Symmetry-driven 3D Reconstruction from Concept Sketches

Felix Hähnlein
felix.hahnlein@inria.fr
Inria, Université Côte d'Azur
France

Yulia Gryaditskaya
yulia.gryaditskaya@gmail.com
Surrey Institute for People Centred AI and Centre for Vision, Speech and Signal Processing (CVSSP), University of Surrey
United Kingdom

Alla Sheffer
sheffa@cs.ubc.ca
University of British Columbia
Canada

Adrien Bousseau
adrien.bousseau@inria.fr
Inria, Université Côte d'Azur
France

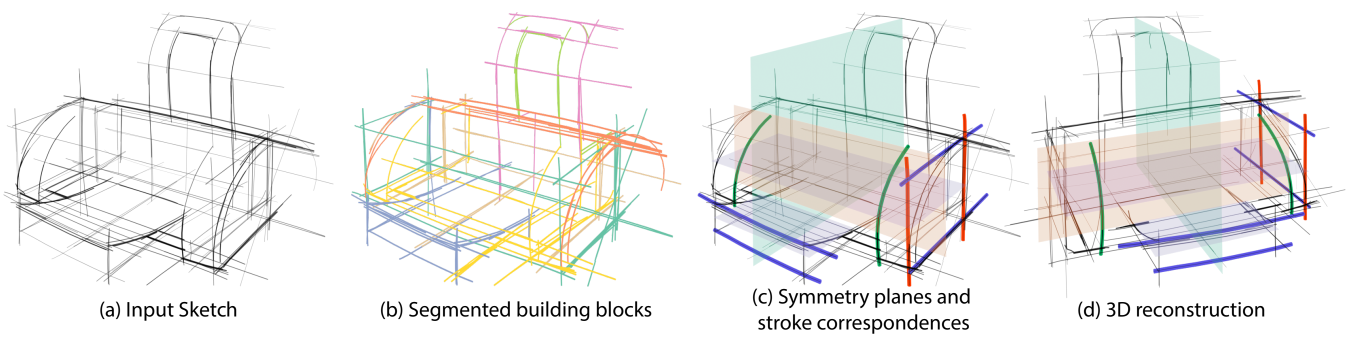


Figure 1: Concept sketches are dominated by symmetric strokes both in the shapes they depict, and in the construction lines employed to draw these shapes in perspective (a). Our algorithm first decomposes the sketch into locally-symmetric groups of strokes (b), and proceeds to identify pairs of strokes that are symmetric with respect to triplets of axis-aligned planes (c). At its core, our method selects the stroke correspondences that result in the most symmetric and well connected shape (d). We only show a subset of the symmetry planes and correspondences for illustration purpose.

ABSTRACT

Concept sketches, ubiquitously used in industrial design, are inherently imprecise yet highly effective at communicating 3D shape to human observers. We present a new symmetry-driven algorithm for recovering designer-intended 3D geometry from concept sketches. We observe that most concept sketches of human-made shapes are structured around locally symmetric building blocks, defined by triplets of orthogonal symmetry planes. We identify potential building blocks using a combination of 2D symmetries and drawing order. We reconstruct each such building block by leveraging a combination of perceptual cues and observations about designer drawing choices. We cast this reconstruction as an integer programming problem where we seek to identify, among the large set of candidate symmetry correspondences formed by approximate pen strokes, the subset that results in the most symmetric and well-connected shape. We demonstrate the robustness of our approach by reconstructing 82 sketches, which exhibit significant over-sketching, inaccurate perspective, partial symmetry, and other imperfections. In a comparative study, participants judged our results as superior to the state-of-the-art by a ratio of 2:1.

CCS CONCEPTS

• **Computing methodologies → Shape modeling**.

KEYWORDS

industrial design, sketches, line drawing, sketch-based modeling, 3D reconstruction, symmetry

ACM Reference Format:
Felix Hähnlein, Yulia Gryaditskaya, Alla Sheffer, and Adrien Bousseau. 2022. Symmetry-driven 3D Reconstruction from Concept Sketches. In *Special Interest Group on Computer Graphics and Interactive Techniques Conference Proceedings (SIGGRAPH '22 Conference Proceedings), August 7–11, 2022, Vancouver, BC, Canada.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3528233.3530723

1 INTRODUCTION

Concept sketches are commonly used in industrial design, both to explore 3D shape ideas during the design process itself and to communicate these ideas to colleagues and clients via drawings

SIGGRAPH '22 Conference Proceedings, August 7–11, 2022, Vancouver, BC, Canada

that leverage human ability to ideate a shape from a set of descriptive, semantically diverse, yet inexact curves. While humans can easily parse a concept sketch, reconstructing a 3D shape from such a sketch remains an open problem. A key element in concept sketches that artists aim to explore and convey is symmetry. Symmetry is ubiquitous in nature, and even more so in human-made environments; whether it is motivated by aesthetics, ergonomics, or manufacturability, designers very often create shapes that exhibit multiple degrees of symmetry[1]. In this paper, we leverage the abundance of *axis-aligned reflective symmetries* in concept sketches to automatically lift these sketches to 3D (Figure 1).

While symmetry has been considered for a number of 3D modeling tasks [Mitra et al. 2013], detecting and exploiting symmetry in real-world concept sketches raises specific challenges as well as unique opportunities. On the one hand, concept sketches are approximate, incomplete representations of 3D shapes, composed of unstructured, swiftly-drawn pen strokes that only form partial symmetry correspondences. Notably, many strokes serve as transient *construction lines* that designers draw on and around the object to achieve accurate perspective [Gryaditskaya et al. 2019]. These construction lines introduce significant visual clutter which further challenges the automatic detection of symmetries, as individual strokes often have multiple *candidate correspondences* with respect to a given symmetry plane.

On the other hand, construction lines themselves often represent symmetric primitives – also called *scaffolds* [Gryaditskaya et al. 2020; Schmidt et al. 2009] – that approximate the envisioned shape and serve as anchors for designers to position smooth surface curves. Construction lines thus provide a great amount of additional symmetry and connectivity cues that we leverage in our reconstruction algorithm. They also provide a coordinate system that defines the symmetry axes of both the shape and the scaffold.

Based on the observations above, we propose an algorithm that takes as input a concept sketch – represented as a sequence of vector strokes recorded with a pen tablet – and jointly identifies symmetry correspondences between strokes and deduces the connectivity and depth of the 3D drawing[2]. This formulation combines both binary unknowns (which strokes form a symmetry pair, and which strokes intersect) and continuous unknowns (where to position the strokes and their respective symmetry planes along the depth axis to best satisfy the symmetry and intersection relationships). We address this combinatorial challenge by casting 3D reconstruction as a binary assignment problem. Given a large set of candidate symmetry correspondences, the core of our method selects the subset of *compatible* correspondences that yields the most symmetric and well-connected 3D drawing, as measured by a score function that we derived from the analysis of designer practices.

This approach, however, does not generalize as-is to sketches of complex shapes which may contain multiple, possibly local, reflective symmetries. Moreover, directly solving for the best 3D reconstruction for all input strokes at once is computationally infeasible. We address this challenge by segmenting the inputs into locally symmetric *building blocks*. In constructing the blocks we leverage insights about designer workflow. Each constructed block exhibits dense symmetry correspondences with respect to a single triplet of axis-aligned symmetry planes. We integrate our binary assignment algorithm within a search procedure to find the best position of the three planes for each triplet. We then reconstruct the blocks in order of appearance in the drawing sequence, such that the geometry recovered for a block provides context to anchor subsequent blocks.

We demonstrate the versatility and robustness of our approach by reconstructing more than 80 real-world sketches. Our method significantly improve over the recent work by Gryaditskaya et al. [2020], as validated by a comparative study where participants found our results more plausible by a ratio of 2:1.

2 RELATED WORK

We focus our discussion on methods that lift a single drawing to 3D, especially those that leverage symmetry for this purpose. We refer readers to surveys on sketch-based modeling [Bonnici et al. 2019; Cordier et al. 2016; Olsen et al. 2009] and symmetry-based geometry processing [Mitra et al. 2013] for more general discussions.

Our method addresses part of the longstanding challenge of assigning depth to pen strokes in a drawing, as pioneered by Lipson and Shpitalni [1996]. Since each stroke point can have arbitrary depth, additional constraints are necessary to make this problem well-posed. Early methods focused on polyhedral shapes where many of the strokes are straight, form planar faces, and are parallel or perpendicular [Liu et al. 2008; Yang et al. 2013]. Closer to our application domain are methods that leverage the drawing techniques of designers, including *cross-section lines* that depict curvature directions [Shao et al. 2011; Xu et al. 2014] and construction lines that form polyhedral *scaffolds* around space curves [Schmidt et al. 2009]. While we benefit from characteristics of design drawings, such as the frequent presence of axis-aligned construction lines, we do not impose any particular technique requirements on the input.

We formulate depth estimation as an optimization problem, constrained by intersection and symmetry relationships between strokes. A complementary trend casts 3D reconstruction as a machine learning problem, using large datasets of synthetic drawings to train deep neural networks [Delanoy et al. 2018; Guillard et al. 2021; Li et al. 2018; Wang et al. 2020; Zhang et al. 2021; Zhong et al. 2020a,b]. But Gryaditskaya et al. [2019] showed that deep networks trained on synthetic drawings fail on concept sketches due to the abundance of construction lines that are not well modeled by existing non-photorealistic rendering algorithms.

Most methods for 3D reconstruction from a single drawing require precise input or manual annotations to compute geometric relationships between the pen strokes [Chen et al. 2008; Gingold et al. 2009; Olsen et al. 2011]. A notable exception is the recent work by Gryaditskaya et al. [2020] that lifts real-world concept sketches to 3D by identifying whether two strokes that intersect in the drawing should intersect in 3D, and then deducing the stroke depth from these intersections. They solve the resulting binary optimization problem using a tailored search algorithm that progresses over the drawing sequence to reconstruct small groups of strokes at a time. Our approach outperforms this state-of-the-art algorithm by jointly identifying intersections and symmetry correspondences. In addition to providing strong 3D geometry cues, symmetry allows us to

[1]On a random set of 500 models from the ABC dataset [Koch et al. 2019], we found that only 4% had a part without any axis-aligned symmetry plane.
[2]Code and data available at https://ns.inria.fr/d3/SymmetrySketch/

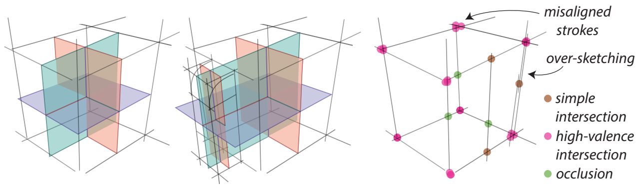


Figure 2: Drawing properties. Concept sketches often exhibit global and local reflective symmetries with respect to axis-aligned planes (a). Designers use existing intersections to anchor new strokes, yielding high-valence intersections; however, some of the intersections are due to occlusion. Furthermore, individual lines are often mis-aligned and over-sketched (b).

compute multiple candidate 3D reconstructions for many strokes at once, and then to select the best overall reconstruction from these candidates using an efficient commercial binary solver.

Reflective symmetry has long been used as a geometric constraint for sketch-based modeling [Bae et al. 2008; Cordier et al. 2013, 2011; Miao et al. 2015; Öztireli et al. 2011; Plumed et al. 2016]. However, most methods assume the presence of a single global reflective symmetry. Furthermore, existing methods typically attempt to identify a small set of confident symmetry correspondences using geometric heuristics and then fix these correspondences before moving on to 3D reconstruction [Öztireli et al. 2011], or only evaluate different possible subsets of correspondences over simple drawings [Cordier et al. 2013]. These strategies often fail on real-world concept sketches that exhibit significant clutter and inaccuracy. In contrast, we formulate our problem as an integer program, allowing us to robustly and efficiently select symmetry correspondences based on the quality of the resulting 3D reconstruction.

Reflective symmetry has also inspired computer vision algorithms based on multi-view geometry [Köser et al. 2011; Sinha et al. 2012], or more recently unsupervised learning [Wu et al. 2020]. Our formulation is inspired by the work of Jayadevan et al. [2017] and Xue et al. [2011], who reconstruct symmetric shapes from photographs by selecting symmetry correspondences between edges of the image such that the resulting shape is composed of a small number of planar faces. Concept sketches, however, contain many more lines than photographs because designers frequently draw hidden parts of the object, and draw construction lines on and around the object. We handle those characteristics of concept sketches by formulating a tailored score function that we derive from observations about how designers draw, and that is not restricted to planar shapes. Our formulation also distinguishes 3D intersections from occlusions, which is an additional challenge ubiquitous in design sketches due to the presence of hidden and construction lines.

3 DRAWING PROPERTIES

Our algorithm is motivated by the key visual cues observers employ to recover 3D shape from concept sketches and is designed to overcome several challenges posed by such sketches.

Symmetry. Most manufactured objects exhibit a global reflective symmetry, along with several local symmetries (Fig. 2a). Designers decompose such objects into axis-aligned geometric primitives that are themselves symmetric with respect to triplets of symmetry planes (e.g. cuboids and cylinders) [Eissen and Steur 2011; Henry 2012; Robertson and Bertling 2013]. Designers often construct complex shapes by drawing one primitive at a time, such that consecutive strokes depict self-contained locally-symmetric parts.

Anchoring. To draw lines in perspective, designers often leverage existing intersections as *anchors* through which they trace new lines. This strategy results in tightly-connected *scaffolds* where multiple strokes meet at high-valence intersections and most strokes are part of two or more intersections [Gryaditskaya et al. 2020] (Fig. 2b).

Coverage. Designers produce legible sketches by avoiding extending strokes beyond their intended length. As a consequence, well-anchored strokes typically do not extend beyond their two farthest apart 3D intersections [Gryaditskaya et al. 2020].

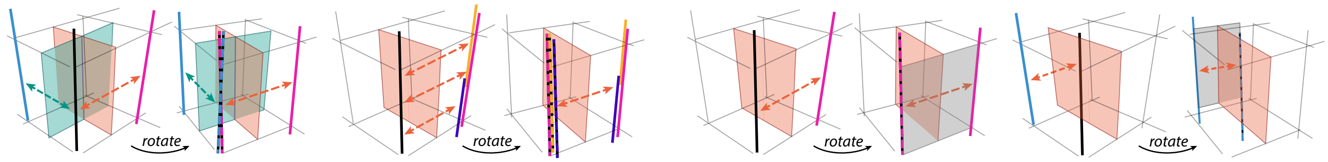
Inaccuracy and Ambiguity. Concept sketches are often over-sketched, with multiple intermittent or overlapping strokes depicting the same line. Concept sketches are also drawn under approximate perspective, with pen strokes that do not intersect precisely [Gryaditskaya et al. 2020]. Finally, because designers commonly draw hidden lines, many intersections are due to occlusion (Fig. 2b).

4 METHOD OVERVIEW

Our method takes as input a concept sketch captured with a pen tablet, where each pen stroke is represented as a pair consisting of a polyline and a time stamp. We assume that the sketch represents a shape drawn in perspective, and we apply a vanishing point detector to calibrate the perspective camera [Gryaditskaya et al. 2020]. Given the camera matrix and a symmetry plane, two strokes that are symmetric with respect to that plane can be lifted to 3D via standard multi-view geometry, as detailed in supplemental materials. Our core reconstruction challenge can therefore be formulated as identifying all symmetry planes present in the sketch and detecting the pairs of strokes symmetric with respect to these planes. Once all such pairs are reconstructed, we use their geometry to reconstruct the remaining non-symmetric strokes they intersect.

While we expect all symmetry planes in our sketches to be aligned with the major axes, we do not *a priori* know the number or location of these planes. A key observation behind our method is that artists typically draw symmetric strokes soon after each other, and mentally break the objects they draw into blocks where each block has only one plane of symmetry for each axis; they then draw these blocks approximately sequentially, first depicting the content of one block and then the next. Based on this observation we decompose the drawing into likely *symmetric building-blocks*, where each block exhibits symmetry correspondences with respect to a single triplet of axis-aligned planes. We identify these blocks by analyzing the local density of symmetry correspondences along the drawing sequence. We reconstruct each block in sequence, fixing the reconstruction of previous blocks to anchor new ones.

Even for a single block, the problem of computing the planes and the symmetric stroke pairs remains highly challenging. We do not know in advance which strokes form symmetry pairs, nor do



(a) Correspondences from different planes (b) Correspondences from over-sketching (c) Concurrent reconstructions due to depth ambiguity

Figure 3: A stroke can have multiple *compatible* symmetry correspondences, either because it is symmetric to other strokes with respect to several planes (a), or because it is symmetric to several strokes depicting the same over-sketched line (b). Due to drawing inaccuracy, the 3D reconstructions given by each correspondence do not perfectly coincide, as depicted in the rotated views. A stroke can also have multiple *incompatible* correspondences, each yielding a very different 3D reconstruction (c). In this example, the black stroke could be reconstructed as lying on the front part of the cube due to its correspondence with the pink stroke (c, left), or lying far at the back due to its correspondence with the blue stroke (c, right).

we know where to position the triplet of symmetry planes along their respective axes. We tackle this challenge by predicting and evaluating multiple plane positions and a large set of candidate symmetry correspondences. For each position, we reconstruct all candidate pairs of symmetric strokes and select the compatible subset that yields the best 3D reconstruction of the entire block. We formulate the quality of a reconstruction in the form of a score function that favors the presence of multiple symmetries, and anchoring of strokes within the block and its predecessors. Combined with suitable constraints, this formulation is amenable to efficient maximization as an integer program where symmetry correspondences and stroke intersections are optimization variables. We use this formulation to obtain the optimal plane triplet and set of stroke pairs symmetric with respect to each of the triplet's planes.

Algorithm 1 summarizes the main steps of this approach. In the sections below, we first describe the integer program that forms the core of our method and which operates on a triplet of axis-aligned symmetry planes and a set of candidate stroke correspondences computed with respect to these planes (Sec. 5). We then describe the details of our overall workflow including how we find the plane triplet of each block by searching over the most likely positions of axis-aligned planes, and how we segment the drawing into symmetric building blocks, each associated with a single triplet of planes (Sec. 6). The last stage of our method reconstructs non-symmetric strokes based on their intersections with symmetric ones (Sec. 7).

We refer the interested reader to supplemental materials for details on how we detect and reconstruct individual candidate symmetry correspondences between straight, curved, and elliptic strokes. In a nutshell, we consider all stroke pairs that are aligned with one of the three vanishing points, and keep the ones for which we can match the two strokes closely by reflecting, translating and scaling one of the strokes in the direction of the vanishing point.

Algorithm 1: Symmetric sketch reconstruction

$\mathcal{C} \longleftarrow$ Identify candidate symmetry correspondences
$\mathcal{B} \longleftarrow$ Identify local symmetric building-blocks `// Sec.6.1`
sketchReconstruction $\longleftarrow \emptyset$
for *Building-block* $\in \mathcal{B}$ **do**
  $\mathcal{P} \longleftarrow$ Identify most likely triplets of planes `// Sec.6.4`
  bestScore $\longleftarrow$ 0; bestTriplet $\longleftarrow$ 0; blockReconstruction $\longleftarrow \emptyset$
  for *Triplet* $\in \mathcal{P}$ **do**
    `// Reconstruct and group compatible correspondences`
    $\mathcal{S} \longleftarrow$ LiftTo3D($\mathcal{C}$, Triplet) `// Sec.5.1`
    `// Select the best subset of correspondences`
    (Score, Reconstruction) $\longleftarrow$
      IntegerProgram($\mathcal{S}$, sketchReconstruction)`// Sec.5`
    if *Score* $>$ *bestScore* **then**
      bestScore $\longleftarrow$ Score; bestTriplet $\longleftarrow$ Triplet
      blockReconstruction $\longleftarrow$ Reconstruction
    end
  end
  Append(sketchReconstruction, blockReconstruction)
end

5 BLOCK RECONSTRUCTION

The main component of our method is a block reconstruction procedure that, given a triplet of symmetry planes and a set of candidate symmetry correspondences, identifies the true correspondences to deduce the 3D position of their respective strokes.

Figure 3 illustrates the main challenges we face when attempting to solve this task. A single stroke can have multiple symmetry correspondences with respect to the three planes of the triplet, and each correspondence gives a 3D reconstruction of that stroke. But due to drawing inaccuracy, these reconstructions often do not coincide perfectly. Furthermore, some candidate correspondences might be erroneous and yield reconstructions that are not *compatible* with the reconstructions given by other correspondences. The main goal of our algorithm is to identify groups of correspondences that produce compatible stroke reconstructions, and to select among these groups the ones that yield the best reconstruction of the block.

We measure the quality of a reconstruction as a function of two terms: one measuring the degree of symmetry provided by the reconstruction and the other measuring how well connected the reconstruction is. The first term depends solely on which correspondences we select, while the second term also depends on whether the 3D strokes reconstructed from these correspondences intersect. These terms thus involve two sets of binary variables, one that indicates for each candidate correspondence if it is selected, and the other that indicates for each stroke intersection if it occurs in 3D. We formulate this optimization as an integer program, which we make efficient by expressing all constraints in a linear form, and only one term of the score function in a quadratic form.

5.1 Grouping Compatible Reconstructions

We denote 2D entities with lower-case letters (x), 3D entities with upper-case letters (X), and binary variables associated to these entities with bold typeface ($\mathbf{x}$, $\mathbf{X}$). Given a candidate pair of symmetric 2D strokes s_p and s_q and a corresponding symmetry plane, we define their symmetric 3D reconstructions as S_{pq} for s_p, and S_{qp} for s_q. We denote as $\mathbf{c}_{pq}$ the binary variable that indicates whether a pair pq is selected as a symmetry correspondence in the solution.

Since a stroke s_p can have multiple candidate correspondences, we first identify which correspondences form compatible groups of reconstructions. We denote these groups as S_p^k, $k \in [1, K_p]$. We consider two reconstructions S_{pq} and S_{pr} of a stroke to be compatible if they nearly coincide, *i.e.* if the maximum distance between the two reconstructions is below 10% of the length of the longest candidate reconstruction of s_p. For each group, we compute a representative geometry by averaging its constituent reconstructions, which we will need later on to detect intersections between groups and to evaluate the quality of the correspondences selected for each group.

Equipped with groups of compatible reconstructions for each stroke, we then must ensure that the group we select for a given stroke is compatible with the groups we select for all other strokes. We enforce compatibility of the overall reconstruction by associating each group with a binary variable $\mathbf{S}_p^k$, which is equal to 1 if the k^{th} group is selected. We then add the constraint that at most one group should be selected per stroke: $\sum_k \mathbf{S}_p^k \leq 1$. We also ensure that a group is only selected if at least one of its correspondences has been selected, which we express as the constraint $\mathbf{S}_p^k \leq \sum_{\mathbf{c} \in C_p^k} \mathbf{c}$, where C_p^k denotes the set of all correspondences forming group S_p^k. Similarly, a group has to be selected if any of its correspondences has been selected, $\mathbf{S}_p^k \geq \mathbf{c}\ \forall \mathbf{c} \in C_p^k$.

5.2 Optimizing for Symmetry

The first term of our score function favors 3D reconstructions that are symmetric with respect to multiple planes in the given triplet. To express this property, we associate each stroke s_p with a triplet of auxiliary binary variables $\mathbf{x}_p$, $\mathbf{y}_p$, $\mathbf{z}_p$, where each variable indicates whether there is at least one correspondence $\mathbf{c}_{pq}$ that has been selected for the corresponding axis-aligned plane. We then seek to maximize the sum over all strokes

$$F_{\text{symmetry}} = \sum_p \mathbf{x}_p + \mathbf{y}_p + \mathbf{z}_p. \tag{1}$$

Due to overdrawing, curves in concept drawings are often drawn as a series of contiguous strokes, with each stroke forming partial symmetry correspondences with some of the strokes depicting the symmetric curve. Simply counting such partial correspondences without accounting for the degree of overlap between the strokes and the reflections of their counterparts can skew the metric toward pairs with minuscule partial overlaps. To ensure that the 3D reconstruction of a curved stroke is well supported by its partial correspondences with other strokes, we measure the overlap between each partial reconstruction S_{pq} and S'_{qp}, where S'_{qp} denotes the reflection of the symmetric reconstruction S_{qp} with respect to its symmetry plane. We measure the support of curved strokes as

$$F_{\text{support}} = \sum_{p,q} \text{overlap}(S_{pq}, S'_{qp})\mathbf{c}_{pq}. \tag{2}$$

This term is not necessary for straight strokes because their geometry is fully determined by a single correspondence, even if partial.

Finally, since only a subset of the correspondences that form a group might be selected, we measure whether this subset is representative of the entire group. We do so by penalizing the maximum distance between each selected 3D reconstructions S_{pq} and the average geometry of the group:

$$F_{\text{proximity}} = \sum_{p,k} \sum_{c_{pq} \in C_p^k} \text{dist}(S_p^k, S_{pq})\mathbf{c}_{pq}. \tag{3}$$

5.3 Optimizing for Connectivity

The second part of our score function prioritizes reconstructions that follow the principles of *anchoring* and *coverage*. To evaluate these properties, we first need to identify intersections between groups of compatible reconstructions that form the solution. We denote a 2D intersection between two strokes s_p and s_r as i_{pr}, and $\mathbf{i}_{pr}$ is the binary variable that indicates whether the two strokes intersect in 3D. We express $\mathbf{i}_{pr}$ in a linear form thanks to symmetric *range constraints*, which only allow an intersection to occur if the depth difference between the groups selected for the two strokes is small (see supplemental materials for detailed equations).

Stroke Anchoring. We consider a stroke to be *well-anchored* if it is part of at least two intersections of high valence. To identify such intersections, we first group nearby intersections as done by Gryaditskaya et al. [2020]. We then consider an intersection to be high valence if it is grouped with intersections between strokes converging to three different vanishing points, or to two vanishing points and to a third arbitrary direction. We count the number of high valence intersections along each stroke s_p, and activate an auxiliary variable $\mathbf{w}_p$ if the stroke is weakly anchored by being part of one intersection, and another variable $\mathbf{f}_p$ if it is fully anchored by being a part of two intersections. We penalize the strokes inversely to their degree of anchoring:

$$F_{\text{anchoring}} = -\sum_p \left(2 - \mathbf{w}_p - \mathbf{f}_p\right) \mathbf{s}_p, \tag{4}$$

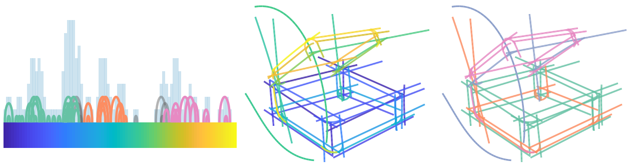
where we activate the binary variable $\mathbf{s}_p$ if one of the groups of compatible reconstructions of s_p has been selected, $\mathbf{s}_p = \sum_k \mathbf{S}_p^k$.

Coverage. We favor 3D interpretations that cover the input strokes well by maximizing the distance between the first and last intersections selected along a stroke:

$$F_{\text{coverage}} = \sum_p \left(\max_{i \in \mathcal{I}_p} t_p(i) - \min_{i \in \mathcal{I}_p} t_p(i) \right) \mathbf{s}_p, \tag{5}$$

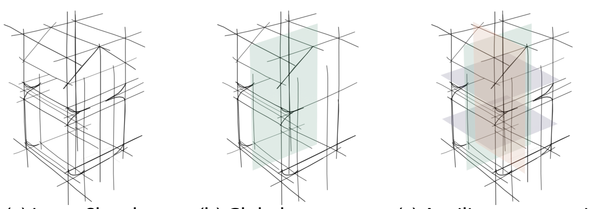
where $\mathcal{I}_p$ denotes the set of intersections selected along stroke s_p, and $t_p(i)$ denotes the arc-length parameter value of intersection i.

To identify the maximum and minimum arc-length parameters of selected 3D intersections, we associate each intersection i_{pr} with two binary variables, $\mathbf{a}_{pr}^F$ and $\mathbf{a}_{pr}^L$, which respectively specify if the intersection should be considered as the first or last intersection



(a) Sketch sequence (b) Input sketch (c) Segmented blocks

Figure 4: We segment the stroke sequence into symmetric building blocks based on the density of local symmetry correspondences (a). We color-code the input strokes by order of appearance, from blue to yellow (a,b). In this example, the algorithm identifies one block for the bottom cuboid (green), one for its rounded edges (orange), and one for the slanted cuboid (pink).



(a) Input Sketch (b) Global symmetry (c) Auxiliary symmetries

Figure 5: We first reconstruct all strokes that are symmetric with respect to a global plane (b). We then fix this partial reconstruction to serve as a scaffold for reconstructing multiple local symmetries (c).

along the stroke. Accompanying constraints ensure that we select at most one minimal and one maximal intersection per stroke, that each intersection cannot be both minimal and maximal at the same time, and that only selected 3D intersections are considered (see supplemental materials for detailed equations). Equipped with these variables, we rewrite the coverage term as:

$$F_{\text{coverage}} = \sum_{p} \sum_{i_{pr} \in \mathcal{I}_p} t_p(i_{pr}) \mathbf{a}_{pr}^{L} - \sum_{i_{pr} \in \mathcal{I}_p} t_p(i_{pr}) \mathbf{a}_{pr}^{F}. \qquad (6)$$

We detail in our supplemental material how we treat the special case of ellipses, for which coverage cannot be measured since they do not have a start and end; instead, we favor 3D reconstructions of ellipses that are most circular.

5.4 Score Function

We combine the terms described above to form the score function to be maximized subject to the listed constraints:

$$F_{\text{total}} = F_{\text{symmetry}} + F_{\text{support}} + F_{\text{proximity}} + F_{\text{anchoring}} + F_{\text{coverage}}, \quad (7)$$

where the binary optimization variables $\mathbf{i}_{pq}$ and $\mathbf{c}_{pq}$ select intersections and symmetry correspondences respectively. We solve this optimization problem with the commercial solver *Gurobi* [Gurobi Optimization, LLC 2021]. We use a fixed set of weights $\lambda_{\text{symmetry}} = 2$, $\lambda_{\text{support}} = 10$, $\lambda_{\text{proximity}} = -100$, $\lambda_{\text{anchoring}} = 5$, and $\lambda_{\text{coverage}} = 4$.

6 ALGORITHM DETAILS

6.1 Computing Building Blocks

We handle complex sketches by grouping strokes that represent largely independent symmetric parts, or building blocks, each with its own triplet of symmetry planes. This decomposition breaks the reconstruction problem into a series of sub-problems, which contributes to the scalability of our method and enables us to process inputs with multiple symmetry planes per axis.

We identify self-contained symmetric building blocks by leveraging the observations that their strokes are typically drawn consecutively, and that they share many internal symmetry correspondences. Treating the input sketch as an ordered sequence of pen strokes, we say that a correspondence between stroke s_p and s_q *spans* over s_i if $p < i < q$. We then count, for each stroke, the number of correspondence candidates that span over it. We restrict this computation to correspondences between nearby strokes, *i.e.* those that span at most 5 strokes. As illustrated in Figure 4, the resulting histogram exhibits distinct modes, one for each highly symmetric part. We segment this histogram at each of its local minima to form *symmetric building blocks*. We merge neighboring building blocks if they are too small (less than 10 strokes), and we split building blocks in half if they are too large (more than 30 strokes).

6.2 Sequential Block Reconstruction

We reconstruct the building blocks in their order of appearance in the drawing sequence. We use the reconstruction of each building block as a scaffold for the next one, such that the selected compatible reconstructions act as a geometric context for subsequent strokes with which they share symmetric correspondences and intersections. We account for this context by computing the score function using the strokes of all building blocks processed so far, even though we only optimize for the strokes of the current building block. Note that while we search for a new triplet of axis-aligned symmetry planes for each new building block, we always consider the planes used by preceding building blocks so that several building blocks can share the same planes if appropriate.

6.3 Enforcing Globally Symmetric Reconstruction

Typical design sketches contain a mixture of local and global symmetries. We assume and enforce the presence of at least one global symmetry plane. We identify this global symmetry by performing two passes of our algorithm (Fig. 5). Our first pass enforces global symmetry: for each major axis, we place an axis-orthogonal plane Π_{global} at the origin and then run our algorithm with the additional constraint that all reconstructed strokes need to have a symmetry correspondence with respect to this plane. We keep the best-scoring solution across these three axes. This first pass yields a globally symmetric, albeit incomplete, reconstruction of the sketch. In a second pass, we optimize all remaining strokes while keeping the globally-symmetric reconstruction fixed, leaving each stroke free to form correspondences with respect to any of the planes.

6.4 Computing Plane Triplets per Block

Computing the reconstruction for each block requires a triplet of symmetry planes. When only one symmetry plane exists, we can fix the scale of the scene by positioning the plane at the origin. When a triplet of symmetry planes is present, however, we need to solve for the position of each plane along its axis.

We find the optimal position for a triplet by fixing the first plane at the origin, and then searching among candidate positions of the two other planes the one that yields the best reconstruction overall, as measured by Equation 7. To obtain these candidate positions, we first place the three planes at the origin and reconstruct all symmetric correspondences of each plane separately. We then leverage the fact that translating a plane along its axis is equivalent to translating and scaling the 3D reconstruction of the strokes that are symmetric with respect to that plane. We loop over each 2D intersection i_{pr} and deduce how to position the planes such that the reconstructed strokes S_p^k and S_r^l effectively intersect in 3D.

Computing a complete reconstruction for each candidate plane triplet would be prohibitive. Since the set of 3D strokes reconstructed for each plane is sparse, however, only a few triplets yield intersections between many strokes. Based on this observation, we first compute, for each candidate triplet, the coverage term of all candidate reconstructions of all strokes; this gives us an upper bound of our score function. We then only run the integer program for the positions that maximize this upper bound (we keep the best 64 positions in our experiments). We further speed up this iterative search by providing each run of the integer program with the best score obtained so far, allowing the branch-and-bound solver to trim any branch of the solution tree that has a lower score.

7 COMPLETING THE RECONSTRUCTION

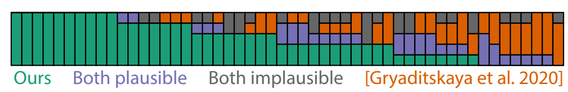
Artist drawings contain strokes with no symmetric counterparts, as well as strokes which have such counterparts but which were either not matched with these counterparts or assigned different sub-par symmetric counterparts by our block-level processing. We complete the reconstruction by addressing both types of strokes.

We improve symmetric correspondences by reevaluating all reconstructed strokes with low reconstruction confidence. We define a stroke as low confidence if it is part of only one symmetry correspondence and its coverage term is below 0.5. For any such stroke s, we go through all symmetry planes and all other strokes s'; we reflect each such stroke around the plane and measure the distance between the 2D projection of the reflected stroke and s. We reconstruct the stroke s by matching it with s' if the computed projection is within our projection tolerance and the new reconstruction improves the coverage of s. We reconstruct the remaining non-symmetric strokes using anchoring and coverage cues, leveraging their 2D intersections with previously reconstructed strokes. We perform this reconstruction using a greedy version of the algorithm by Gryaditskaya et al. [2020], which is sufficient in our context since few of the non-reconstructed strokes interact.

8 EVALUATION AND RESULTS

Fig. 6 compares our results to the ones obtained by Gryaditskaya et al. [2020]. Overall, our reconstructions are more faithful to the input sketch, with fewer dangling strokes and non-symmetric distortions. We provide additional results and comparisons on 82 sketches as supplemental materials, in the form of turntable videos. These sketches correspond to the subset of first-impression concept sketches from OpenSketch [Gryaditskaya et al. 2019] on which we managed to calibrate a perspective camera using the algorithm of Gryaditskaya et al. [2020]; this calibration failed on 21 sketches due to lack of axes-aligned lines. Our algorithm takes a few minutes for simple sketches (Fig. 4) to a dozen of minutes for complex ones (Fig. 1).

Comparative study. We conducted a perceptual study to quantify our improvement over prior work. We recruited 10 participants, which we distributed in two groups that assessed 26 sketches each. For each sketch, we presented the input along with our reconstruction and the reconstruction by Gryaditskaya et al. [2020]. The two reconstructions were colored to highlight their differences, similar to Fig. 6. They were displayed in the form of looping videos showing slight view changes, and were ordered randomly. The participants were asked to indicate whether the first reconstruction is more plausible, the second reconstruction is more plausible, both are equally plausible, or both are equally implausible. The plot below visualizes the 5 answers over all 52 sketches. Our results were judged more plausible 51% of the time, while the results of Gryaditskaya et al. [2020] were judged more plausible 23% of the time. Both reconstructions were judged equally plausible 14% of the time, and equally implausible 12% of the time. We conducted a paired samples t-test on these results, which showed that this improvement over prior work is highly statistically significant (p-value of 1.0e-07).



Limitations. Our method is sensitive to the detection of the initial set of candidate symmetry correspondences. If the detection is too permissive, the integer program can get overwhelmed by too many erroneous correspondences. If the selection is too strict, it might miss correct correspondences. We describe in supplemental materials the heuristics we use to select likely correspondences. Future work might improve on these heuristics, possibly using machine learning. Symmetry correspondences are especially difficult to detect on long, over-sketched curves drawn under strong foreshortening. We improved the quality of our reconstruction on some over-sketched drawings by running the method of Liu et al. [2018] to aggregate nearby strokes. But we observed that such filtering can degrade results on other sketches as it tends to merge strokes that correspond to different parts of the shape. A potential avenue for future work would be to also treat stroke aggregation as part of the optimization, letting the binary solver decide if two strokes should be merged to improve the reconstruction.

9 CONCLUSION

We have presented the first algorithm that leverages the abundance of multiple axis-aligned reflective symmetries in concept sketches to automatically reconstruct such sketches. By casting the joint selection of symmetry correspondences and stroke intersections

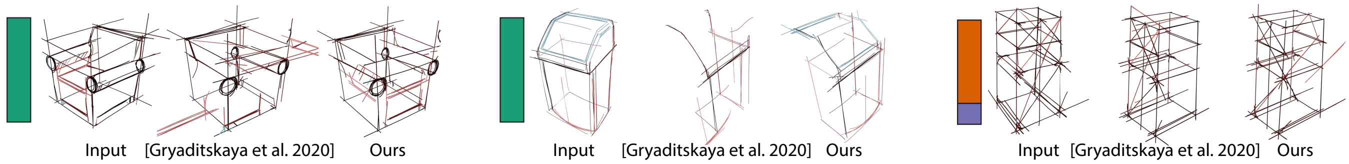


Figure 6: Comparison to the method by Gryaditskaya et al. [2020]. We highlight the strongest differences in red, and provide for each sketch the answers from our comparative study (Ours, Gryaditskaya et al. [2020], Both plausible). Our method produces fewer dangling strokes (sofa) and can reconstruct disconnected strokes (trash bin, blue strokes). Our results were judged more plausible on the majority of cases, although a few were judged inferior, such as the house (right) where our method wrongly interprets two diagonal lines as symmetric, which stands out in the otherwise correct reconstruction.

as an integer program, our method successfully reconstructs real-world sketches despite significant clutter and imprecision in such input. Our formulation solves for the depth of the pen strokes but keeps their position in the drawing plane fixed. Yet, the structural information we recover holds great potential to *regularize* the 3D reconstruction, for instance by snapping nearby symmetry planes and intersections while enforcing the selected symmetry relationships.

ACKNOWLEDGMENTS

This work was supported by the ERC Starting Grant D^3 (ERC-2016-STG 714221), the Natural Sciences and Engineering Research Council of Canada (NSERC) grant RGPIN-2018-03944, and research and software donations from Adobe.

REFERENCES

Seok-Hyung Bae, Ravin Balakrishnan, and Karan Singh. 2008. ILoveSketch: as-natural-as-possible sketching system for creating 3d curve models. In *Proc. UIST*. ACM.

Alexandra Bonnici, Alican Akman, Gabriel Calleja, Kenneth P Camilleri, Patrick Fehling, Alfredo Ferreira, Florian Hermuth, Johann Habakuk Israel, Tom Landwehr, Juncheng Liu, et al. 2019. Sketch-based interaction and modeling: where do we stand? *Artificial intelligence for engineering design analysis and manufacturing* (2019), 1–19.

Xuejin Chen, Sing Bing Kang, Ying-Qing Xu, Julie Dorsey, and Heung-Yeung Shum. 2008. Sketching Reality: Realistic Interpretation of Architectural Designs. *ACM Transactions on Graphics (TOG)* 27, 2, Article 11 (may 2008), 15 pages.

Frederic Cordier, Hyewon Seo, Mahmoud Melkemi, and Nickolas S. Sapidis. 2013. Inferring Mirror Symmetric 3D Shapes from Sketches. *Computer Aided Design* 45, 2 (2013).

Frederic Cordier, Hyewon Seo, Jinho Park, and Jun Yong Noh. 2011. Sketching of mirror-symmetric shapes. *IEEE Transactions on Visualization and Computer Graphics* 17, 11 (2011), 1650–1662.

Frederic Cordier, Karan Singh, Yotam Gingold, and Marie-Paule Cani. 2016. Sketch-based modeling. In *SIGGRAPH ASIA 2016 Courses*. 1–222.

Johanna Delanoy, Mathieu Aubry, Phillip Isola, Alexei Efros, and Adrien Bousseau. 2018. 3D Sketching using Multi-View Deep Volumetric Prediction. *Proceedings of the ACM on Computer Graphics and Interactive Techniques* 1, 21 (2018).

Koos Eissen and Roselien Steur. 2011. *Sketching: The Basics.* Bis Publishers.

Yotam Gingold, Takeo Igarashi, and Denis Zorin. 2009. Structured annotations for 2D-to-3D modeling. In *ACM Transactions on Graphics (Proc. SIGGRAPH Asia)*, Vol. 28.

Yulia Gryaditskaya, Felix Hähnlein, Chenxi Liu, Alla Sheffer, and Adrien Bousseau. 2020. Lifting Freehand Concept Sketches into 3D. *ACM Transactions on Graphics (Proc. SIGGRAPH Asia)* (2020).

Yulia Gryaditskaya, Mark Sypesteyn, Jan Willem Hoftijzer, Sylvia Pont, Fredo Durand, and Adrien Bousseau. 2019. OpenSketch: A Richly-Annotated Dataset of Product Design Sketches. *ACM Transactions on Graphics (Proc. SIGGRAPH Asia)* (2019).

Benoit Guillard, Edoardo Remelli, Pierre Yvernay, and Pascal Fua. 2021. Sketch2Mesh: Reconstructing and Editing 3D Shapes From Sketches. In *Proc. IEEE International Conference on Computer Vision (ICCV)*.

Gurobi Optimization, LLC. 2021. Gurobi Optimizer Reference Manual. https://www.gurobi.com

Kevin Henry. 2012. *Drawing for product designers.* Laurence King Publishing.

Vijai Jayadevan, Aaron Michaux, Edward J. Delp, and Zygmunt Pizlo. 2017. 3D Shape Recovery From Real Images Using a Symmetry Prior. In *Computational Imaging*.

Sebastian Koch, Albert Matveev, Zhongshi Jiang, Francis Williams, Alexey Artemov, Evgeny Burnaev, Marc Alexa, Denis Zorin, and Daniele Panozzo. 2019. Abc: A big cad model dataset for geometric deep learning. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 9601–9611.

Kevin Köser, Christopher Zach, and Marc Pollefeys. 2011. Dense 3D Reconstruction of Symmetric Scenes from a Single Image. In *DAGM-Symposium (Lecture Notes in Computer Science, Vol. 6835)*. Springer, 266–275.

Changjian Li, Hao Pan, Yang Liu, Xin Tong, Alla Sheffer, and Wenping Wang. 2018. Robust flow-guided neural prediction for sketch-based freeform surface modeling. In *ACM Transactions on Graphics (Proc. SIGGRAPH Asia)*. ACM, 238.

H Lipson and M Shpitalni. 1996. Optimization-based reconstruction of a 3D object from a single freehand line drawing. *Computer-Aided Design* 28, 8 (1996).

Chenxi Liu, Enrique Rosales, and Alla Sheffer. 2018. StrokeAggregator: consolidating raw sketches into artist-intended curve drawings. *ACM Transactions on Graphics (Proc. SIGGRAPH Asia)* 37, 4 (2018), 97.

Jianzhuang Liu, Liangliang Cao, Zhenguo Li, and Xiaoou Tang. 2008. Plane-based optimization for 3D object reconstruction from single line drawings. *IEEE Transaction on Pattern Analysis Machine Intelligence* 30, 2 (2008), 315–327.

Yongwei Miao, Feixia Hu, Xudong Zhang, Jiazhou Chen, and Renato Pajarola. 2015. SymmSketch: Creating symmetric 3D free-form shapes from 2D sketches. *Computational Visual Media* 1 (2015), 3–16.

Niloy J. Mitra, Mark Pauly, Michael Wand, and Duygu Ceylan. 2013. Symmetry in 3D Geometry: Extraction and Applications. *Computer Graphics Forum* 32, 6 (2013).

Luke Olsen, Faramarz Samavati, and Joaquim Jorge. 2011. NaturaSketch: Modeling from Images and Natural Sketches. *IEEE Computer Graphics and Applications* (2011).

Luke Olsen, Faramarz F Samavati, Mario Costa Sousa, and Joaquim A Jorge. 2009. Sketch-based modeling: A survey. *Computers & Graphics* 33, 1 (2009), 85–103.

A Cengiz Öztireli, Umut Uyumaz, Tiberiu Popa, Alla Sheffer, and Markus Gross. 2011. 3D modeling with a symmetric sketch. In *Proceedings of the Eighth Eurographics Symposium on Sketch-Based Interfaces and Modeling*. 23–30.

Raquel Plumed, Pedro Company, and Peter A.C. Varley. 2016. Detecting mirror symmetry in single-view wireframe sketches of polyhedral shapes. *Computers & Graphics* 59 (2016), 1–12.

Scott Robertson and Thomas Bertling. 2013. *How to Draw: drawing and sketching objects and environments from your imagination.* Design Studio Press.

Ryan Schmidt, Azam Khan, Karan Singh, and Gord Kurtenbach. 2009. Analytic drawing of 3D scaffolds. In *ACM transactions on graphics (Proc. SIGGRAPH Asia)*, Vol. 28.

Cloud Shao, Adrien Bousseau, Alla Sheffer, and Karan Singh. 2011. CrossShade: shading concept sketches using cross-section curves. *ACM Transactions on Graphics (Proc. SIGGRAPH)* 31, 4 (2011).

Sudipta Sinha, Krishnan Ramnath, and Rick Szeliski. 2012. Detecting and Reconstructing 3D Mirror Symmetric Objects. In *European Conference on Computer Vision*.

Jiayun Wang, Jierui Lin, Qian Yu, Runtao Liu, Yubei Chen, and Stella X. Yu. 2020. 3D Shape Reconstruction from Free-Hand Sketches.

Shangzhe Wu, Christian Rupprecht, and Andrea Vedaldi. 2020. Unsupervised Learning of Probably Symmetric Deformable 3D Objects from Images in the Wild. In *CVPR*.

Baoxuan Xu, William Chang, Alla Sheffer, Adrien Bousseau, James McCrae, and Karan Singh. 2014. True2Form: 3D curve networks from 2D sketches via selective regularization. *ACM Transactions on Graphics (Proc. SIGGRAPH)* 33, 4 (2014).

Tianfan Xue, Jianzhuang Liu, and Xiaoou Tang. 2011. Symmetric piecewise planar object reconstruction from a single image. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. IEEE.

Linjie Yang, Jianzhuang Liu, and Xiaoou Tang. 2013. Complex 3d general object reconstruction from line drawings. In *IEEE Int. Conference on Computer Vision*.

Song-Hai Zhang, Yuan-Chen Guo, and Qing-Wen Gu. 2021. Sketch2Model: View-Aware 3D Modeling From Single Free-Hand Sketches. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Yue Zhong, Yulia Gryaditskaya, Honggang Zhang, and Yi-Zhe Song. 2020a. Deep Sketch-Based Modeling: Tips and Tricks. In *Int. Conference on 3D Vision (3DV)*.

Yue Zhong, Yonggang Qi, Yulia Gryaditskaya, Honggang Zhang, and Yi-Zhe Song. 2020b. Towards practical sketch-based 3d shape generation: The role of professional sketches. *IEEE Transactions on Circuits and Systems for Video Technology* 31, 9 (2020).