Symmetry-driven 3D Reconstruction from Concept Sketches Supplemental materials

FELIX HÄHNLEIN, Inria, Université Côte d'Azur, France
YULIA GRYADITSKAYA, Surrey Institute for People Centred AI and Centre for Vision, Speech and Signal Processing (CVSSP), University of Surrey, United Kingdom
ALLA SHEFFER, University of British Columbia, Canada
ADRIEN BOUSSEAU, Inria, Université Côte d'Azur, France

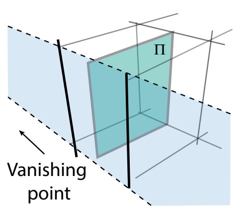
ACM Reference Format:
Felix Hähnlein, Yulia Gryaditskaya, Alla Sheffer, and Adrien Bousseau. 2022. Symmetry-driven 3D Reconstruction from Concept Sketches Supplemental materials. In *Special Interest Group on Computer Graphics and Interactive Techniques Conference Proceedings (SIGGRAPH '22 Conference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

We first describe how we detect and reconstruct individual symmetry correspondences (Sec. 1). We then provide detailed equations of our integer program (Sec. 2), and a discussion of computation time for the main stages of the method (Sec. 3). Finally, we provide a gallery of reconstructions obtained with our method at the end of this document. We encourage readers to view our results and comparisons with animated viewpoint changes on our supplemental webpage.

1 GENERATING SYMMETRY CANDIDATES

We describe how we identify candidate pairs of symmetric strokes in the input sketch, and how we lift each pair to 3D given a symmetry plane and a calibrated camera.

1.1 Detecting 2D Symmetry Correspondences



The inset illustrates the geometry of reflective symmetry in a perspective drawing. Given a symmetry plane Π that is axis-aligned, two strokes s_p and s_q that are symmetric with respect to Π lie in the same triangle formed by joining the endpoints of the strokes to the vanishing point corresponding to the axis perpendicular to Π (blue polygon in inset). We leverage this property to only consider that one stroke can be symmetric to another if the triangles they form with the vanishing point significantly overlap. In practice, we adopt a tolerance of 70% of overlap to account for sketching inaccuracy. We complement this criteria with additional heuristics to further reduce the set of candidates for straight strokes, curved strokes, and ellipses, as described next. Furthermore, we create a correspondence between each stroke and itself, which is necessary to reconstruct strokes that lie in the symmetry plane, or that is self-symmetric with respect to the plane.

Straight Strokes. Design drawings are often constructed using axis-aligned straight strokes [Gryaditskaya et al. 2020]. We leverage this observation to restrict the search of symmetry correspondences to pairs of straight strokes that converge to the same vanishing point, or to none.

Curved Strokes. Due to foreshortening, a pair of symmetric 3D curves can produce very different 2D curves under perspective projection. Considering all possible pairs, however, would overwhelm the integer program with many erroneous correspondences. We distill a set of plausible correspondences by only considering pairs of curved strokes that match well after transforming one of the strokes with a 2D translation, scaling, and reflection along the vanishing direction corresponding to the axis perpendicular to the symmetry plane. We keep such pairs if their Chamfer distance is below 5% of the length of the longer stroke.

Ellipses. We assume that ellipses represent 3D circles under perspective projection. When this is the case, designers align the minor axis of the ellipse with the vanishing point corresponding to the axis perpendicular to the 3D circle [Eissen and Steur 2011]. Following this observation, we only form candidate correspondences for ellipse pairs whose minor axes are aligned (up to 15°), as these typically depict cross-sections of the same cylinder.

1.2 Lifting Symmetry Pairs to 3D

Straight Strokes. Given the perspective camera, a symmetry plane Π, and two strokes s_p and s_q positioned in the image plane, we first build two planes Π_p and Π_q, each going through its respective strokes and through the camera. We then construct the plane Π'_q that is symmetric of Π_q with respect to Π. Intersecting Π_p and Π'_q gives us an infinite 3D line L_{pq}, on which we project s_p to obtain its 3D reconstruction S_{pq}. Finally, we reflect L_{pq} with respect to the symmetry plane to obtain L_{qp}, on which we project s_q to obtain S_{qp}.

Curved Strokes. Reconstructing curved strokes is more involved because such strokes often only correspond partially to other strokes. We therefore first reconstruct each part that is in correspondence with another stroke, and then stitch together multiple parts. The multiple combinations of parts yield several candidate reconstructions of the entire stroke.

We represent the 3D geometry S_{pq} of a curved stroke s_p as a cubic Bézier curve and further equip S_{pq} with an interval $[a_{pq}, b_{pq}]$

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

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

that bounds the part of the stroke that is in correspondence with the other stroke s_q, where a_{pq} and b_{pq} are arc-length parameters. Given the perspective camera and a symmetry plane Π, we identify and reconstruct corresponding parts of a stroke pair (S_{pq}, S_{qp}) by solving an optimization problem, where the variables are the depth of the Bézier control points along their camera rays and the boundaries of the intervals $[a_{pq}, b_{pq}]$ and $[a_{qp}, b_{qp}]$, such that the two curves are most symmetric in 3D and share the largest interval. Formally, we minimize an energy of the form:

$$E_{\text{curve}} = \lambda E_{\text{symmetry}} + (1-\lambda)E_{\text{interval}}, \tag{1}$$

where the two terms are weighted by λ, which we set to 0.85 in our experiments. The first term, E_{symmetry}, measures the L_2 distance between one curve and the reflection of the other one with respect to the symmetry plane, sampled uniformly along their respective intervals. Denoting by S'_{pq} the reflection of S_{pq} with respect to Π:

$$E_{\text{symmetry}} = 1 - \exp\left(-\|S'_{pq}[a_{pq}, b_{pq}] - S_{qp}[a_{qp}, b_{qp}]\|\right), \tag{2}$$

where the exponentiation normalizes the distance to $[0, 1]$. The second term penalizes short intervals:

$$E_{\text{interval}} = 1 - \exp\left(-\min\left(1 - \left(b_{pq} - a_{pq}\right), 1 - \left(b_{qp} - a_{qp}\right)\right)\right). \tag{3}$$

We solve this optimization using Sequential Least Squares Programming as implemented in SciPy.

Given multiple partial reconstructions S_{pq} and their intervals $[a_{pq}, b_{pq}]$, we next construct consolidated interpretations S^k_p by considering all combinations of reconstructions that either have non-overlapping intervals, or have overlapping intervals and similar 3D geometry (a Hausdorff distance below 10% of the partial curve's lengths). For each such combination, we uniformly sample the partial reconstructions and fit a new Bézier curve by optimizing the depth of its control points such that the curve runs close to the resulting point cloud. We regularize this fitting with a *minimal foreshortening* term [Xu et al. 2014] such that the curve extrapolates smoothly over parts of the stroke that have no correspondence. Finally, we merge consolidated interpretations that are geometrically close.

Ellipses. We assume that ellipses represent 3D circles. To lift a pair of ellipses s_p and s_q to 3D, we project s_p on a plane Π_{pq} and s_q on the reflected plane Π'_{pq}, where we optimize Π_{pq} such that the resulting 3D reconstructions S_{pq} and S_{qp} are most symmetric and most circular. We optimize the position and orientation of the plane to minimize the sum of two terms

$$E_{\text{ellipse}} = E_{\text{symmetry}} + E_{\text{eccentricity}}, \tag{4}$$

where E_{symmetry} measures the distance between one lifted ellipse the reflection of the other one

$$E_{\text{symmetry}} = 1 - \exp\left(-\|S'_{pq} - S_{qp}\|\right). \tag{5}$$

To maximize circularity, we minimize the so-called *eccentricity* of the 3D ellipses:

$$E_{\text{eccentricity}} = 1 - \exp - \max(e(S_{pq}), e(S_{qp})) \tag{6}$$

where $e(S_{pq}) = \frac{\sqrt{(a^2+b^2)}}{a}$ with a and b being the major and minor axes of the ellipse.

2 INTEGER PROGRAM FORMULATION

We now detail the terms and constraints of our integer program. **We will release our reference implementation upon acceptance to ease reproduction**

2.1 Optimizing for symmetry

2.1.1 Maximize symmetry. To recover the most symmetric sketch, we maximize stroke reconstructions which are symmetric w.r.t. multiple planes. We account for a symmetry correspondence along a certain axis with the binary variables $\boldsymbol{x_p}$, $\boldsymbol{y_p}$, $\boldsymbol{z_p}$, which if selected, mean that s_p is symmetric along an x, y or z symmetry plane respectively.

Symmetry correspondence selection between s_p and s_q is realized by selecting the binary variable $\boldsymbol{c_{pq}}$. $\boldsymbol{x_p}$, $\boldsymbol{y_p}$ and $\boldsymbol{z_p}$ can only be selected if at least one correspondence is selected along the respective axis.

$$\sum_q \boldsymbol{c^x_{pq}} \geq 1 - M * (1 - \boldsymbol{x_p}) \tag{7}$$

$$\sum_q \boldsymbol{c^y_{pq}} \geq 1 - M * (1 - \boldsymbol{y_p}) \tag{8}$$

$$\sum_q \boldsymbol{c^z_{pq}} \geq 1 - M * (1 - \boldsymbol{z_p}) \tag{9}$$

where c^k_{pq} is a symmetry correspondence along axis k. The first term of our score function maximizes the total amount of differently oriented symmetries.

$$F_{\text{symmetry}} = \sum_p \mathbf{x}_p + \mathbf{y}_p + \mathbf{z}_p. \tag{10}$$

2.1.2 Symmetry correspondence selection. A stroke s_p can have symmetry correspondences with several other strokes. Spattially compatible stroke candidates are merged into groups S^k_p.

A group must have at least one selected correspondence.

$$\boldsymbol{S^k_p} \leq \sum_q \boldsymbol{c_{pq}} \tag{11}$$

Similarly, a group has to be selected if at least one if its correspondences is selected.

$$S^k_p \leq \sum_{\mathbf{c} \in C^k_p} \mathbf{c} \tag{12}$$

, where C^k_p denotes the set of all correspondences forming group S^k_p.

Finally, a group can only be selected if the stroke's selection variable $\boldsymbol{s_p}$ has also been selected.

$$\sum_k \boldsymbol{S^k_p} \leq \boldsymbol{s_p} \tag{13}$$

Two strokes s_p and s_q can have several symmetry correspondence *candidates* with each other, the set of which is denoted by C_{pq}. However, only one of them should contribute to the selected group.

$$\sum_{c \in C_{pq}} \boldsymbol{c} \leq 1, \forall p, q \tag{14}$$

The first pass of our method constraints all strokes to be at least symmetric w.r.t. one global symmetry axis $k \in \{x, y, z\}$. A stroke can only be selected if at least one of its correspondences is along k.

$$\boldsymbol{s_p} \leq \sum_q \boldsymbol{c_{pq}^k} \tag{15}$$

To favor the selection of closeby reconstructions S_{pq}, we minimize the distance between stroke reconstructions and the selected stroke group $\text{dist}(S_p^k, S_{pq})$:

$$F_{\text{proximity}} = \sum_{p,k} \sum_{c_{pq} \in C_p^k} \text{dist}(S_p^k, S_{pq})\mathbf{c_{pq}}. \tag{16}$$

Curved lines. 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 then measure the overall support of curved strokes as

$$F_{\text{support}} = \sum_{p,q} \text{overlap}(S_{pq}, S'_{qp})\mathbf{c}_{pq}. \tag{17}$$

2.2 Connectivity

2.2.1 Intersection selection. A naive solution to identify 3D intersections would be to pre-compute intersections between all pairs of groups (S_p^k, S_q^l), and then to set the binary variable $\mathbf{i}_{pq}$ to 1 during optimization if the selected groups form an intersecting pair. However, this solution is computationally prohibitive because it requires associating a binary variable $\mathbf{i}_{pq}^{kl}$ with each possible pair of group, and constraining this variable to only be selected if and only if the two groups are selected, yielding numerous *quadratic* constraints of the form $\mathbf{i}_{pq}^{kl} = \mathbf{S_p^k S_q^l}$. Our solution to avoid these additional binary variables consists in detecting intersections between consolidated groups *within* the integer program. Before optimization, we project each 2D intersection i_{pq} onto all 3D groups S_p^k and S_q^l. During optimization, we evaluate the distance between these projections for the selected groups. Directly computing such distances is a non-linear operation, formulating which in an integer program would require quadratic constraints. We instead employ a *symmetric range constraints* formulation, where we only *prohibit* the selection of an intersection if the depth difference between the projections is outside a specified range. If the difference lies within these bounds, we let the solver decide whether $\mathbf{i}_{pq}$ should be set to 1 to maximize the connectivity terms of the objective function.

More specifically, the projected depth of i_{pq} onto S_p^k is $depth(S_p^k, i_{pq})$. Since only a single stroke group can be selected, the projected depth of i_{pq} on the final reconstruction of s_p is

$$depth(s_p, i_{pq}) = \sum_k depth(S_p^k, i_{pq})\boldsymbol{S_p^k} \tag{18}$$

Employing a symmetric range constraint formulation, i_{pq} can only be selected as a 3D intersection if the following hold:

$$depth(s_p, i_{pq}) - depth(s_q, i_{pq}) \leq \tau(i_{pq}) \tag{19}$$

and

$$-(depth(s_p, i_{pq}) - depth(s_q, i_{pq})) \leq \tau(i_{pq}) \tag{20}$$

where $\tau(i_{pq})$ is a threshold set to 10% of the median length of all stroke groups S_p^k and S_q^l.

For $\boldsymbol{i_{pq}}$ to be selected, equations 2.2.1 and 2.2.1 must hold. In our integer program, this is achieved via the following constraints

$$depth(s_p, i_{pq}) - depth(s_q, i_{pq}) \leq \tau(i_{pq}) + M(1 - \boldsymbol{i_{pq}}) \tag{21}$$

$$-(depth(s_p, i_{pq}) - depth(s_q, i_{pq})) \leq \tau(i_{pq}) + M(1 - \boldsymbol{i_{pq}}) \tag{22}$$

, where M is a big constant. This ensures that $\boldsymbol{i_{pq}}$ is equal to 0 if the depth difference is bigger than $\tau(i_{pq})$.

Additionally, an intersection i_{pq} can only be selected if both s_p and s_q have been reconstructed via the selection of a stroke group.

$$\boldsymbol{i_{pq}} \leq \boldsymbol{S_p^k} \tag{23}$$

$$\boldsymbol{i_{pq}} \leq \boldsymbol{S_q^k} \tag{24}$$

To avoid the reconstruction of dangling strokes, we enforce that a selected stroke must have at least one intersection.

$$\boldsymbol{s_p} \leq \sum_q \boldsymbol{i_{pq}} \tag{25}$$

2.2.2 Line coverage. To measure sketch connectivity, we compute the line-coverage for each stroke:

$$F_{\text{coverage}} = \sum_p \left(\max_{i \in \mathcal{I}_p} t_p(i) - \min_{i \in \mathcal{I}_p} t_p(i) \right) \mathbf{s}_p, \tag{26}$$

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.

In our integer program, we choose the minimum and maximum intersection by equipping each intersection i_{pq} with two binary variables $\boldsymbol{a_{pq}^F}$ and $\boldsymbol{a_{pq}^L}$, respectively. The line-coverage can then be reformulated as

$$F_{\text{coverage}} = \sum_p \sum_{i_{pr} \in \mathcal{I}_p} t_p(i_{pr})\mathbf{a}_{pr}^L - \sum_{i_{pr} \in \mathcal{I}_p} t_p(i_{pr})\mathbf{a}_{pr}^F. \tag{27}$$

For each stroke, at most one minimum and maximum intersection can be selected.

$$\sum_q \boldsymbol{a_{pq}^F} \leq 1 \tag{28}$$

$$\sum_q \boldsymbol{a_{pq}^L} \leq 1 \tag{29}$$

An intersection can only be selected as a minimum or maximum intersection if i_{pq} is a 3D intersection and not an occlusion.

$$\boldsymbol{a_{pq}^L} \leq \boldsymbol{i_{pq}} \tag{30}$$

$$\boldsymbol{a_{pq}^F} \leq \boldsymbol{i_{pq}} \tag{31}$$

At least one minimum and maximum intersection should be selected if a 3D intersection is selected along stroke s_p.

$$\sum_q \boldsymbol{a_{pq}^F} \geq \boldsymbol{i_{pq}}, \forall i_{pq} \tag{32}$$

$$\sum_q \boldsymbol{a_{pq}^L} \geq \boldsymbol{i_{pq}}, \forall i_{pq} \tag{33}$$

2.2.3 *Ellipses.*

Coverage cannot be measured on ellipses, since they do not have a start and end. Following Gryaditskaya et al. [2020], we instead favor 3D reconstructions of ellipses that are most circular. We achieve this goal by penalizing less circular interpretations $e(S_p^k) = \frac{\sqrt{(a^2+b^2)}}{a}$ where a and b are the major and minor axes of the reconstructed ellipse, yielding the penalty term for elliptic strokes

$$F_{\text{ellipses}} = -\sum_{p,k} e(S_p^k)\mathrm{S}_p^k. \tag{34}$$

We use F_{ellipses} in place of F_{coverage} for elliptic strokes.

2.2.4 *Stroke anchoring.*

We favor *well-anchored* stroke reconstructions. A stroke is weakly anchored if it passes through a single high-valence intersection and fully anchored if it passes through at least 2 high-valence intersections. The anchoring quality of a stroke s_p is expressed by the binary variables w_p and f_p, representing if s_p is weakly or fully anchored, respectively. We penalize the strokes inversely to their degree of anchoring:

$$F_{\text{anchoring}} = -\sum_{p} \left(2 - \mathbf{w}_p - \mathbf{f}_p\right)\mathbf{s}_p, \tag{35}$$

For each high-valence intersection v_{pq}, we count the number of differently axis-classified strokes involved in neighbouring intersections. An intersection belongs to a certain axis-class if one of its two strokes belongs to that axis-class. In pre-processing, during camera calibration, each stroke gets assigned an axis label which indicates which one of the three major vanishing points it converges towards, or if it does not converge to one at all. So each stroke s_p has an axis label $k \in x, y, z, w$, with w meaning that a stroke does not converge to one of the major vanishing points.

Each high-valence intersection v_{pq} is equipped with axis-class activation variables $\boldsymbol{k_{pq}}$, where k denotes the axis. Each such axis variable comes with a set of intersections that involve strokes from the corresponding axis $\mathcal{I}_{pq}^k$.

$\boldsymbol{k_{pq}}$ can only be activated if at least one of its intersections is selected:

$$\boldsymbol{k_{pq}} \leq \sum_{i \in \mathcal{I}_{pq}^k} \boldsymbol{i} \tag{36}$$

And a high-valence intersection can only be activated if there are at least 3 differently aligned axis labels selected:

$$\sum_{k} \boldsymbol{k_{pq}} \geq 3 - M(1 - \boldsymbol{v_{pq}}) \tag{37}$$

Selected high-valence intersections should be at a certain distance from each other to provide sufficient anchoring. To account for this, we impose that the arc-parameter distance of selected high-valence intersections is bigger than 0.5, i.e., it should span at least half of the stroke. Here, we use a similar mechanism to the measuring the line-coverage.

For each stroke, we select only two high-valence intersections, a *first* and a *last* high-valence intersection. We equip each high-valence intersection v_{pq} with two binary variables $\boldsymbol{v_{pq}^F}$ and $\boldsymbol{v_{pq}^L}$.

At most one first and last high-valence intersection can be selected:

$$\sum_{q} \boldsymbol{v_{pq}^F} \leq 1 \tag{38}$$

$$\sum_{q} \boldsymbol{v_{pq}^L} \leq 1 \tag{39}$$

Select at most two high-valence intersections:

$$\sum_{q} \boldsymbol{v_{pq}} \leq 2 \tag{40}$$

High-valence intersections can only be selected as a first or last intersection, if the actual high-valence intersection is realized:

$$\boldsymbol{v_{pq}^L} \leq \boldsymbol{v_{pq}} \tag{41}$$

A fully anchored strokes must have high-valence intersections that are sufficiently distant from each other:

$$\sum_{q} \boldsymbol{v_{pq}^L} t(i_{pq}) - \sum_{q} \boldsymbol{v_{pq}^F} t(i_{pq}) \geq 0.5 - M(1 - \boldsymbol{f_p}) \tag{42}$$

Weakly anchored stroke variables $\boldsymbol{w_p}$ should only be activated if there is at least one high-valence intersection.

$$\boldsymbol{w_p} \leq \sum_{q} \boldsymbol{v_{pq}} \tag{43}$$

2.3 Score Function

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

$$F_{\text{total}} = F_{\text{symmetry}} + F_{\text{support}} + F_{\text{proximity}} + F_{\text{anchoring}} + F_{\text{coverage}} + F_{\text{ellipses}}, \tag{44}$$

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 $\lambda_{\text{symmetry}} = 2$, $\lambda_{\text{support}} = 10$, $\lambda_{\text{proximity}} = -100$, $\lambda_{\text{anchoring}} = 5$, $\lambda_{\text{coverage}} = 4$ and $\lambda_{\text{ellipses}} = 1$..

3 TIMINGS

We provide timings for sketches of increasing complexity in Table 1. Computation is dominated by three main stages, of roughly equal time. The first stage, pre-processing, is necessary to calibrate the perspective camera and compute intersections between all strokes. The second stage consists in detecting and reconstructing all candidate symmetry correspondences. Finally, we call our main algorithm to perform the global and local symmetric reconstructions.

4 3D RECONSTRUCTIONS

We provide on the next pages a gallery of 3D reconstructions produced with our method. For each result, we show the input sketch (left) and a 3D rotation (right).

REFERENCES

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

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

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

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* 33, 4 (2014).

Strokes	Pre-processing	Candidate generation	Global reconstruction	Local reconstruction	Completion	Total time
61	1.8	0.4	3.7	0.8	0.2	5.0
116	3.5	1.9	3.3	1.2	1.4	9.7
165	2.5	0.7	1.5	1.1	0.4	5.5
215	3.7	2.4	3.5	2.2	0.5	10.6
393	6.6	3.1	5.3	1.5	1.1	14.9

Table 1. Runtime in minutes for representative sketches.