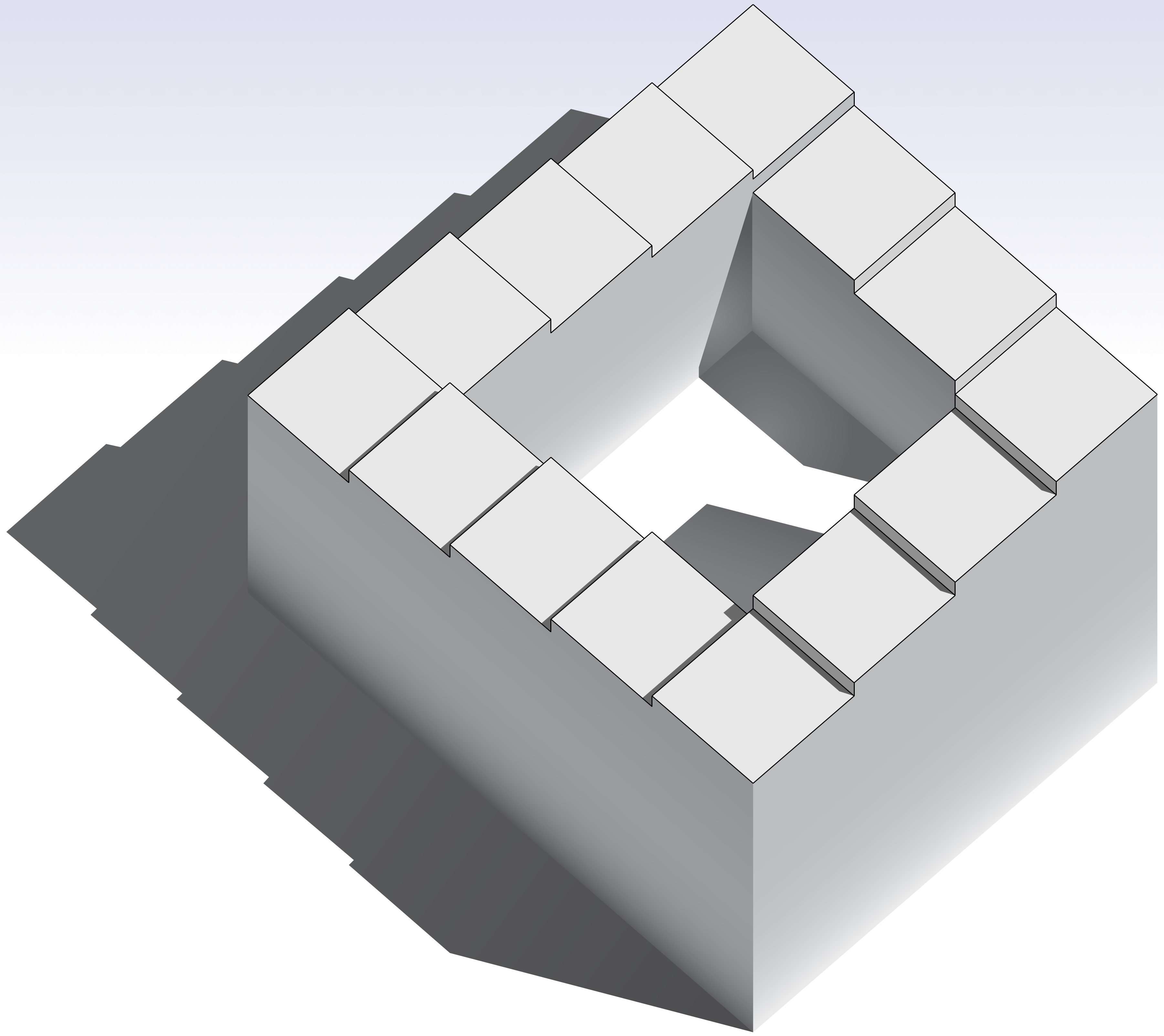
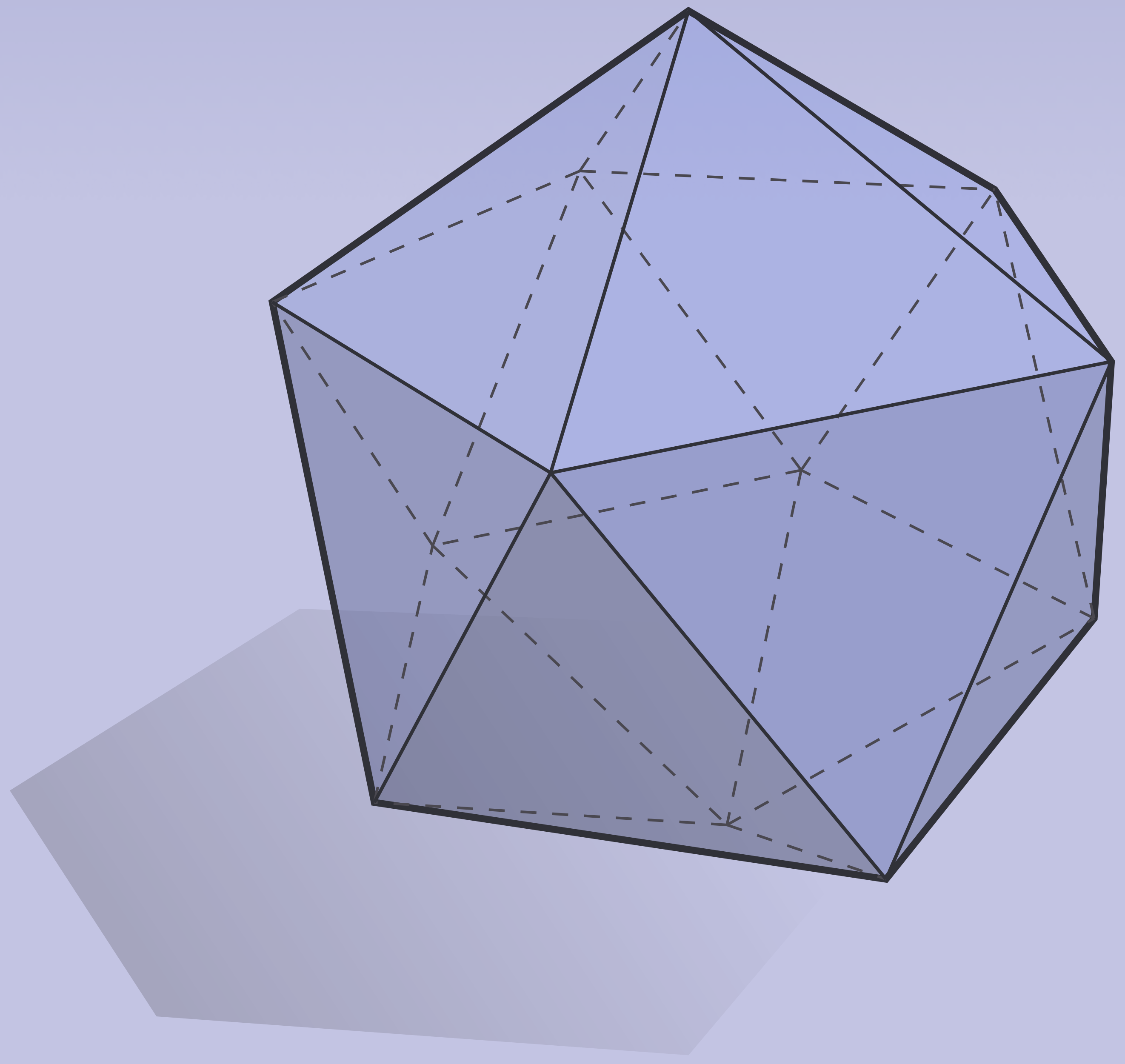


Discrete Differential Geometry: An Applied Introduction

Keenan Crane • CMU 15-458/858

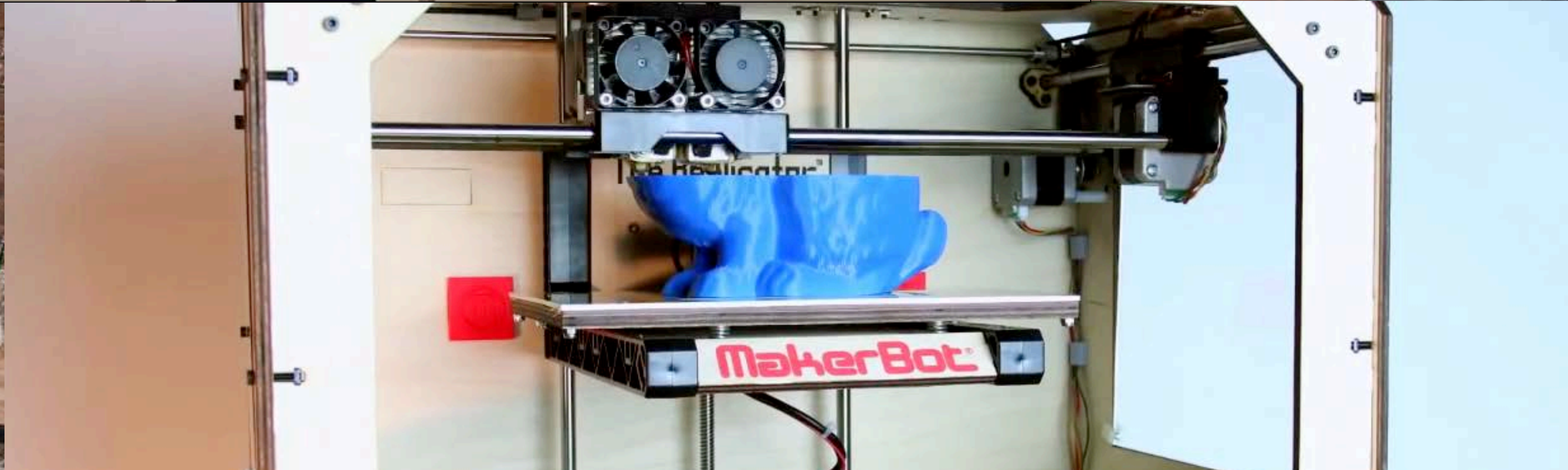
Lecture 1:
Overview



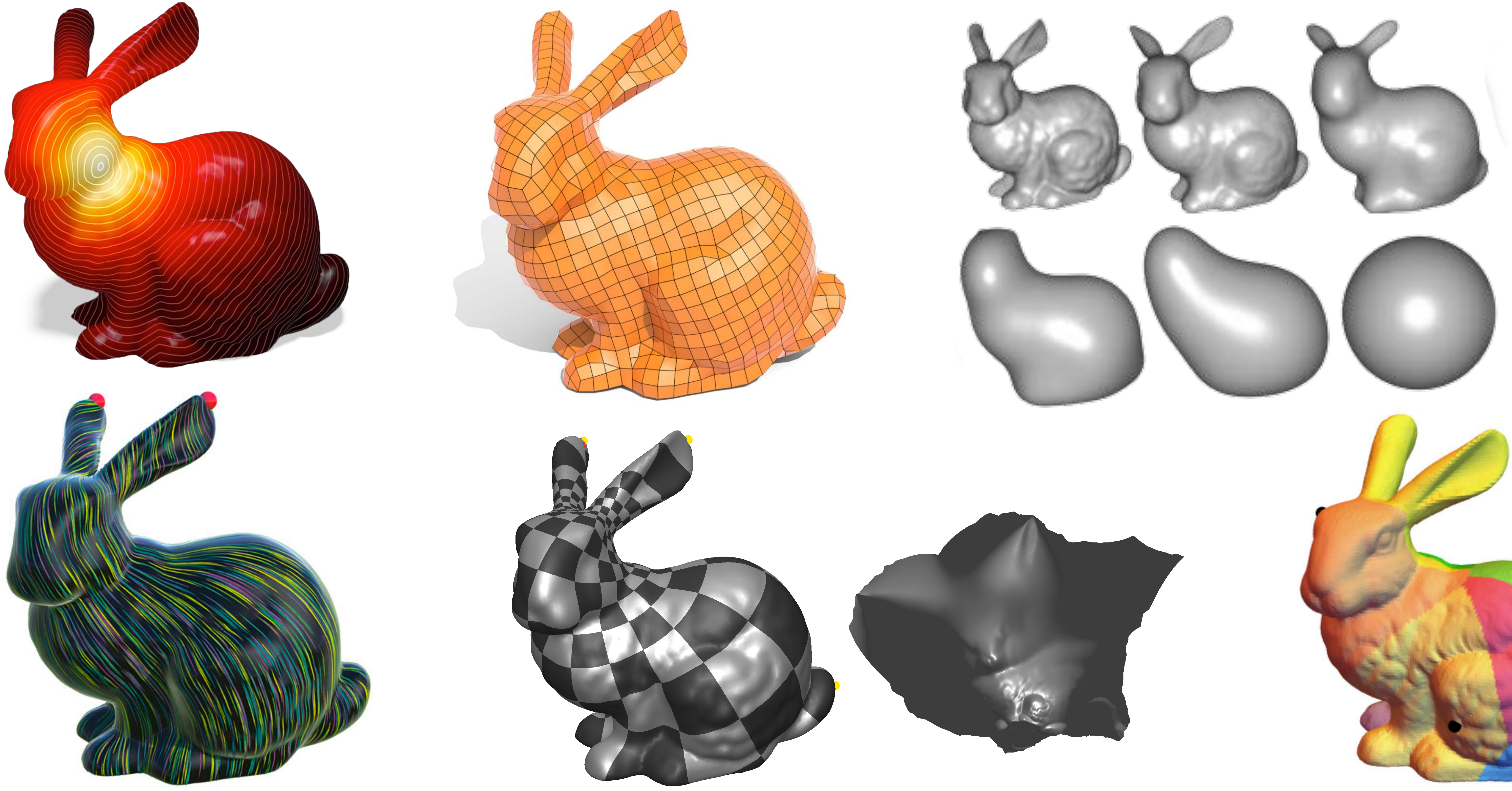
Discrete Differential Geometry: An Applied Introduction

CMU 15-458/858 • Keenan Crane

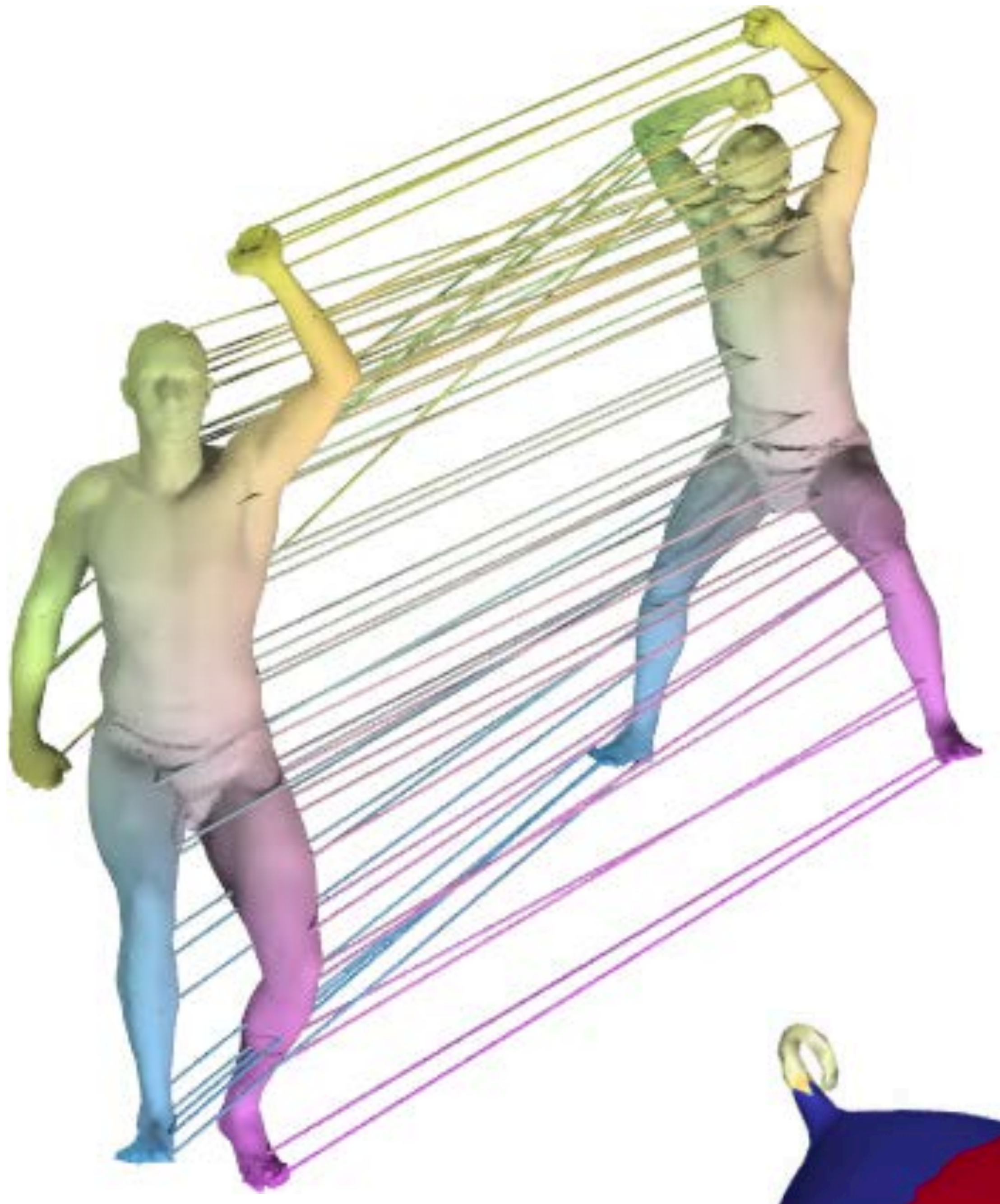
Geometry is Coming…

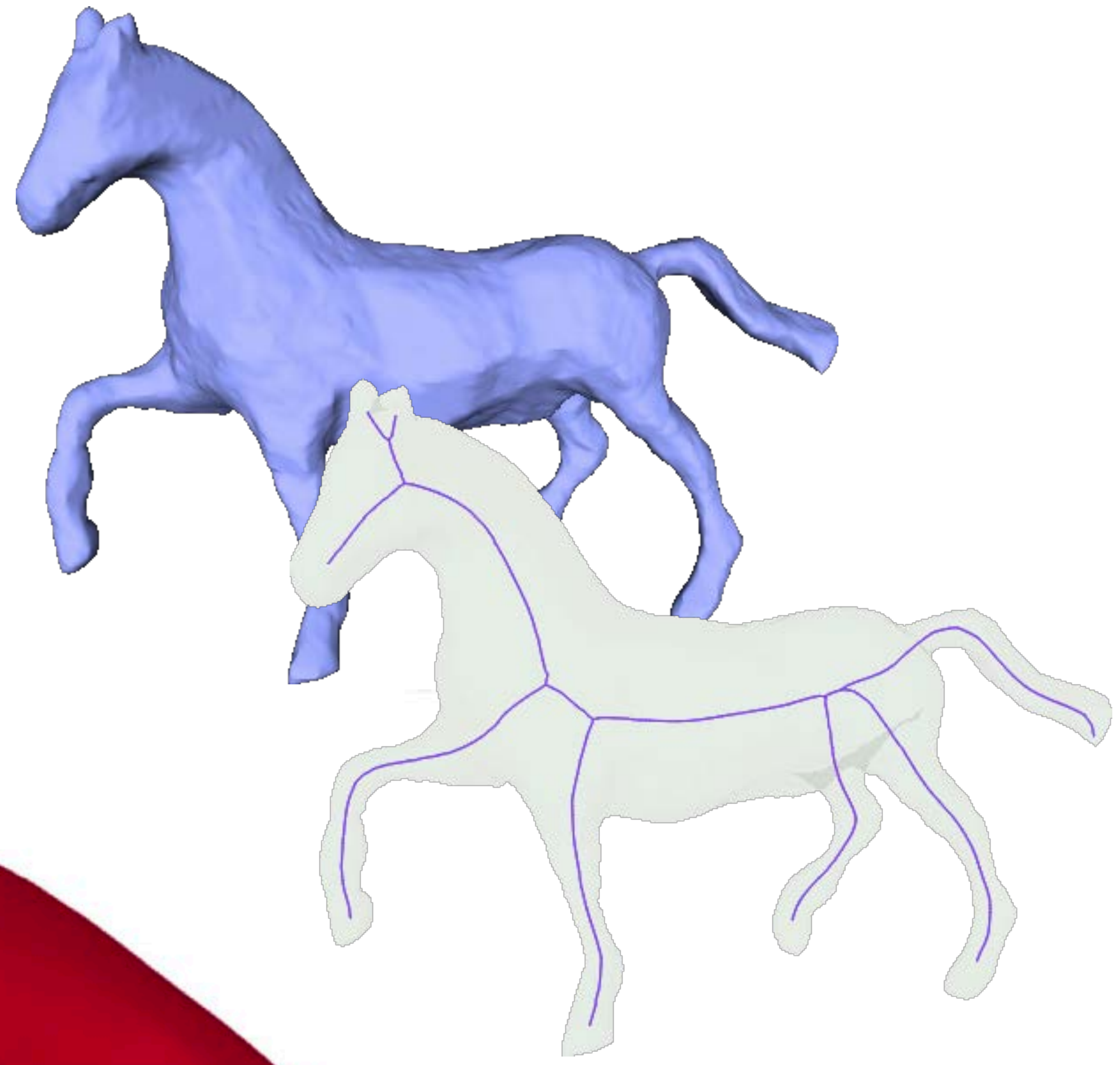


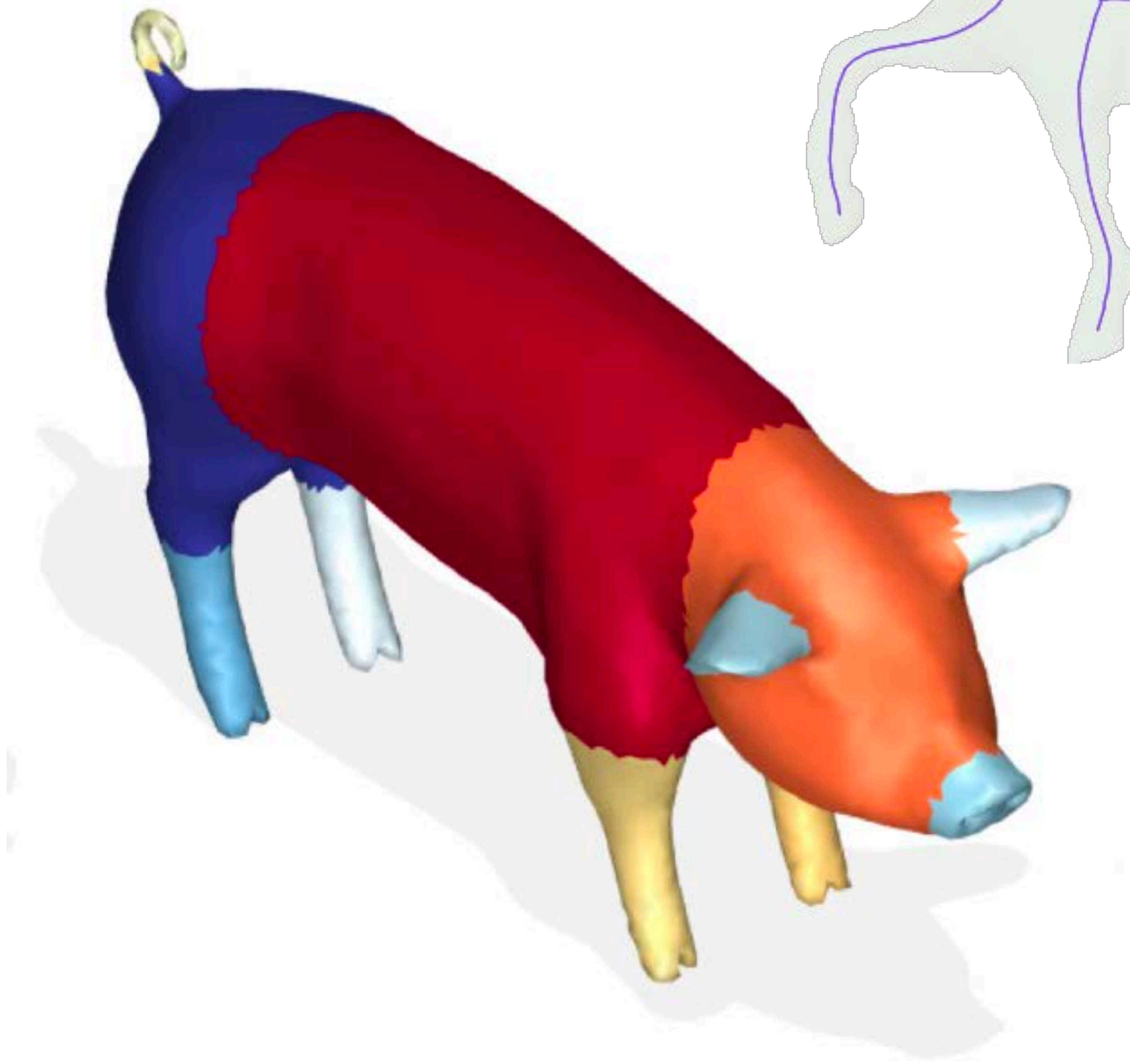
Applications of DDG: Geometry Processing

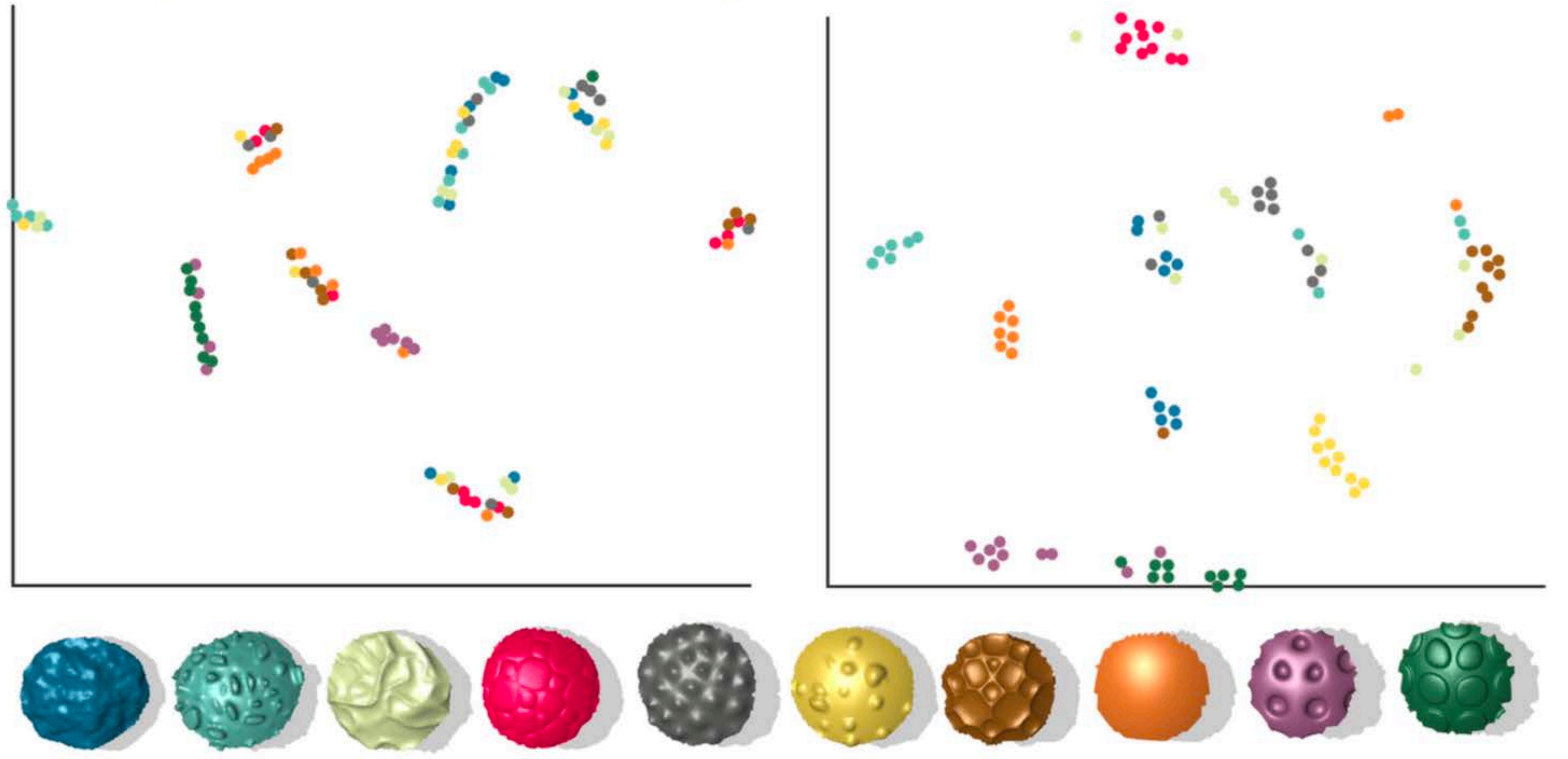


Applications of DDG: Shape Analysis

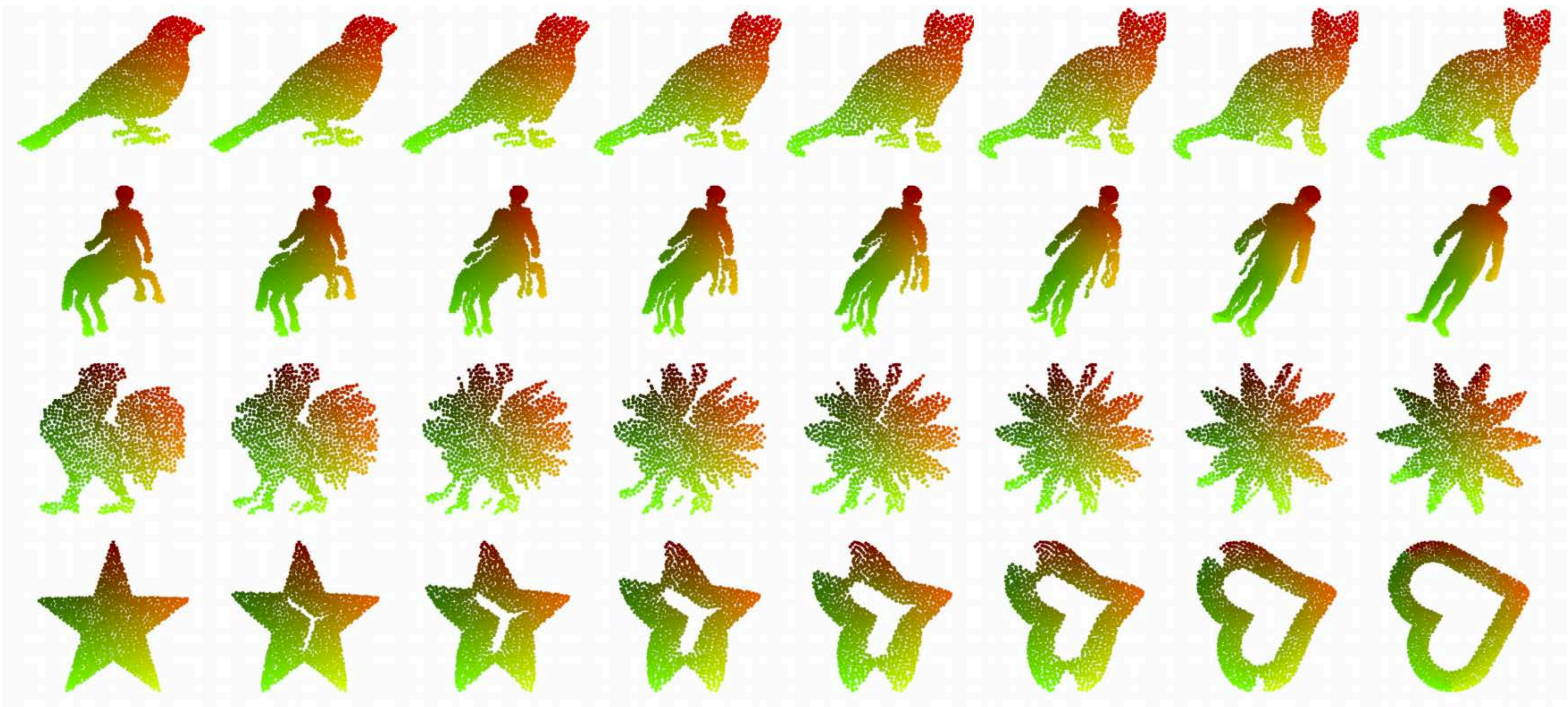


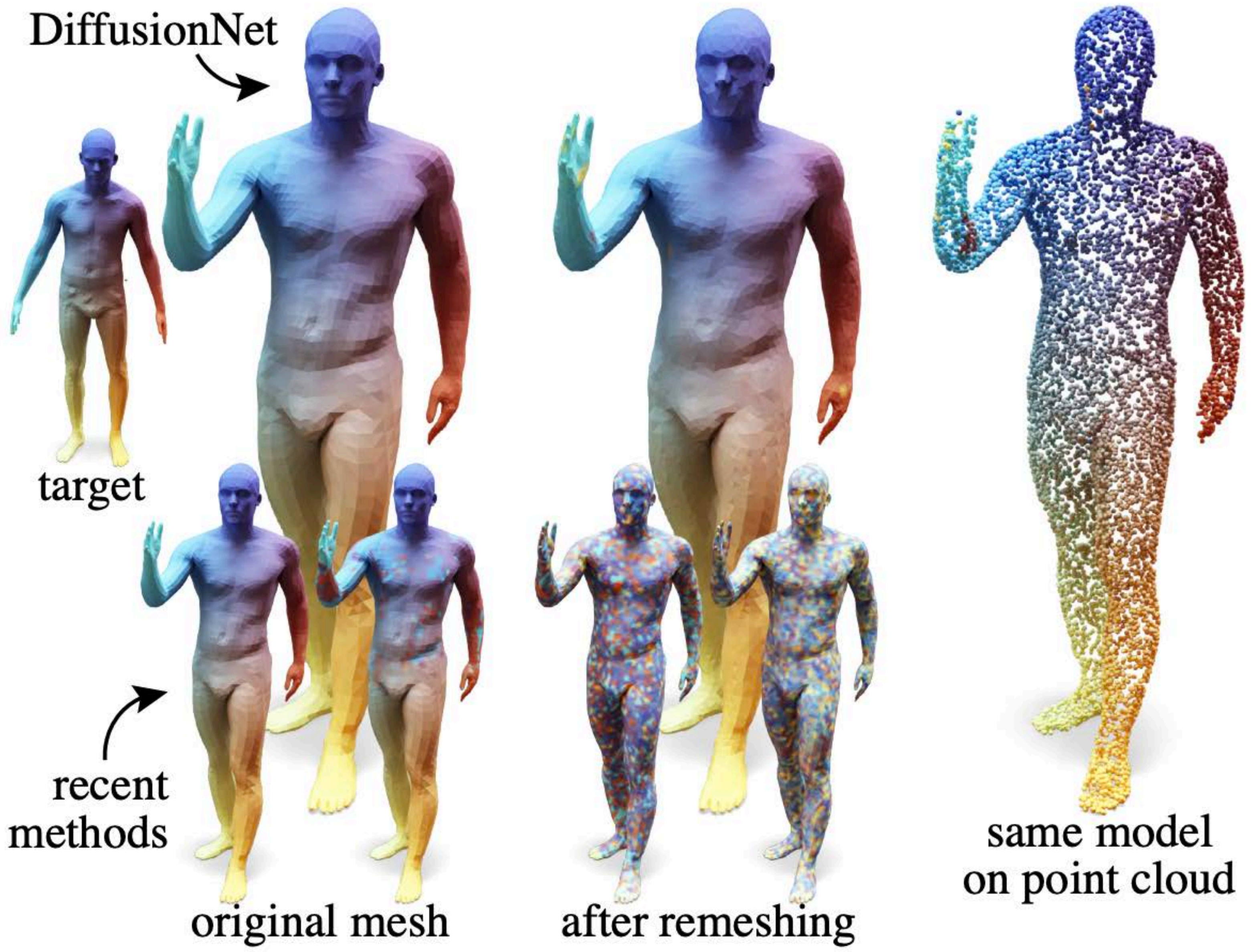


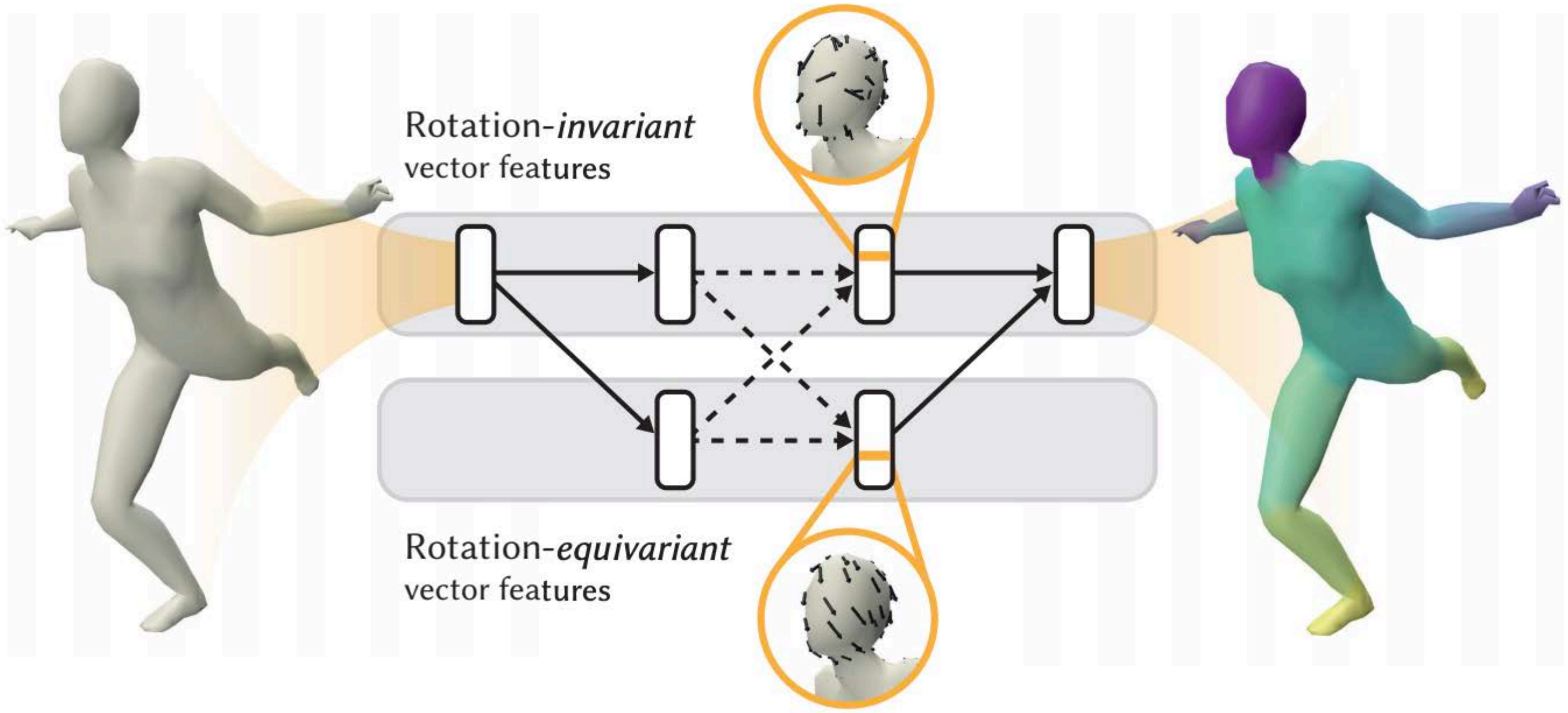


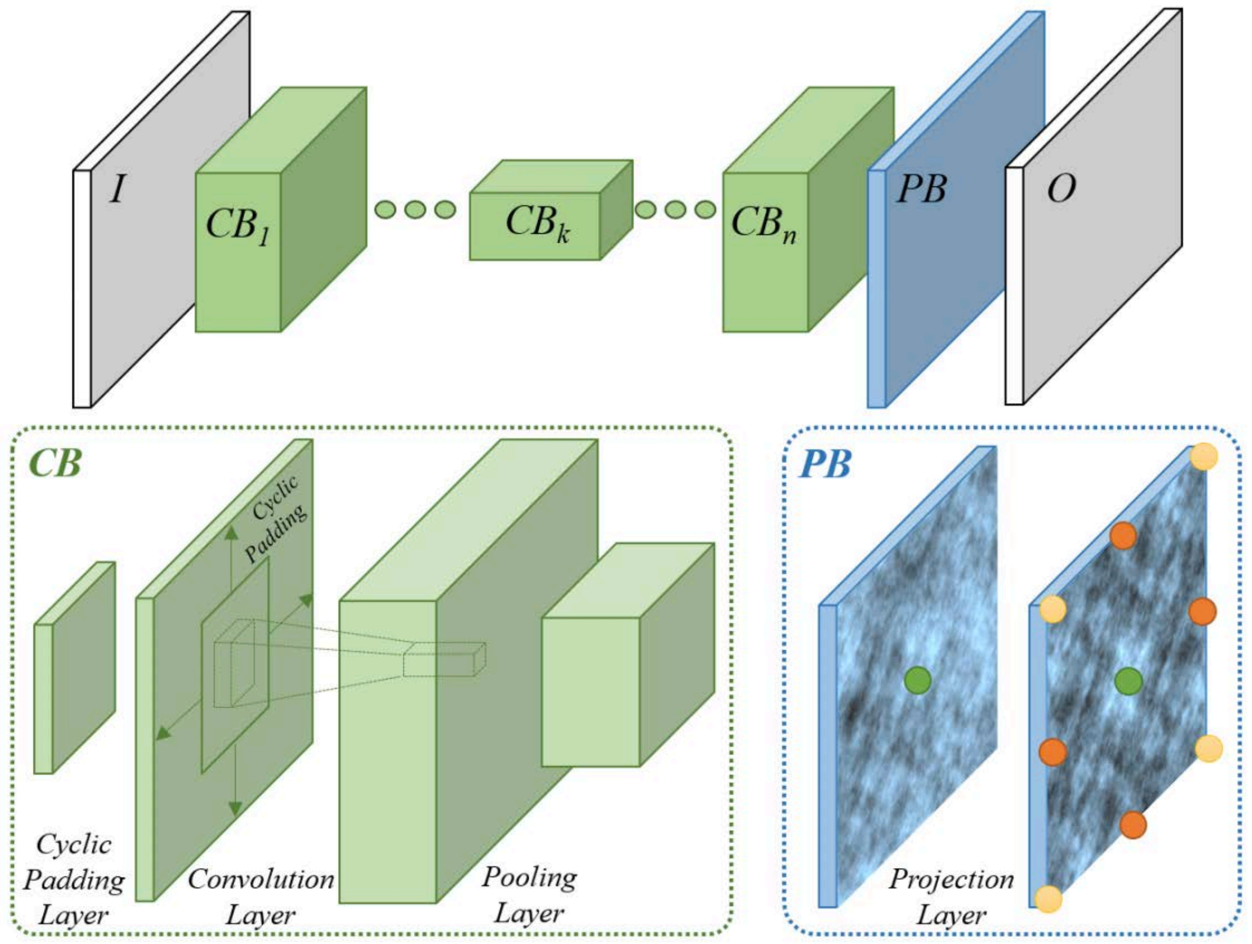


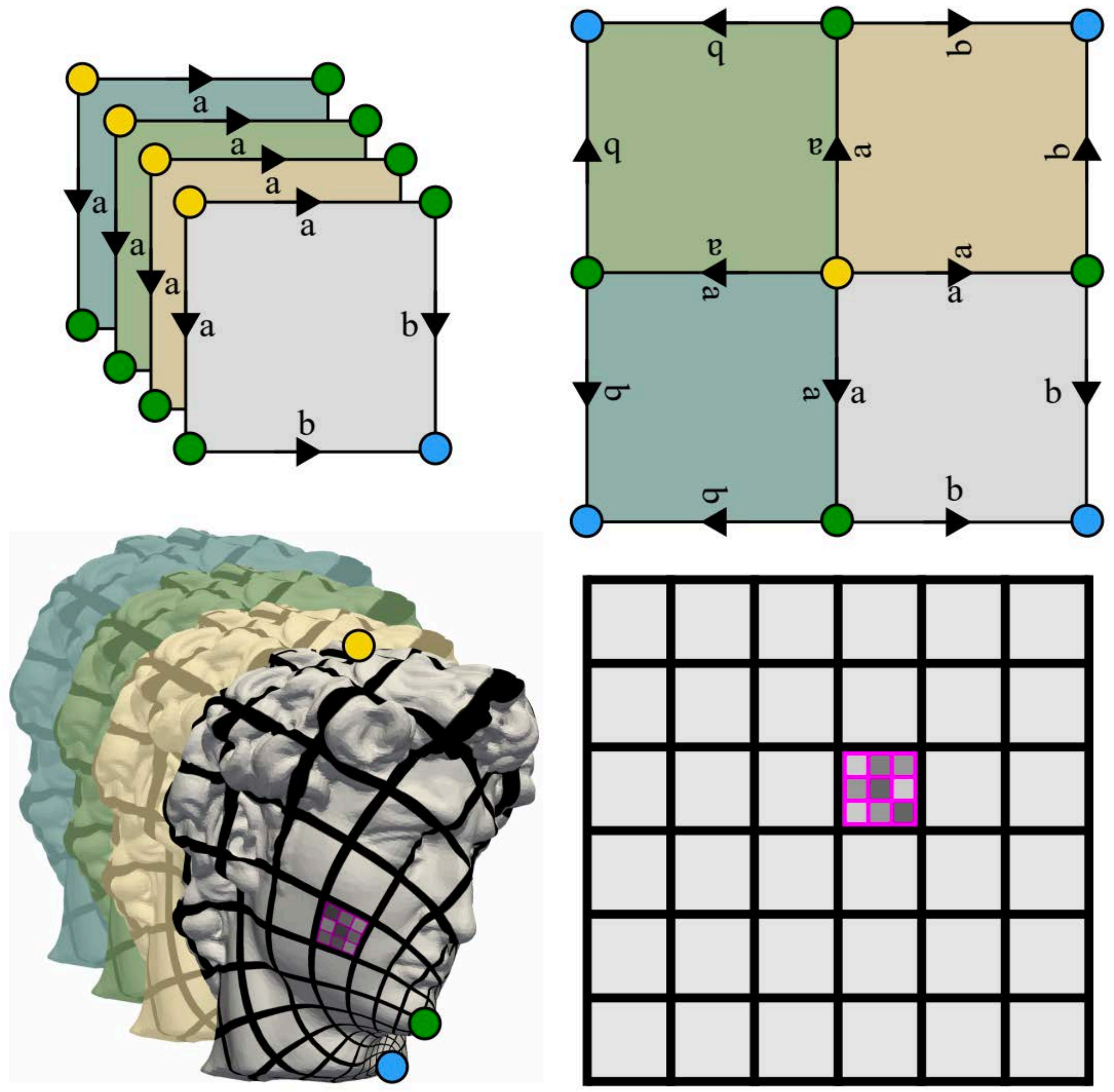
Applications of DDG: Machine Learning



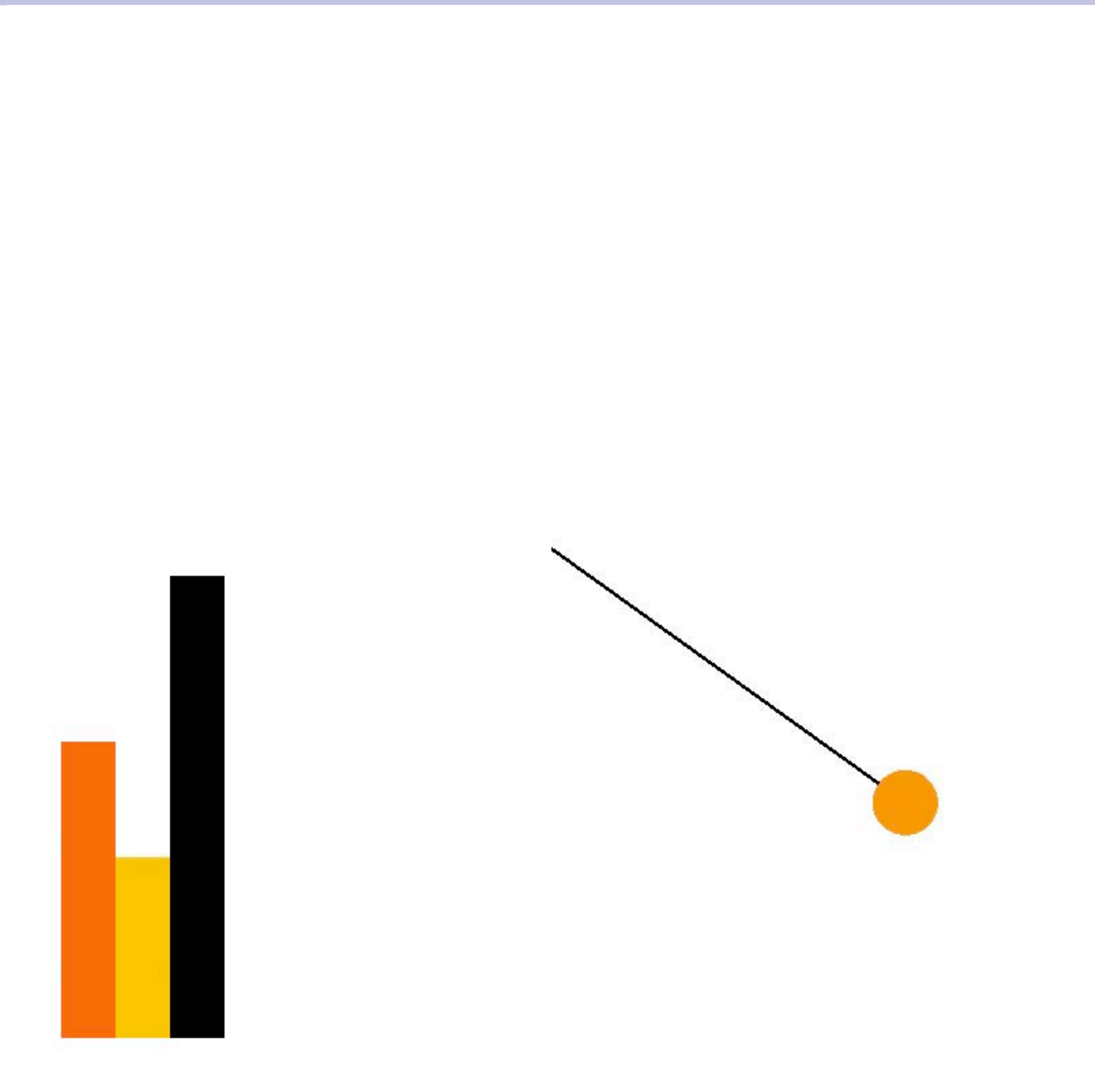


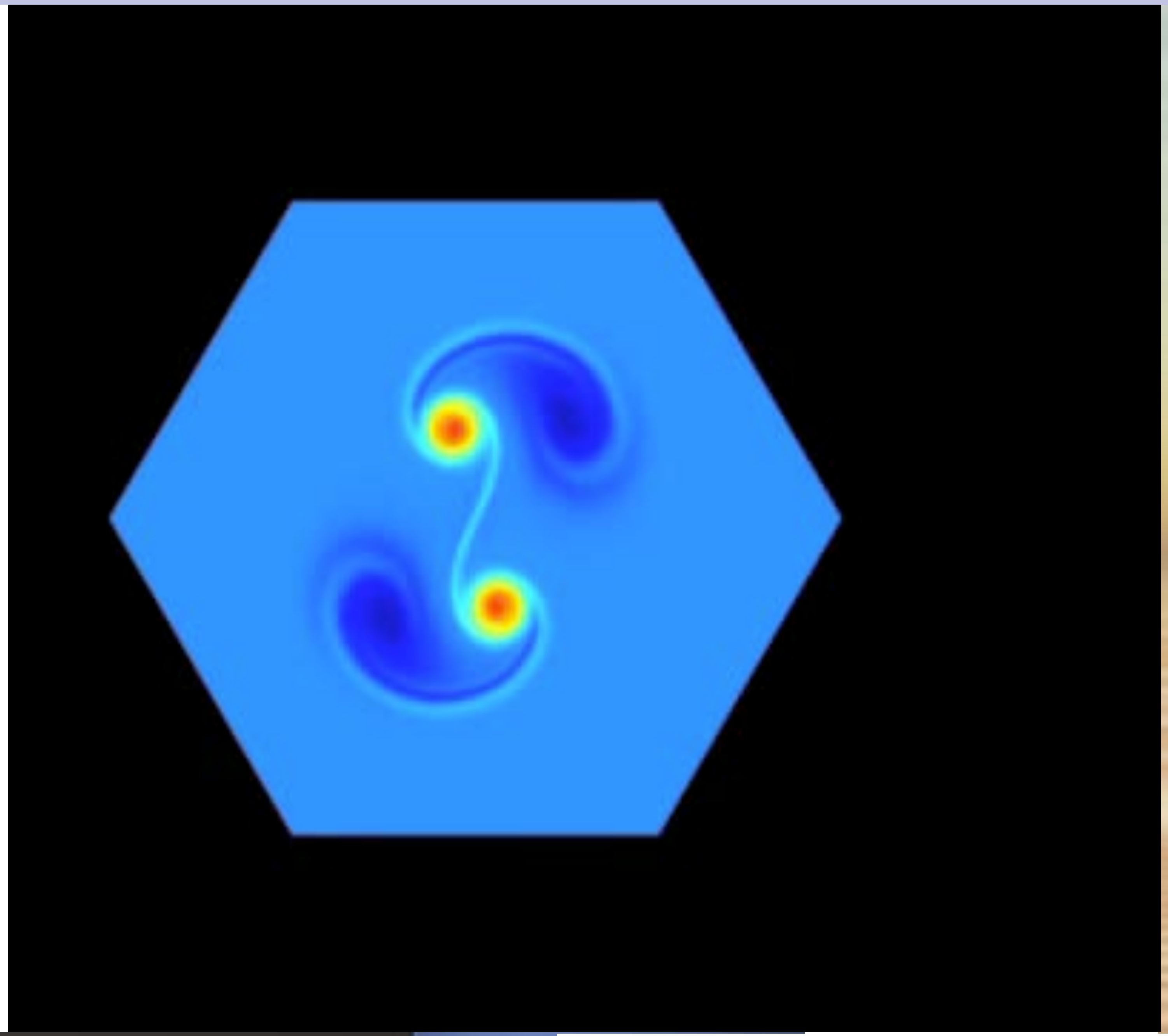


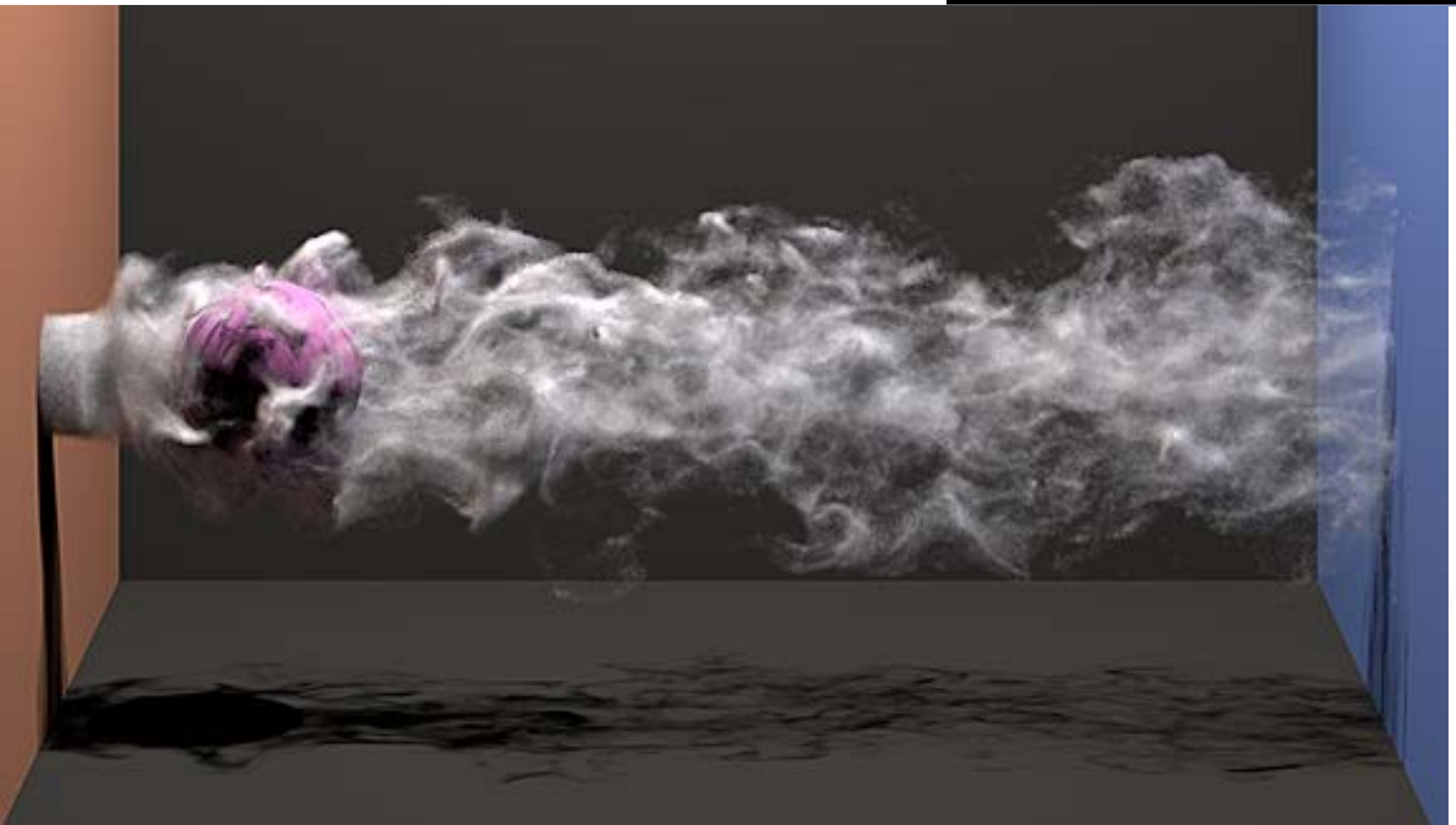


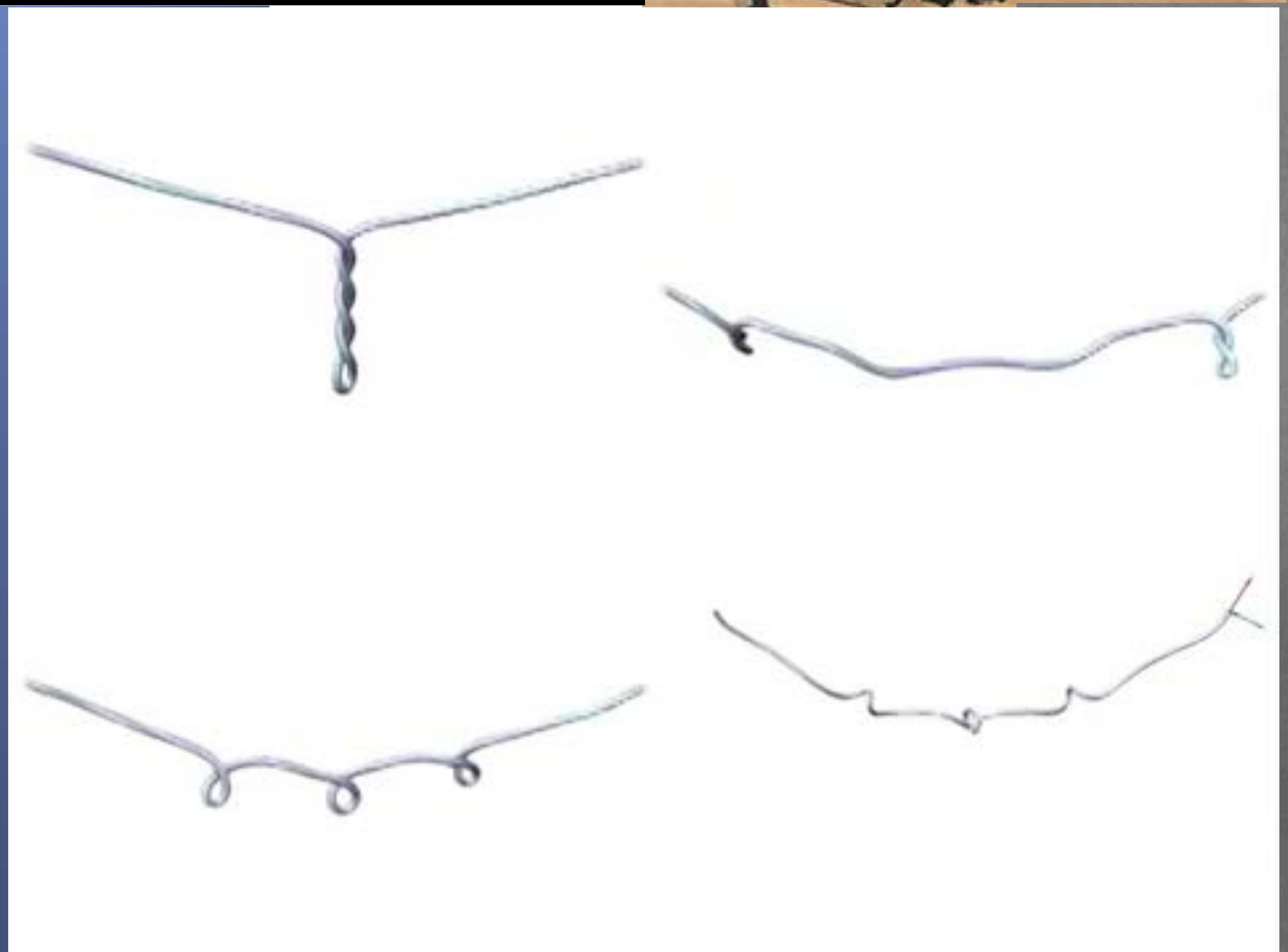


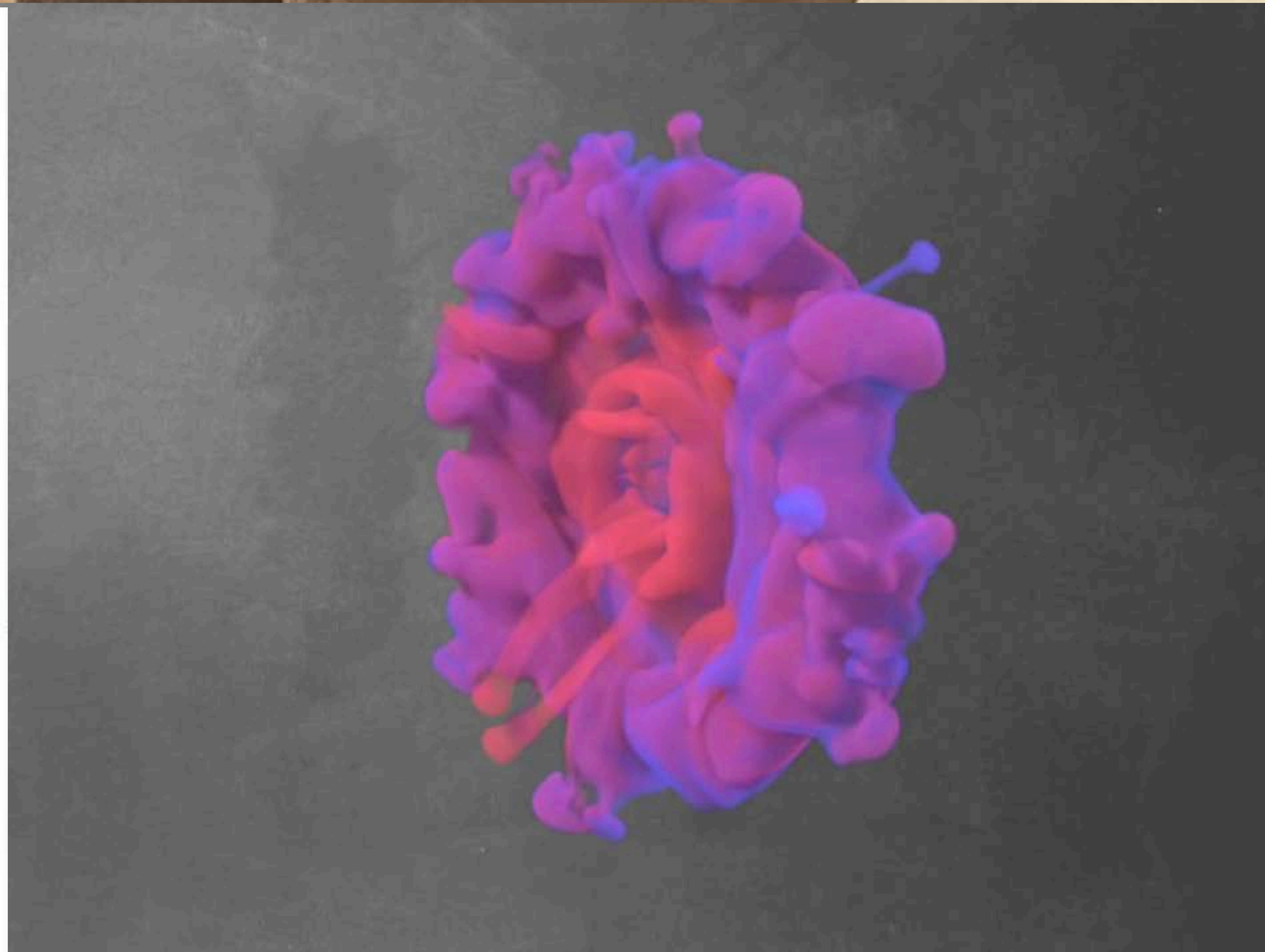
Applications of DDG: Numerical Simulation





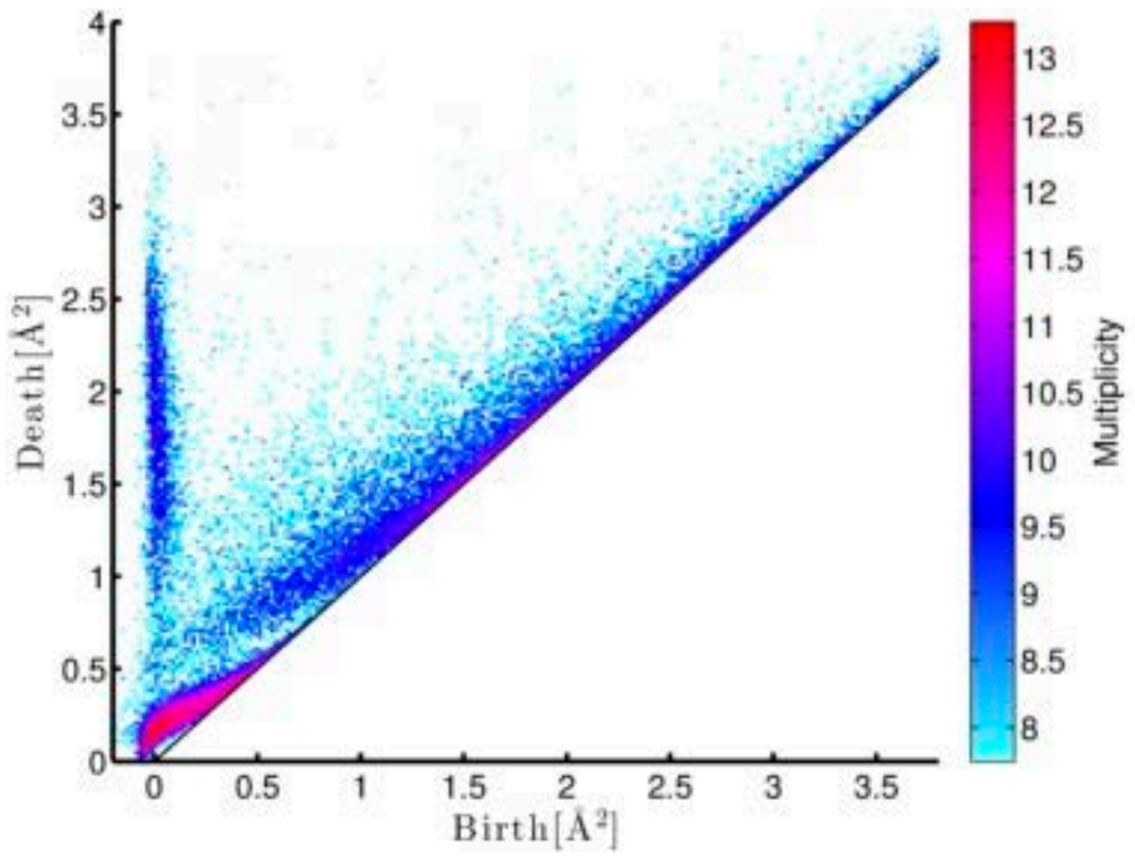


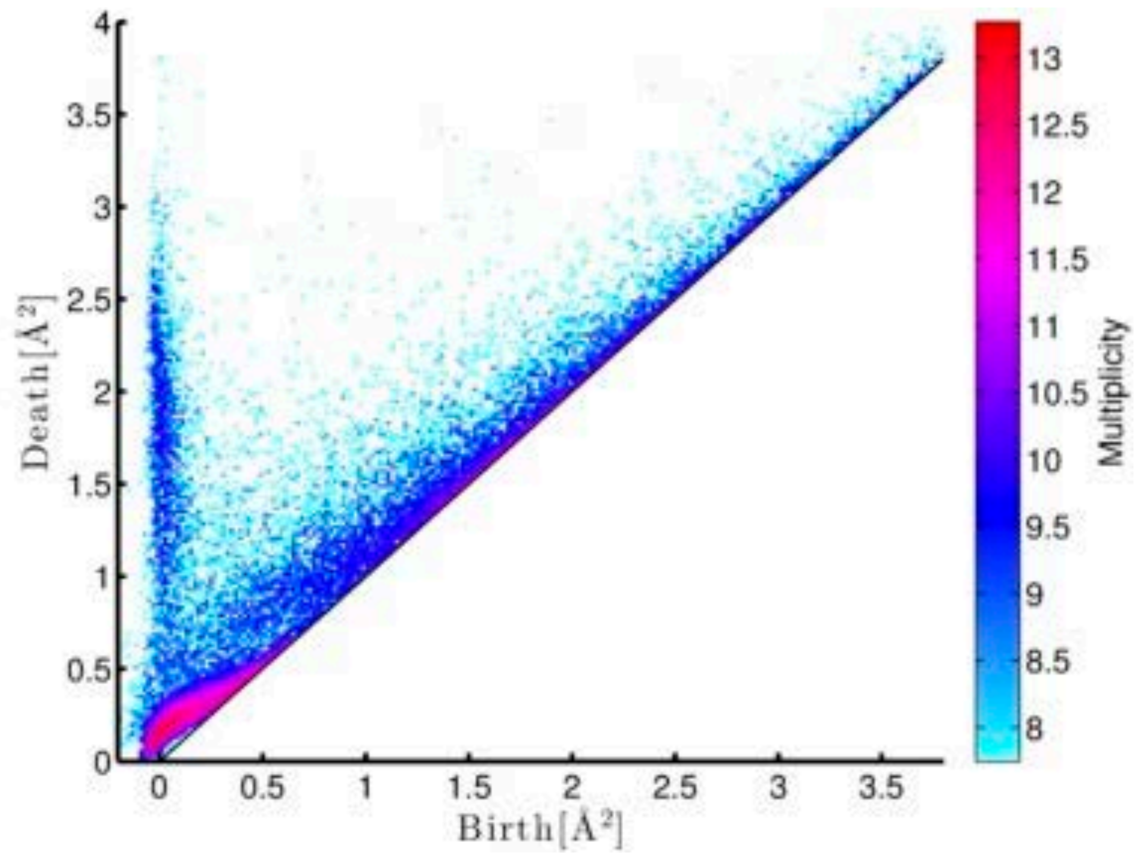


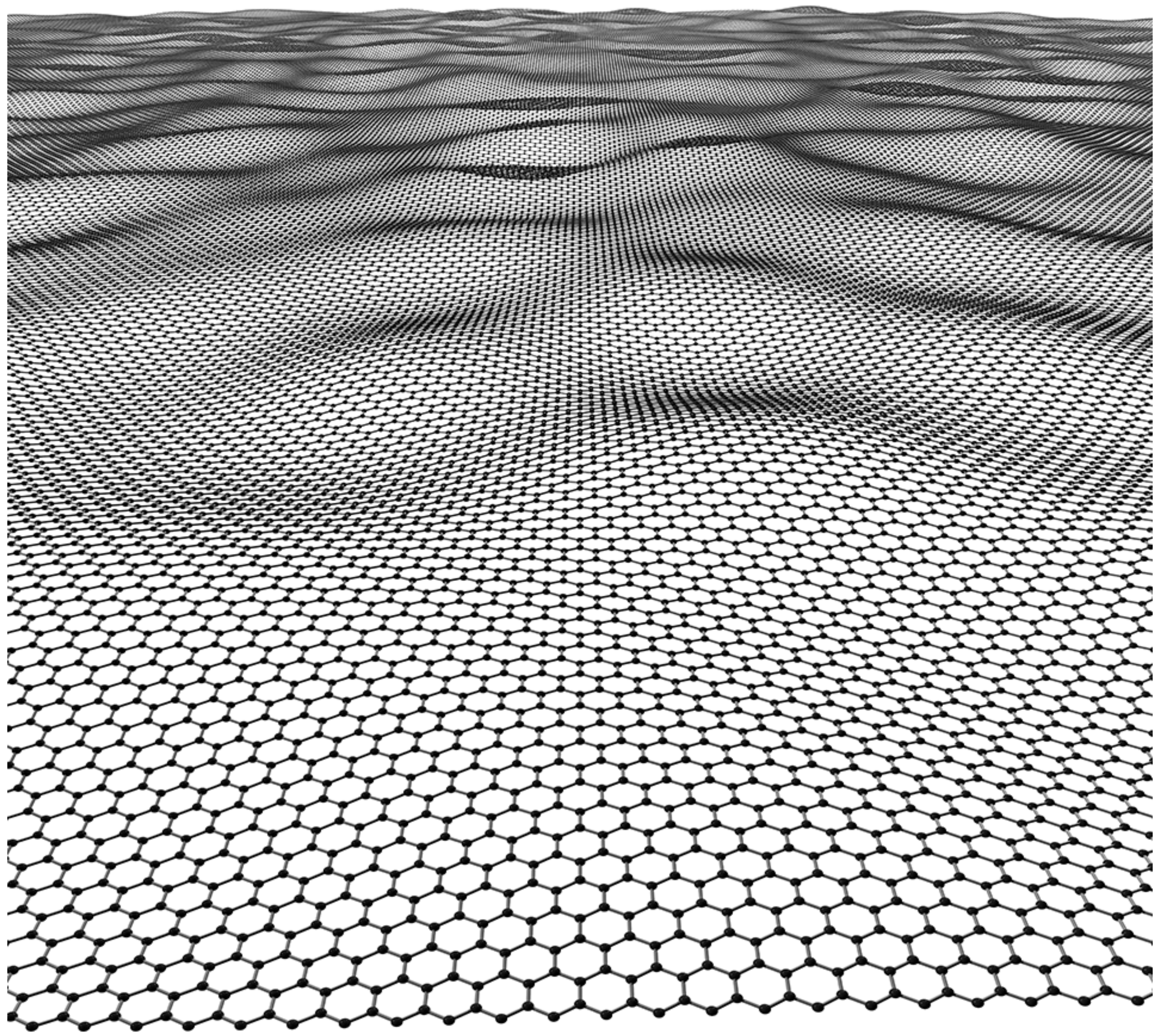


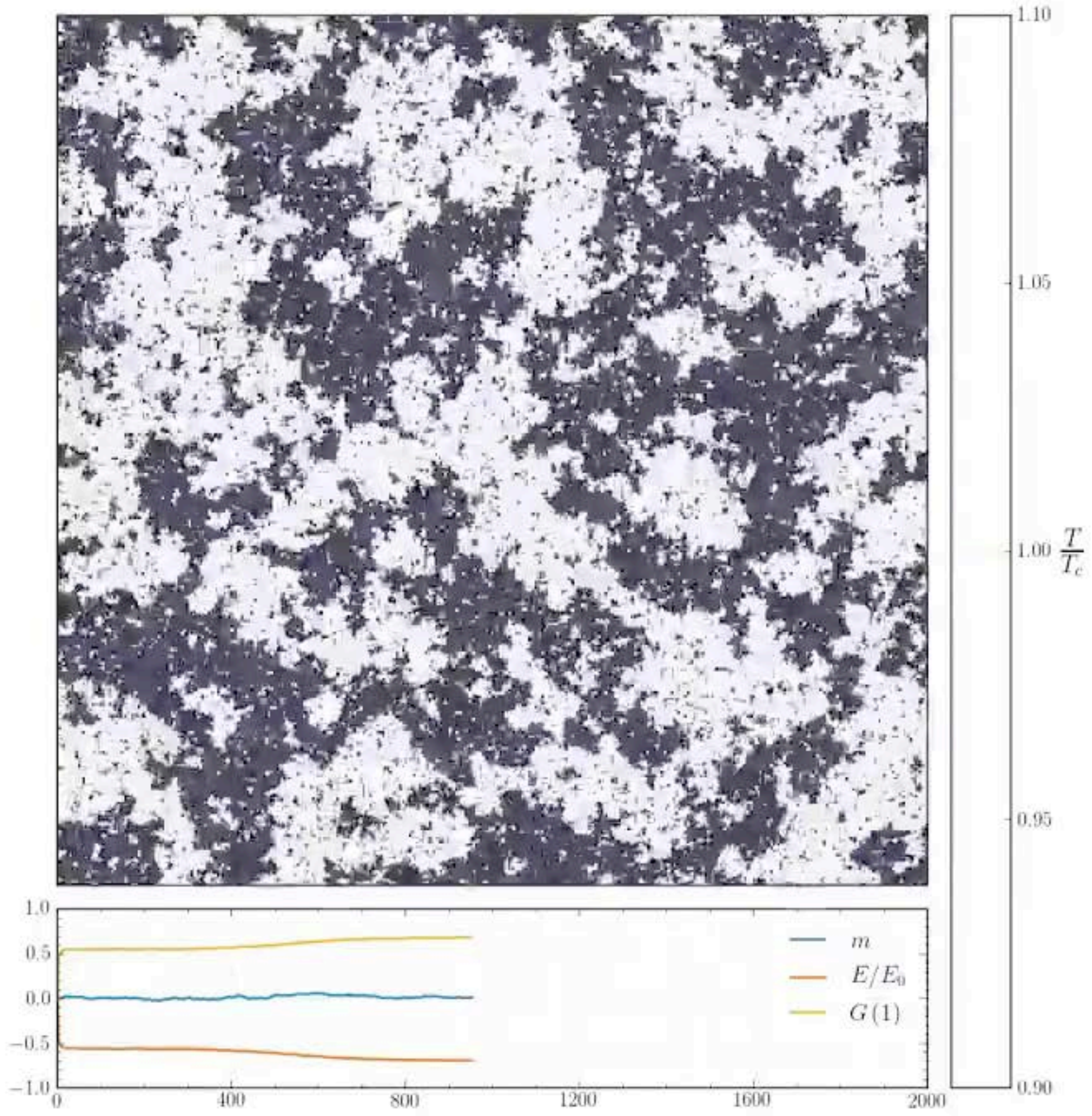
Applications of DDG: Architecture & Design

Applications of DDG: Discrete Models of Nature



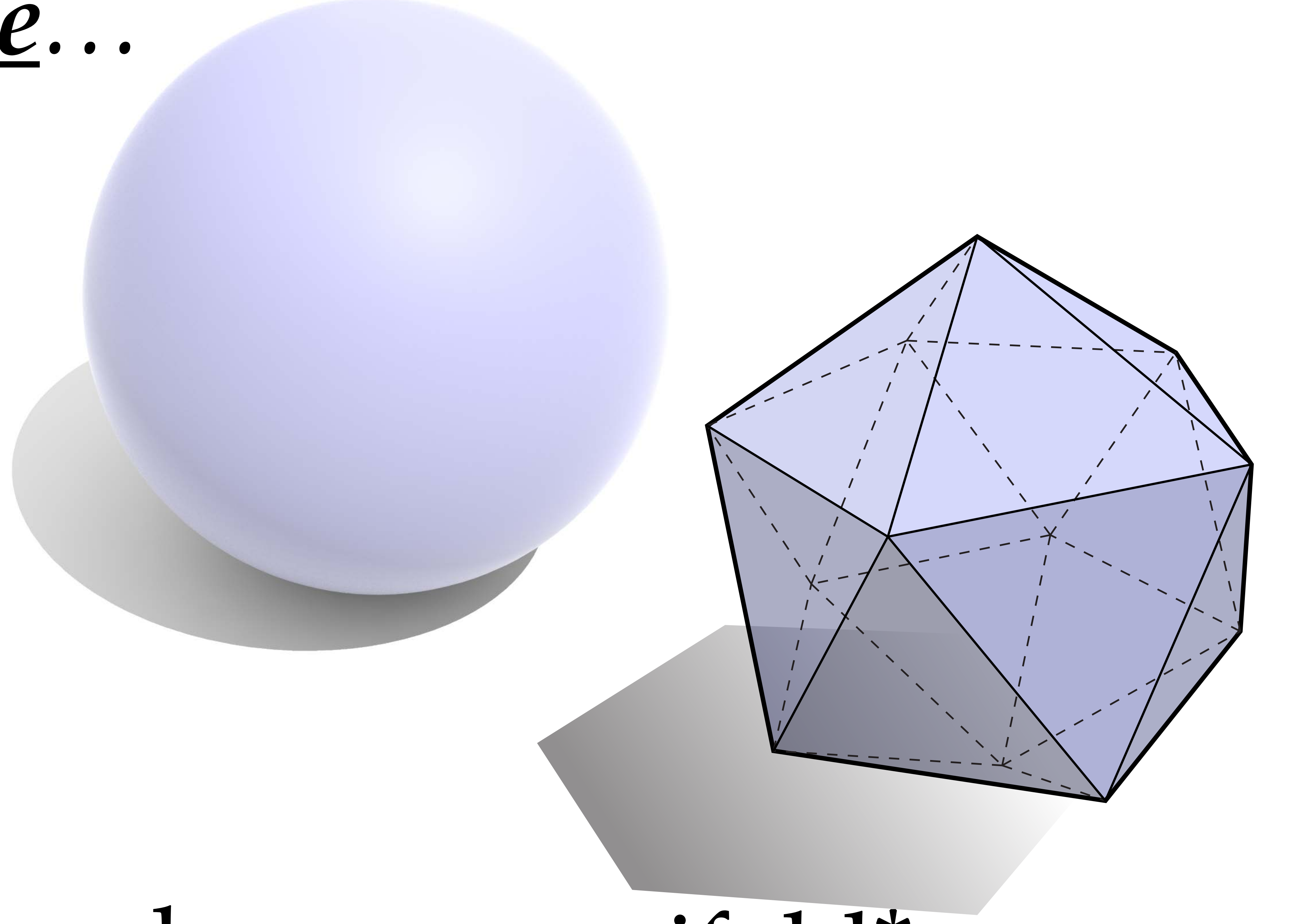






What Will We Learn in This Class?

- First and foremost: ***how to think about shape***…
  - …mathematically (differential geometry)
  - …computationally (geometry processing)
  - **Central Theme:** *link these two perspectives*
- Why? Shape is everywhere!
  - Every time you have a constraint $f(x) = 0$, you have a manifold*
  - *computational biology, industrial design, computer vision, machine learning, architecture, computational mechanics, fashion, medical imaging*…



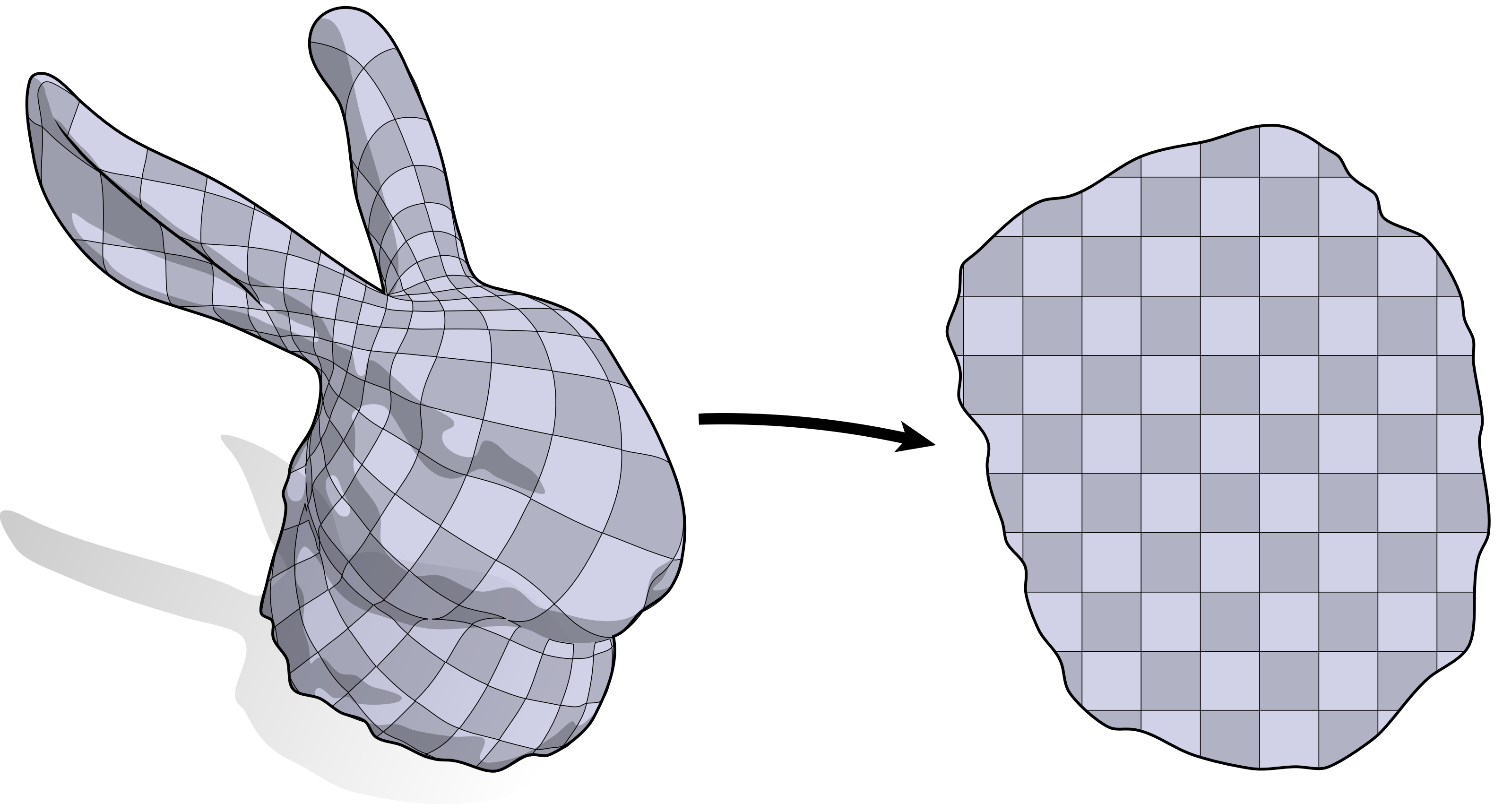
*Must be sufficiently regular, *etc.*

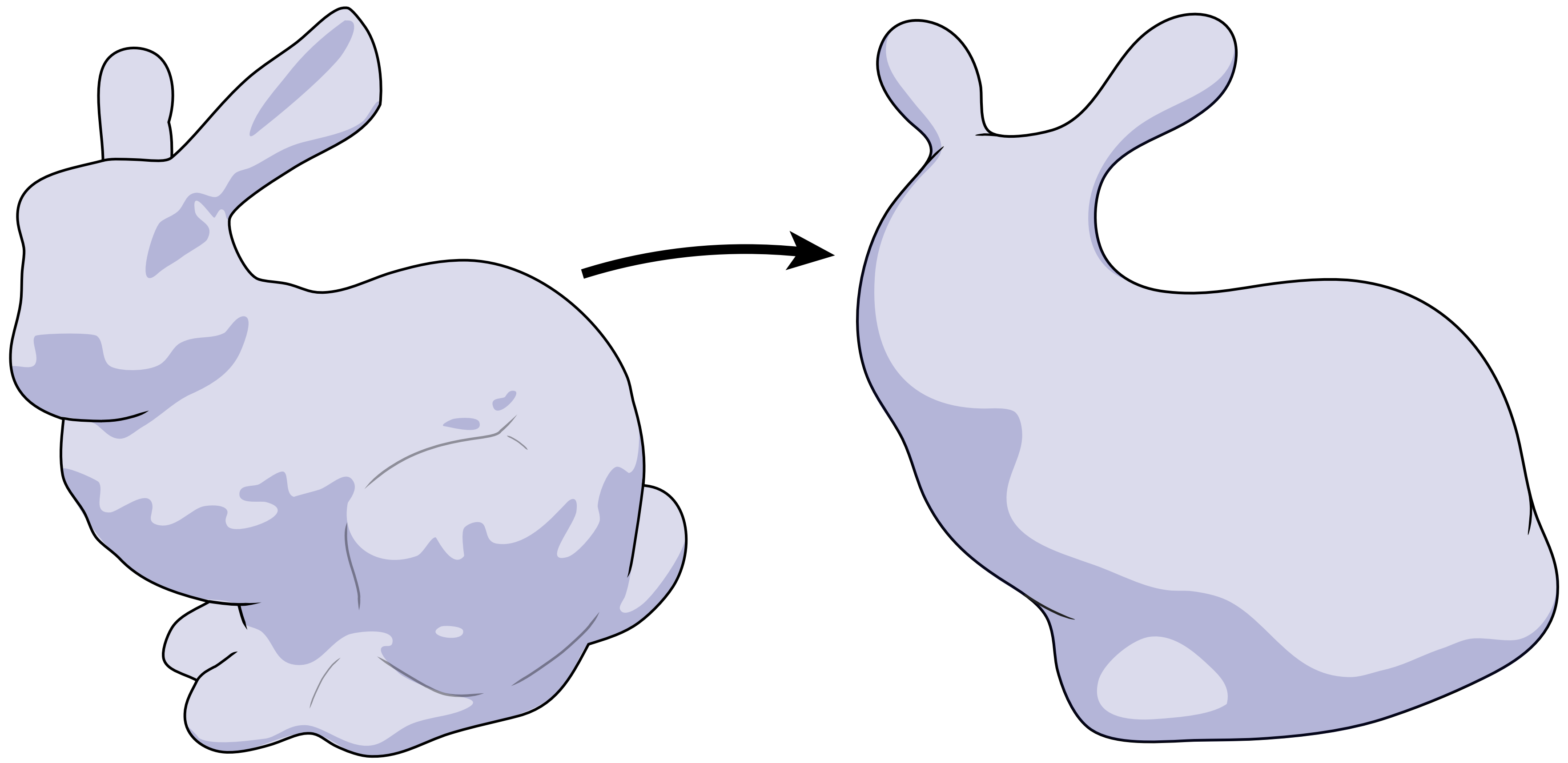
What <u>won't</u> we learn in this class?

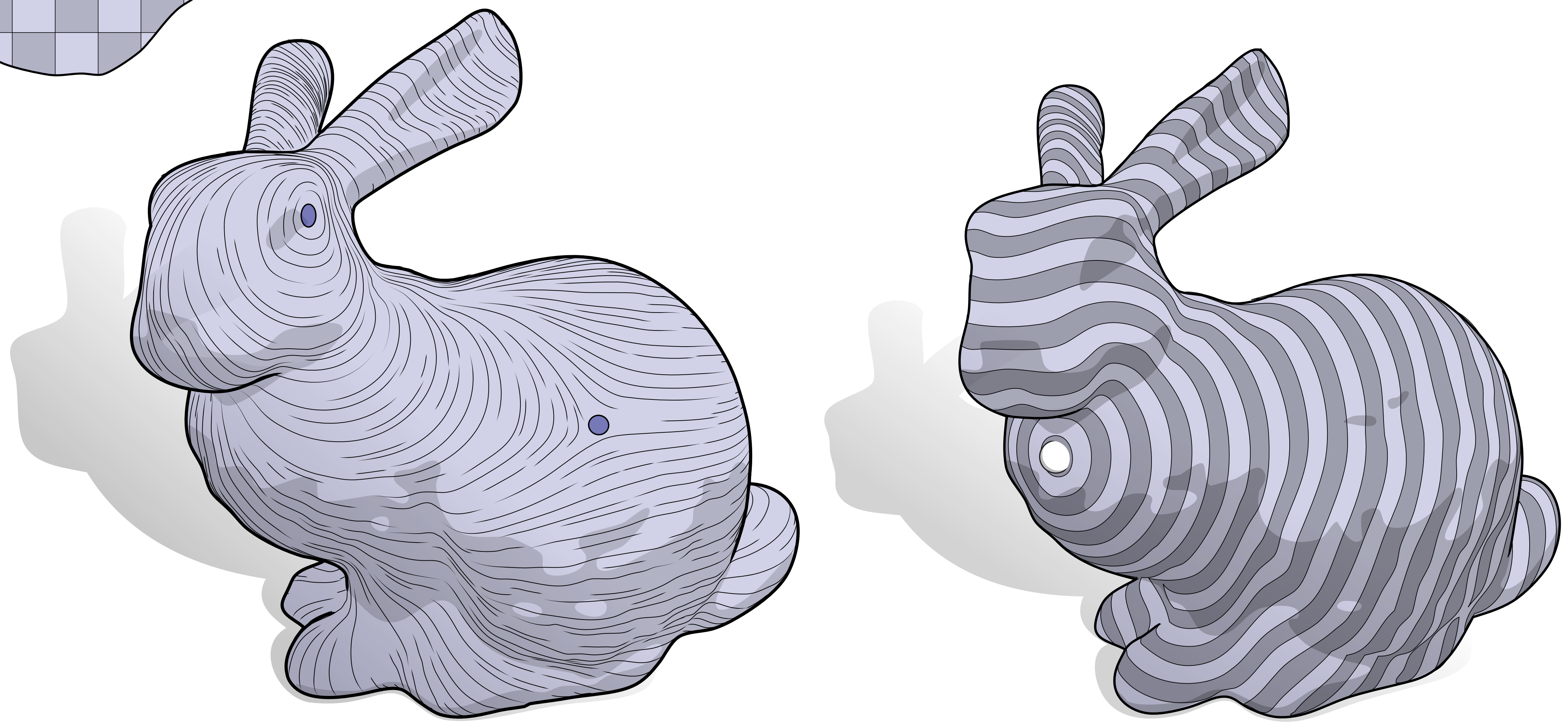
- *We won't learn everything!*
  - Many viewpoints on differential geometry we *don't* have time to cover
  - Huge number of algorithms we *won't* be able to cover
- Depending on your goals & interests the specific set of algorithms we cover this semester <u>may not be directly useful!</u>
  - *e.g.*, you may care about point clouds and computer vision; we will focus mostly polygons and applications in geometry processing
- Recall main goal: ***learn how to think about shape!***
  - Fundamental knowledge you gain here *will* translate to other contexts

Assignments

- **Derive** geometric algorithms from first principles (pen-and-paper)
- **Implement** geometric algorithms (coding)
  - Discrete surfaces
  - Exterior calculus
  - Curvature
  - Smoothing
  - Parameterization
  - Distance computation
  - Direction Field Design

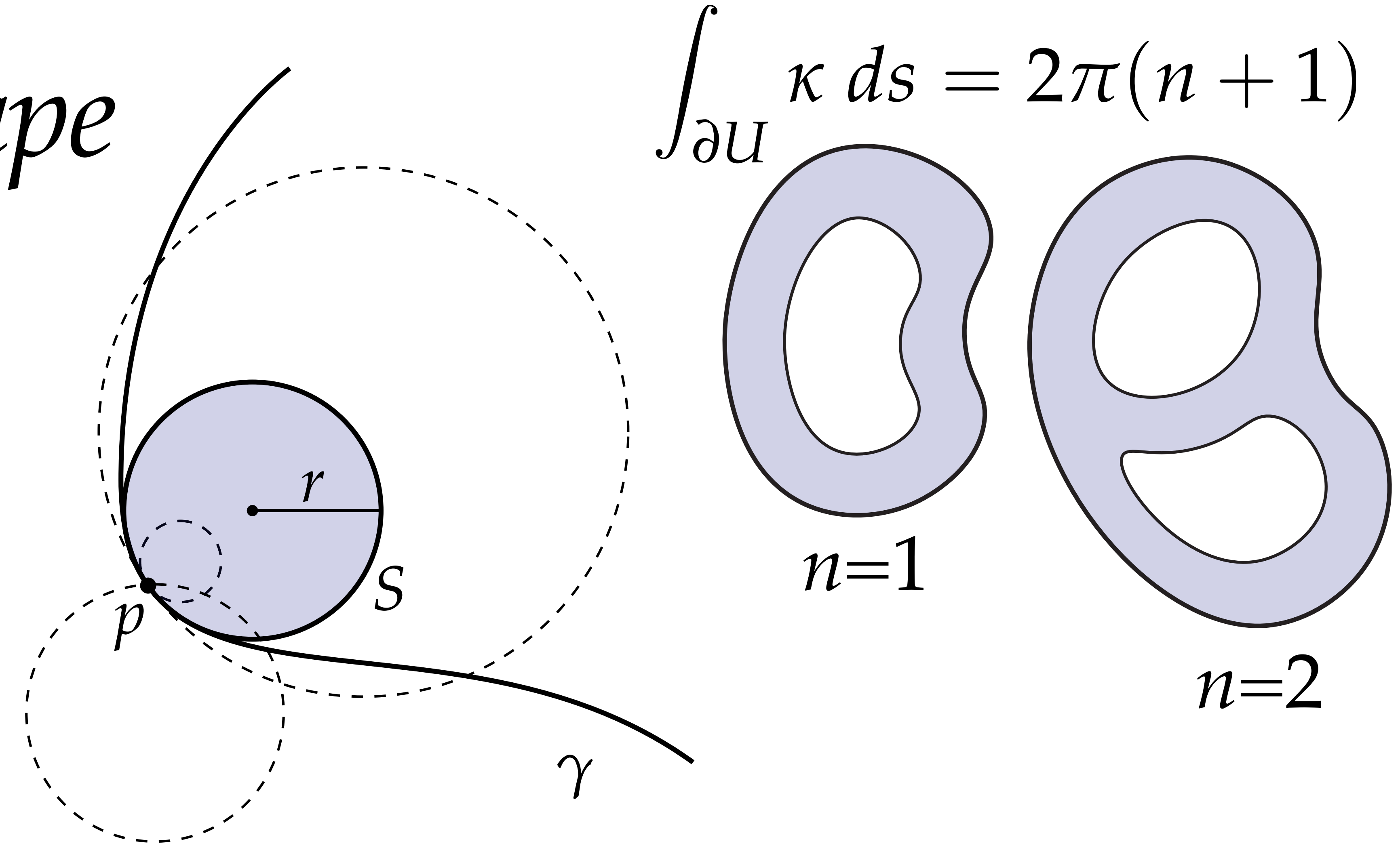


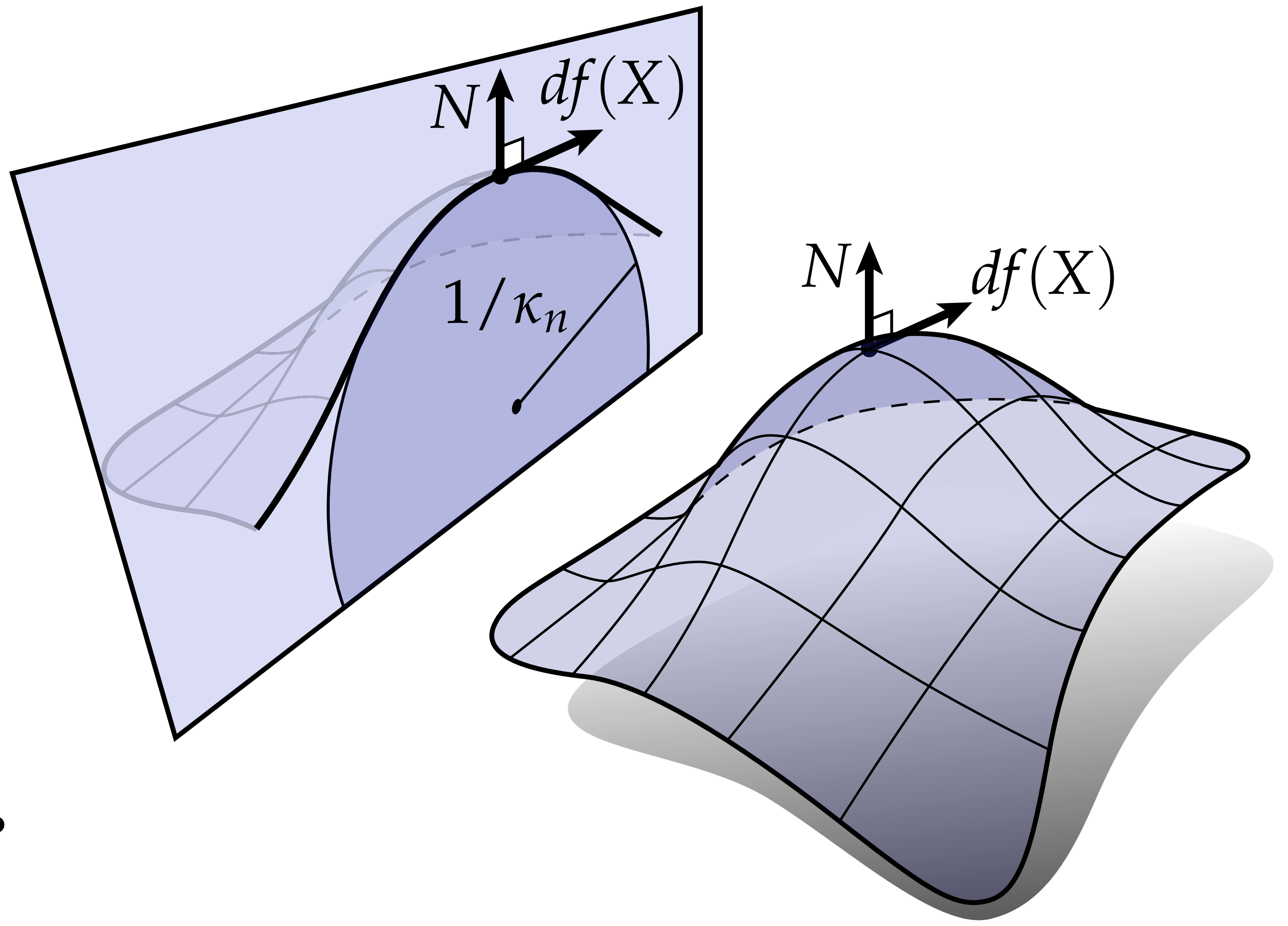




What is Differential Geometry?

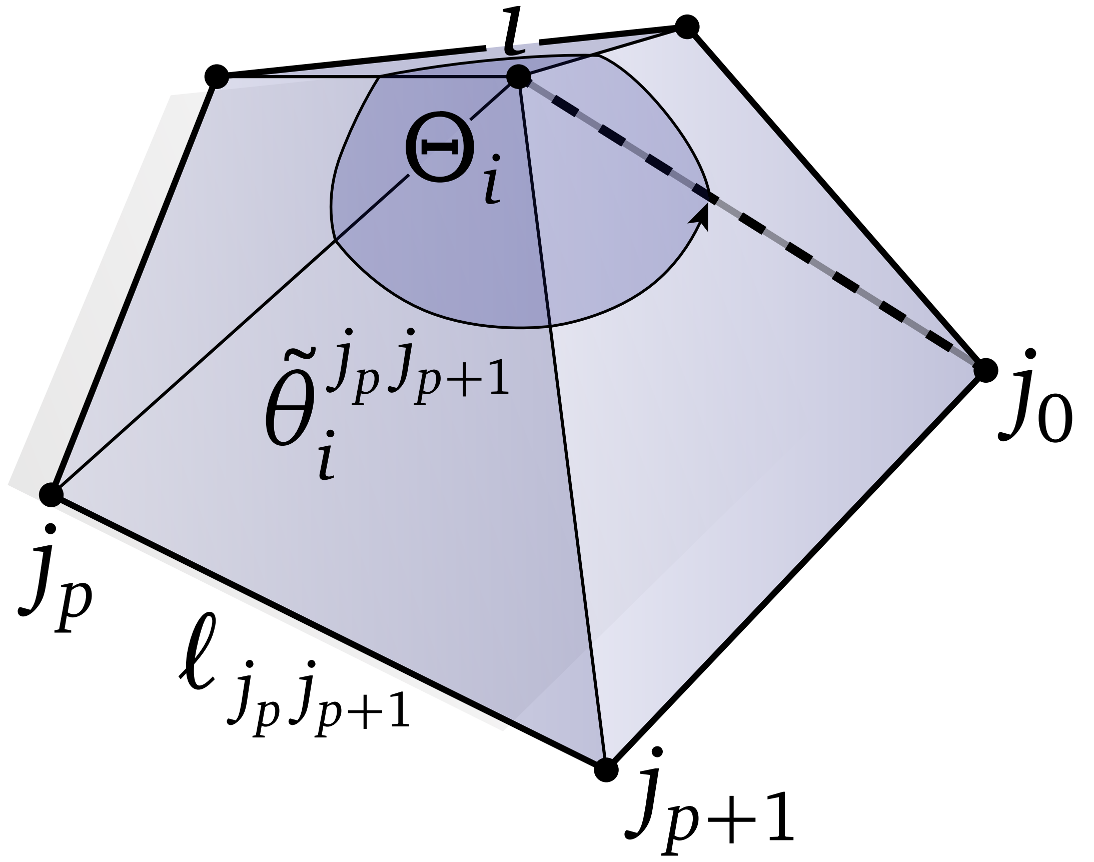
- **Language** for talking about *local properties of shape*
  - How fast are we traveling along a curve?
  - How much does the surface bend at a point?
  - *etc.*
- …and their connection to *global properties of shape*
  - So-called “local-global” relationships.
- Modern language of geometry, physics, statistics, …
- Profound impact on scientific & industrial development in 20th century



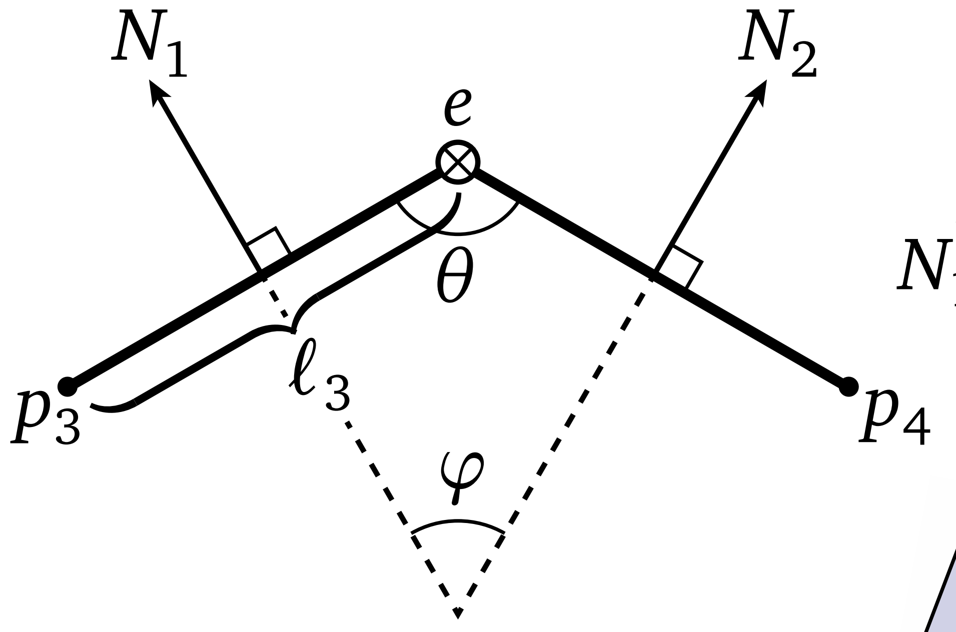


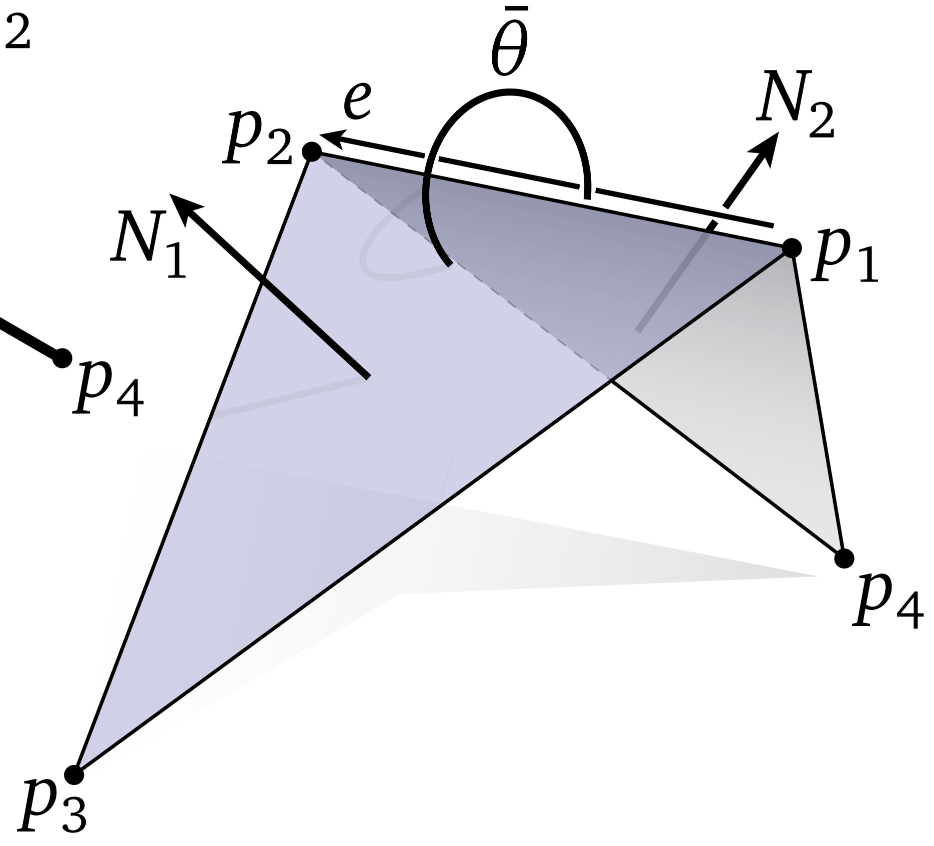
What is ***Discrete*** *Differential Geometry?*

- Also a language describing local properties of shape
  - *Infinity no longer allowed!*
  - No longer talk about derivatives, infinitesimals…
  - Everything expressed in terms of lengths, angles…



- Surprisingly little is lost!
  - Faithfully captures many fundamental ideas

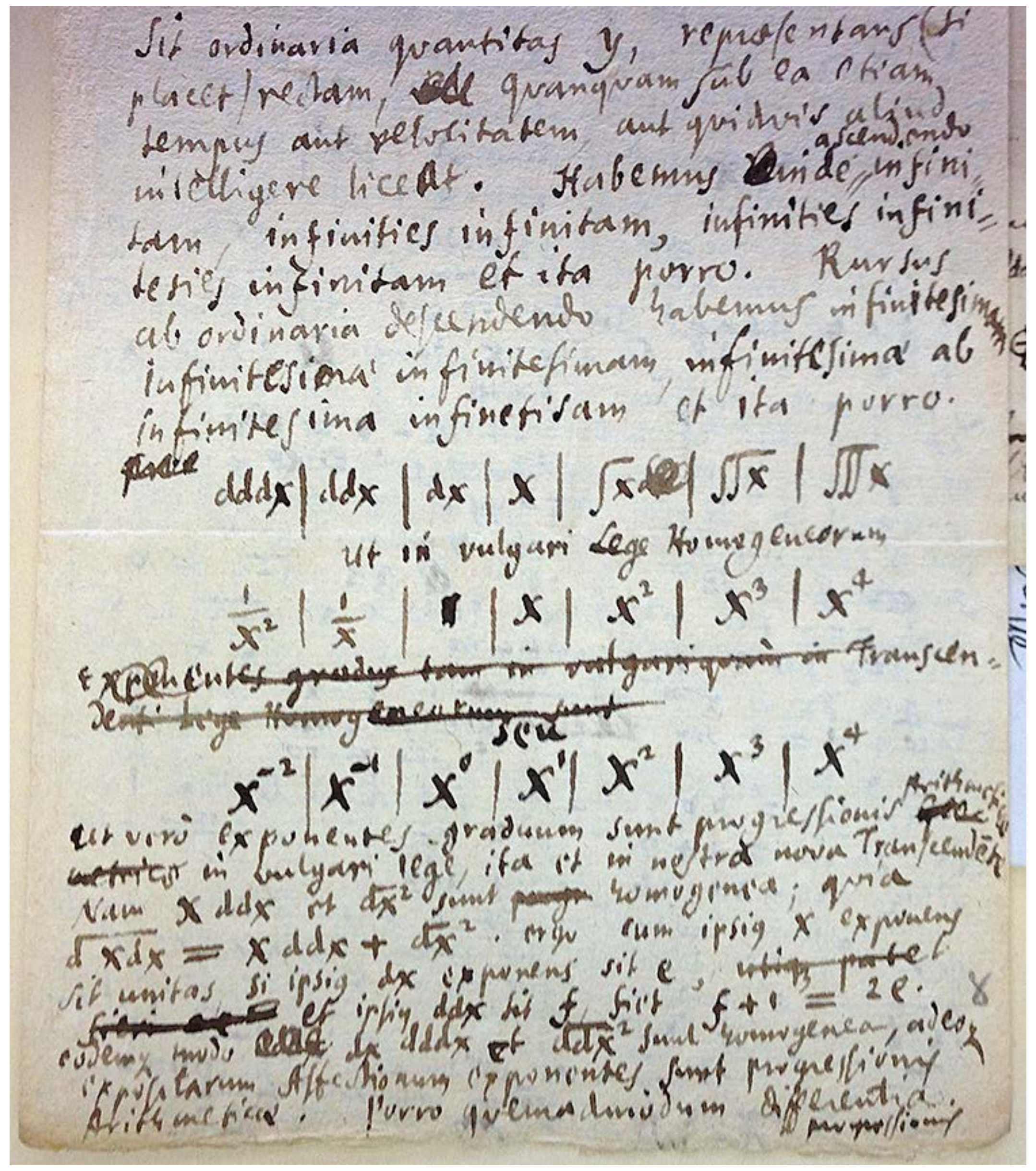


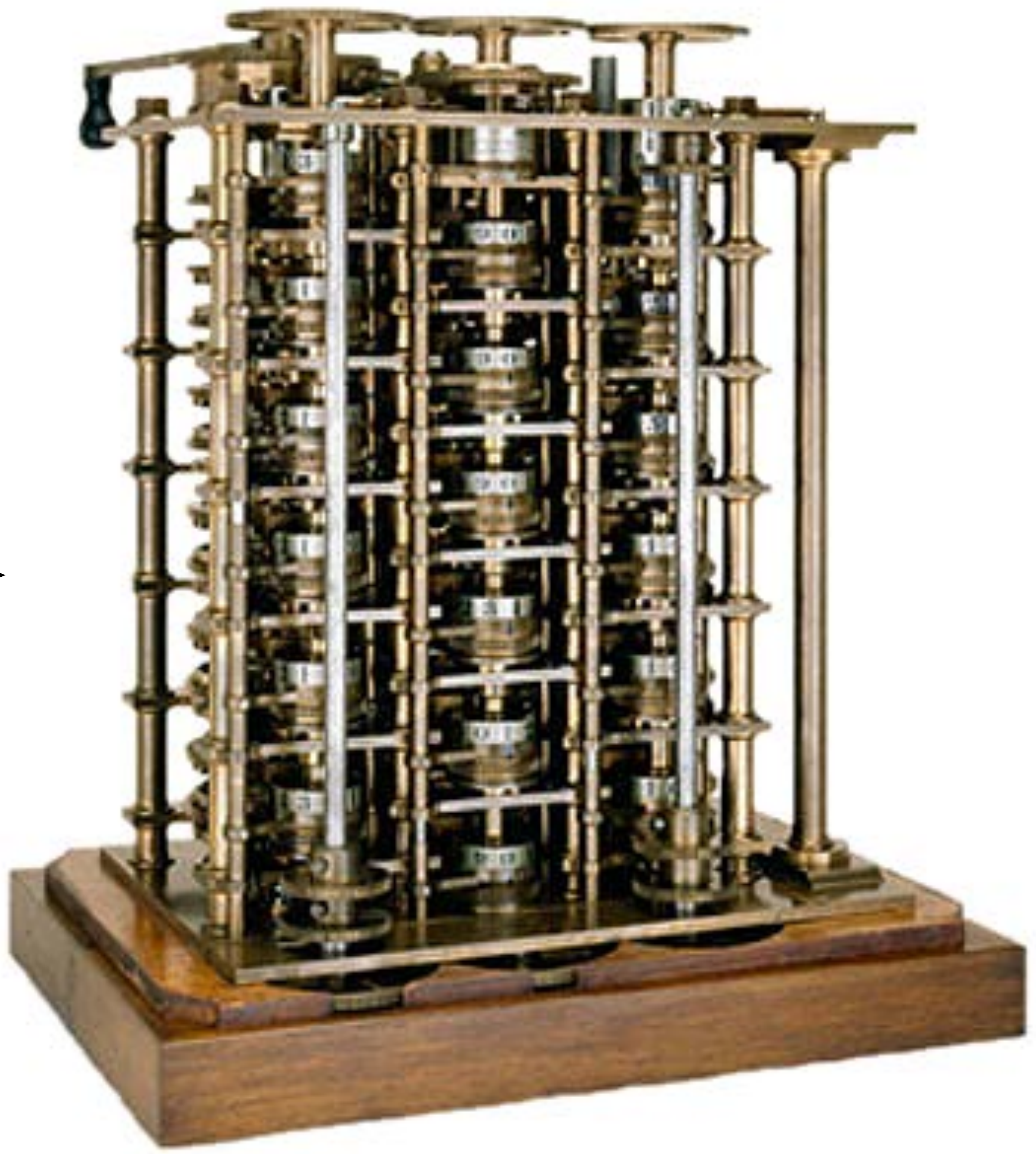


- Modern language for geometric computing
- Increasing impact on science & technology in 21st century

GRAND VISION

Translate differential geometry into **language** suitable for *computation*.





How can we get there?

A common "game" is played in DDG to obtain discrete definitions:

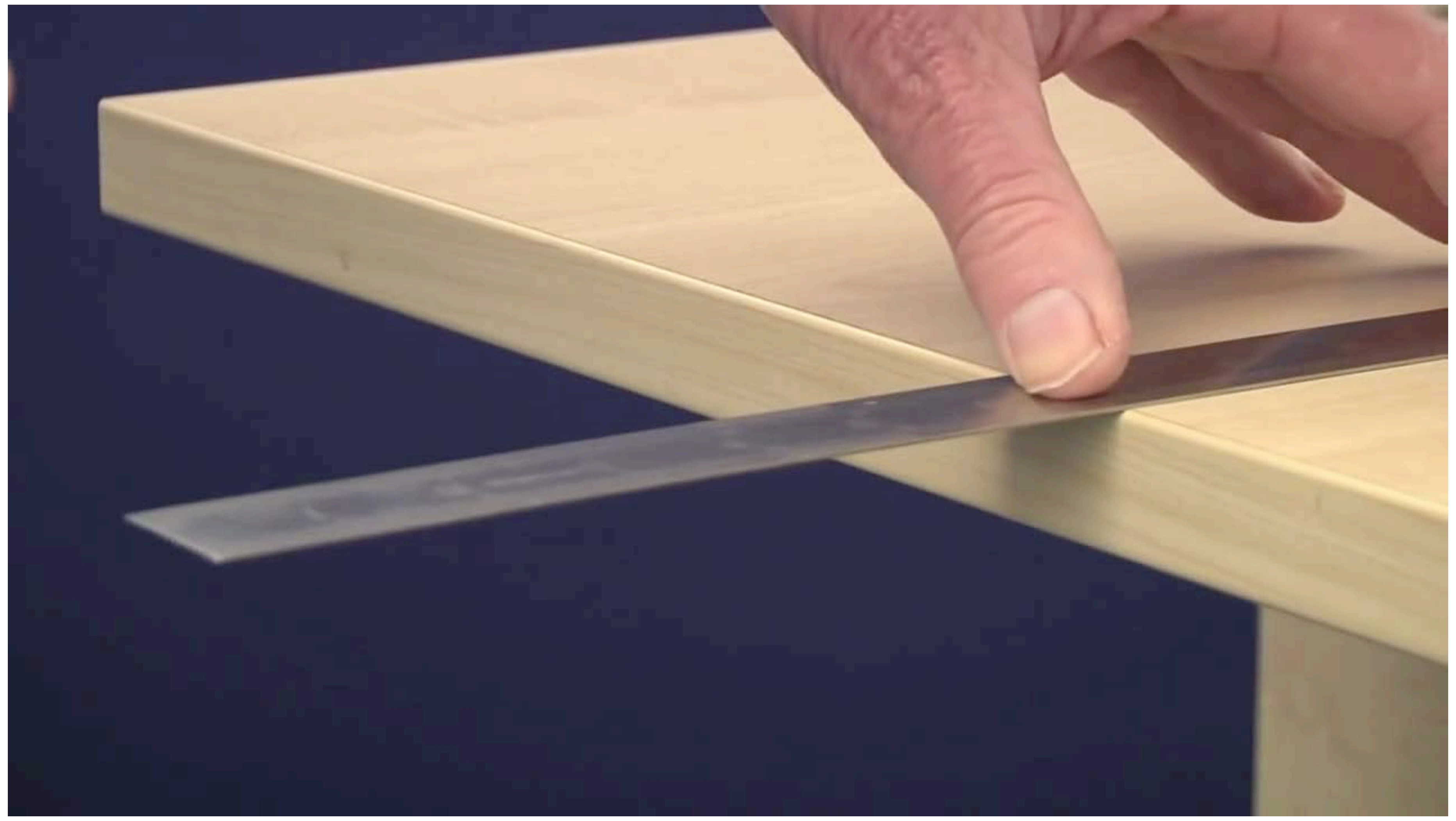
1. Write down several **equivalent** definitions in the smooth setting.
2. Apply each smooth definition to an object in the discrete setting.
3. Determine which properties are captured by each resulting **inequivalent** discrete definition.

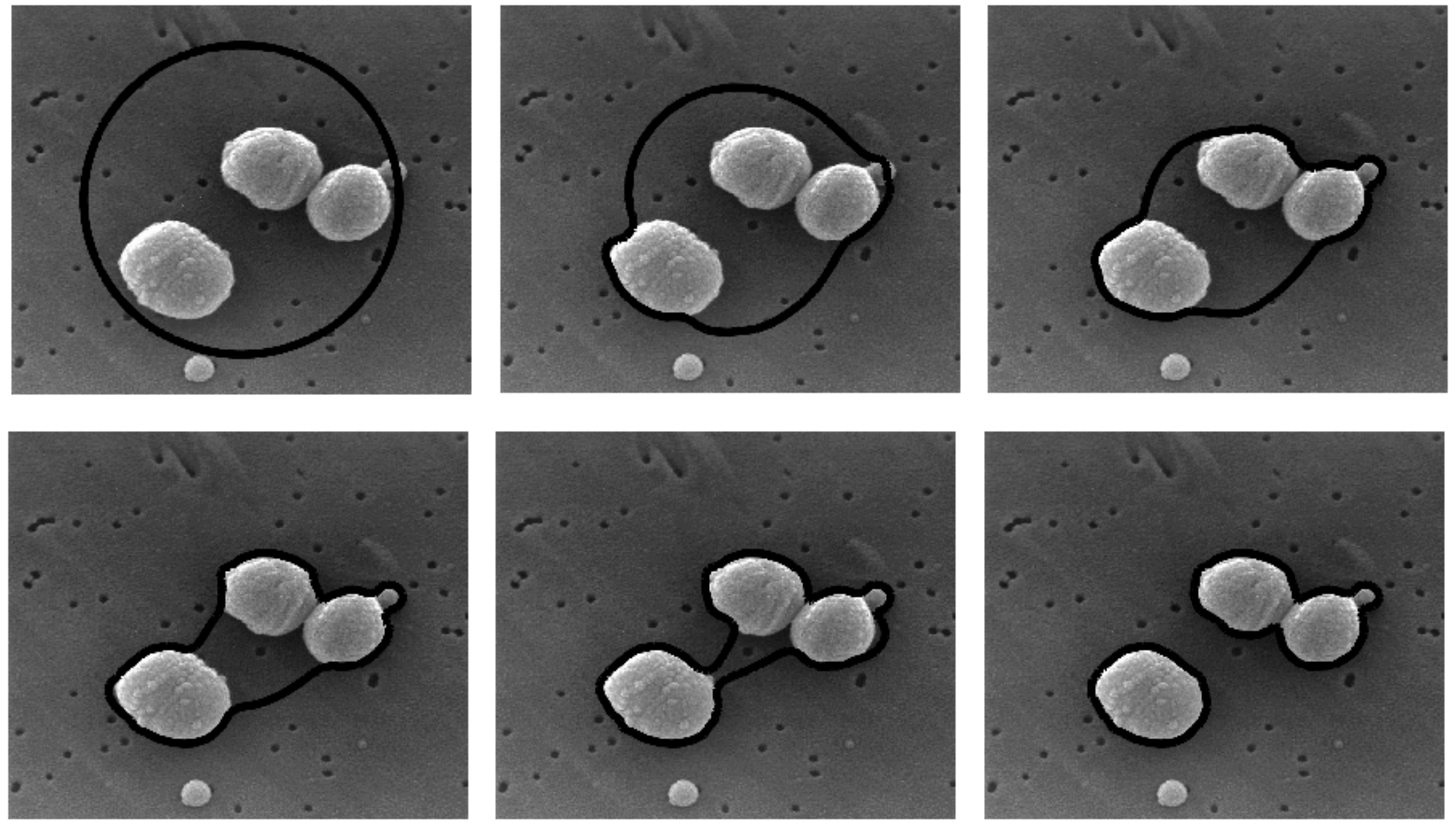
One often encounters a so-called *"no free lunch"* scenario: no single discrete definition captures *all* properties of its smooth counterpart.

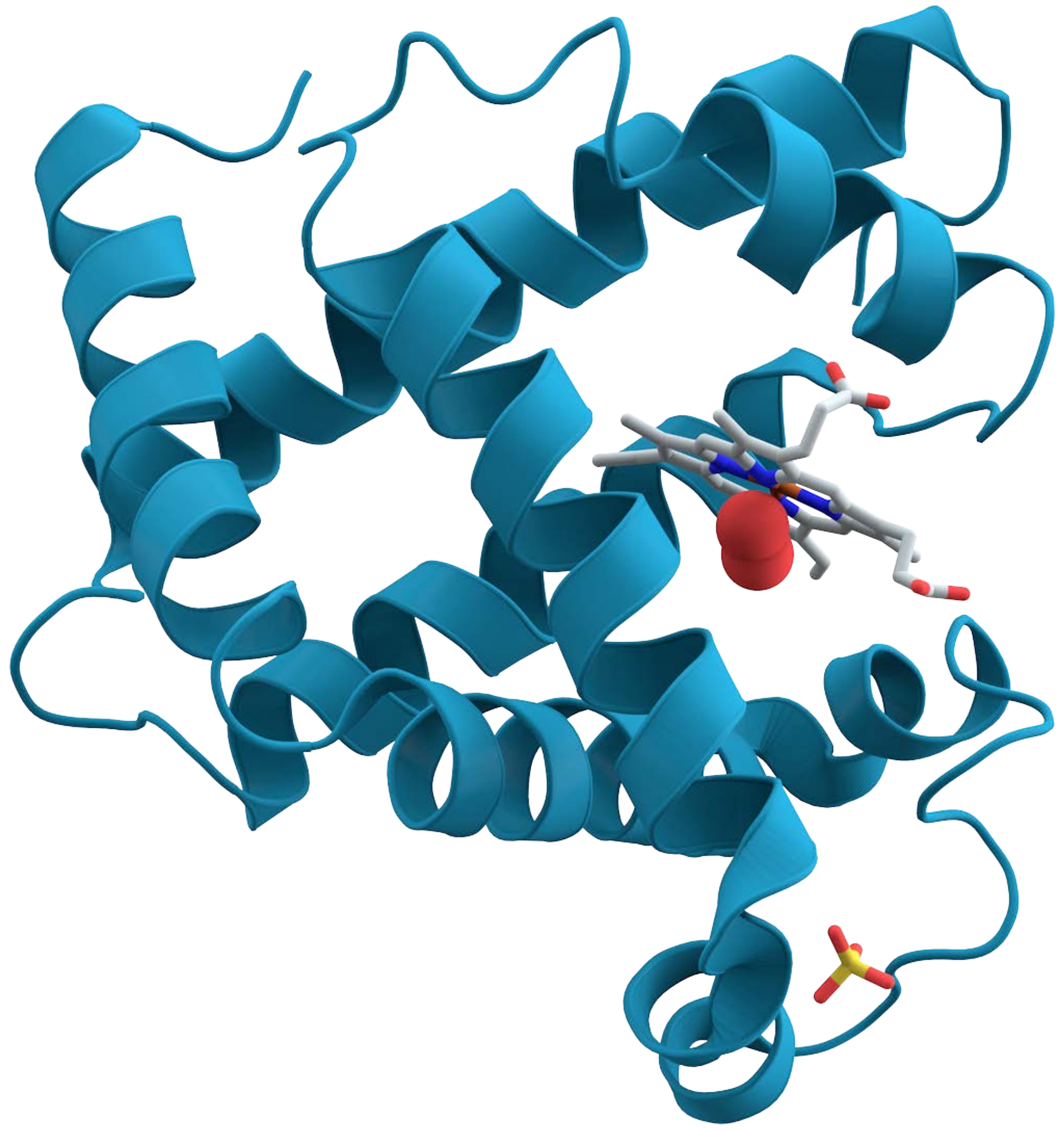
Example: Discrete Curvature of Plane Curves

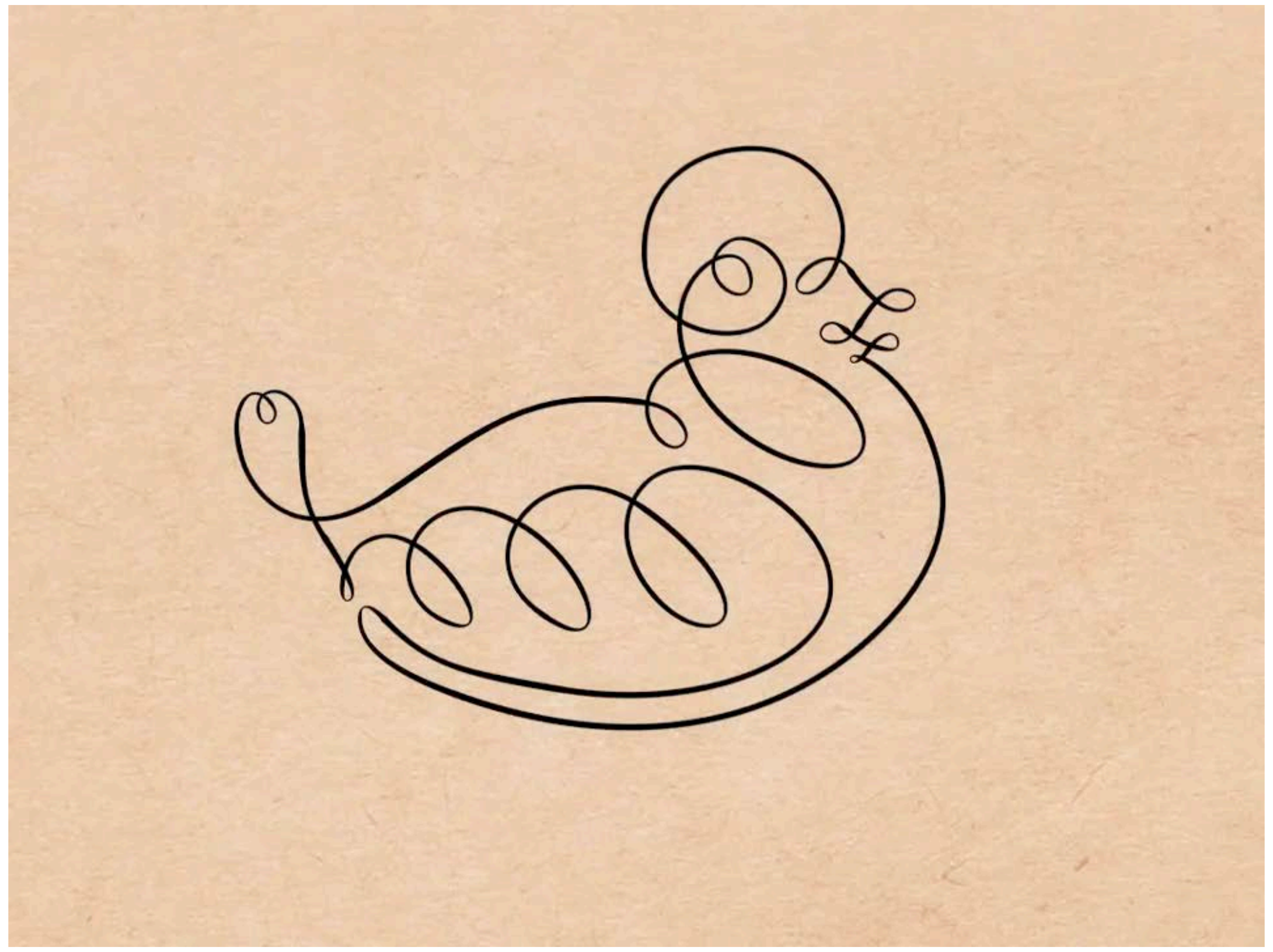
- **Toy example:** *curvature of plane curves*
  - Roughly speaking: "how much it bends"
  - First review smooth definition
  - Then play The Game to get discrete definition(s)
  - Will discover that no single definition is "best"
  - *Pick the definition best suited to the application*
- **Today** we will quickly cover a lot of ground…
- Will start more slowly from the basics **next lecture**

Curvature of a Curve—Motivation



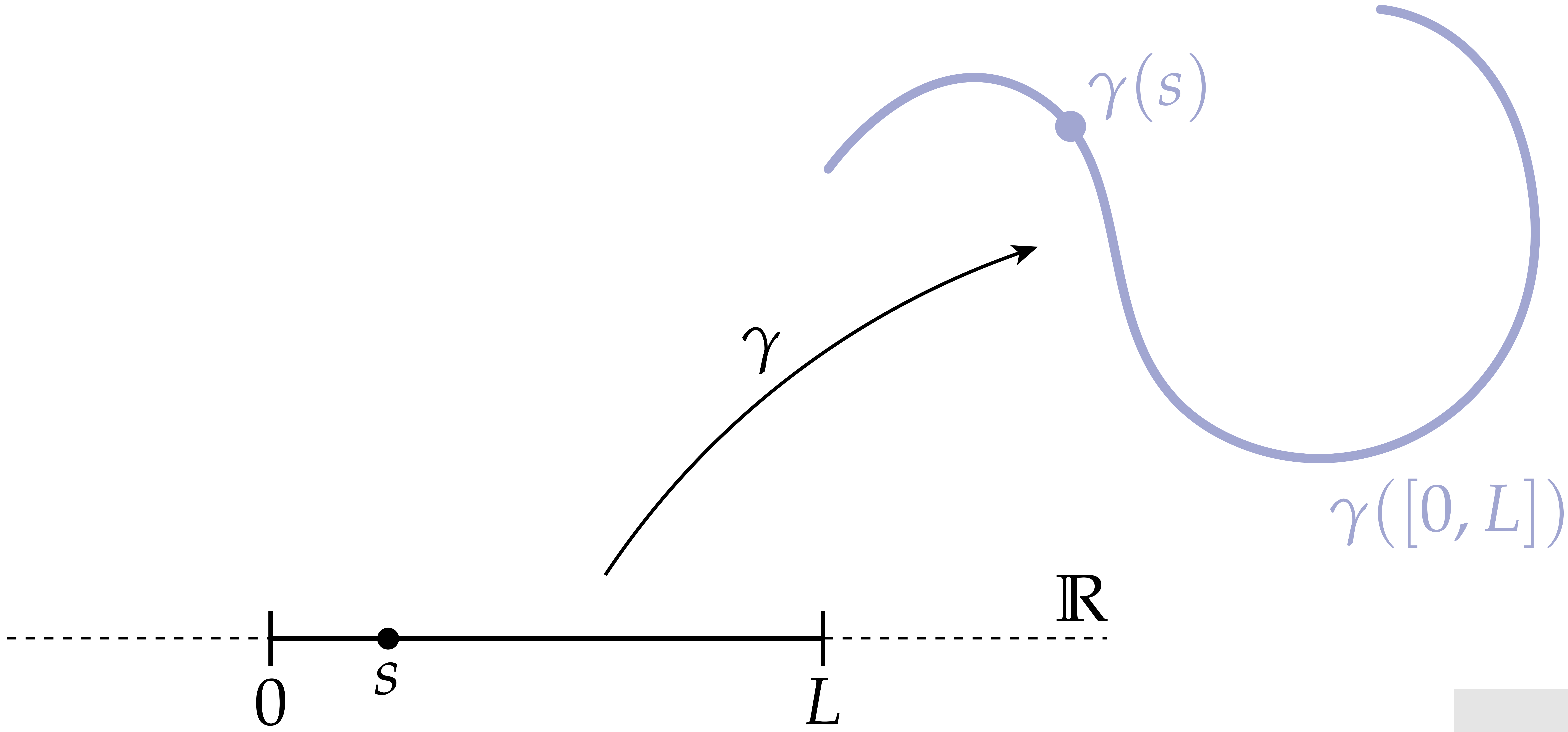






Curves in the Plane

In the smooth setting, a **parameterized curve** is a map* taking each point in an interval [0,L] of the real line to some point in the plane $\mathbb{R}^2$:



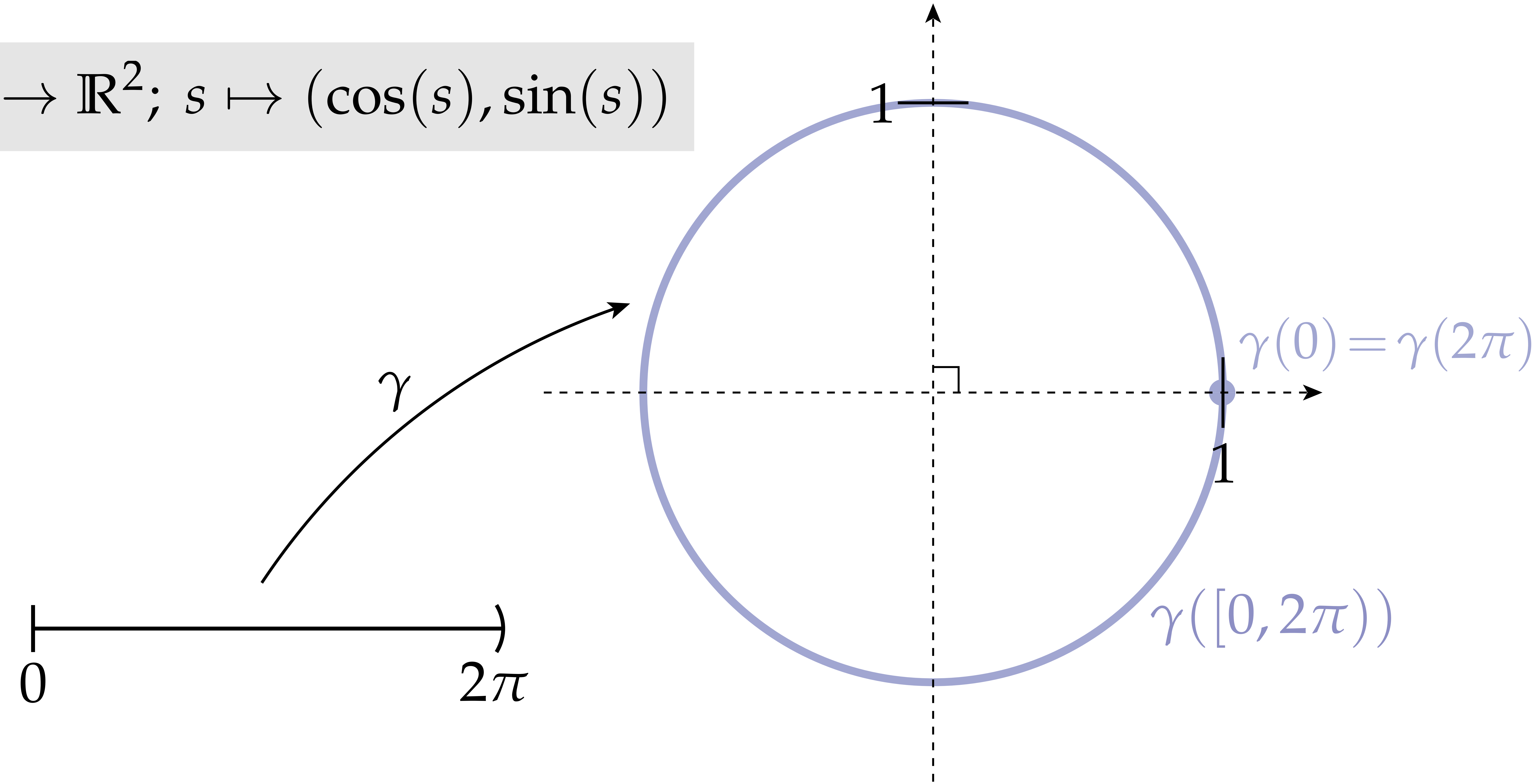
*Continuous, differentiable, smooth…

$$\gamma : [0, L] \to \mathbb{R}^2$$

Curves in the Plane—Example

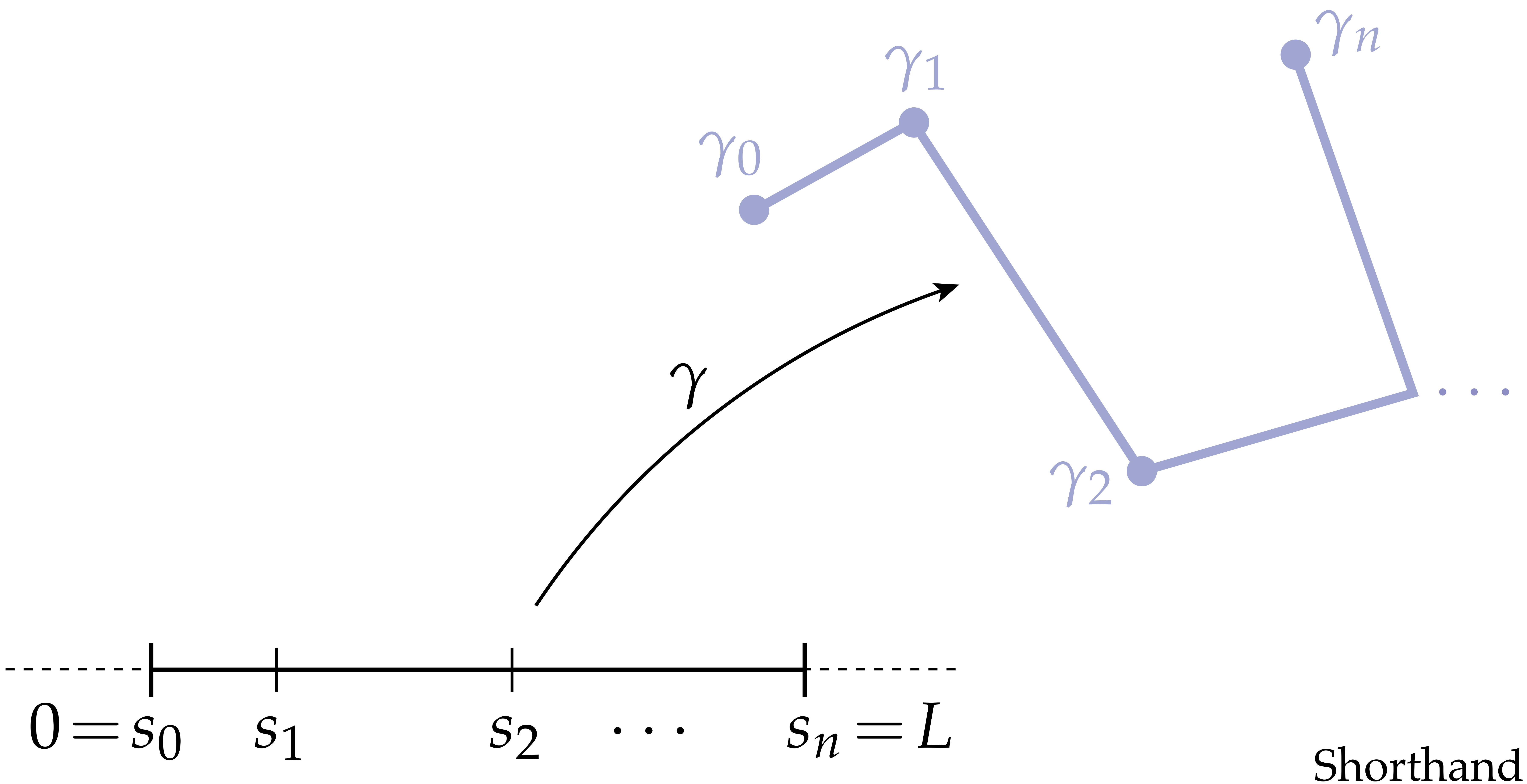
As an example, we can express a circle as a parameterized curve γ:

$$\gamma : [0, 2\pi) \to \mathbb{R}^2;\ s \mapsto (\cos(s), \sin(s))$$



Discrete Curves in the Plane

Special case: a **discrete curve** is a *piecewise linear* parameterized curve, *i.e.*, it is a sequence of **vertices** connected by straight line segments:

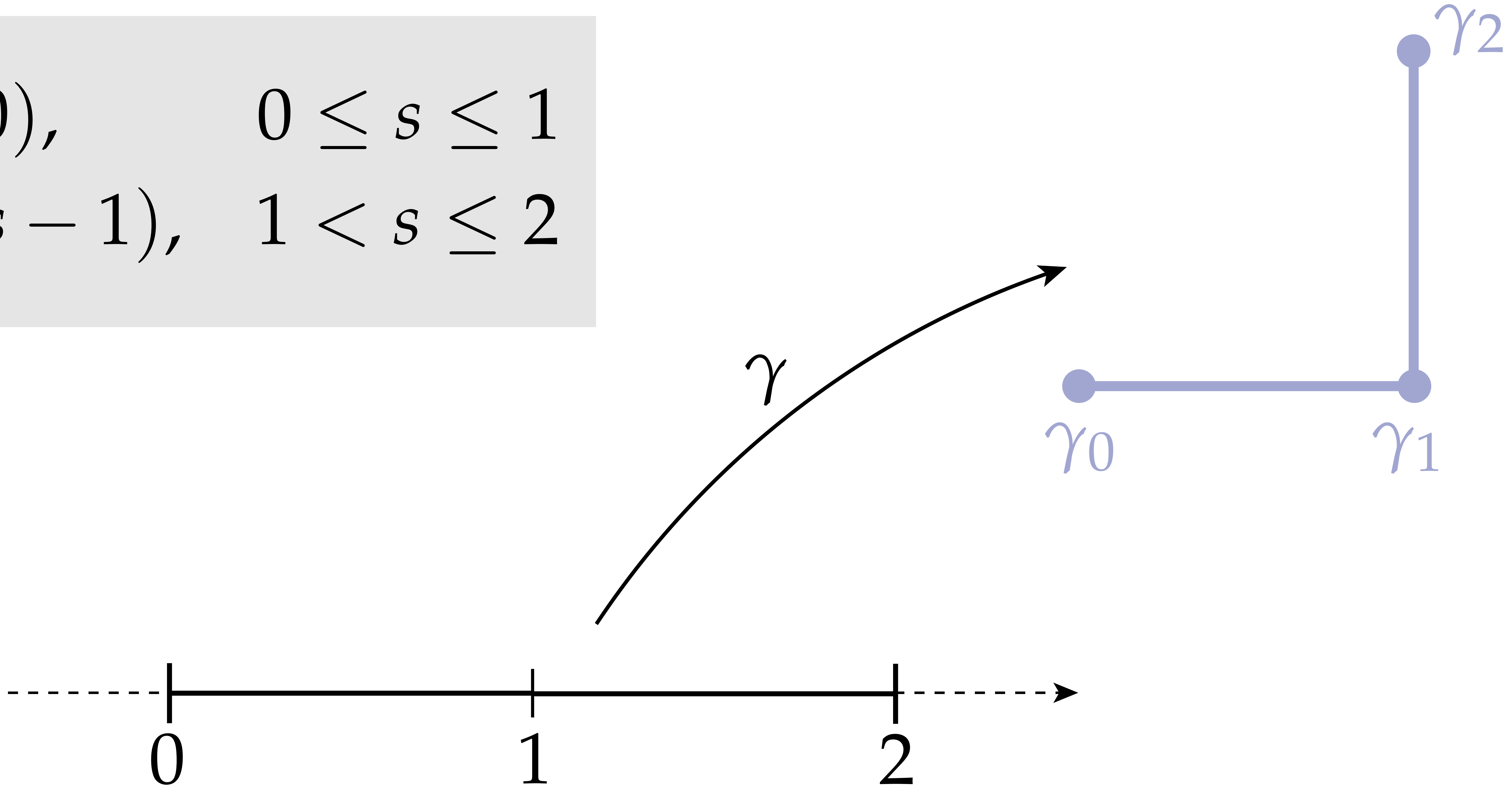


Shorthand: $\gamma_i := \gamma(s_i)$

Discrete Curves in the Plane—Example

A simple example is a curve comprised of two segments:

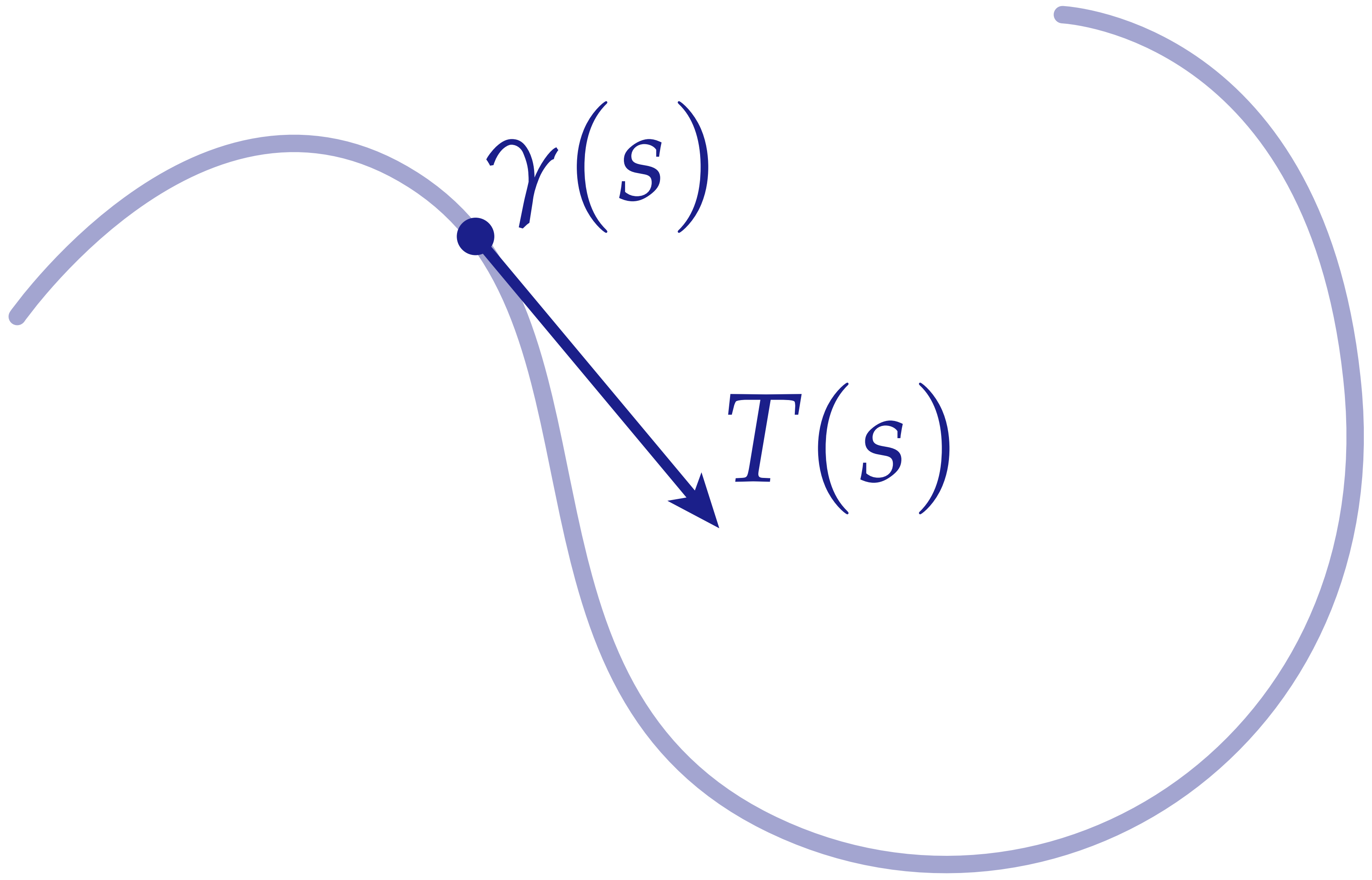
$$\gamma(s) := \begin{cases} (s,0), & 0 \leq s \leq 1 \\ (1, s-1), & 1 < s \leq 2 \end{cases}$$



Tangent of a Curve

- Informally, a vector is *tangent* to a curve if it "just barely grazes" the curve.
- More formally, the **unit tangent** (or just **tangent**) of a parameterized curve is the map obtained by normalizing its first derivative*:

$$T(s) := \tfrac{d}{ds}\gamma(s) \Big/ \Big|\tfrac{d}{ds}\gamma(s)\Big|$$



- If the derivative already has unit length, then we say the curve is **arc-length parameterized** and can write the tangent as just

$$T(s) := \tfrac{d}{ds}\gamma(s)$$

*Assuming curve never slows to a stop, *i.e.*, assuming it's "regular"

Tangent of a Curve—Example

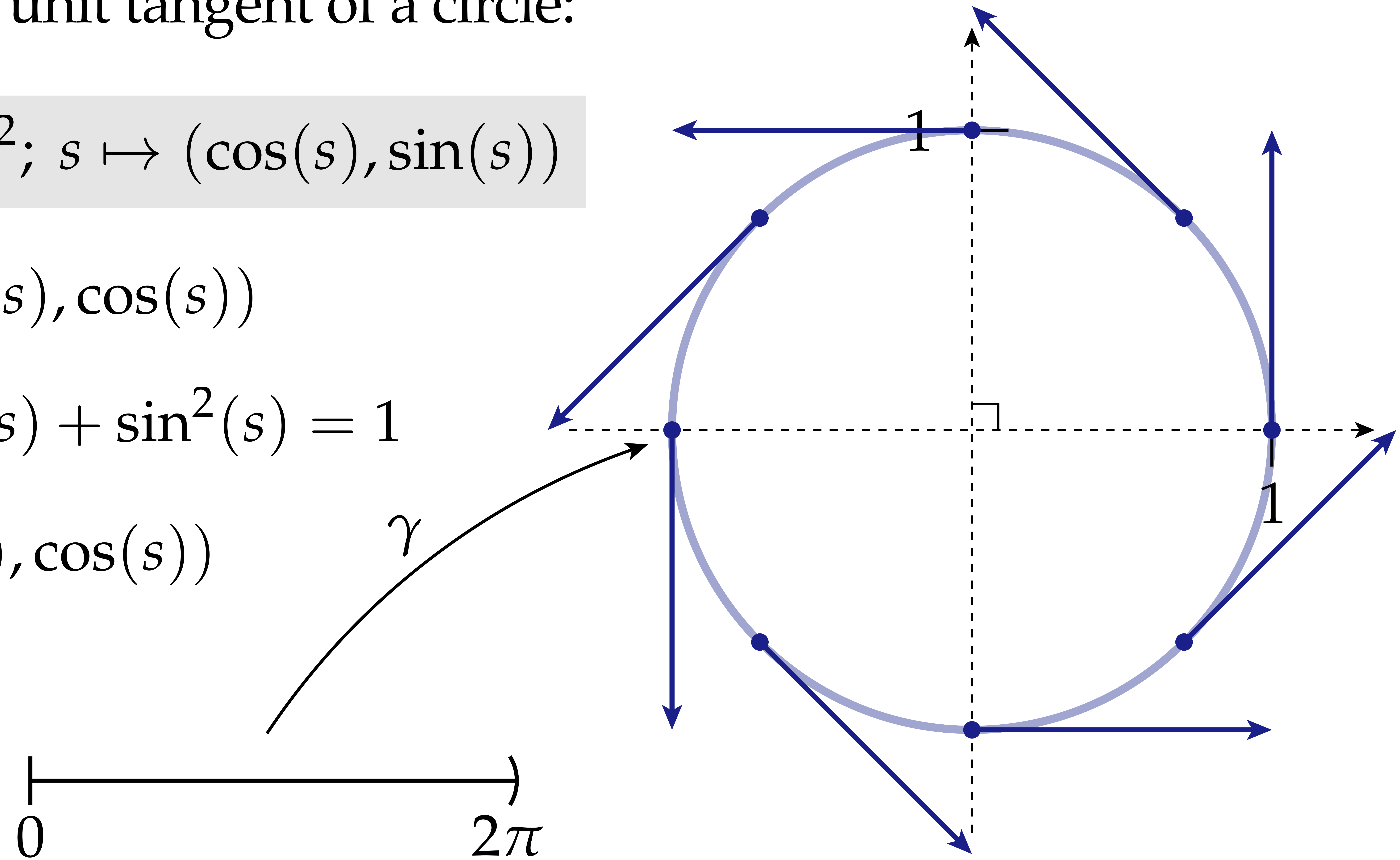
Let's compute the unit tangent of a circle:

$$\gamma : [0, 2\pi) \to \mathbb{R}^2;\ s \mapsto (\cos(s), \sin(s))$$

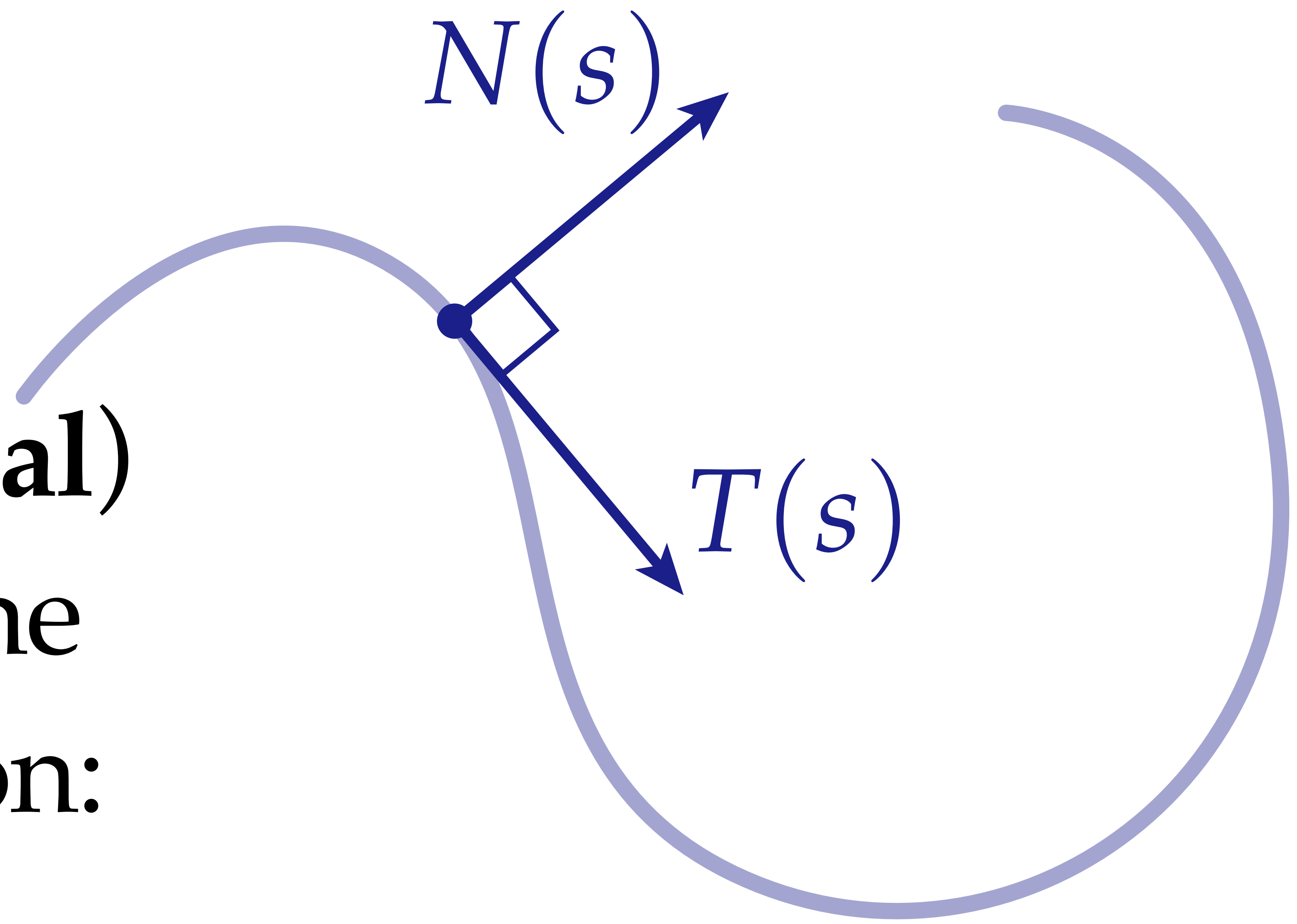
$$\frac{d}{ds}\gamma(s) = (-\sin(s), \cos(s))$$

$$\left|\frac{d}{ds}\gamma(s)\right| = \cos^2(s) + \sin^2(s) = 1$$

$$\Rightarrow T = (-\sin(s), \cos(s))$$



Normal of a Curve



- Informally, a vector is *normal* to a curve if it "sticks straight out" of the curve.
- More formally, the **unit normal** (or just **normal**) can be expressed as a quarter-rotation $\mathcal{J}$ of the unit tangent in the counter-clockwise direction:

$$N(s) := \mathcal{J}T(s)$$

- In coordinates (x,y), a quarter-turn can be achieved by* simply exchanging x and y, and then negating y:

$$(x,y) \overset{\mathcal{J}}{\mapsto} (-y,x)$$

*Why does this work?

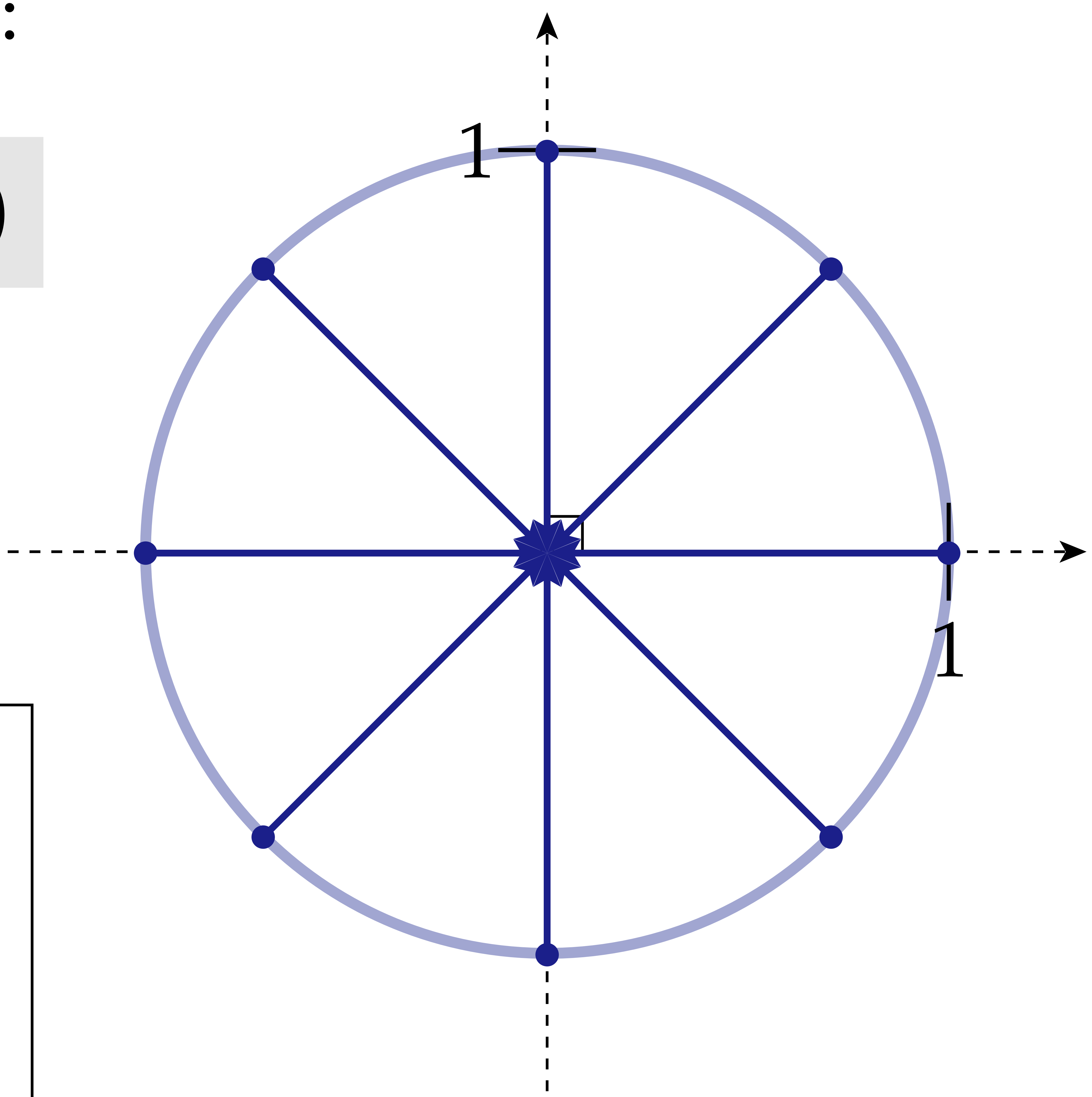
Normal of a Curve—Example

Let's compute the unit normal of a circle:

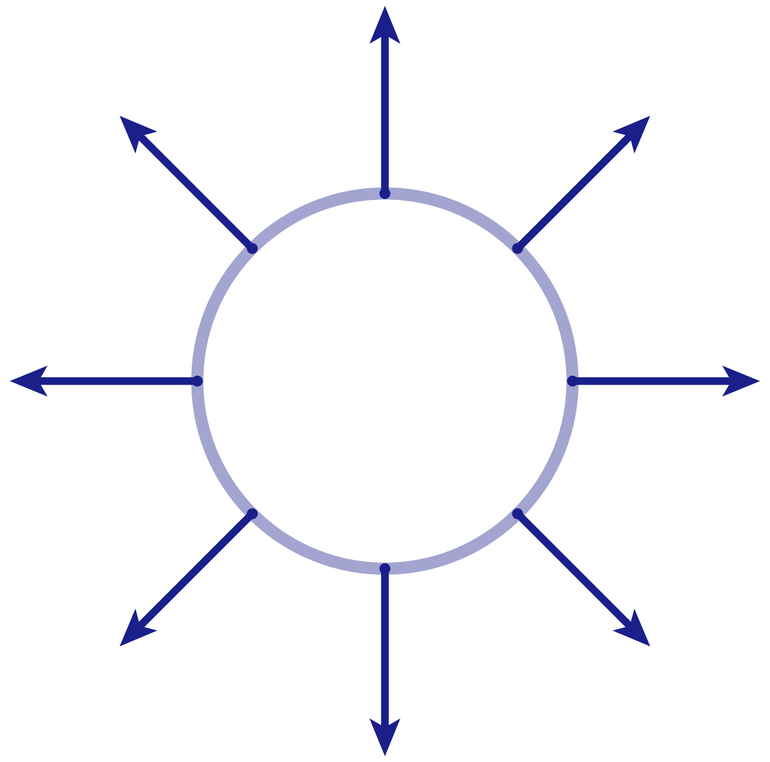
$$\gamma : [0, 2\pi) \to \mathbb{R}^2;\ s \mapsto (\cos(s), \sin(s))$$

$$T(s) = (-\sin(s), \cos(s))$$

$$N(s) = \mathcal{J}T(s) = (-\cos(s), -\sin(s))$$

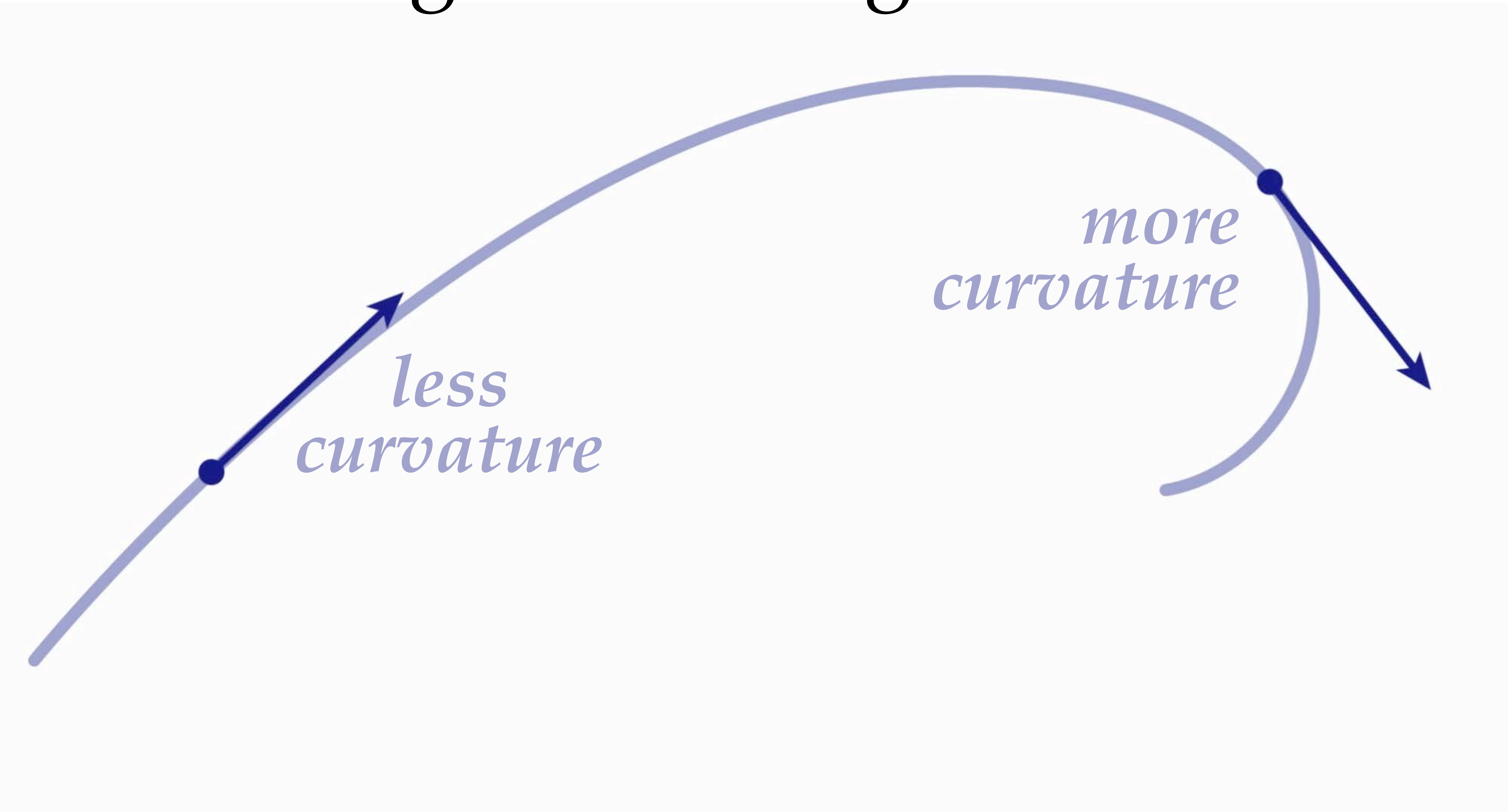


Note: could also adopt the convention $N = -\mathcal{J}T$. (Just remain consistent!)



Curvature of a Plane Curve

- Informally, curvature describes "how much a curve bends"
- More formally, the **curvature** of an arc-length parameterized plane curve can be expressed as the rate of change in the tangent*

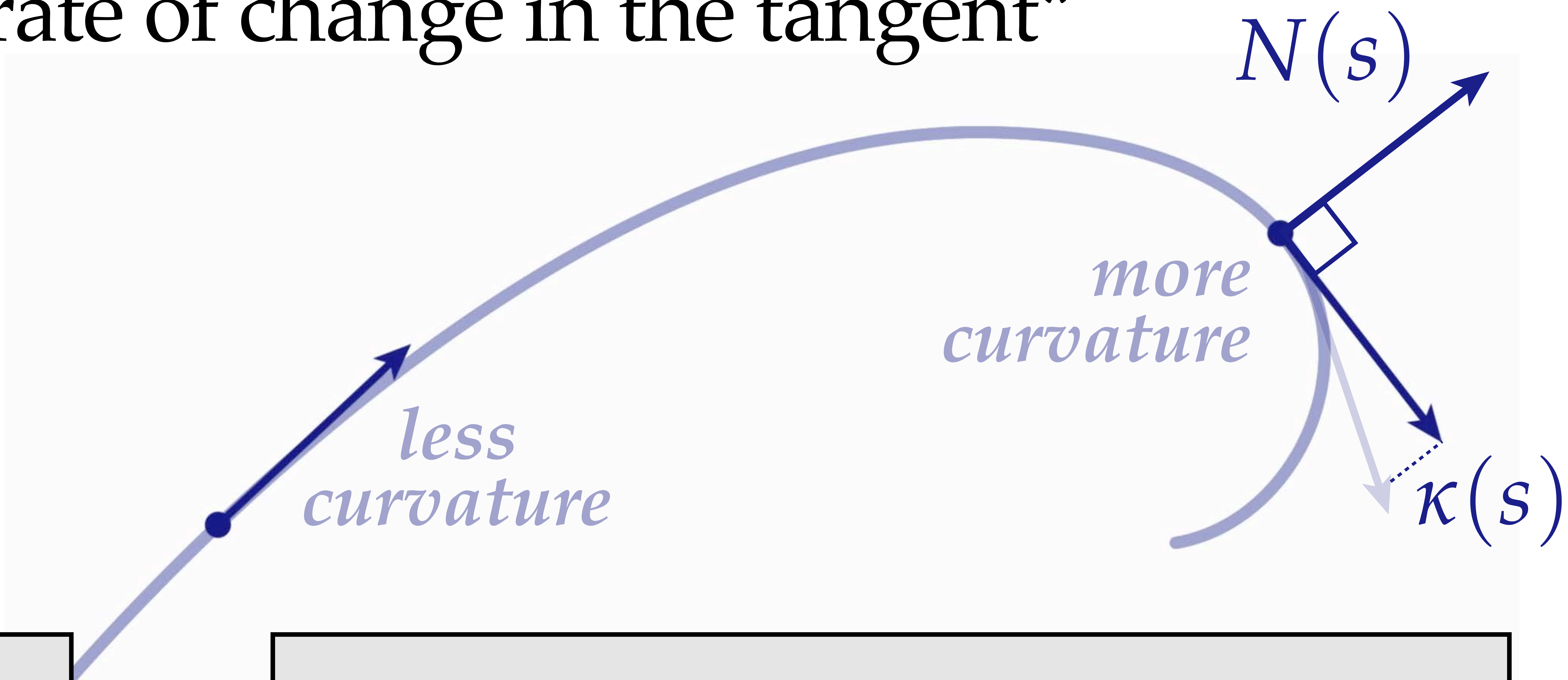


Curvature of a Plane Curve

- Informally, curvature describes "how much a curve bends"
- More formally, the **curvature** of an arc-length parameterized plane curve can be expressed as the rate of change in the tangent*

$$\kappa(s) := \langle N(s), \tfrac{d}{ds} T(s) \rangle$$

$$= \langle N(s), \tfrac{d^2}{ds^2} \gamma(s) \rangle$$



KEY IDEA I

Curvature is a *second derivative.*

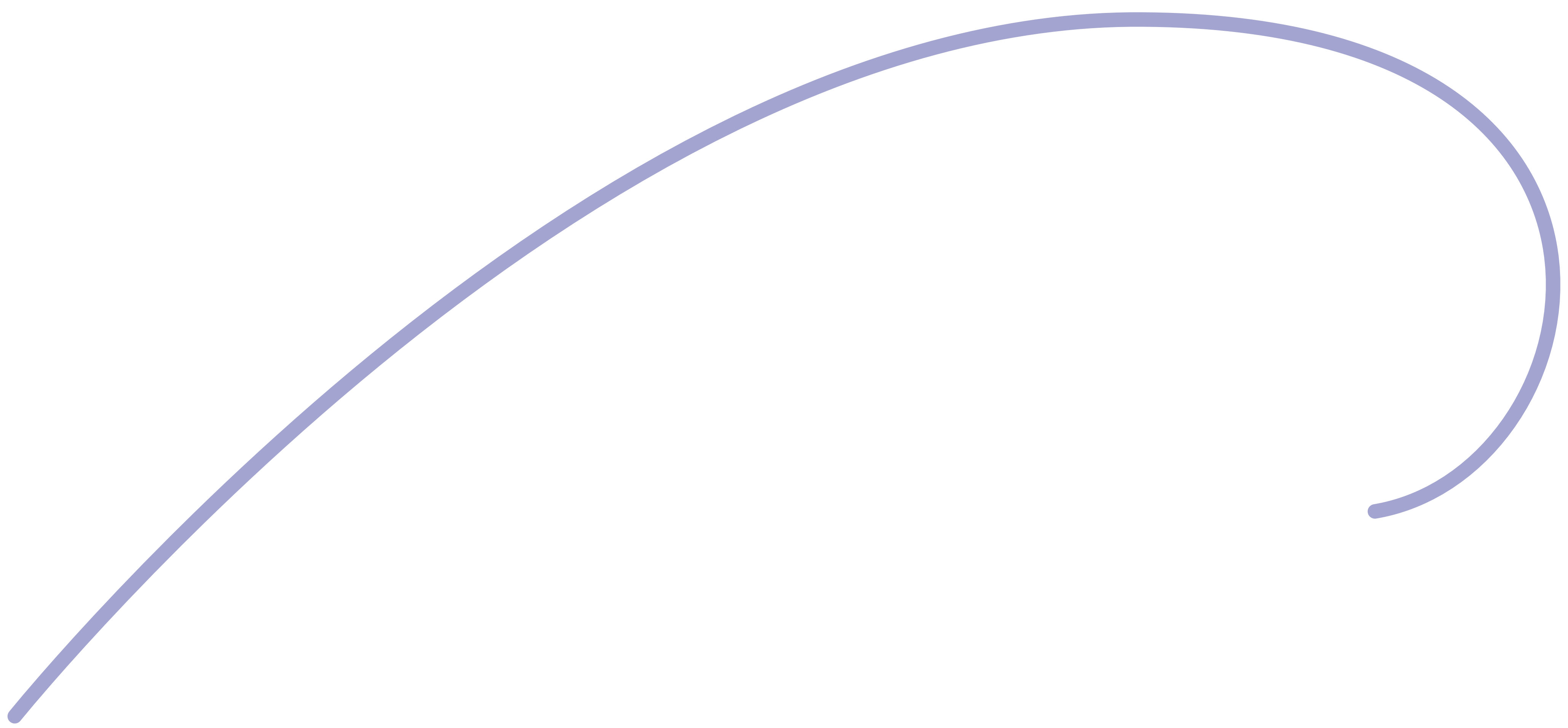
KEY IDEA II

Curvature is a *signed quantity.*

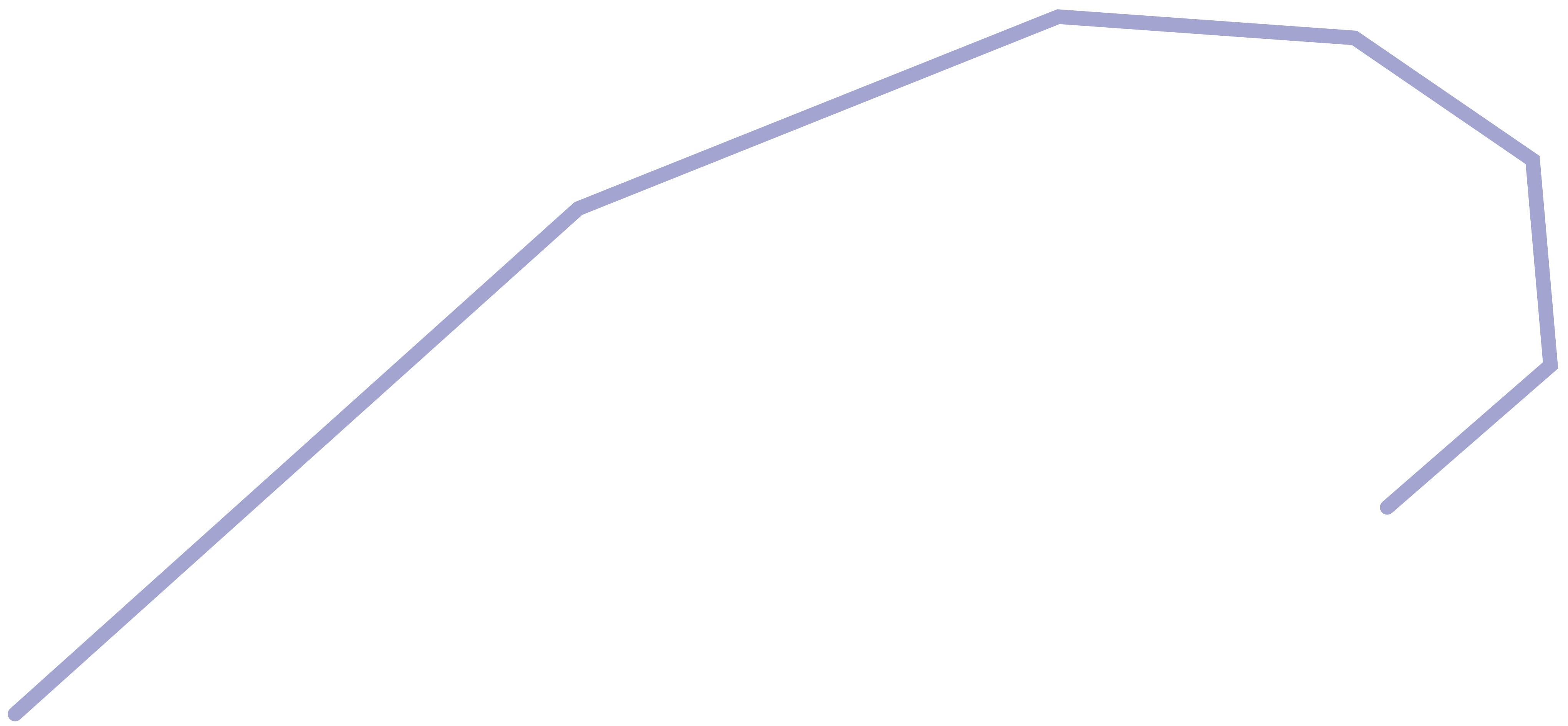
*Here, angle brackets denote the usual dot product: $\langle (a,b),(x,y) \rangle := ax + by$

Curvature: From Smooth to Discrete

Smooth



Discrete



Key idea

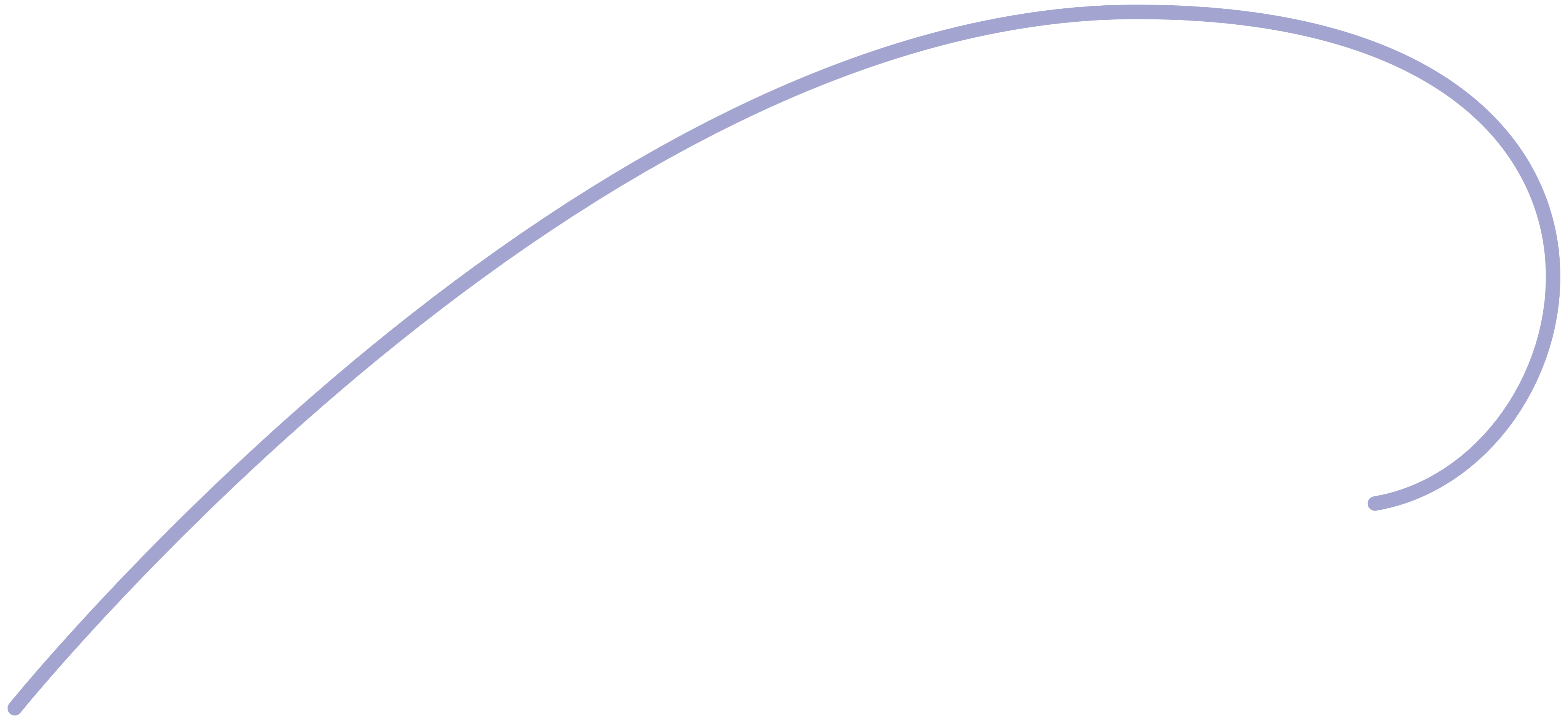
Curvature is a *second derivative*.

$$\kappa = \langle \mathcal{J} \tfrac{d}{ds}\gamma, \tfrac{d^2}{ds^2}\gamma \rangle$$

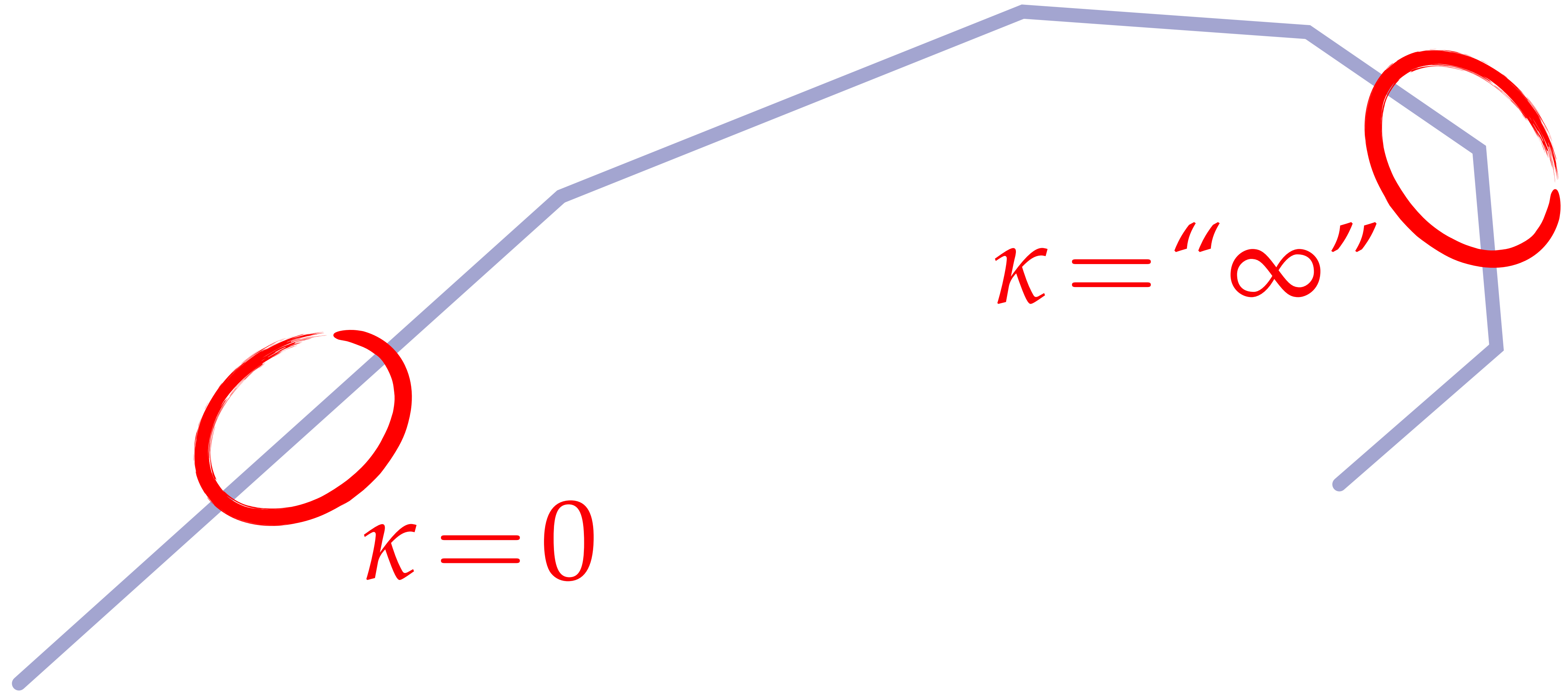
Can we directly apply this definition to a discrete curve?

Curvature: From Smooth to Discrete

SMOOTH



DISCRETE



~~**KEY IDEA**~~

~~Curvature is a *second derivative*.~~

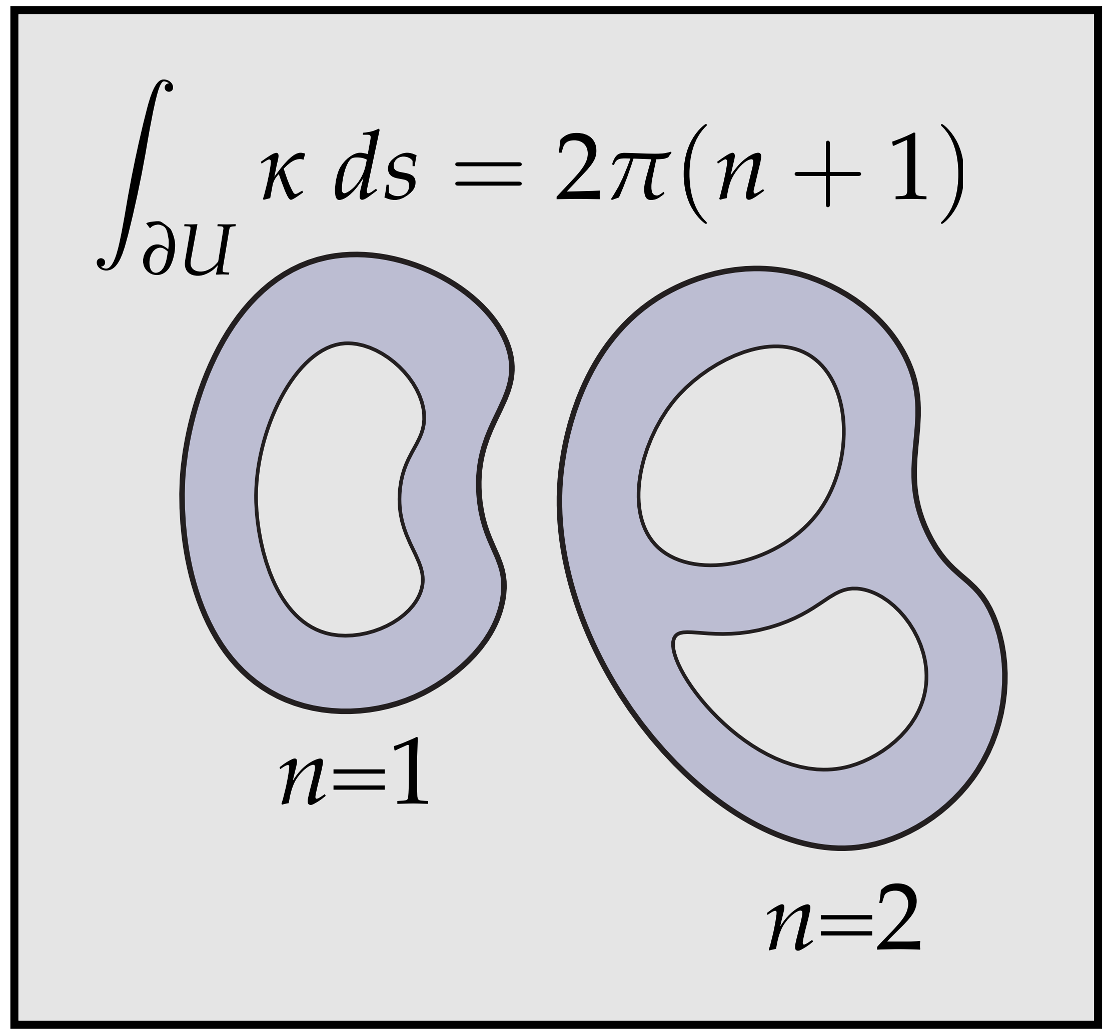
$$\cancel{\kappa = \left\langle \mathcal{J} \frac{d}{ds}\gamma, \frac{d^2}{ds^2}\gamma \right\rangle}$$

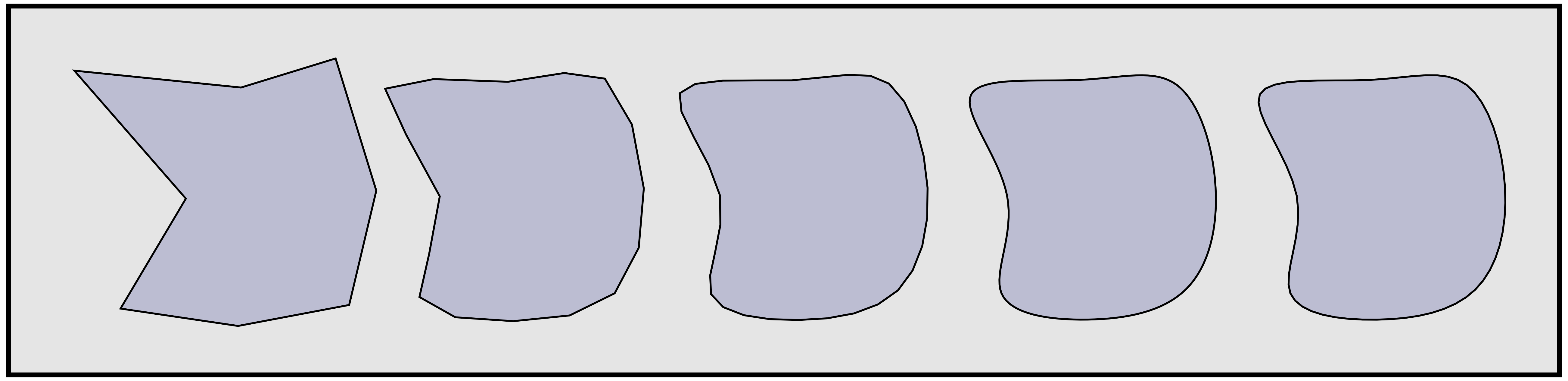
Can we directly apply this definition to a discrete curve?

No! Will get either zero or “∞”. Need to think about it another way…

When is a Discrete Definition "Good?"

- How will we know if we came up with a good definition?
- Many different criteria for "good":
  - *satisfies (some of the) same properties/theorems as smooth curvature*
  - *converges to smooth value as we refine our curve*
  - *efficient to compute / solve equations*
  - …

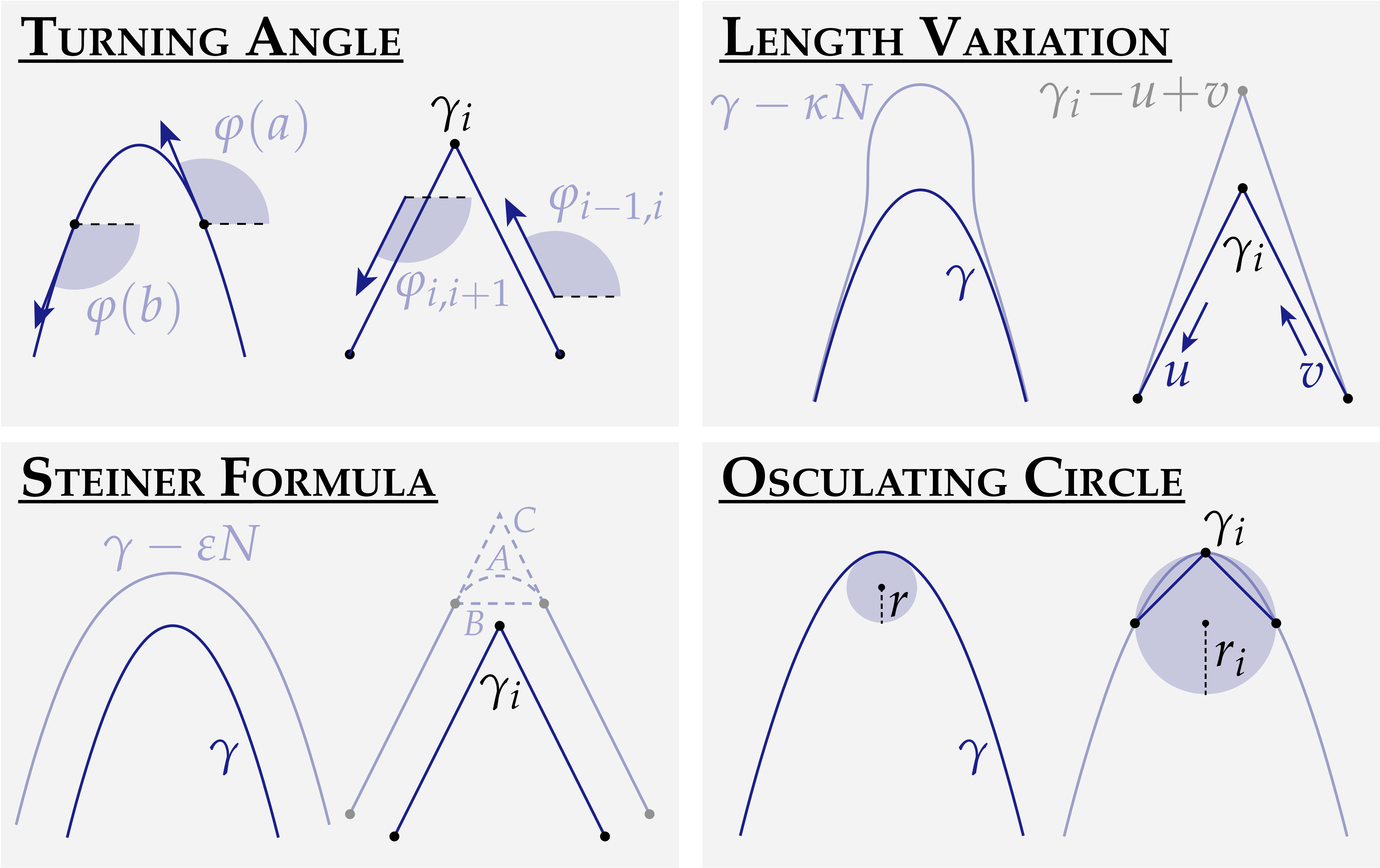




```
Complex Ta = gamma[i] - gamma[i-1];
Complex Tb = gamma[i+1] - gamma[i];
double kappa = (Tb*Ta.inv()).arg();
```

Playing the Game

- In the **smooth** setting, there are several other **equivalent** definitions of curvature.
- <u>IDEA:</u> perhaps some of these definitions can be applied *directly* to our discrete curve!
- Actually, all four can—and will have different consequences…

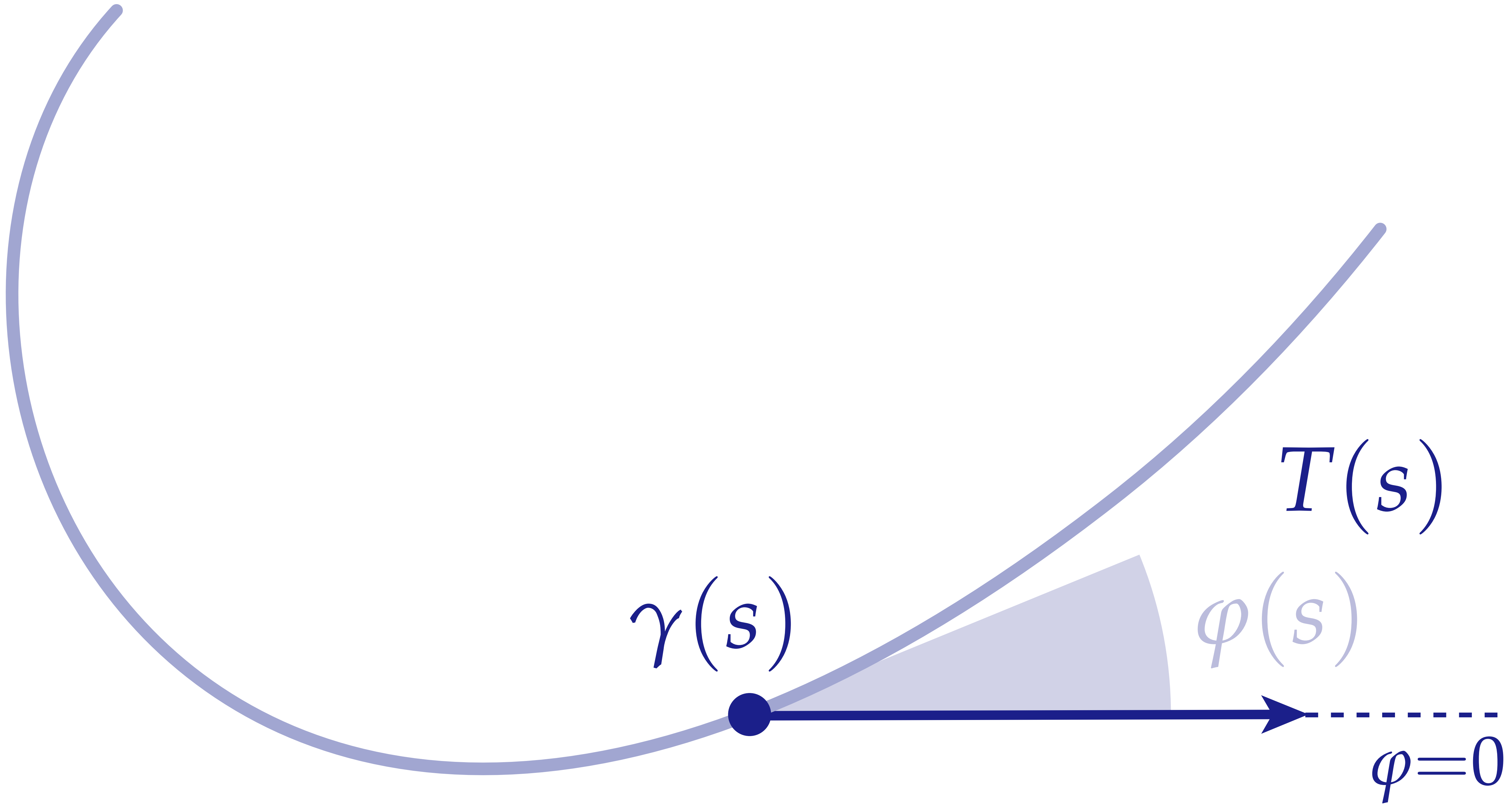


Turning Angle

- Our initial definition of curvature was the *rate of change of the tangent in the normal direction*.

$$\kappa(s) = \langle N(s), \tfrac{d}{ds}\gamma(s)\rangle$$

- Equivalently, we can measure the *rate of change of the angle the tangent makes with the horizontal*:



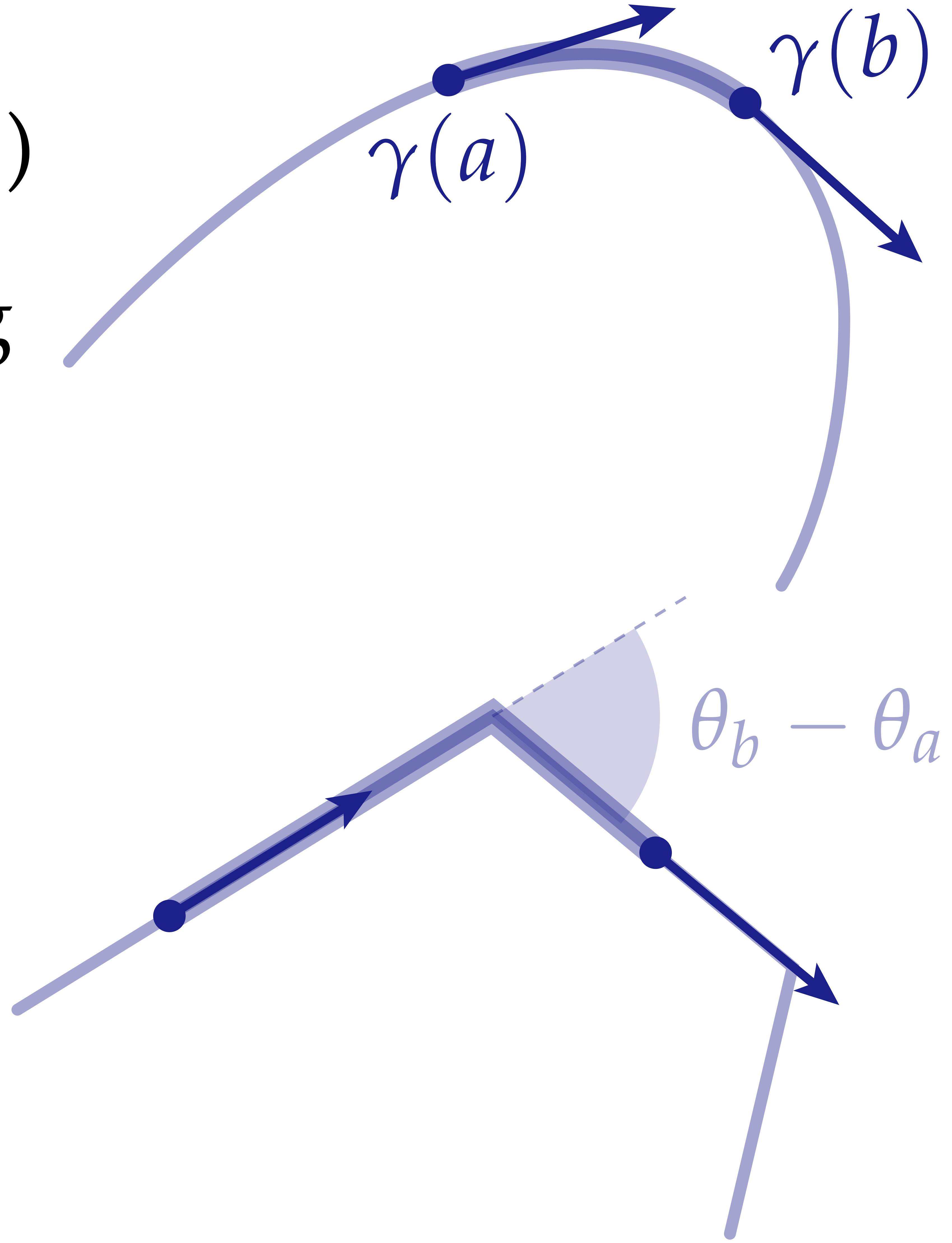
$$\boxed{\kappa(s) = \tfrac{d}{ds}\varphi(s)}$$

Integrated Curvature

- Still can't evaluate curvature at vertices of a discrete curve (*at what rate does the angle change?*)
- But let's consider the *integral* of curvature along a short segment:

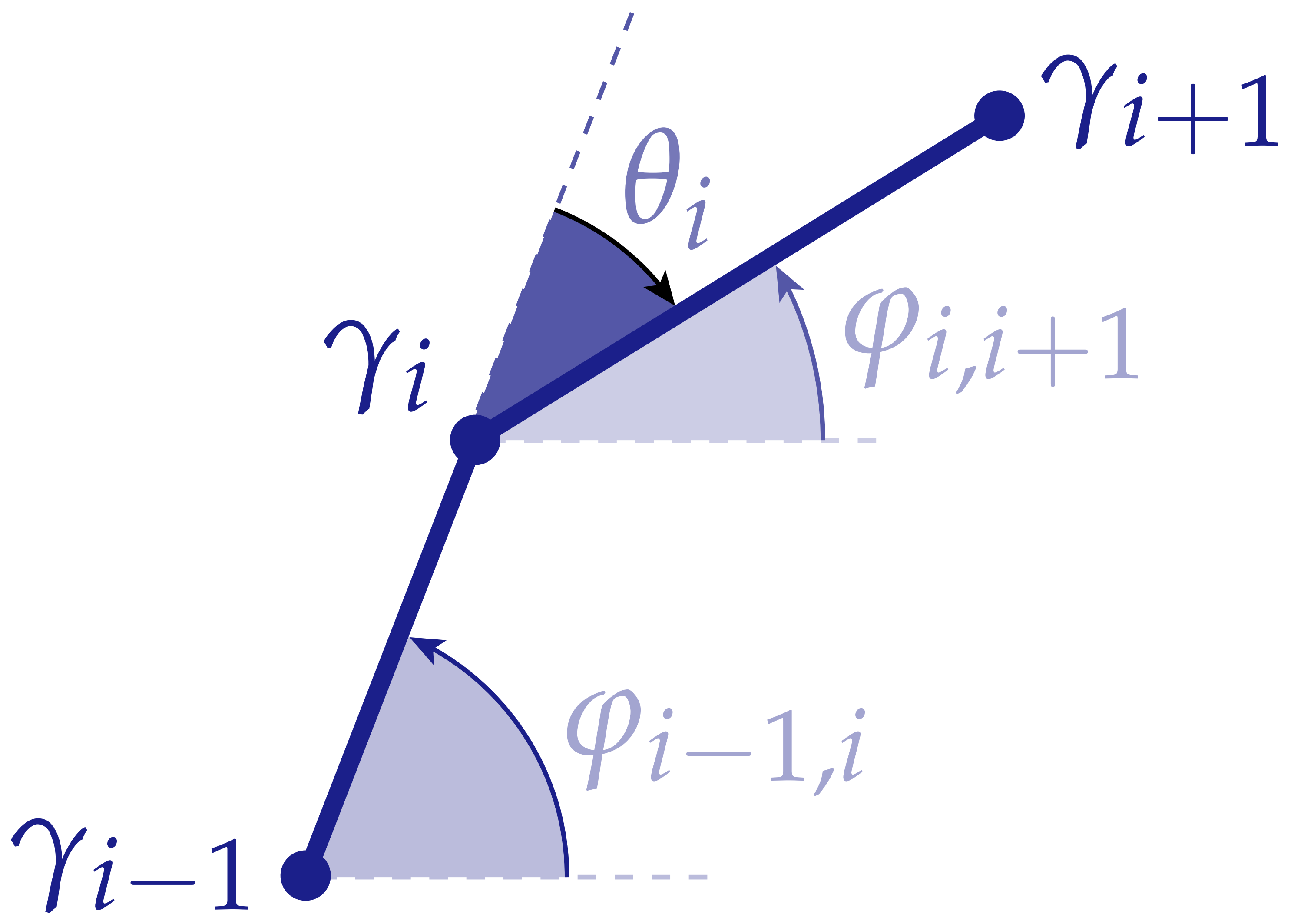
$$\int_a^b \kappa(s)\, ds = \int_a^b \tfrac{d}{ds}\varphi(s)\, ds = \varphi(b) - \varphi(a)$$

- Instead of *derivative* of angle, we now just have a *difference* of angles.
- **This definition works for our discrete curve!**



Discrete Curvature (Turning Angle)

- This formula gives us our first definition of discrete curvature, as just the *turning angle* at the vertex of each curve*:



$$\theta_i := \text{angle}(\gamma_i - \gamma_{i-1}, \gamma_{i+1} - \gamma_i)$$

$$\boxed{\kappa_i^A := \theta_i \qquad \textbf{(turning angle)}}$$

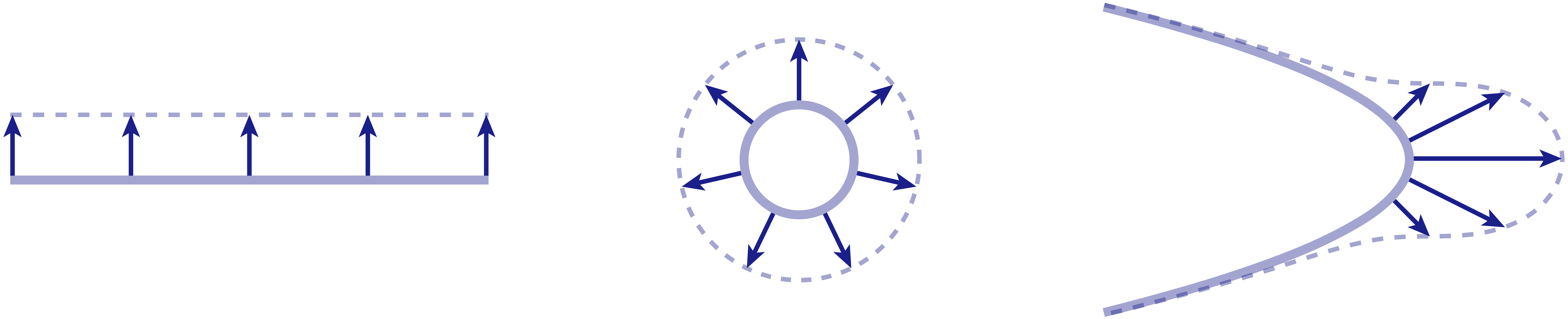
- Common theme: most natural discrete quantities are often <u>*integrated*</u> rather than <u>*pointwise*</u> values.
- Here: *total change in angle*, rather than *derivative of angle*.

Length Variation

- Are there *other* ways to get a definition for discrete curvature?
- Well, here's a useful fact about curvature from the smooth setting:

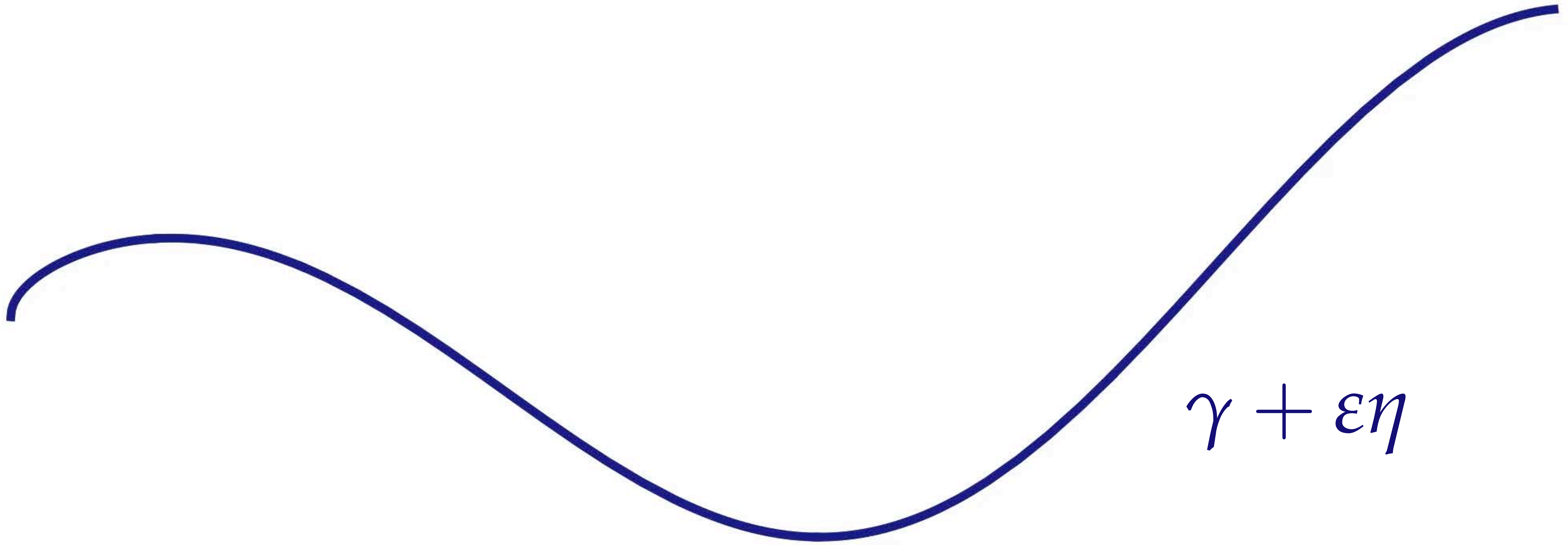
> *The fastest way to decrease the length of a curve is to move it in the normal direction, with speed proportional to curvature.*

- **Intuition**: in flat regions, normal motion doesn't change curve length; in curved regions, the change in length (*per unit length*) is large:



Length Variation

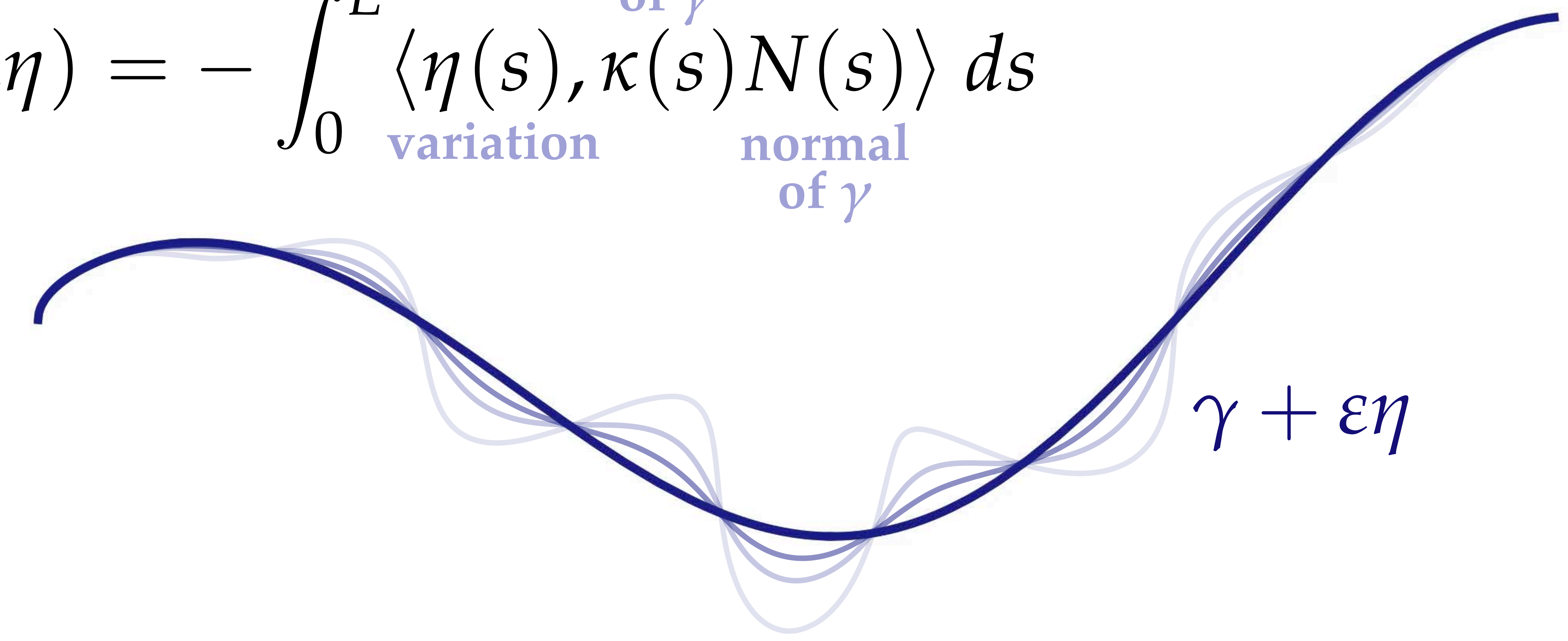
- More formally, consider an *arbitrary* change in the curve γ, given by a function $\eta : [0, L] \to \mathbb{R}^2$ with $\eta(0) = \eta(L) = 0$.



Length Variation

- More formally, consider an *arbitrary* change in the curve γ, given by a function $\eta : [0, L] \to \mathbb{R}^2$ with $\eta(0) = \eta(L) = 0$.
  Then

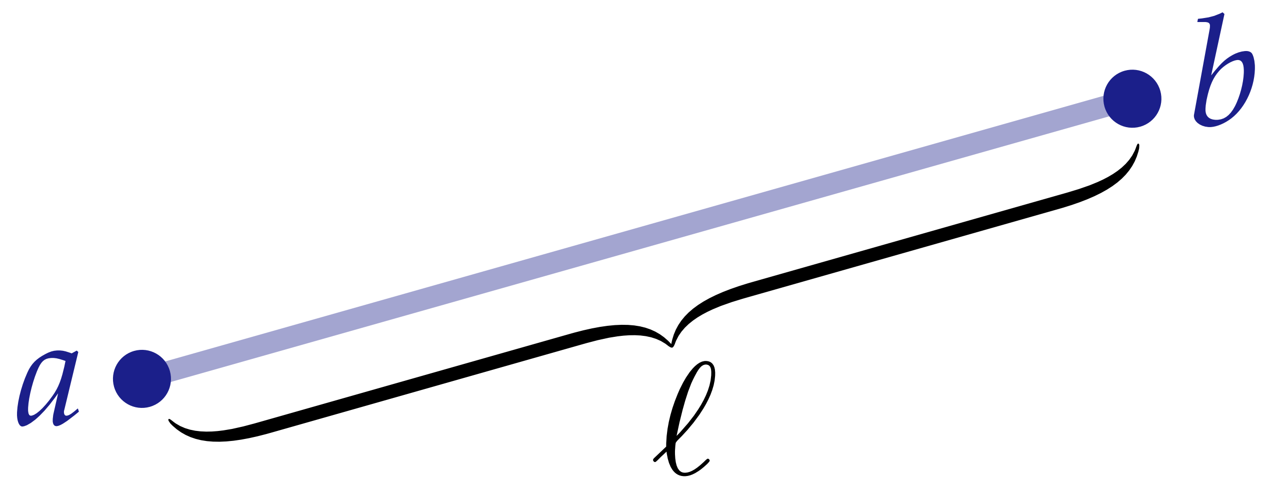
$$\frac{d}{d\varepsilon}\Big|_{\varepsilon=0} \mathrm{length}(\gamma + \varepsilon\eta) = -\int_0^L \langle \underbrace{\eta(s)}_{\text{variation}}, \overbrace{\kappa(s)}^{\text{curvature of } \gamma} \underbrace{N(s)}_{\text{normal of } \gamma} \rangle \, ds$$



- Therefore, the motion that most quickly decreases length is $\eta = \kappa N$.

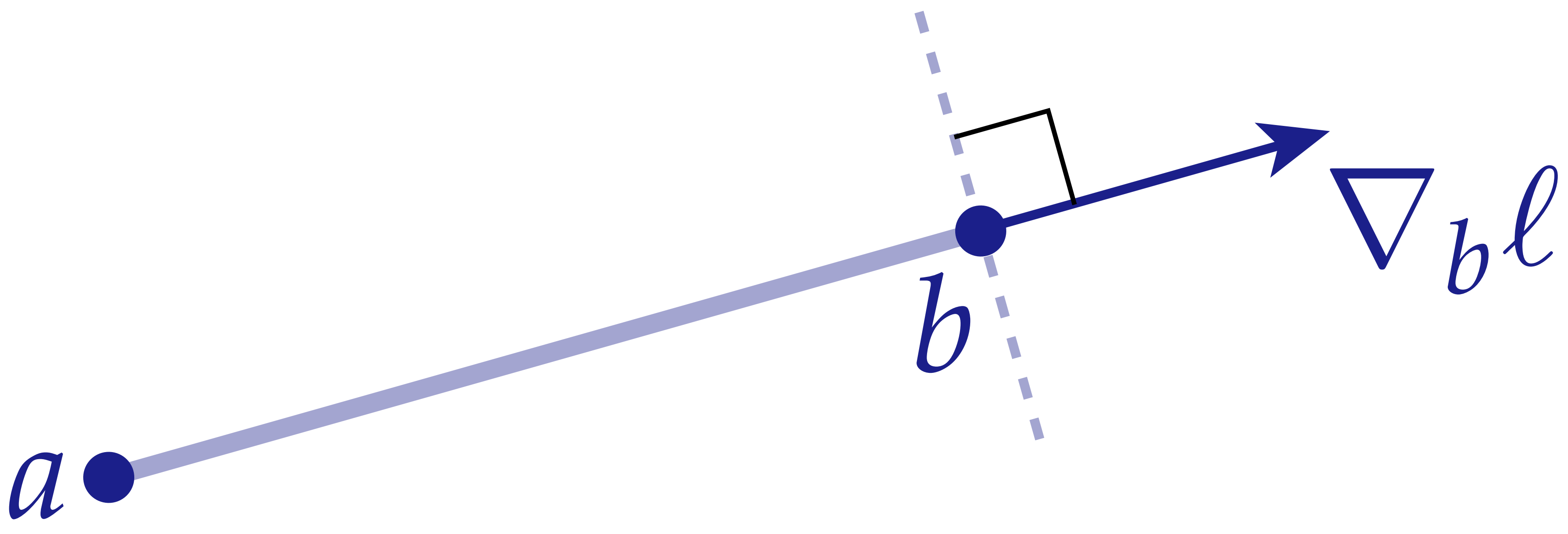
Gradient of Length for a Line Segment

- This all becomes much easier in the discrete setting: just take the gradient of length with respect to vertex positions.
- First, a warm-up exercise. Suppose we have a *single* line segment:



$$\ell := |b - a|$$

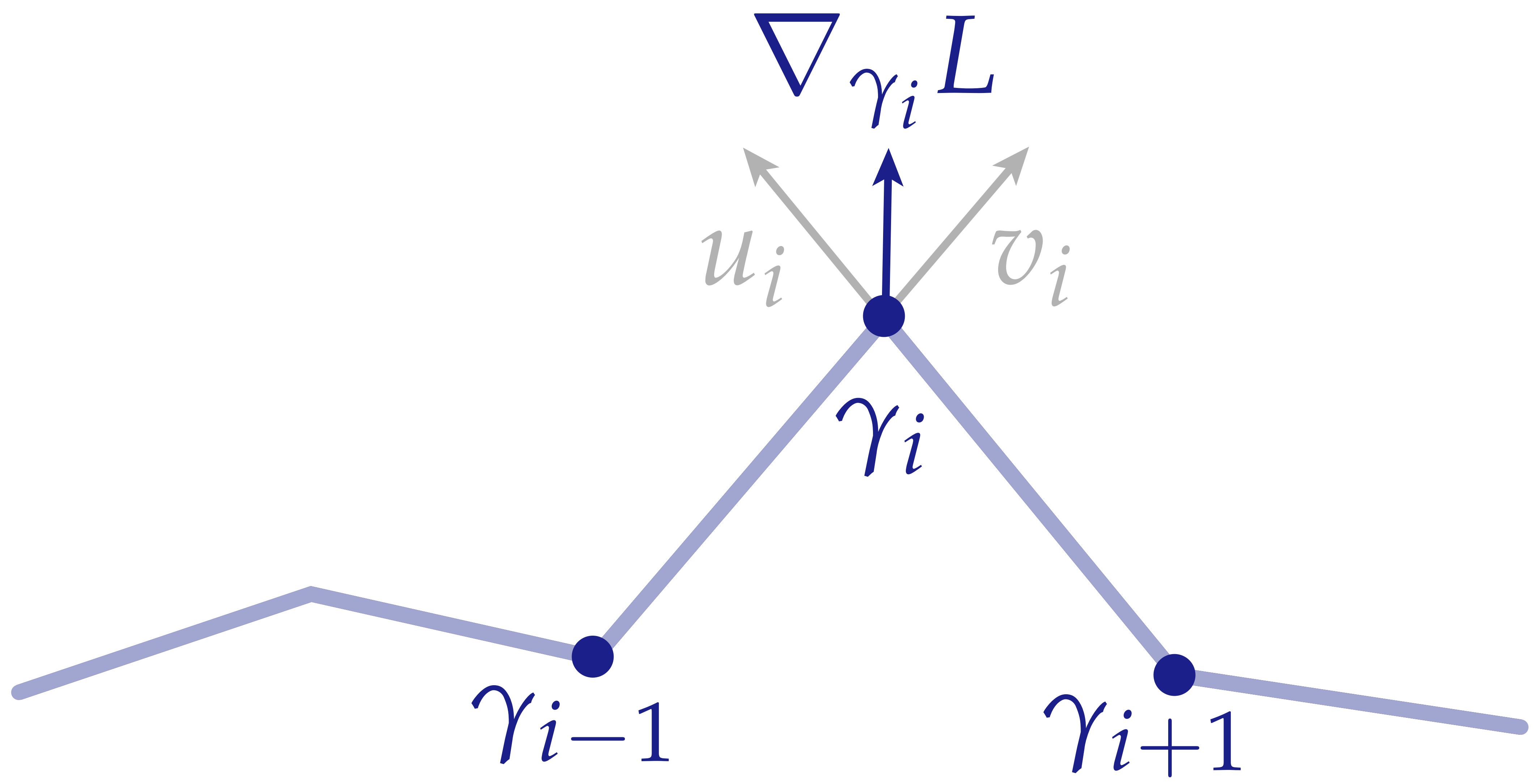
- Which motion of b most quickly increases this length?

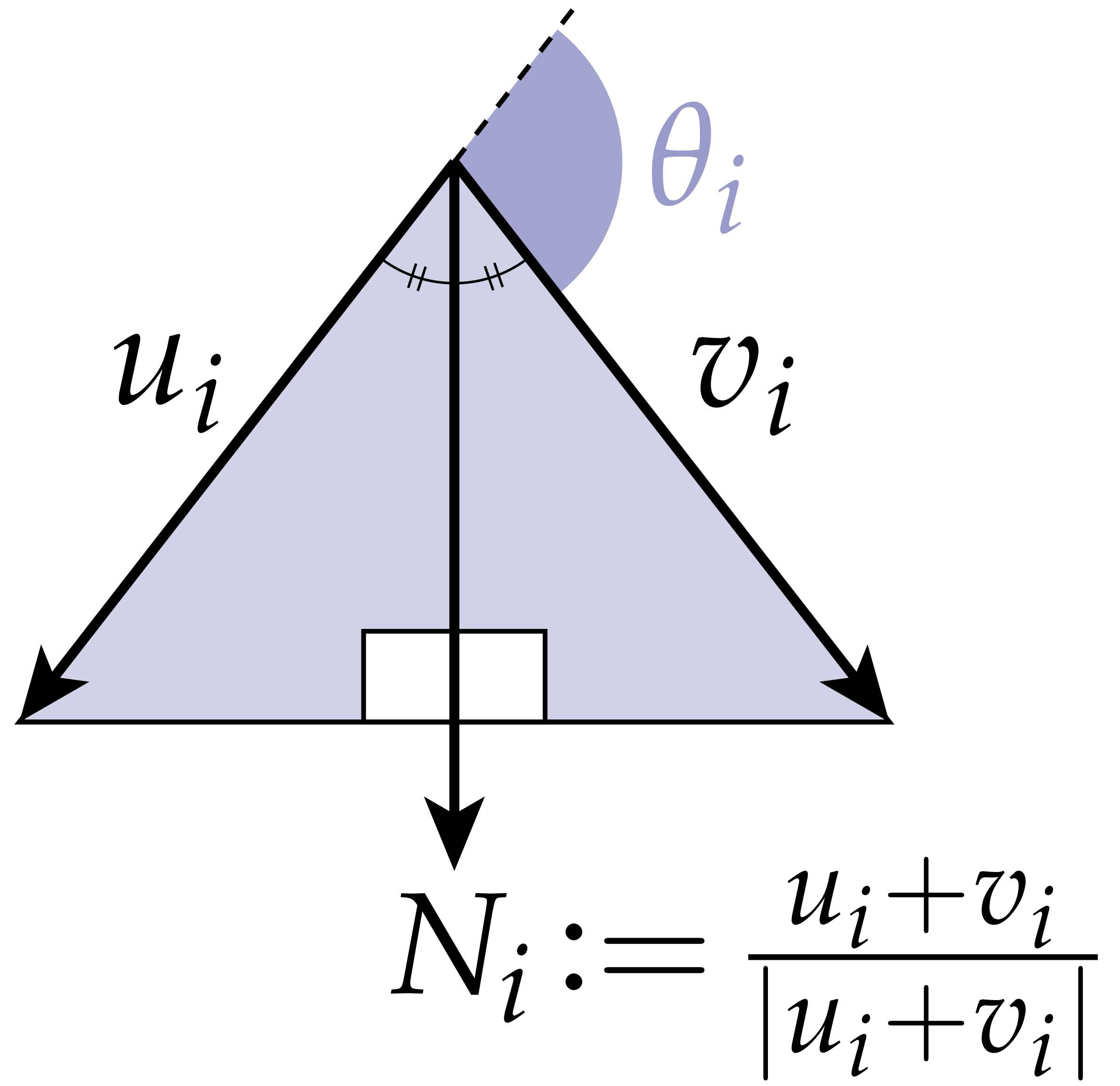


$$\nabla_b \ell = (b - a)/\ell$$

Gradient of Length for a Discrete Curve

- To find the motion that most quickly increases the *total* length L, we now just sum the contributions of each segment:



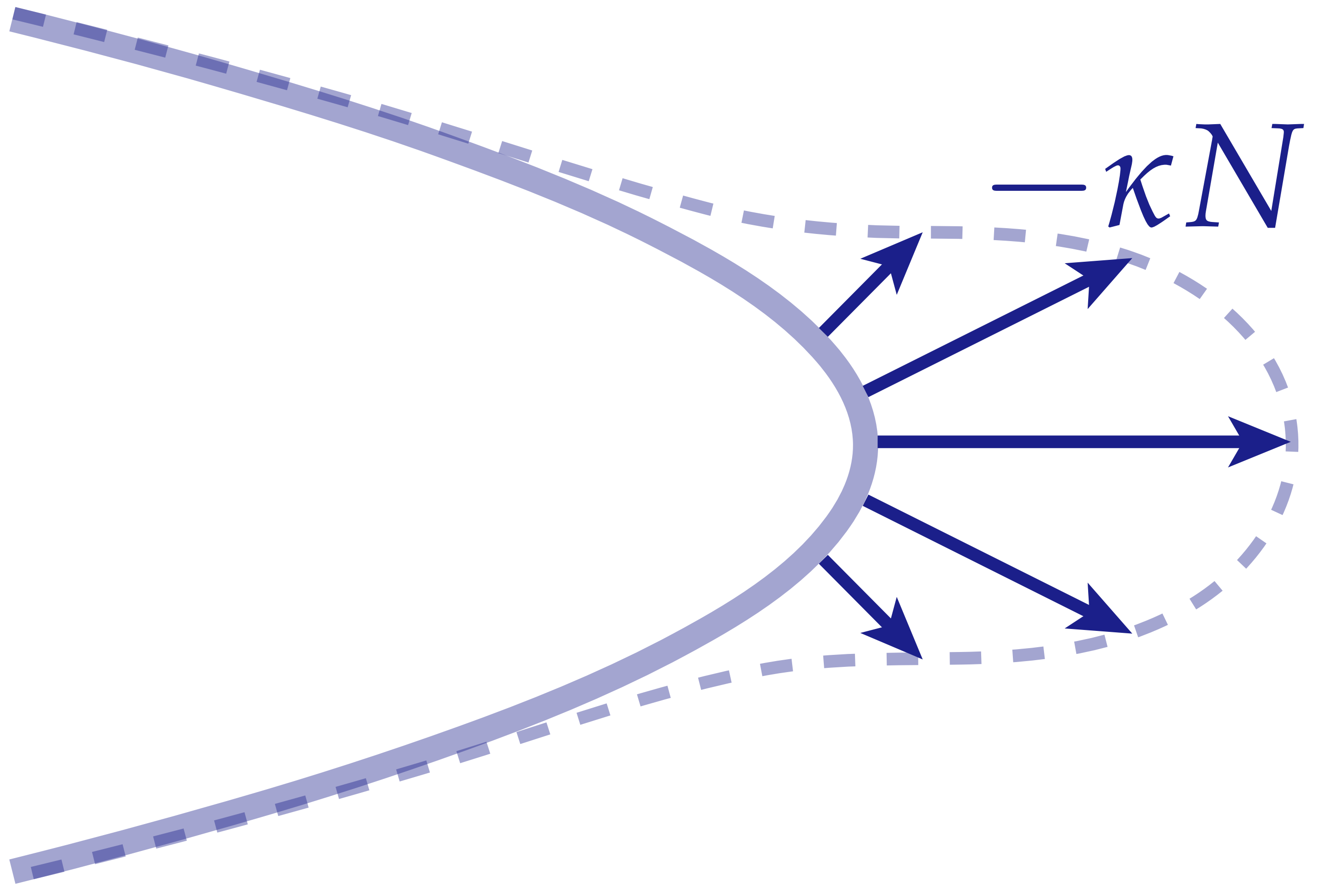


- Using some simple trigonometry, we can also express the length gradient in terms of the exterior angle θ_i and the angle bisector N_i:

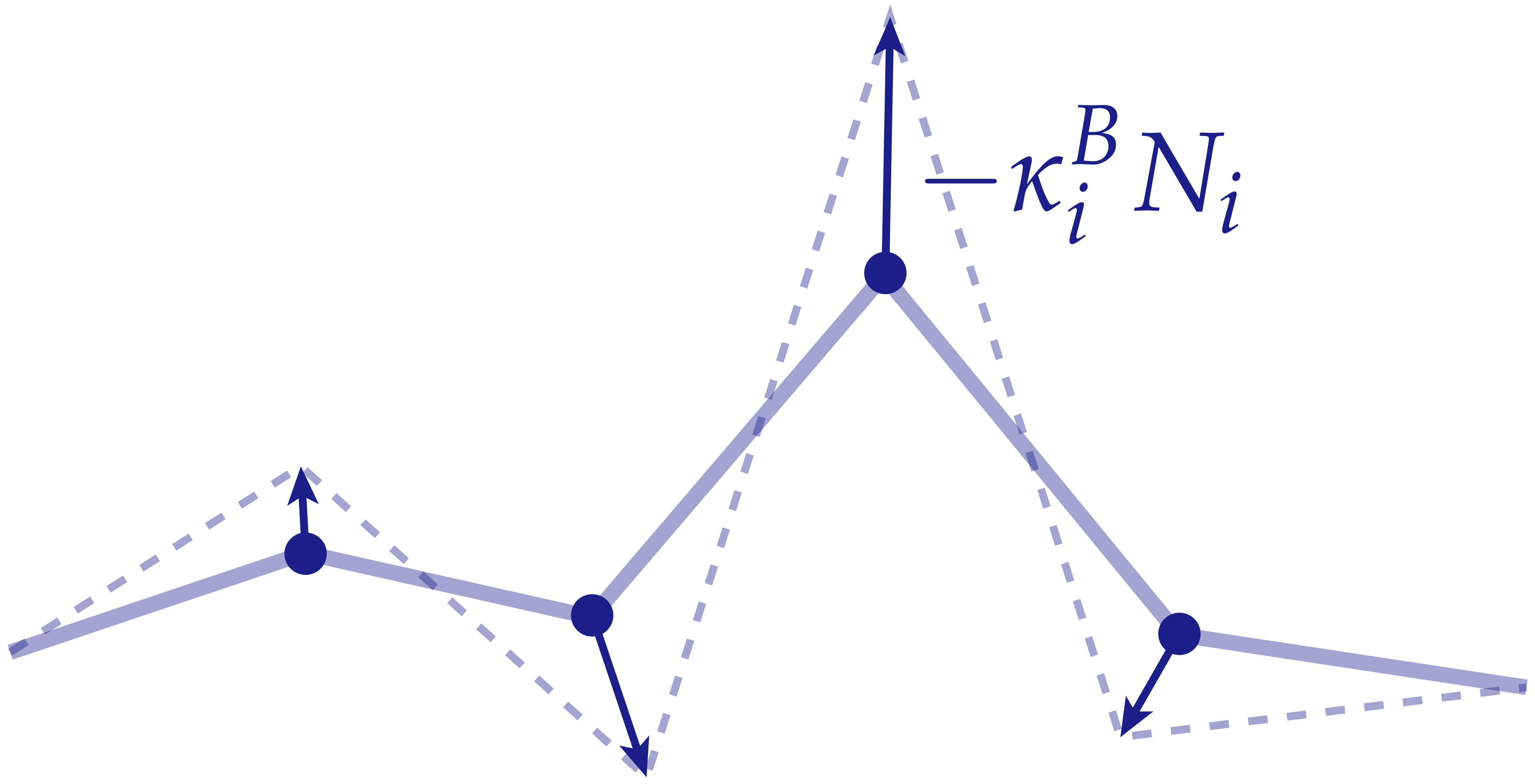
$$\nabla_{\gamma_i} L = 2 \sin(\theta_i / 2) N_i$$

Discrete Curvature (Length Variation)

- How does this help us define discrete curvature?
- Recall that in the smooth setting, the gradient of length is equal to the curvature times the normal.
- Hence, our expression for the *discrete* length variation provides a definition for the *discrete* curvature times the *discrete* normal.

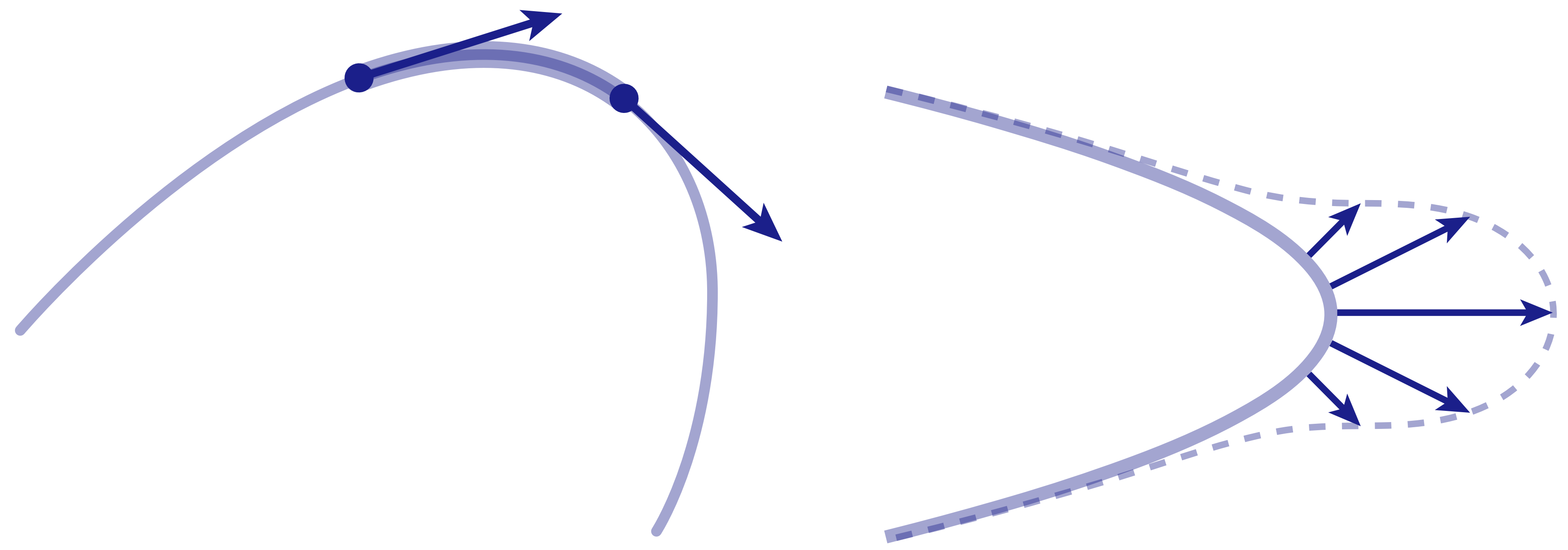


$$\boxed{\begin{array}{c}\kappa_i^B N_i := 2\sin(\theta_i/2)N_i \\ \textbf{(length variation)}\end{array}}$$



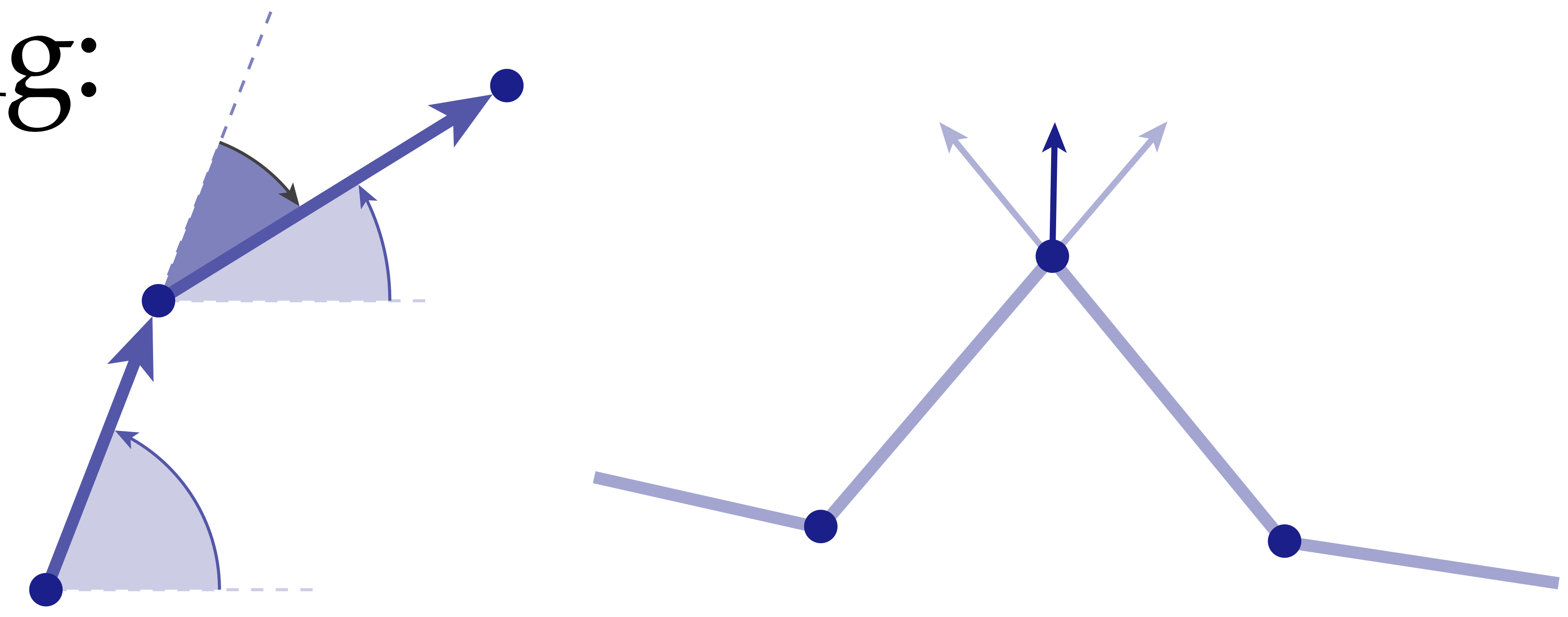
A Tale of Two Curvatures

- To recap what we've done so far: we considered two **equivalent** definitions in the smooth setting:



  1. turning angle
  2. length variation

- These perspectives led to two **inequivalent** definitions of curvature in the discrete setting:



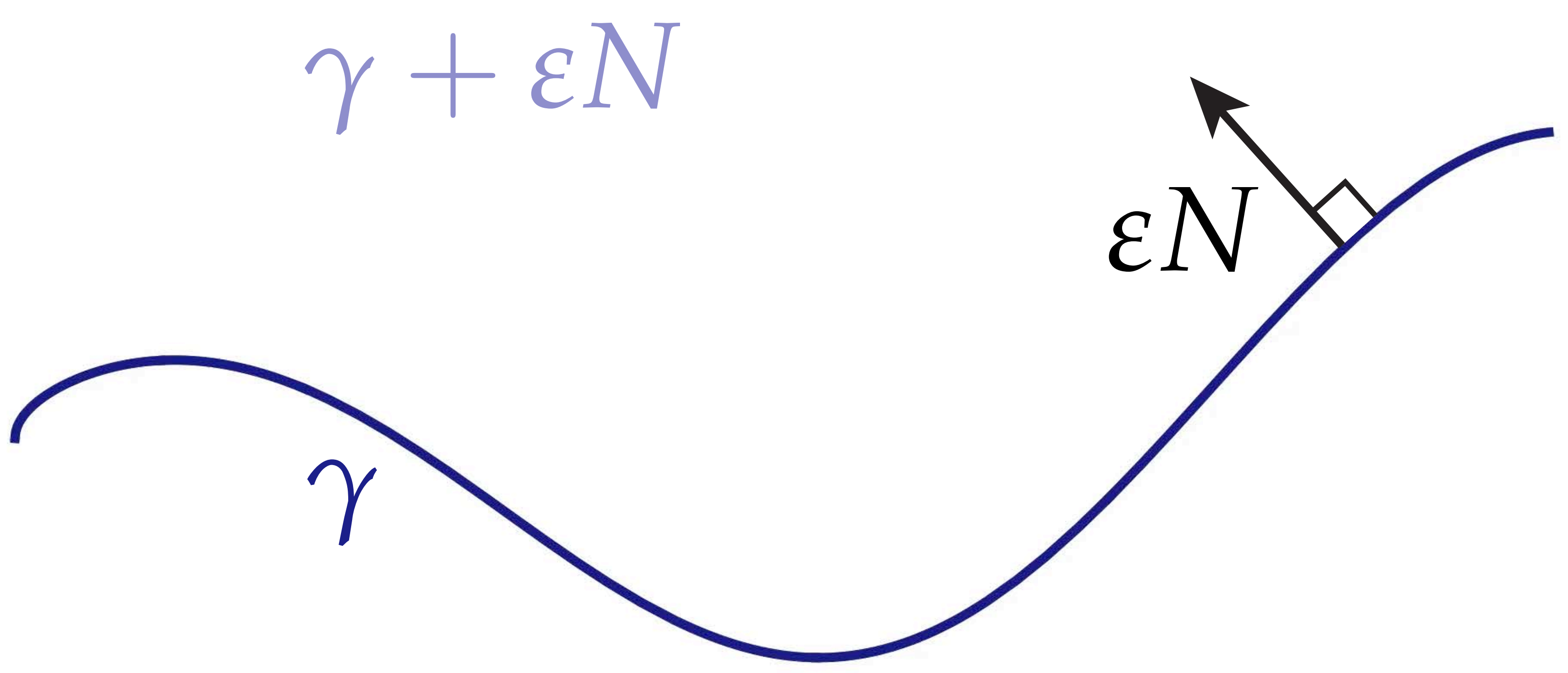
  1. $\kappa_i^A := \theta_i$
  2. $\kappa_i^B := 2\sin(\theta_i/2)$

- For *small* angles, both definitions agree ($\sin(\varepsilon) \approx \varepsilon$).
- Is one "better"? Are there more possibilities? Let's keep going…

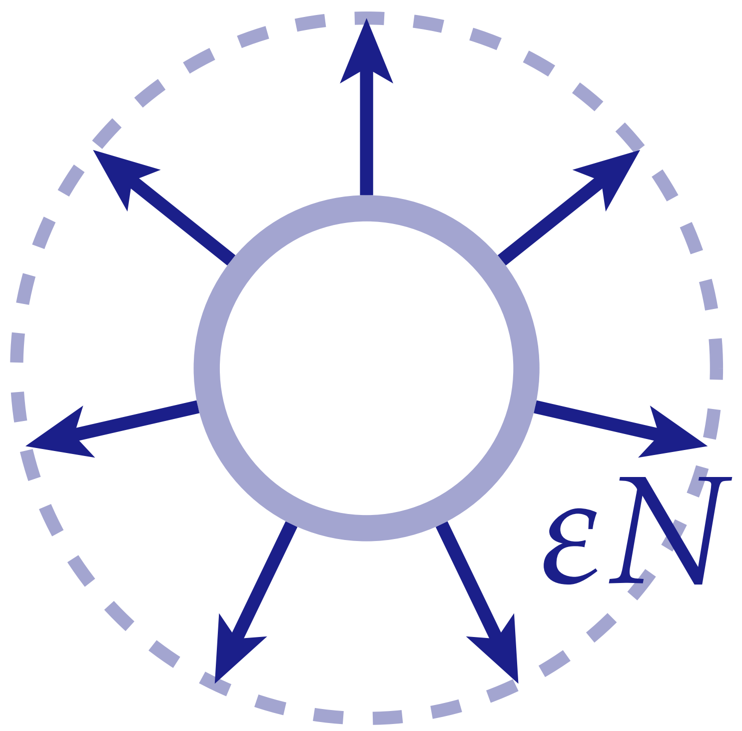
Steiner Formula

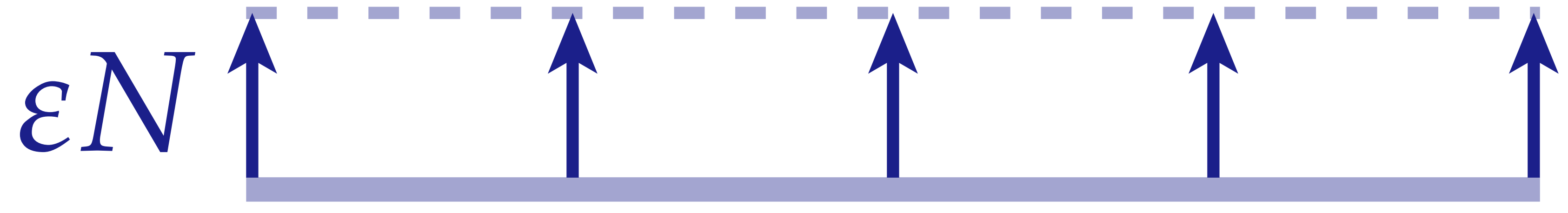
- Steiner's formula is closely related to our last approach: it says that if we move at a *constant* speed in the normal direction, then the change in length is proportional to curvature:

$$\text{length}(\gamma+\varepsilon N) = \text{length}(\gamma) - \varepsilon \int_0^L \kappa(s)\, ds$$



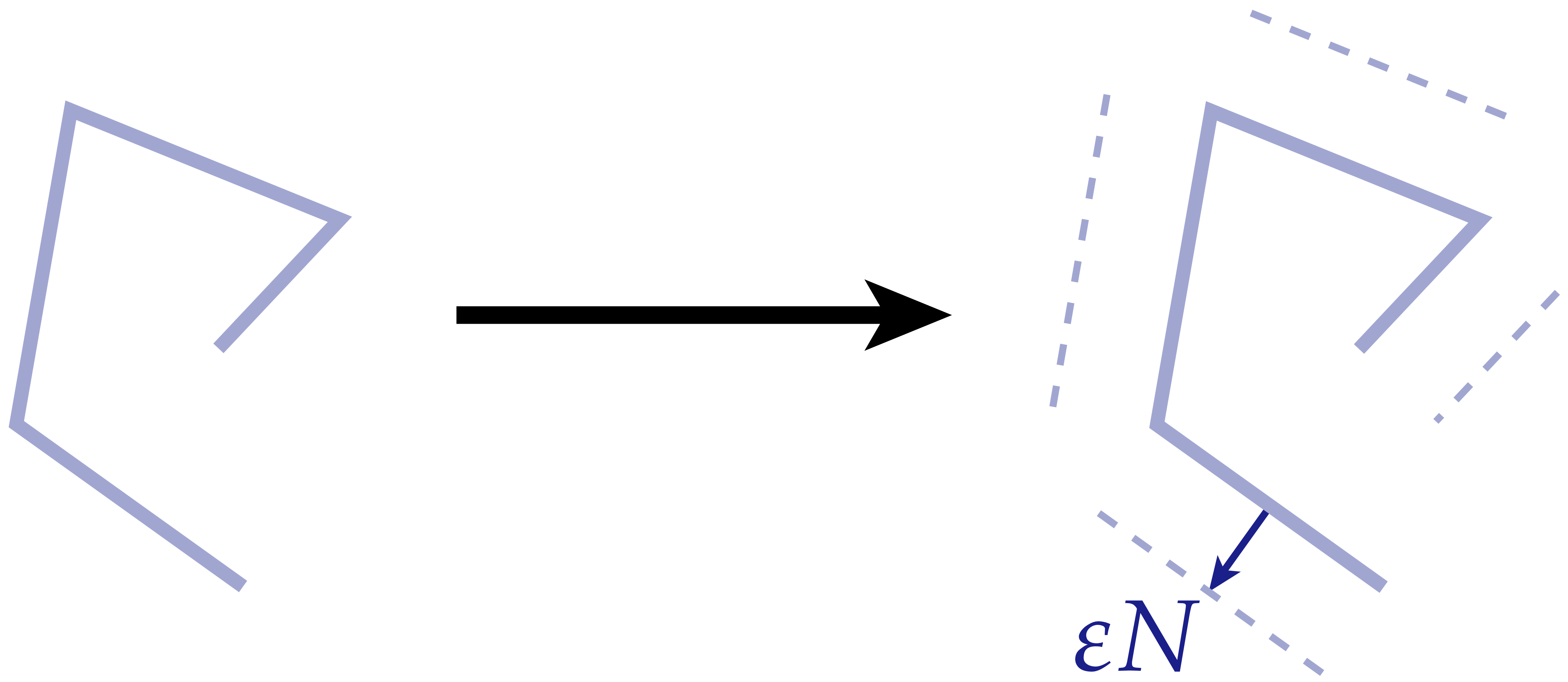
- The intuition is the same as before: for a constant-distance normal offset, length will change in curved regions but not flat regions:





Discrete Normal Offsets

- How do we apply normal offsets in the discrete case?
- The first problem is that *normals* are not defined at vertices!
- We can at very least offset individual edges along their normals:

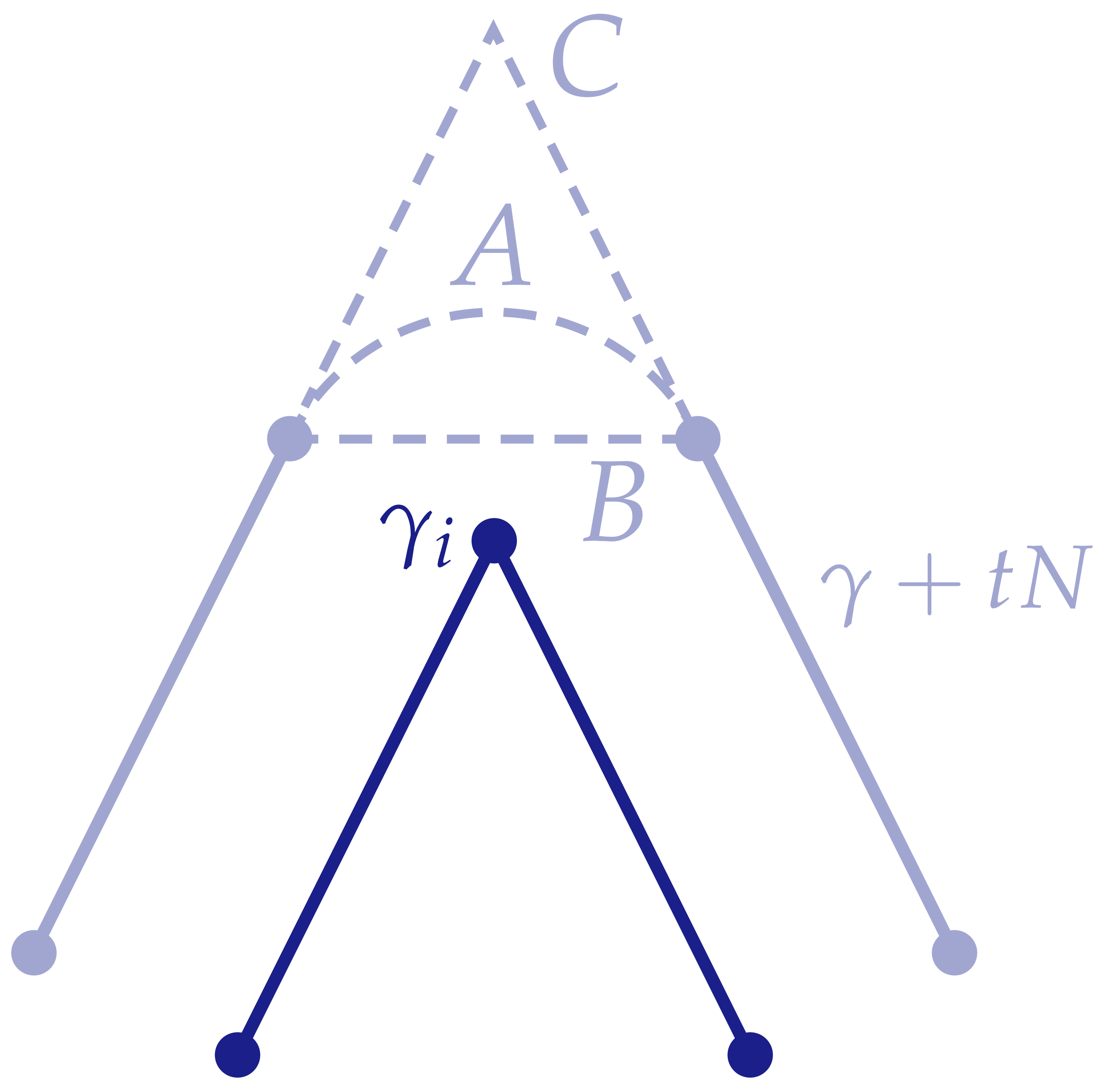


- Question: how should we connect the normal-offset segments to get the final normal-offset curve?

Discrete Normal Offsets

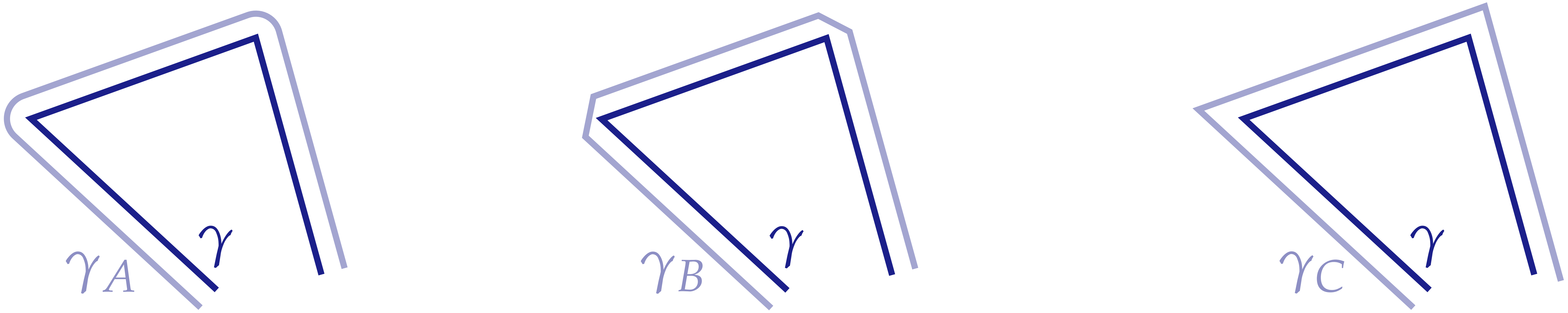
- There are several natural ways to connect offset segments:
  - (A) along a circular arc of radius ε
  - (B) along a straight line
  - (C) extend edges until they intersect
- If we now compute the total length of the connected curves, we get (after some work…):

$$\begin{aligned}
\text{length}_A &= \text{length}(\gamma) - \varepsilon \textstyle\sum_i \theta_i \\
\text{length}_B &= \text{length}(\gamma) - \varepsilon \textstyle\sum_i 2\sin(\theta_i/2) \\
\text{length}_C &= \text{length}(\gamma) - \varepsilon \textstyle\sum_i 2\tan(\theta_i/2)
\end{aligned}$$



Discrete Curvature (Steiner Formula)

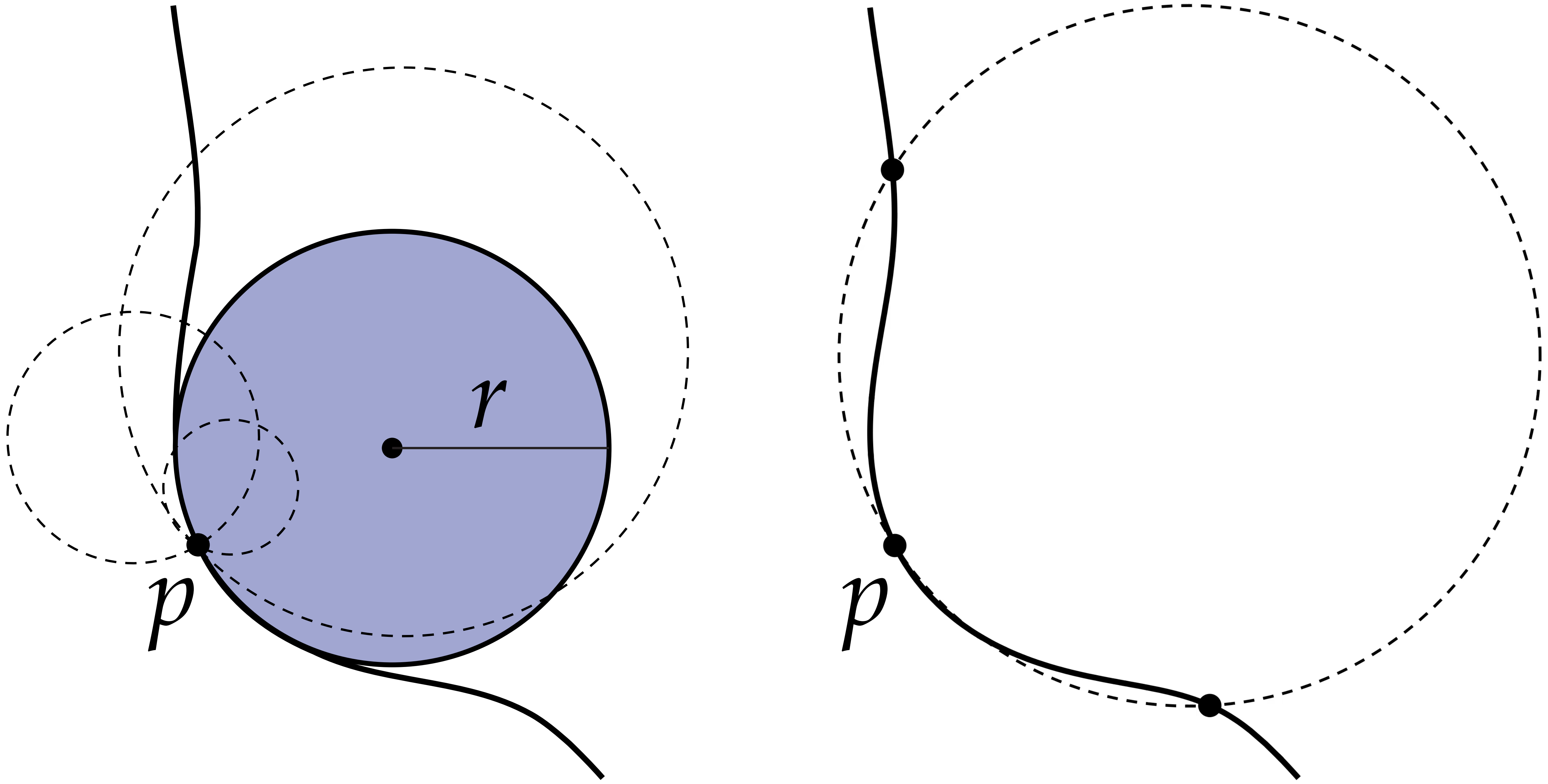
- Steiner's formula says change in length is proportional to curvature
- Hence, we get yet another definition for curvature by comparing the original and normal-offset lengths.
- In fact, we get *three* definitions—two we've seen and one we haven't:



$\kappa_i^A := \theta_i$ $\qquad$ $\kappa_i^B := 2\sin(\theta_i/2)$ $\qquad$ $\kappa_i^C := 2\tan(\theta_i/2)$

Osculating Circle

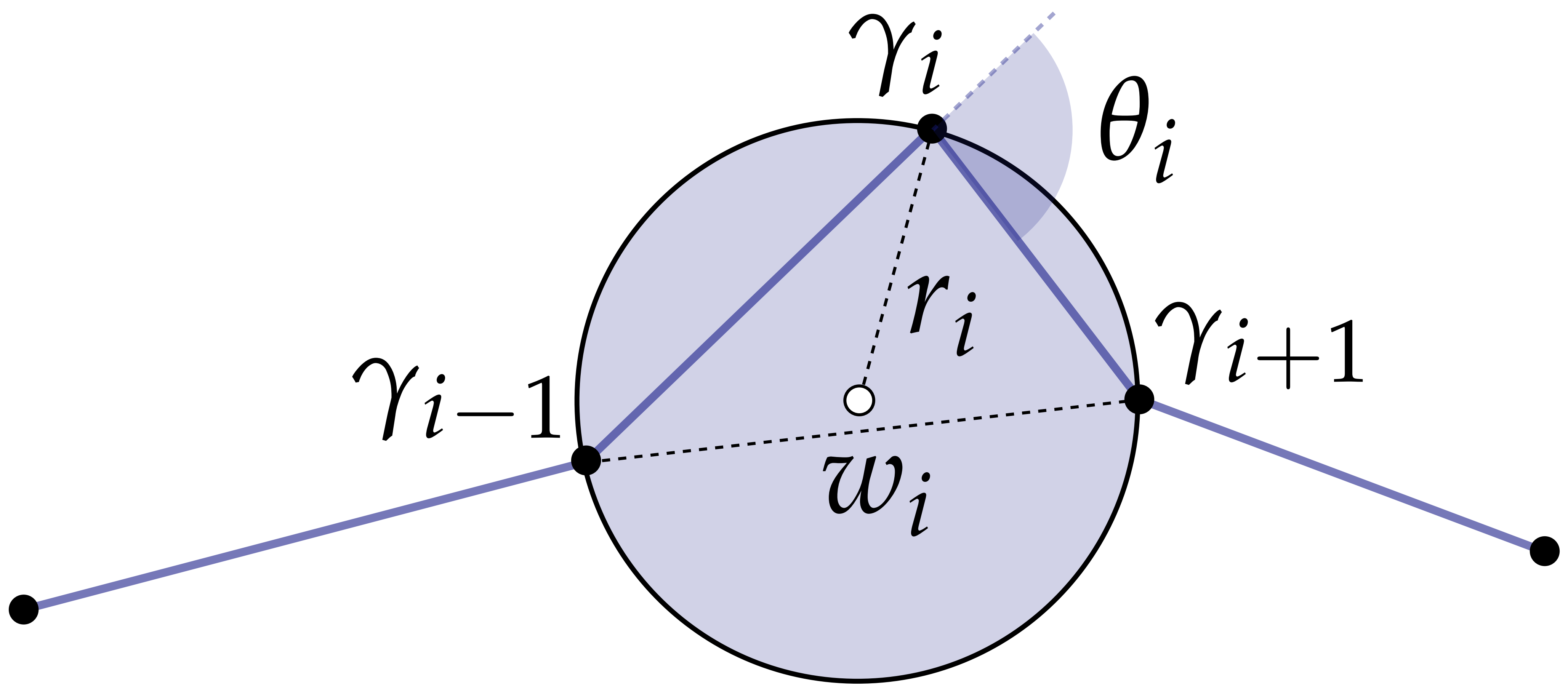
- One final idea is to consider the **osculating circle**, which is the circle that best approximates a curve at a point p



- More precisely, if we consider a circle passing through p and two equidistant neighbors to the "left" and "right" (resp.), the osculating circle is the limiting circle as these neighbors approach p.
- The curvature is then the reciprocal of the radius: $\kappa(p) = \dfrac{1}{r(p)}$

Discrete Curvature (Osculating Circle)

- A natural idea, then, is to consider the *circumcircle* passing through three consecutive vertices of a discrete curve:



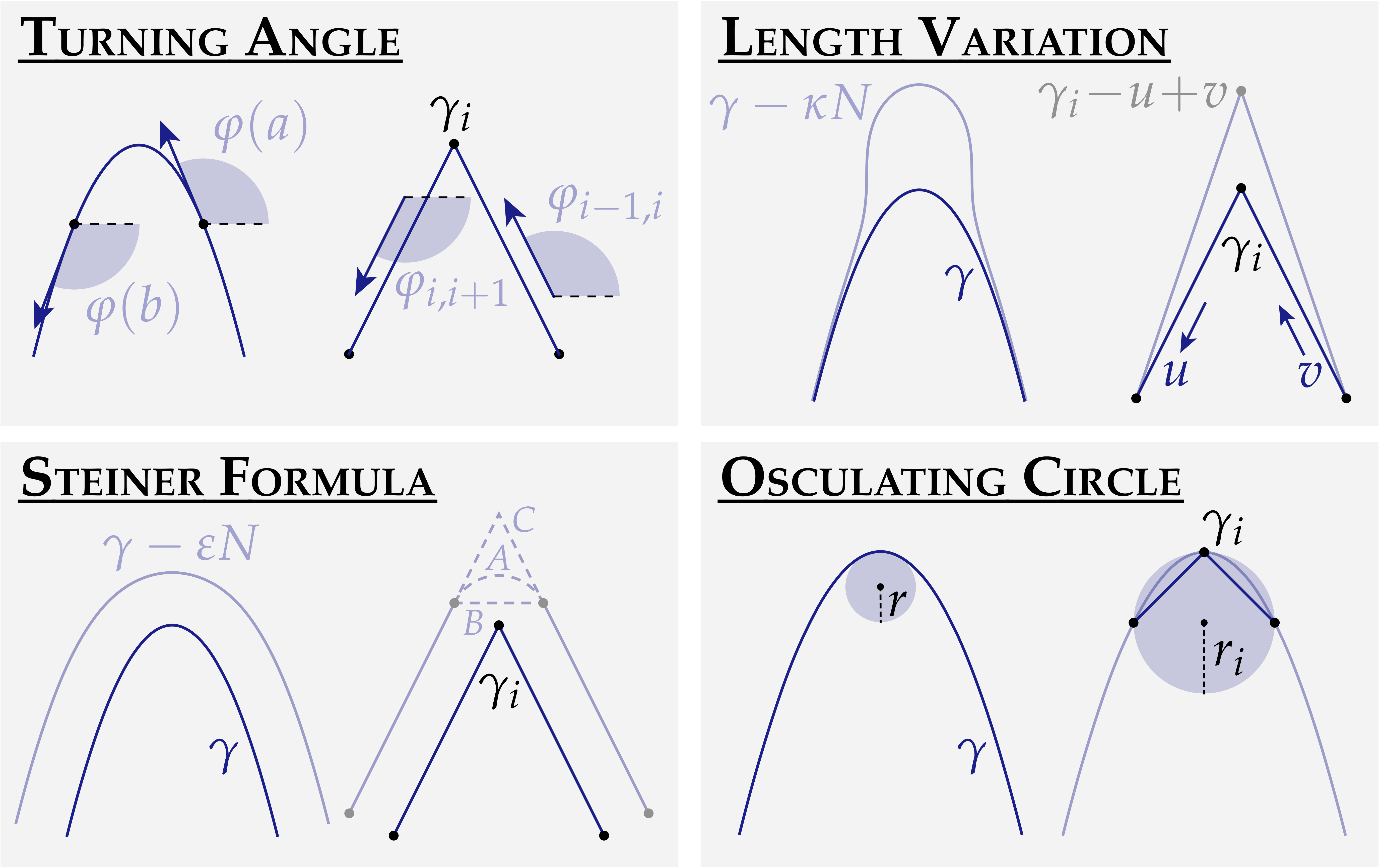
$$w_i := |\gamma_{i+1} - \gamma_{i-1}|$$

- Our *fourth* discrete curvature is then the reciprocal of the radius:

$$\kappa_i^D := \frac{1}{r_i} = 2\sin(\theta_i)/w_i$$

A Tale of Four Curvatures

- Starting with four **equivalent** definitions of smooth curvature, we ended up with four **inequivalent** definitions for discrete curvature:

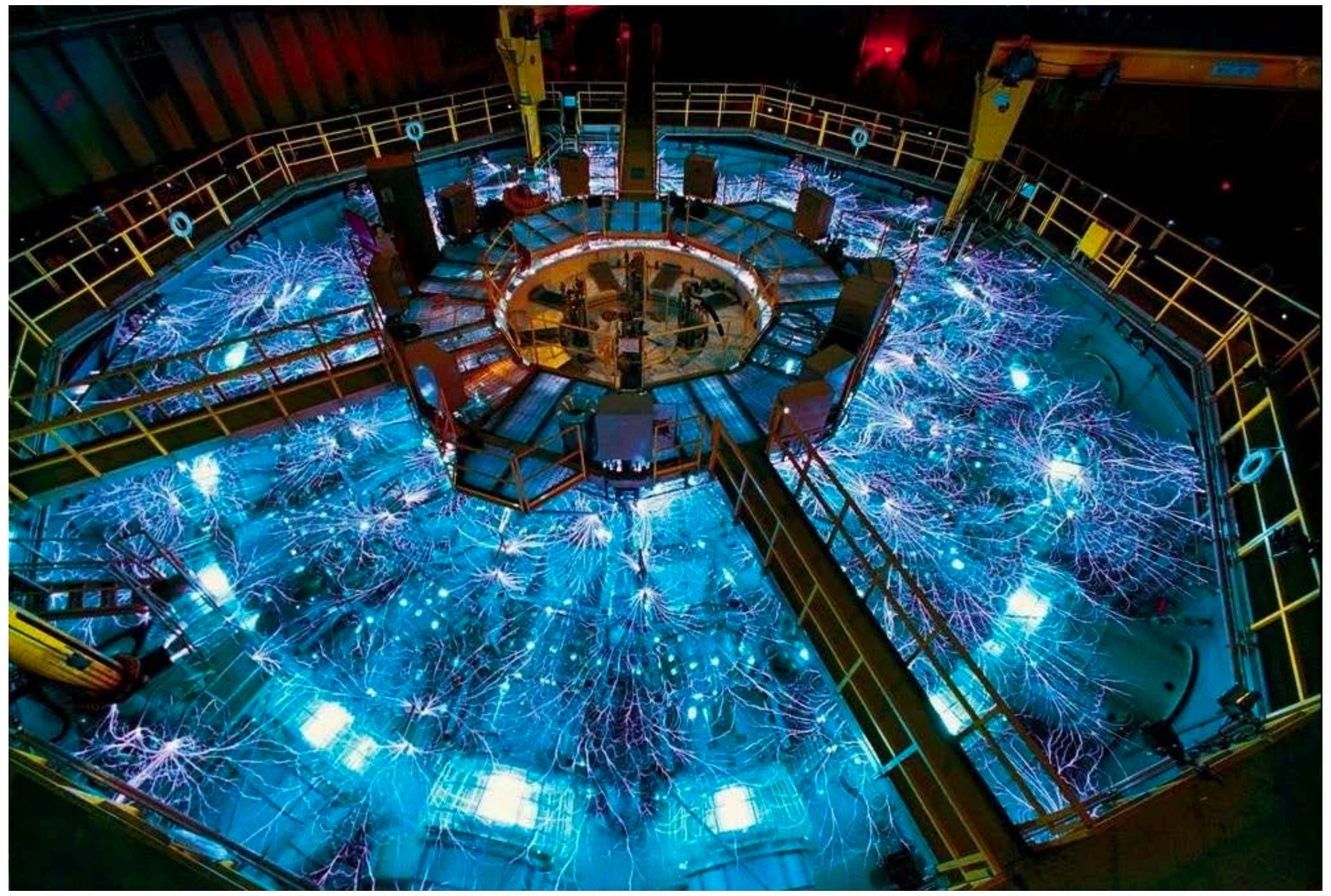


So… *which one should we use?*

Pick the Right Tool for the Job!

- **Answer:** pick the right tool for the job!
- For a given application, which properties are most important to us? How much computation are we willing to do? *Etc.*
- *E.g.*, for one physical simulation you might care most about energy; for another you might care about vorticity.
- What kind of trade offs do we have in geometric problems?

image: Sandia National Laboratories



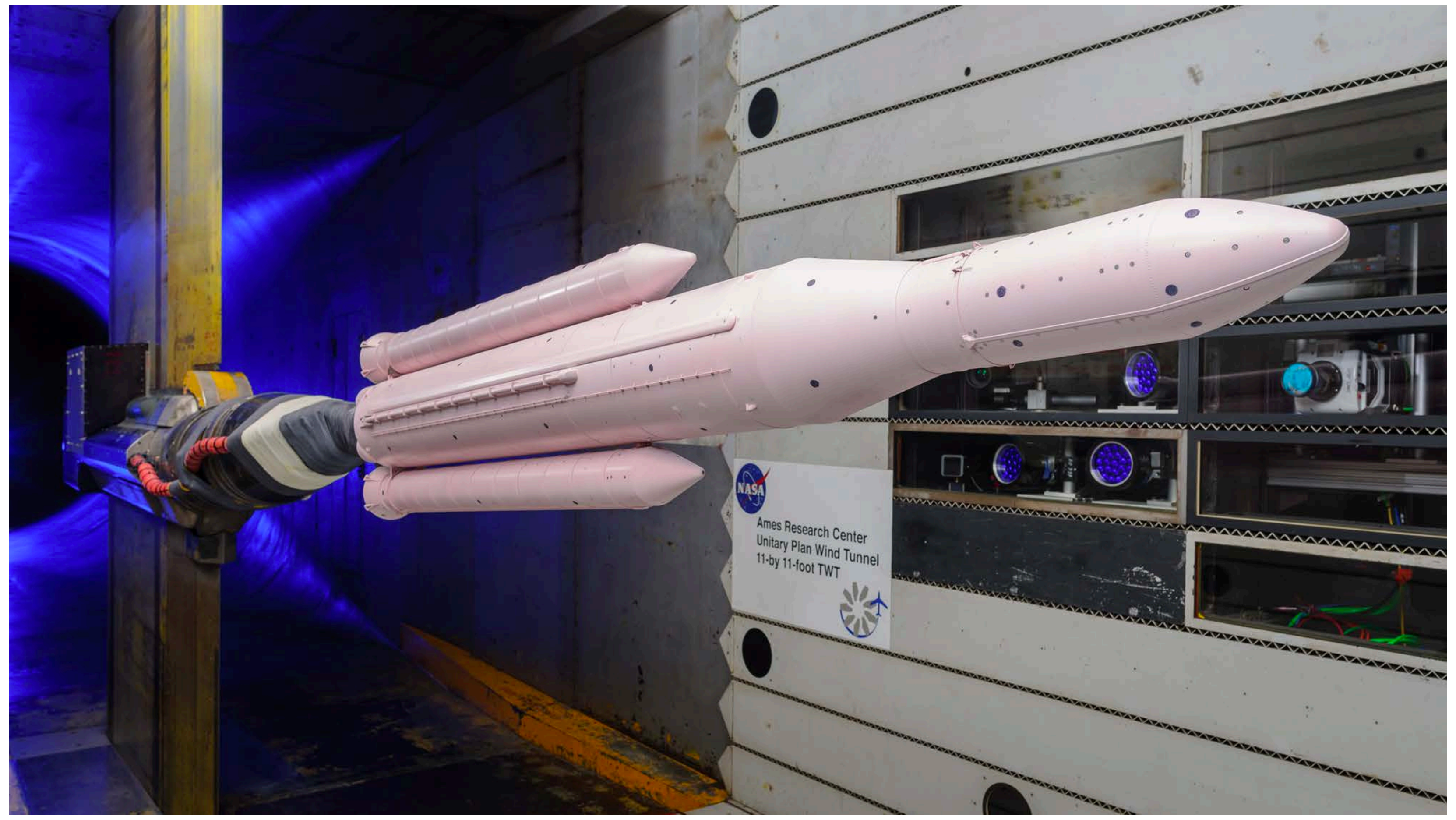
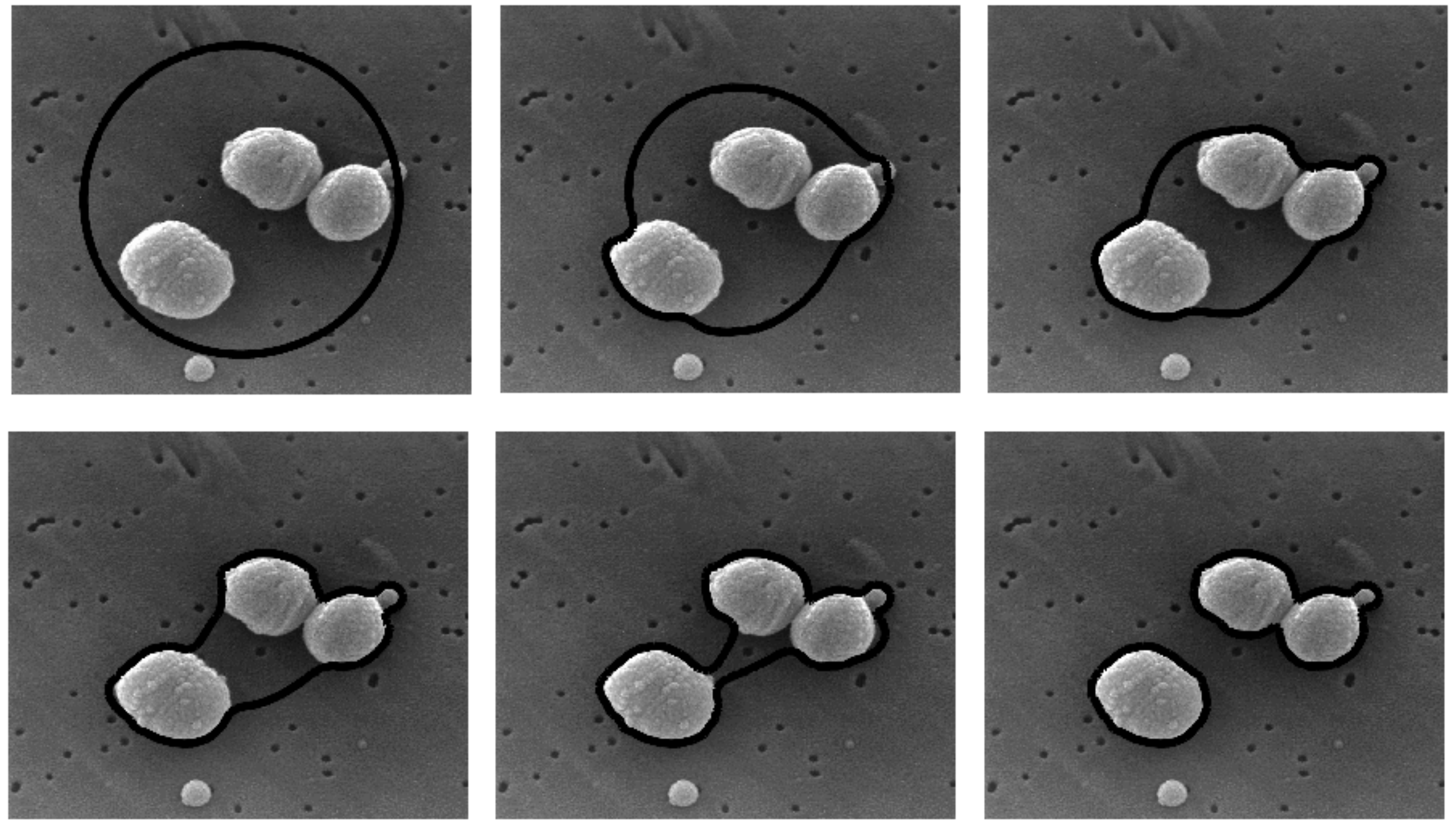
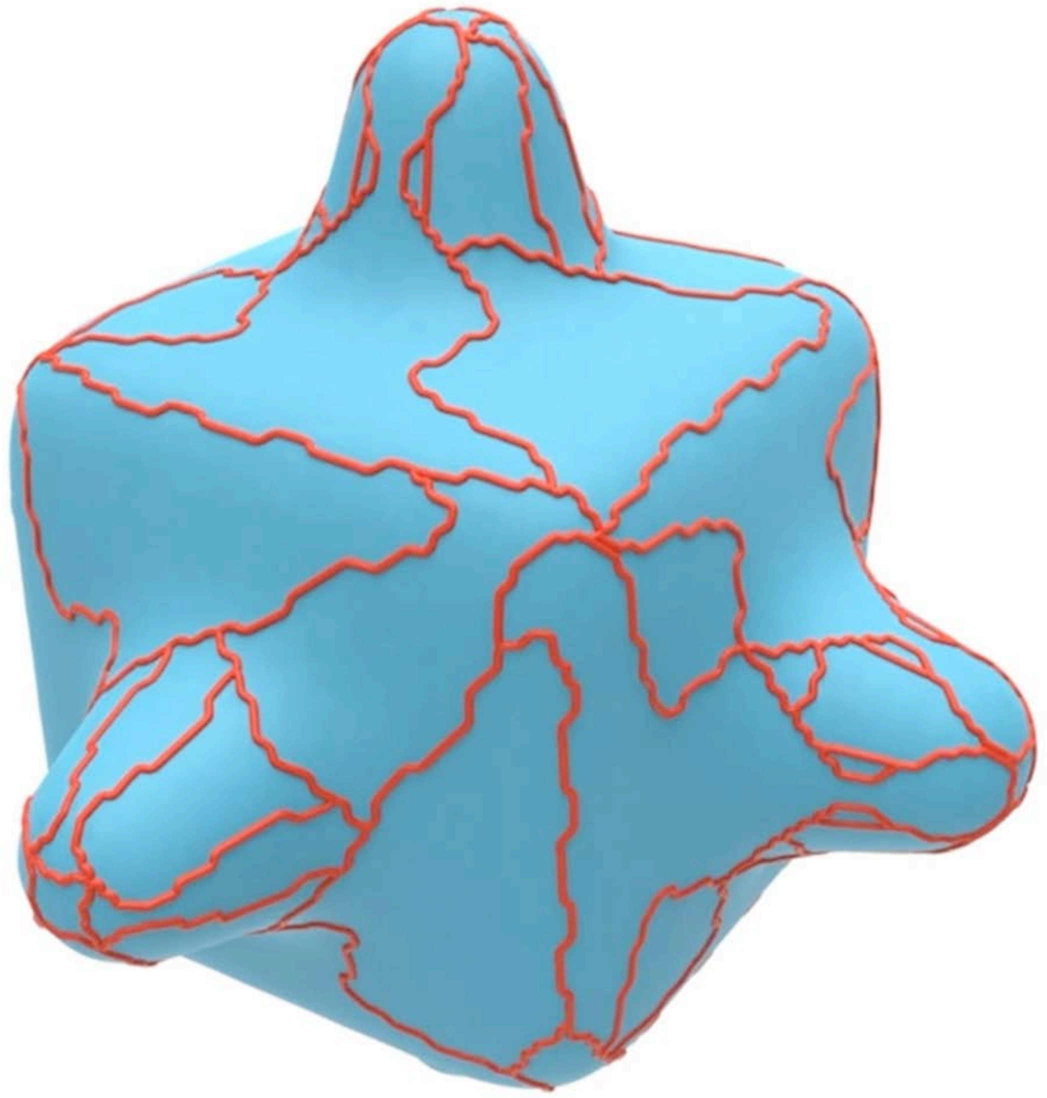
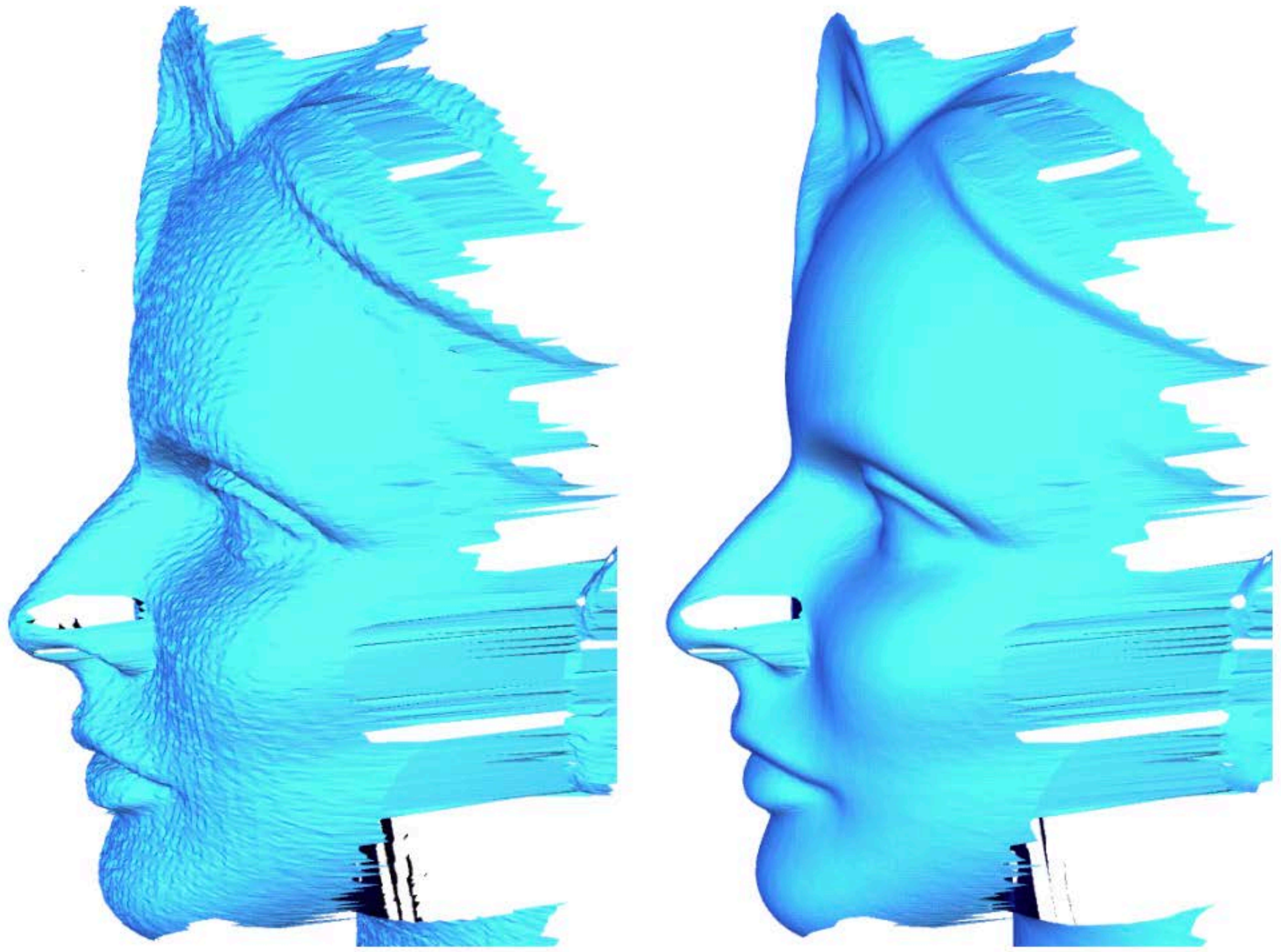


image: NASA Ames/Dominic Hart

Curvature Flow

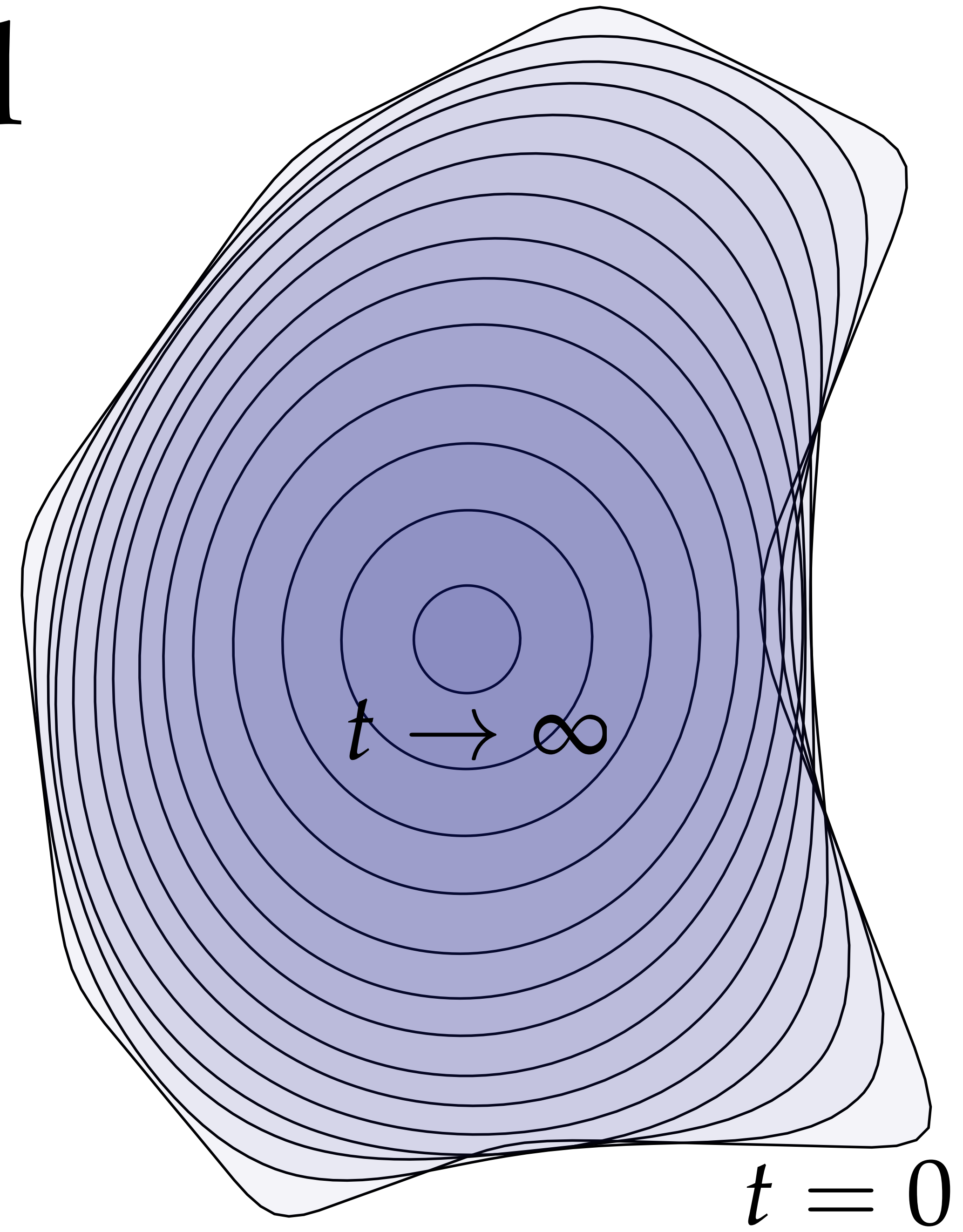






Toy Example: Curve Shortening Flow

- A simple version is *curve shortening flow*, where a closed curve moves in the normal direction with speed proportional to curvature:



$$\tfrac{d}{dt}\gamma(s,t) = \kappa(s,t)N(s,t)$$

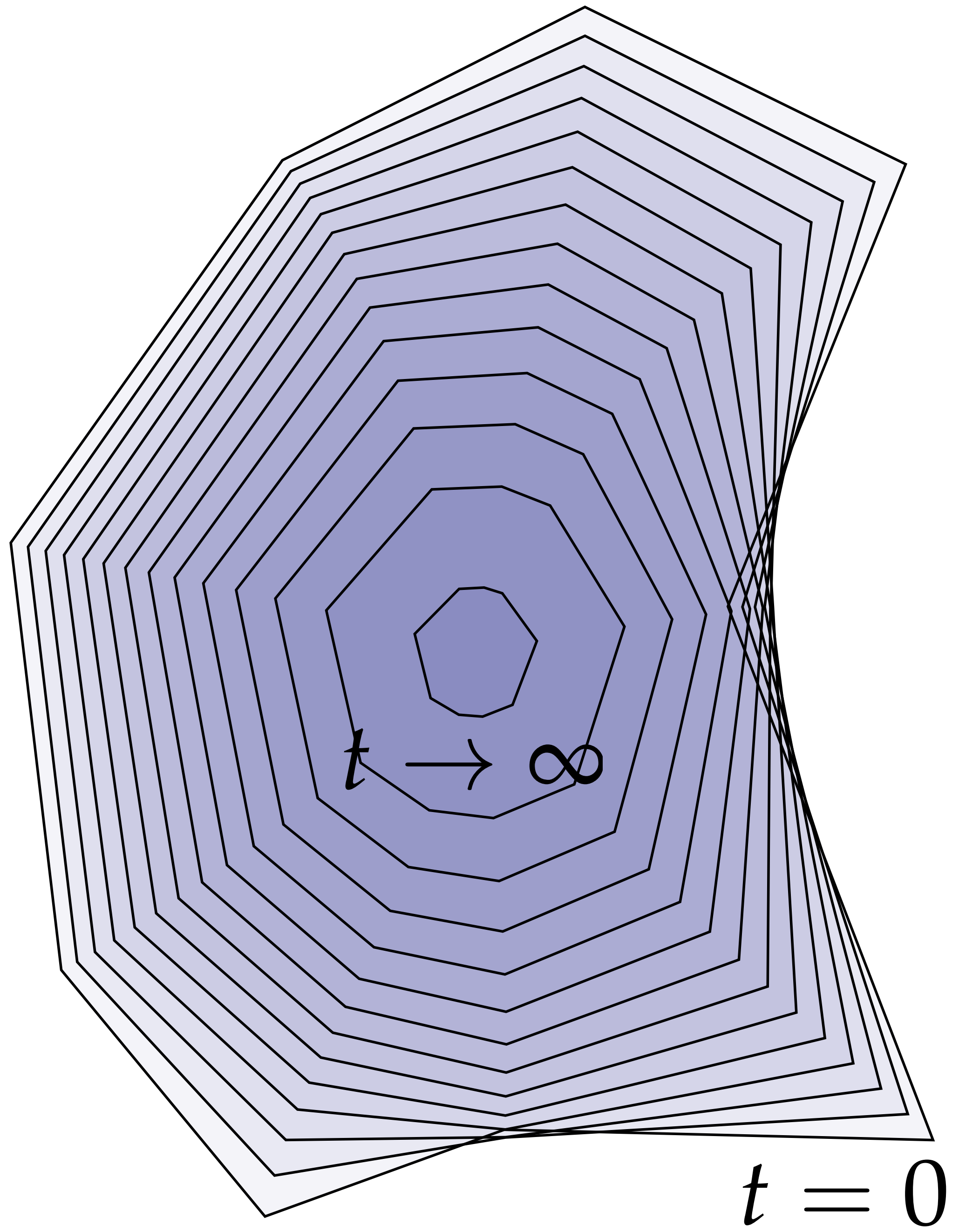
- Some key properties:
  - (**TOTAL**) Total curvature remains constant throughout the flow.
  - (**DRIFT**) The center of mass does not drift from the origin.
  - (**ROUND**) Up to rescaling, the flow is stationary for circular curves.

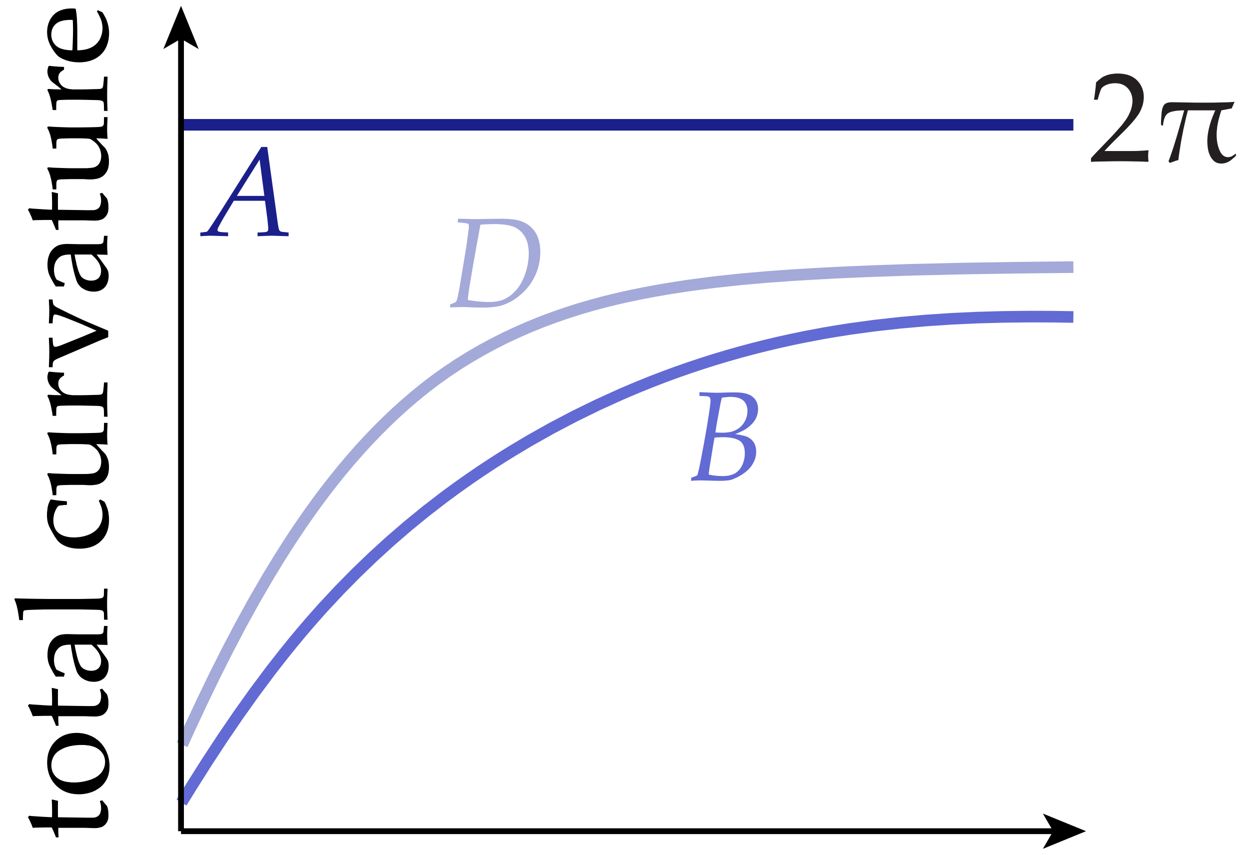
Discrete Curvature Flow—No Free Lunch

- We can approximate curvature flow by repeatedly moving each vertex a little bit in the direction of the discrete curvature normal:

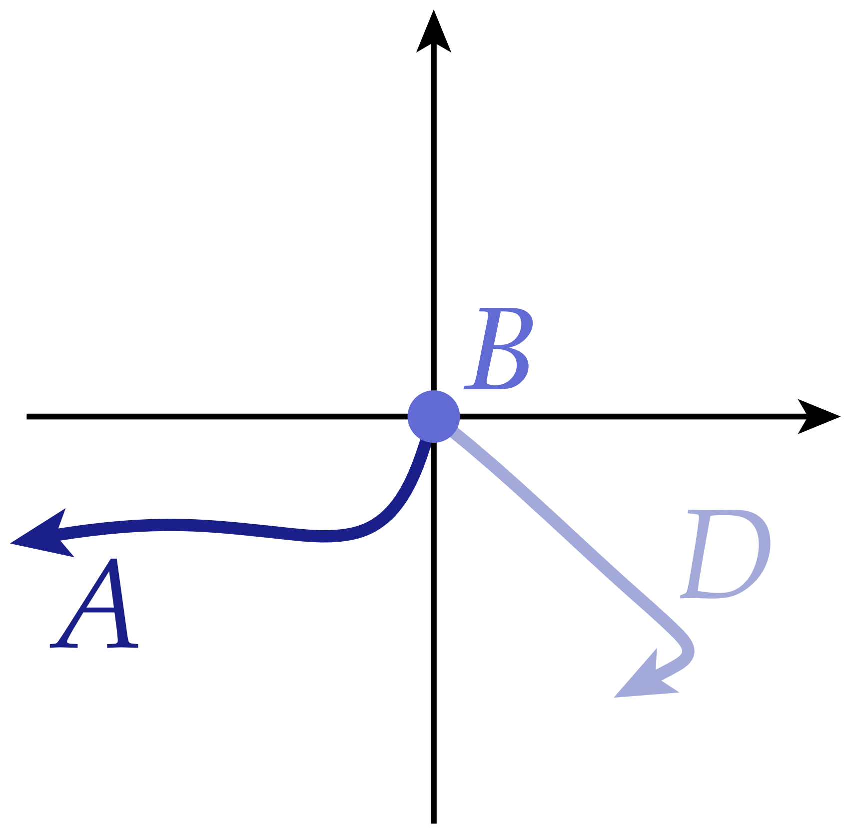
$$\gamma_i^{t+1} = \gamma_i^t + \tau\kappa_i N_i$$

- But **no** choice of discrete curvature simultaneously captures all three properties of the smooth flow*:

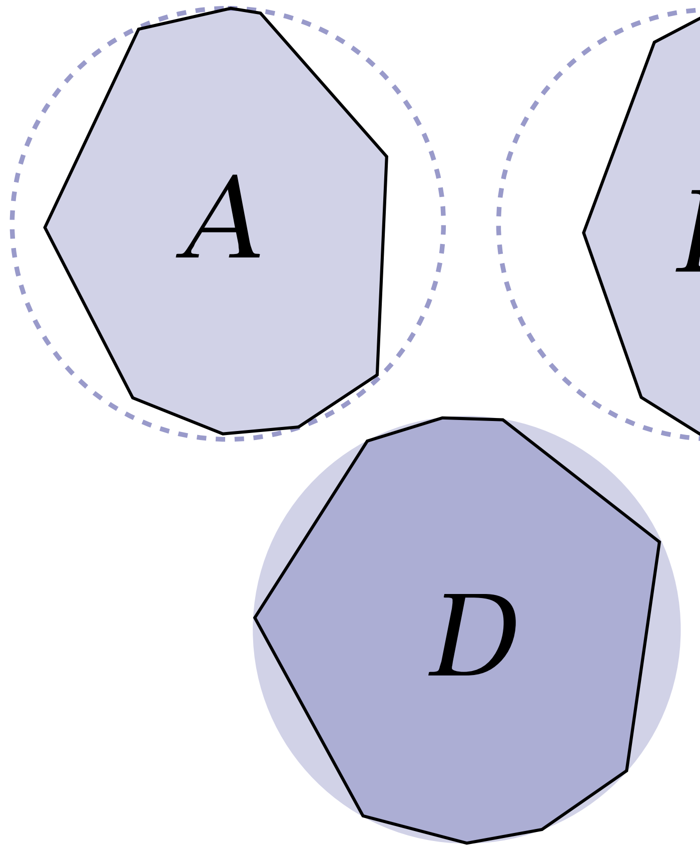




time



drift from center



roundness

	Total	Drift	Round
κ^A	✓	×	×
κ^B	×	✓	×
κ^D	×	×	✓

*In fact, it's impossible!

No Free Lunch—Other Examples

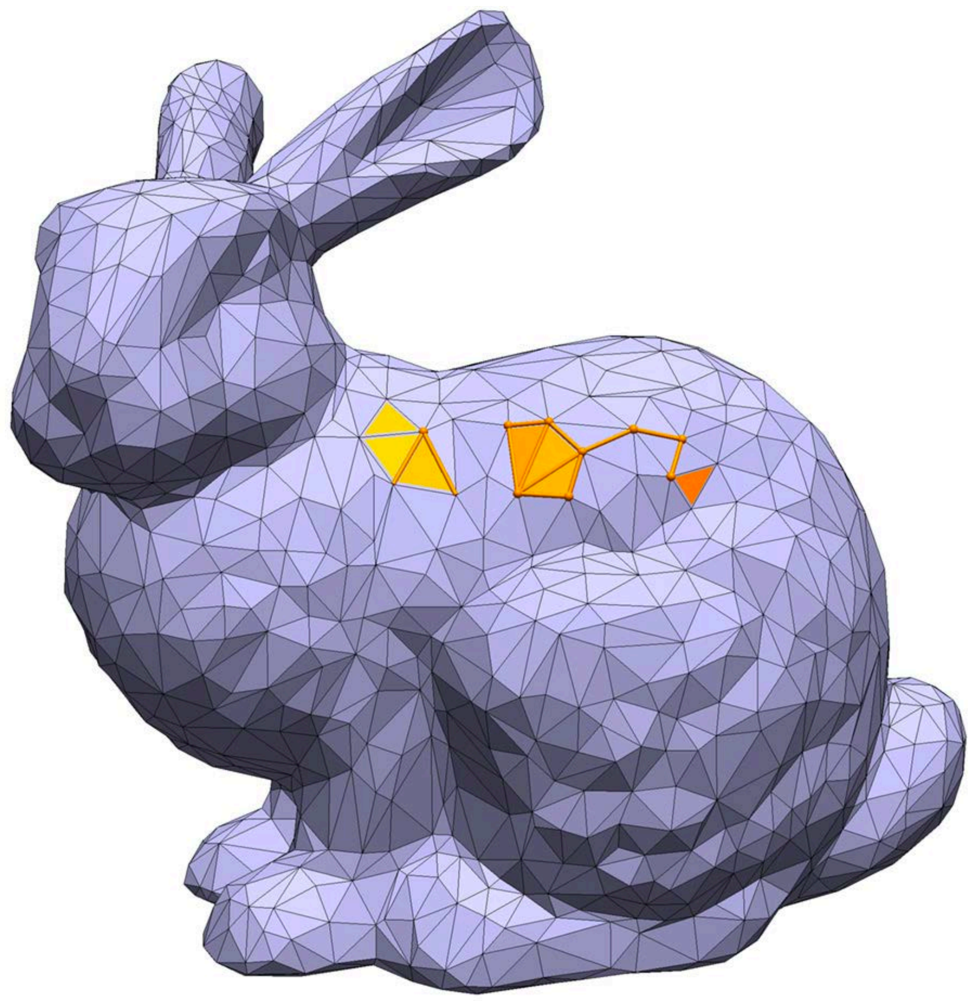
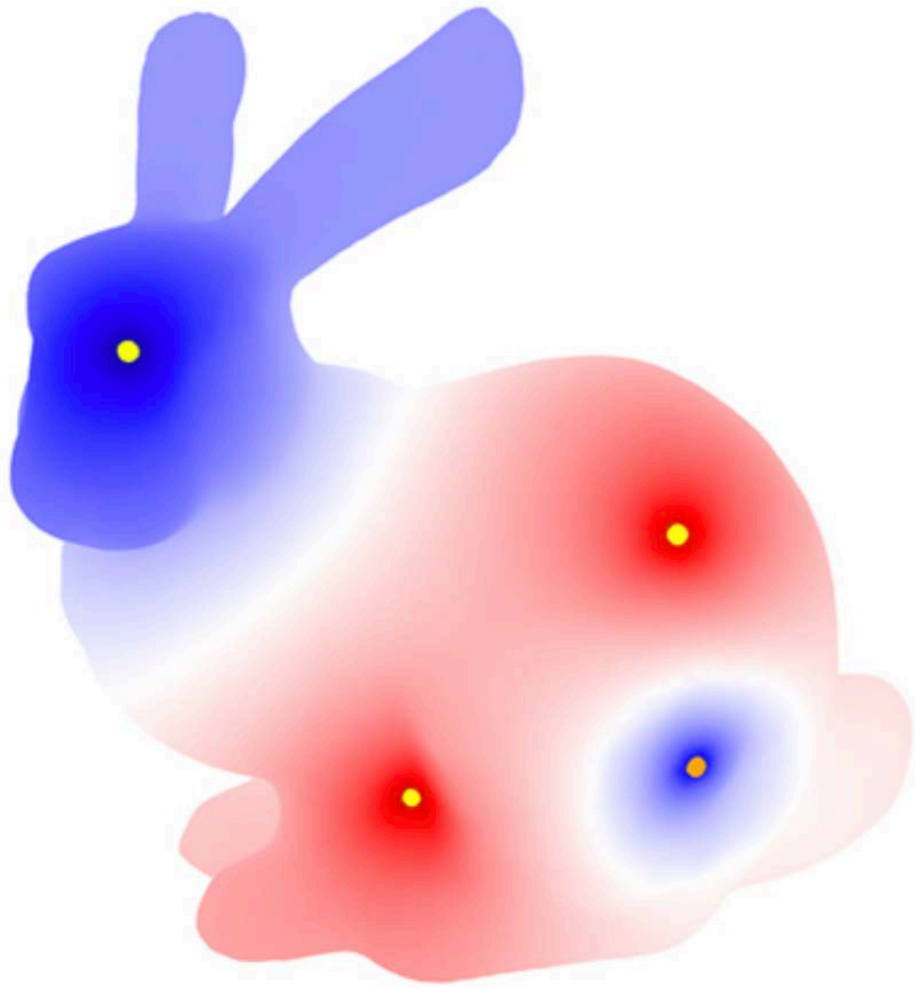
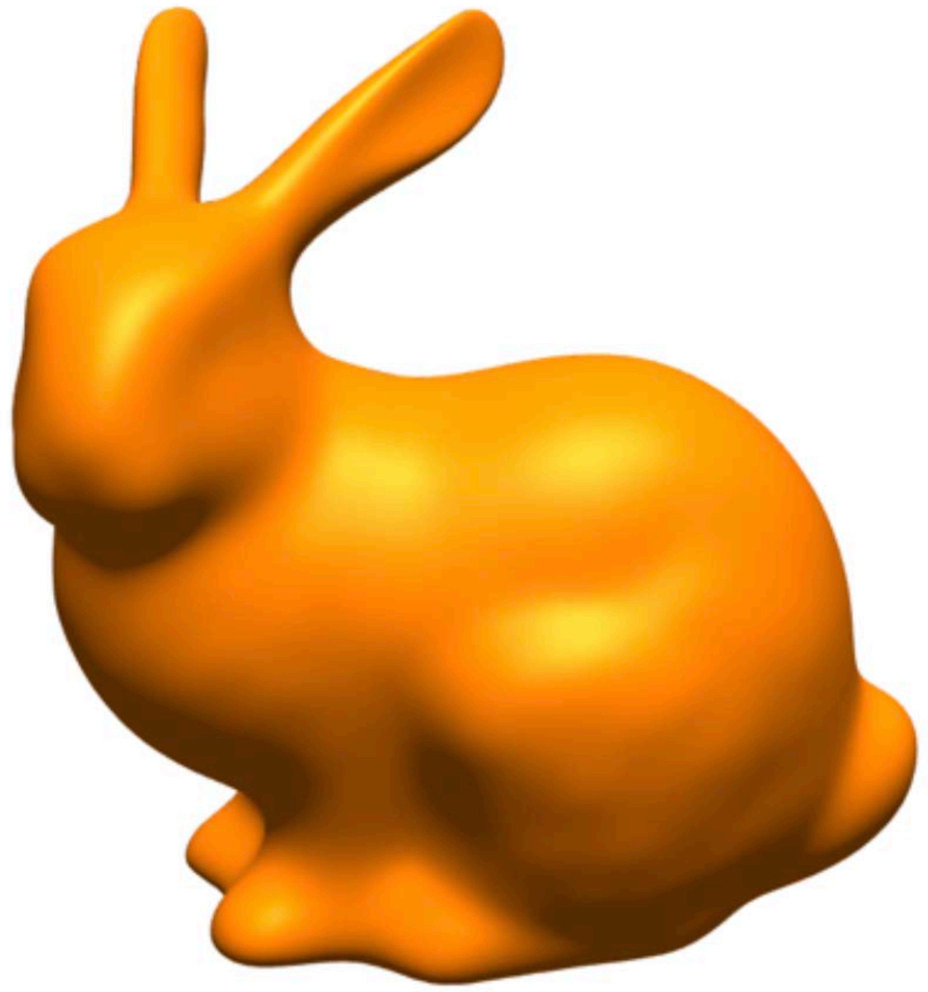
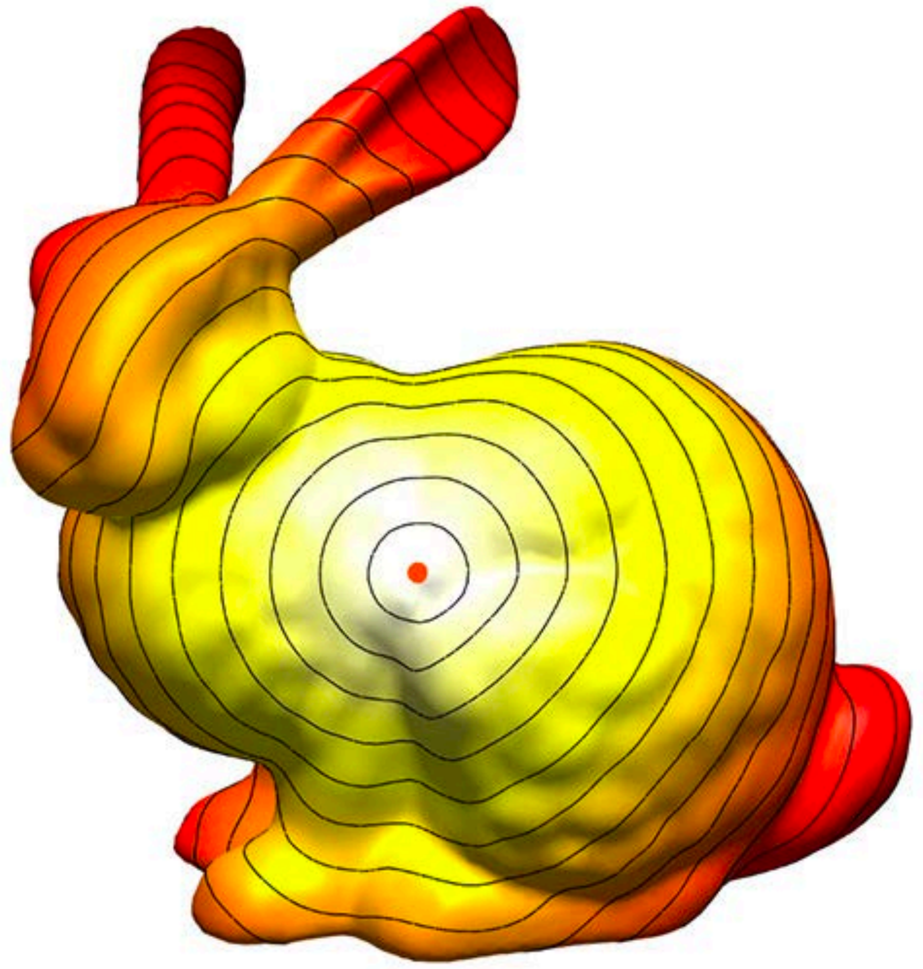
- Beyond this "toy" problem, the *no free lunch* scenario is quite common when we try to find finite/computational versions of smooth objects.
- <u>Many</u> examples (**physics:** conservation of energy, momentum, & symplectic form for conservative time integrators; **geometry:** discrete Laplace operators)
- At a more practical level: **The Game** played in DDG often leads to new & unexpected approaches to geometric algorithms (simpler, faster, stronger guarantees, …)
- Will see *much* more of this as the course continues!

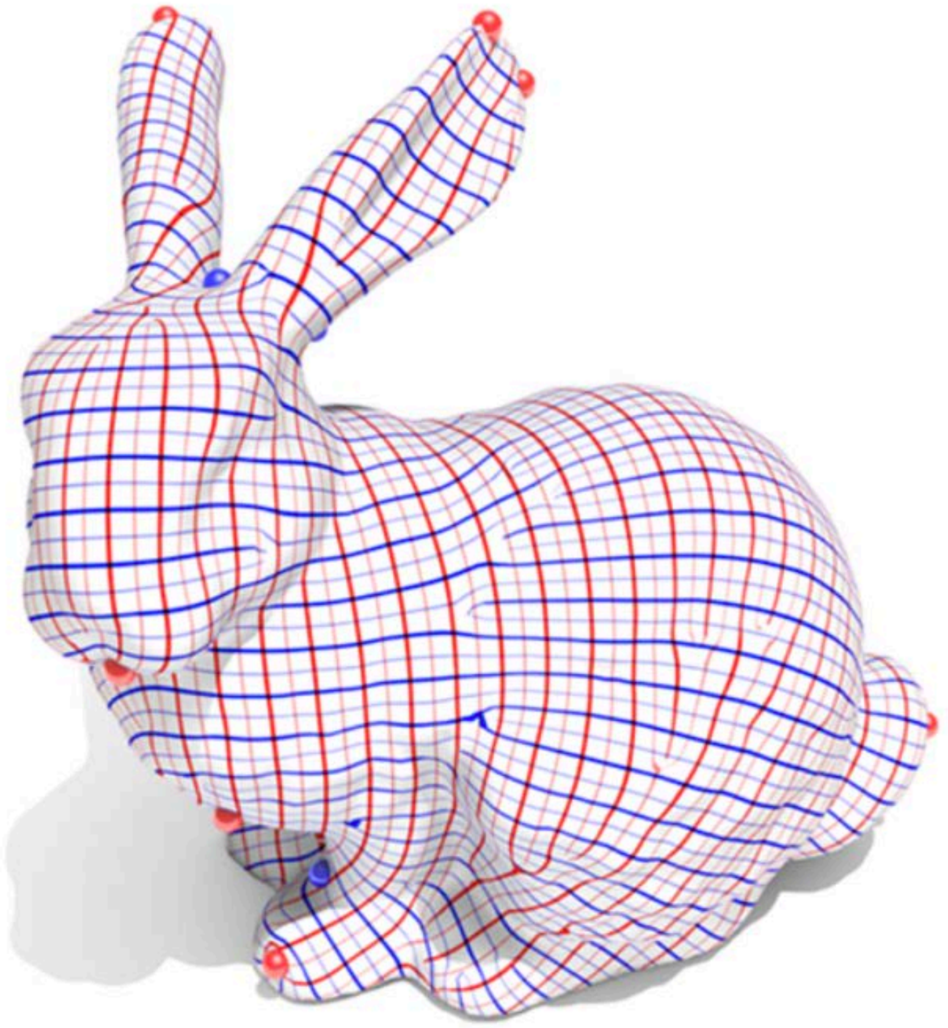
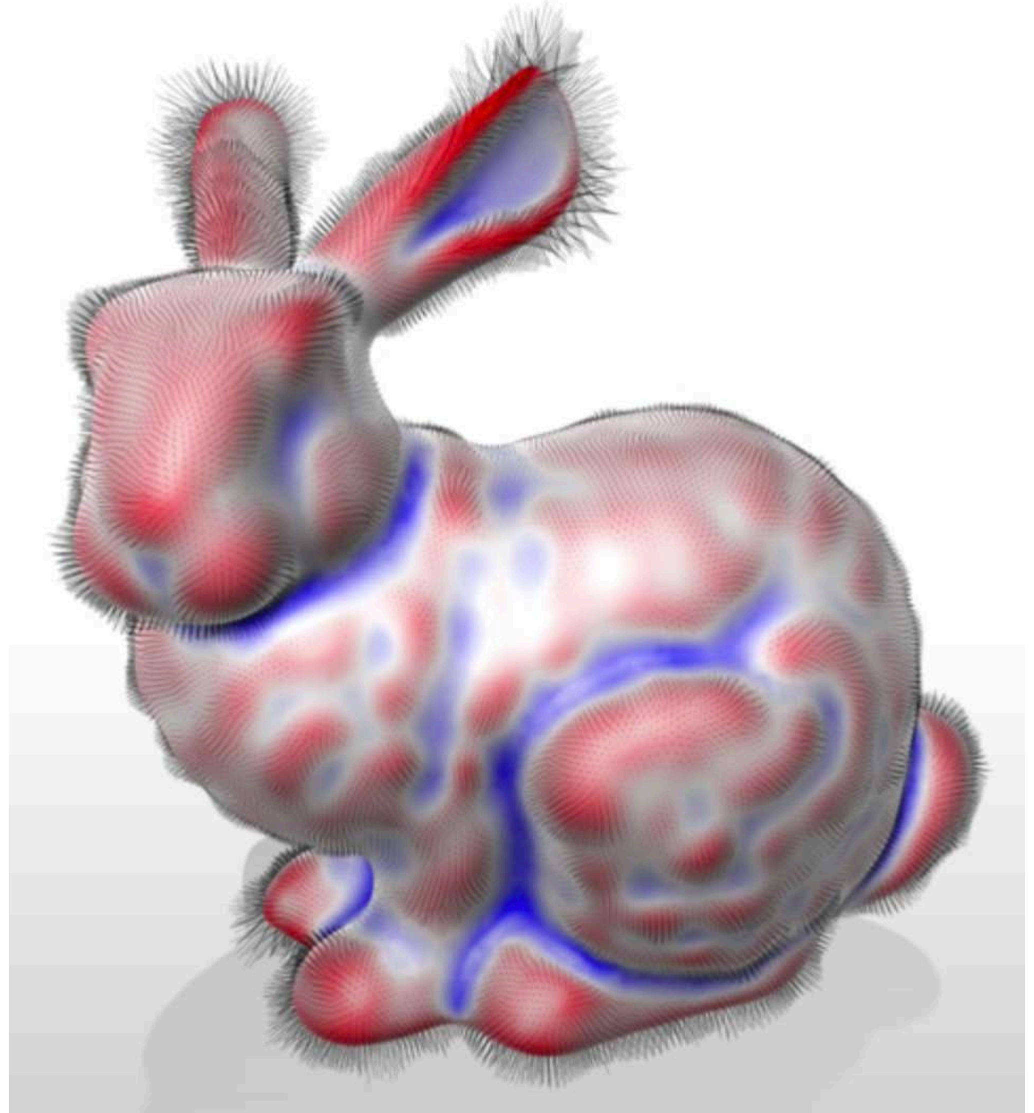
Course Roadmap

- Combinatorial Surfaces
- Exterior Calculus
  - Exterior Algebra (linear algebra)
  - Differential Forms (3D calculus)
  - Discrete Exterior Calculus
- Curves (2D & 3D)
  - Smooth
  - Discrete
- Surfaces
  - Smooth
  - Discrete
- Curvature
- Laplace-Beltrami
- Geodesics
- Conformal Geometry
- Homology & Cohomology
- (Additional Topics)

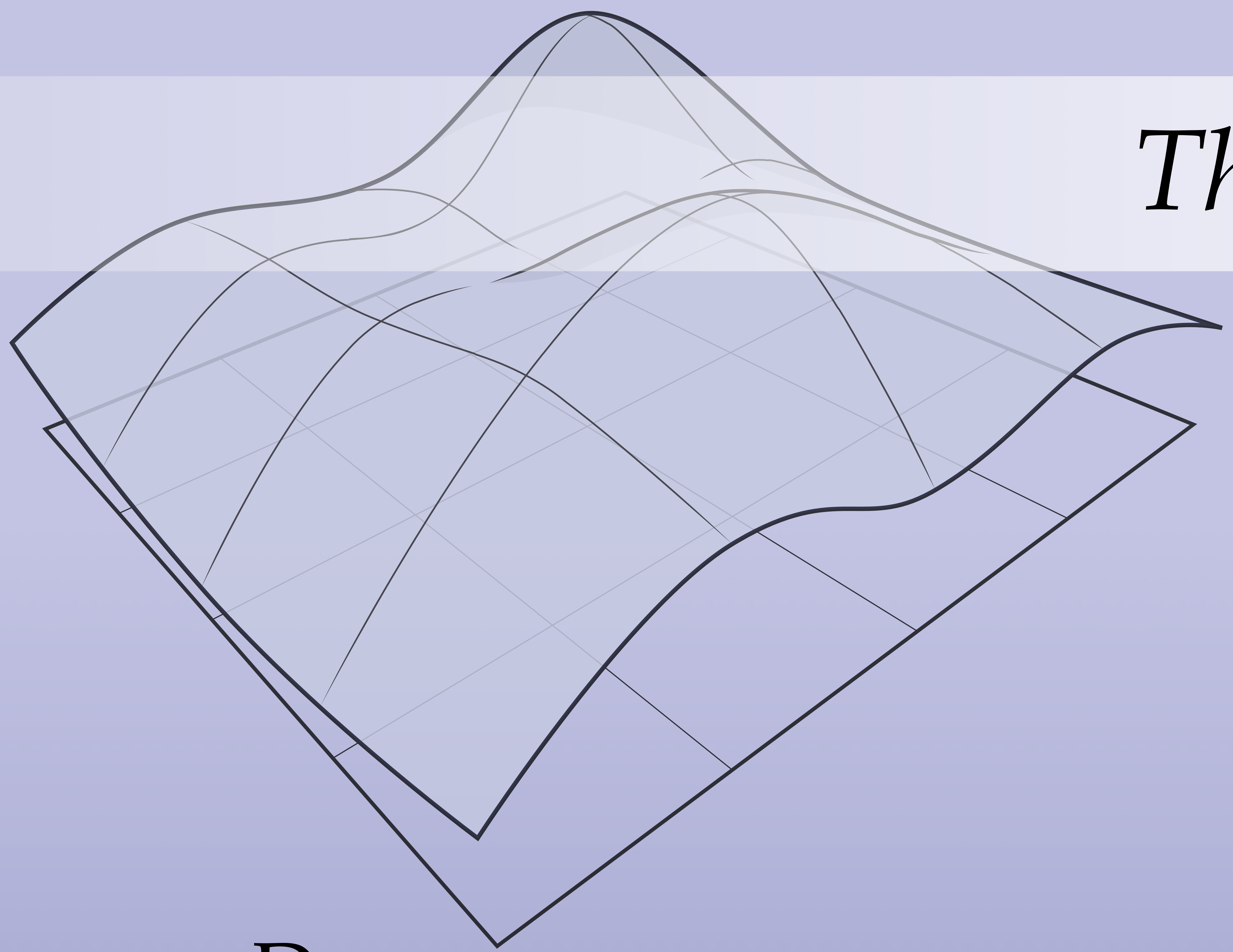
…don't worry if these words sound intimidating right now!

Applications & Hands-On Exercises

Thanks!



DISCRETE DIFFERENTIAL GEOMETRY:
AN APPLIED INTRODUCTION

Keenan Crane • CMU 15-458/858B • Fall 2017