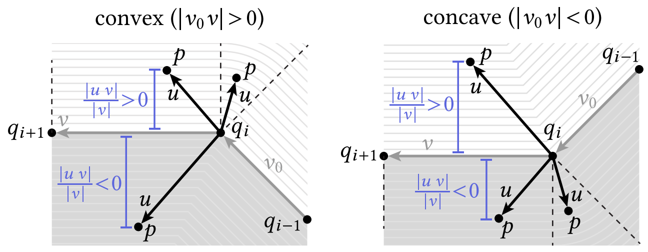


Figure 17: Quantities used to evaluate a differentiable expression for the SDF of a simple, convex or nonconvex polygon.

A FORMULAS FOR PENETRATION DEPTH

This appendix catalogues expressions for the penetration depth of common shape pairs, building on 2D SDF expressions provided by Quilez [2021]. When available, closed-form expressions are generally preferable to polygonal approximation—providing more accurate evaluation at lower computational cost. Throughout we assume operations like max and abs are applied to vectors componentwise.

A.1 Circle-Circle

Consider two circles A, B with centers $\mathbf{c}_A, \mathbf{c}_B \in \mathbb{R}^2$ and radii $r_A, r_B \in \mathbb{R}_{>0}$. Their difference $C = A - B$ is a circle with center $\mathbf{c}_C = \mathbf{c}_A - \mathbf{c}_B$ and radius $r_C = r_A + r_B$. The penetration depth is then

$$\phi_C(\mathbf{0}) = |\mathbf{c}_C| - r_C.$$

A.2 Rectangle-Rectangle

Consider two rectangles A, B with centers $\mathbf{c}_A, \mathbf{c}_B \in \mathbb{R}^2$, widths $w_A, w_B \in \mathbb{R}_{>0}$, and heights $h_A, h_B \in \mathbb{R}_{>0}$. Their difference $C = A - B$ is a rectangle with center $\mathbf{c}_C = \mathbf{c}_A - \mathbf{c}_B$, width $w_C = w_A + w_B$, and height $h_C = h_A + h_B$. The penetration depth is then

$$\phi_C(\mathbf{0}) = |\max(\mathbf{d}, (0,0))| + \min(\max(\mathbf{d}_x, \mathbf{d}_y), 0),$$

where $\mathbf{d} = \mathrm{abs}(\mathbf{c}_A) - (w_C, h_C)$.

A.3 Shape-Circle

Suppose B is any shape with known SDF ϕ_B. For circle A with center $\mathbf{c}_A$ and radius r_A, the SDF of the difference $C = B - A$ is then just

$$\phi_C(\mathbf{0}) = \phi_{-B}(-\mathbf{c}_A) - r_A.$$

Intuitively, a Minkowski sum with a circular disk "thickens" shape B, which can be achieved by simply offsetting the SDF of B.

A.4 Polygon-Polygon

The Minkowski difference of two polygons A, B can be computed using the methods discussed in Section 3.3. The SDF $\phi_C(\mathbf{p})$ of the resulting polygon C could then be evaluated by multiplying unsigned distance by a sign indicating whether $\mathbf{p}$ is inside or outside C [Quilez 2020]. However, this approach leads to unstable gradient evaluation: on the polygon boundary, the gradient of the SDF is well-defined—but the intermediate unsigned distance is not differentiable. Since penetration depth $\phi_C(0)$ is a central quantity in our algorithm, we use a different, carefully-designed formula. This formula need only work for simple polygons, since the polygon in question is always the boundary of a solid region $C \subset \mathbb{R}^2$.

Algorithm 1 PolygonSDF($\mathbf{p}, \mathbf{q}_0, \ldots, \mathbf{q}_n$)

Input: A point $\mathbf{p} \in \mathbb{R}^2$ and a polygon $\mathbf{q}_0, \ldots, \mathbf{q}_n \in \mathbb{R}^2$ with $\mathbf{q}_n = \mathbf{q}_0$.
Output: The signed distance to the given polygon, evaluated at $\mathbf{p}$.

```
$d \leftarrow \infty$                          ▷ unsigned distance (squared)
$e \leftarrow \top$                            ▷ $\top$ if edge, $\bot$ if vertex
$j \leftarrow 0$                               ▷ edge index, only used if $e = \top$
$s \leftarrow 1$                               ▷ sign, only used if $e = \bot$
$\mathbf{v}_0 \leftarrow \mathbf{q}_0 - \mathbf{q}_{n-1}$      ▷ vector for previous edge
for $i = 0, \ldots, n-1$ do                    ▷ iterate over edges
    $\mathbf{u} \leftarrow \mathbf{p} - \mathbf{q}_i$
    $\mathbf{v} \leftarrow \mathbf{q}_{i+1} - \mathbf{q}_i$
    $z \leftarrow \langle \mathbf{v}, \mathbf{v} \rangle$      ▷ squared length of edge
    if $0 \le \langle \mathbf{u}, \mathbf{v} \rangle < z$ then    ▷ closest point to line lies on edge
        $d' \leftarrow |\mathbf{u}\ \mathbf{v}|^2/z$
        if $d' < d$ then
            $(d, e, j) \leftarrow (d', \top, i)$
    else                                       ▷ vertex may be closer than edge
        $d' \leftarrow \langle \mathbf{u}, \mathbf{u} \rangle$
        if $d' < d$ then
            $(d, e, s) \leftarrow (d', \bot, 1)$
            if $|\mathbf{v}_0\ \mathbf{v}| < 0$ then      ▷ concave vertex
                $s \leftarrow -1$
    $\mathbf{v}_0 \leftarrow \mathbf{v}$
if $e = \top$ then
    $\mathbf{u} \leftarrow \mathbf{p} - \mathbf{q}_j$
    $\mathbf{v} \leftarrow \mathbf{q}_{j+1} - \mathbf{q}_j$
    return $|\mathbf{u}\ \mathbf{v}|/|\mathbf{v}|$              ▷ signed distance for edge
else return $s\sqrt{d}$                        ▷ signed distance for vertex
```

In particular, suppose we are given a list of vertex coordinates $\mathbf{q}_0, \ldots, \mathbf{q}_n = \mathbf{q}_0$ in counterclockwise order, describing a simple but possibly nonconvex polygon (Figure 17). We will use $|\mathbf{a}\ \mathbf{b}|$ to denote the determinant of a 2×2 matrix with columns $\mathbf{a}$ and $\mathbf{b}$. For any point $\mathbf{p} \in \mathbb{R}^2$, there are two possibilities: the closest point to $\mathbf{p}$ is either a vertex $\mathbf{q}_i$, or it lies on an edge with endpoints $\mathbf{q}_i, \mathbf{q}_{i+1}$. Let

$$\mathbf{u} := \mathbf{p} - \mathbf{q}_i \quad \text{and} \quad \mathbf{v} := \mathbf{q}_{i+1} - \mathbf{q}_i.$$

The distance to any vertex $\mathbf{q}_i$ is then

$$d_{\mathbf{q}_i}(\mathbf{p}) := |\mathbf{u}|.$$

The closest point can be on the edge only if

$$0 < \langle \mathbf{u}, \mathbf{v} \rangle < \langle \mathbf{v}, \mathbf{v} \rangle. \tag{3}$$

If this condition holds, the signed distance to the edge is the (signed) height of triangle (q_i, p, q_{i+1}), which can be obtained by dividing its area (given by a cross product) by its base length:

$$d^{\pm}_{\mathbf{q}_i\mathbf{q}_{i+1}}(\mathbf{p}) = |\mathbf{u}\ \mathbf{v}|/|\mathbf{v}|.$$

The procedure, then, is to compute $d_{\mathbf{q}_i}$ for all vertices, and $d^{\pm}_{\mathbf{q}_i\mathbf{q}_{i+1}}$ for all edges that satisfy Equation 3. If the smallest value from this set is on an edge, then we just return this value. Otherwise, the closest point is some vertex $\mathbf{q}_i$; the distance to this vertex is positive if it is convex, and negative if it is concave. More precisely, $\mathbf{q}_i$ is convex if and only if $|\mathbf{v}_0\ \mathbf{v}| > 0$, where $\mathbf{v}_0 := \mathbf{q}_i - \mathbf{q}_{i-1}$. Algorithm 1 gives an optimized version of this procedure.

B HALFPLANE TRICK

When A and B are both convex polygons, their difference C is also convex—and can hence be expressed as an intersection of half planes. In particular, for each edge e of ∂A, the signed distance to the corresponding half plane H_e is given by

$$\phi_{H_e^A}(\mathbf{x}) = \langle \mathbf{x}, \mathbf{n}_e \rangle - \max_{\mathbf{v}\in V_{-B}} \langle \mathbf{p}_e + \mathbf{v}, \mathbf{n}_e \rangle,$$

where $\mathbf{n}_e$ is the outward unit normal vector of e, $\mathbf{p}_e$ is any point of e, and V_{-B} is the vertex set of $-B$. The SDF $\phi_{H_e^{-B}}(\mathbf{x})$ is defined analogously (maximizing over $\mathbf{v} \in V_A$). The SDF for C can then be approximated as

$$\tilde{\phi}_C(\mathbf{x}) = \max \left[\max_{e\in A} \phi_{H_e^A}(\mathbf{x}), \max_{e\in -B} \phi_{H_e^{-B}}(\mathbf{x}) \right].$$

Note that $\tilde{\phi}_C(\mathbf{x}) = \phi_C(\mathbf{x})$ for points x inside C, but they slightly differ outside, where $\tilde{\phi}$ satisfies the eikonal property $|\nabla\tilde{\phi}| = 1$, but is not an SDF. Such an approximation is sometimes called a *pseudo-SDF* [Marschner et al. 2023] or simply a *unit gradient field (UGF)* [Courter 2023]. For two convex polygons of size m and n, this halfplane trick has time complexity $O(mn)$, whereas the explicit sum of convex polygons takes $O(m+n)$ time.

C ADDITIONAL ENERGY FORMULAS

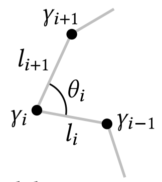
For reproducibility our examples, we provide formulas for the specific energies used in the generated examples.

C.1 Elastic Curves

The optimization of elastic curve γ in Figure 1 (A) and Figure 6, *left* is primarily achieved by minimizing the elastic energy

$$E = \int_0^L \kappa^2(s)\mathrm{d}s,$$

where κ is the curvature and L denotes the length of γ. The curves are represented by polygonal lines (parametrized by the vertex positions) and the elastic energy integral is approximated by a finite sum $E \approx \frac{1}{2}\sum_i \kappa_i^2(l_i + l_{i+1})$, where $l_i = \|\gamma_i - \gamma_{i-1}\|$ is the length of i-th line segment and κ_i is the discrete curvature at the vertex γ_i. To compute the discrete curvature we use the Steiner formula with line segment corner expansion $\kappa_i = 2\sin(\theta_i/2)$ from [Crane and Wardetzky 2017], where θ_i represents the angle at the vertex γ_i (see inset).



The specific energy used in the figures was $\mathcal{E}_e = (E - E_0)^2$, where E_0 is a prescribed constant set to 0 in Figure 6 and 1200 in Figure 15. To improve the optimization stability, the curve was also subjected to a penalty $\mathcal{P}_{el} = \sum_i (l_i - \bar{l})^2$ enforcing uniform distribution of points, where $\bar{l}$ is the mean edge length. Finaly, an objective $\mathcal{E}_l = (\sum_i l_i - L_0)^2$ encourages curves meet a prescribed total length L_0, set to 1000 and 1500 for Figure 6 and Figure 15, respectively. In total, the energies associated with each curve read

$$\mathcal{E} = \mathcal{E}_e + \mathcal{E}_l \quad \text{and} \quad \mathcal{P} = \mathcal{P}_m + \mathcal{P}_{el},$$

where $\mathcal{P}_m$ is sum of all Minkowski penalties present in the examples. These examples also include no-overlap penalties between all curves and disks, and containment penalties within the fixed outer region (detailed in Section 3).

C.2 Spectral Graph Energy

Spectral graph layout is a widely used method for graph visualization [Koren 2003]. For a graph $\mathcal{G} = (V, E)$ with vertices $V = \{1, \dots, n\}$ and edges $E \subset V \times V$, the spectral method finds the optimal embedding $\mathbf{v} = \{\mathbf{v}_1, \dots, \mathbf{v}_n\}$ by solving

$$\begin{aligned} \min_{\mathbf{v}} \quad & \mathcal{E} := \textstyle\sum_{(i,j)\in E} \|\mathbf{v}_i - \mathbf{v}_j\|^2, \\ \text{s.t.} \quad & \operatorname{var}(\mathbf{v}) := \textstyle\sum_{i\in V} \|\mathbf{v}_i - \bar{\mathbf{v}}\|^2 = 1, \end{aligned}$$

where $\bar{\mathbf{v}}$ is the mean of $\mathbf{v}$. Since both the objective and constraint are differentiable, we can directly implement it within our framework and simultaniously avoid overlap or enforce containment using Minkowski penalties. We use the same energy $\mathcal{E}$ but slightly reorganize the penalty to arrive at a graph centered at the origin:

$$\mathcal{P} = \left(\textstyle\sum_{i\in V} \|\mathbf{v}_i\|^2 - Sn\right)^2 + \|\textstyle\sum_{i\in V} \mathbf{v}_i\|^2,$$

where S, set to 3600 in Figure 6, dictates the scale of the graph.

C.3 t-SNE Energy

Another common energy-based visualization method is t-SNE introduced in [van der Maaten and Hinton 2008]. Here, the goal is to find a meaningful low-dimensional embedding $\mathbf{y}_1, \dots, \mathbf{y}_n \in \mathbb{R}^2$ of dataset with high-dimensional vectors $\mathbf{x}_1, \dots, \mathbf{x}_n \in \mathbb{R}^d$. The optimized energy is equal to Kullback-Leibler divergence

$$\mathcal{E} = \sum_{i,j=1}^{n} p(\mathbf{x}_j, \mathbf{x}_i, \sigma) \log\left(\frac{p(\mathbf{x}_j, \mathbf{x}_i, \sigma)}{p(\mathbf{y}_j, \mathbf{y}_i, 1)}\right)$$

of conditional probabilities computed as

$$p(\mathbf{x}_i, \mathbf{x}_j, \sigma) = \frac{\exp(-\|\mathbf{x}_i - \mathbf{x}_j\|^2/(2\sigma^2))}{\sum_{k\neq i}\exp(-\|\mathbf{x}_i - \mathbf{x}_k\|^2/(2\sigma^2))}.$$

The streamlined implementation, used in Figure 6, sets σ^2 to a fixed value $2 \cdot 10^{-4}$ instead of adjusting it based on the perplexity as suggested in [van der Maaten and Hinton 2008]. While minimizing the objective $\mathcal{E}$ we also simultaneously prevent overlap of the embedded text bounding boxes using Minkowski disjoint penalty $\mathcal{P}_d$ for all text pairs and even with additional circular element.

C.4 Point Repulsion

The example in Figure 12 Minkowski disjoint penalty $\mathcal{P}_d$ for line segments representing molecular bonds and simultaneously minimized the repulsive $\mathcal{E}_r$ and prescribed distance $\mathcal{E}_d$ energies:

$$\mathcal{E}_r = \sum_{i\neq j} \frac{1}{\|\mathbf{x}_i - \mathbf{x}_j\|^2 + \varepsilon}, \qquad \mathcal{E}_d = \sum_{(i,j)\in B} (\|\mathbf{x}_i - \mathbf{x}_j\| - d)^2,$$

where $\mathbf{x}_i \in \mathbb{R}^2$ denote the positions of atomic centers, B is the set of molecular bonds and the constants are set to $\varepsilon = 10^{-5}$ and $d = 60$.

C.5 Globally Injective Flattening

The example in Figure 11 minimizes a surface distortion energy $\mathcal{E}_D$ subject to non-overlap constraints between all pairs of triangles that share neither edges nor vertices, enforced via Minkowski penalties $\mathcal{P}_d$. We also constrain the area of the flattened mesh to be no smaller than a target area $\mathcal{A}_0$, to prevent it from collapsing to a point. A further penalty $\mathcal{P}_c$ keeps triangles inside the outer rectangle, and

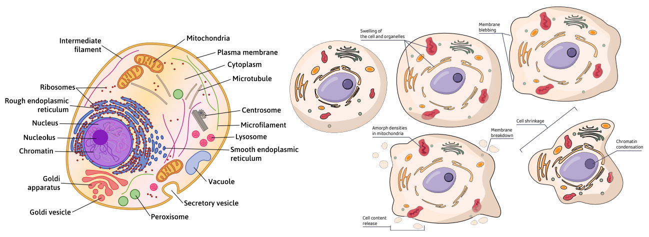


Figure 18: Original vector art "remixed" in Figure 9. Organelle shapes were extracted from the image at left, and rearranged within cell membrane shapes extracted from the image at right. (Illustrations by Volodymyr Ishchuk and Olha Pohrebniak, used with permission.)

Fig.	location	time (s)
1	*(S)*	4.08
1	*(I)*	17.54
1	*(G)*	32.15
1	*(R)*	60.57
1	*(A)*	73.25
1	*(P)*	29.35
1	*(H)*	74.24
9	*2nd from right*	23.42
11	*bottom right*	31.73
11	*top right*	22.69
7	*upper middle right*	0.04
15		73.25
7	*top right*	0.31
7	*lower middle left*	0.01
7	*bottom right*	0.06
7	*upper middle left*	1.57
7	*top left*	0.05
7	*middle center*	0.01
7	*lower middle right*	0.17
7	*bottom center right*	0.10
7	*bottom center left*	0.00
7	*middle left*	0.04
6	*center*	1.88
6	*right*	4.95
6	*left*	4.26
8	*left*	21.53
8	*center left*	16.02
8	*center right*	182.95
8	*right*	261.49
14	*left*	0.93
14	*center*	0.58
14	*right*	3.40
10	*left*	2.61
10	*right*	0.38
13		0.26
12		0.25

Table 2: Per-figure time to optimize diagrams shown throughout the paper. Timings were measured on a 2021 M1 Max Macbook Pro. As noted in Section 5.3, longer times correspond to naïve evaluation of all $O(n^2)$ pairs of penalties in dense packing scenarios.

in some instances, a penalty $\mathcal{P}_d$ prevents intersection with a fixed circular disk.

More specifically, the distortion energy is given by the discrete *Dirichlet energy* of the mapping from the input 3D mesh into the 2D plane [Crane et al. 2013, Section 7.4]. Letting $\mathbf{p}_i \in \mathbb{R}^2$ denote the 2D coordinates of vertex i, this energy can be expressed as a sum over all triangles ijk in the mesh M:

$$\mathcal{E}_D(\mathbf{p}) = \sum_{ijk\in M} w^k_{ij}(\mathbf{p}_i - \mathbf{p}_j)^2 + w^i_{jk}(\mathbf{p}_j - \mathbf{p}_k)^2 + w^j_{ki}(\mathbf{p}_k - \mathbf{p}_i)^2.$$

Here w are the *cotan weights*, given for each edge ij of triangle ijk by $w^k_{ij} = \frac{1}{2}\cot\theta^{jk}_i$, where θ^{jk}_i is the interior angle at corner i on the input 3D mesh. (Note that these weights are constant with respect to $\mathbf{p}$.) The area of the flattened mesh can be expressed via the *shoelace formula* [Crane et al. 2013, Exercise 7.9], obtained as a sum over edges ij in the mesh boundary ∂M:

$$\mathcal{A}(\mathbf{p}) = \tfrac{1}{2}\sum_{ij\in\partial M} \mathbf{p}_i \times \mathbf{p}_j.$$

To enforce the condition $\mathcal{A} \geq \mathcal{A}_0$, we add a penalty $\mathcal{P}_{\text{area}} = \max(0, \mathcal{A}_0 - \mathcal{A})^2$ to the objective of Equation 2.