Supplementary Material: Voronoi Scissors

1 Regularization

Following [Wu et al. 2024], our regularization term is as follows:

$$\mathcal{L}_{\text{reg}} = w_{\text{wall}}\mathcal{L}_{\text{wall}} + w_{\text{topo}}\mathcal{L}_{\text{topo}} + w_{\text{Lloyd}}\mathcal{L}_{\text{Lloyd}}$$

Wall loss encourages simple, axis-aligned boundaries between pieces by penalizing the L_1 norm of each internal edge vector:

$$\mathcal{L}_{\text{wall}} = w_{\text{wall}} \sum_{(v_i,v_j)\in\mathcal{E}} \|v_i - v_j\|_{L_1}$$

where $\mathcal{E}$ is the set of edges between adjacent pieces, and v_i, v_j are the Voronoi vertices of each edge.

Lloyd loss regularizes site density by encouraging each site $\mathbf{p}^A_{km}$ (resp. $\mathbf{p}^B_{km}$) to remain near the centroid $\mathbf{c}^A_{km}$ (resp. $\mathbf{c}^B_{km}$) of its Voronoi cell:

$$\mathcal{L}_{\text{Lloyd}} = w_{\text{Lloyd}} \sum_{k=1}^{K}\sum_{m=1}^{M} \left(\|\mathbf{p}^A_{km} - \mathbf{c}^A_{km}\|^2 + \|\mathbf{p}^B_{km} - \mathbf{c}^B_{km}\|^2\right)$$

Topology loss ensures each piece forms a single connected region and that specified inter-piece adjacencies are satisfied, by pulling each site toward a target position $\mathbf{t}_{km}$:

$$\mathcal{L}_{\text{topo}} = w_{\text{topo}} \sum_{k=1}^{K}\sum_{m=1}^{M} \left(\|\mathbf{p}^A_{km} - \mathbf{t}^A_{km}\|^2 + \|\mathbf{p}^B_{km} - \mathbf{t}^B_{km}\|^2\right)$$

where $\mathbf{t}_{km}$ is set to the nearest site within the same piece's connected component if the piece is fragmented, or to the nearest site of an unmet adjacent piece otherwise. In our implementation, we use $w_{\text{wall}} = 0.001$, $w_{\text{topo}} = 15.0$, $w_{\text{Lloyd}} = 0.2$ for all the experiments.

We empirically set $w_{\text{topo}} = 15.0$, which encourages pieces to become connected early enough for $\mathcal{L}_{\text{iso}}$ to act on the full piece while preserving sufficient flexibility to avoid poor local minima. In practice, this weight leads to a single connected component by the end of the optimization (500/500 initializations for the hat–ghost example). As shown in Fig. 1, increasing the weight to $w_{\text{topo}} = 30.0$ leads to a higher L_2 distance (0.0239 vs. 0.0202) between corresponding boundary points. Handling disconnected components explicitly requires dedicated topology repair; instead, we simply discard such rare initializations.

2 Degrees of Freedom Analysis of the Coarse Graph Deformation

In this section, we analyze the degrees of freedom (DoF) of the coarse graph deformation. We show that the optimization is generally not over-constrained.

In Eq. 6, vertex positions from both graphs and transformation matrices are optimization variables: each vertex contributes two unknowns, and each transformation contributes three unknowns. The constraints are obtained by treating each polygon independently and summing over all polygon vertices. Let V, E, f, B, and n_i denote the numbers of vertices, edges, bounded faces, boundary edges, and vertices of bounded face i, respectively, in a coarse graph. Let N_v and N_c denote the numbers of variables and constraints. Then $N_v = 3f + 2 \times 2V, N_c = 2\sum_{i=1}^{f} n_i$. By the handshaking lemma, $\sum_{v\in V}\deg(v) - B = 2E - B = \sum_{i=1}^{f} n_i$. By Euler's formula, DoF $= N_v - N_c = 4V + 3f - 2(2E - B) = 4 + 2B - f$.

We observe a positive DoF in all of our experiments.

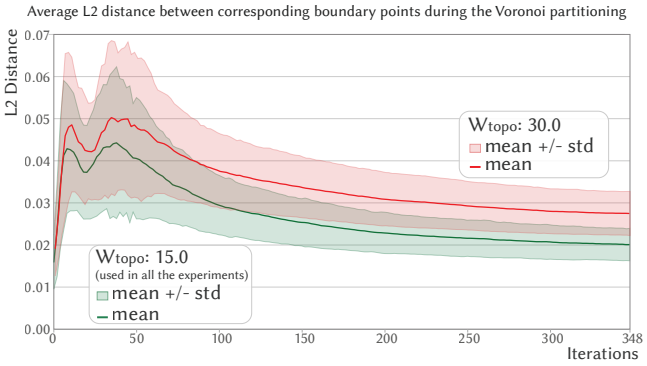

Fig. 1. **Ablation of w_{topo} over 500 random initializations for the four-piece hat–ghost design.** The curves show the mean average L_2 distance between corresponding boundary points as a function of optimization iterations, with shaded regions indicating ±1 standard deviation. All shapes are normalized to unit area. We use $w_{\text{topo}} = 15.0$ (green) throughout our experiments, while $w_{\text{topo}} = 30.0$ (red) converges to a higher L_2 distance.

3 Multiple Solutions

Each successful initialization will produce a valid dissection results. In Fig. 2, we show the three best dissection solutions found for the Bunny–Egg shape pair.

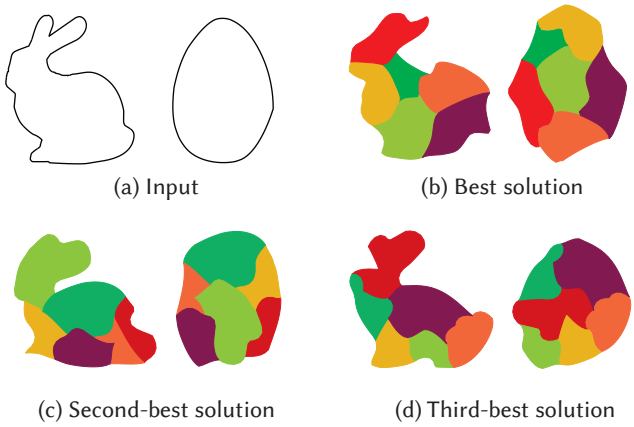

Fig. 2. Top three dissection solutions for the Bunny–Egg shape pair.

4 Parameters

We evaluate the key parameters of the optimization process described in Section 5.1, over which we perform a grid search. As shown in Fig. 3, α is the minimum normalized boundary distance between adjacent vertices required for inserting a new vertex, as described in Line 4 of Alg. 2. Increasing α results in fewer inserted vertices, while decreasing it produces more. λ_{shape} controls how

strongly vertices are attracted to the target shape — larger values better preserve the target geometry. The barrier distance $\hat{d}$ governs strip thickness; smaller values produce thinner strips. Together, these parameters perturb the graph to introduce greater structural variation. In our implementation, we search over $\alpha \in \{0.08, 0.09, 0.10, 0.11, 0.12\}$, $\lambda_{\text{shape}} \in \{0.2, 2.0\}$, and $\hat{d} \in \{0.100, 0.105, 0.110, 0.115\}$. We test all parameter combinations and select the one with the smallest L_2 distance to the input shape.

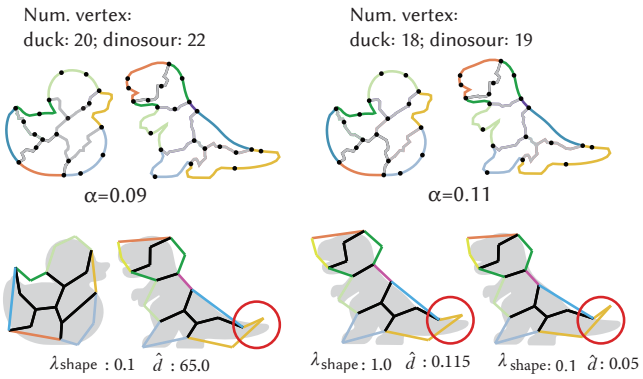

Fig. 3. Effect of optimization parameters. **Top:** The threshold α controls when a new vertex is inserted onto the target face. A smaller α generally yields more inserted vertices (left), while a larger α yields fewer (left). **Bottom:** A larger λ_{shape} more strongly preserves vertices on the target shape. A smaller barrier distance $\hat{d}$ enables producing slightly thinner strips.

5 Chains Statistic

We report the average number of chains and average number of edges per chain as the number of pieces increases. We observe that the number of chains increases linearly with the number of pieces — this is expected, as more pieces allow the boundary to be divided more finely. The average number of edges per chain, on the other hand, grows logarithmically with the number of pieces, indicating that internal connections scale sub-linearly as the piece count grows.

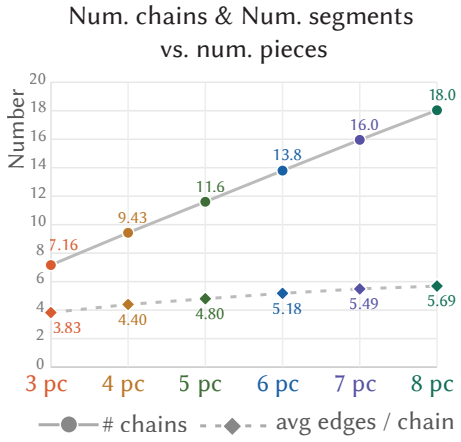

Fig. 4. successful rate, approximation error

6 Isometry Loss $\mathcal{L}_{\text{iso}}$ and its Gradient

Alg. 1 outlines the isometry loss $\mathcal{L}_{\text{iso}}$ and its gradient.

Algorithm 1 Isometry Loss $\mathcal{L}_{\text{iso}}$ and Gradient for Piece k

Require: $\gamma_k^B = \{\mathbf{b}_{t_B}\}_{t_B=1}^{T_B}$, $\gamma_k^A = \{\mathbf{a}_{t_A}\}_{t_A=1}^{T_A}$, M, $\varepsilon > 0$
Ensure: Scalar loss $\mathcal{L}_{\text{iso}}$, gradients $\{\partial \mathcal{L}_{\text{iso}} / \partial \mathbf{b}_{t_B}\}_{t_B=1}^{T_B}$

— Preprocessing —
Resample $\gamma_k^A \to \{\mathbf{a}_m\}_{m=1}^M$, $\gamma_k^B \to \{\mathbf{b}_m\}_{m=1}^M$ by arc-length
$\bar{\mathbf{b}} \leftarrow \frac{1}{M}\sum_m \mathbf{b}_m$, $\bar{\mathbf{a}} \leftarrow \frac{1}{M}\sum_m \mathbf{a}_m$
$\mathbf{p}_m \leftarrow \mathbf{b}_m - \bar{\mathbf{b}}$, $\mathbf{q}_m \leftarrow \mathbf{a}_m - \bar{\mathbf{a}}$
— Discrete rotation search (detached) —
for $\theta \in \{0°, 1°, \ldots, 359°\}$ **do**
  $L(\theta) \leftarrow \min_\kappa \frac{1}{M}\sum_{m=1}^M \left\| R(\theta)\,\mathbf{p}_m - \mathbf{q}_{(m+\kappa) \bmod M} \right\|^2$
end for
$\theta^* \leftarrow \arg\min_\theta L(\theta)$
$\mathbf{t}^* \leftarrow \bar{\mathbf{a}} - R(\theta^*)\,\bar{\mathbf{b}}$

— CPD-EM correspondence refinement (detached) —
Initialize $W_{t_A t_B} \propto \exp\!\left(-\|\mathbf{a}_{t_A} - R(\theta^*)\mathbf{b}_{t_B} - \mathbf{t}^*\|^2 / 2\sigma^2\right)$, $\sigma^2 \leftarrow 0.1$
for $t = 1$ **to** 5 **do**
  $R^*, \mathbf{t}^* \leftarrow \text{WeightedProcrustes}(\gamma_k^A, \gamma_k^B, W_{t_A t_B})$
  $W_{t_A t_B} \propto \exp\!\left(-\|\mathbf{a}_{t_A} - R^*\mathbf{b}_{t_B} - \mathbf{t}^*\|^2 / 2\sigma^2\right)$
  $\sigma^2 \leftarrow \dfrac{\sum_{t_A,t_B} W_{t_A t_B}\, \|\mathbf{a}_{t_A} - R^*\mathbf{b}_{t_B} - \mathbf{t}^*\|^2}{2\sum_{t_A,t_B} W_{t_A t_B}}$
end for

— Forward pass (differentiable) —
$A \leftarrow \sum_{t_A,t_B} W_{t_A t_B}\, \mathbf{p}_{t_B}\, \mathbf{q}_{t_A}^\top$
$\alpha \leftarrow A_{00} + A_{11}$, $\beta \leftarrow A_{10} - A_{01}$, $\rho \leftarrow \max\!\left(\sqrt{\alpha^2 + \beta^2}, \varepsilon\right)$
$c^* \leftarrow \alpha/\rho$, $s^* \leftarrow \beta/\rho$, $R^* \leftarrow \left(\begin{smallmatrix} c^* & -s^* \\ s^* & c^* \end{smallmatrix}\right)$
$\mathbf{e}_{t_A t_B} \leftarrow R^*\mathbf{p}_{t_B} - \mathbf{q}_{t_A} \quad \forall t_A, t_B$
$\mathcal{L}_{\text{iso}} \leftarrow \sum_{t_A,t_B} W_{t_A t_B}\, \|\mathbf{e}_{t_A t_B}\|^2$

— Backward pass —
$\dfrac{\partial \mathcal{L}_{\text{iso}}}{\partial R^*} \leftarrow 2\sum_{t_A,t_B} W_{t_A t_B}\, \mathbf{e}_{t_A t_B}\, \mathbf{p}_{t_B}^\top$
$\gamma \leftarrow \dfrac{1}{\rho^2}\,\text{tr}\!\left(\left(\dfrac{\partial \mathcal{L}_{\text{iso}}}{\partial R^*}\right)^\top R^* J\right)$, $J = \left(\begin{smallmatrix} 0 & -1 \\ 1 & 0 \end{smallmatrix}\right)$
$G_A \leftarrow \gamma \begin{pmatrix} -\beta & -\alpha \\ \alpha & -\beta \end{pmatrix}$
for $t_B = 1$ **to** T_B **do**
  $\mathbf{v}_{t_B} \leftarrow \sum_{t_A} W_{t_A t_B}\, \mathbf{q}_{t_A}$
  $\dfrac{\partial \mathcal{L}_{\text{iso}}}{\partial \mathbf{p}_{t_B}} \leftarrow \underbrace{2\sum_{t_A} W_{t_A t_B}\, R^{*\top}\mathbf{e}_{t_A t_B}}_{\text{direct}} + \underbrace{G_A\,\mathbf{v}_{t_B}}_{\text{indirect}}$
end for
for $t_B = 1$ **to** T_B **do**
  $\dfrac{\partial \mathcal{L}_{\text{iso}}}{\partial \mathbf{b}_{t_B}} \leftarrow \dfrac{\partial \mathcal{L}_{\text{iso}}}{\partial \mathbf{p}_{t_B}} - \dfrac{W_{\cdot t_B}}{W_{\text{tot}}}\sum_{t'_B} \dfrac{\partial \mathcal{L}_{\text{iso}}}{\partial \mathbf{p}_{t'_B}}$
end for
return $\mathcal{L}_{\text{iso}}$, $\{\partial \mathcal{L}_{\text{iso}} / \partial \mathbf{b}_{t_B}\}_{t_B=1}^{T_B}$

7 Common Refinement

We detail the Common Refinement algorithm (Alg. 2), which equalizes vertex counts across corresponding polygon pairs described in Sec. 5.1. The algorithm is applied S iterations. At each iteration, for each pair of corresponding pieces, we try to equalize their numbers of vertices. To do so, we insert vertices into the piece with fewer vertices and propagate each inserted vertex through the chain of correspondences to other edges. The propagation terminates whenever the newly propagated vertex reaches the external boundary of the shape. We set $S = 10$ in our implementation, though convergence is typically reached in fewer iterations. As a safeguard, if convergence is not reached after 10 iterations, we restrict further insertions to the external boundary, which guarantees termination. In practice, this safeguard is barely needed.

Algorithm 2 Common Refinement

Require: For each piece pair γ_k^A, γ_k^B: corner sets C_k^A, C_k^B; normalized arc-length tolerance $\alpha \in (0, 1)$

Assume: $|C_k^A| \geq |C_k^B|$

for each piece pair (γ_k^A, γ_k^B), $k \in \{1, \ldots, K\}$ **do**
  for each corner $\mathbf{a} \in C_k^A$ **do**
    $\mathbf{q} \leftarrow \mathrm{Q}_k \mathbf{a}$ ▷ map corner to the frame of B via rigid transform $\mathrm{Q}_k \in SE(2)$
    $s^* \leftarrow \textsc{ClosestPoint}(\mathbf{q}, \gamma_k^B)$ ▷ $s^* \in [0, 1)$: arc-length param
    if $\textsc{CircularDistance}(s^*, C_k^B) > \alpha$ **then**
      $C_k^B \leftarrow C_k^B \cup \{\gamma_k^B(s^*)\}$ ▷ insert new corner into B at s^*
    end if
  end for
  while $|C_k^A| > |C_k^B|$ **do**
    compute candidate parameters s_i^* by mapping corners in C_k^A onto γ_k^B
    $s^* \leftarrow \arg\max_{s_i^*} \textsc{CircularDistance}(s_i^*, C_k^B)$
    $C_k^B \leftarrow C_k^B \cup \{\gamma_k^B(s^*)\}$
  end while
  while $|C_k^B| > |C_k^A|$ **do**
    compute candidate parameters s_i^* by mapping corners in C_k^B onto γ_k^A
    $s^* \leftarrow \arg\max_{s_i^*} \textsc{CircularDistance}(s_i^*, C_k^A)$
    $C_k^A \leftarrow C_k^A \cup \{\gamma_k^A(s^*)\}$
  end while
end for
return $C^A = \bigcup_k C_k^A, \qquad C^B = \bigcup_k C_k^B$