	0.3
Patch Split Angle α	60°	60°	60°	60°	60°
Tooth Width	3mm	3mm	3mm	3mm	3mm
Material Thickness h	3mm	3mm	3mm	3mm	3mm
N Patches	38	28	22	11	32
Size (AABB in cm)	15*25*22	20*25*14	29*15*23	40*10*40	15*25*19
Cutting Time	5h10	1h40	4h00	1h20	2h30
Assembly Time	4h30	0h40	2h00	2h00	4h20
d_H(sub QM $\leftrightarrow$ target)	2.1%	2.1%	3.3%	1.1%	1.9%
d_H(scan $\rightarrow$ target)	6.9%	11.9%	4.5%	5.7%	7.5%

Table 2: *Parameters and information for the shown test cases.*

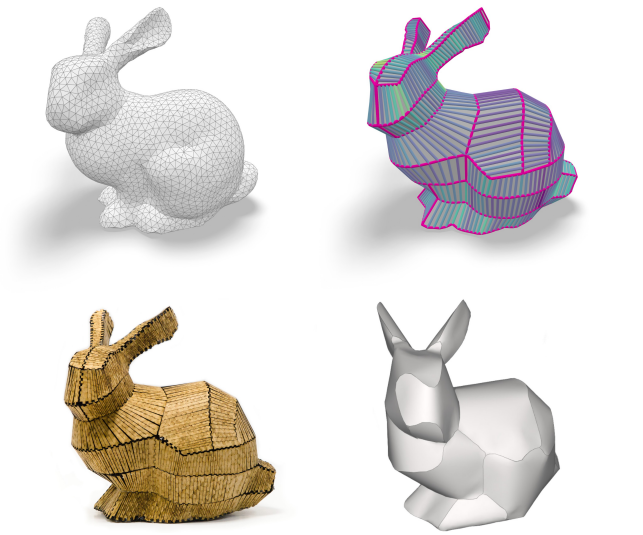


Figure 12: *Fabrication of the Bunny model using our method: Input (top left), the refined quad mesh with colorcoded adaptive cut densities (top right), assembled object (bottom left) and the result of [IRHSH20] with 26 patches (bottom right).*

Figure 13: *For the Lilium model: Input (top left), the refined quad mesh with colorcoded adaptive cut densities (top right), assembled object (bottom left) and the result of [IRHSH20] which also has 11 patches (bottom right).*

borders, which is a side effect of using quad instead of triangle strips.

A more recent method to create developable patches that are suitable for papercraft stems from [IRHSH20]. We included their results for the Bunny (26 patches) and Lilium model (11 patches) in the corresponding Figures 12 and 13. We can again see the difference in patch structure, where our method strongly reflects the underlying quad mesh and thus yields more uniform patches. Measuring the Haussdorff distance shows that it is slightly lower for our quad strips (Bunny: 1.9% in comparison to 2.8%, Lilium: 1.1% in comparison to 2.1%), although our method does not eliminate negative Gauss curvature.

As a brief experimental verification to our kerf cut density computation, we used a simple cylindrical shape. This ring unfolds into a single strip with constant curvature, which makes the effect of the cut density easy to observe. Using our default values, we created rings of different diameters with ease. Reducing the number cuts further is possible, although a reduction by more than 25% often leads to material failure. The remaining flexibility stems from the connecting points of the twist beams, and a slightly higher stress tolerance of our material than the MDF evaluated by [CTJ*20]. For a more in-depth material analysis, we refer to [CTJ*20] and [LS22].

Lastly, we laser scanned the built objects and compared them to their digital counterparts. They approximate the target shapes very well locally, although the accumulated inaccuracies made during assembly without supporting structures can lead to slight global deformations, which are less prevalent in objects without open boundaries. The resulting Haussdorff distances to the targets lie in the range of 5-10% for the raw scan data of our test objects, which are measured without noise filtering or mitigation of the outwards offset from only scanning the surface of the object, see Table 2.

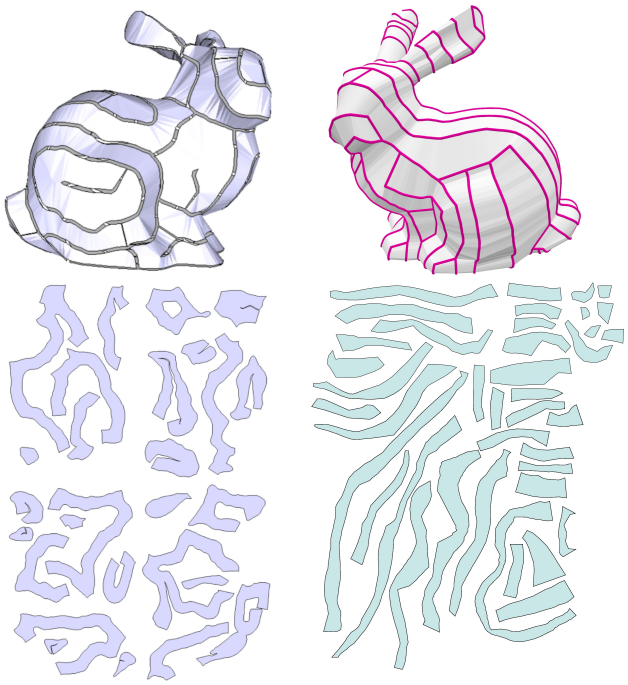


Figure 14: *Left: Papercraft bunny by [MS04], consisting of 27 patches. Right: Papercraft emulated using our method, consisting of 40 patches. The use of quad instead of triangle strips affects the overall shape of the patches.*

4.1. Limitations

While our methods minimizes the approximation error between the input freeform shape and the bent output panels, the achievable approximation tolerance is bounded by the structure and resolution of the input quad mesh. If the a posteriory approximation error is above a desired threshold, we generate a finer input quad mesh and re-run the procedure. Since quad mesh generation has become a commodity in recent years it is very easy to adapt them to user-specified requirements.

Since we take the stiffness and thickness of the materials into account, we can predict where the required kerf cut density is higher than a certain limit. This limit is usually fixed and given by the fabrication method. Our method is currently not able to resolve these situations automatically without user guidance or an improvement of the input quality.

5. Conclusion

We propose an automatic pipeline for the fabrication of 3D shapes using bent 2D cutouts from stiff material planes. By choosing optimal bending directions on a quad mesh generated from the input via the formulation as an integer linear program, quad patches are formed. At the boundaries of these patches box joints are applied, to allow for easy assembly into a 3D object. To ensure sufficient flexibility of the patches when considering material properties and to reduce fabrication times as much as possible, an adaptive kerf pattern is applied. We show the feasibility of our method by applying it to different inputs using medium-density fibreboard and its flexibility by emulating simple papercraft by only adapting two parameters.

Our method significantly reduces the required time investment to create suitable cutouts and makes freeform shape fabrication much more accessible. The previously significant user involvement is reduced to making optional aesthetic choices or resolving predicted bending issues for the given input.

During the automatic merging of the quads, we only allow the creation of quad strips, although wider patches can be beneficial in, e.g. flat regions. A possible direction for future research is the optimization of optimal non-conforming quad patch layouts without internal singularities, which could automate the creation of wider patches and as such also help decouple the output from the input resolution.

Acknowledgements

Open Access funding enabled and organized by Projekt DEAL.

References

[ASS*21] ABDULLAH M., SOMMERFELD R., SEIDEL L., NOACK J., ZHANG R., ROUMEN T., BAUDISCH P.: Roadkill: Nesting Laser-Cut Objects for Fast Assembly. In *The 34th Annual ACM Symposium on User Interface Software and Technology* (Virtual Event USA, Oct. 2021), ACM, pp. 972–984. URL: https://dl.acm.org/doi/10.1145/3472749.3474799, doi:10.1145/3472749.3474799. 2

[ATK*21] ABDULLAH M., TARAZ M., KOMMANA Y., KATAKURA S., KOVACS R., SHIGEYAMA J., ROUMEN T., BAUDISCH P.: FastForce: Real-Time Reinforcement of Laser-Cut Structures. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems* (Yokohama Japan, May 2021), ACM, pp. 1–12. URL: https://dl.acm.org/doi/10.1145/3411764.3445466, doi:10.1145/3411764.3445466. 2

[BCE*13] BOMMES D., CAMPEN M., EBKE H.-C., ALLIEZ P., KOBBELT L.: Integer-grid maps for reliable quad meshing. *ACM TOG 32*, 4 (July 2013), 1–12. doi:10.1145/2461912.2462014. 7, 8

[BGM*15] BEYER D., GUREVICH S., MUELLER S., CHEN H.-T., BAUDISCH P.: Platener: Low-Fidelity Fabrication of 3D Objects by Substituting 3D Print with Laser-Cut Plates. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems* (Seoul Republic of Korea, Apr. 2015), ACM, pp. 1799–1806. URL: https://dl.acm.org/doi/10.1145/2702123.2702225, doi:10.1145/2702123.2702225. 2

[BLP*13] BOMMES D., LÉVY B., PIETRONI N., PUPPO E., SILVA C., TARINI M., ZORIN D.: Quad-mesh generation and processing: A survey. *Computer Graphics Forum 32*, 6 (Sept. 2013), 51–76. doi:10.1111/cgf.12014. 3

[BSK*19] BAUDISCH P., SILBER A., KOMMANA Y., GRUNER M., WALL L., REUSS K., HEILMAN L., KOVACS R., RECHLITZ D., ROUMEN T.: Kyub: A 3D Editor for Modeling Sturdy Laser-Cut Objects. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems* (Glasgow Scotland Uk, May 2019), ACM, pp. 1–12. URL: https://dl.acm.org/doi/10.1145/3290605.3300796, doi:10.1145/3290605.3300796. 2

[BVHSH21] BINNINGER A., VERHOEVEN F., HERHOLZ P., SORKINE-HORNUNG O.: Developable approximation via gauss image thinning. *Computer Graphics Forum 40*, 5 (Aug. 2021), 289–300. doi:10.1111/cgf.14374. 2

[car19] CARDIOMAN: Face model, 2019. CC 4.0 License. URL: https://sketchfab.com/3d-models/face-mesh-44f05170b0ca4bad9fbcb97d66442c53. 7

[CL18] CAPONE M., LANZARA E.: Kerf bending: ruled double curved surfaces manufacturing. In *Blucher Design Proceedings* (São Carlos, BR, Nov. 2018), Editora Blucher, pp. 653–660. doi:10.5151/sigradi2018-1389. 2

[CL19] CAPONE M., LANZARA E.: Parametric Kerf Bending: Manufacturing Double Curvature Surfaces for Wooden Furniture Design. In *Digital Wood Design*, Bianconi F., Filippucci M., (Eds.), vol. 24. Springer International Publishing, Cham, 2019, pp. 415–439. Series Title: Lecture Notes in Civil Engineering. doi:10.1007/978-3-030-03676-8_15. 2

[Cra] CRANE K.: Bob model. CC0 1.0 Universal License. URL: https://www.cs.cmu.edu/~kmcrane/Projects/ModelRepository/. 7

[CTJ*20] CHEN R., TURMAN C., JIANG M., KALANTAR N., MORENO M., MULIANA A.: Mechanics of kerf patterns for creating freeform structures. *Acta Mechanica 231*, 9 (Sept. 2020), 3499–3524. doi:10.1007/s00707-020-02713-8. 2, 6, 9

[EKS*10] EIGENSATZ M., KILIAN M., SCHIFTNER A., MITRA N. J., POTTMANN H., PAULY M.: Paneling architectural freeform surfaces. In *ACM SIGGRAPH 2010 papers* (Los Angeles California, July 2010), ACM, pp. 1–10. doi:10.1145/1833349.1778782. 2

[GSP19] GAVRIIL K., SCHIFTNER A., POTTMANN H.: Optimizing B-spline surfaces for developability and paneling architectural freeform surfaces. *Computer-Aided Design 111* (June 2019), 29–43. doi:10.1016/j.cad.2019.01.006. 2

[IRHSH20] ION A., RABINOVICH M., HERHOLZ P., SORKINE-HORNUNG O.: Shape approximation by developable wrapping. *ACM TOG 39*, 6 (Dec. 2020), 1–12. doi:10.1145/3414685.3417835. 2, 9

[IRWP22] INZA V. C., RIST F., WALLNER J., POTTMANN H.: Developable quad meshes, 2022. arXiv:2210.04099, doi:10.48550/ARXIV.2210.04099. 2

[Iva15] IVANIŠEVIĆ D.: Super flexible laser cut plywood, 2015. URL: http://lab.kofaktor.hr/en/portfolio/super-flexible-laser-cut-plywood/. 2, 5

[JKS05] JULIUS D., KRAEVOY V., SHEFFER A.: D-Charts: Quasi-Developable Mesh Segmentation: Geometry IV. *Computer Graphics Forum 24*, 3 (Sept. 2005), 581–590. doi:10.1111/j.1467-8659.2005.00883.x. 2

[JWR*20] JIANG C., WANG C., RIST F., WALLNER J., POTTMANN H.: Quad-mesh based isometric mappings and developable surfaces. *ACM TOG 39*, 4 (Aug. 2020). doi:10.1145/3386569.3392430. 2

[KTA*23] KATAKURA S., TARAZ M., ABDULLAH M., METHFESSEL P., RAMBOLD L., KOVACS R., BAUDISCH P.: Kerfmeter: Automatic Kerf Calibration for Laser Cutting. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems* (Hamburg Germany, Apr. 2023), ACM, pp. 1–13. URL: https://dl.acm.org/doi/10.1145/3544548.3580914, doi:10.1145/3544548.3580914. 2

[LPRM02] LEVY B., PETITJEAN S., RAY N., MAILLOT J.: Least squares conformal maps for automatic texture atlas generation. ACM TOG, New York, NY, USA, Aug. 2002. doi:10.1145/3596711.3596734. 2

[LPW*06] LIU Y., POTTMANN H., WALLNER J., YANG Y.-L., WANG W.: Geometric modeling with conical meshes and developable surfaces. In *ACM SIGGRAPH 2006 Papers on - SIGGRAPH '06* (Boston, Massachusetts, 2006), ACM Press, p. 681. doi:10.1145/1179352.1141941. 2

[LS21] LORENZONI B. R., SILVA F. P. D.: Bending techniques for flat materials using cut patterns: a review. *Global Journal of Engineering and Technology Advances 7*, 2 (May 2021), 091–102. doi:10.30574/gjeta.2021.7.2.0070. 2

[LS22] LORENZONI B. R., SILVA F. P. D.: Geometric analysis of the MDF kerf-bending structure accuracy. *International Journal of Space Structures 37*, 2 (June 2022), 135–149. doi:10.1177/09560599221081015. 2, 6, 9

[MGE07] MASSARWI F., GOTSMAN C., ELBER G.: Papercraft Models using Generalized Cylinders. *15th Pacific Conference on Computer Graphics and Applications (PG'07)* (2007), 148–157. 2

[Mit06] MITANI J.: Strip creation for designing curved papercraft models adopting mesh subdivision scheme. *NICOGRAPH International* (2006). 2

[MS04] MITANI J., SUZUKI H.: Making papercraft toys from meshes using strip-based approximate unfolding. In *ACM SIGGRAPH 2004 Papers* (Los Angeles California, Aug. 2004), ACM, pp. 259–263. doi:10.1145/1186562.1015711. 2, 8, 10

[PKW*20] PELLIS D., KILIAN M., WANG H., JIANG C., MÜLLER C., POTTMANN H.: Architectural freeform surfaces designed for cost-effective paneling through mold re-use. *Advances in Architectural Geometry* (2020), 2–16. 2

[PSB*08] POTTMANN H., SCHIFTNER A., BO P., SCHMIEDHOFER H., WANG W., BALDASSINI N., WALLNER J.: Freeform surfaces from single curved panels. In *ACM SIGGRAPH 2008 papers* (Los Angeles California, Aug. 2008), ACM, pp. 1–10. doi:10.1145/1399504.1360675. 2

[RAS*20] ROUMEN T., APEL I., SHIGEYAMA J., MUHAMMAD A., BAUDISCH P.: Kerf-Canceling Mechanisms: Making Laser-Cut Mechanisms Operate across Different Laser Cutters. In *Proceedings of the 33rd Annual ACM Symposium on User Interface Software and Technology* (Virtual Event USA, Oct. 2020), ACM, pp. 293–303. URL: https://dl.acm.org/doi/10.1145/3379337.3415895, doi:10.1145/3379337.3415895. 2

[SBB*22] SCHMIDT P., BORN J., BOMMES D., CAMPEN M., KOBBELT L.: TinyAD: Automatic differentiation in geometry processing made simple. *Computer Graphics Forum 41*, 5 (2022). 4

[SDW*16] SONG P., DENG B., WANG Z., DONG Z., LI W., FU C.-W., LIU L.: CofiFab: coarse-to-fine fabrication of large 3D objects. *ACM Transactions on Graphics 35*, 4 (July 2016), 1–11. URL: https://dl.acm.org/doi/10.1145/2897824.2925876, doi:10.1145/2897824.2925876. 2

[SGC18] STEIN O., GRINSPUN E., CRANE K.: Developability of triangle meshes. *ACM TOG 37*, 4 (Aug. 2018), 1–14. doi:10.1145/3197517.3201303. 2

[Sta] STANFORD C. G. L.: Bunny model. URL: http://graphics.stanford.edu/data/3Dscanrep/. 8

[STL06] SHATZ I., TAL A., LEIFMAN G.: Paper craft models from meshes. *The Visual Computer 22*, 9-11 (Sept. 2006), 825–834. doi:10.1007/s00371-006-0067-6. 2

[VVHSH22] VERHOEVEN F., VAXMAN A., HOFFMANN T., SORKINE-HORNUNG O.: Dev2PQ: Planar Quadrilateral Strip Remeshing of Developable Surfaces. *ACM TOG 41*, 3 (June 2022), 1–18. doi:10.1145/3510002. 2

[Wan08] WANG C. C.: WireWarping: A fast surface flattening approach with length-preserved feature curves. *Computer-Aided Design 40*, 3 (Mar. 2008), 381–395. URL: https://linkinghub.elsevier.com/retrieve/pii/S0010448507002655, doi:10.1016/j.cad.2007.11.011. 2

[WJW*22] WANG C., JIANG C., WANG H., TELLIER X., POTTMANN H.: Architectural Structures from Quad Meshes with Planar Parameter Lines. *SSRN Electronic Journal* (2022). doi:10.2139/ssrn.4144321. 2

[YCS23] YUAN C., CAO N., SHI Y.: A Survey of Developable Surfaces: From Shape Modeling to Manufacturing, Apr. 2023. arXiv:2304.09587 [cs]. 2

[ZEK*17] ZARRINMEHR S., ETTEHAD M., KALANTAR N., BORHANI A., SUEDA S., AKLEMAN E.: Interlocked archimedean spirals for conversion of planar rigid panels into locally flexible panels with stiffness control. *Computers & Graphics 66* (Aug. 2017), 93–102. doi:10.1016/j.cag.2017.05.010. 2, 5

[ZFO*22] ZHAO Z.-Y., FANG Q., OUYANG W., ZHANG Z., LIU L., FU X.-M.: Developability-driven piecewise approximations for triangular meshes. *ACM TOG 41*, 4 (July 2022), 1–13. doi:10.1145/3528223.3530117. 2

[ZLS07] ZAYER R., LÉVY B., SEIDEL H.-P.: Linear angle based parameterization. In *Fifth Eurographics Symposium on Geometry Processing - SGP 2007* (Barcelone, Spain, July 2007), Alexander Belyaev M. G., (Ed.), Eurographics Association, pp. 135–141. 3, 4

[ZW11] ZHANG Y., WANG C. C. L.: WireWarping++: Robust and Flexible Surface Flattening With Length Control. *IEEE Transactions on Automation Science and Engineering 8*, 1 (Jan. 2011), 205–215. URL: http://ieeexplore.ieee.org/document/5492299/, doi:10.1109/TASE.2010.2051665. 2

Appendix A: Adaptive Kerf Cuts

Calculation of the locally required number of beams n_e from:

$$1 = \frac{\phi \cdot r}{l \cdot n_e} \cdot \frac{G}{\tau_{\text{yield}}}. \tag{9}$$

Given constants are G, τ_{yield}, l and the target angle ϕ. With rectangular beam crossections:

$$r = \frac{1}{2} \cdot \sqrt{h^2 + w_b^2} \tag{10}$$

$$w_b = w_e / n_e \tag{11}$$

We solve for n_e by squaring and substituting:

$$A = \left(\frac{\phi \cdot G}{l \cdot \tau_{\text{yield}}} \right)^2 \tag{12}$$

$$z = n_e^2 \tag{13}$$

$$z = A \cdot \left(\frac{h^2}{4} + \frac{w_e^2}{4 \cdot z} \right) \tag{14}$$

$$0 = z^2 - A \cdot \frac{h^2}{4} \cdot z - A \frac{w_e^2}{4} \tag{15}$$

$$z = A \cdot \frac{h^2}{8} + \sqrt{(A \cdot \frac{h^2}{8})^2 - A \frac{w_e^2}{4}} \tag{16}$$

$$n_e = \sqrt{z} \tag{17}$$