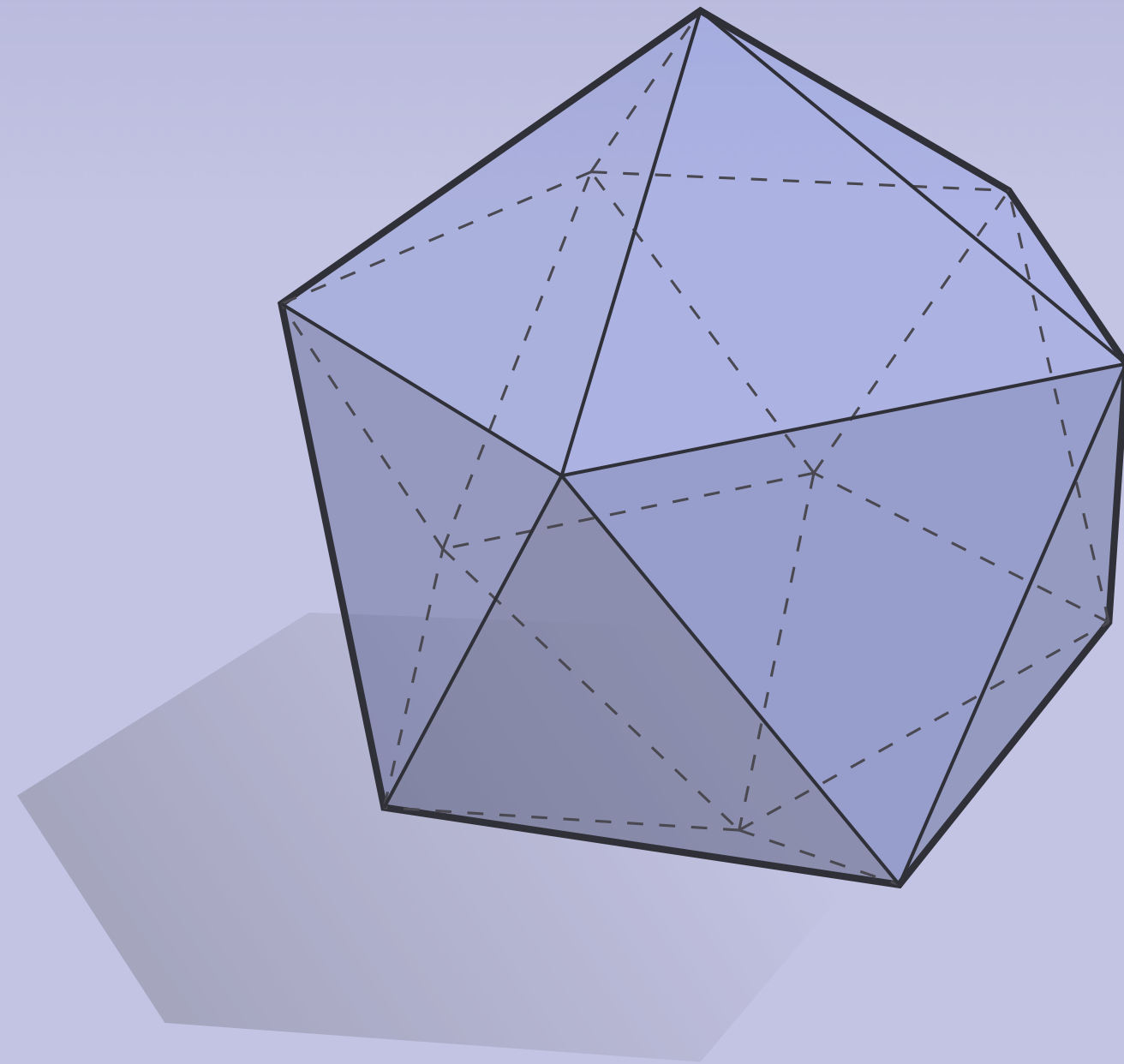


DISCRETE DIFFERENTIAL  
GEOMETRY:  
AN APPLIED INTRODUCTION

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

# LECTURE 7: CURVES

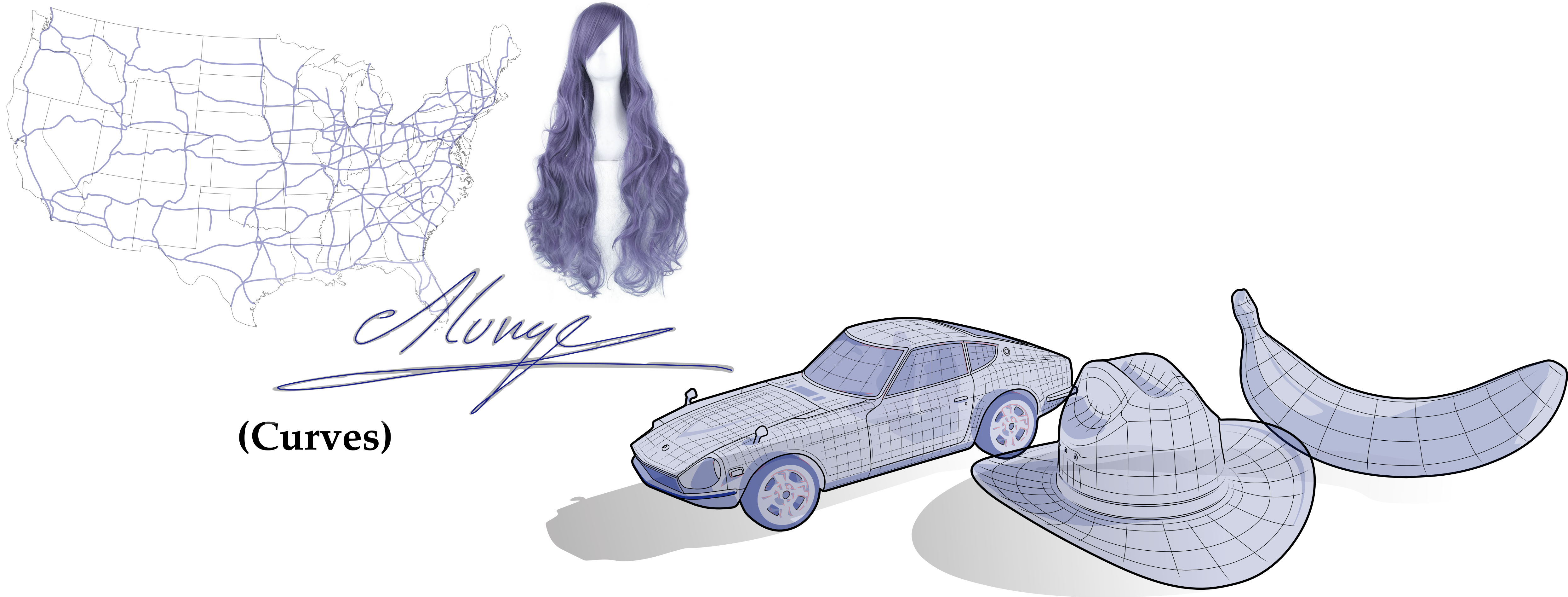


## DISCRETE DIFFERENTIAL GEOMETRY: AN APPLIED INTRODUCTION

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

# Curves & Surfaces

- Much of the geometry we encounter in life well-described by *curves* and *surfaces*\*



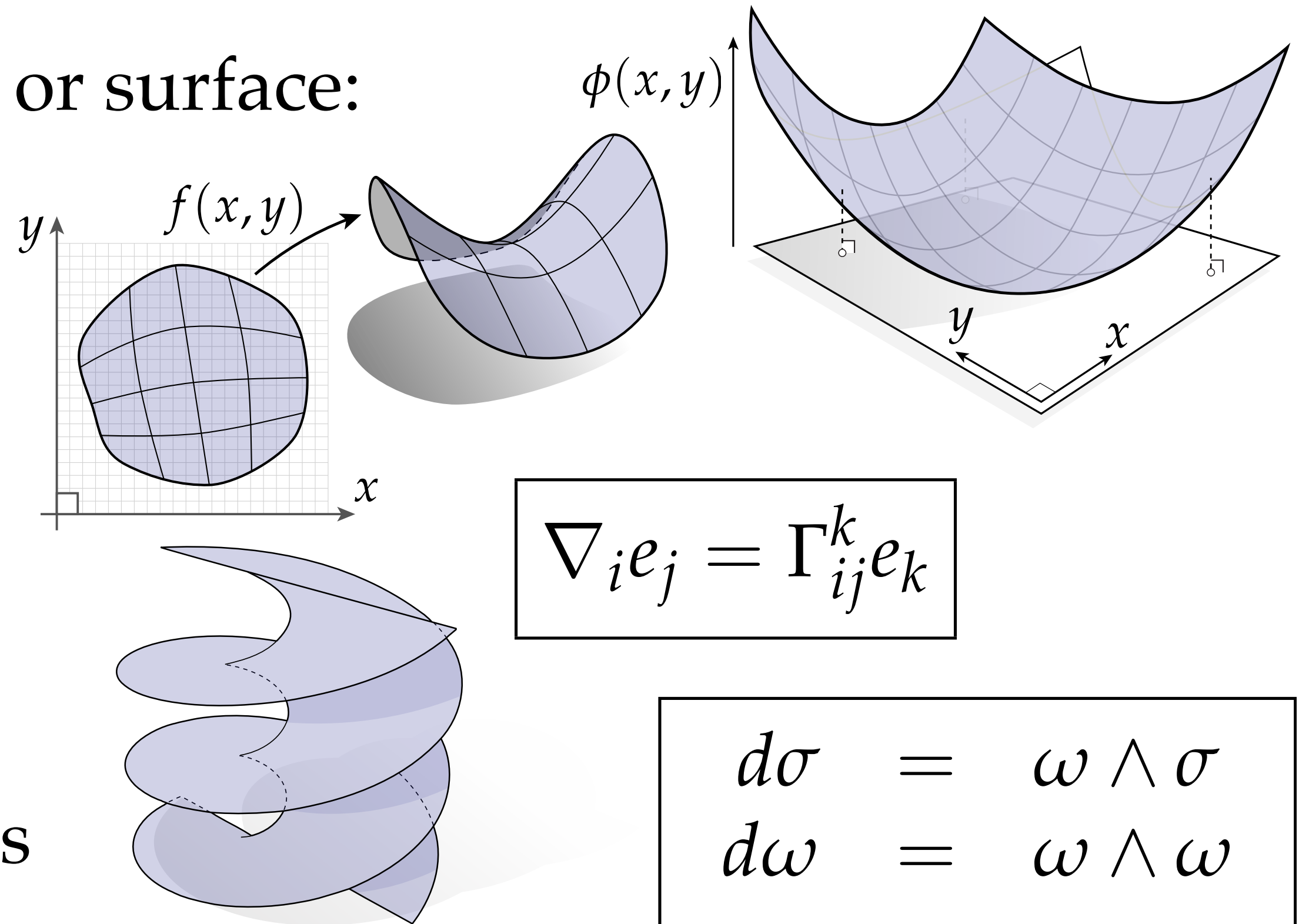
\*Or solids... but the boundary of a solid is a surface!

**(Surfaces)**



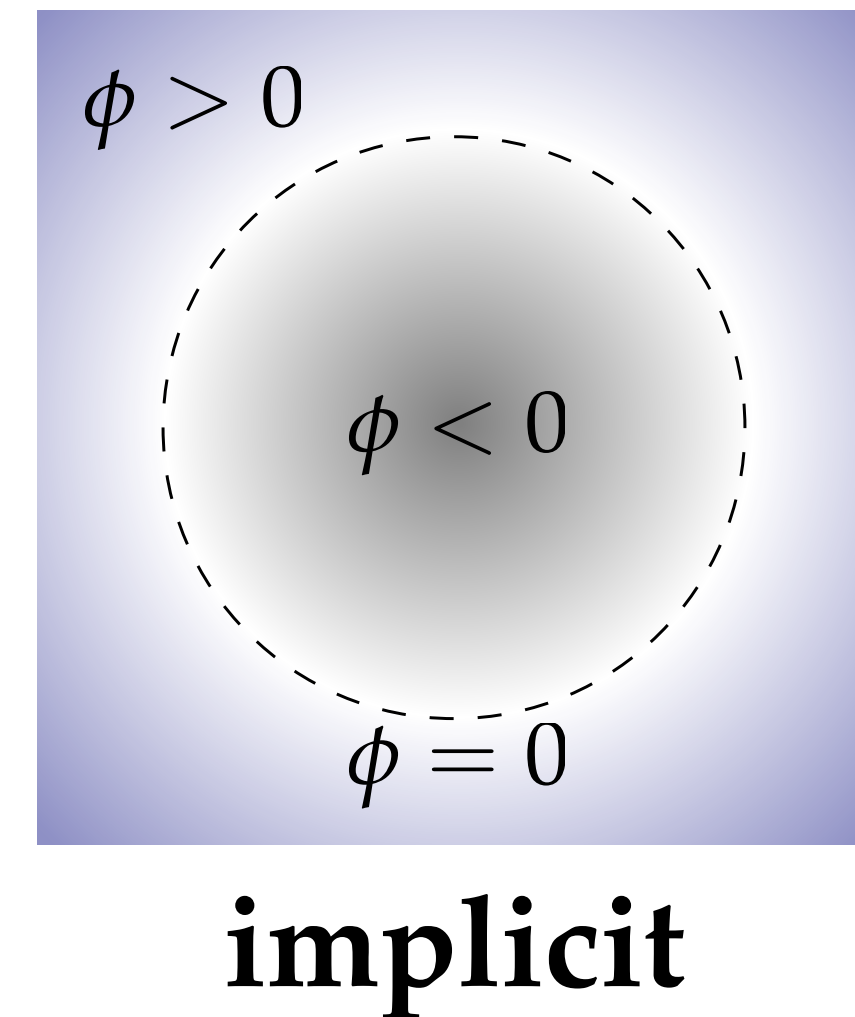
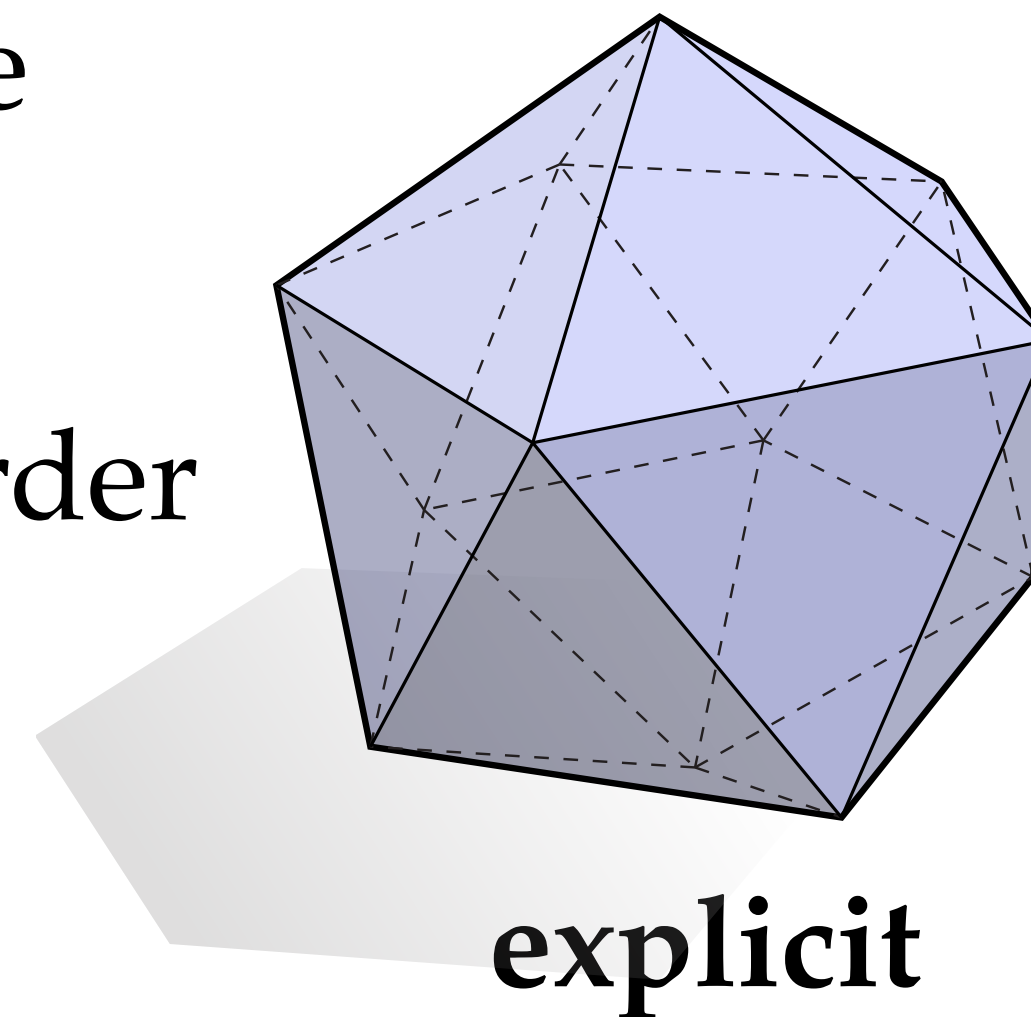
# Smooth Descriptions of Surfaces

- Many ways to express the geometry of a curve or surface:
  - height function over tangent plane
  - local parameterization
  - Christoffel symbols — coordinates / indices
  - **differential forms** — “coordinate free”
  - moving frames — builds on differential forms
  - Riemann surfaces (*local*); Quaternionic functions (*global*)
- People can get very religious about these different “dialects”... best to be multilingual!
- We'll dive deep into one description (**differential forms**) and touch on others



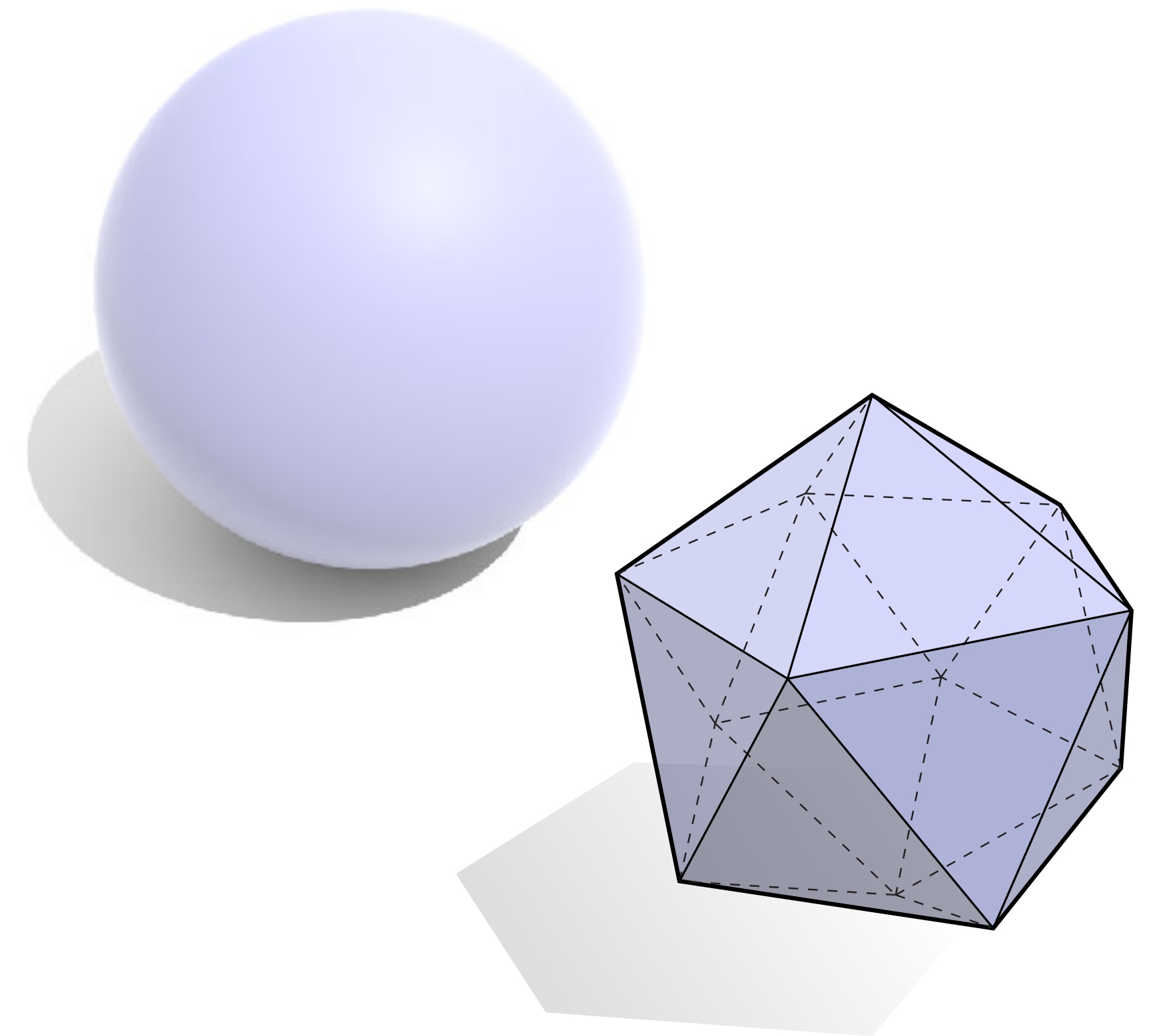
# Discrete Descriptions of Curves & Surfaces

- Also *many* ways to discretize a surface
- For instance:
  - **implicit** — *e.g.*, zero set of scalar function on a grid
    - good for changing topology, high accuracy
    - expensive to store / adaptivity is harder
    - hard to solve sophisticated equations *on* surface
  - **explicit** — *e.g.*, polygonal surface mesh
    - changing topology, high-order continuity is harder
    - cheaper to store / adaptivity is much easier
    - more mature tools for equations *on* surfaces
- Don't be “religious”; use the right tool for the job!



# Curves & Surfaces — Overview

- **Goal:** understand curves & surfaces from complementary smooth and discrete points of view.
- **Smooth setting:**
  - express geometry via differential forms
  - will first need to think about *vector-valued* forms
- **Discrete setting:**
  - use explicit mesh as domain
  - express geometry via discrete differential forms
- **Payoff:** will become very easy to switch back & forth between smooth setting (*scribbling in a notebook*) and discrete setting (*running algorithms on real data!*)







# *Vector Valued Differential Forms*

# Vector Valued $k$ -Forms

- So far, we've defined a  $k$ -form as a linear map from  $k$  vectors to a real number
- For working with curves and surfaces in  $R^n$ , it will be essential to generalize this definition to *vector-valued  $k$ -forms*.
- In particular, a **vector-valued  $k$ -form** is a multi-linear map from  $k$  vectors in a vector space  $V$  to some other vector space  $U$  (not necessarily  $U=V$ )
  - So far, for instance, all of our forms have been  $R$ -valued  $k$ -forms on  $R^n$  ( $V=R^n, U=R$ )
  - A  $R^3$ -valued 2-form on  $R^2$  would instead be a multilinear, fully-antisymmetric symmetric map from a pair of vectors  $u, v$  in  $R^2$  to a single vector in  $R^3$ :

$$\alpha : \mathbb{R}^2 \times \mathbb{R}^2 \rightarrow \mathbb{R}^3 \quad \alpha(u, v) = -\alpha(v, u)$$

$$\alpha(au + bv, w) = a\alpha(u, w) + b\alpha(v, w), \quad \forall u, v, w \in \mathbb{R}^2, a, b \in \mathbb{R}$$

**Q:** What kind of object is a  $R^2$ -valued 0-form on  $R^2$ ?



# Vector-Valued $k$ -forms — Example

Consider for instance the following  $\mathbb{R}^3$ -valued 1-form on  $\mathbb{R}^2$ :

$$\alpha := \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix} e^1 + \begin{bmatrix} 4 \\ 5 \\ 6 \end{bmatrix} e^2$$

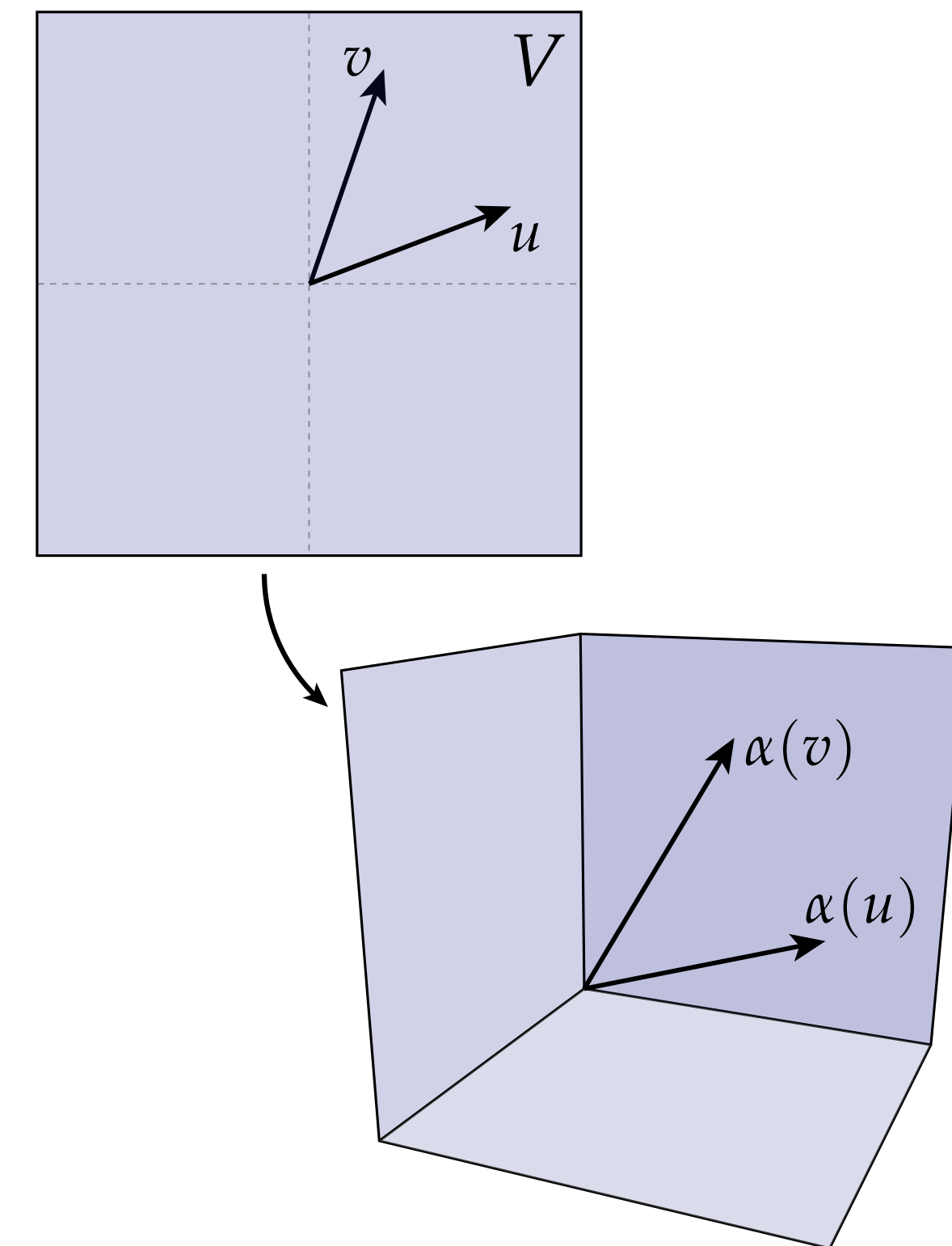
**Q:** What do we get if we evaluate this 1-form on the vector

$$u := e_1 - e_2$$

**A:** Evaluation is not much different from real-valued forms:

$$\alpha(u) = \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix} \cancel{e^1(e_1 - e_2)} \overset{1}{\rightarrow} + \begin{bmatrix} 4 \\ 5 \\ 6 \end{bmatrix} \cancel{e^2(e_1 - e_2)} \overset{-1}{\rightarrow} = \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix} - \begin{bmatrix} 4 \\ 5 \\ 6 \end{bmatrix} = \begin{bmatrix} -3 \\ -3 \\ -3 \end{bmatrix}$$

**Key idea:** coefficients just have a different type



# Wedge Product of Vector-Valued $k$ -Forms

- Most important change is how we evaluate wedge product for vector-valued forms.
- Consider for instance a pair of  $\mathbb{R}^3$ -valued 1-forms:

$$\alpha, \beta : V \rightarrow \mathbb{R}^3$$

- To evaluate their wedge product on a pair of vectors  $u, v$  we would normally write:

$$(\alpha \wedge \beta)(u, v) = \alpha(u)\beta(v) - \alpha(v)\beta(u)$$

- If  $\alpha$  and  $\beta$  were *real*-valued, then  $\alpha(u)$ ,  $\beta(v)$ ,  $\alpha(v)$ ,  $\beta(u)$ , would just be *real numbers*, so we could just multiply the two pairs and take their difference.
- But what does it mean to take the “product” of two vectors from  $\mathbb{R}^3$ ?
- Many possibilities (*e.g.*, dot product), but if we want result to be an  $\mathbb{R}^3$ -valued 2-form, the product we choose must produce another 3-vector!

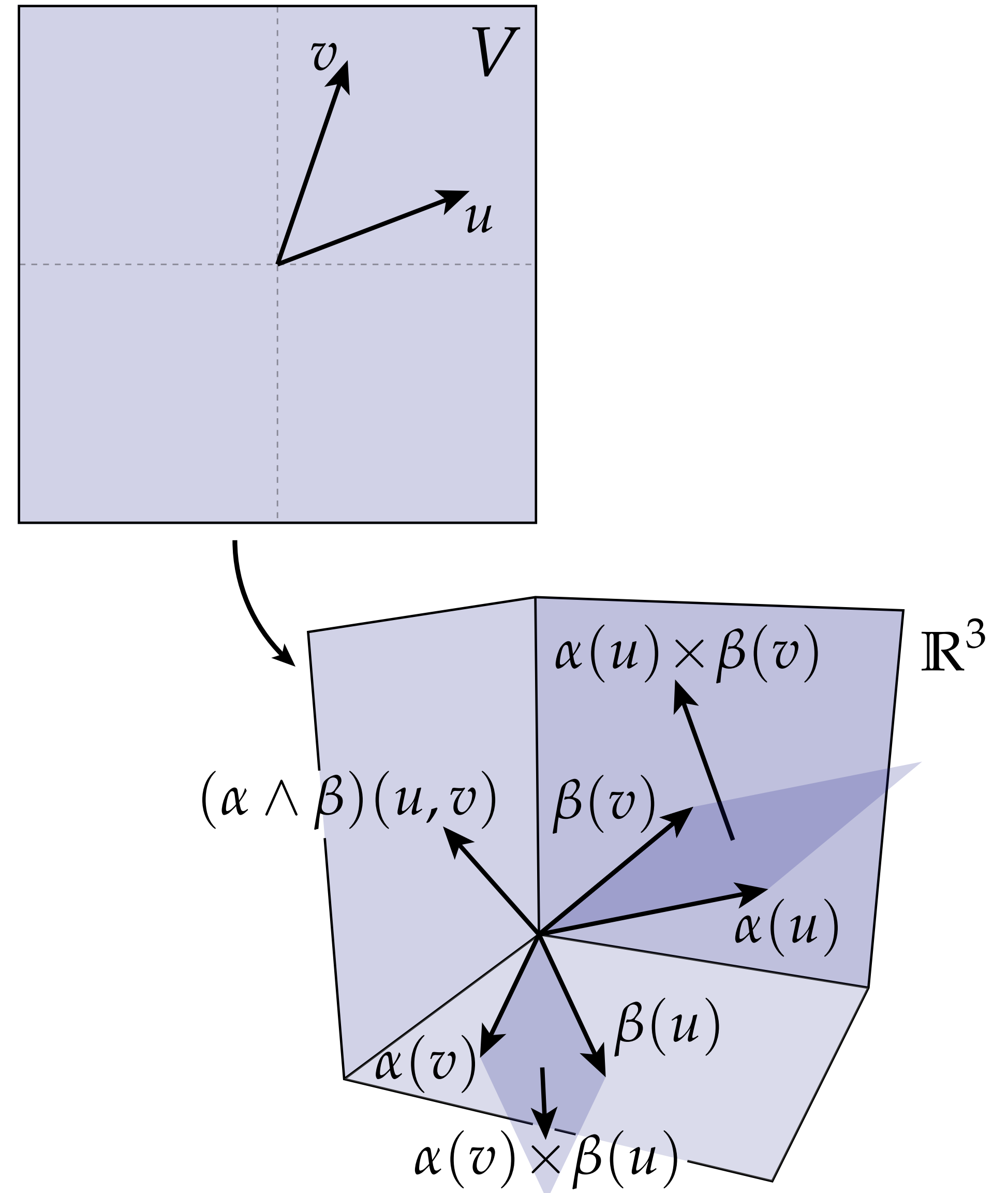
# Wedge Product of $\mathbb{R}^3$ -Valued $k$ -Forms

- Most common case for our study of surfaces:
  - $k$ -forms are  $\mathbb{R}^3$ -valued
  - use **cross product** to multiply 3-vectors

$$\alpha, \beta : V \rightarrow \mathbb{R}^3$$

$$\alpha \wedge \beta : V \times V \rightarrow \mathbb{R}^3$$

$$(\alpha \wedge \beta)(u, v) := \alpha(u) \times \beta(v) - \alpha(v) \times \beta(u)$$





# $R^3$ -valued 1-forms: Antisymmetry & Symmetry

With real-valued forms, we observed antisymmetry in both the wedge product of 1-forms as well as the application of the 2-form to a pair of vectors, *i.e.*,

$$(\alpha \wedge \beta)(u, v) = -(\alpha \wedge \beta)(v, u)$$

$$(\beta \wedge \alpha)(u, v) = -(\alpha \wedge \beta)(u, v)$$

What happens w /  $R^3$ -valued 1-forms? Since cross product is antisymmetric, we get

$$\begin{aligned}(\alpha \wedge \beta)(v, u) &= \alpha(v) \times \beta(u) - \alpha(u) \times \beta(v) \\ &= -(\alpha(u) \times \beta(v) - \alpha(v) \times \beta(u))\end{aligned}$$

$$\Rightarrow \boxed{(\alpha \wedge \beta)(u, v) = -(\alpha \wedge \beta)(v, u)}$$

**(no change)**

$$\begin{aligned}(\beta \wedge \alpha)(u, v) &= \beta(u) \times \alpha(v) - \beta(v) \times \alpha(u) \\ &= \alpha(u) \times \beta(v) - \alpha(v) \times \beta(u) \\ &= (\alpha \wedge \beta)(u, v)\end{aligned}$$

$$\Rightarrow \boxed{\alpha \wedge \beta = \beta \wedge \alpha}$$

**(big change!)**

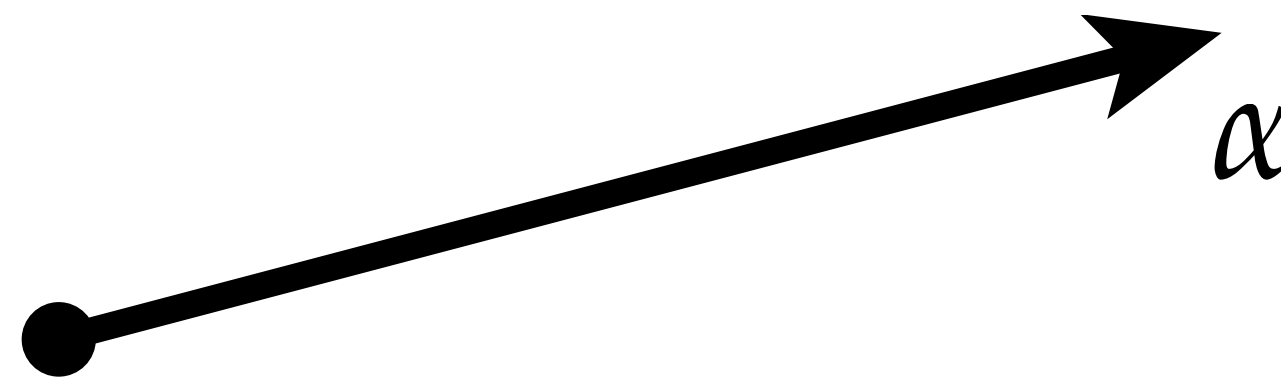
# $R^3$ -valued 1-forms: Self-Wedge

Likewise, we saw that wedging a real-valued 1-form with itself yields zero:

$$\alpha \wedge \alpha = 0$$

Q: What was the *geometric* interpretation?

A: Parallelogram spanned by two copies of the same vector has zero area!



...But, no longer true with  $(R^3, \times)$ -valued 1-forms:

$$(\alpha \wedge \alpha)(u, v) = \alpha(u) \times \alpha(v) - \alpha(v) \times \alpha(u) = 2\alpha(u) \times \alpha(v) \neq 0$$

Geometric meaning will become clearer as we work with surfaces.

# Vector-Valued Differential $k$ -Forms

- Just as we distinguished between a  $k$ -form (value at a single point) and a *differential  $k$ -form* (value at every point in space), we will also say that a *vector-valued differential  $k$ -form* is a vector-valued  $k$ -form at each point of space.
- Just like any differential form, a vector-valued differential  $k$ -form gets evaluated on  $k$  vector fields  $X_1, \dots, X_k$ .
- **Example:** an  $\mathbb{R}^3$ -valued differential 1-form on  $\mathbb{R}^2$  (with coordinates  $u, v$ ):

$$\alpha := \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} du + \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} dv$$

**Q:** What does this 1-form do to any given vector field  $X$  on the plane?

**A:** It simply “copies” it to the  $yz$ -plane in 3D.



# *Exterior Derivative on Vector-Valued Forms*

Unlike the wedge product, not much changes with the exterior derivative. For instance, if we have an  $\mathbb{R}^n$ -valued  $k$ -form we can simply imagine we have  $n$  real-valued  $k$ -forms and differentiate as usual.

## **Example.**

Consider an  $\mathbb{R}^2$ -valued differential 0-form  $\phi_{(x,y)} := \begin{bmatrix} x^2 \\ xy \end{bmatrix}$

$$\text{Then } d\phi = \frac{\partial \phi}{\partial x} dx + \frac{\partial \phi}{\partial y} dy = \begin{bmatrix} 2x \\ y \end{bmatrix} dx + \begin{bmatrix} 0 \\ x \end{bmatrix} dy$$

## **Example.**

Consider an  $\mathbb{R}^2$ -valued differential 1-form  $\alpha_{(x,y)} := \begin{bmatrix} x^2 \\ xy \end{bmatrix} dx + \begin{bmatrix} xy \\ y^2 \end{bmatrix} dy$

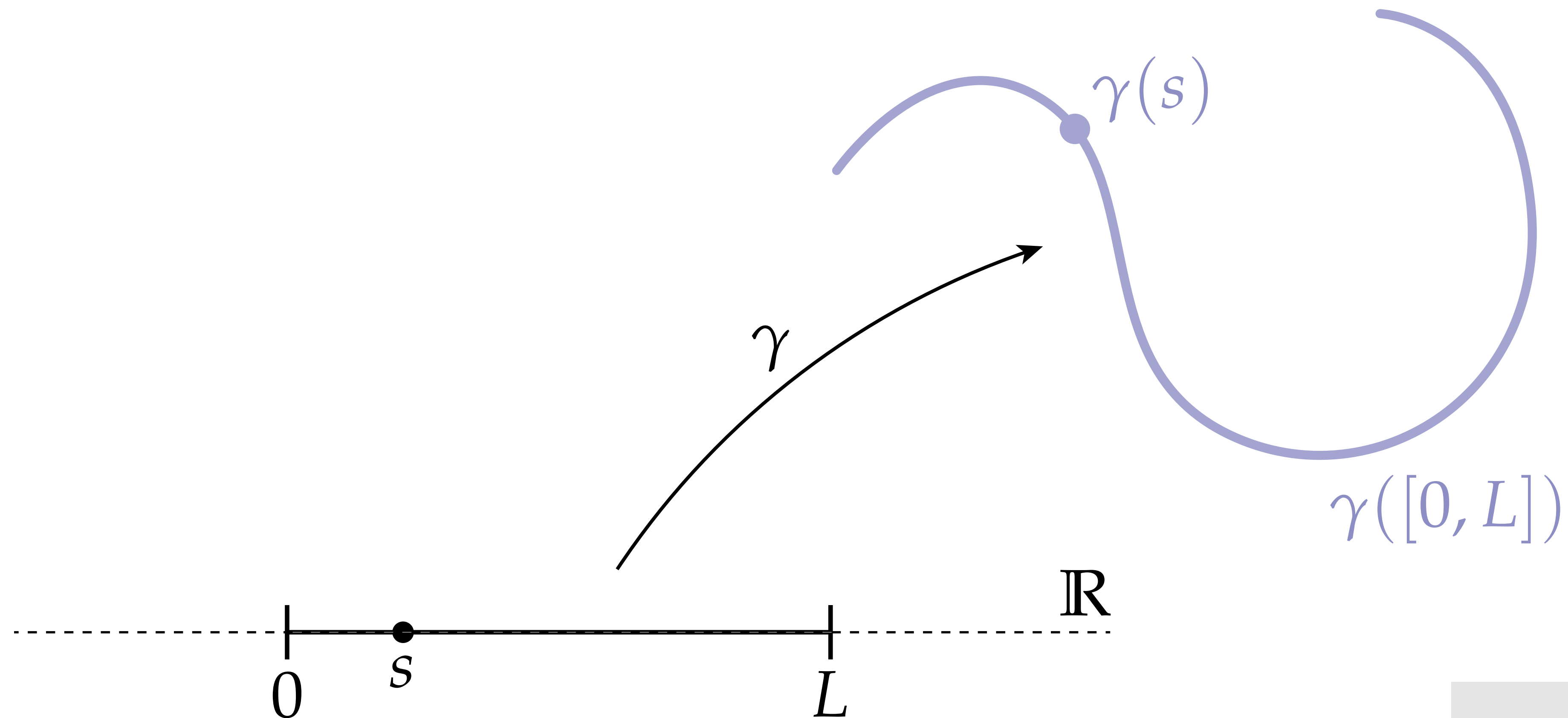
$$\text{Then } d\alpha = \left( \begin{bmatrix} 2x \\ y \end{bmatrix} dx + \begin{bmatrix} 0 \\ x \end{bmatrix} dy \right) \wedge dx + \left( \begin{bmatrix} y \\ 0 \end{bmatrix} dx + \begin{bmatrix} x \\ 2y \end{bmatrix} dy \right) \wedge dy = \begin{bmatrix} y \\ -x \end{bmatrix} dx \wedge dy$$

The background features a series of overlapping, semi-transparent curves in shades of light blue and grey. These curves create a complex, layered geometric pattern. A central white rectangular area serves as a backdrop for the title text. The overall aesthetic is clean and modern, with a focus on mathematical or geometric forms.

# *Planar Curves*

# Parameterized Plane Curve

- A **parameterized plane 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...

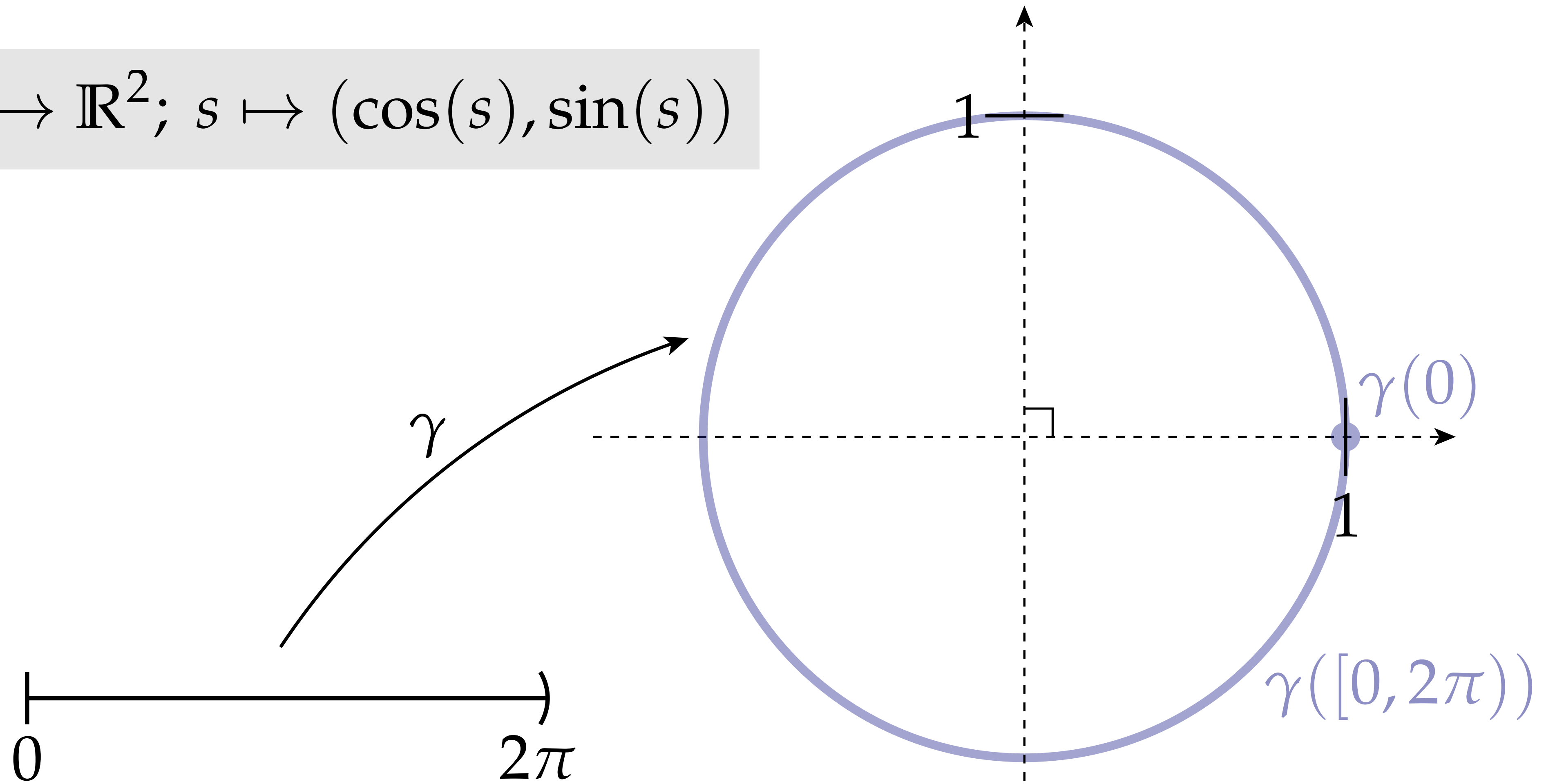
$$\gamma : [0, L] \rightarrow \mathbb{R}^2$$



# Curves in the Plane—Example

- As an example, we can express a circle as a parameterized curve  $\gamma$ :

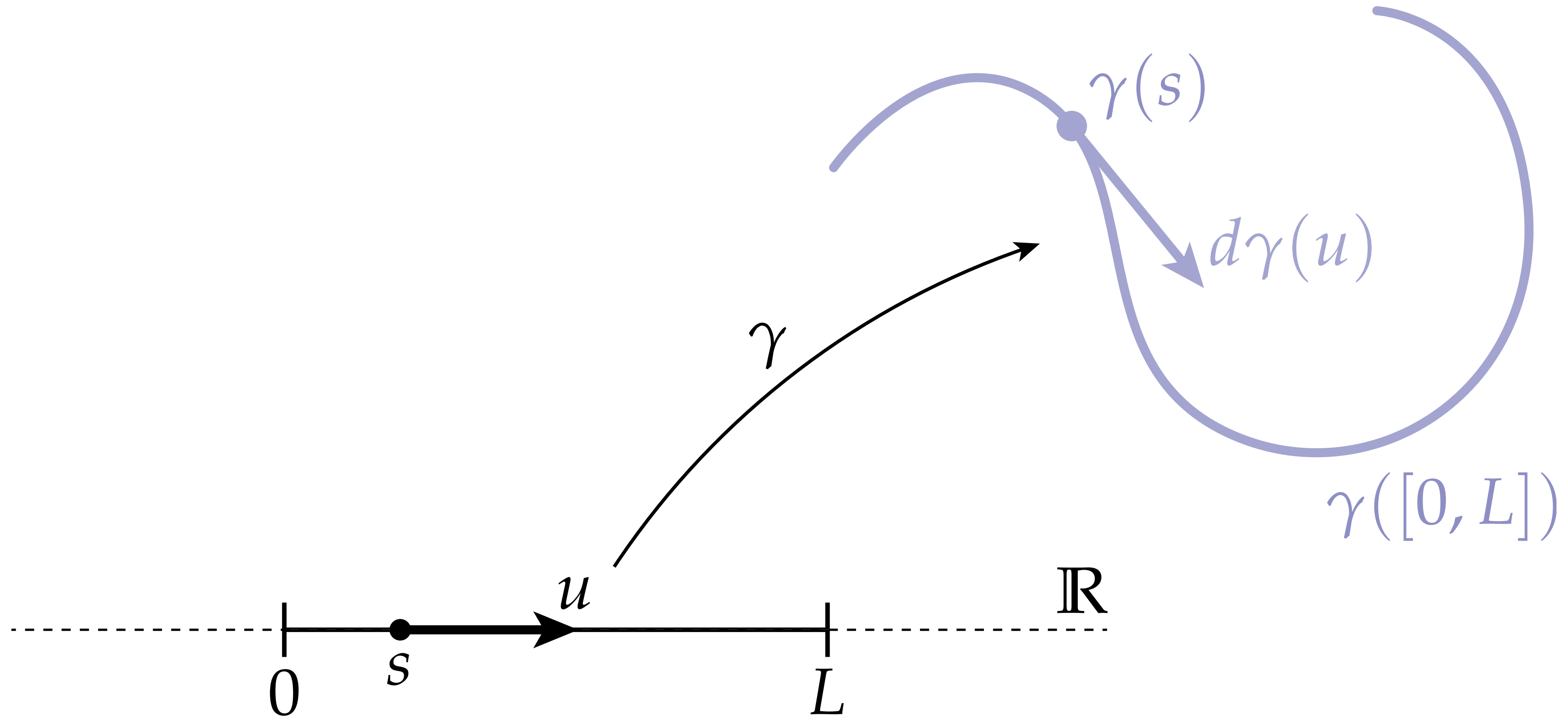
$$\gamma : [0, 2\pi) \rightarrow \mathbb{R}^2; s \mapsto (\cos(s), \sin(s))$$



The circle is an example of a *closed* curve, meaning that endpoints meet.

# Differential of a Curve

- If we think of a parameterized curve as an  $\mathbb{R}^2$ -valued 0-form on an interval of the real line, then the differential (or exterior derivative) says how vectors on the domain get mapped into the plane:



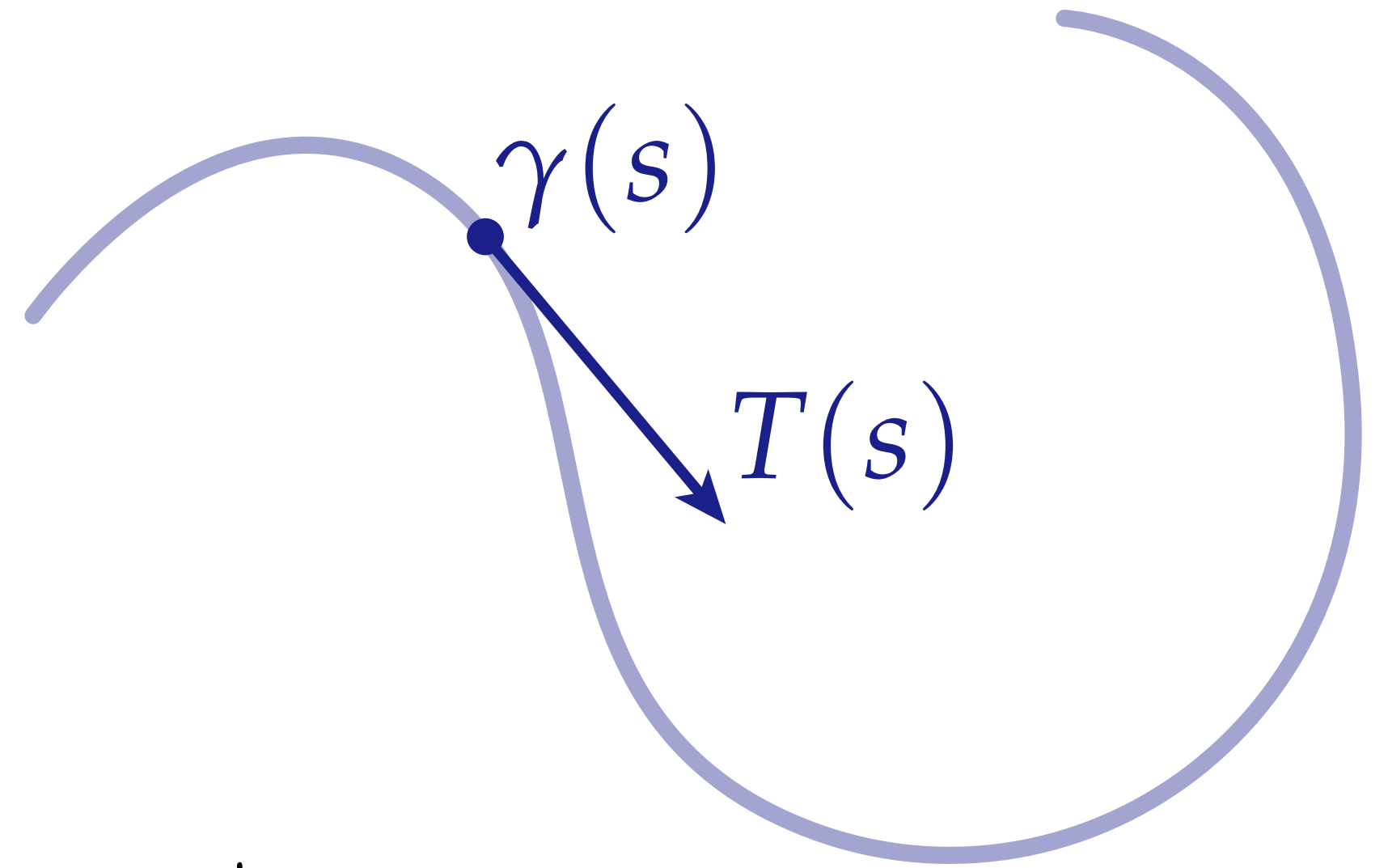
# Tangent of a Curve

- 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 regular curve is the map obtained by normalizing its first derivative:

$$T(s) := \frac{d}{ds} \gamma(s) / \left| \frac{d}{ds} \gamma(s) \right| = d\gamma\left(\frac{d}{ds}\right) / \left| d\gamma\left(\frac{d}{ds}\right) \right|$$

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

$$T(s) := \frac{d}{ds} \gamma(s) = d\gamma\left(\frac{d}{ds}\right)$$





# Tangent of a Curve—Example

- Let's compute the unit tangent of a circle:

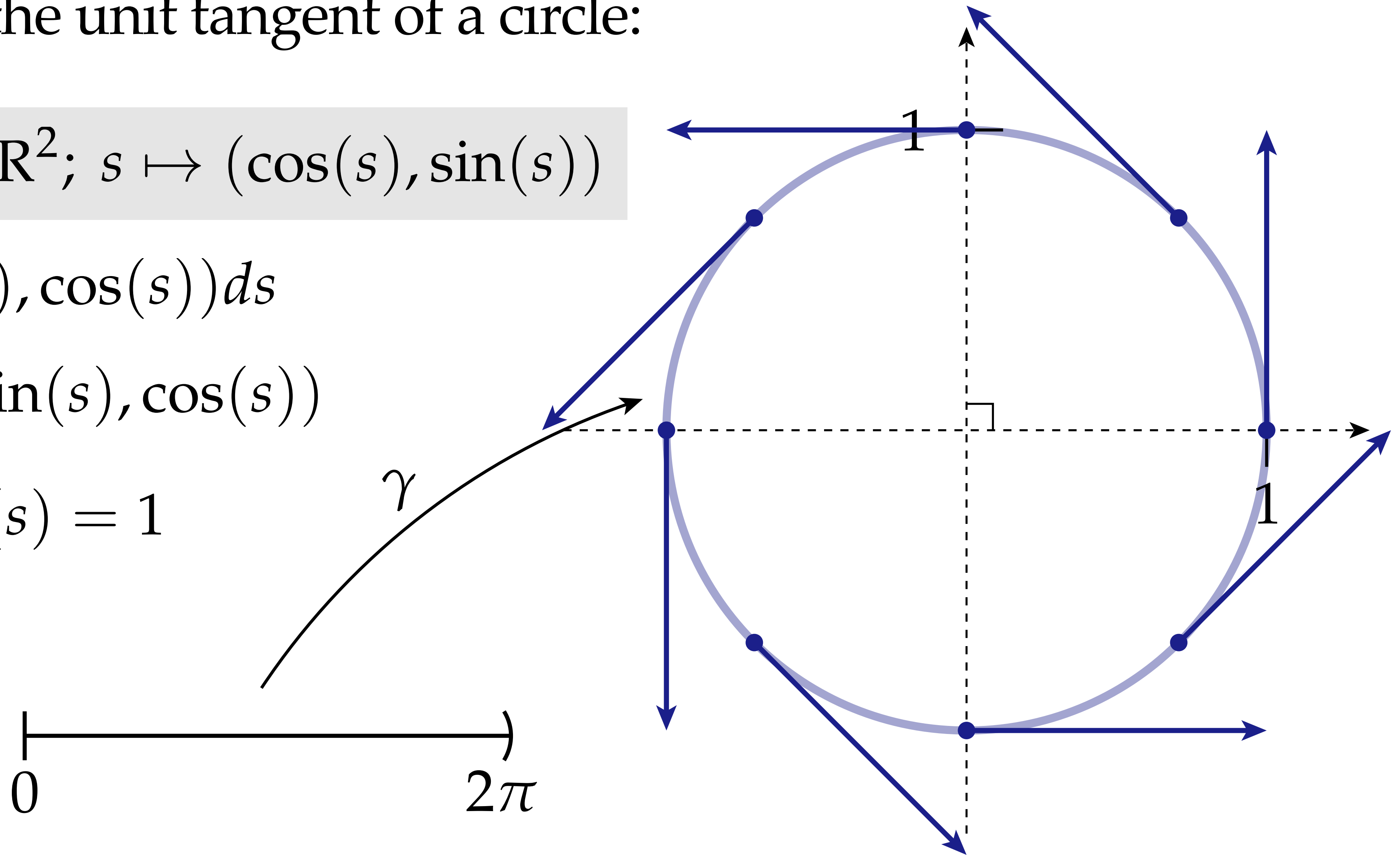
$$\gamma : [0, 2\pi) \rightarrow \mathbb{R}^2; s \mapsto (\cos(s), \sin(s))$$

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

$$d\gamma\left(\frac{\partial}{\partial s}\right) = (-\sin(s), \cos(s))$$

$$\cos^2(s) + \sin^2(s) = 1$$

$$\Rightarrow T = d\gamma\left(\frac{d}{ds}\right)$$

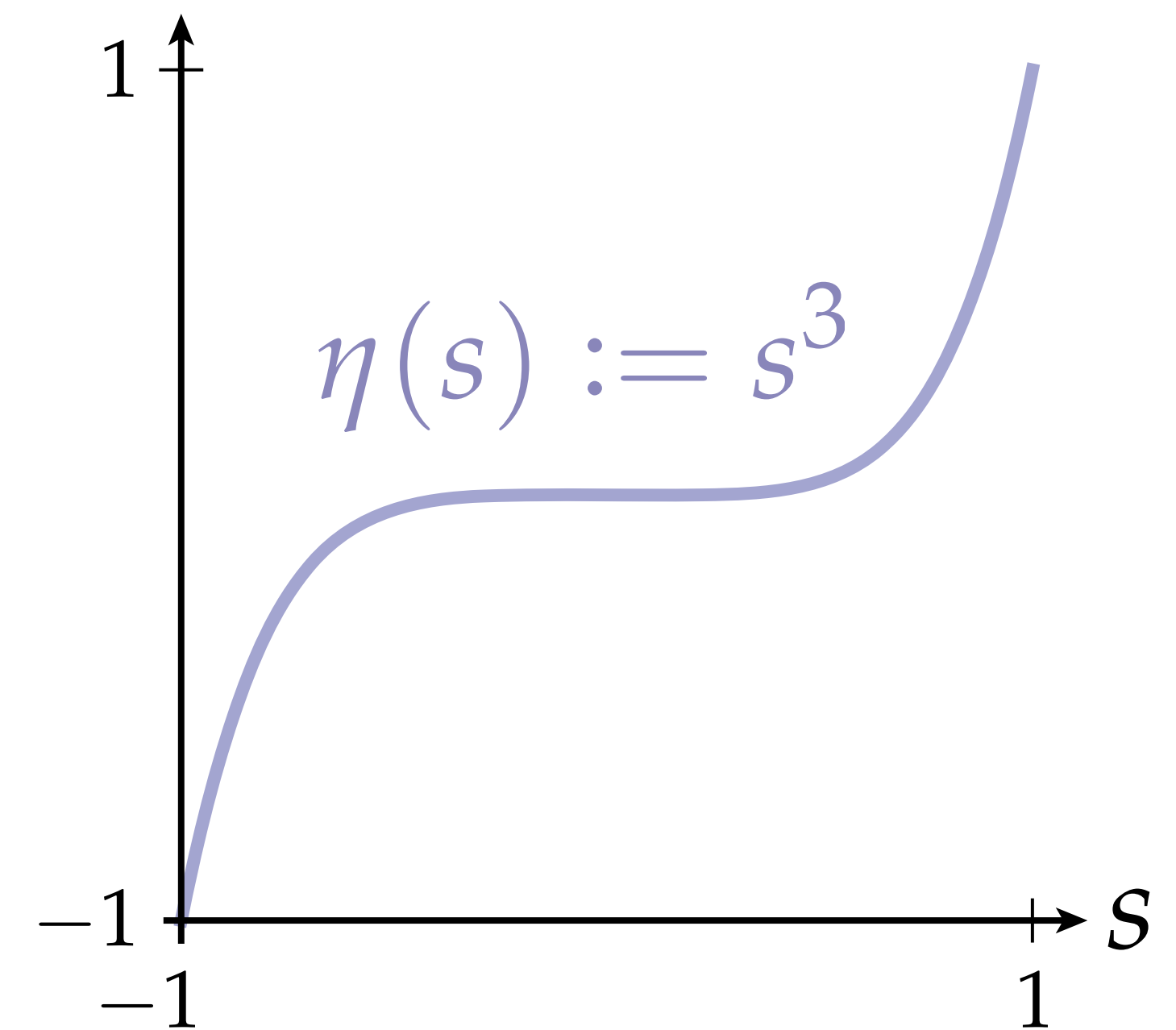
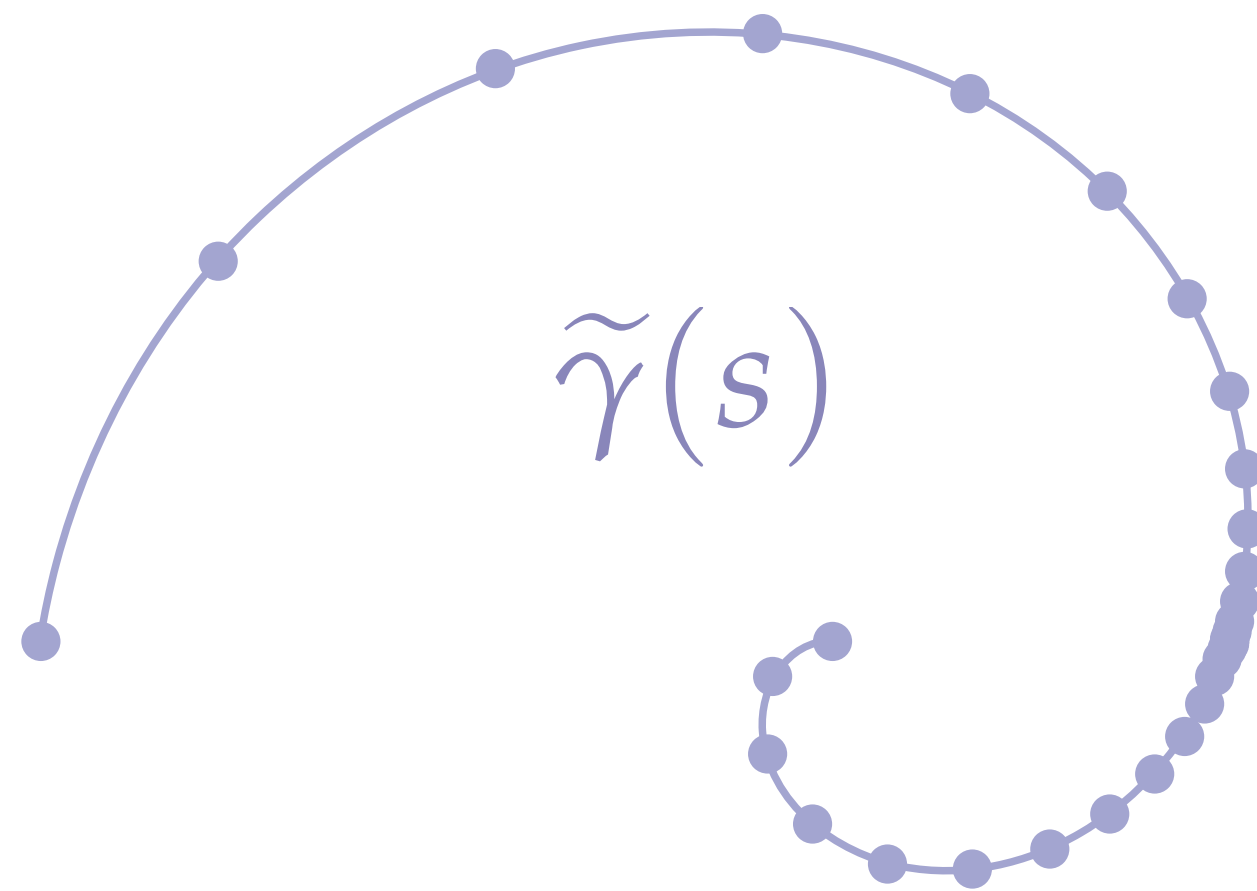
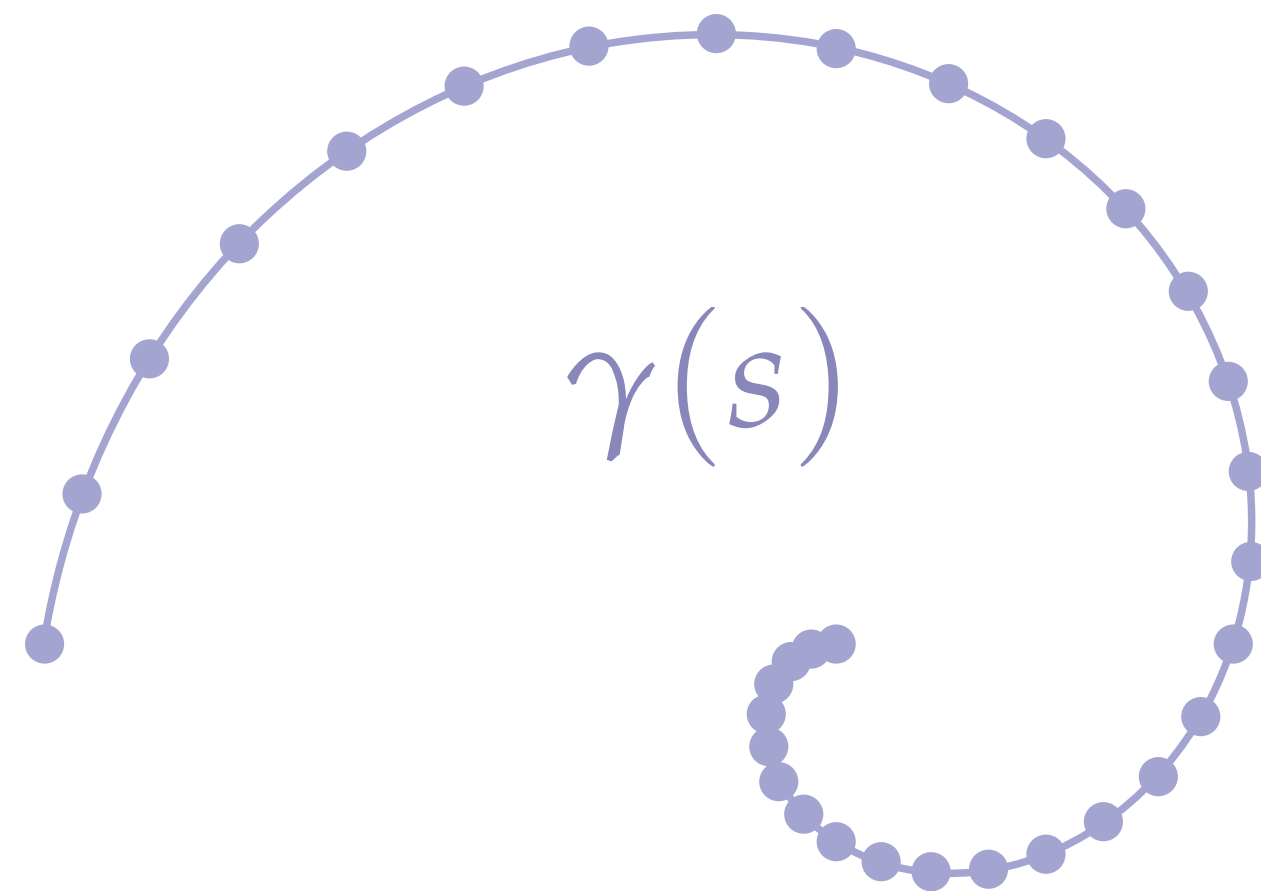


# Reparameterization of a Curve

- We can *reparameterize* a curve  $\gamma : \mathbb{R} \supset I \rightarrow \mathbb{R}^n$  by composing it with a bijection  $\eta : I \rightarrow I$  to obtain a new parameterized curve

$$\tilde{\gamma}(s) := \gamma(\eta(s))$$

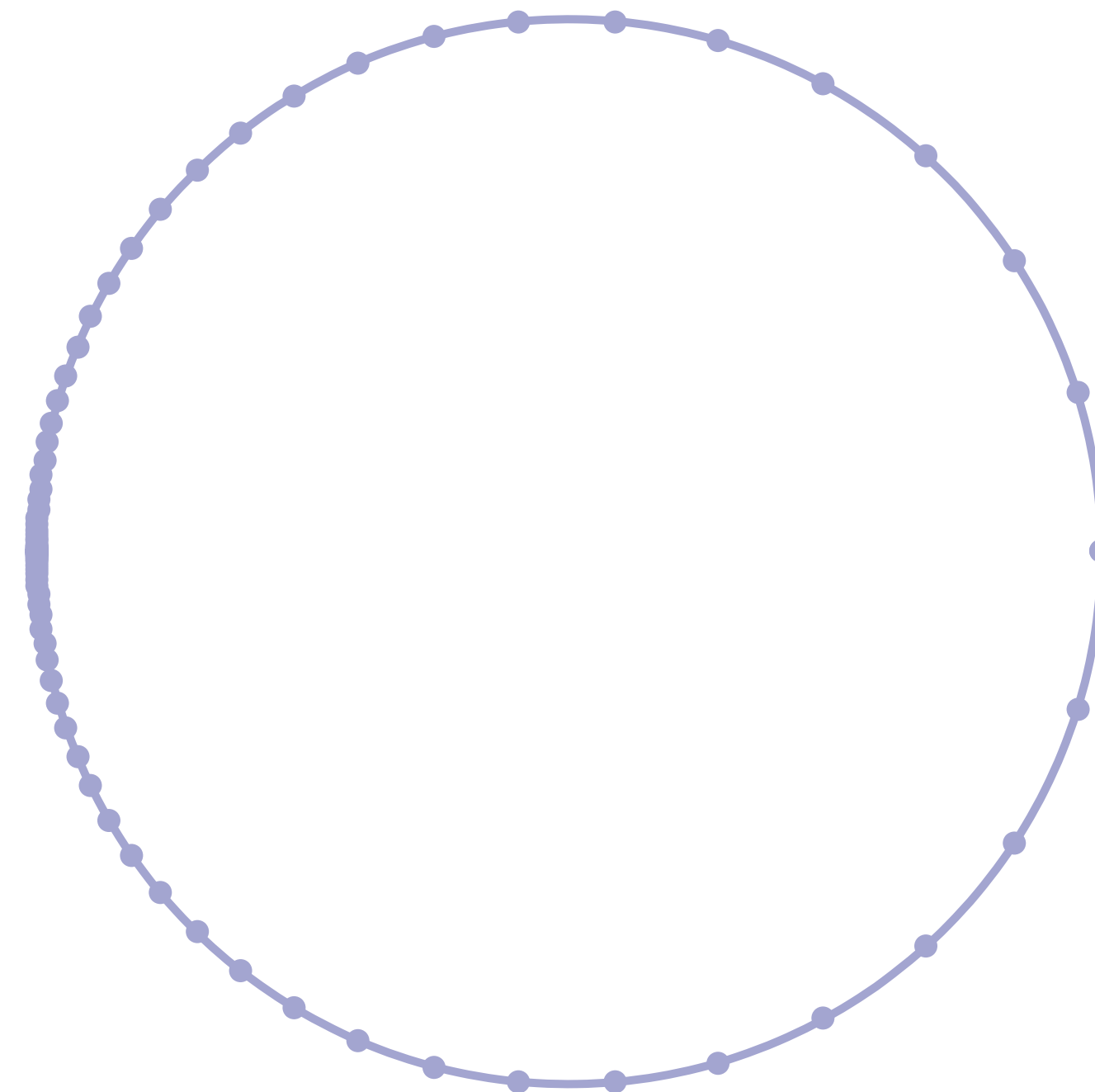
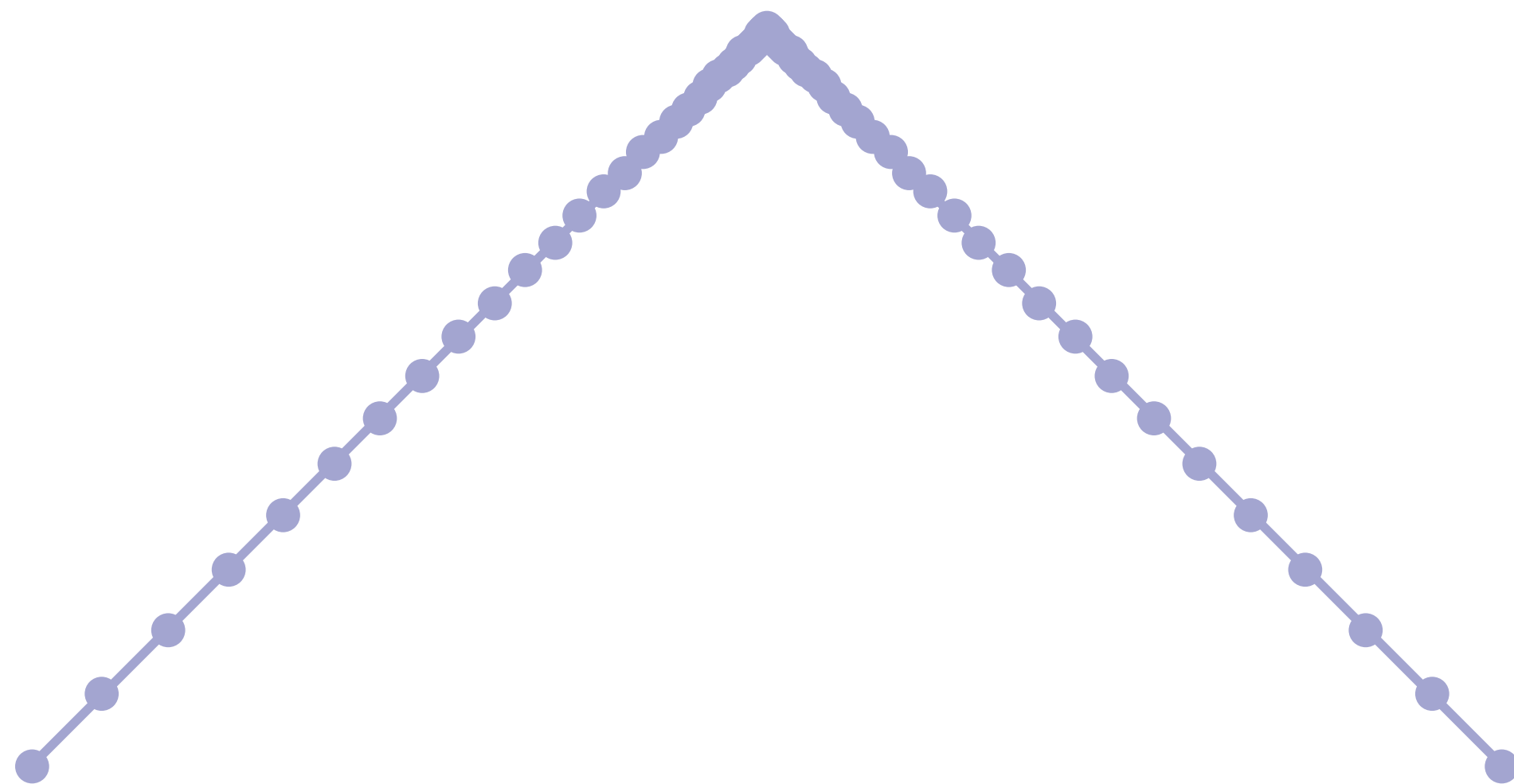
- The *image* of the new curve is the same, even though the map itself changes. For example:



$$\gamma(s) := (1 + s)(\cos(\pi s), \sin(\pi s))$$

# Regular Curve / Immersion

- A parameterized curve is *regular* (or *immersed*) if the differential is nonzero everywhere, *i.e.*, if the curve “never slows to zero”
- Without this condition, a parameterized curve may look non-smooth but actually be differentiable everywhere, or look smooth but fail to have well-defined tangents.

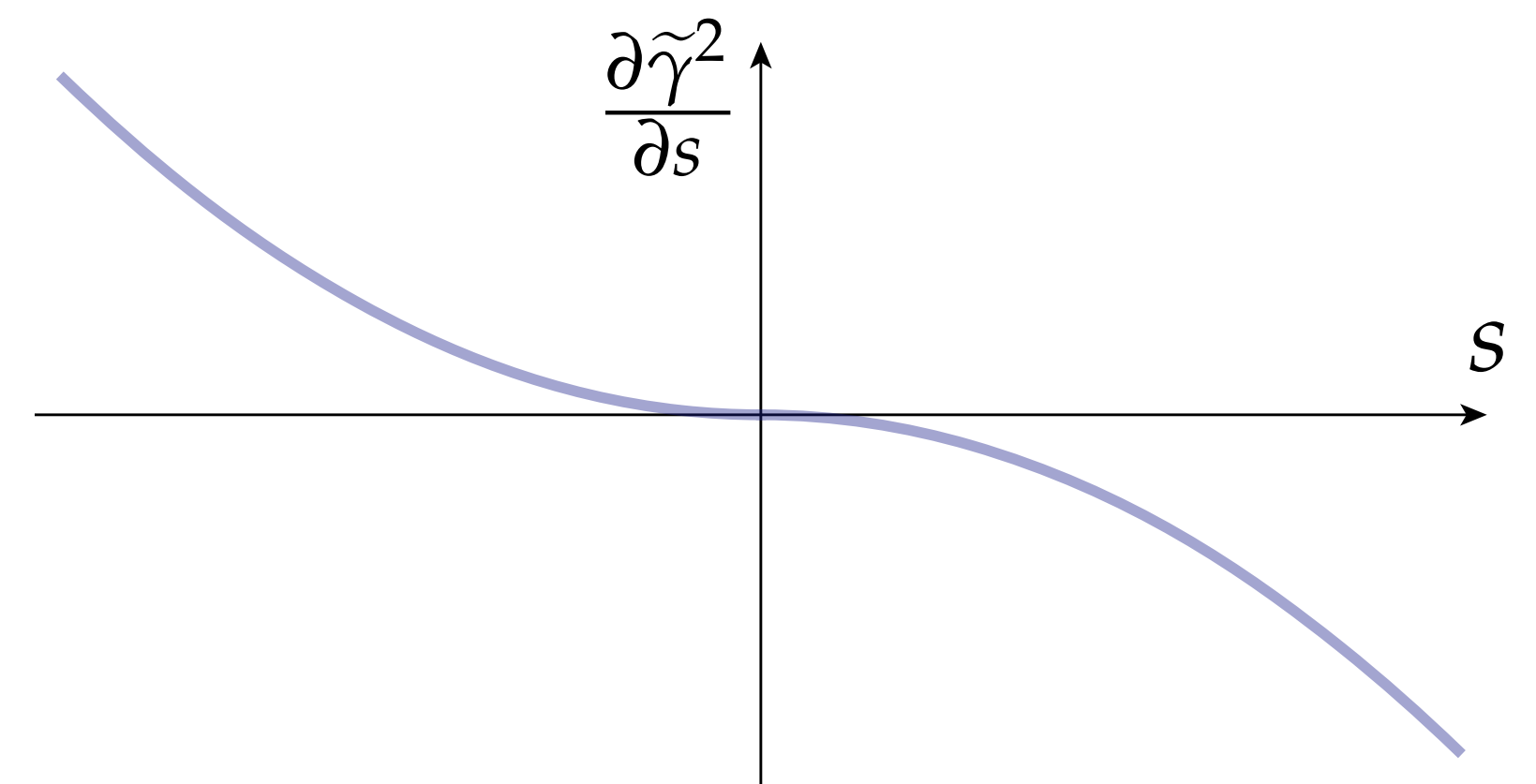
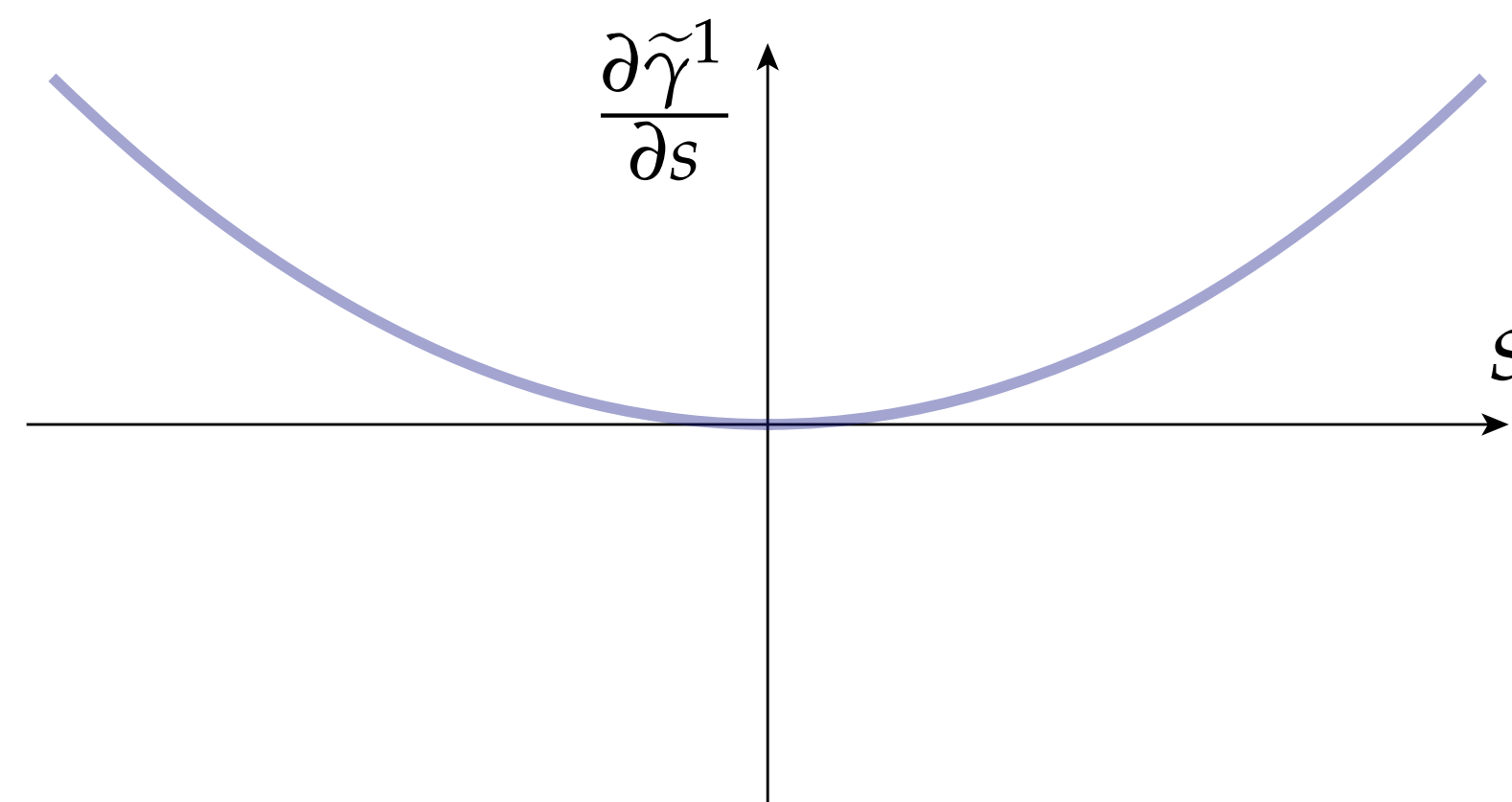
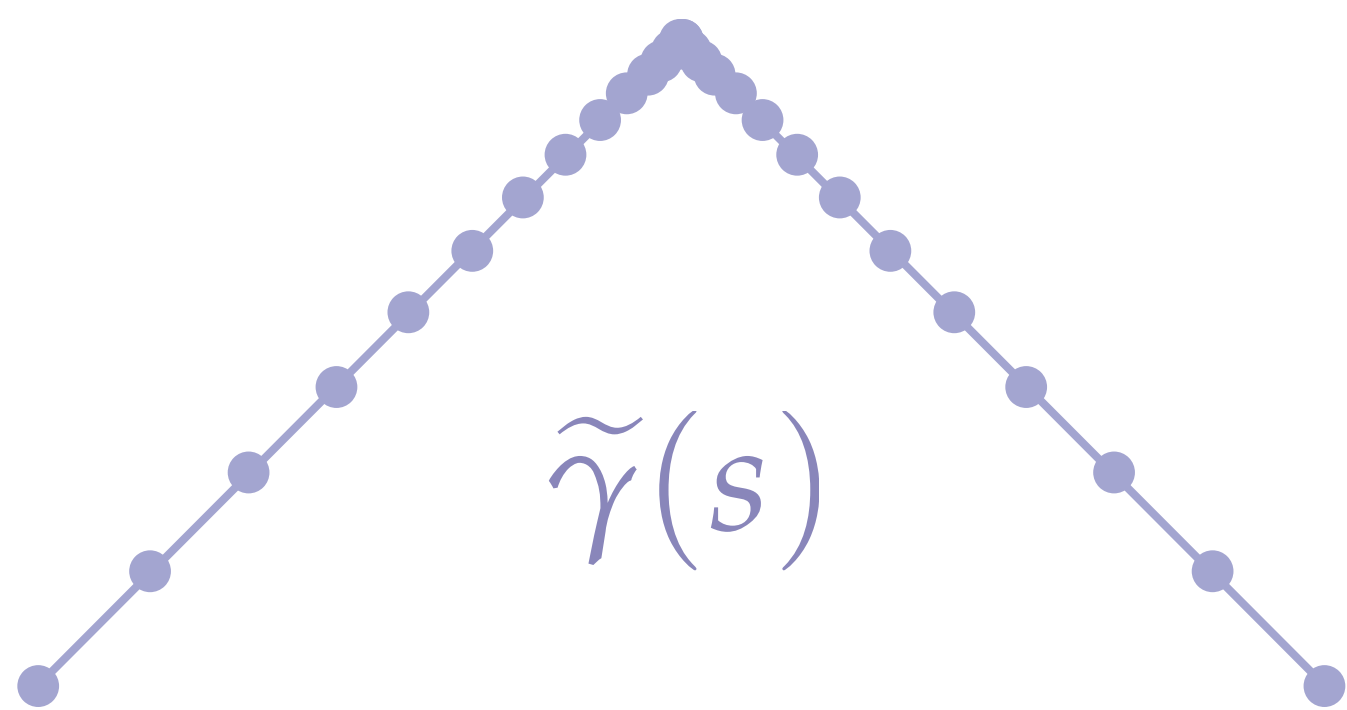


# Irregular Curve—Example

- Consider the reparameterization of a piecewise linear curve:

$$\eta(s) := s^3 \quad \gamma(s) := \begin{cases} (s, s) & s \leq 0 \\ (s, -s) & s > 0 \end{cases} \quad \tilde{\gamma}(s) = \begin{cases} (s^3, s^3) & s \leq 0, \\ (s^3, -s^3) & s > 0 \end{cases}$$

- Even though the reparameterized curve has a continuous first derivative, this derivative goes to zero at  $s = 0$ :



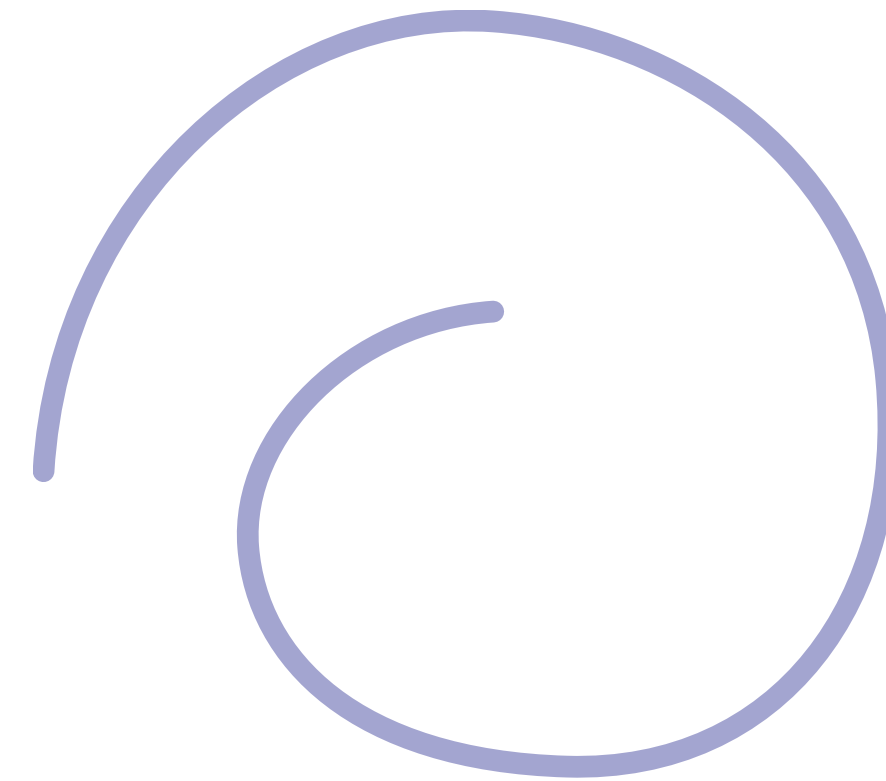
- Hence, (still) can't define tangent at zero.



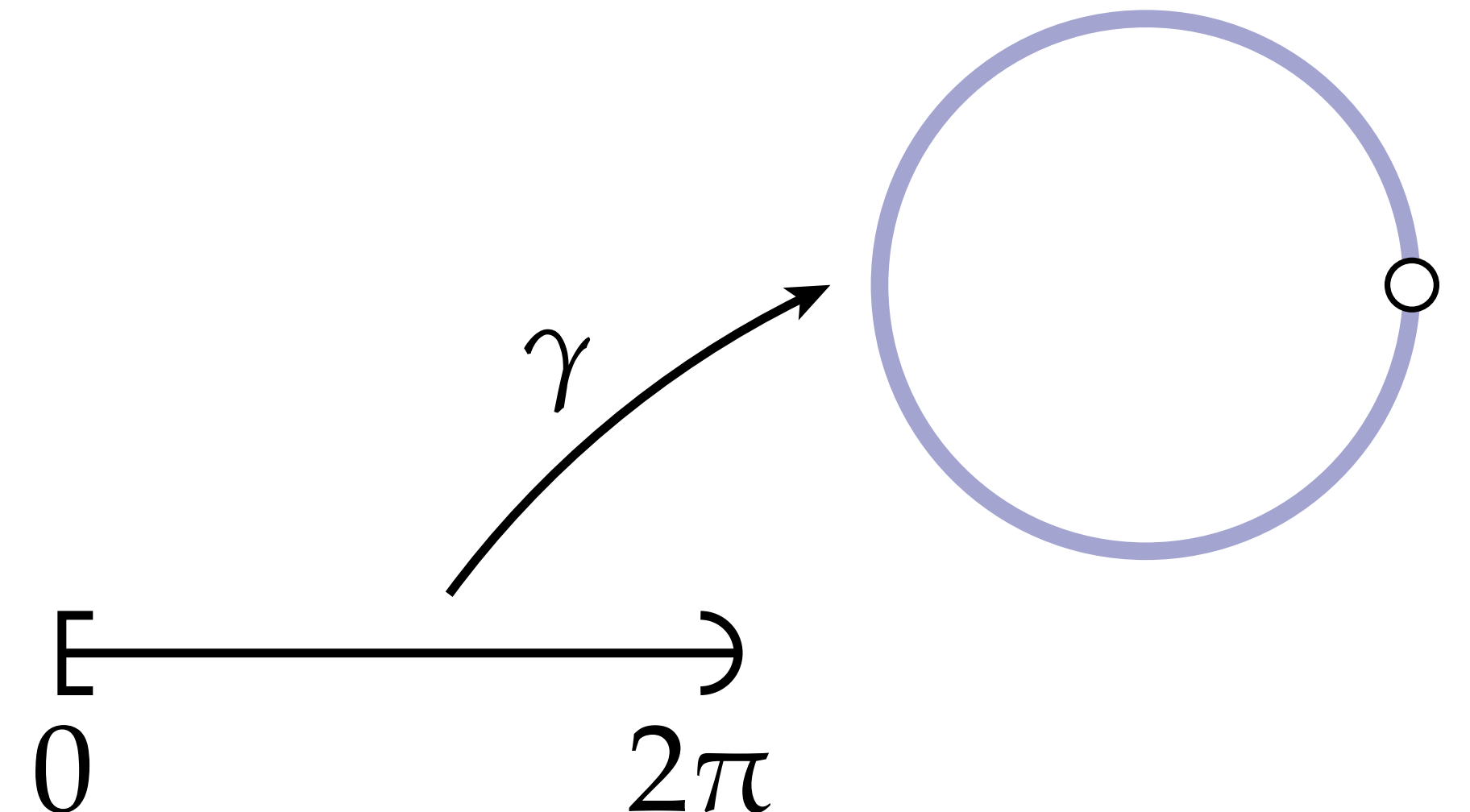
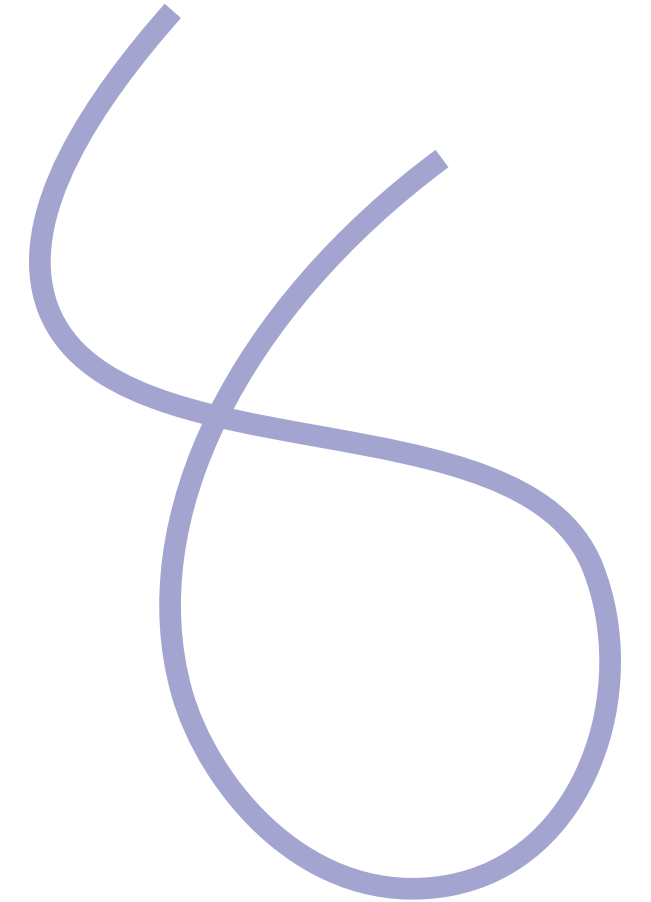
# Embedded Curve

- Roughly speaking, an *embedded* curve does not cross itself
- More precisely, a curve is embedded if it is a continuous and injective map from its domain to its image, and the inverse map is also continuous
- **Q:** What's an example of a continuous injective curve that is not embedded?
- **A:** A half-open interval mapped to a circle (inverse is not continuous)

embedded



not embedded



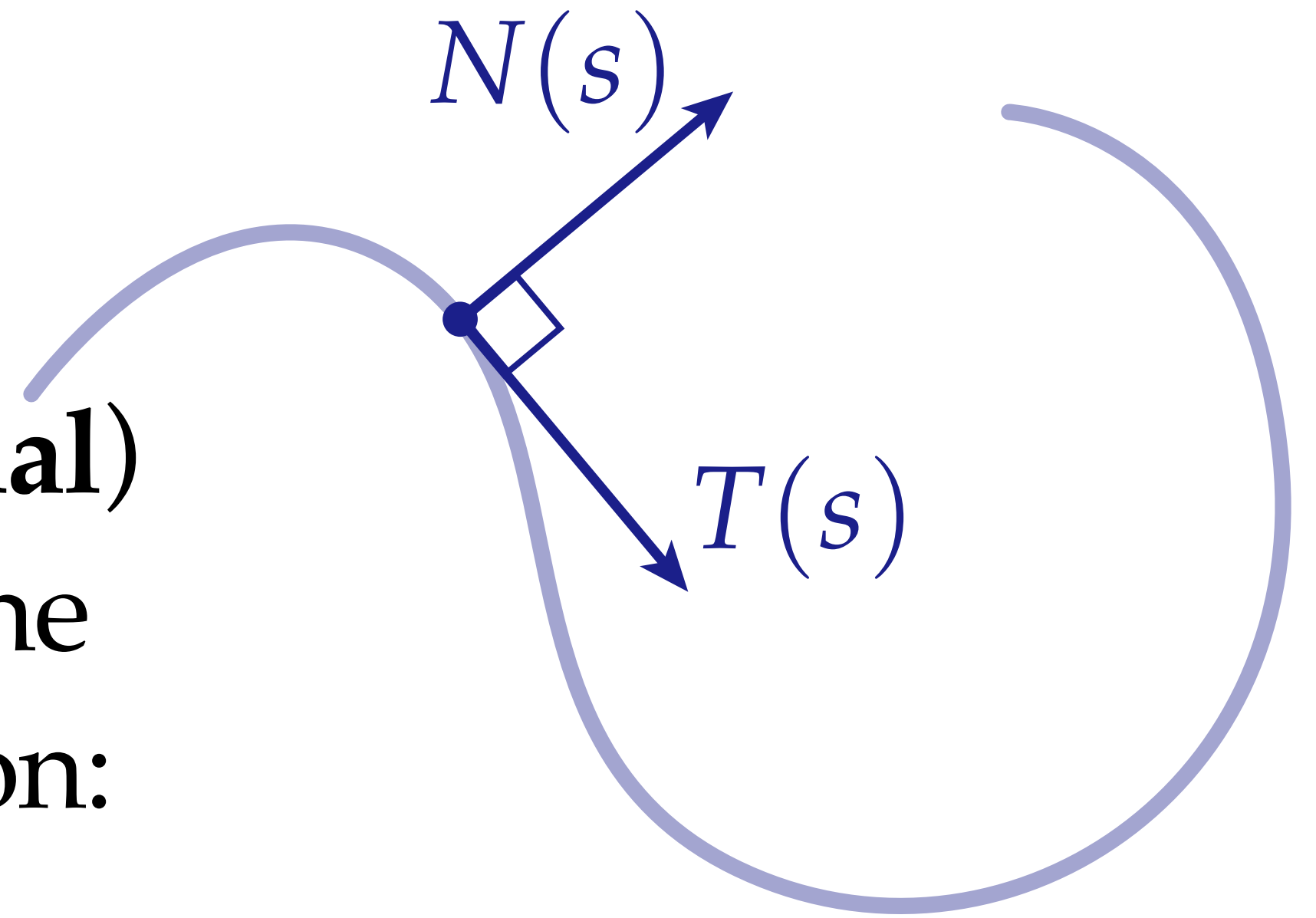
# Normal of a Curve

- 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  $y$ :

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



\*Why does this work?

# Normal of a Curve—Example

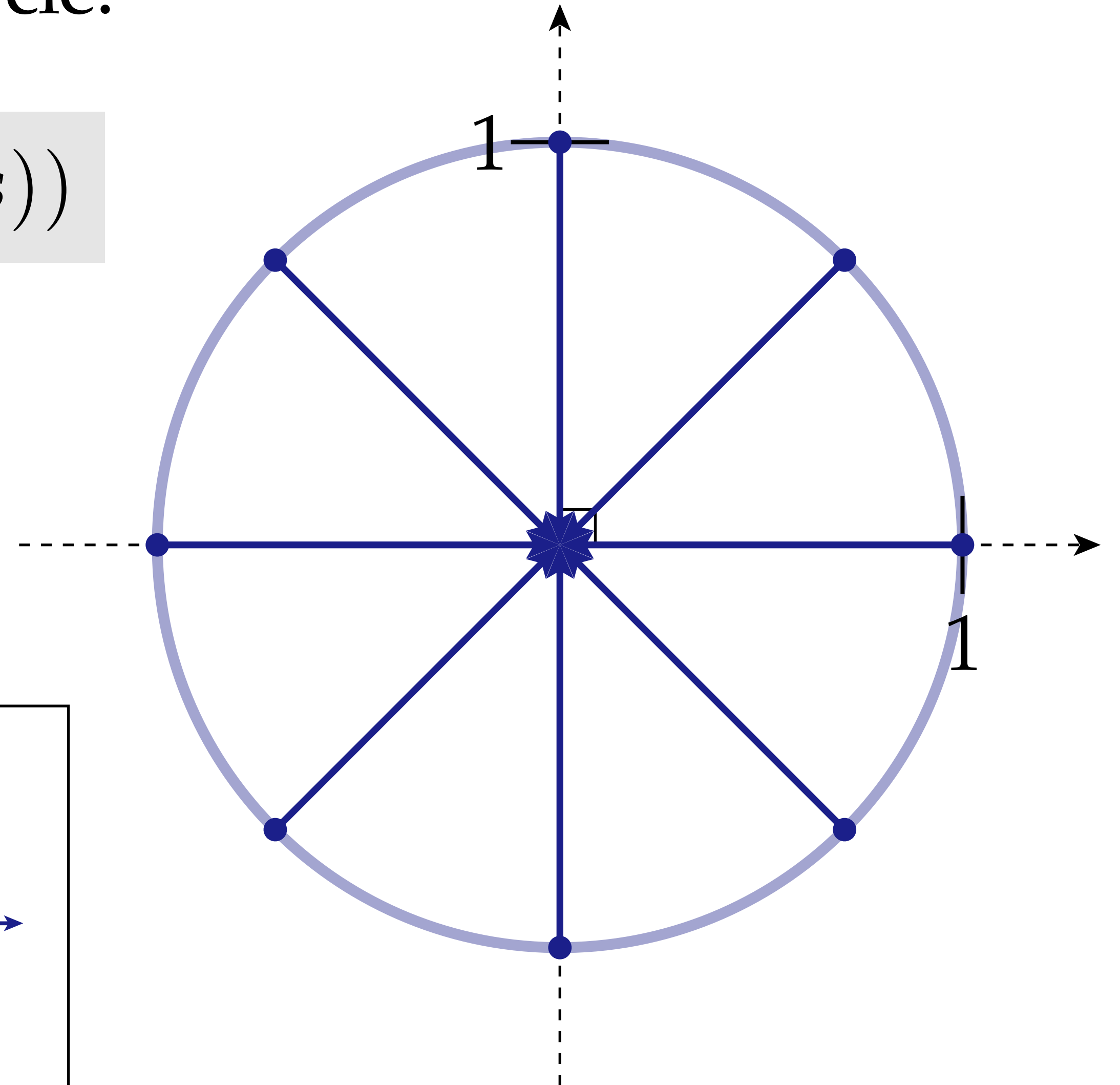
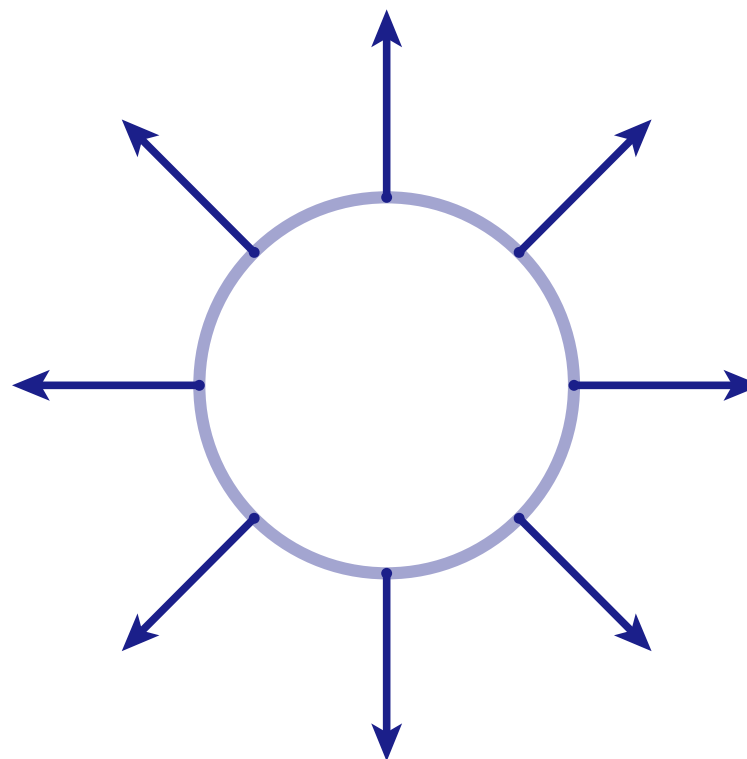
- Let's compute the unit normal of a circle:

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

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

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

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



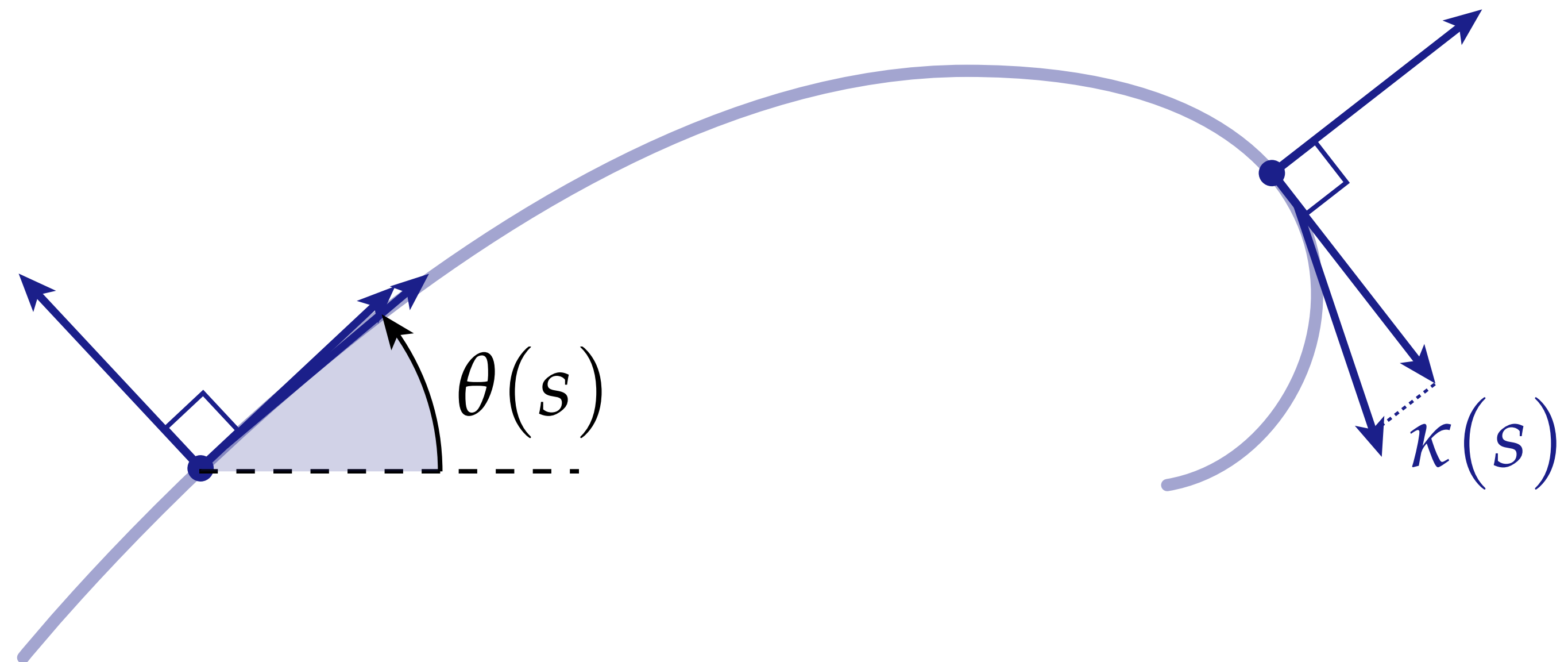
# Curvature of a Plane Curve

- 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

$$\begin{aligned}\kappa(s) &:= \langle N(s), \frac{d}{ds} T(s) \rangle \\ &= \langle N(s), \frac{d^2}{ds^2} \gamma(s) \rangle\end{aligned}$$

Equivalently:

$$\kappa(s) = \frac{d}{ds} \theta(s)$$



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



# *Fundamental Theorem of Plane Curves*

**Fact.** Up to rigid motions, an arc-length parameterized plane curve is uniquely determined by its curvature.

**Q:** Given only the curvature function, how can we recover the curve?

**A:** Just “invert” the two relationships  $\frac{d}{ds}\theta = \kappa$ ,  $\frac{d}{ds}\gamma = T$

*First integrate curvature to get angle:  $\theta(s) := \int_0^s \kappa(t) dt$*

*Then evaluate unit tangents:  $T(s) := (\cos(\theta), \sin(\theta))$*

*Finally, integrate tangents to get curve:  $\gamma(s) := \int_0^s T(t) dt$*

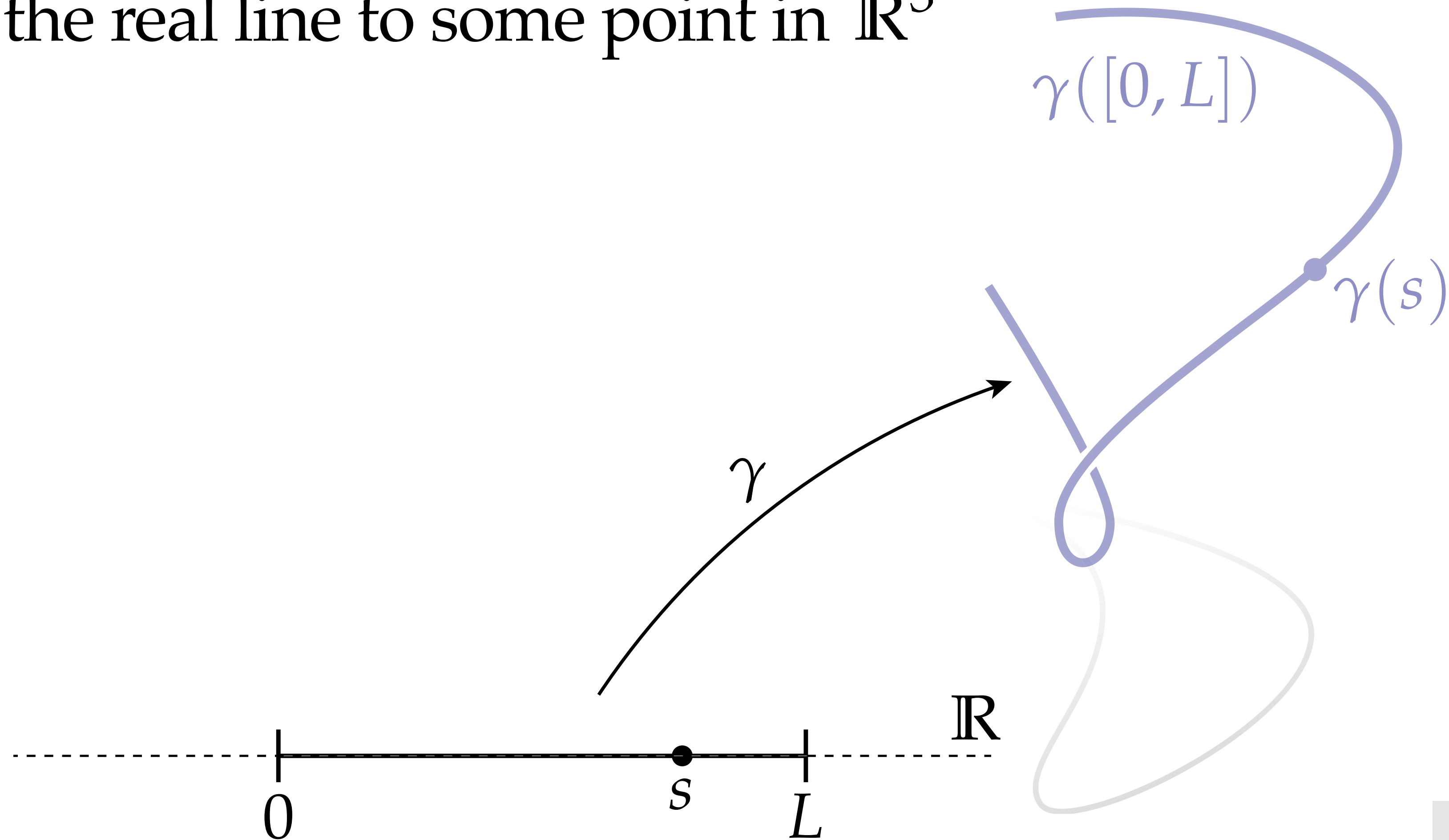
**Q:** What about the rigid motion? Will this work for *closed* curves?



# *Space Curves*

# Parameterized Space Curve

- A **parameterized space curve** is a map\* taking each point in an interval  $[0, L]$  of the real line to some point in  $\mathbb{R}^3$



\*Continuous, differentiable, smooth...

$$\gamma : [0, L] \rightarrow \mathbb{R}^3$$

# Pushforward of Vectors on a Space Curve

Suppose we apply the differential of a parameterized space curve to a vector field  $X$  on its domain:

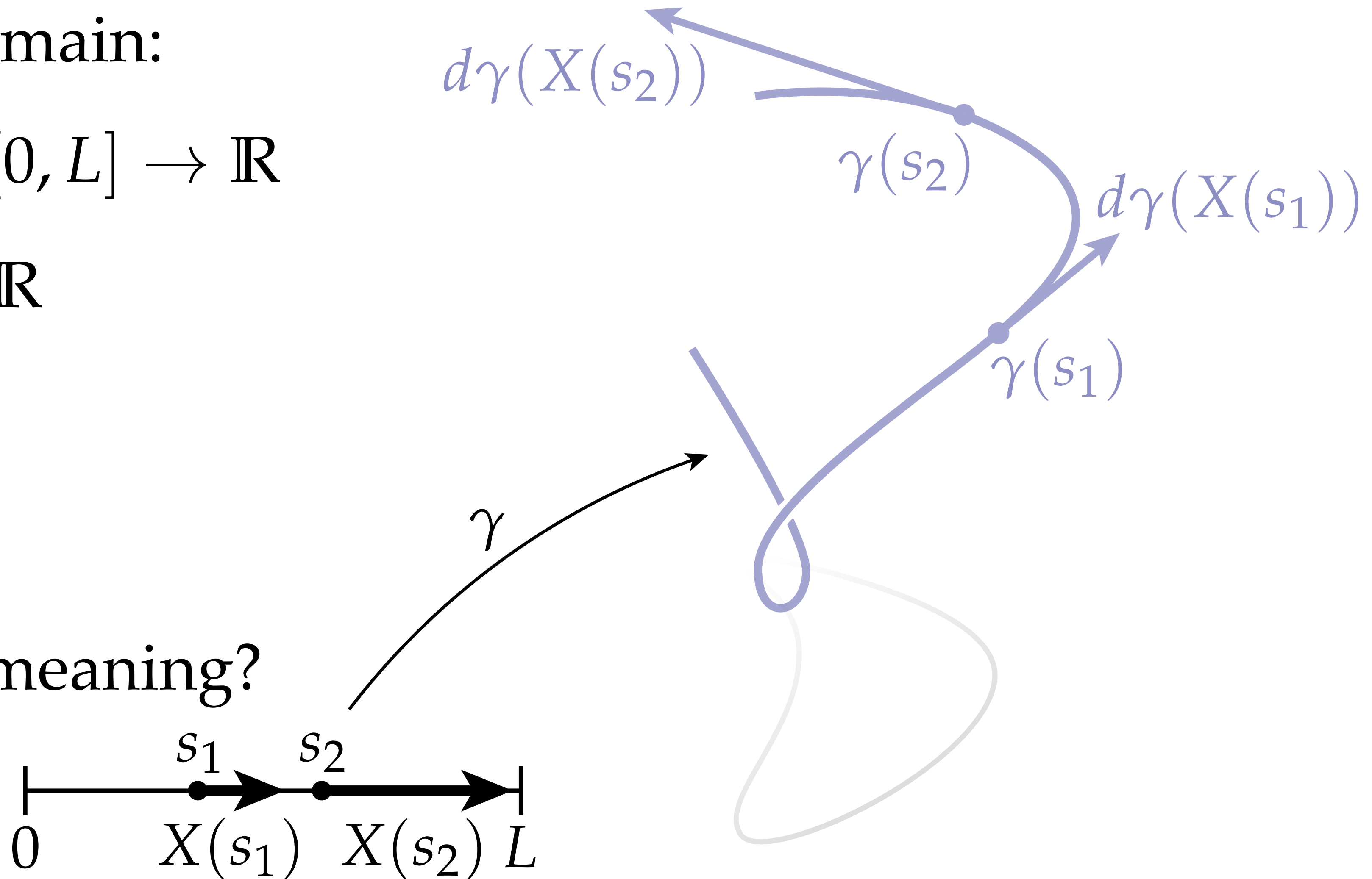
$$\gamma := (x, y, z), \quad x, y, z : [0, L] \rightarrow \mathbb{R}$$

$$X := a \frac{\partial}{\partial s}, \quad a : [0, L] \rightarrow \mathbb{R}$$

$$d\gamma = \left( \frac{\partial x}{\partial s}, \frac{\partial y}{\partial s}, \frac{\partial z}{\partial s} \right) ds$$

$$d\gamma(X) = a \left( \frac{\partial x}{\partial s}, \frac{\partial y}{\partial s}, \frac{\partial z}{\partial s} \right)$$

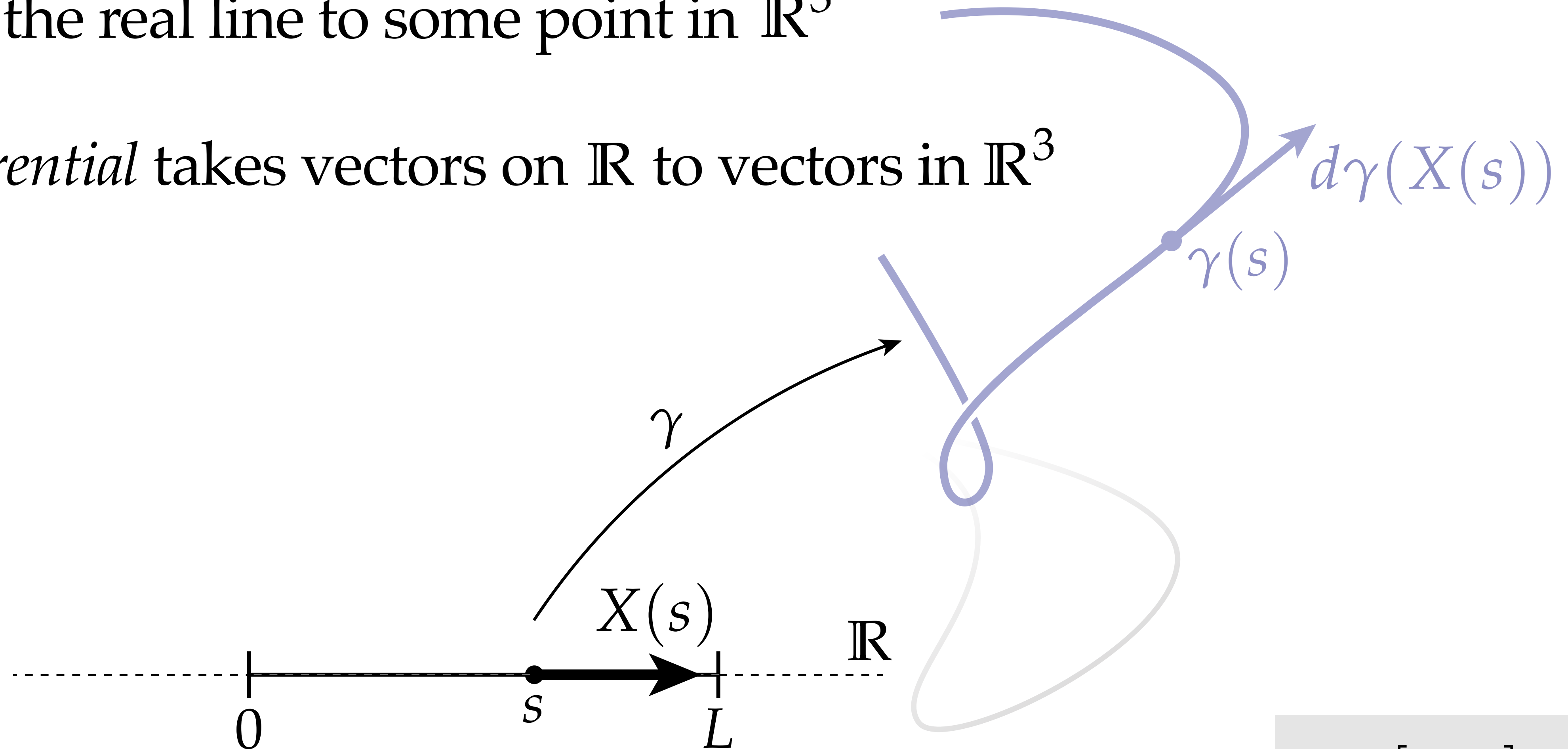
**Q:** What's the *geometric* meaning?





# Parameterized Space Curve

- A **parameterized space curve** is a map\* taking each point in an interval  $[0, L]$  of the real line to some point in  $\mathbb{R}^3$
- Its *differential* takes vectors on  $\mathbb{R}$  to vectors in  $\mathbb{R}^3$

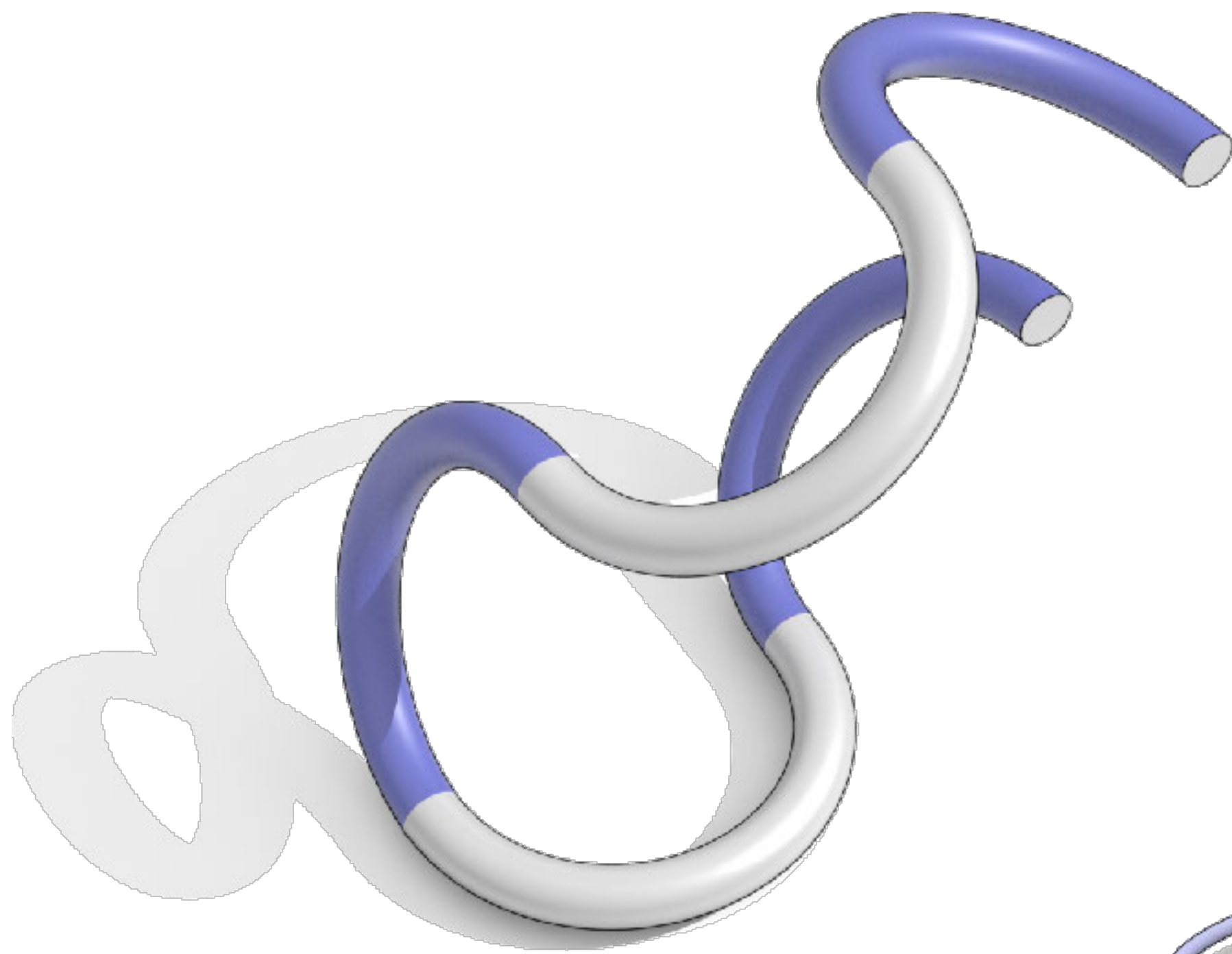


\*Continuous, differentiable, smooth...

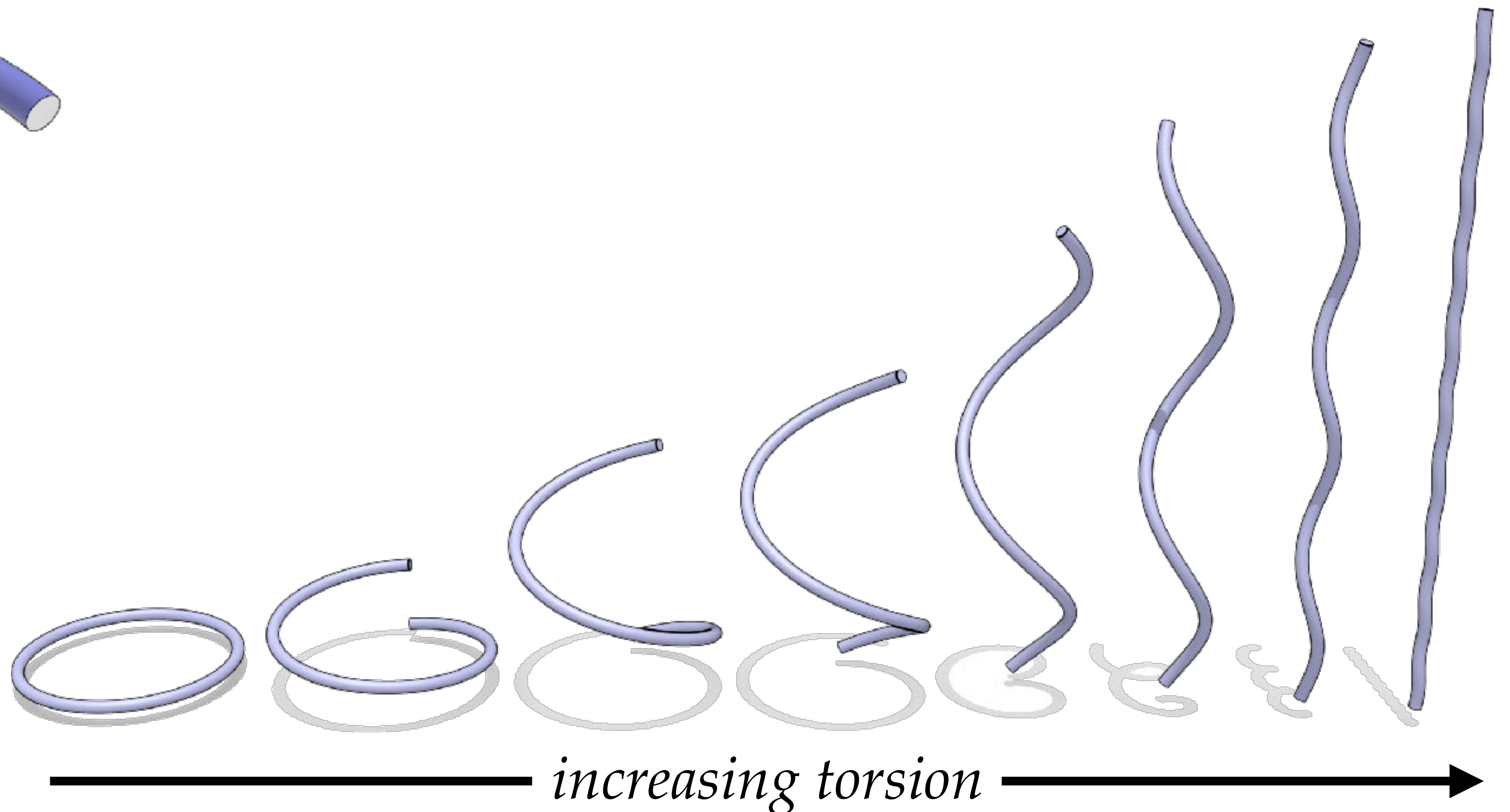
$$\gamma : [0, L] \rightarrow \mathbb{R}^3$$

# Curvature and Torsion of a Space Curve

- For a plane curve, *curvature* captured the notion of “bending”
- For a space curve we also have *torsion*, which captures “twisting”



**Intuition:** torsion is  
“out of plane bending”



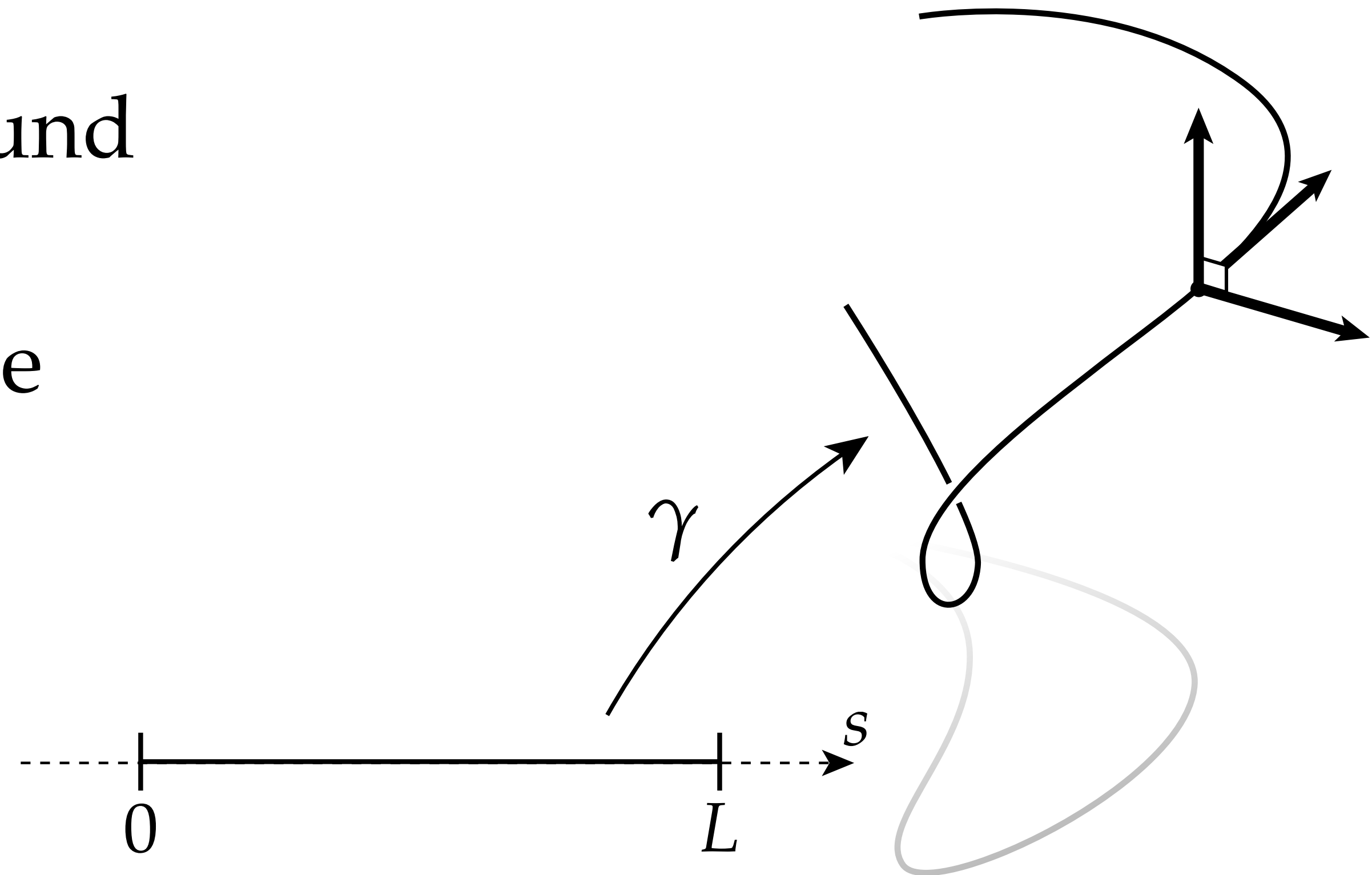
# Frenet Frame

- Each point of a space curve has a natural coordinate frame called the *Frenet frame*, which depends only on the local geometry
- As in the plane, the tangent  $T$  is found by differentiating the curve, and differentiating the tangent yields the curvature times the normal  $N$
- The binormal  $B$  then completes an orthonormal basis with  $T$  and  $N$

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

$$N(s) := \frac{d}{ds} T / \left| \frac{d}{ds} T \right|$$

$$B(s) := T(s) \times N(s)$$



# Frenet-Serret Equation

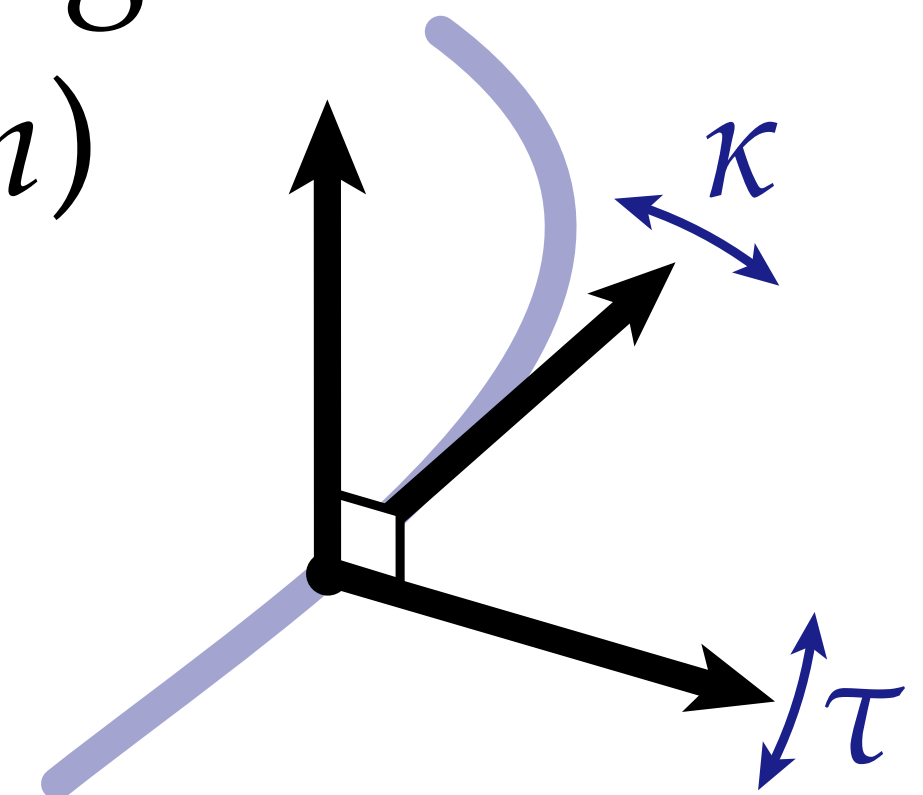
- Curvature  $\kappa$  and torsion  $\tau$  can be defined in terms of the change in the Frenet frame as we move along the curve:

$$\frac{d}{ds} \begin{bmatrix} T \\ N \\ B \end{bmatrix} = \begin{bmatrix} 0 & -\kappa & 0 \\ \kappa & 0 & -\tau \\ 0 & \tau & 0 \end{bmatrix} \begin{bmatrix} T \\ N \\ B \end{bmatrix}$$

- Most importantly, change in the tangent describes bending (*curvature*); change in binormal describes twisting (*torsion*)

$$\kappa = -\langle N, \frac{d}{ds} T \rangle$$

$$\tau = \langle N, \frac{d}{ds} B \rangle$$





# Example—Helix

- Let's compute the Frenet frame, curvature, and torsion for a *helix*\*

$$\gamma(s) := (a \cos(s), a \sin(s), bs)$$

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

$$\left| \frac{d}{ds}\gamma \right| = \sqrt{a^2 + b^2} = 1$$

$$\Rightarrow T(s) = \frac{d}{ds}\gamma(s)$$

$$\frac{d}{ds}T(s) = -a(\cos(s), \sin(s), 0)$$

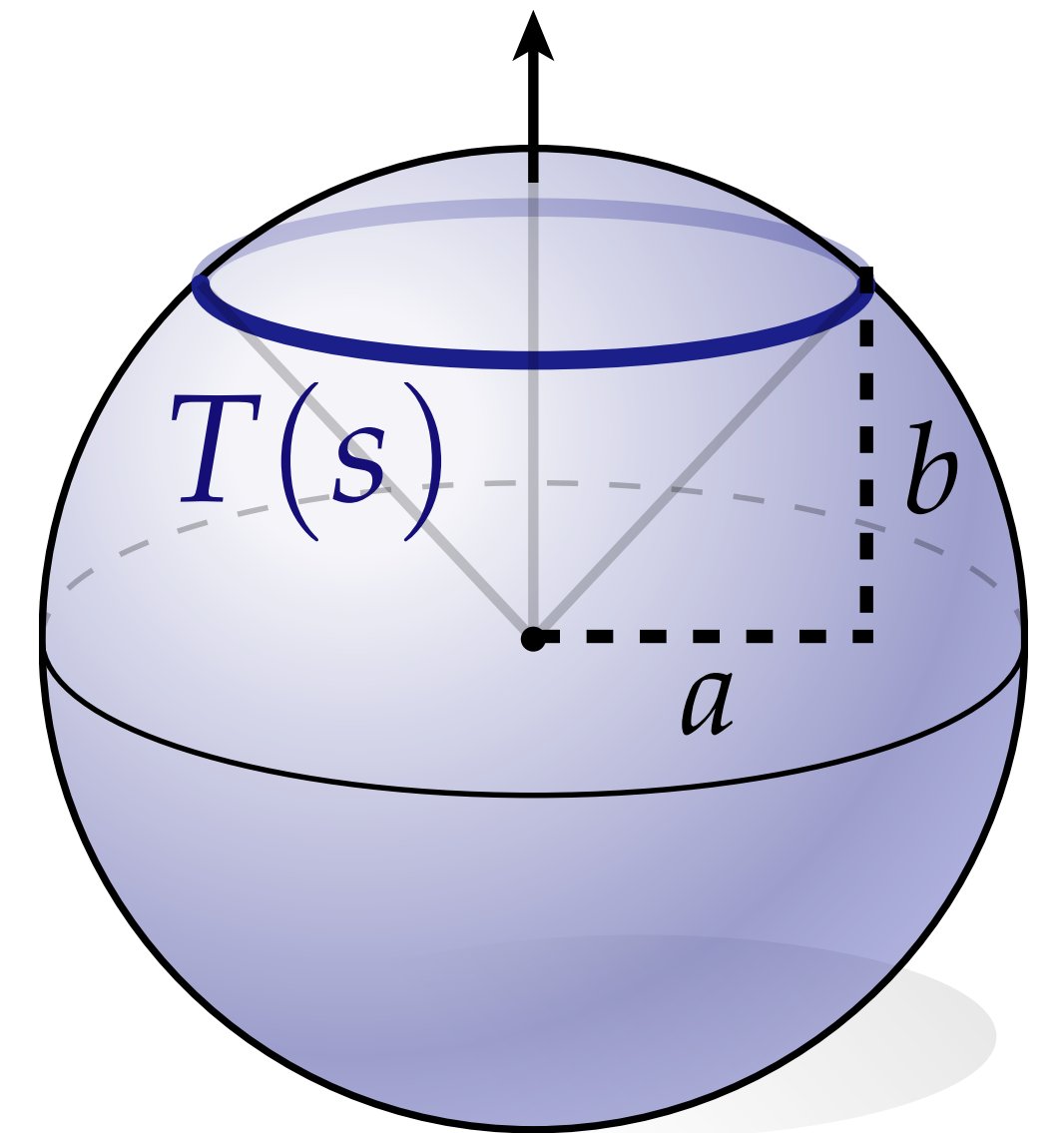
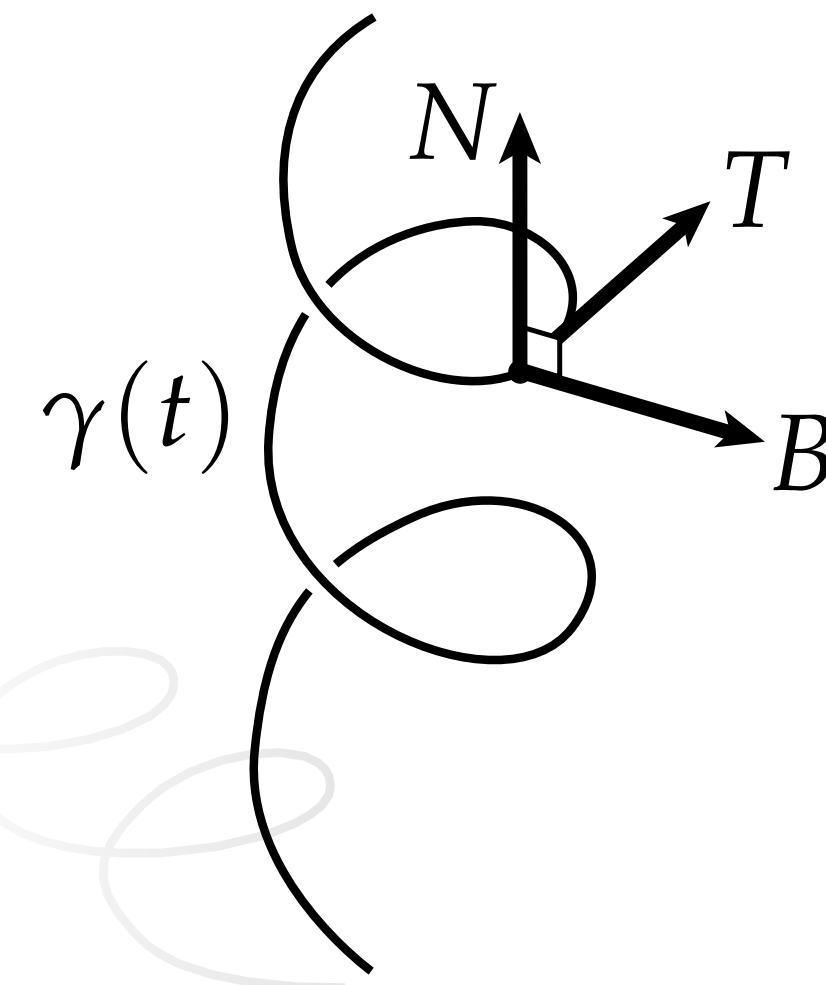
$$\Rightarrow \kappa(s) = -a, \quad N(s) = (\cos(s), \sin(s), 0)$$

$$B(s) = T(s) \times N(s) =$$

$$(-b \sin(s), b \cos(s), -a)$$

$$\frac{d}{ds}B(s) = -b(\cos(s), \sin(s), 0)$$

$$\Rightarrow \tau(s) = -b$$

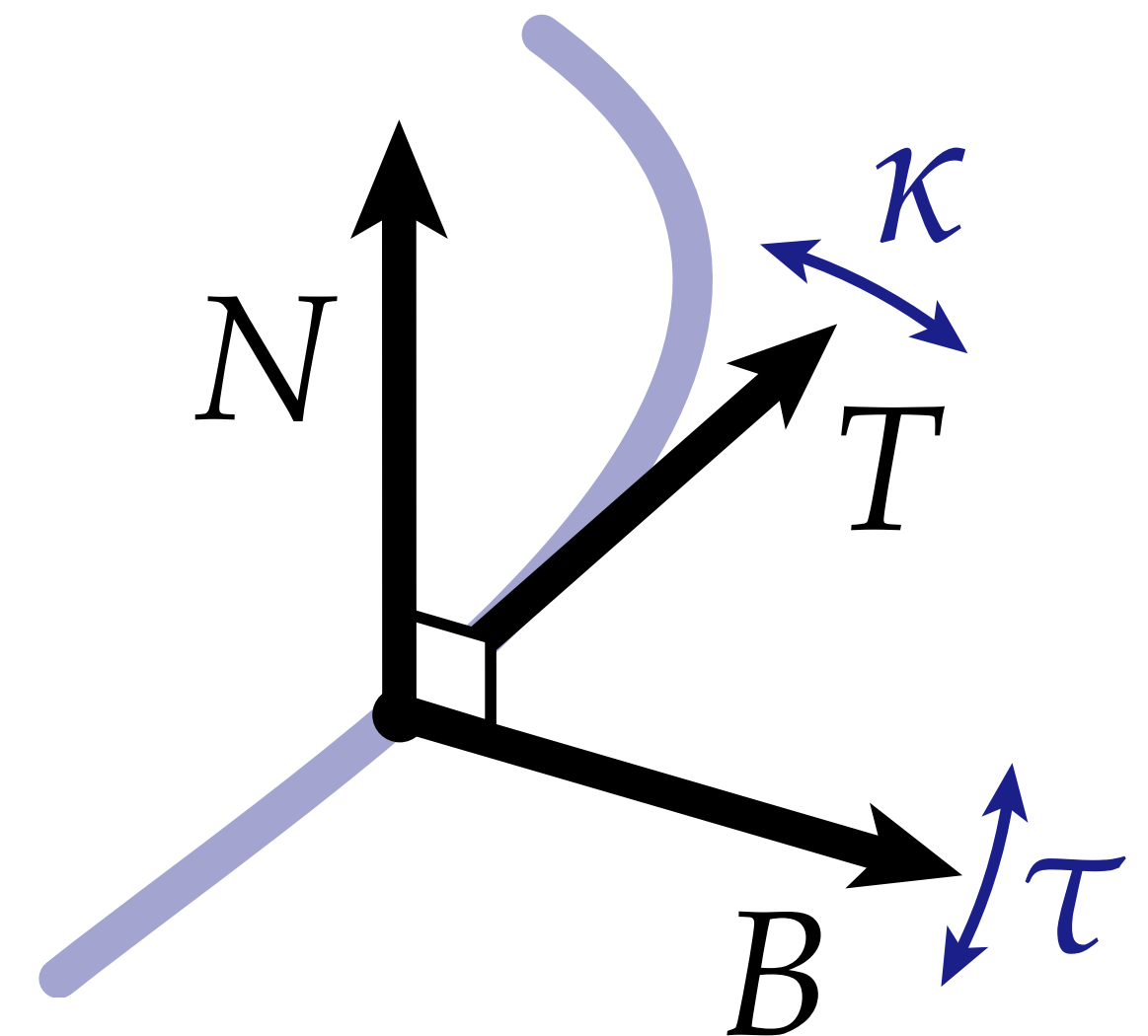


\*For simplicity, let's pick  $a, b$  such that  $a^2 + b^2 = 1$ .

# Fundamental Theorem of Space Curves TODO

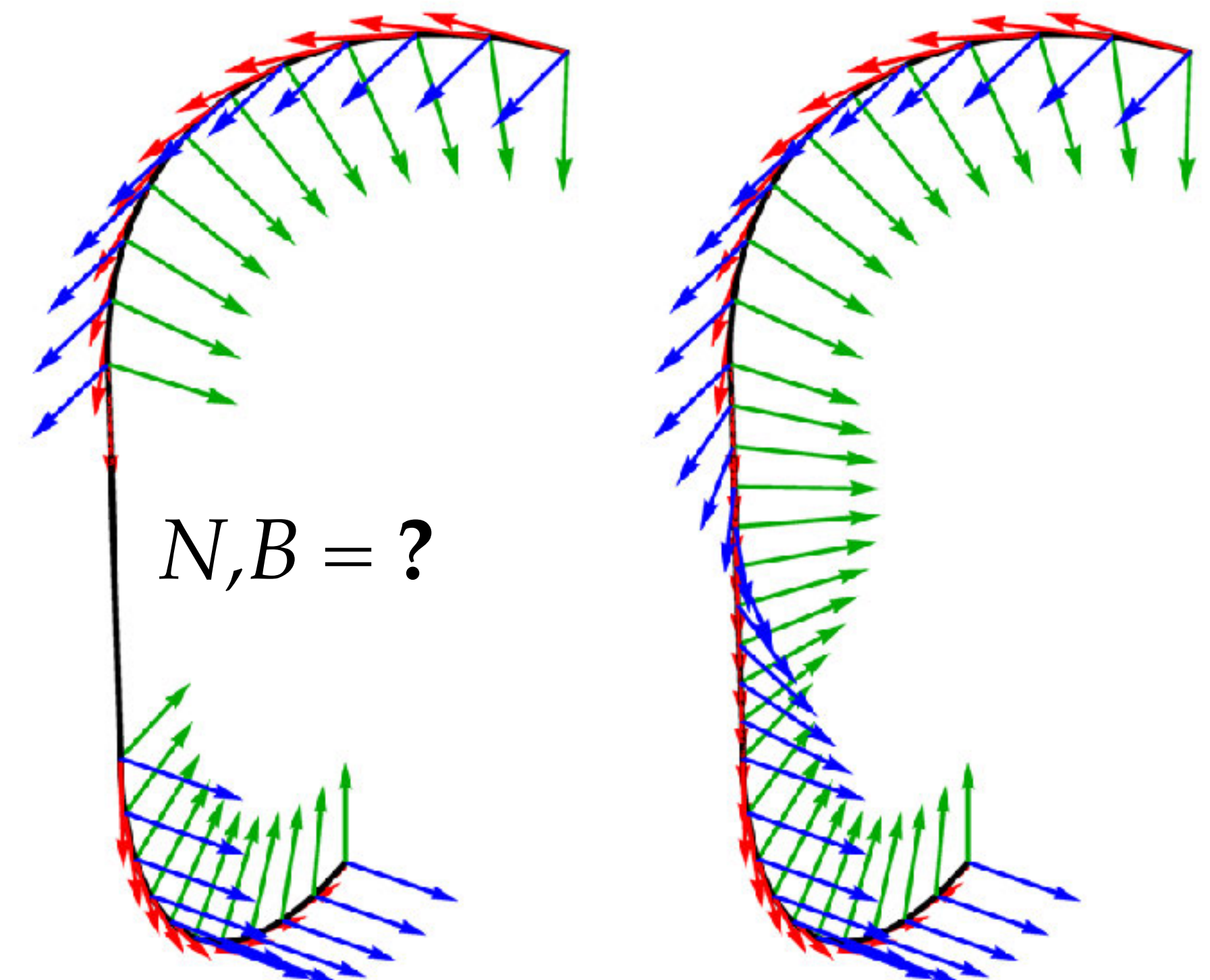
- The *fundamental theorem of space curves* tells us we can also go the other way: given the curvature and torsion of an arc-length parameterized space curve, we can recover the curve itself
- In 2D we just had to integrate a single ODE; here we integrate a system of three ODEs—namely, Frenet-Serret!

$$\frac{d}{ds} \begin{bmatrix} T \\ N \\ B \end{bmatrix} = \begin{bmatrix} 0 & -\kappa & 0 \\ \kappa & 0 & -\tau \\ 0 & \tau & 0 \end{bmatrix} \begin{bmatrix} T \\ N \\ B \end{bmatrix}$$



# Adapted Frames on Curves

- **Q:** If our curve has a straight piece, is the Frenet frame well-defined?
- **A:** No, we don't have a clear normal / binormal (since, *e.g.*,  $dT/ds = 0$ )
- However, there are many ways to choose an *adapted frame*
- Any orthonormal frame including  $T$
- *E.g.*, *least-twisting* frame (Bishop)
  - Unlike Frenet, *global* rather than *local*
- First example of *moving frames*
- (Will see more later for surfaces...)





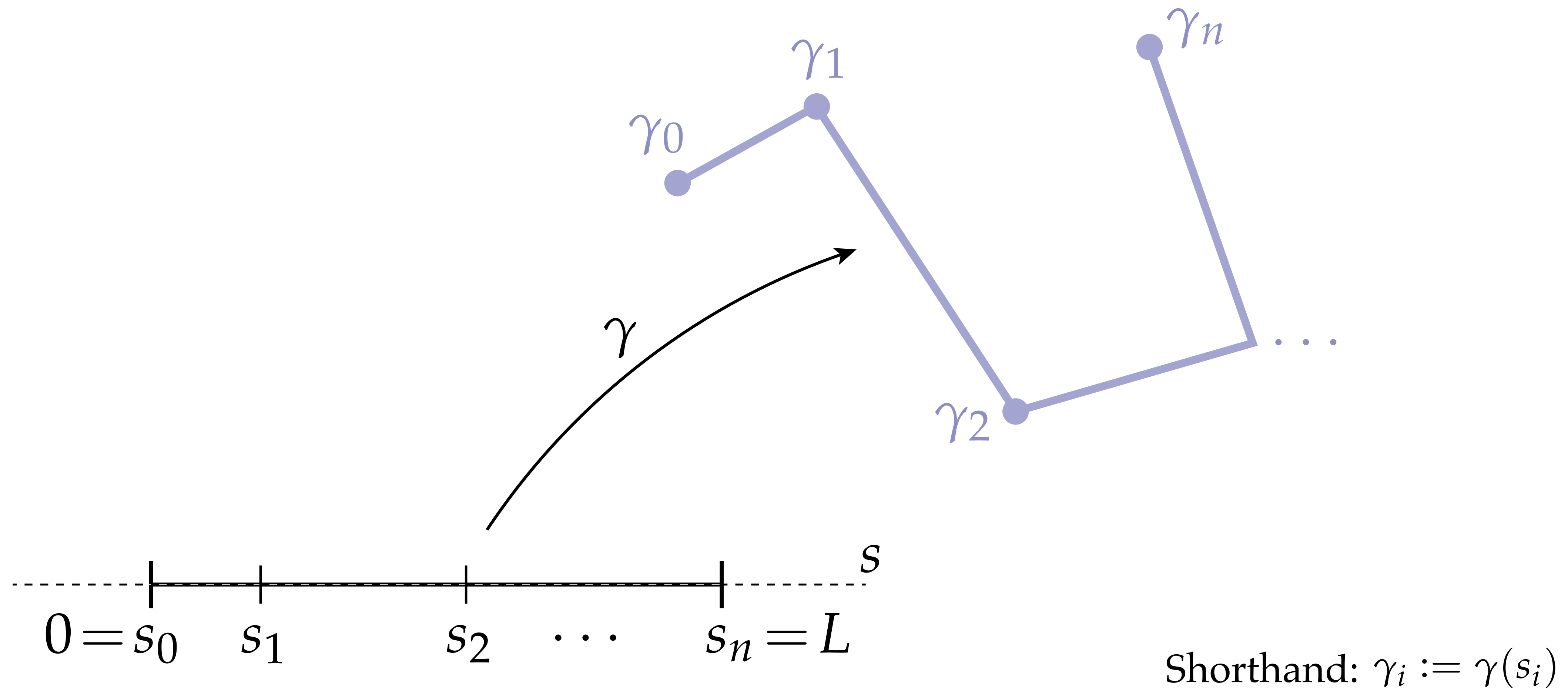
The background features a series of overlapping, curved lines that create a grid-like pattern. These lines are light gray and curve across the frame. A prominent white horizontal band runs through the center, providing a high-contrast area for the text. The overall color palette is a range of light blues and purples, with the white band acting as a focal point.

# *Discrete Curves*



# Discrete Curves in the Plane

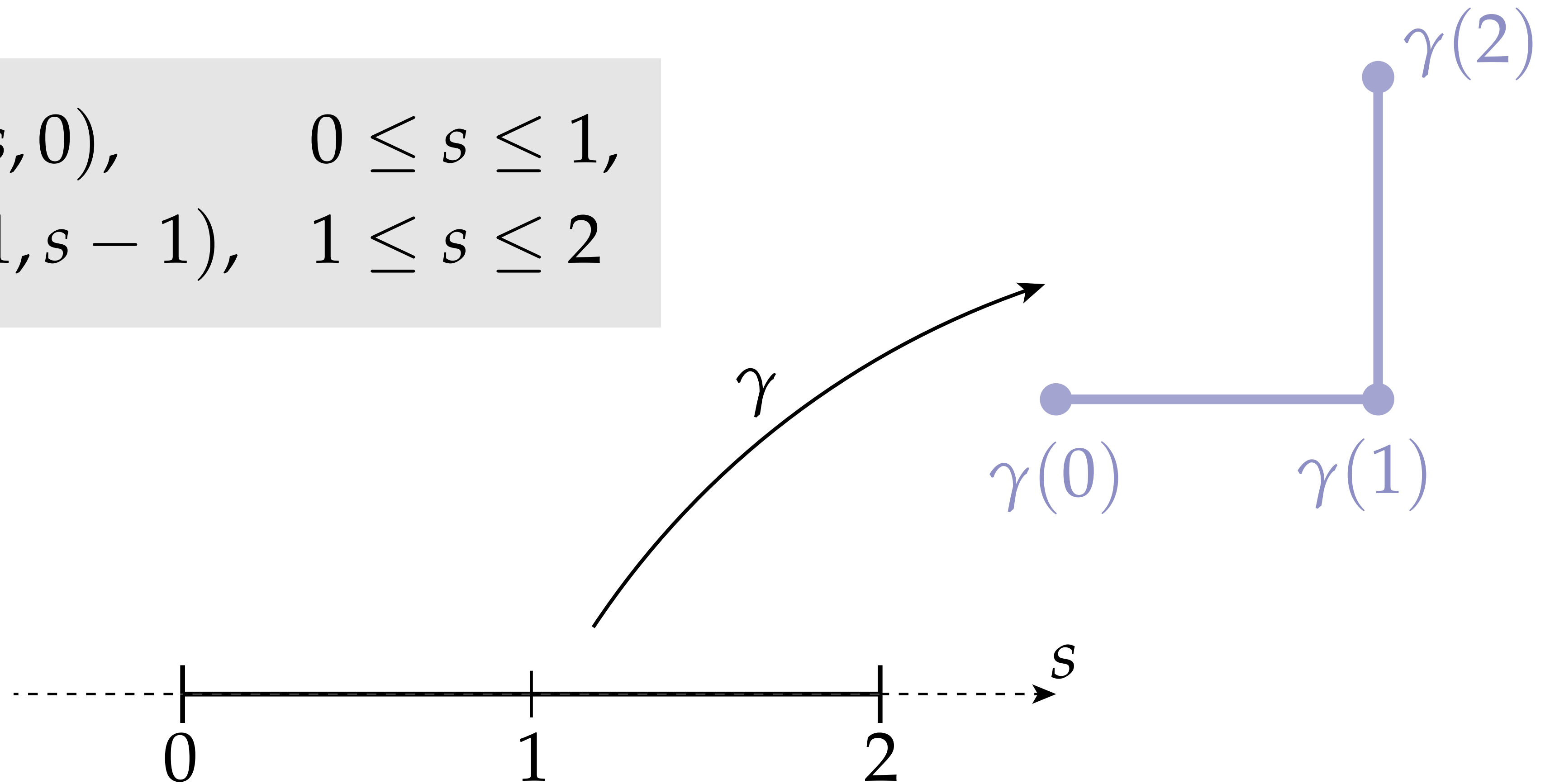
- We'll define a **discrete curve** as a *piecewise linear* parameterized curve, *i.e.*, a sequence of points connected by straight line segments:



# Discrete Curves in the Plane—Example

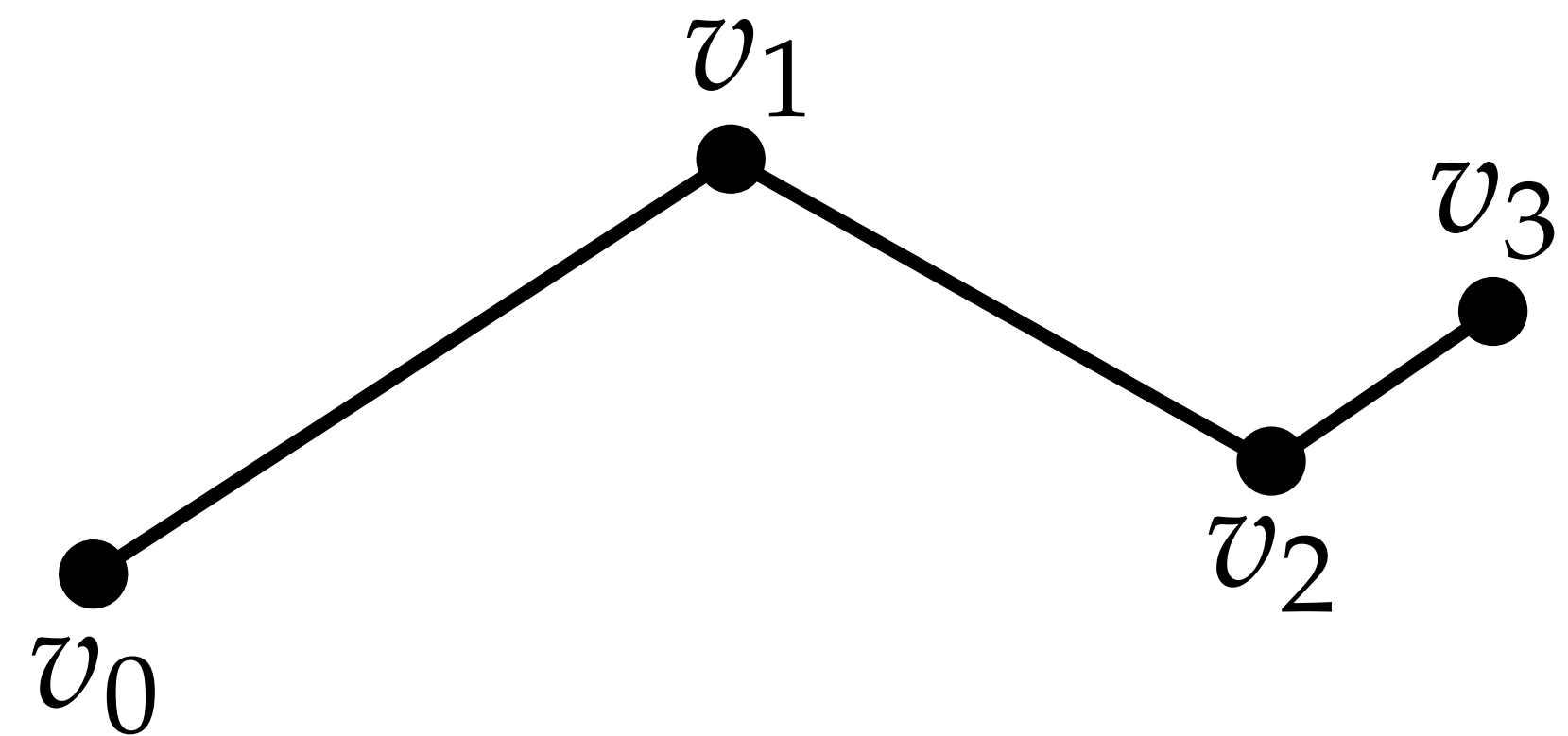
- A simple example is a curve comprised of two segments:

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



# Discrete Curves and Discrete Differential Forms

- Equivalently, a discrete curve is determined by a discrete,  $R^n$ -valued 0-form on a manifold simplicial 1-complex
- The 0-form values give the location of the vertices; interpolation by Whitney bases (hat functions) gives the map from each edge to  $R^n$

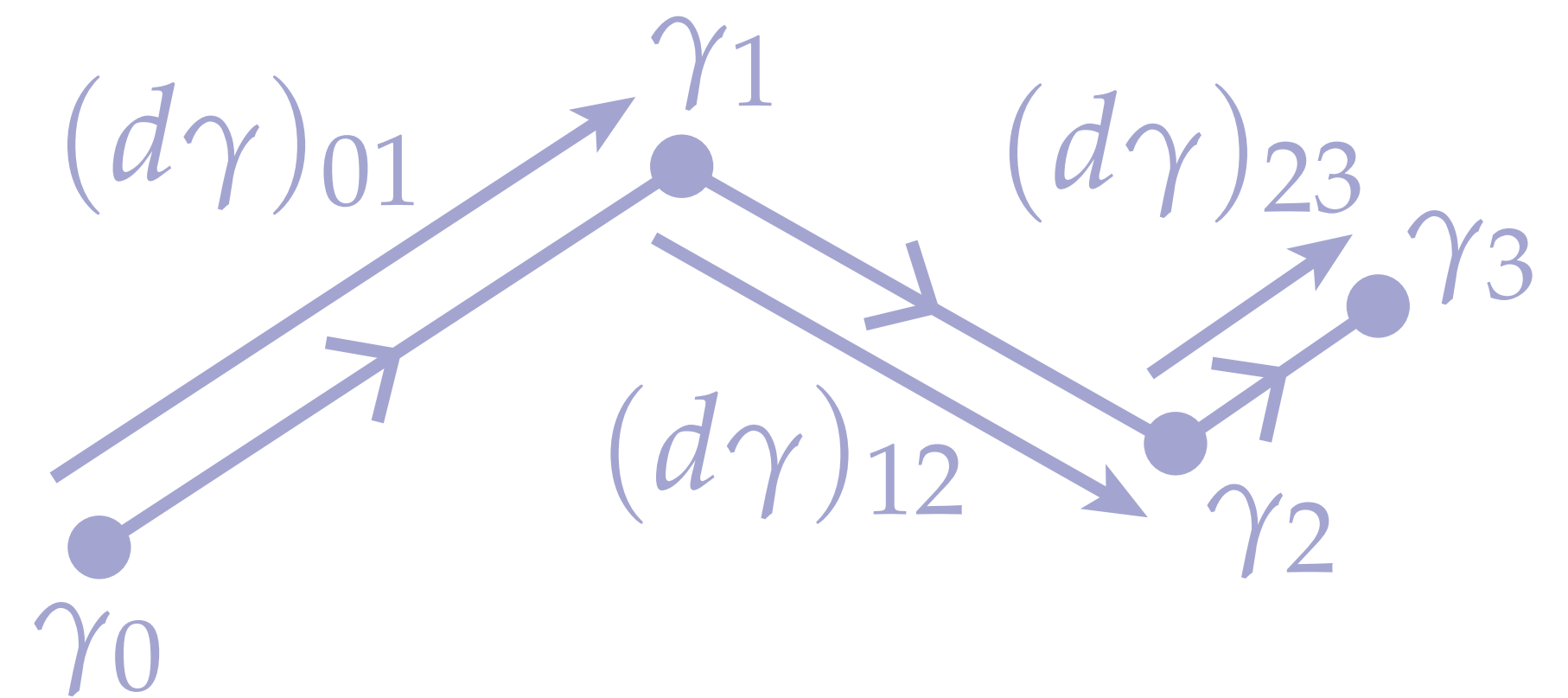
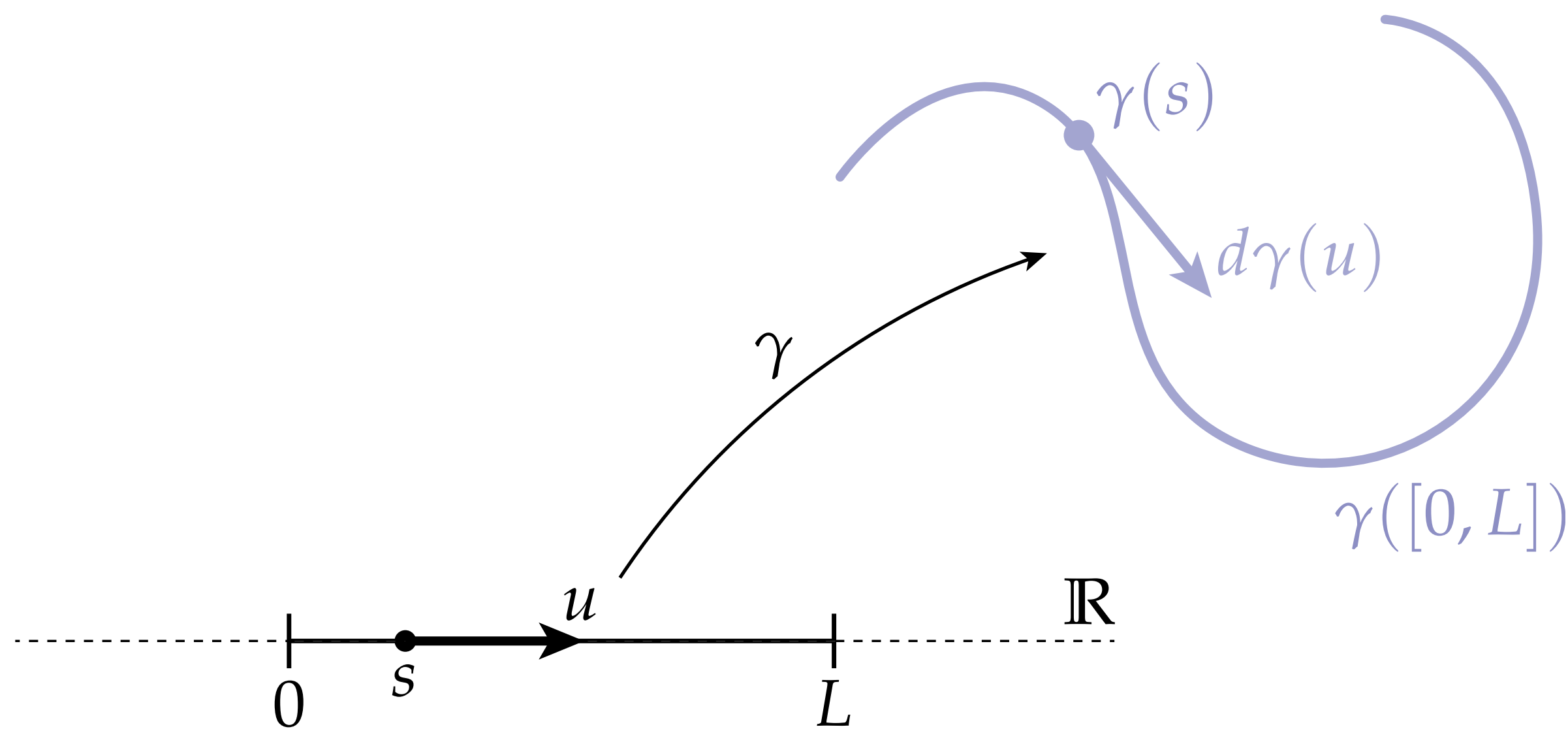


$$K = \{ (v_0, v_1), (v_1, v_2), (v_2, v_3), \\ (v_0), (v_1), (v_2), (v_3), \emptyset \}$$

$$\begin{aligned} \gamma(v_0) &= (33, 66) \\ \gamma(v_1) &= (79, 36) \\ \gamma(v_2) &= (118, 58) \\ \gamma(v_3) &= (134, 47) \end{aligned}$$

# Differential of a Discrete Curve

- We can now directly translate statements about **smooth** curves expressed via **smooth** exterior calculus into statements about **discrete** curves expressed using **discrete** exterior calculus
- Simple example: the *differential* just becomes the edge vectors:

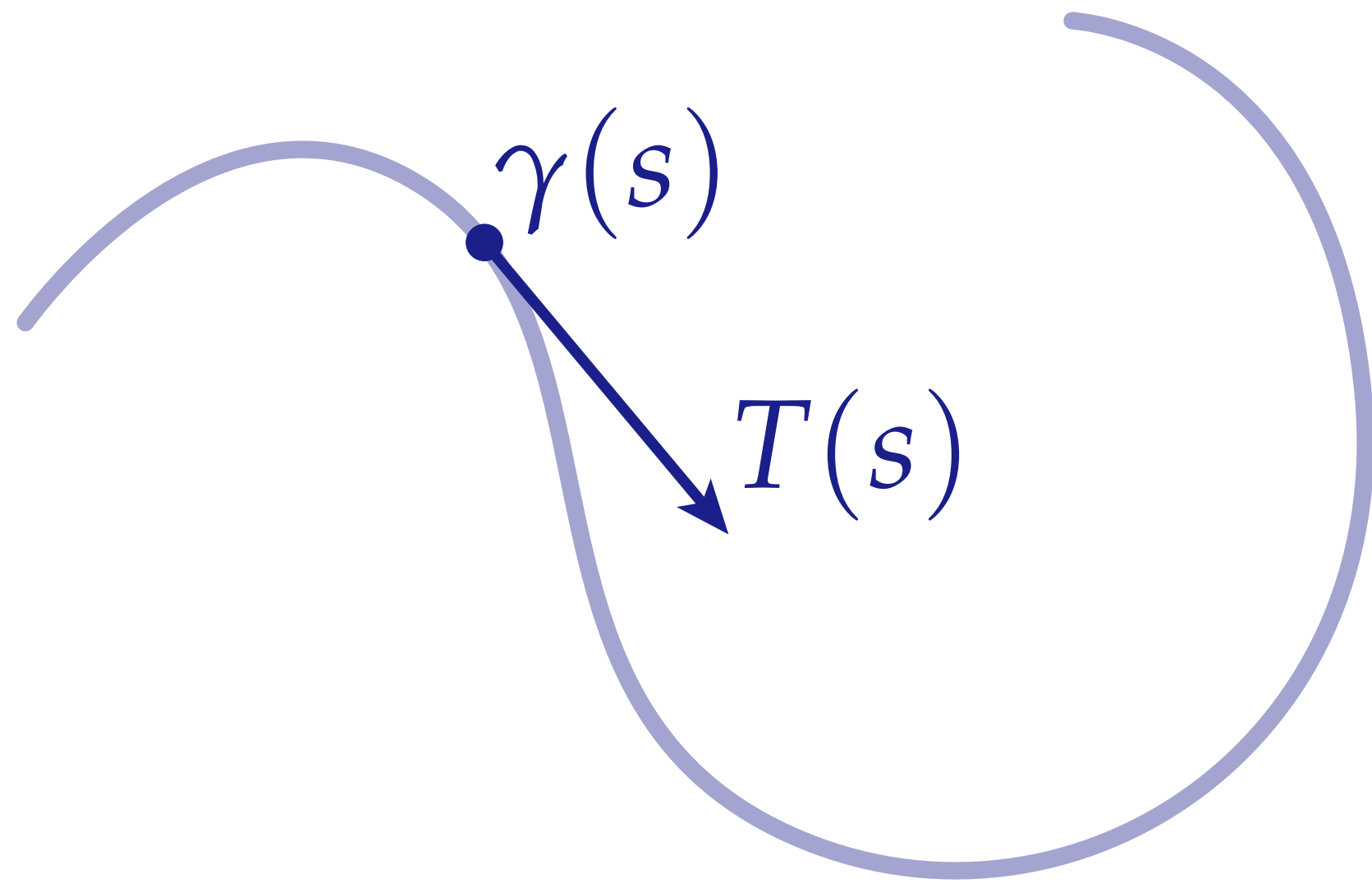


$$(d\gamma)_{ij} = \gamma_j - \gamma_i$$

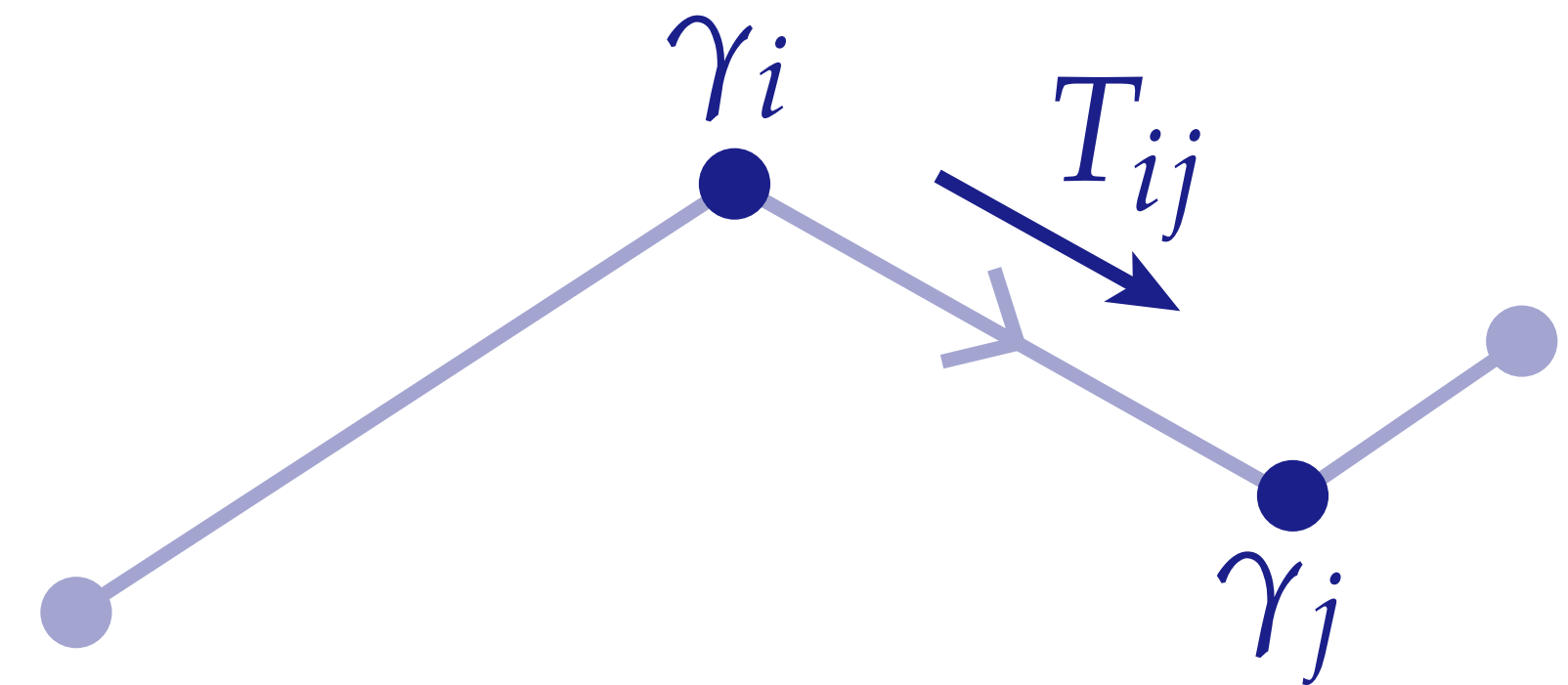


# Discrete Tangent

- As in smooth setting, can simply normalize differential to obtain tangents, yielding a vector per edge\*



$$T(s) := d\gamma\left(\frac{d}{ds}\right) / \left|d\gamma\left(\frac{d}{ds}\right)\right|$$

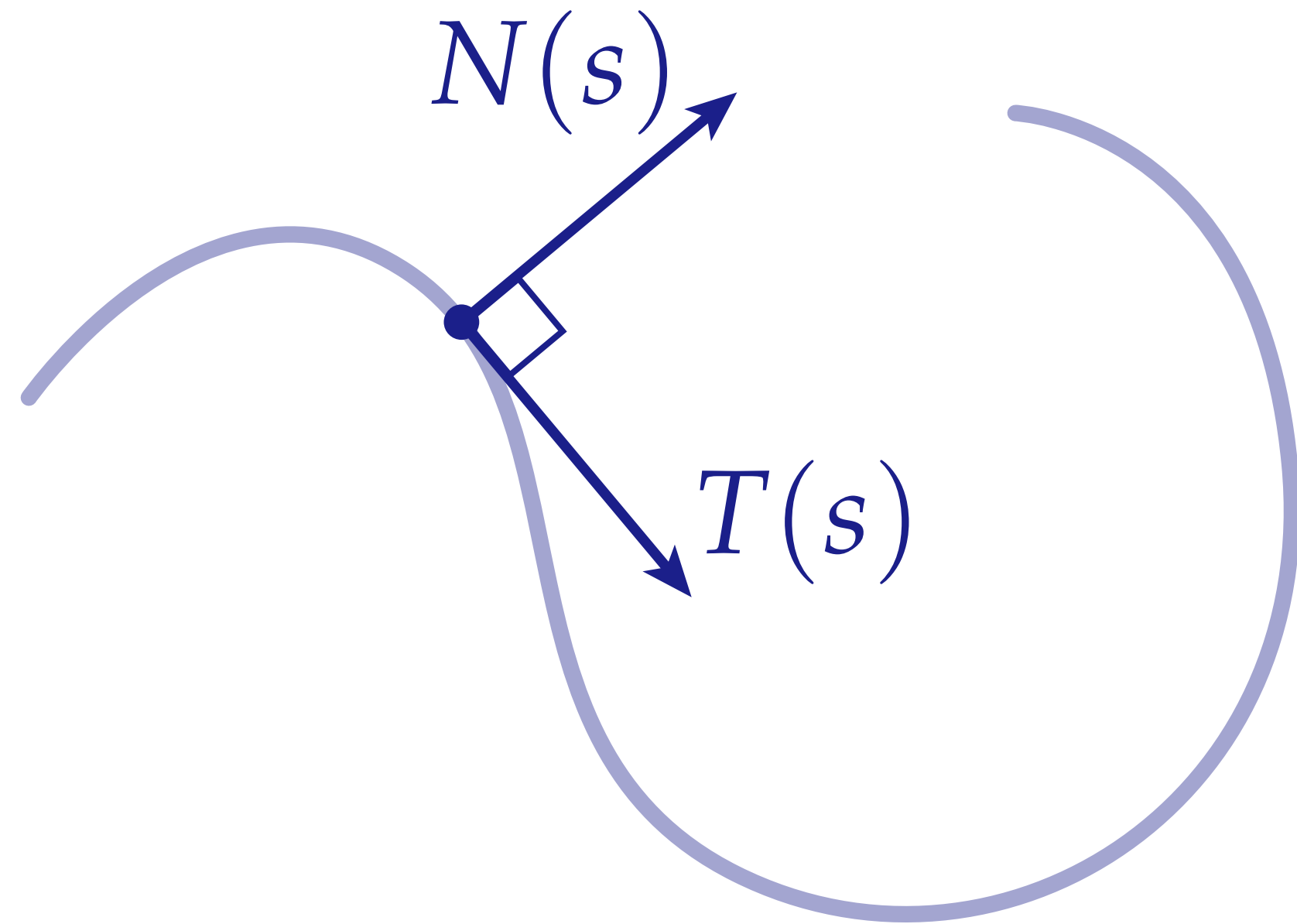


$$T_{ij} := (d\gamma)_{ij} / |(d\gamma)_{ij}|$$

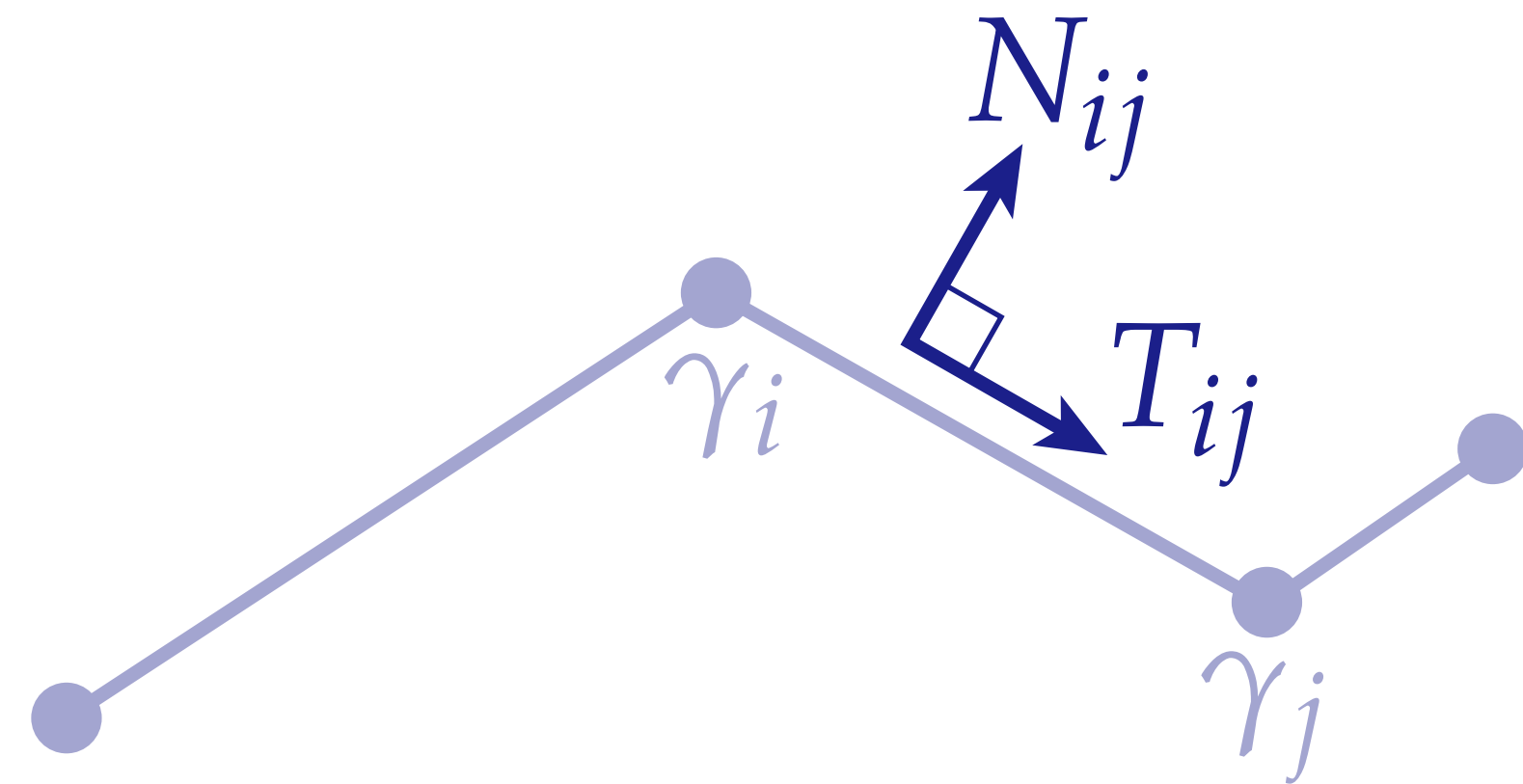
\*And no definition of the tangent at vertices!

# Discrete Normal

- As in the smooth setting, we can express the (discrete) normals of a planar curve as a 90-degree rotation of the (discrete) tangent:



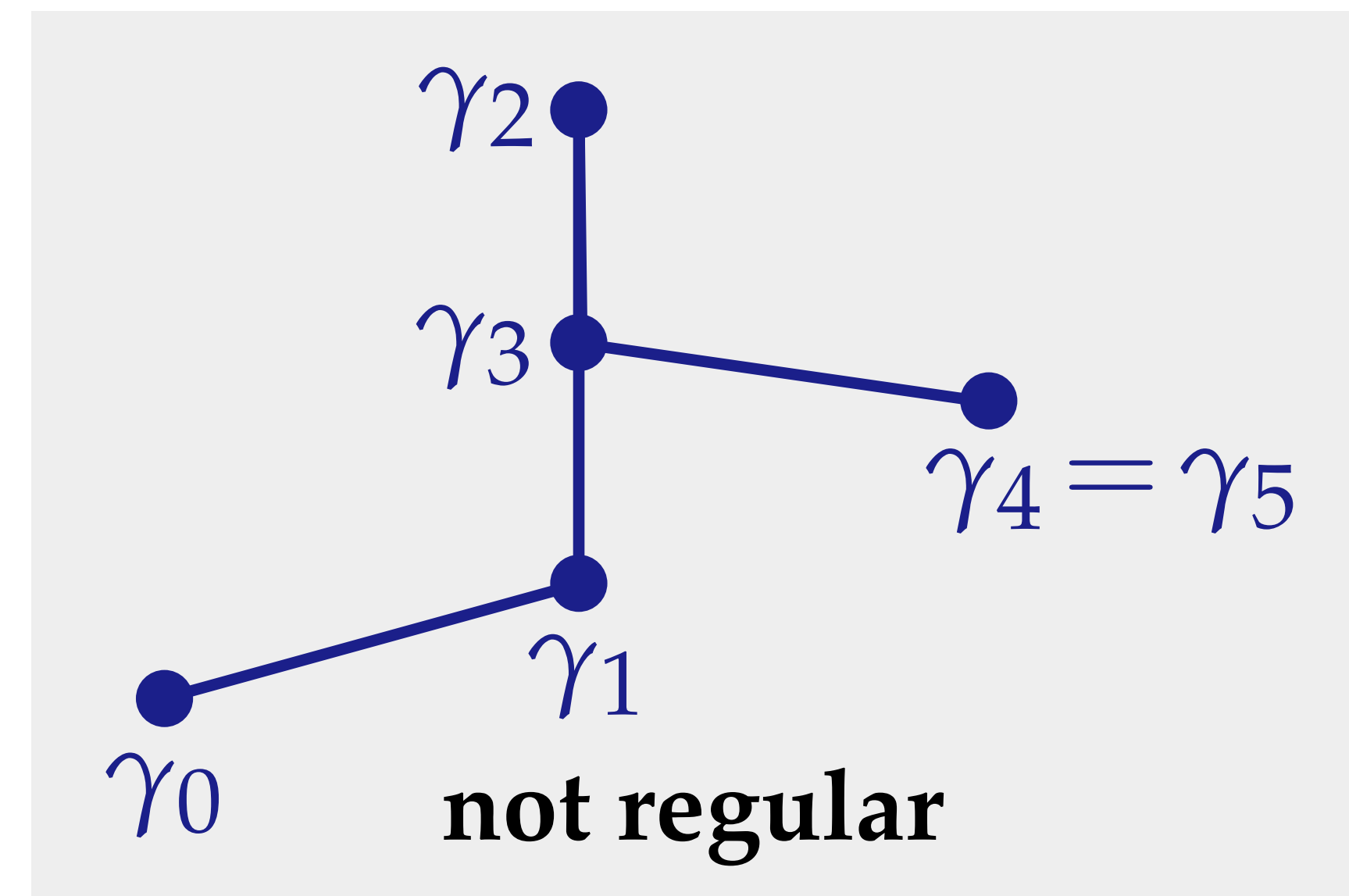
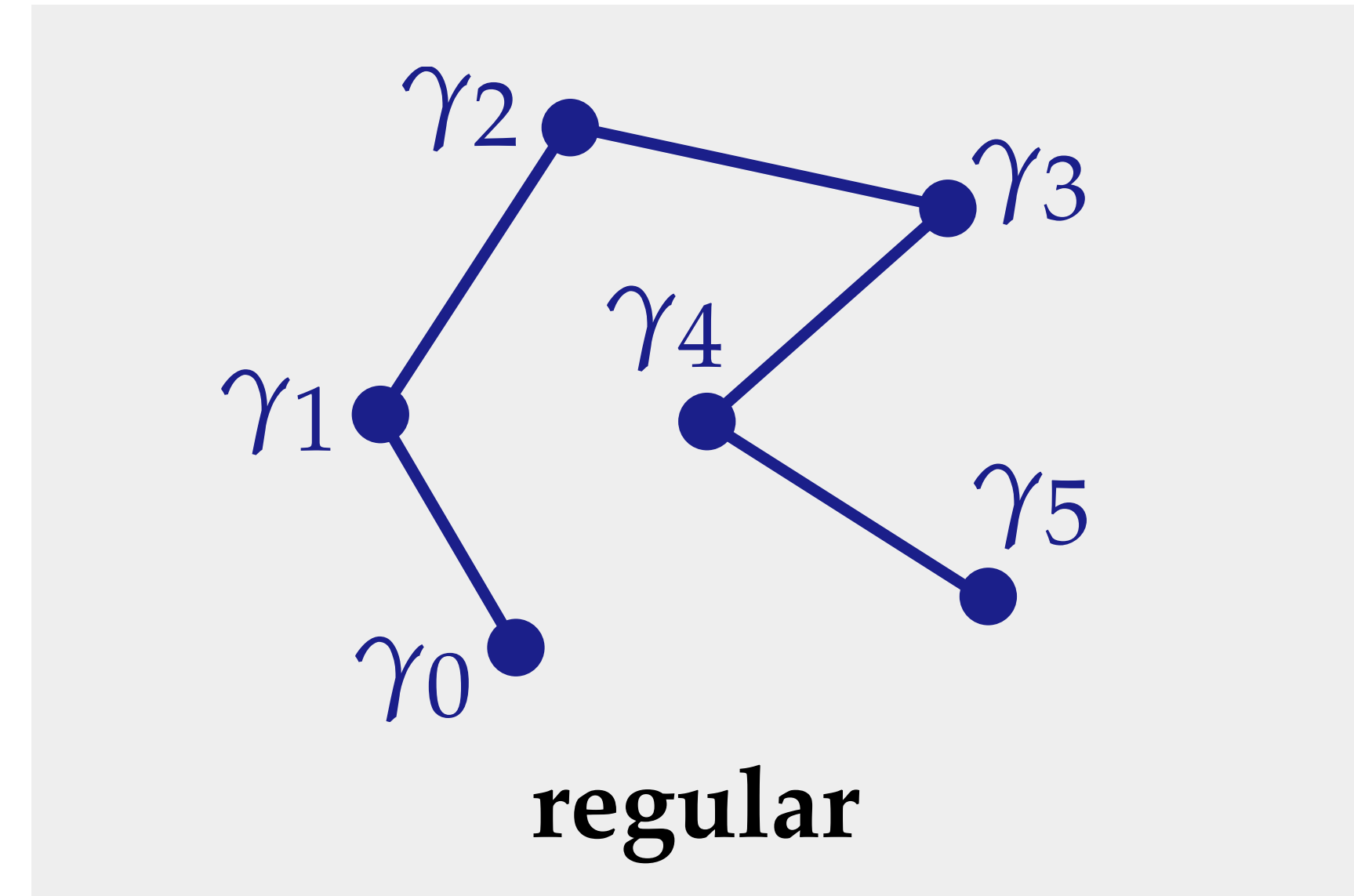
$$N(s) = \mathcal{J}T(s)$$



$$N_{ij} = \mathcal{J}T_{ij}$$

# Regular Discrete Curve / Discrete Immersion

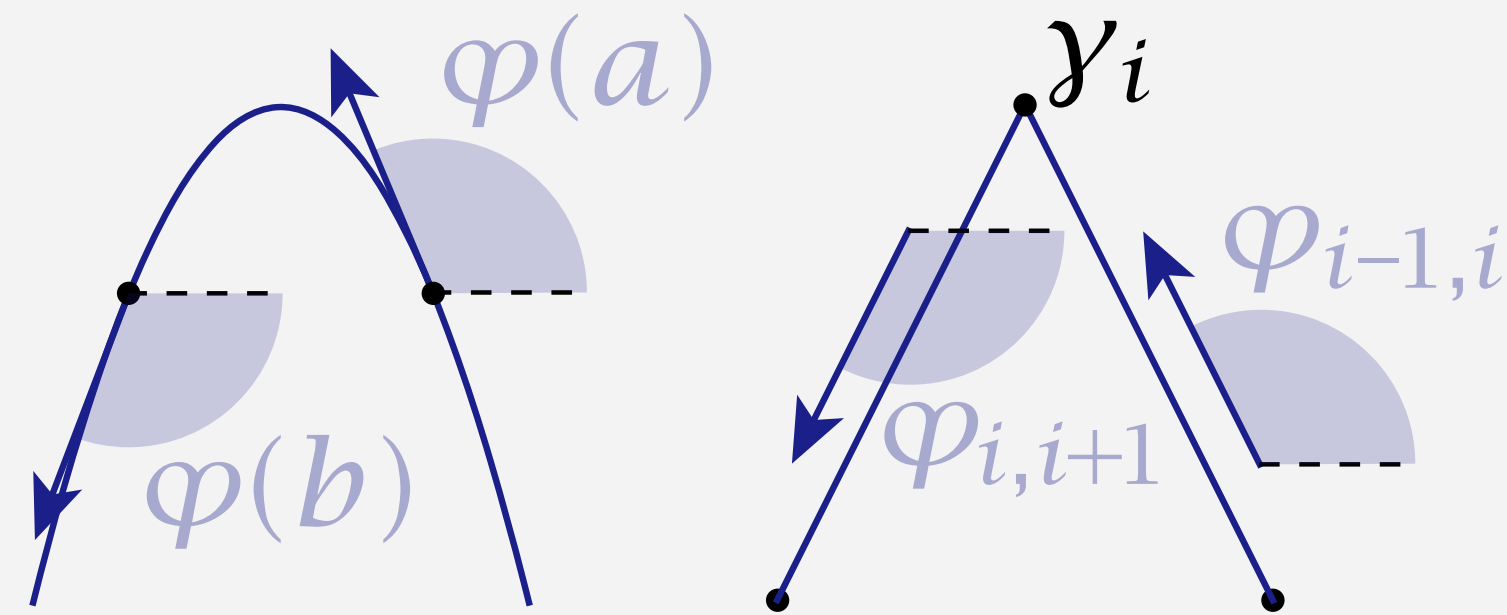
- Recall that a smooth curve is *regular* if its differential is nonzero; this condition helps avoid “bad behavior” like sharp cusps
- For a discrete curve, a nonzero differential merely prevents zero edge lengths; need something stronger to get “nice” curves
- In particular, a *regular discrete curve* or *discrete immersion* is a discrete curve that is a locally injective map
- Rules out zero edge lengths *and* zero angles



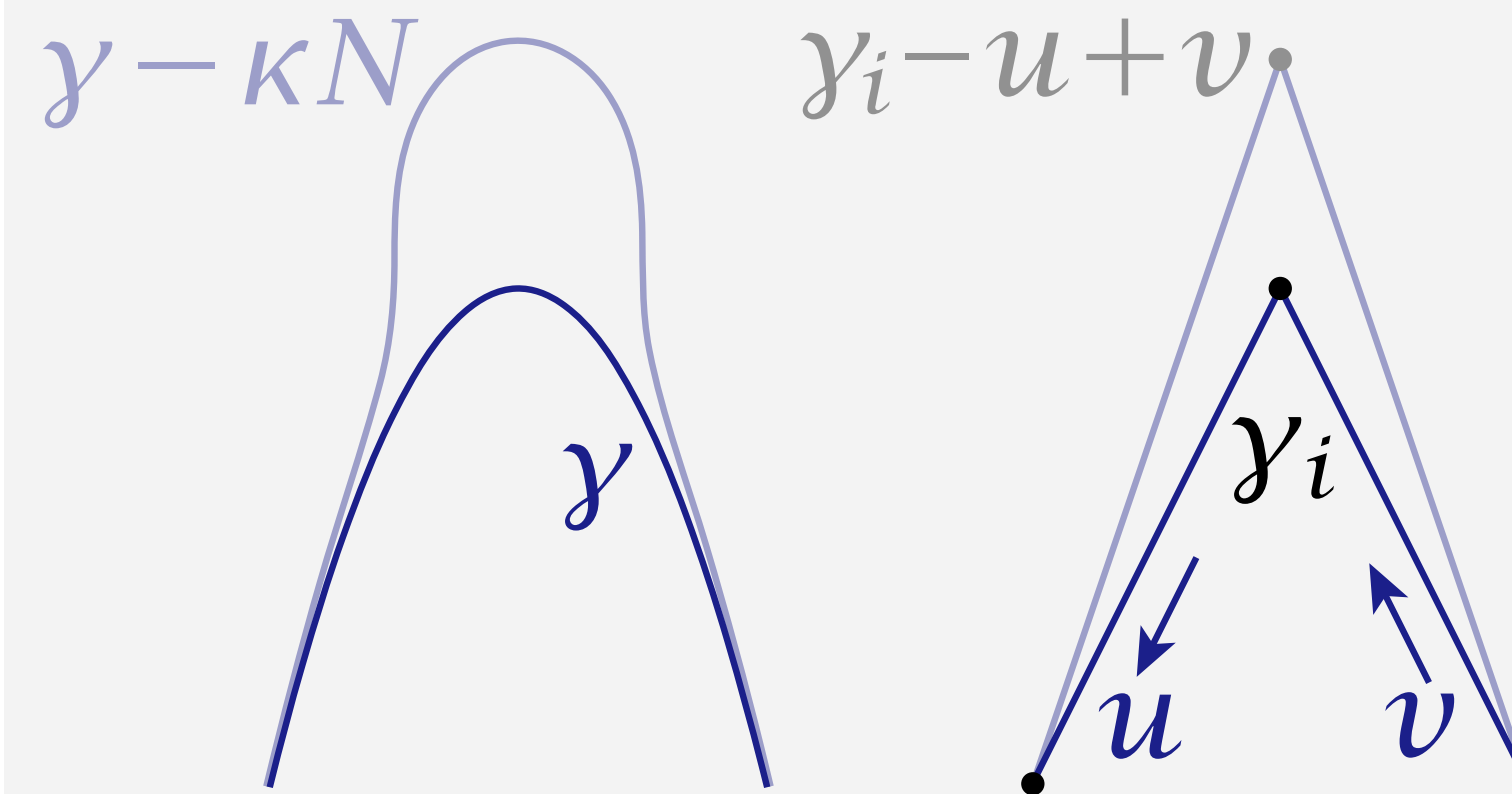
# Discrete Curvature

- For a regular discrete curve, discrete curvature has several definitions

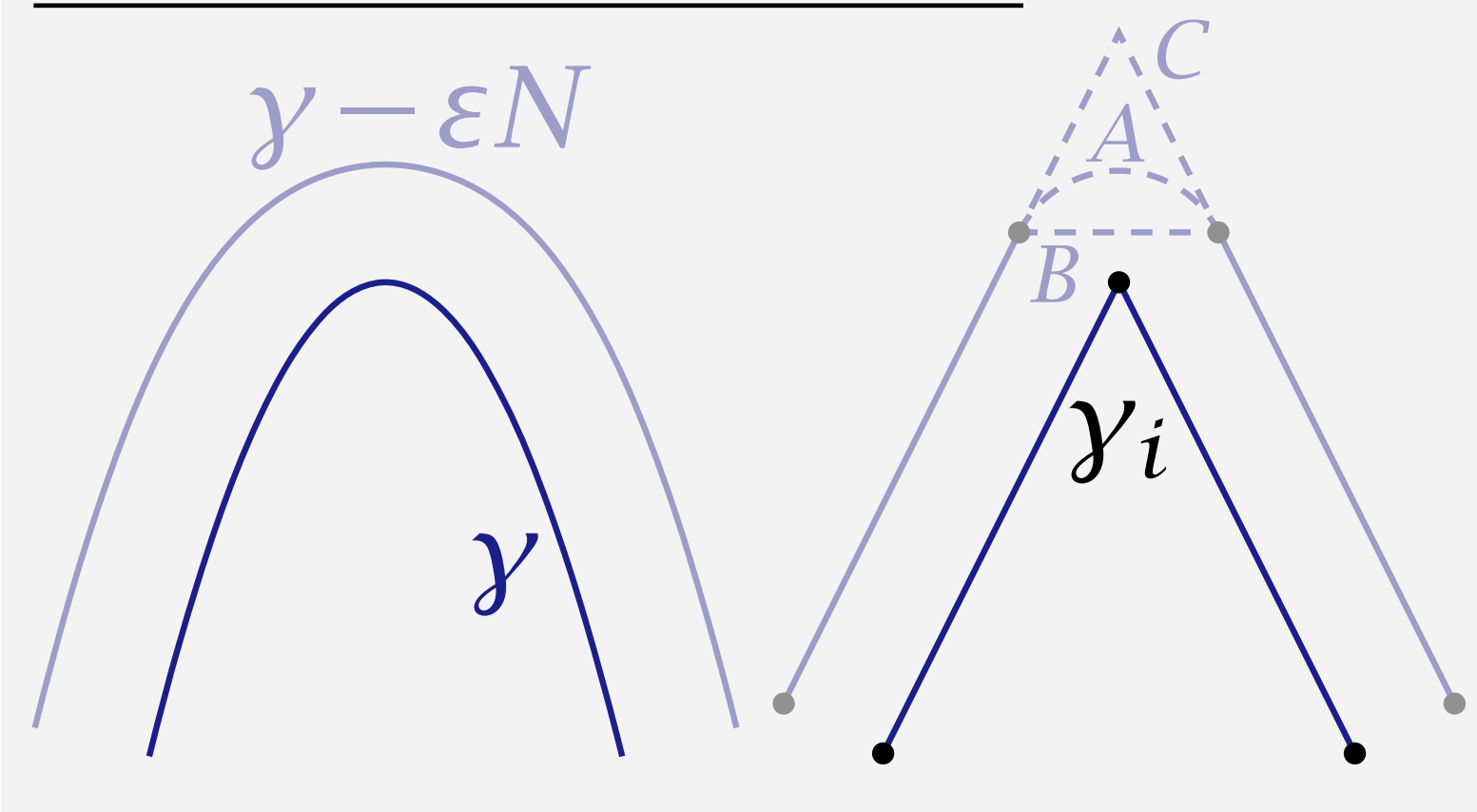
## TURNING ANGLE



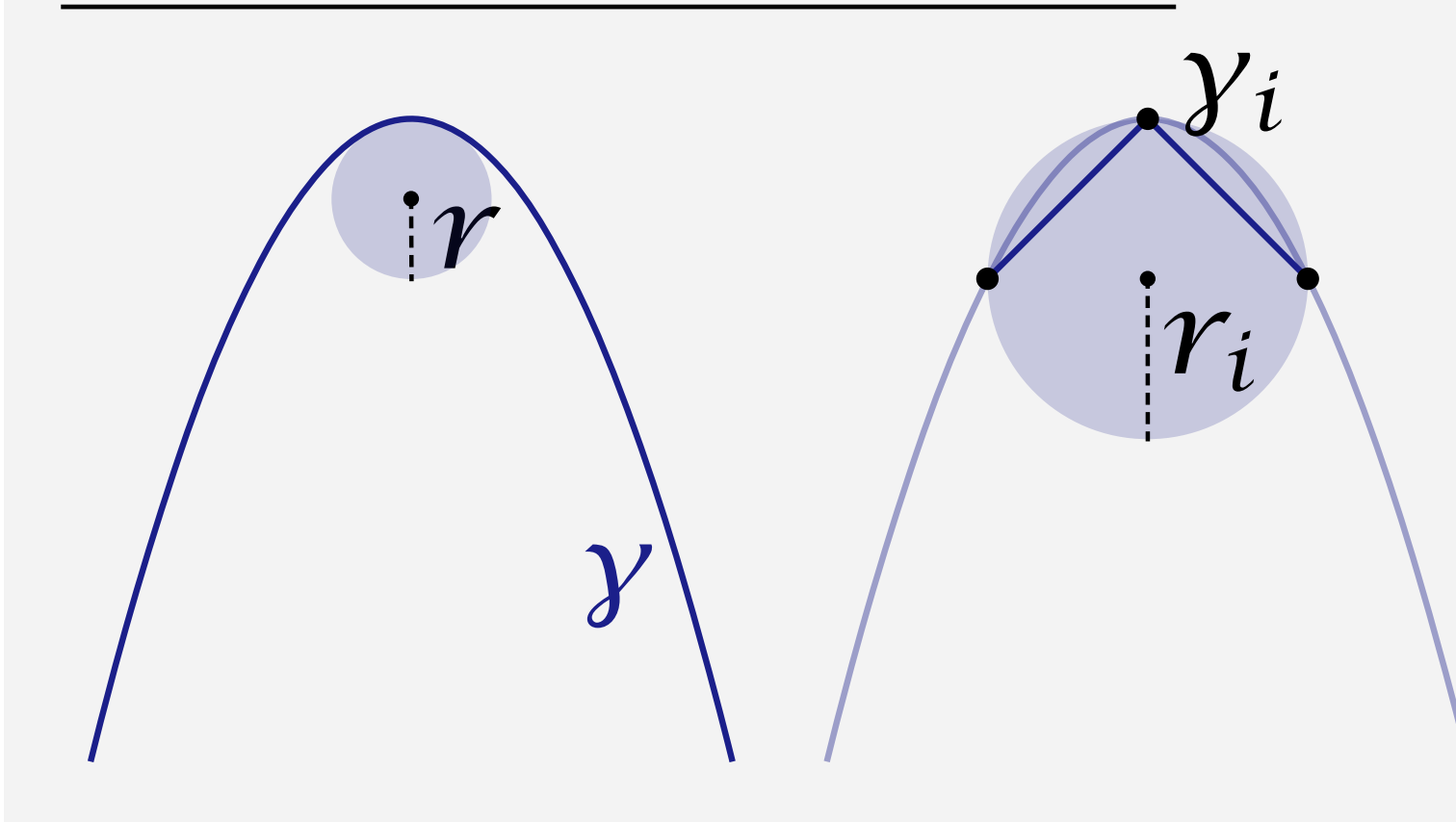
## LENGTH VARIATION



## STEINER FORMULA



## OSCULATING CIRCLE





# Fundamental Theorem of Discrete Plane Curves

**Fact.** Up to rigid motions, a regular discrete plane curve is uniquely determined by its edge lengths and turning angles.

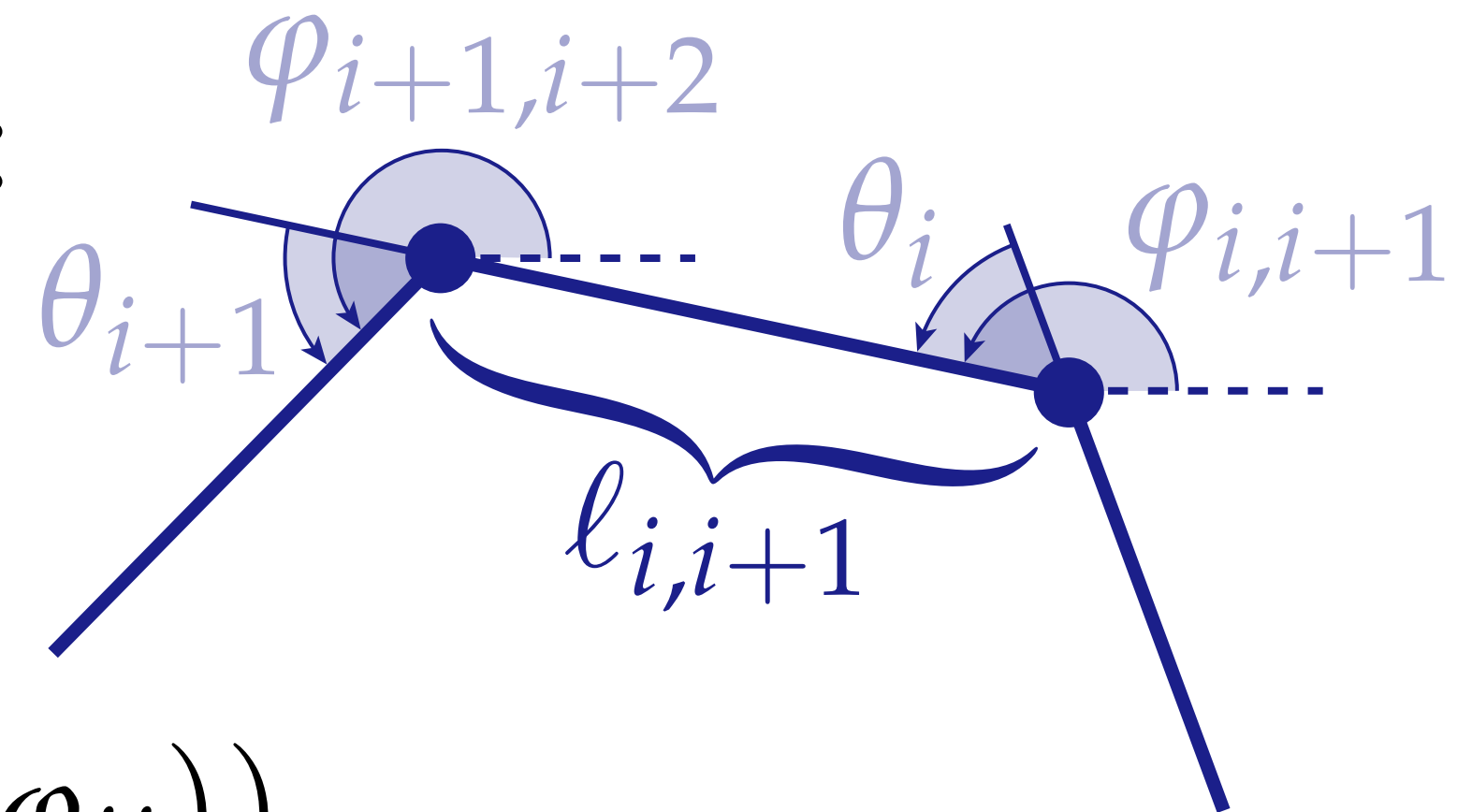
**Q:** Given only this data, how can we recover the curve?

**A:** Mimic the procedure from the smooth setting:

Sum curvatures to get angles:  $\varphi_{i,i+1} := \sum_{k=1}^i \theta_k$

Evaluate unit tangents:  $T_{ij} := (\cos(\varphi_{ij}), \sin(\varphi_{ij}))$

Sum tangents to get curve:  $\gamma_i := \sum_{k=1}^i \ell_{k,k+1} T_{k,k+1}$



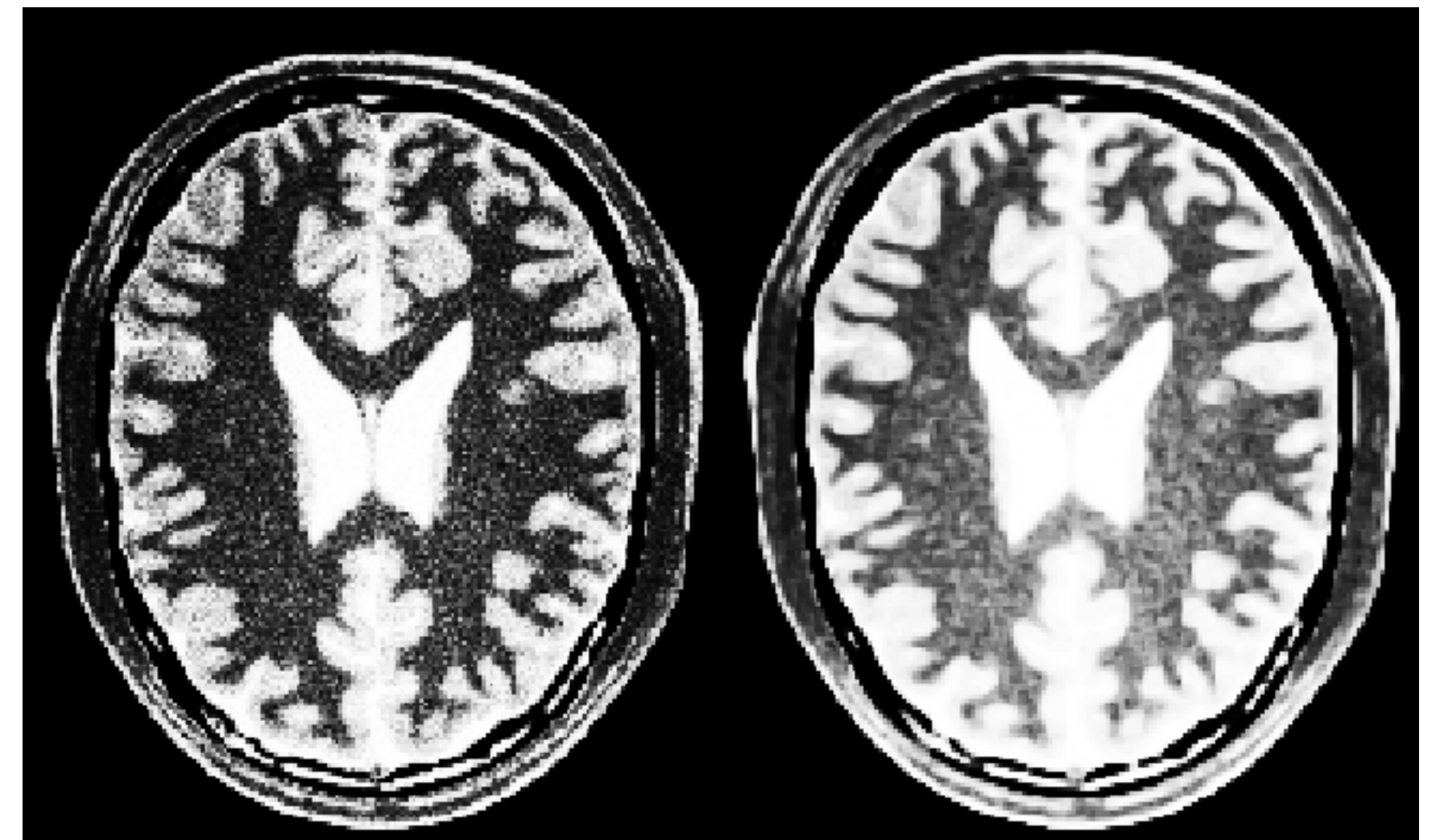
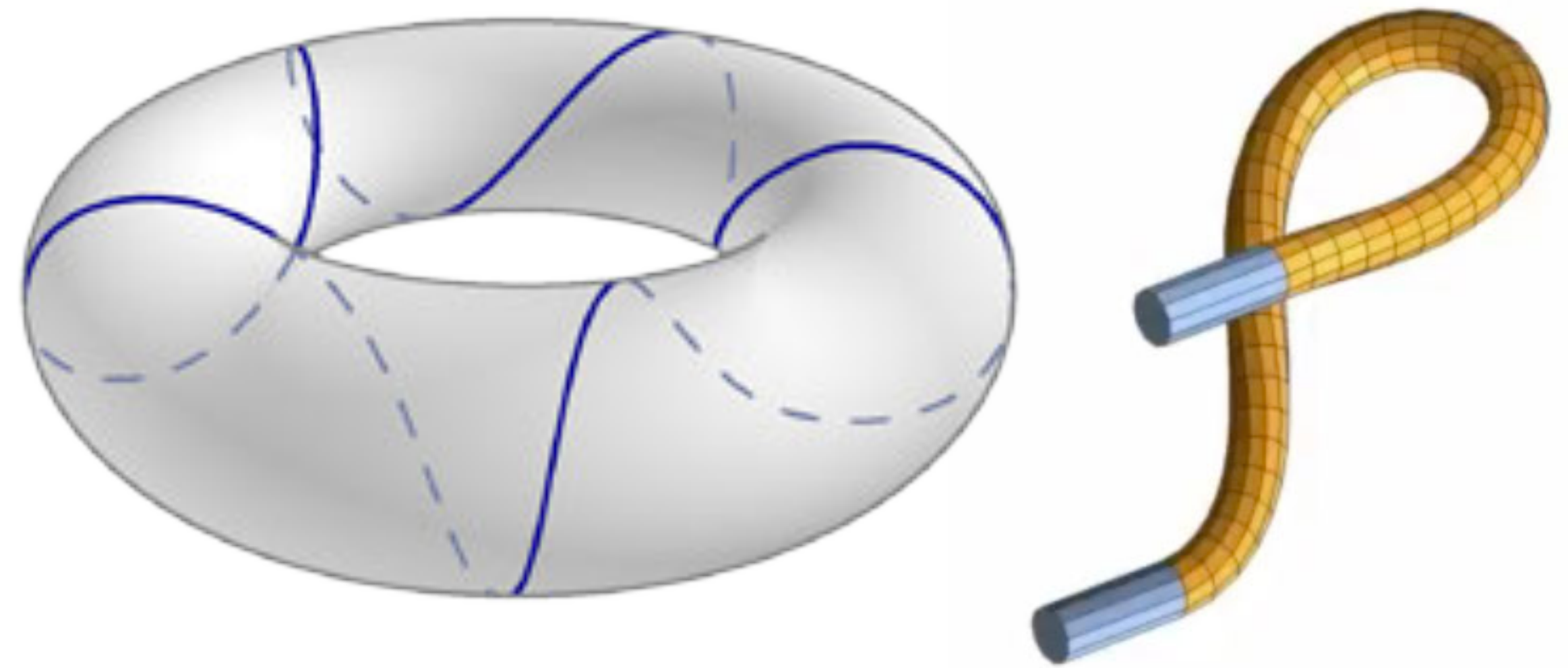
**Q:** Rigid motions?

The background features a series of overlapping, curved lines that create a grid-like pattern. These lines are in various shades of light blue and grey. A prominent white horizontal band runs across the center of the image, providing a high-contrast area for the text. The overall aesthetic is clean and modern, with a focus on geometric shapes and color gradients.

# *Curvature Flow*

# Curvature Flow on Curves

- A *curvature flow* is a time evolution of a curve (or surface) driven by some function of its curvature.
- Such flows model physical *elastic rods*, can be used to find shortest curves (*geodesics*) on surfaces, or might be used to smooth noisy data (e.g., image contours).
- Two common examples: *length-shortening flow* and *elastic flow*.





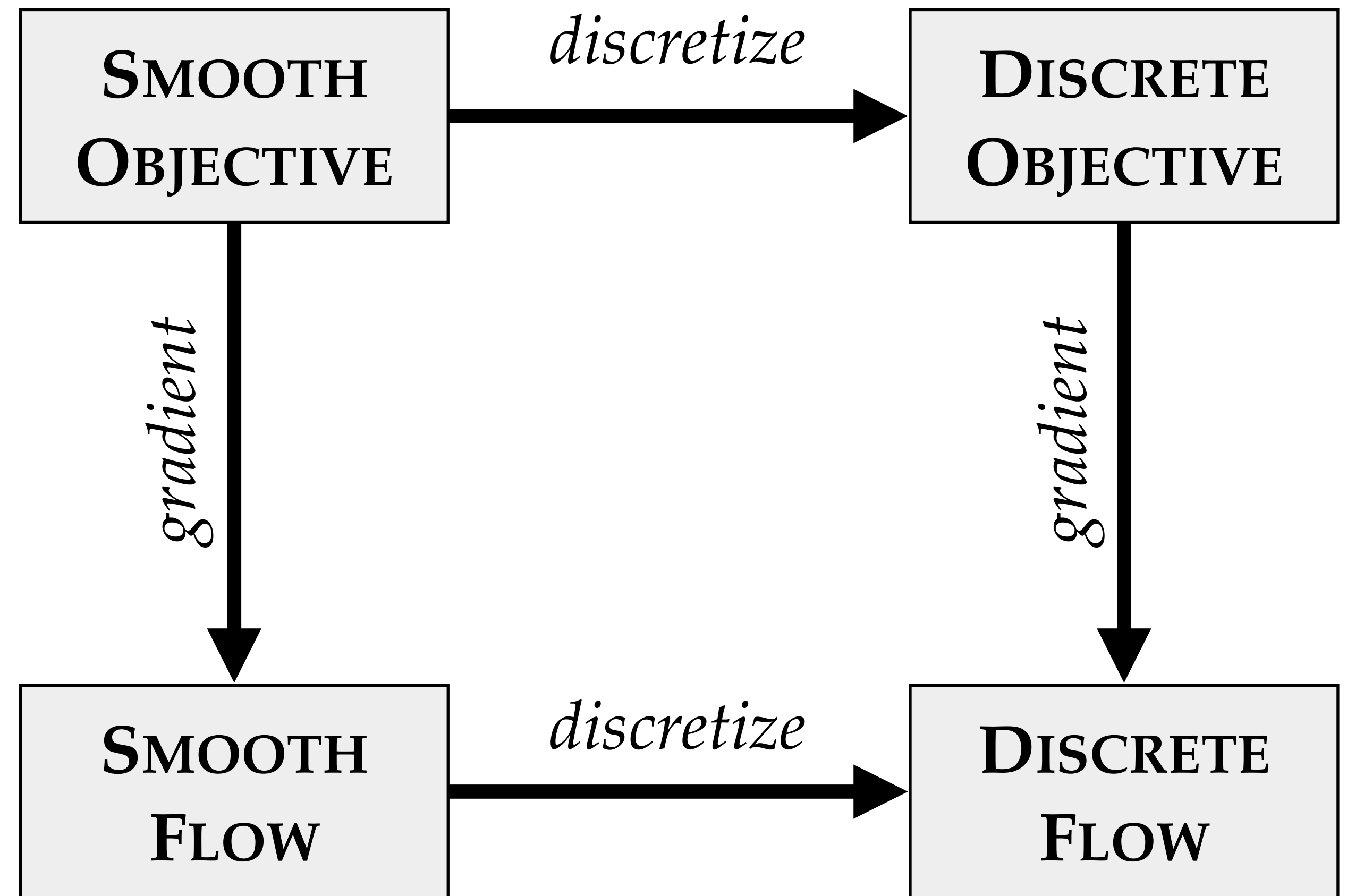
# Discretizing a Gradient Flow

- Two possible paths for discretizing any gradient flow:

1. **First** derive the gradient of the objective in the smooth setting, **then** discretize the resulting evolution equation.

2. **First** discretize the objective itself, **then** take the gradient of the resulting discrete objective.

- In general, *will not lead to the same numerical scheme/algorithm!*



(Does **NOT** commute in general.)

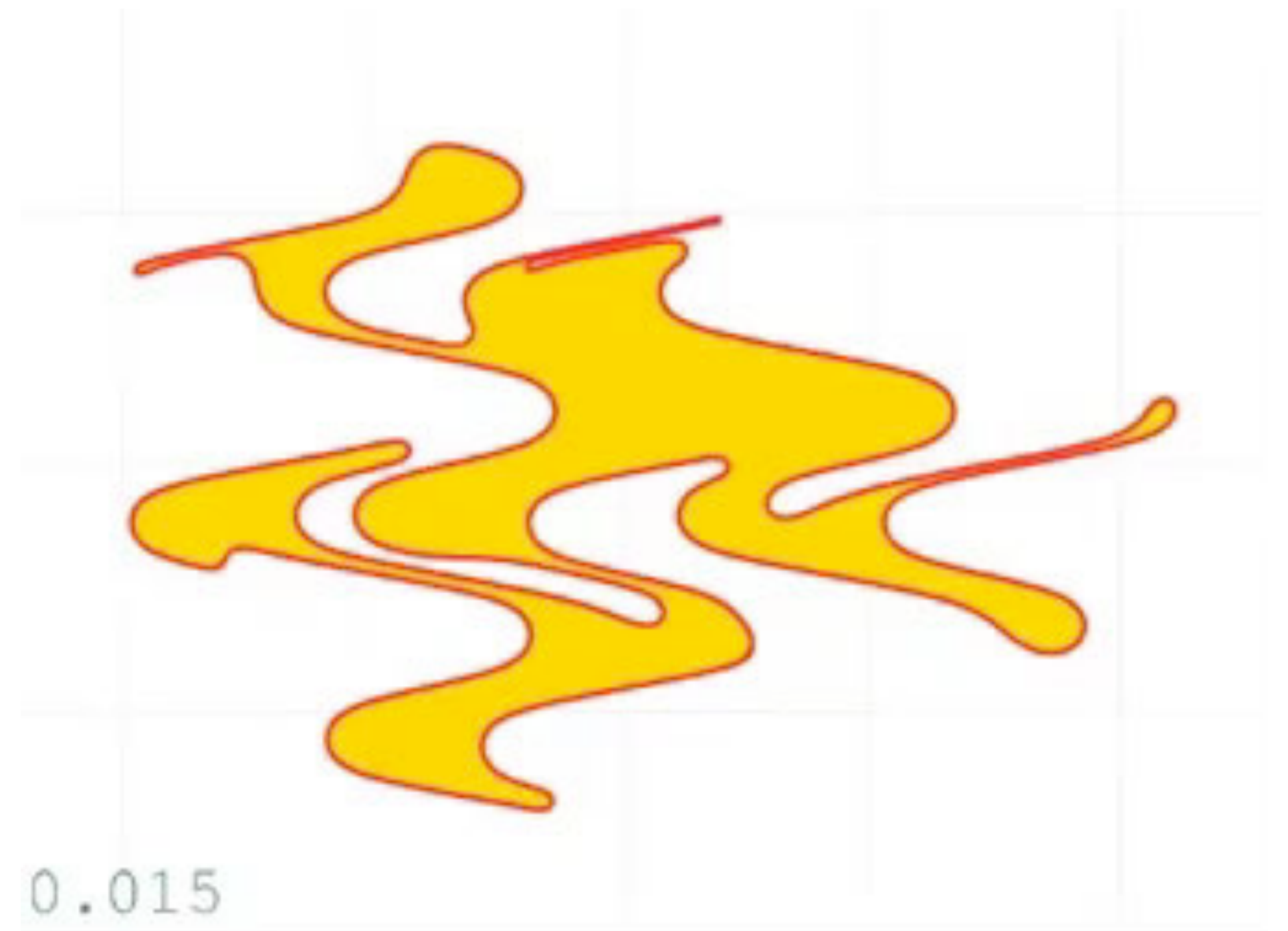


# Length Shortening Flow

- The objective for length shortening flow is simply the total length of the curve; the flow is then the  $(L^2)$  gradient flow.
- For closed curves, several interesting features (Gage-Grayson-Hamilton):
  - Center of mass is preserved
  - Curves flow to “round points”
  - Embedded curves remain embedded

$$\text{length}(\gamma) := \int_0^L \left| \frac{d}{ds} \gamma \right| ds$$

$$\frac{d}{dt} \gamma = -\nabla_{\gamma} \text{length}(\gamma)$$

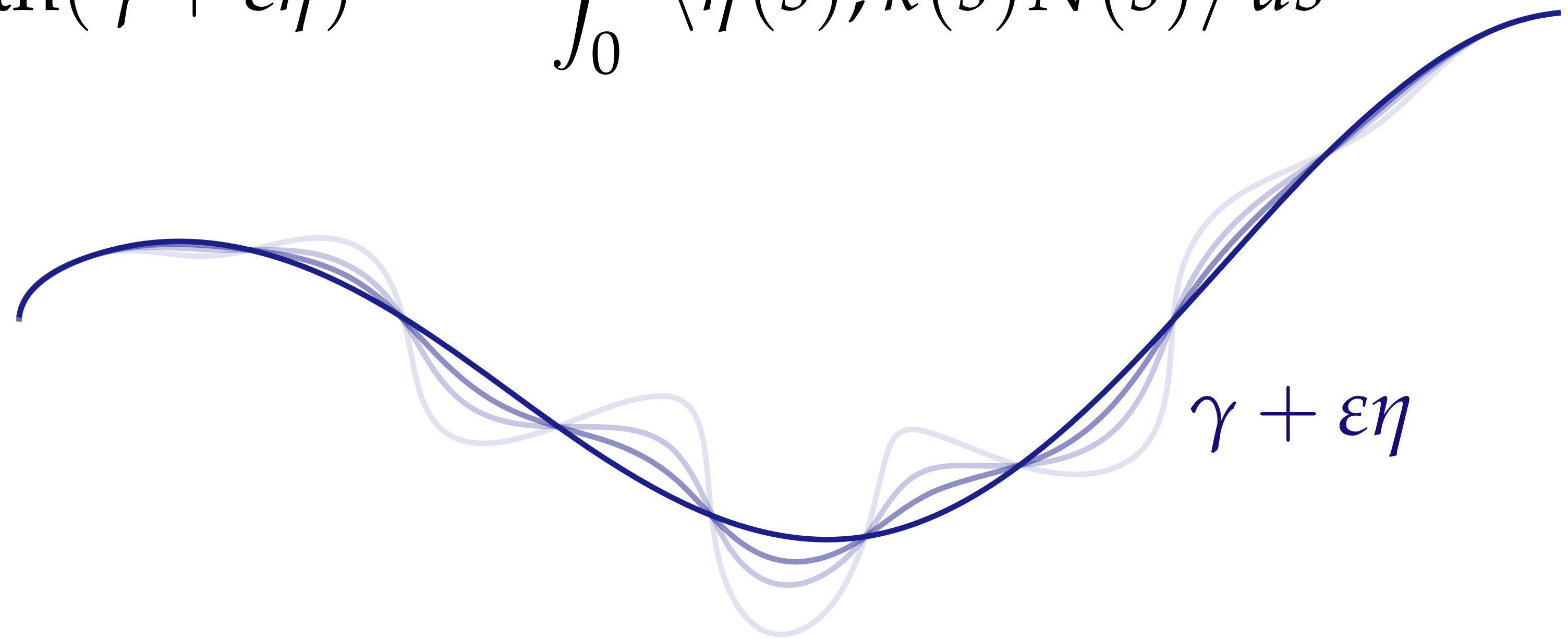


*credit: Sigurd Angenent*

# Length Shortening Flow

Let  $\text{length}(\gamma)$  denote the total length of a regular plane curve  $\gamma : [0, L] \rightarrow \mathbb{R}^2$ , and consider a variation  $\eta : [0, L] \rightarrow \mathbb{R}^2$  vanishing at endpoints. One can then show that

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



**Key idea:** quickest way to reduce length is to move in the direction  $\kappa N$ .

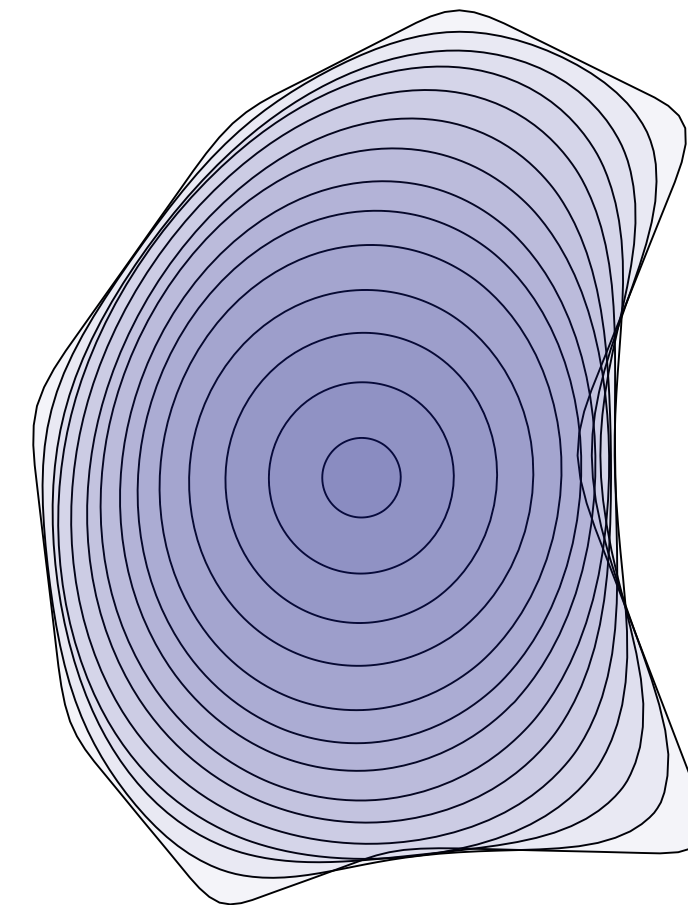
# Length Shortening Flow—Forward Euler

- At each moment in time, move curve in normal direction with speed proportional to curvature
- “Smooths out” curve (e.g., noise), eventually becoming circular
- Discretize by replacing time derivative with difference in time; smooth curvature with one (of many) curvatures
- Repeatedly add a little bit of  $\kappa N$  (“forward Euler method”)

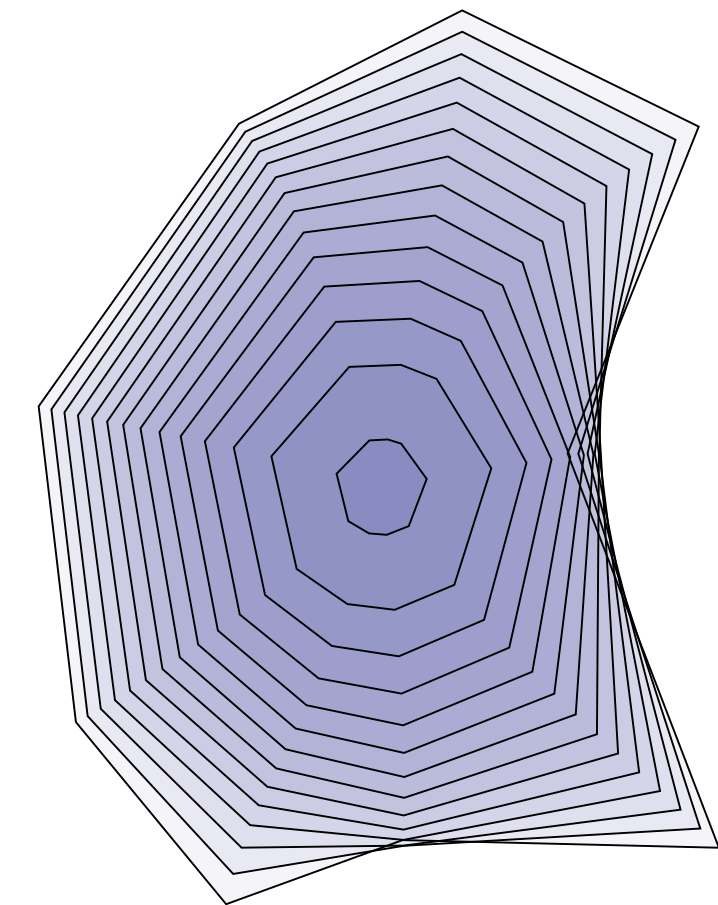
$$\frac{d}{dt} \gamma(s, t) = -\kappa(s, t) N(s, t)$$

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

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



**smooth**



**discrete**

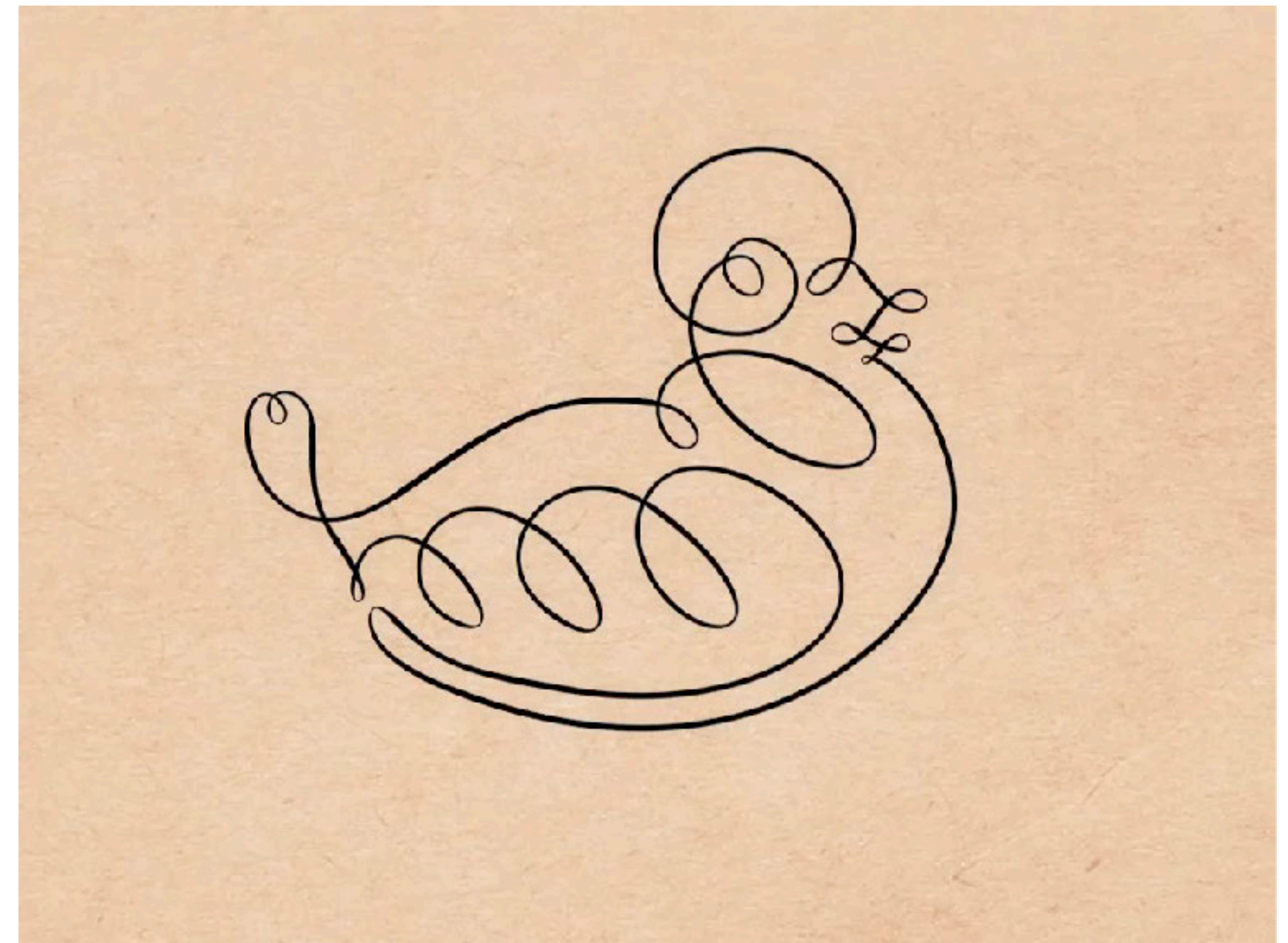


# Elastic Flow

- Basic idea: rather than shrinking length, try to reduce *bending* (curvature)
- Objective is integral of squared curvature; elastic flow is then gradient flow on this objective
- Minimizers are called *elastic curves*
- More interesting w/ constraints (e.g., endpoint positions & a tangents)

$$E(\gamma) := \int_0^L \kappa(s)^2 ds$$

$$\frac{d}{dt}\gamma = -\nabla_{\gamma} E(\gamma)$$



# *Isometric Elastic Flow*

- Different way to smooth out a curve is to directly “shrink” curvature
- Discrete case: “scale down” turning angles, then use the fundamental theorem of discrete plane curves to reconstruct
- Extremely stable numerically; exactly preserves edge lengths
- Challenge: how do we make sure closed curves remain closed?

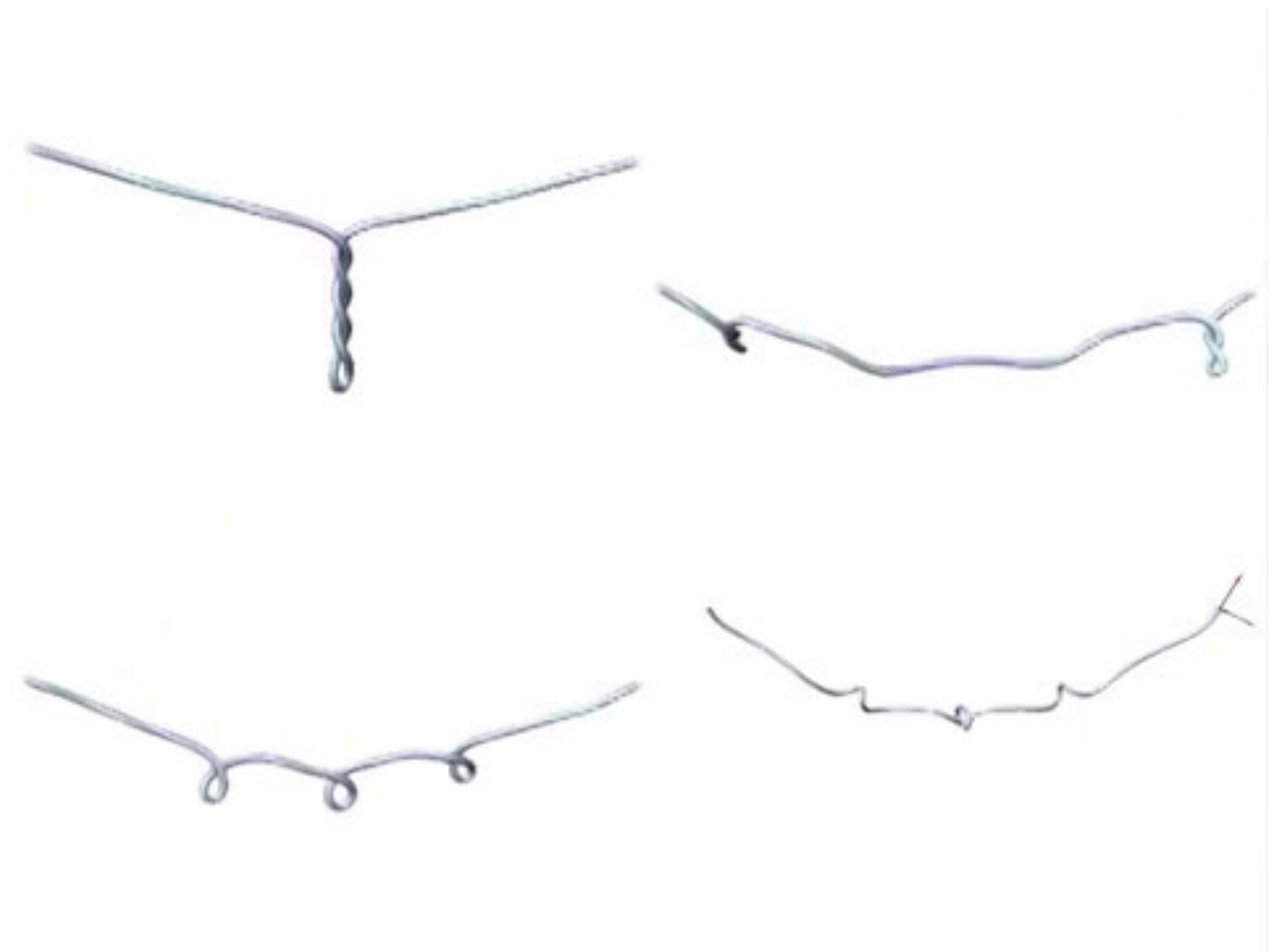


From Crane et al, “*Robust Fairing via Conformal Curvature Flow*”



# Elastic Rods

- For space curve, can also try to minimize both *curvature* **and** *torsion*
- Both in some sense measure “non-straightness” of curve
- Provides rich model of *elastic rods*
- Lots of interesting applications (simulating hair, laying cable, ...)



From Bergou et al, “Discrete Elastic Rods”



# Reading Assignment

- Readings from papers on curve algorithms (will be posted online)

Robust Fairing via Conformal Curvature Flow

Keenan Crane  
Caltech

Ulrich Pinkall  
TU Berlin

Peter Schröder  
Caltech

Abstract

We present a formulation of Willmore flow for triangulated surfaces that permits extraordinarily large time steps and naturally preserves the quality of the input mesh. The main insight is that Willmore flow becomes remarkably stable when expressed in curvature space – we develop the precise conditions under which curvature is allowed to evolve. The practical outcome is a highly efficient algorithm that naturally preserves texture and does not require remeshing during the flow. We apply this algorithm to surface fairing, geometric modeling, and construction of constant mean curvature (CMC) surfaces. We also present a new algorithm for length-preserving flow on planar curves, which provides a valuable analogy for the surface case.

CR Categories: 1.3.5 [Computer Graphics]: Computational Geometry and Object Modeling—Geometric algorithms, languages, and systems

Keywords: digital geometry processing, discrete differential geometry, geometric modeling, surface fairing, shape spaces, conformal geometry, quasiconformal, symplectic geometry

Links: [DL](#) [PDF](#)

1 Introduction

At the most basic level, a curvature flow produces successively smoother approximations of a given piece of geometry (e.g., a curve or surface), by reducing a fairing energy. Such flows have far-ranging applications in fair surface design, inpainting, denoising, and biological modeling [Helfrich 1973; Carham 1970]; they are also the central object in mathematical problems such as the Willmore conjecture [Pinkall and Sterling 1987].

Numerical methods for curvature flow suffer from two principal difficulties: (I) a severe time step restriction, which often yields unacceptably slow evolution and (II) degeneration of mesh elements, which necessitates frequent remeshing or other corrective devices. We circumvent these issues by (I) using a curvature-based representation of geometry, and (II) working with conformal transformations, which naturally preserve the aspect ratio of triangles. The resulting algorithm stably integrates time steps orders of magnitude larger than existing methods (Figure 1), resulting in substantially faster real-world performance (Section 6.4.2).

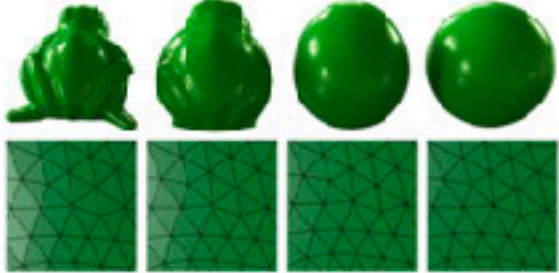


Figure 1: A decailed frog flows to a round sphere in only three large, explicit time steps (top). Meanwhile, the quality of the triangulation (bottom) is almost perfectly preserved.

2 Preliminaries

We adopt two essential conventions from Crane et al. [2011]. First, we interpret any surface in  $\mathbb{R}^3$  (e.g., a triangle mesh) as the image of a conformal immersion (Section 2.2.1). Second, we interpret three-dimensional vectors as imaginary quaternions (Section 2.3). Proofs in the appendix make use of quaternion-valued differential forms; interested readers may benefit from the material in [Rambhov et al. 2002; Crane 2013].

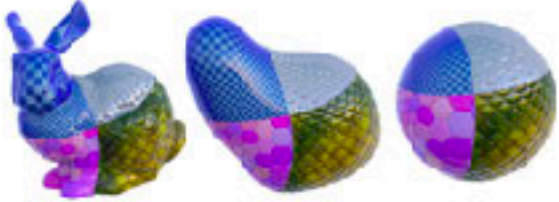


Figure 2: Our flow gracefully preserves the appearance of texture throughout all stages of the flow.

Discrete Elastic Rods

Miklós Bergou  
Columbia University

Max Wardetzky  
Freie Universität Berlin

Stephen Robinson  
Columbia University

Basile Audoly  
CNRS / UPMC Univ Paris 06

Eitan Grinspan  
Columbia University

Figure 1: Experiment and simulation: A simple (trefoil) knot tied on an elastic rope can be turned into a number of fascinating shapes when twisted. Starting with a twist-free knot (left), we observe both continuous and discontinuous changes in the shape, for both directions of twist. Using our model of Discrete Elastic Rods, we are able to reproduce experiments with high accuracy.

Abstract

We present a discrete treatment of adjoined framed curves, parallel transport, and holonomy, thus establishing the language for a discrete geometric model of thin flexible rods with arbitrary cross section and undeformed configuration. Our approach differs from existing simulation techniques in the graphics and mechanics literature both in the kinematic description—we represent the material frame by its angular deviation from the natural Bishop frame—as well as in the dynamical treatment—we treat the centerline as dynamic and the material frame as quasistatic. Additionally, we describe a manifold projection method for coupling rods to rigid bodies and simultaneously enforcing rod inextensibility. The use of quasistatics and constraints provides an efficient treatment for stiff twisting and stretching modes; at the same time, we retain the dynamic bending of the centerline and accurately reproduce the coupling between bending and twisting modes. We validate the discrete rod model via quantitative buckling, stability and coupled-mode experiments, and via qualitative knot-tying comparisons.

CR Categories: 1.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism—Animation

Keywords: rods, strands, discrete holonomy, discrete differential geometry

1 Introduction

Recent activity in the field of discrete differential geometry (DDG) has fueled the development of simple, robust, and efficient tools for geometry processing and physical simulation. The DDG approach to simulation begins with the laying out of a physical model that is discrete from the ground up; the primary directive in designing this model is a focus on the preservation of key geometric structures that characterize the actual (smooth) physical system [Grinspan 2006].

Notably lacking is the application of DDG to physical modeling of elastic rods—curve-like elastic bodies that have one dimension (“length”) much larger than the others (“cross-section”). Rods have many interesting potential applications in animating knots, sutures, plants, and even kinematic skeletons. They are ideal for modeling deformations characterized by stretching, bending, and twisting. Stretching and bending are captured by the deformation of a curve called the centerline, while twisting is captured by the rotation of a material frame associated to each point on the centerline.

1.1 Goals and contributions

Our goal is to develop a principled model that is (a) simple to implement and efficient to execute and (b) easy to validate and test for convergence, in the sense that solutions to static problems and trajectories of dynamic problems in the discrete setup approach the solutions of the corresponding smooth problems. In pursuing this goal, this paper advances our understanding of discrete differential geometry, physical modeling, and physical simulation.

Elegant model of elastic rods

We build on a representation of elastic rods introduced for purposes of analysis by Langer and Singer [1996], arriving at a reduced coordinate formulation with a minimal number of degrees of freedom for extensible rods that represents the centerline of the rod explicitly and represents the material frame using only a scalar variable (§4.2). Like other reduced coordinate models, this avoids the need for stiff constraints that couple the material frame to the centerline, yet unlike other (e.g., curvature-based) reduced coordinate models, the explicit centerline representation facilitates collision handling and rendering.

Efficient quasistatic treatment of material frame

We additionally emphasize that the speed of sound in elastic rods is much faster for twisting waves than for bending waves. While this has long been established to the best of our knowledge it has not been used to simulate general elastic rods. Since in most applications the slower waves are of interest, we treat the material frame quasistatically (§5). When we combine this assumption with our reduced coordinate representation, the resulting equations of motion (§7) become very straightforward to implement and efficient to execute.

Geometry of discrete framed curves and their connections

Because our derivation is based on the concepts of DDG, our discrete model retains very distinctly the geometric structure of the smooth setting—in particular, that of parallel transport and the forces induced by the variation of holonomy (§6). We introduce

Discrete Viscous Threads

Miklós Bergou  
Columbia University

Basile Audoly  
UPMC Univ. Paris 06 & CNRS

Etienne Vouga  
Columbia University

Max Wardetzky  
Universität Göttingen

Eitan Grinspan  
Columbia University

Figure 1: A thin thread of viscous fluid is poured onto a moving belt, creating a dazzling array of intricate patterns. Simulations using our model reproduce this rich and complex behavior. Translucent thread: experiment (Chiu-Webster and Lister 2006); gold thread: simulation.

Abstract

We present a continuum-based discrete model for thin threads of viscous fluid by drawing upon the Rayleigh analogy to elastic rods, decomposing canonical coiling, folding, and breakup in dynamic simulations. Our derivation emphasizes space-time symmetry, which sheds light on the role of time-parallel transport in eliminating—without approximation—all but an  $O(1)$  band of entries of the physical system’s energy Hessian. The result is a fast, unified, implicit treatment of viscous threads and elastic rods that closely reproduces a variety of fascinating physical phenomena, including hysteretic transitions between coiling regimes, competition between surface tension and gravity, and the first numerical fluid-mechanical sewing machine. The novel implicit treatment also yields an order of magnitude speedup in our elastic rod dynamics.

CR Categories: 1.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism—Animation

Keywords: viscous threads, coiling, Rayleigh analogy, elastic rods, hair simulation

1 Introduction

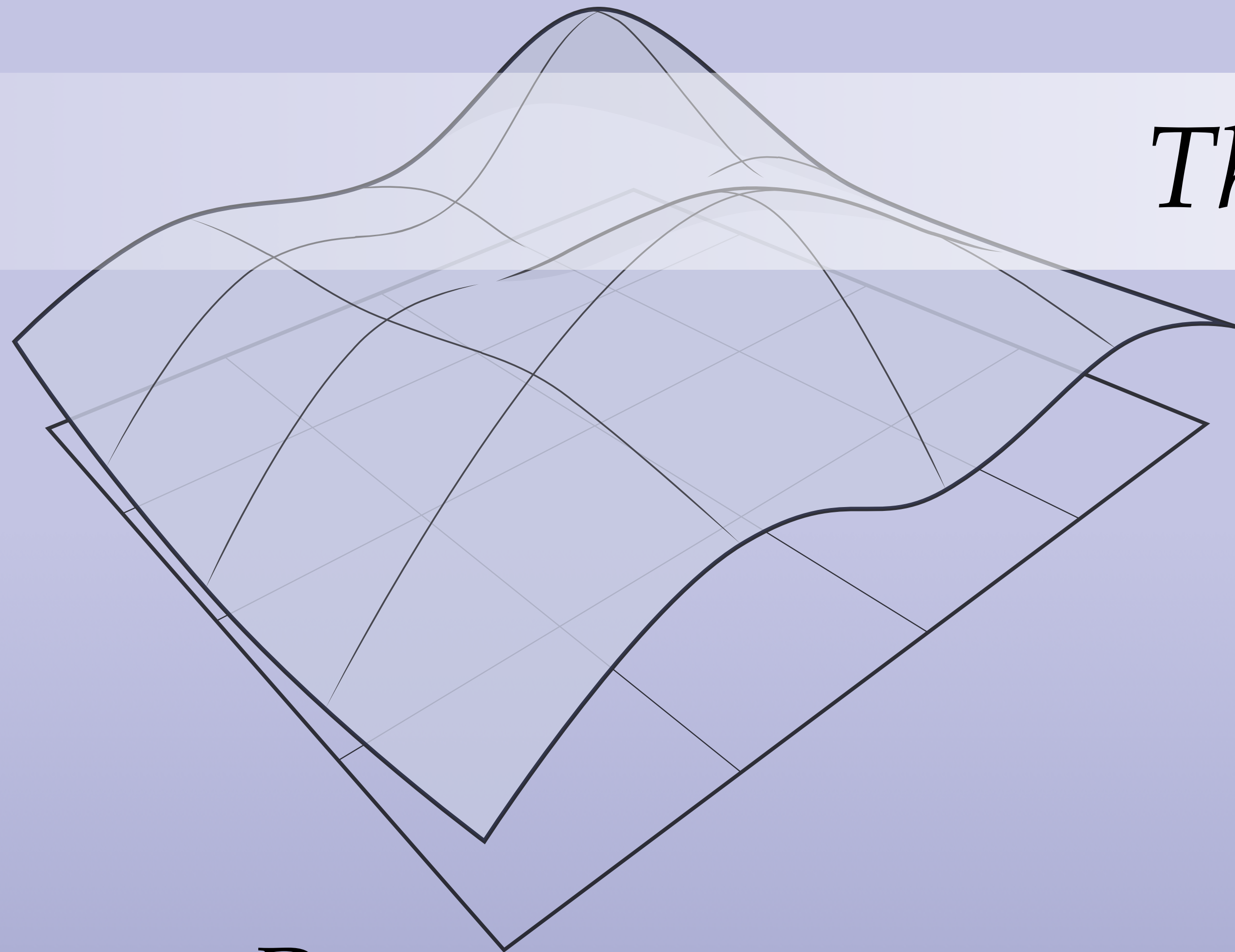
A curious little mystery of afternoon tea is the *folding, coiling, and unraveling* of a thin thread of honey as it falls upon a freshly baked scone. Understanding the motion of this viscous thread is a gateway to simulation tools whose utility spans film-making, gaming, and engineering: for example, in over 30% of worldwide textile manufacturing processes, threads of viscous liquid polymers (often incorporating recycled materials) are entangled to form nonwoven fabric used in baby diapers, bandages, envelopes, upholstery, air (“HEPA”) filters, surgical gowns, light-traffic capes, aviation control, felt, frost protection, and tea sachets [Andreassen et al. 1997].

Viscous threads display fascinating behaviors that are challenging to accurately reproduce with existing simulation techniques. For example, a viscous thread steadily poured onto a moving belt creates a sequence of “sewing machine” patterns (see Fig. 1). While in theory, it is possible to accurately compute the motion of a viscous thread using a general, volumetric fluid simulator, there are no reports of successes to date, perhaps because the resolution needed for a sufficiently accurate reproduction requires prohibitively expensive runtimes.

In contrast to volumetric approaches, we model viscous threads by their formal analogy to elastic rods, for which relatively inexpensive computational tools are readily available. Both viscous threads and elastic rods are amenable to a reduced-coordinate model operating on a centerline curve decorated with a cross-sectional material frame. Predicting the motion of viscous threads requires taking into account the competition between external forces, surface tension, and the material’s resistance to stretching, bending, and twisting rates. Thus, with the exception of surface tension, which generally plays a negligible role for elastic materials, an existing implementation of stretching, bending, and twisting for an elastic rod can be easily repurposed for simulating a viscous thread.



*Thanks!*



DISCRETE DIFFERENTIAL  
GEOMETRY:  
AN APPLIED INTRODUCTION

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