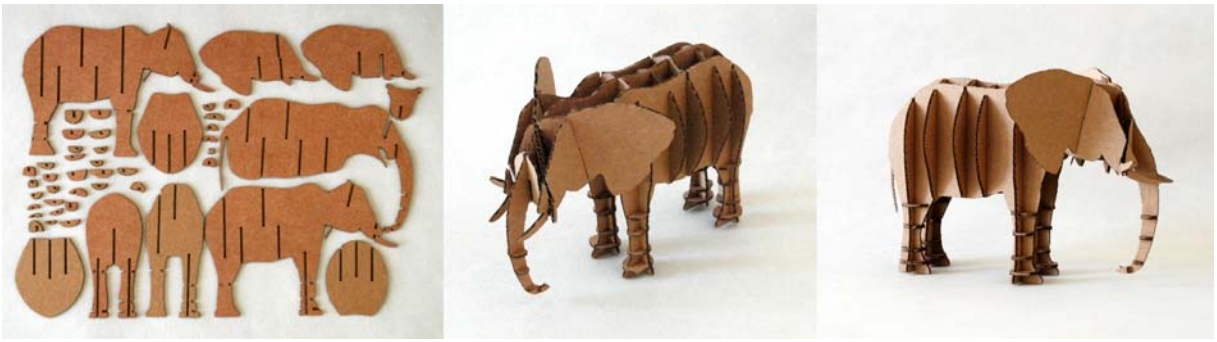
EUROGRAPHICS 2012 / P. Cignoni, T. Ertl
(Guest Editors)

Volume 31 (2012), Number 2

crdbrd: Shape Fabrication by Sliding Planar Slices

Kristian Hildebrand Bernd Bickel Marc Alexa

TU Berlin



Abstract

We introduce an algorithm and representation for fabricating 3D shape abstractions using mutually intersecting planar cut-outs. The planes have prefabricated slits at their intersections and are assembled by sliding them together. Often such abstractions are used as a sculptural art form or in architecture and are colloquially called 'cardboard sculptures'. Based on an analysis of construction rules, we propose an extended binary space partitioning tree as an efficient representation of such cardboard models which allows us to quickly evaluate the feasibility of newly added planar elements. The complexity of insertion order quickly increases with the number of planar elements and manual analysis becomes intractable. We provide tools for generating cardboard sculptures with guaranteed constructibility. In combination with a simple optimization and sampling strategy for new elements, planar shape abstraction models can be designed by iteratively adding elements. As an output, we obtain a fabrication plan that can be printed or sent to a laser cutter. We demonstrate the complete process by designing and fabricating cardboard models of various well-known 3D shapes.

1. Introduction

We present data structures and algorithms for the generation of cardboard sculptures: given as input the surface of a 3D object, we obtain a set of planar elements that approximates the object and can be physically fabricated by sliding each of the planes onto one another. Among many non-trivial aspects in this construction process we focus particularly on the difficult combinatorial problems resulting from respecting the physical fabrication of the model.

We show that only a small subset of all planar shape abstractions is constructible (see section 4). Furthermore, the slice insertion order and direction, which have a exponential combinatorial complexity, significantly influence the visual quality of the figure. As we show in the results section, a random order of planes is far from optimal due to required clipping of colliding parts.

Already the problem of finding good planes, regardless of whether the planes lead to a constructible model, is difficult. Automatically finding an optimal set of planes covering all important geometric features is challenging, as geometric cover problems are known to be NP-hard [Hoc97]. Addressing this problem, related work has suggested optimization heuristics [DDSD03], or very recently an approach that progressively selects planes to maximize feature coverage based on principles inferred from a user study [MSM11].

Figure 1: (Left to right) Abstraction of 3D models as cardboard cutout ©MOMA, H-Construction is a 'Slide-Together' geometric construction by George W. Hart ©George W. Hart, cardboard model by MUJI ©MUJI.

Our work is motivated by an increasing demand of real-world prototypes for visualization. While rapid prototyping technologies such as 3D printers can be used to fabricate realistic replicas, they are expensive, and the process is slow (printing time in the order of hours). In contrast, our proposed method allows fabricating 3D shape approximations fast and easily, and only requires equipment that is available in every office. Notably, while we share the goal of creating papercraft models from meshes, our approach significantly differs from related work that unfolds the surface [MS04] or tries to minimize the surface distance between 3D object and abstraction [MGE06, STL06]. In this sense, we trade realism with abstraction and fabrication complexity.

Based on an analysis of construction rules, we propose an extended binary space partitioning (BSP) tree that additionally includes the insertion direction of planes, as an efficient representation of such models. This data structure allows to evaluate the feasibility of newly added planar elements at any time in the insertion process. As an input, we start with a closed surface mesh of an arbitrary 3D object. In combination with a simple optimization and sampling strategy for new elements, planar shape abstraction models are designed by iteratively adding elements. As an output, we obtain a fabrication plan that can be printed or sent to a cutting device, such as a cut plotter or laser cutter. We demonstrate the complete process by designing and fabricating cardboard models of various well-known 3D shapes.

The contributions of our research can be summarized as follows:

- We introduce a novel representation for *cardboard models* based on an extended binary space partitioning data structure.
- We propose a set of construction rules that respect physical constraints and guarantee that every piece can be slide onto the current construction.
- We present an automatic pipeline to generate constructible piece-wise planar shape abstractions and demonstrate its functionality with a variety of 3D models.

2. Related Work

Computer graphics and related fields have extensively studied processes for computing efficient shape representations and simplifications. Our work is most closely related to computational paper architecture and methods that try to approximate surface patches with a low number of simple primitives. For a general survey on mesh simplification techniques we refer to [LRC*02].

Abstraction and Shape Decomposition is often used as a method for effective visual communication. Mi et al. [MDS09] propose a part-based representation and method for decomposing a 2D shape into a few simple parts that reveal important features. DeCarlo and Stone [DS10] show that the abstract shape is understood in the same way as the detailed one and shares a common visual explanation for the important features. This concept was extended to the idea of an exoskeleton as an abstraction of shapes [dGGDV10]. The exoskeleton as the external shell is a combination of geometry and perceptual approximations that result in a set of disk-like patches. In the same spirit, Mehra et al. [MZL*09] extract only the characteristic curves of 3D man-made shapes to provide a compact and representative version of the models. Pushing the level of abstraction even further, Decoret et al. [DDSD03] suggest billboard clouds as an extremely simplified but still powerful representation for 3D objects. Similar to our approach in this work, planes are used as basic primitives, but are treated as an unstructured set and benefit from image-based impostors. Recently, McCrae et al. [MSM11] proposed a powerful approach for generating shape-proxies consisting of planar sections based on principles inferred from user studies. Although the generation of paper puppets as an potential application is shown, there is no guarantee that the computed shape-proxies are physically constructible only by sliding slices. In general, common to all of these methods is their ability to efficiently represent shape with simple primitives. However, these representations are not developed and directly suited for physical construction. As demonstrated in the results section, our approach can be applied in combination with various sampling strategies for planar sections and therefore can be seen orthogonal to these methods.

Computational Models for constructing and designing papercrafts received attention in the computer graphics community. Existing methods can be classified by the type of basic elements used for fabrication. For example e.g. Li et al. [LYMS07] have shown that models can be augmented with papercut patterns to support the perception of texture. In contrast, we will focus in the following on approaches that address forming 3D shapes.

Origami. Papercrafting in art has a long history and dates back nearly 2000 years to the invention of paper. Origami, the art of paper folding, creates intricate structures from a flat piece without cutting or gluing. An overview and in-

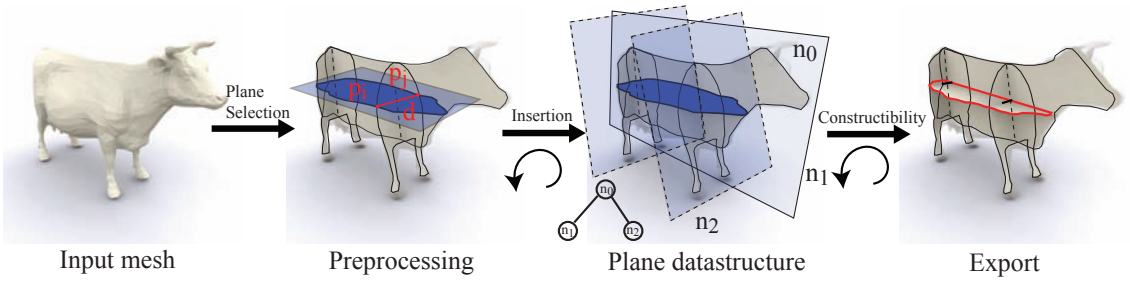


Figure 2: Overview of the construction process. Our pipeline starts with an input mesh. We then perform a preprocessing step which generates a set of polygons that are added iteratively to the cardboard model. By inserting the polygon into our data structure, we split the polygon into individual pieces and efficiently test if those can be physically slid into the cardboard model. If successful the process is finalized by exporting the 2D fabrication plan.

troduction to mathematical folding algorithms can be found in [DO07].

Decomposition into strips. There are several methods that approximate a 3D object with a set of paper strips that can be folded and glued. Mitani and Hiromasa [MS04] segment the input mesh and represent it with a set of strips that can be crafted by bending the paper which also allows to represent smooth features. [MGE06] use a set of developable surfaces each one being a generalized cylinder represented as a strip of triangles. [STL06] follow a similar approach but restrict their elements to cones and planes. In theory, directly cutting and unfolding a polygonal mesh is trivial, but the large number of resulting segments and triangles make this approach impractical for fabrication unless used in combination with geometry simplification techniques [CSAD04, SS10]. In general, all these methods try to minimize the error between the original mesh and the abstracted papercraft model.

Pop-up design. This area investigates the creation of paper models from planar paper layouts that can be popped-up in a rigid and stable manner. Recently, Li et al. [LSH*10] presented an algorithm for computing paper architecture. The shape of buildings is approximated as a set of parallel planes. Another interesting class are v-style pop-ups, which can be opened and closed, i.e. moved into a flat state, without changing the rigidity of the structure or extra force except at two patches. These pop ups can be automatically generated [MS03, LJGH11] and are used for books or card design [Gla02]. While pop-ups are inherently intriguing and mathematically interesting, fabrication might require gluing and the assembly complexity of such models is higher compared to our approach of sliding parts.

Fabrication. Investigating the aspects of intuitive sketching and design, [SLMI11, LIMS10] present innovative systems for furniture design and fabrication. In theory, our representation and approach could be combined with such sketching tools. Notably, Xin et al. [XLF*11] propose a system to decompose a 3D shape into several interlocking parts, allowing to automatically generating burr puzzles. Horoyd et al. [HBLM11] fabricate multilayer models: a parallel stack of 2D images embedded within a semi-transparent medium. In the wider context, computer graphics recently presented several methods for fabricating real-world objects with custom properties. These address the appearance, for example by fabricating microgeometry for surface reflectance [WPMR09], printing BRDFs [MAG*09], or physically reproducing subsurface scattering [HFM*10, DWP*10] and deformation behavior [BBO*10]. In this spirit, our work extends the set of digital fabrication possibilities, and provides a low-cost alternative to 3D printing.

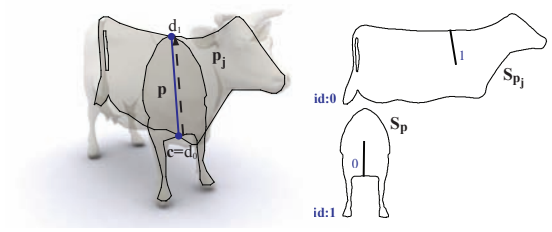


Figure 3: We define p_j as the *pivot polygon* of p and $\vec{d}$ as the *pivot direction* with the *pivot point* c. S_q and S_p define the extracted tiles in the output fabrication plan. We assign each tile an *id* and annotate each slit with the *id* that is connected to the slit.

3. Overview

Given the surface of a connected component in $\mathbb{R}^3$ as input, our goal is to create a *cardboard model* abstraction that is physically constructible.

The output is a set of n planar polygons $p = \{p_1, ..., p_n\}$ including intersection slits. The process is illustrated in Figure 2.

Our design process iteratively generates planar polygons p by intersecting a plane against the input object and already

existing polygons. To define the cardboard model, we specify for each polygon $p_i, i = 2,...,n$ an already existing polygon $p_j, j < i$ to which it is connected to (see Figure 2).

Physically inserting a polygon p from a specific direction is only possible if the insertion path is not blocked. To represent our abstracted object at any time and allow for efficient tests, we simplify the full insertion analysis to a straight line path for the inserted plane. This allows us to perform all tests based on a modified version of a *BSP tree* [FKN80].

Note that the full analysis of all possible insertion paths would be significantly more complex algorithmically, and also not lead to a similar elegant data structure. We are quite sure that the number of cases in which a curved path for insertion of elements would add something to a model are very small.

Linking a polygon p with another polygon p_j that is already part of the cardboard figure defines an intersection segment between p and p_j. Figure 3 shows p_j as the *pivot polygon* for p. In other words the pivot polygon is the polygon we slide on. The polygon intersection segment defines a *pivot point* c and a *pivot direction* $\vec{d}$. This intersection segment defines the insertion slit on the polygons p and p_j. In addition to the data structure, we derive and analyze a set of construction rules that go along with the development of a cuttable layout as shown in Figure 3 and described in sections 4.3 and 4.5.

Our process starts with sampling a set of candidate polygons for insertion. For each of these polygons, we check the insertability and compute a quality measure that tries to quantify the additional features which are covered by this plane. Since the order of insertion is crucial for the outcome we use a branch and bound strategy to determine the best model. Finally, we generate the output layout, which can be printed and cut manually, sent to a cutting plotter, or laser cutter. We tested our pipeline on a large number of models. A subset is shown in Figures 14 and 15.

Keep in mind, as cardboard models become more complex, the insertion ordering cannot be trivially solved. A fully area-preserving solution may not exist even for simple constructions (Figure 4). While in this example it is irrelevant which part is dropped for symmetry reasons, in most real world models the insertion order leads to significantly different outcomes, e.g. Figure 10.

4. Construction

Our input shape is a single connected component in $\mathbb{R}^3$ that is true to scale of our desired constructible output. Consider a current construction state as shown in Figure 2. When inserting an additional polygon p to a cardboard sculpture, there are three possible outcomes:

- The polygon p can be directly put on the pivot polygon p_j and any other polygon with an intersection segment parallel to the insertion direction $\vec{d}$. There exists no polygon blocking the insertion path, so p can be inserted completely.
- We insert p and there is at least one already existing polygon that blocks the insertion path of p except the polygon p_j or any other polygon with an intersection segment parallel to $\vec{d}$. We then split p in parts and insert the parts separately from the directions $\vec{d}$ and $-\vec{d}$. Our data structure guides the splitting process.
- Polygon p is split, but at least one of the resulting planar elements cannot be slit into the cardboard model due to its shape as shown in Figure 7 or the resulting intersection slits that would endanger the stability of the sculpture, as described in Section 4.3 and 4.5.

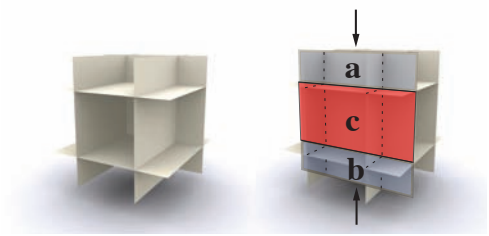


Figure 4: We show an insertion for a symmetric cardboard construction. Note, that the consecutive plane cannot be inserted completely – part c is physically not insertable. Top and bottom parts a and b of the plane can be inserted. Due to symmetry this means that this construction cannot be realized without losing parts of the planes.

In the following subsections we describe each of these cases and our construction algorithm in more detail.

4.1. Preprocessing

We start our iterative construction process by selecting a plane and generating flat polygons defined by the intersection curves of the plane with the original 3D model. Each slice contains a set of connected components, of which each is interpreted as an individual closed polygon p. The outline contours and possible inner hole contours are triangulated. The triangulated plane geometry is used for subsequent intersection computations. Note, as we will describe later in more detail, for all polygons (connected components) on a plane we will test if they can be physically inserted. Only if this is the case, we will consider them as potential candidates for the cardboard model.

4.2. Cardboard Model Data Structure

Before we insert the polygon p into our data structure, we decide on a *pivot polygon* p_j with which we want to link to p. With the choice of p_j we compute the intersection segment between both polygons and identify its endpoints. The endpoints define our *pivot direction* $\vec{d}$ and the *pivot point* c.

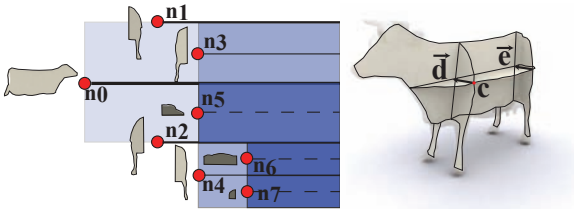


Figure 5: The BSP tree for the cardboard model on the right. The nodes n_i store its part of the polygon geometry and divide the scene in half-spaces.

To maintain the cardboard model geometry we utilize a data structure. It is based on a *Binary Space Partitioning*, which we extend by modifying the insertion algorithm with a set of additional rules. We want to restrict the insertion to geometry that can be slid in safely until an existing plane is blocking the insertion path. Note that this is different from the standard version: in our version planes are not necessarily inserted on both sides of each existing plane, but the behavior rather depends on the type of node.

We also consider intersecting several existing polygons. This is possible if the intersection segment with other planes is parallel to the pivot insertion direction $\vec{d}$ of p.

Figure 5 illustrates the insertion process with the help of a simple example. The insertion of geometry into a BSP is done by recursively traversing the existing nodes n. We follow the traversal path and add new tree nodes:

- in both half-spaces if node n_i contains the pivot polygon or if $\vec{d}$ is parallel to $\vec{e}$, where $\vec{e}$ is the of intersection segment with the polygon in n_i. Since we want to attach on it we can leave the polygon geometry as is.
- in both half-spaces if the new polygon is not intersecting with the polygon in node n_i.
- only in the half-space that contains c. So, only the geometry in the half-space of the pivot point can be inserted.

We store c and $\vec{d}$ for each node n in the BSP. Furthermore n holds a reference to the cardboard model polygon data type and the part of the polygon representing the plane (see Figure 5).

The choice of the pivot polygon, the insertion order and insertion direction change the resulting cardboard model significantly because existing planes block parts in the shape for consecutive planes which potentially leads to discarding parts of these planes during insertion (see Figure 4). Finding an optimal solution is very complex. Therefore we apply an insertion order optimization described in Section 5. Figure 6 illustrates in a simple example the influence of the insertion order and the choice of the pivot polygon.

We now have a correctly clipped planar element p. In order to ensure that we can use p as a valid element for our cardboard sculpture we need to check if the element is *castable*.

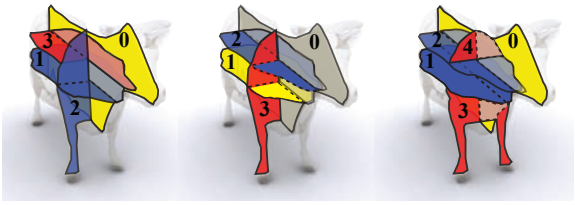


Figure 6: Insertion order makes a difference due to already existing polygons. In red we highlight the last inserted plane, in yellow we show its pivot polygon.

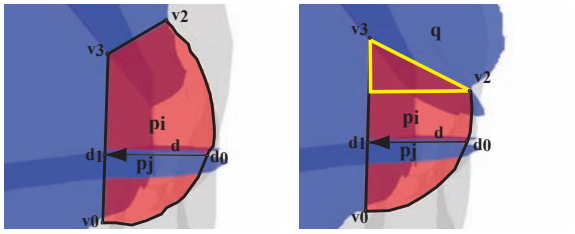


Figure 7: Left: This polygon p is is castable with respect to insertion direction $\vec{d}$. Right: The yellow part of polygon p indicates that it is not castable with respect to the insertion direction $\vec{d}$.

4.3. Castability of Planes

When inserting p into the BSP, it is clipped against other planes that block the insertion path. This clipping process generates a set of clipping points $\vec{c}_k$. Polygon p can only be physically moved in direction $\vec{d}$ when the *signed* distance of all points $\vec{c}_k$ to the insertion direction is monotone along the clipping path (see Figure 7). We note that while the values of the distances may depend on the origin of the computation (e.g. the slit), their monotonicity just depends on the direction. Consequently, we compute the projection vector $\vec{c}_k - (\vec{d}\cdot\vec{c}_k)\vec{d}$ of $\vec{c}_k$ onto $\vec{d}$ and then take the cross product with $\vec{d}$ to get a signed distance value:

$$\vec{d}\times(\vec{c}_k-(\vec{d}\cdot\vec{c}_k)\vec{d}) = \vec{d}\times\vec{c}_k \qquad (1)$$

We check that these values are monotone with the index k.

Figure 7 shows a valid and invalid clipping configuration. The yellow triangle on the right figure indicates a part that violates the condition, because the distance to $\vec{d}_1$ does not decrease for the clipping points $\vec{v}_2$ and the subsequent $\vec{v}_3$. We like to stress that casting is, unfortunately, a global condition, in the following sense: if a polygon p cannot be inserted by moving it in direction $\vec{d}$ it could potentially cut any other polygon in the model during the movement as well. This means we cannot fix the problem just by local modifications, such as just cutting the interfering polygon q in the clipping path as shown in Figure 7.

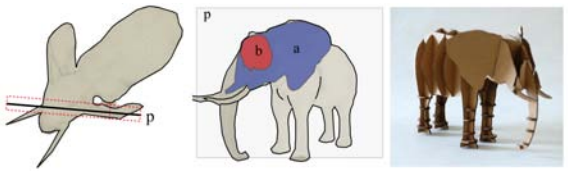


Figure 8: Left: Planar slice p through the curved ear of the elephant model viewed from above. Middle: Resulting planar slice with point gathering a and without b. Right: Fabricated result including the complete elephant ear.

4.4. Gathering Shape Features

So far, we defined the base elements of our cardboard sculptures as planar and within the volume of the input object. Therefore, abstractions of curved thin shells result in a large number of elements. To compensate for a possible loss of abstraction, we propose a more efficient representation by gathering all surface points within a user specified distance from the plane to account for the element thus allowing elements outside the object volume. Intuitively, one can think of it as the projection onto the plane. Figure 8 shows a fabricated example where surface points are gathered for the manually selected elephant ears to complete its geometry.

4.5. Fabrication Plan

In order to physically construct the cardboard model our algorithm exports a printable 2D fabrication plan including cut line annotations, as shown in Figure 3 and in the additional material.

A cut line is the intersection slit between two polygons. It is divided into two parts, resulting in a halfway-cut for each intersecting polygon. In order to make sure that we can fabricate the polygons with slits relative to the thickness of the material it is important to obey a set of simple conditions:

- The intersection slit is not intersecting more than once with the contour of polygon p. Otherwise the tile could be disconnected.
- The slits need to have a minimum connection length with the pivot polygon to support physical stability of the tile and the resulting model.

We add additional annotations to provide step-by-step instructions for assembly of the cardboard sculpture. When physically constructing the cardboard model by hand one simply has to follow the linkage order. We refer the reader to the additional material of the paper that includes a set of fabrication plans.

5. Automatic Construction and Evaluation

Once the basic construction constraints are in place we need to decide on the order of the construction. For a given set of n planes there exists a large number of combinations to assemble the cardboard model. Specifically there is the plane order permutation of $n!$ and the insertion directions given a specific plane order. Some of these combinations lead to the same outcome even though their cardboard model tiles and the order of insertions are different. In order to find an optimal construction we propose a branch-and-bound approach that utilizes a construction tree as shown in Figure 9 with depth n to automatically construct a cardboard model.

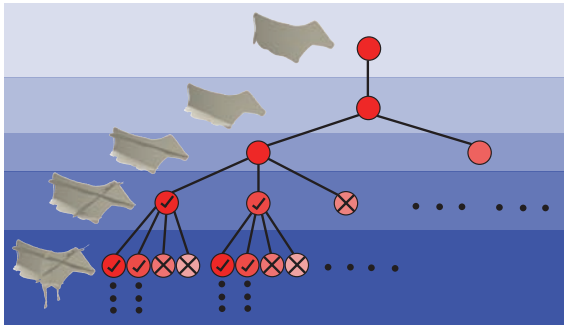


Figure 9: Given a new planar element we evaluate all possible insertions and sort the result by its projected area. We use branch-and-bound and proceed our evaluation with the best leaf models. This reduces the number of construction possibilities drastically.

From an aesthetic point of view it often is desirable to construct the cardboard model with as much of the available polygon area as possible. Therefore, we sort all planes by a score value and start inserting them by the decreasing score on each possible already existing polygon.

This leads to a number of intermediate cardboard models as nodes in the construction tree which we again sort by their score. Depth-first search quickly provides useful bounds and allows pruning of suboptimal solutions. Furthermore, we discard less significant intermediate construction results working only with a set of k nodes for further insertion. This reduces our combination space drastically. We found that $k \in [2,4]$ already gives good results even though it is not guaranteed that this is the optimal solution. A set of possible outcomes is shown in Figure 11 and Figure 12.

In practice, we found that planes close to the surface are important for resembling the silhouette. In addition, we therefore weight the plane's area by its distance to the surface, as shown in Figure 10.

6. Plane Selection

Selecting a set of representative planes is challenging because the visual quality of the resulting figure is dependent on factors such as coverage of geometric features, human perception, and visual aesthetics. In theory, our construction method works in combination with any plane selection algorithm. For our pipeline, we incorporated various sampling strategies, including axis-aligned grid sampling, a simple

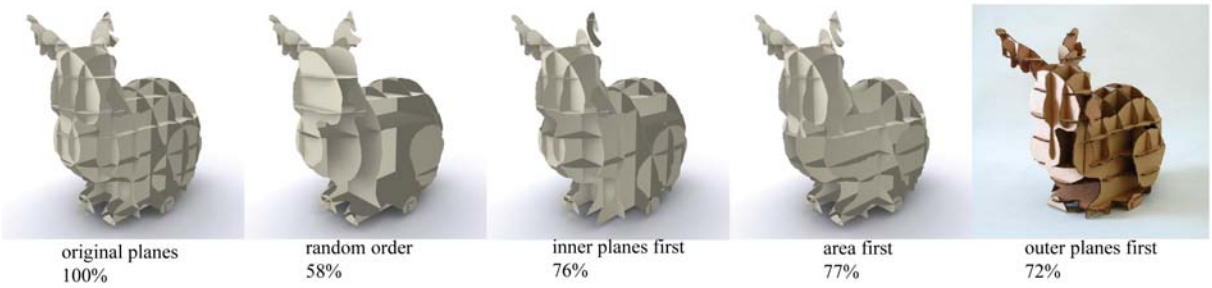


Figure 10: Axis-aligned plane sampling in an 8x8x8 grid. The order of insertion is determined by the decreasing score of the plane. The score is a weighting of area and the polygons distance from the surface. Therewith we control the construction order from inside to outside. If the score is equal for all polygons the insertion is random. It is noticeable that a construction starting with the larger outer planes reconstructs the silhouette much better with nearly the same overall area.

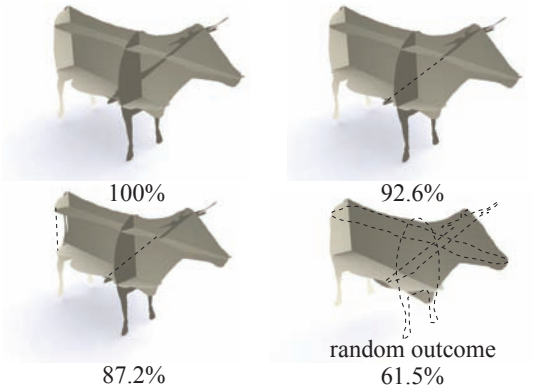


Figure 11: Given a set of planes a number possible outcomes and its insertion area in percentage for a specific insertion order is shown. The random outcome represents the median of 100 random insertion orders. Dashed lines indicate parts that could not be added because of a non-optimal insertion order.

yet powerful automatic process to generate general planar abstraction that supports visual regularity patterns, and the option for manual plane selection. We also generate results based on a sampling strategy learned from user data as recently proposed by McCrae et al. [MSM11] as can be seen in Figure 12.

6.1. Axis-aligned Grid Sampling

Given a number of equidistant planes, we sample the axis-aligned bounding box for each axis of the 3D shape by sweeping the planes through the model searching for the largest overall cross section area. As this method does not take into account any knowledge about the shape itself, it usually requires a larger number of planes compared to more sophisticated methods, but the resulting model look aesthetically pleasing, compact, and regular. This approach works

Figure 12: Left, middle: Given a set of planes generated by [MSM11] we apply our insertion strategy in an optimal (left) and not optimal (right) construction case. Right: Cardboard model [MSM11]

well for 'low-frequency' objects as shown in Figure 14, but for reproducing thin features, the number of required planes makes construction impractical.

6.2. Plane Quality based Sampling

For an adaptive sampling strategy, we require a quality value μ for each polygon in plane space Γ. We propose a measure based on two importance factors - a measure of symmetry and how well a polygon covers geometric shape features. Both measures can be efficiently evaluated in image space on the GPU (see Figure 13).

The *distance symmetry term* $D \in [0,1]$ encodes the difference of distances from a point on the plane to its two projected points on the mesh along the plane normal in front and back space of the plane with $D = 1 - (d_f - d_b)$.

The *surface symmetry term* $I \in [0,1]$ is a measure of correspondence between the mesh surface normal $\vec{l}$ and the plane normal $\vec{n}$. It is computed with $1 - (\vec{n_f} \cdot \vec{l_f}) - (\vec{n_b} \cdot \vec{l_b})$. We would like to favor planes with normals oriented in its average surface direction. Figure 13 shows the resulting image buffers for both terms over the polygon p_i.

In order to compute the quality μ over the mesh surface Ω for the polygon we define

$$\mu = \int_{\Omega} V \cdot I \cdot D$$

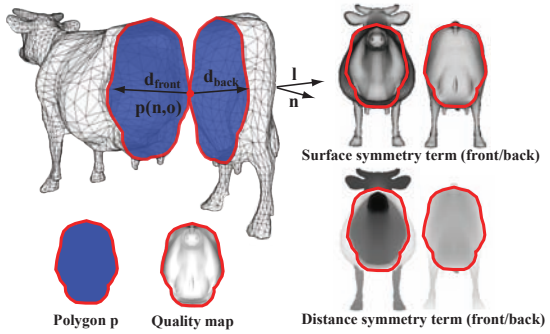


Figure 13: Evaluating the quality of the plane. Evaluation buffers: Distance in front and back of the plane, i.e. the distance symmetry term D. Gouraud shading with the light source pointing opposite to the plane normal results in the surface symmetry term I. We sum over the resulting quality map to compute our quality measure.

with $V \in \{0,1\}$ as *visibility term*. During an iterative selection of planes each plane covers geometric shape features by its projected volume. The surface covered by the volume is marked as invisible $V = 0$ for consecutive planes. It is unity only for surface parts that are not yet visited and lie in the projection of the polygon in the plane normal direction. Figure 13 shows the different results for D, I and the quality result.

We represent potential polygons by the normal direction $\vec{n}$ and the offset $o \in [0,b]$ to the origin, where b is the radius of the 3D model bounding sphere. We uniformly sample the directions for $\vec{n}$ by choosing points on unit sphere using Saff et al.'s [SK97] point distribution algorithm. We define our plane space Γ for all samples each with a set of offsets.

Given the quality measure for each polygon, we evaluate the set of planes of Γ and store our evaluated candidate polygons in a *priority queue*. We greedily pick the polygon with the highest measure μ and add it to a list of polygons for the construction process. Since each new polygon p separates a volumetric part of the shape, we mark this part as covered over the mesh progressively with each p. As we are interested in planes that can be slid onto each other, we restrict the sampling to attach only polygons that are nearly orthogonal to p_j, e.g. $n_{p_j} \cdot n_p \leq \varepsilon$, with $\varepsilon < 0.02$.

We proceed with iteratively selecting planes until a the mesh is completely covered or a maximum number of planes is reached.

6.3. Manual Plane Selection

Choosing aesthetically pleasing planar elements for our abstraction is very subjective. An automatic process often does not meet the standard of a user selection, especially when the 3D model is filigree and has very distinct geometric features that cannot be easily represented in a plane, or requires a higher-level understanding of construction. Therefore, we offer a very simple polygon insertion interface that allows to edit the cardboard model within a few minutes and can be used in combination with our automatic approach.

7. Results and Discussion

We tested our approach on numerous 3D objects and fabricated several cardboard models. The fabricated objects contain between 7 (Armadillo in Figure 15) and 48 (Stanford Bunny in Figure 14) elements made out of standard paper, cardboard, plywood or plastic and were manufactured using an Epilog Zing Lasercutter within 5 minutes and were assembled within 5 to 15 minutes. Side-by-side comparisons of the input model and the real fabricated cardboard models are shown in Figure 15 and 14 and the accompanying video. We observed that even with a few polygons the shape of 3D models can be approximated quite well. In fact, these models are extremely low-cost, only require printing and cutting a layout, and could even be created by children. However, our algorithm is not restricted to a low number of polygons, although physical assembly becomes impractical at some point.

For all automatically generated results shown in this paper we either used a grid layout of up to 8x8x8 planes (see Figure 14) or the plane quality evaluation or a combination of both (see Figure 15). For the plane quality evaluation we sampled the plane space approximately with 4000 normal directions and 100 distance offsets. Evaluating the quality of candidate polygons requires the majority of the computation time. For an input model complexity of about 100k triangles, our single-threaded algorithm evaluates about 130 candidates per second, resulting in a total processing time of about one hour on a MacPro using a GeForce GT 120 graphics card. Using the axis-aligned grid sampling we generate a cardboard model within seconds.

As shown in Figure 15, our approach for estimating the plane quality is robust and effective. However, it does not take into account higher level design goals or knowledge about the object itself. We therefore also provide a simple interface, allowing the user to indicate preferred samplings as done for the elephant ears in the teaser.

Limitations and Future Work.

Currently, our plane quality estimation algorithm does not incorporate high level information about the input object such as salient features, symmetry, or texture. Although finding a general quality estimation that respects the aesthetics of the input and output object might be hard, for future work one might consider combining our representation and approach with work in symmetry detection and enhancement [PMW*08] or shape perception.

Figure 14: We show the input meshes and the handcrafted cardboard models using axis-aligned grid sampling to find a set of planes. The process for all models is automatic except for the gorilla where we added an additional plane manually.

Figure 15: Resulting cardboard sculptures created from a completely automatic two-step process. The first step finds to best *skeleton polygons* using the plane quality evaluation. In the second step we additional sample axis-aligned planes. The Armadillo is created only of skeleton polygons.

8. Conclusion

We have presented a novel algorithm and representation for designing and fabricating cardboard sculptures from 3D models. The core component of our pipeline is the efficient construction and representation of such models which guarantees that they can be physically assembled. Fabricating such models is extremely low-cost and simple. We therefore think our method could be appealing to a large audience, and might have impact in areas such as architecture and design where our approach could be an alternative to 3D printing. Our method is a step towards low-cost but widely applicable shape abstraction and fabrication. We hope that our representation and construction algorithm will inspire future work in the area of geometry processing, optimization, and perception.

References

[BBO*10] BICKEL B., BÄCHER M., OTADUY M. A., LEE H. R., PFISTER H., GROSS M., MATUSIK W.: Design and fabrication of materials with desired deformation behavior. *ACM Trans. Graph. 29*, 4 (July 2010), 63:1–63:10. 3

[CSAD04] COHEN-STEINER D., ALLIEZ P., DESBRUN M.: Variational shape approximation. *ACM Transactions on Graphics 23*, 3 (Aug. 2004), 905. 3

[DDSD03] DÉCORET X., DURAND F., SILLION F. X., DORSEY J.: Billboard clouds for extreme model simplification. *ACM Trans. Graph. 22*, 3 (July 2003), 689. 1, 2

[dGGDV10] DE GOES F., GOLDENSTEIN S., DESBRUN M., VELHO L.: Exoskeleton: Curve network abstraction for 3D shapes. *Computers & Graphics* (Nov. 2010). 2

[DO07] DEMAINE E. D., O'ROURKE J.: *Geometric Folding Algorithms. Linkages, Origami, Polyhedra*. Cambridge University Press, 2007. 3

[DS10] DECARLO D., STONE M.: Visual explanations. In *Proc. Int. Symp. on Non-Photorealistic Animation and Rendering* (New York, NY, USA, 2010), NPAR '10, ACM, pp. 173–178. 2

[DWP*10] DONG Y., WANG J., PELLACINI F., TONG X., GUO B.: Fabricating spatially-varying subsurface scattering. *ACM Trans. Graph. 29*, 4 (July 2010), 62:1–62:10. 3

[FKN80] FUCHS H., KEDEM Z. M., NAYLOR B. F.: On visible surface generation by a priori tree structures. *SIGGRAPH Comput. Graph. 14*, 3 (July 1980), 124–133. 4

[Gla02] GLASSNER A.: Interactive pop-up card design. 1. *IEEE Computer Graphics and Applications 22*, 1 (2002), 79–86. 3

[HBLM11] HOLROYD M., BARAN I., LAWRENCE J., MATUSIK W.: Computing and fabricating multilayer models. *ACM Trans. Graph. 30* (Dec. 2011), 187:1–187:8. 3

[HFM*10] HASSAN M., FUCHS M., MATUSIK W., PFISTER H., RUSINKIEWICZ S.: Physical reproduction of materials with specified subsurface scattering. *ACM Trans. Graph. 29*, 4 (July 2010), 61:1–61:10. 3

[Hoc97] HOCHBAUM D. S. (Ed.): *Approximation algorithms for NP-hard problems*. PWS Publishing Co., Boston, MA, USA, 1997. 1

[LIMS10] LIN J., IGARASHI T., MITANI J., SAUL G.: A sketching interface for sitting-pose design. In *Proc. Sketch-Based Interfaces and Modeling Symposium* (Aire-la-Ville, Switzerland, Switzerland, 2010), SBIM '10, Eurographics Association, pp. 111–118. 3

[LJGH11] LI X.-Y., JU T., GU Y., HU S.-M.: A geometric study of v-style pop-ups: Theories and algorithms. *ACM Trans. Graph. 30*, 4 (2011), 98:1–10. 3

[LRC*02] LUEBKE D., REDDY M., COHEN J., VARSHNEY A., WATSON B., HUEBNER R.: *Level of Detail for 3D Graphics*. Morgan Kaufmann, 2002. 2

[LSH*10] LI X.-Y., SHEN C.-H., HUANG S.-S., JU T., HU S.-M.: Popup: automatic paper architectures from 3D models. *ACM Trans. Graph. 29*, 4 (July 2010), 1. 3

[LYMS07] LI Y., YU J., MA K.-L., SHI J.: 3D paper-cut modeling and animation. *Computer Animation and Virtual Worlds 18*, 4-5 (Sept. 2007), 395–403. 2

[MAG*09] MATUSIK W., AJDIN B., GU J., LAWRENCE J., LENSCH H. P. A., PELLACINI F., RUSINKIEWICZ S.: Printing spatially-varying reflectance. *ACM Trans. Graph. 28*, 5 (Dec. 2009), 128:1–128:9. 3

[MDS09] MI X., DECARLO D., STONE M.: Abstraction of 2D shapes in terms of parts. *Proc. Int. Symp. on Non-Photorealistic Animation and Rendering - NPAR '09 1*, c (2009), 15. 2

[MGE06] MASSARWI F., GOTSMAN C., ELBER G.: Papercraft Models using Generalized Cylinders. *Proc. Pacific Conference on Computer Graphics and Applications* (2006). 2, 3

[MS03] MITANI J., SUZUKI H.: Computer aided design for 180-degree flat fold origamic architecture with lattice-type cross sections. *Journal of Grap. Science of Japan 37*, 3 (2003), 3–8. 3

[MS04] MITANI J., SUZUKI H.: Making papercraft toys from meshes using strip-based approximate unfolding. *ACM Trans. Graph. 23*, 3 (Aug. 2004), 259. 2, 3

[MSM11] MCCRAE J., SINGH K., MITRA N. J.: Slices: A shape-proxy based on planar sections. *ACM Trans. Graph. 30*, 6 (2011), to appear. 1, 2, 7

[MZL*09] MEHRA R., ZHOU Q., LONG J., SHEFFER A., GOOCH A., MITRA N. J.: Abstraction of man-made shapes. *ACM Trans. Graph. 28*, 5 (Dec. 2009), 1. 2

[PMW*08] PAULY M., MITRA N. J., WALLNER J., POTTMANN H., GUIBAS L.: Discovering structural regularity in 3D geometry. *ACM Trans. Graph. 27*, 3 (2008), #43, 1–11. 8

[SK97] SAFF E. B., KUIJLAARS A. B. J.: Distributing many points on a sphere. *The Math. Intelligencer 19*, 1 (1997), 5–11. 8

[SLMI11] SAUL G., LAU M., MITANI J., IGARASHI T.: SketchChair: an all-in-one chair design system for end users. In *Proc. Int. Conf. on Tangible, Embedded, and Embodied Interaction* (New York, NY, USA, 2011), TEI '11, ACM, pp. 73–80. 3

[SS10] SINGH M., SCHAEFER S.: Triangle surfaces with discrete equivalence classes. *ACM Trans. Graph. 29*, 4 (July 2010), 1. 3

[STL06] SHATZ I., TAL A., LEIFMAN G.: Paper craft models from meshes. *The Visual Computer 22*, 9-11 (Aug. 2006), 825–834. 2, 3

[WPMR09] WEYRICH T., PEERS P., MATUSIK W., RUSINKIEWICZ S.: Fabricating microgeometry for custom surface reflectance. *ACM Trans. Graph. 28*, 3 (July 2009), 32:1–32:6. 3

[XLF*11] XIN S., LAI C.-F., FU C.-W., WONG T.-T., HE Y., COHEN-OR D.: Making burr puzzles from 3d models. *ACM Trans. Graph. 30*, 4 (August 2011), 97:1–97:8. 3