

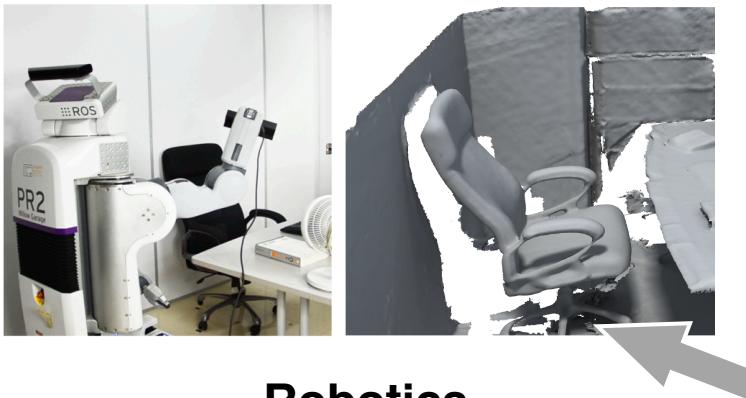
# L1: Introduction

Hao Su

# Agenda

- Syllabus
- Logistics
- Curve Theory

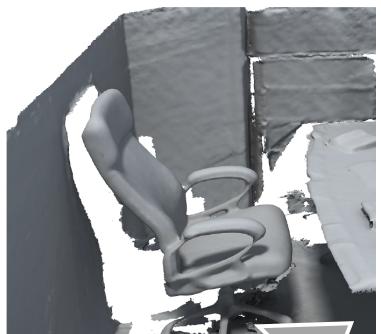
# Geometry Understanding is Important



Robotics



# Geometry Understanding is Important



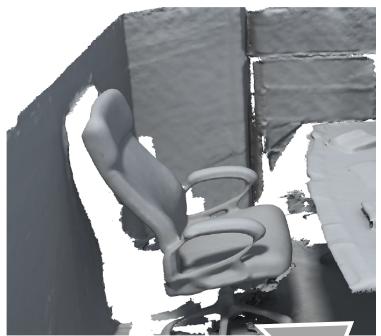
**Robotics**



**Augmented Reality**



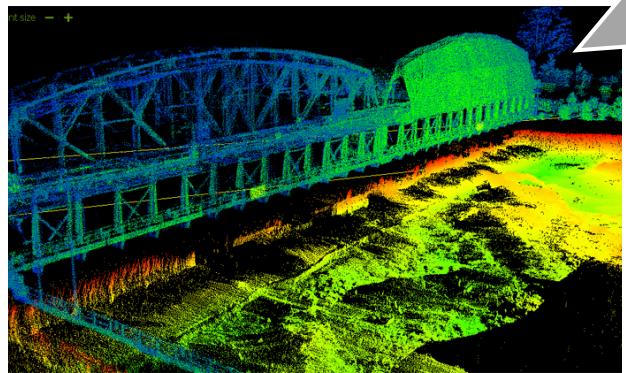
# Geometry Understanding is Important



Robotics

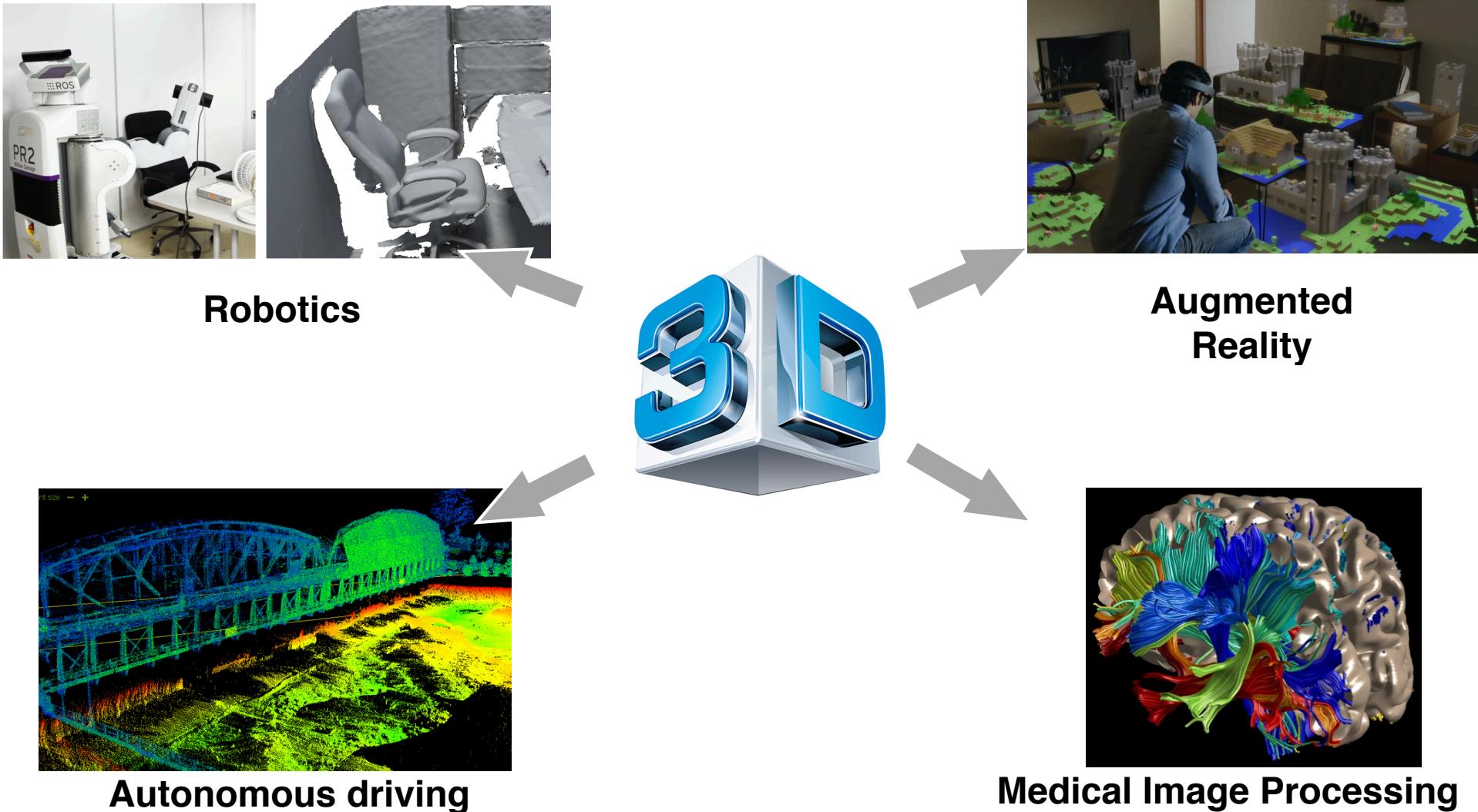


Augmented Reality

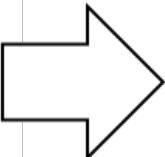


Autonomous driving

# Geometry Understanding is Important



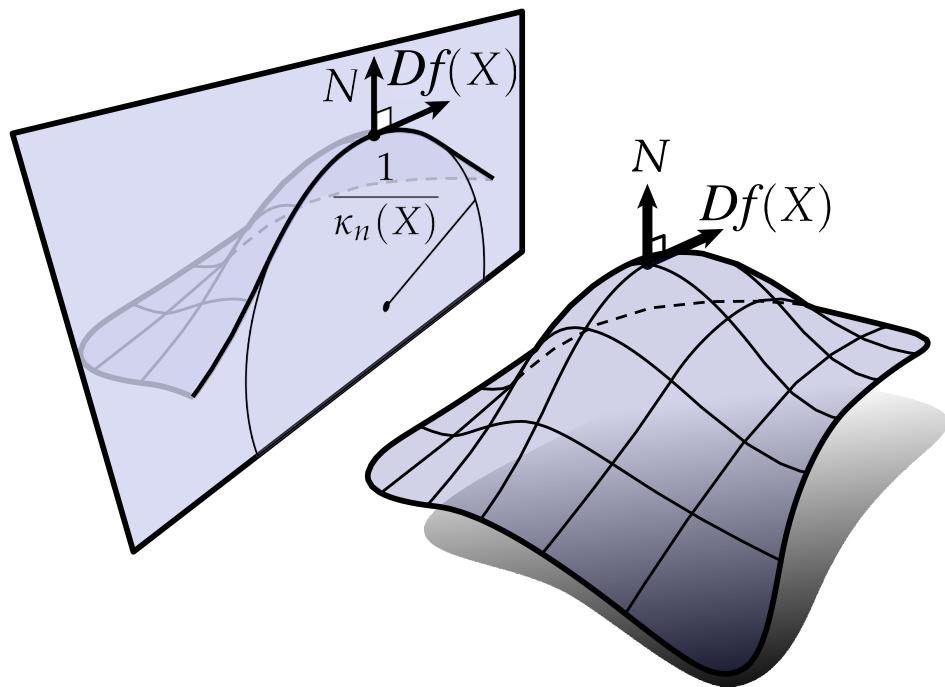
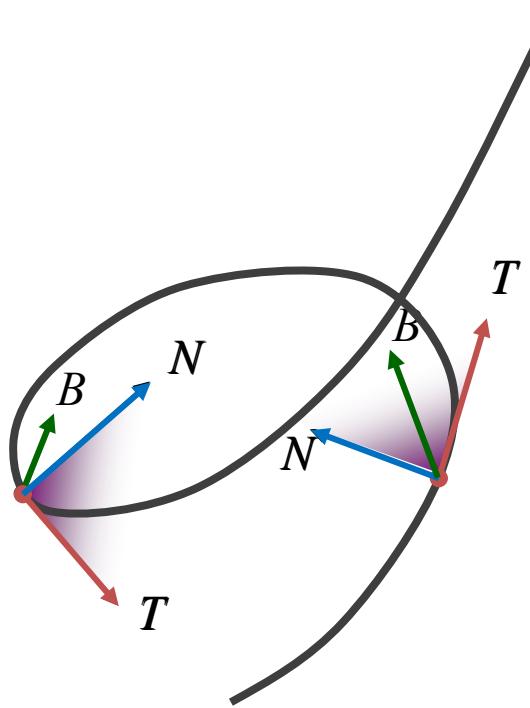
# Use Machine Learning to Understand Geometries



A priori knowledge of  
the 3D world

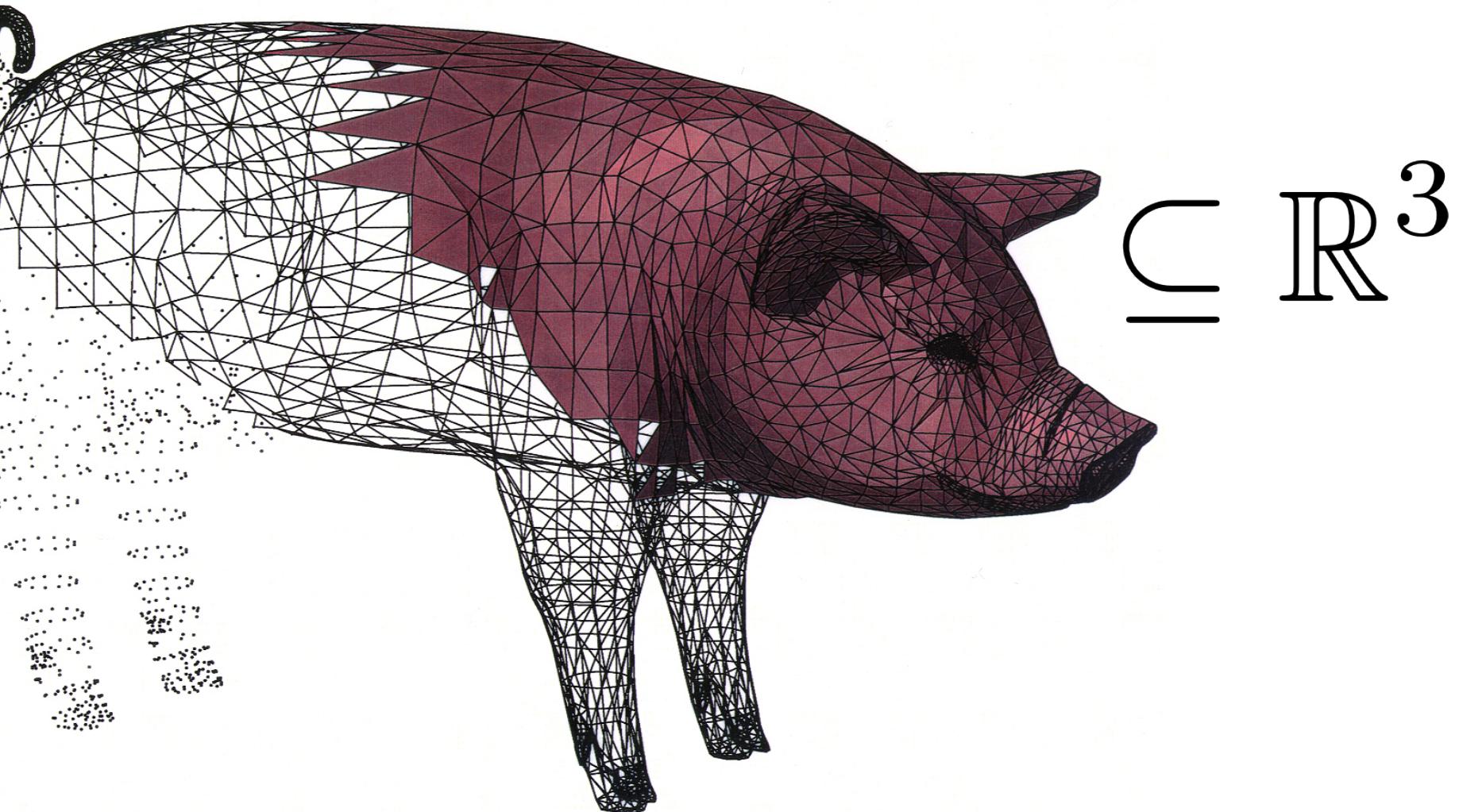
# Topics Covered in This Course

- Geometry theories



# Topics Covered in This Course

- Computer Representation of Geometries



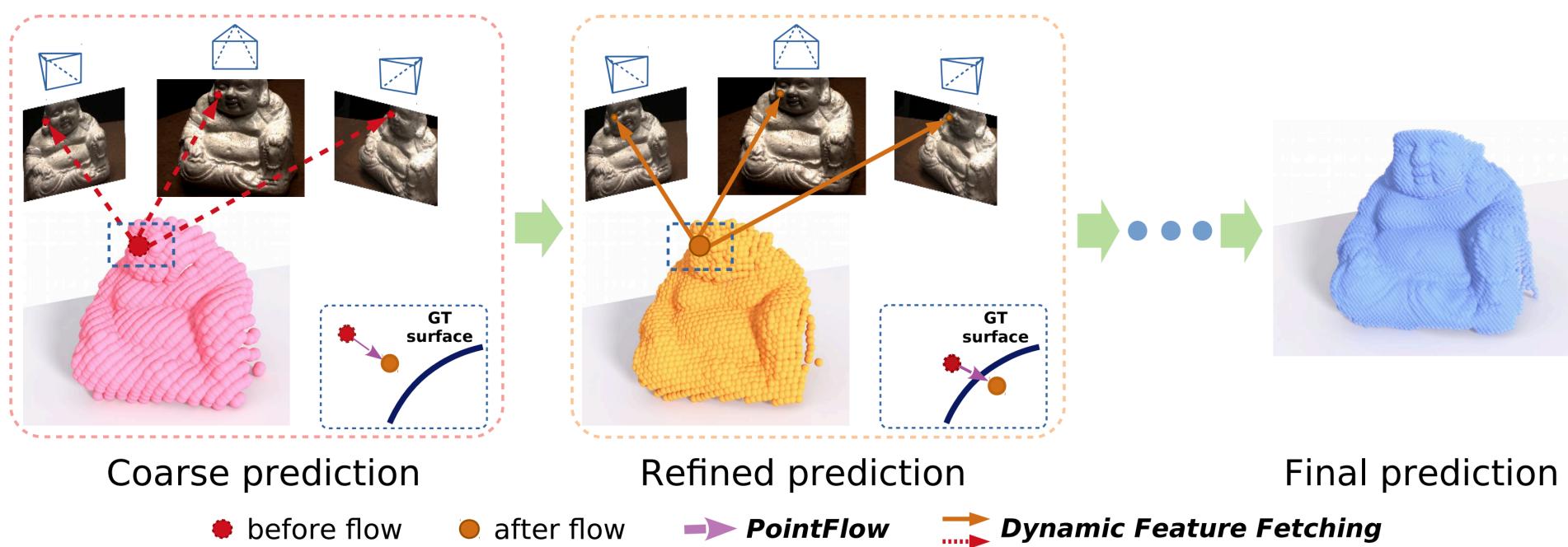
# Topics Covered in This Course

- Sensing: 3D reconstruction from a single image



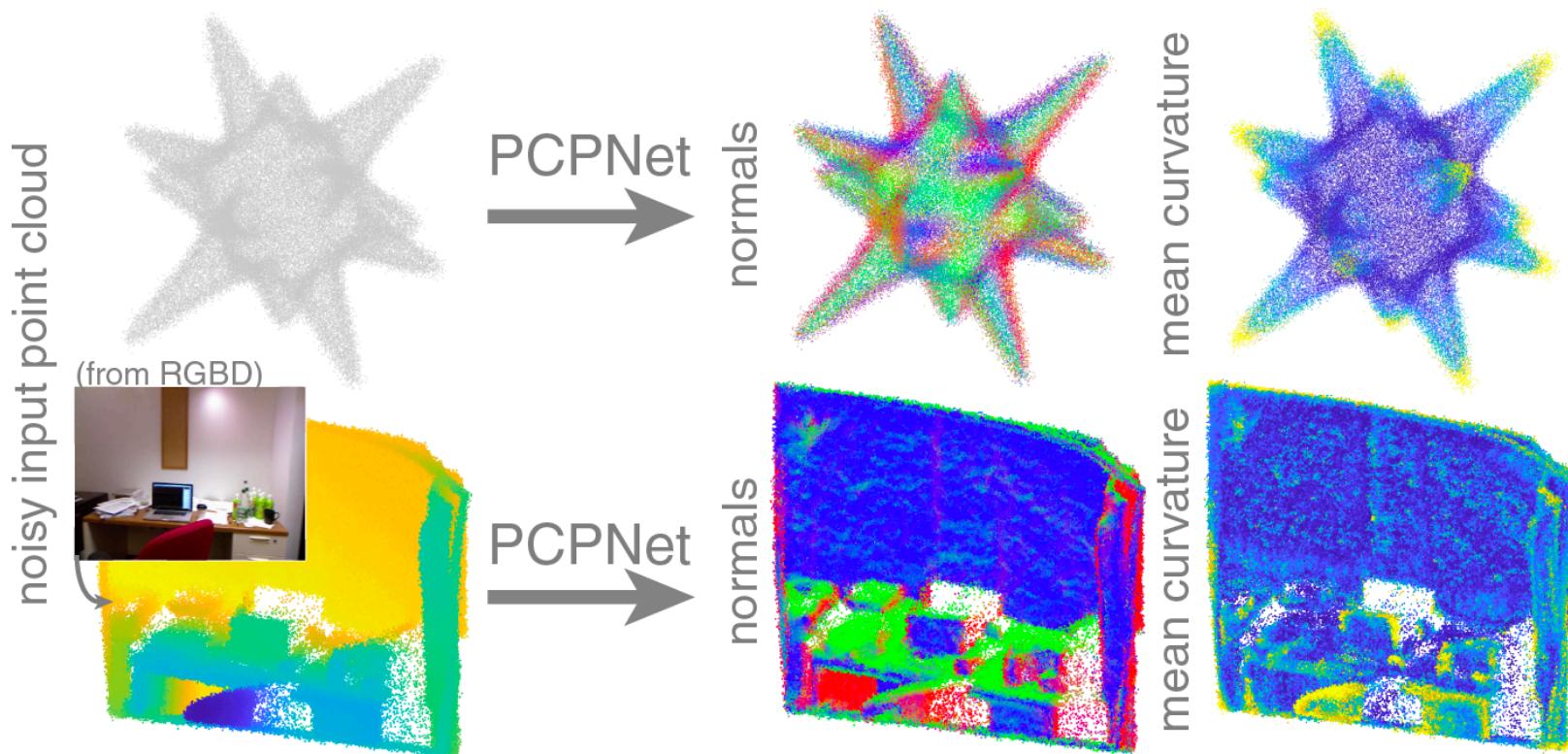
# Topics Covered in This Course

- Sensing: 3D reconstruction from multiple views



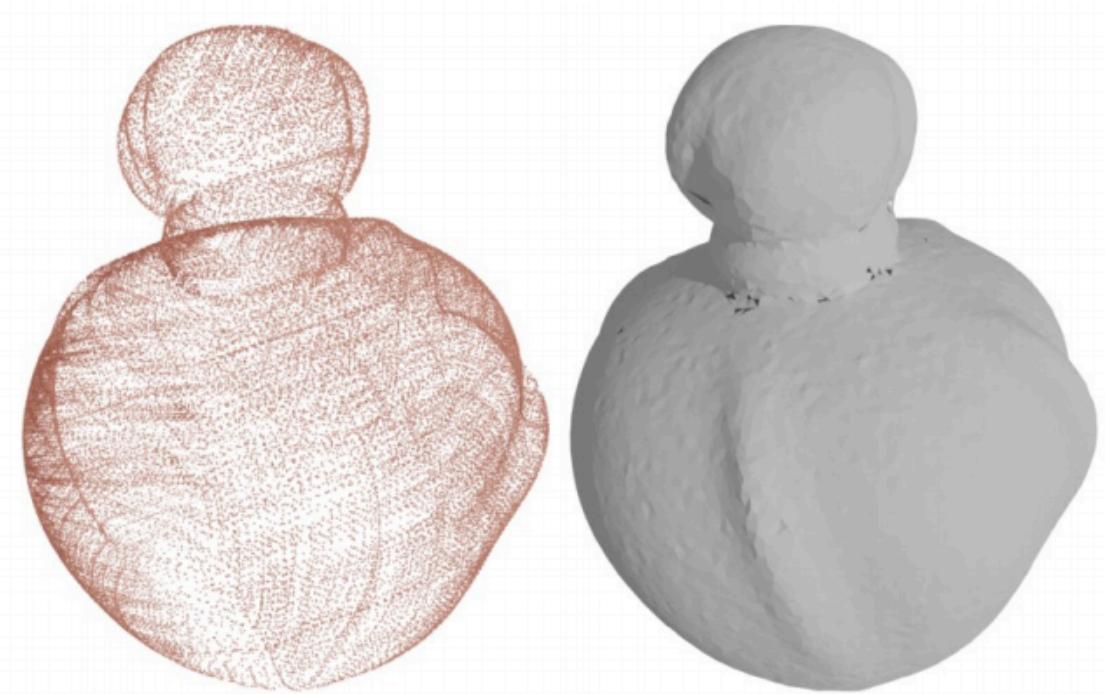
# Topics Covered in This Course

- Geometry Processing: Local geometric property estimation



# Topics Covered in This Course

- Geometry Processing: Surface reconstruction



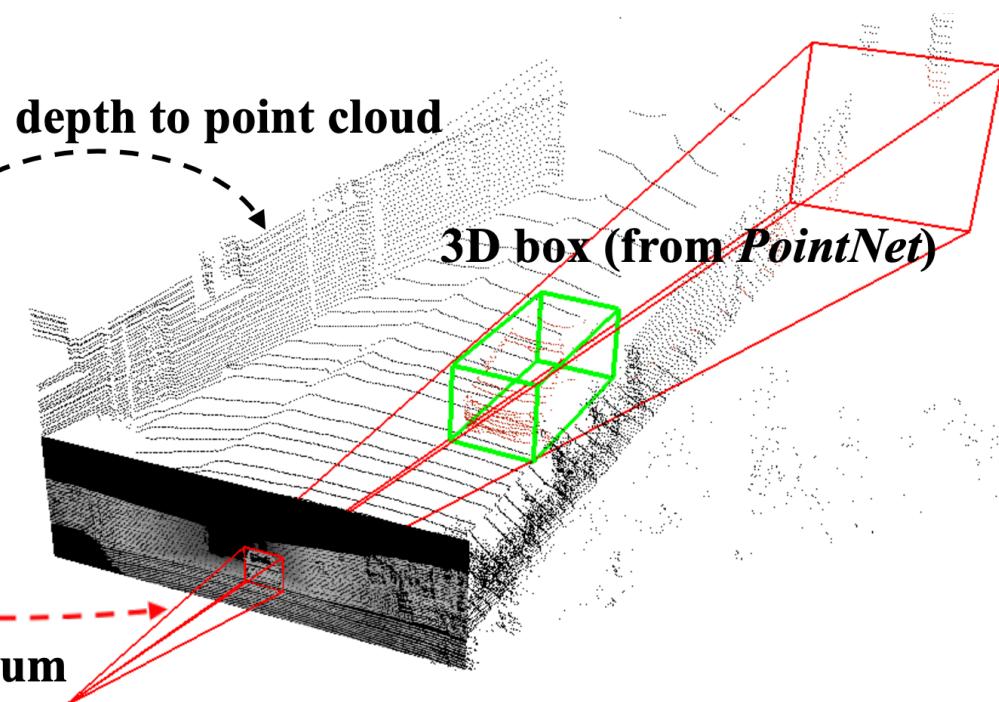
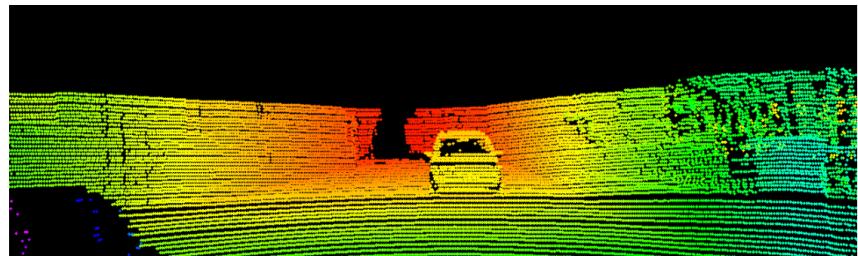
# Topics Covered in This Course

- Recognition: Object classification



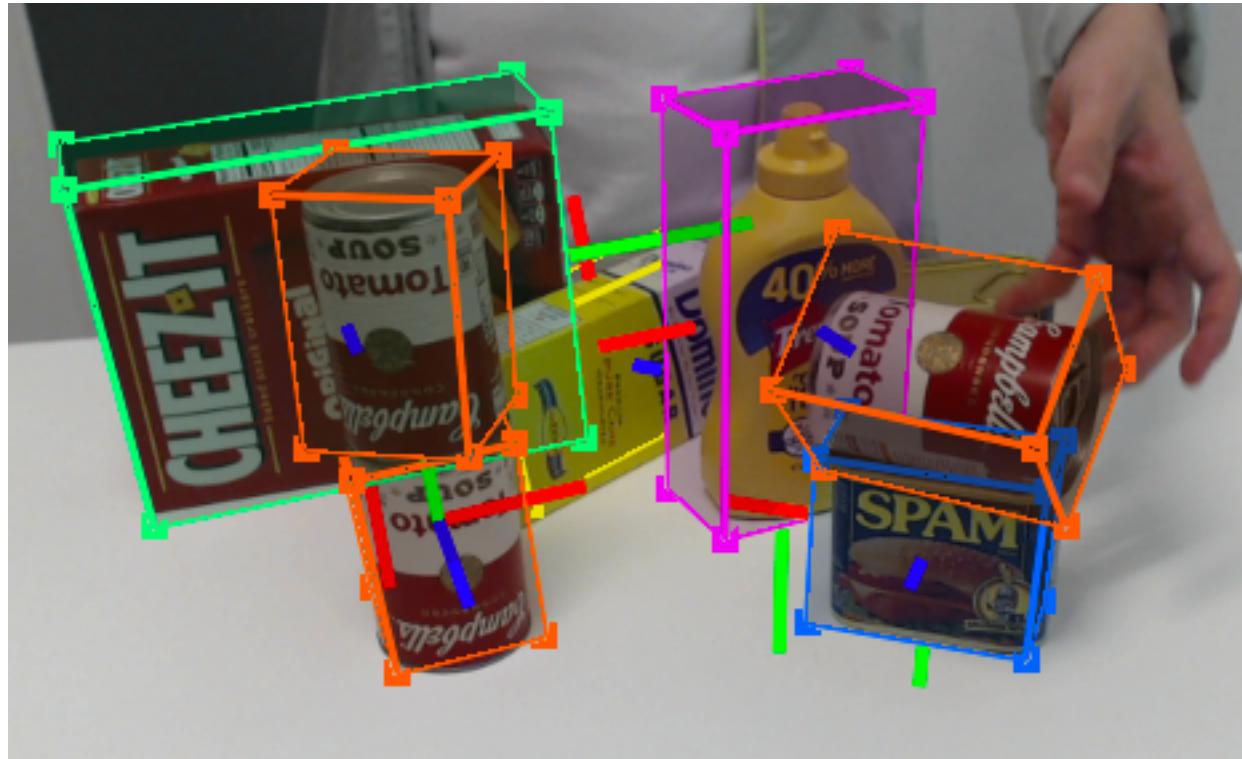
# Topics Covered in This Course

- Recognition: Object detection



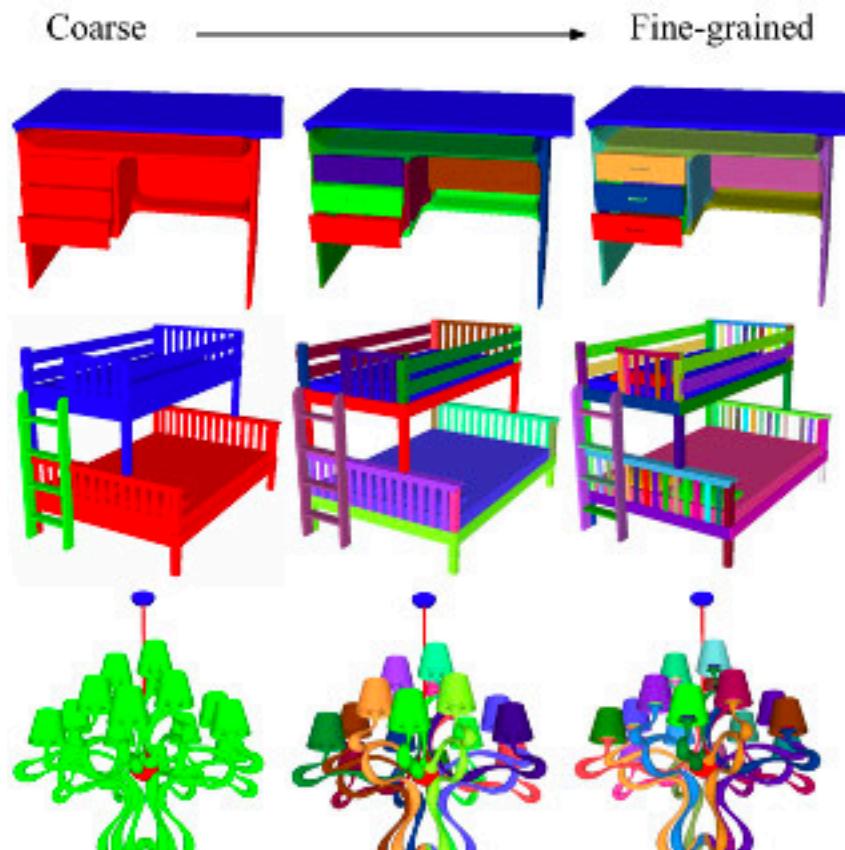
# Topics Covered in This Course

- Recognition: 6D pose estimation

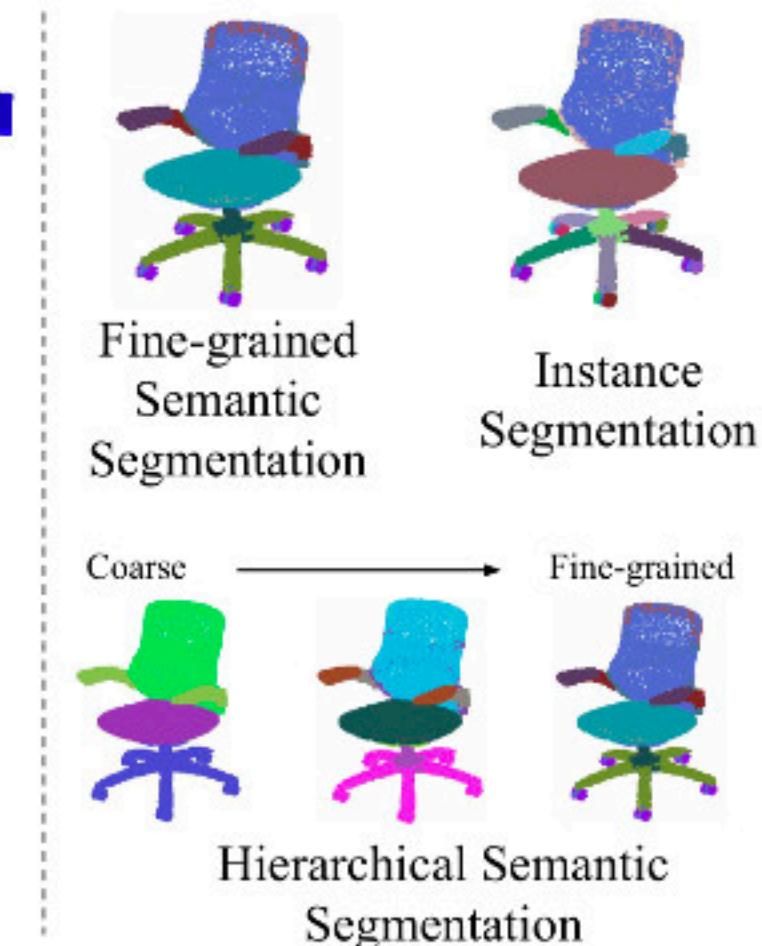


# Topics Covered in This Course

- Recognition: Segmentation

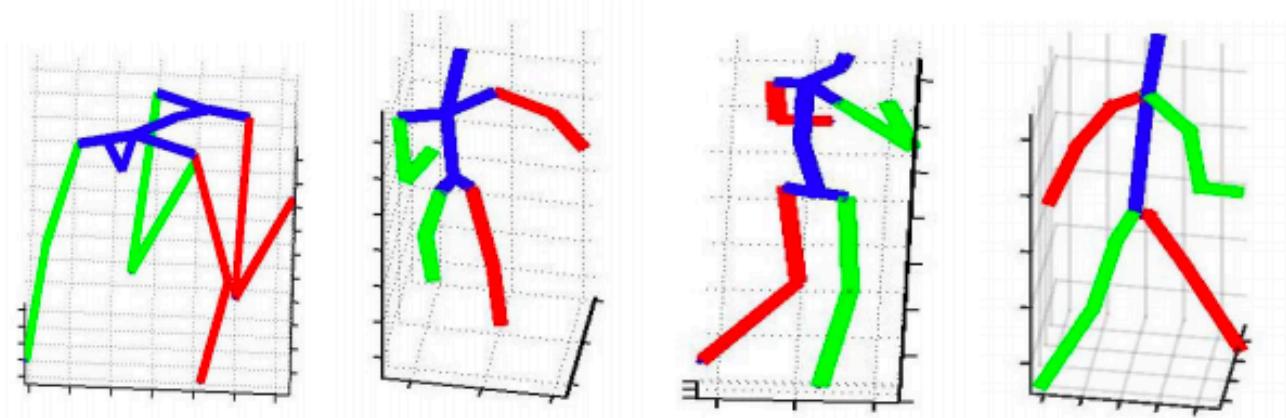
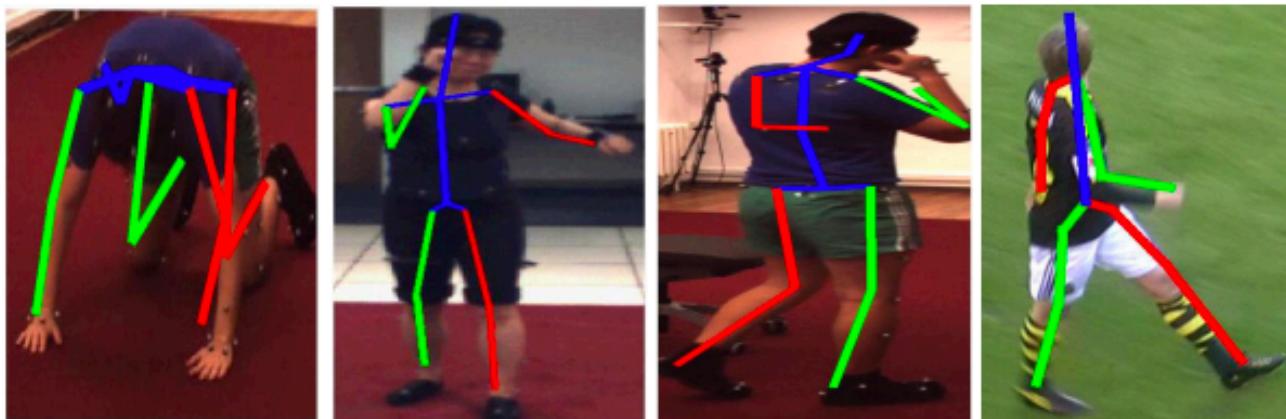


PartNet Dataset



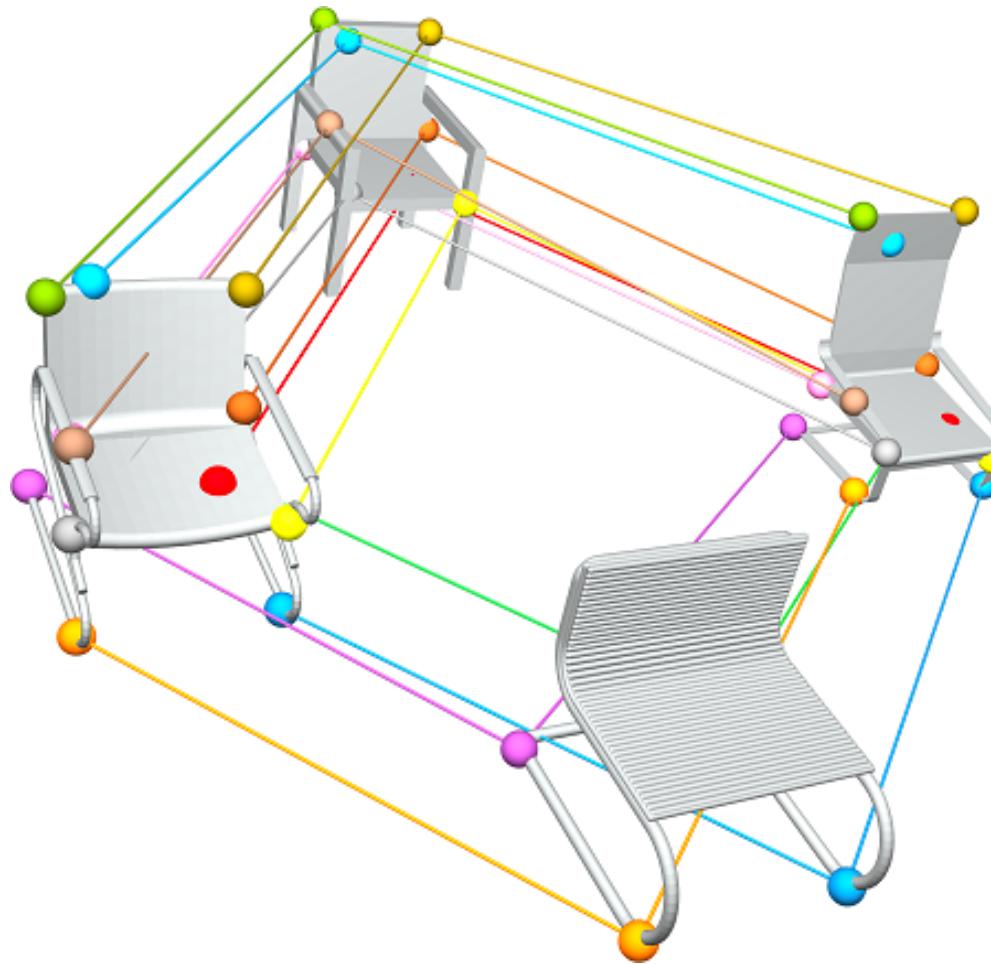
# Topics Covered in This Course

- Recognition: Human pose estimation

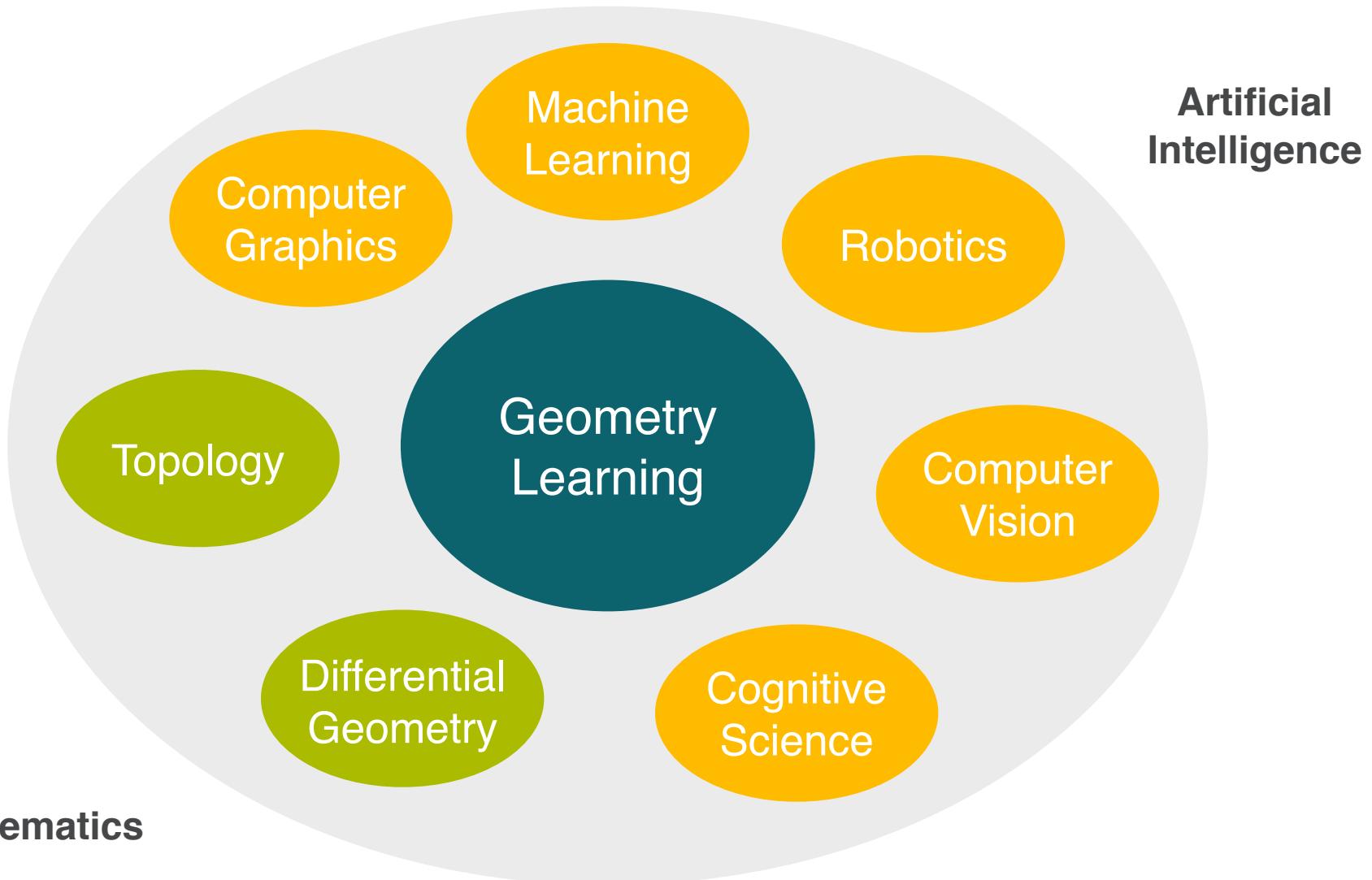


# Topics Covered in This Course

- Relationship Analysis: Shape correspondences



# Highly Interdisciplinary Field



# Course Logistic

# Instructors

Instructor: Hao Su



TA: Fanbo Xiang



# Teaching Goal

- State-of-the-art
  - **Enable** you to read and replicate recent 3D papers in top CV/CG conferences (not industry job oriented)
- Hands-on
  - **Heavy** programming assignments to exercise what are taught in class
- Foundational
  - Theory problems are **proof based**
  - Programming problems ask you to **implement low-level modules from scratch**

# Pre-requisite: Technique

- **Skilled** in Linear Algebra
- **Familiar** with Multi-variable Calculus
- **Familiar** with Probability and Numerical Methods
- **Strong** programming skills
  - Familiar with Linux Toolchain
  - Familiar with python, numpy, and pytorch
- Course/project experiences in computer vision or deep learning

# Background Check

- On Piazza now (HW0)
  - Visible to enrolled and waitlist students
- 5 points in your final grade
- **Mandatory!** We will not grade your subsequent homeworks without seeing your HW0.
- If you are in the waitlist and intend to enroll, you need to submit HW0 by this deadline
- **Due: 1/12/2021**

# Pre-requisite: Resources

- This course requires deep learning resources (to run a 3D recognition challenge)
- Unfortunately, we do not have computational resources to support ~50 students
- Please find the server with the following configuration:
  - $\geq 50G$  disk space
  - $\geq 1$  GPU with 10G memory

# Assignments

- 4 assignments and 1 final project
  - HW0: due week 2 (5 points)
  - HW1: due week 4 (20 points)
  - HW2: due week 6 (20 points)
  - HW3: due week 8 (20 points)
  - Final project: final week (35 points)
  - No mid-term/final exams
- Extra credit for participation 5% (ask/answer questions in class, attend office hours)
- HW0-HW3: theory problems + programming
- Late policy: 15% grade reduction for each 12 hours late. No acceptance 72 hours after the due time.

# 3D Recognition Competition

- HW0-HW3: build individual modules
- Final project: integrate modules and test new ideas. Score by performance ranking. Online evaluation system will be set up
- We estimate  **$\geq 15$  hrs per week** (out of class) solid time commitment
- We allow you to see homework (through Piazza) and attend the competition *even if you audit the course*

# Course Resources

- Course website: <https://haosulab.github.io/ml-meets-geometry/WI21/index.html> (Google “Hao Su” → Prof. Homepage → Teaching → this link)
  - Collaboration policy
  - Lecture slides
  - Office hour and location
- Piazza
  - Homework/Solution release
  - Discussions

# Office Hour

- Time to be determined.
- Please fill in our Piazza survey for information about time zone and etc.

# Questions?

# Curve

- Definition of curve
- Describing the shape of curves by calculus

# Parameterized Curves

Intuition:

- A particle is moving in space
- At time  $t$  its position is given by

$$\gamma(t) = (x(t), y(t))$$

# Example

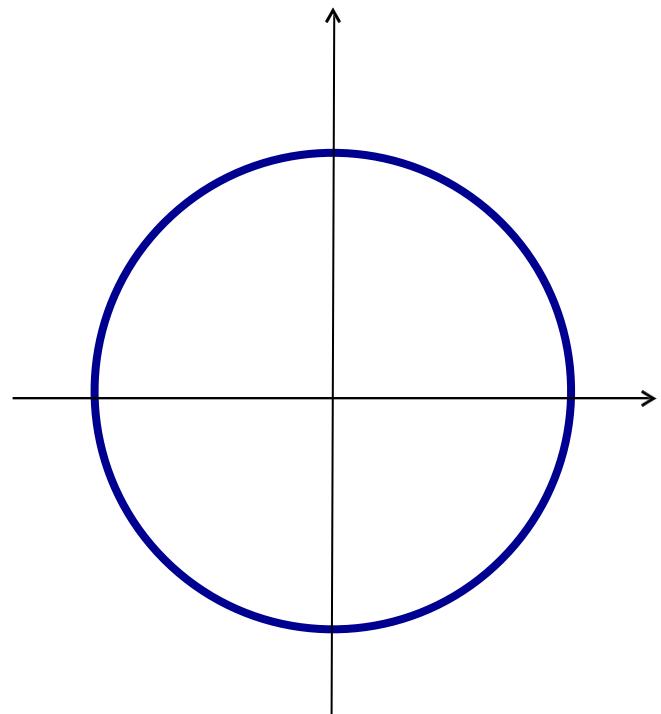
Explicit curve/circle in 2D

$$\mathbf{p} : \mathbb{R} \rightarrow \mathbb{R}^2$$

$$t \mapsto \mathbf{p}(t) = (x(t), y(t))$$

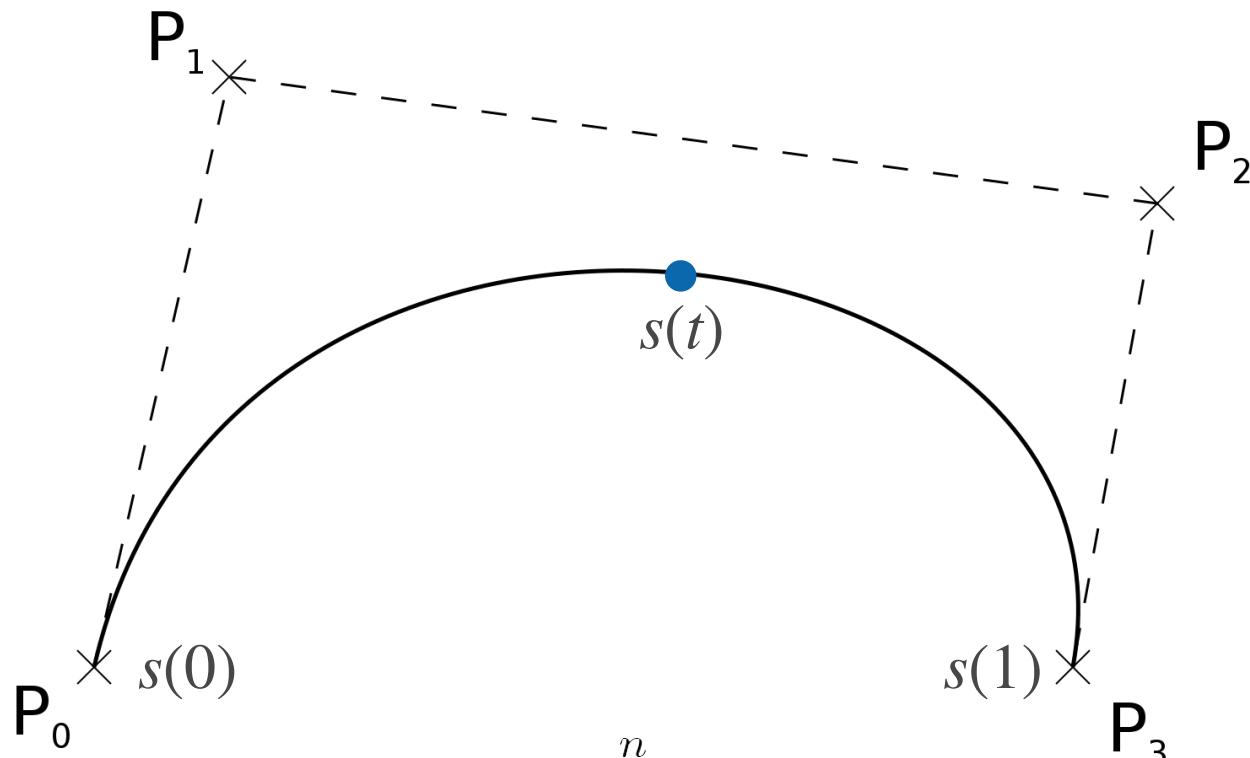
$$\mathbf{p}(t) = r (\cos(t), \sin(t))$$

$$t \in [0, 2\pi)$$



# Application: Bezier Curves, Splines

- Smoothly “interpolate” between a set of points  $P_i$
- Widely used in design (e.g., in your Powerpoint)



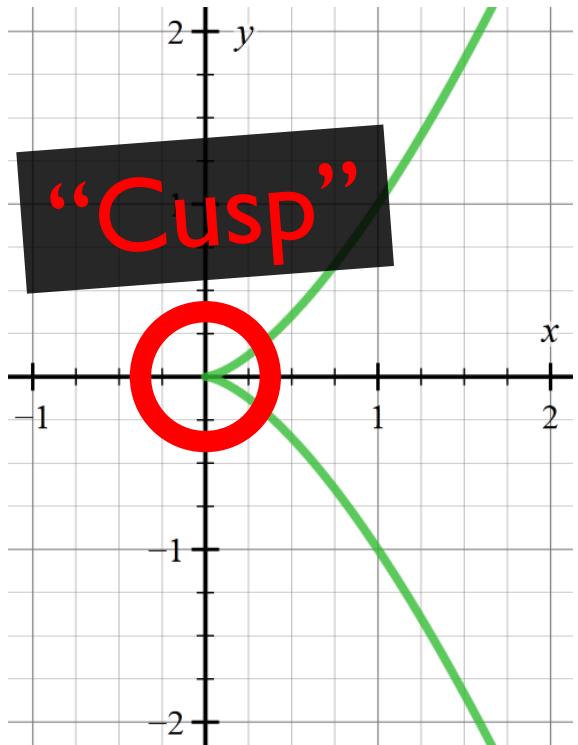
$$s(t) = \sum_{i=0}^n p_i B_i^n(t)$$

# One-dimensional “Manifold”

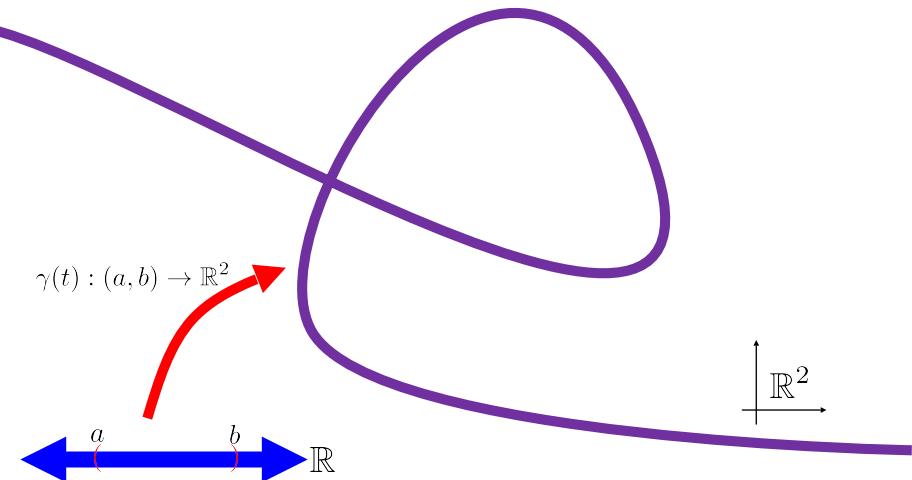


Set of points that locally looks like a line.

# Negative Examples of Manifolds



$$f(t) = (t^2, t^3)$$

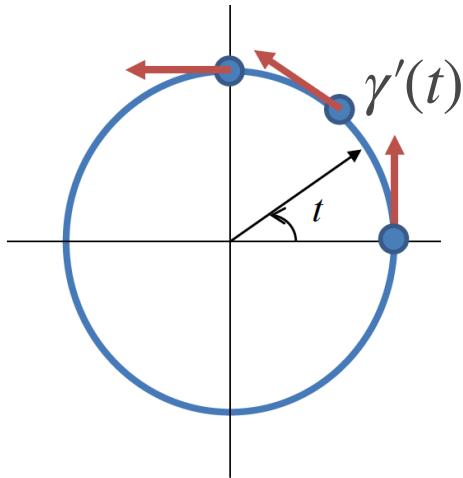


# Tangent

- $\gamma'(t) = (x'(t), y'(t)) \in \mathbb{R}^2$  is the tangent vector of the curve at  $t$

# Quiz: Tangent of a Circle

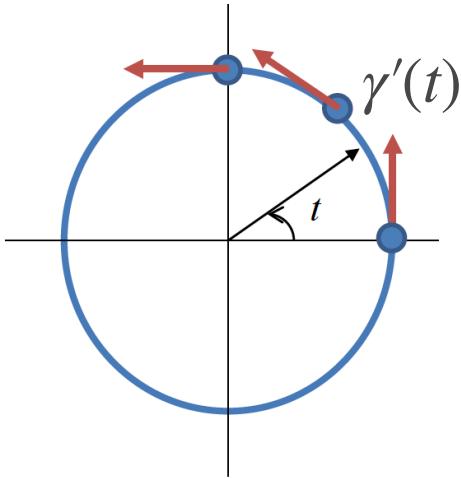
- $\gamma'(t) = (x'(t), y'(t)) \in \mathbb{R}^2$  is the tangent vector of the curve at  $t$



$$\gamma(t) = (\cos(t), \sin(t))$$

# Quiz: Tangent of a Circle

- $\gamma'(t) = (x'(t), y'(t)) \in \mathbb{R}^2$  is the tangent vector of the curve at  $t$



$$\gamma(t) = (\cos(t), \sin(t))$$

$$\gamma'(t) = (-\sin(t), \cos(t))$$

$\gamma'(t)$  - direction of movement

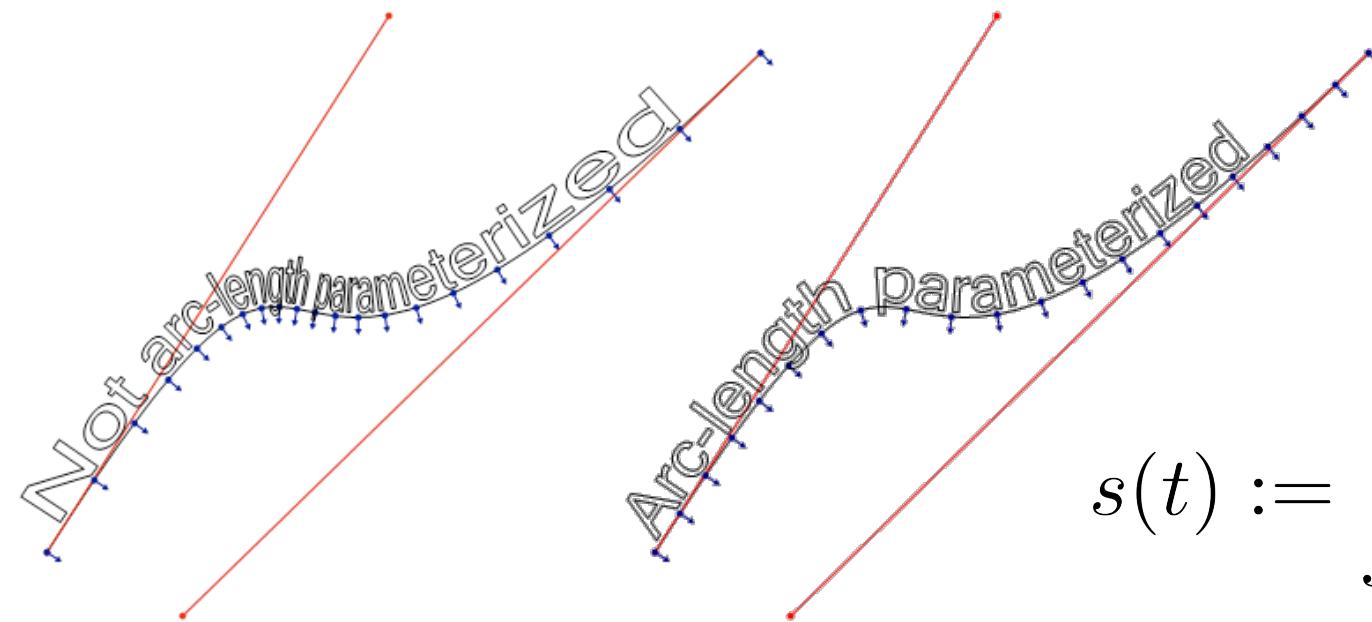
$\|\gamma'(t)\|$  - speed of movement

# Arc Length

$$\int_a^b \|\gamma'(t)\| dt$$

# Parameterization by Arc Length

<http://www.planetclegg.com/projects/WarpingTextToSplines.html>



$$s(t) := \int_{t_0}^t \|\gamma'(t)\| dt$$

$$t(s) := \text{inverse of } s(t)$$

$$\bar{\gamma}(s) = \gamma(t(s))$$

## Constant-speed parameterization

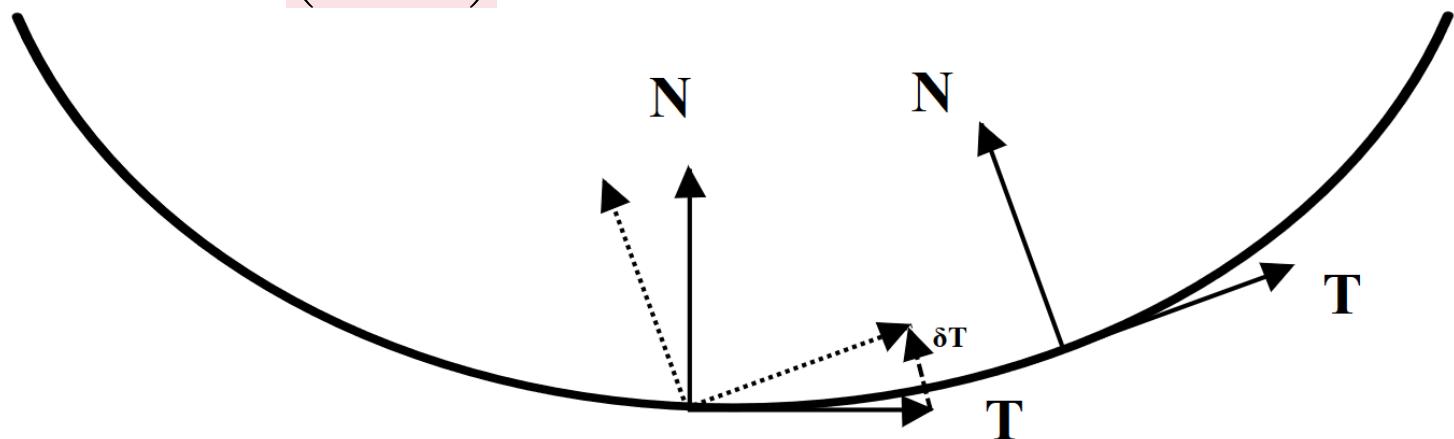
# Moving Frame in 2D

$$T(s) := \gamma'(s)$$

$\implies$  (on board)  $\|T(s)\| \equiv 1$

$$N(s) := JT(s)$$

$$\begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix}$$



# Lemma

$$\frac{d}{ds} \langle u(s), v(s) \rangle = \left\langle \frac{du}{ds}, v \right\rangle + \left\langle u, \frac{dv}{ds} \right\rangle$$

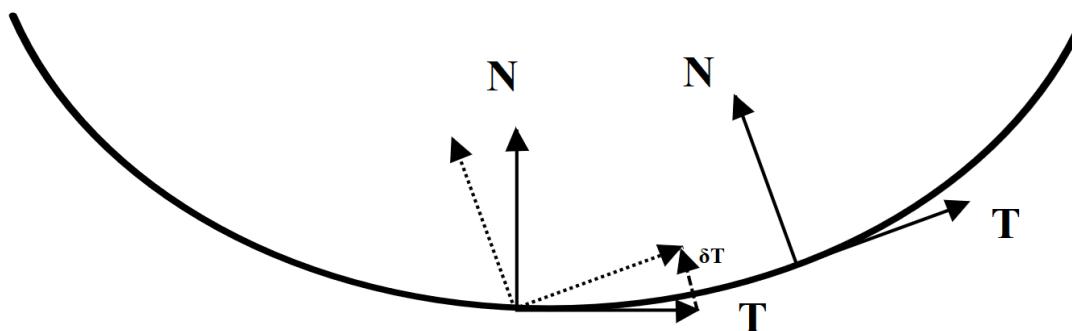
# Derivation of $\|T(s)\| \equiv 1$

(See notes)

# Turtles All The Way Down

On the board:

$$\frac{d}{ds} \begin{pmatrix} T(s) \\ N(s) \end{pmatrix} := \begin{pmatrix} 0 & k(s) \\ -k(s) & 0 \end{pmatrix} \begin{pmatrix} T(s) \\ N(s) \end{pmatrix}$$



[https://en.wikipedia.org/wiki/Frenet%E2%80%93Serret\\_formulas](https://en.wikipedia.org/wiki/Frenet%E2%80%93Serret_formulas)

Use coordinates *from the curve* to express its shape!

$$\frac{d}{ds} \begin{pmatrix} T(s) \\ N(s) \end{pmatrix} := \begin{pmatrix} 0 & k(s) \\ -k(s) & 0 \end{pmatrix} \begin{pmatrix} T(s) \\ N(s) \end{pmatrix}$$

(See notes)

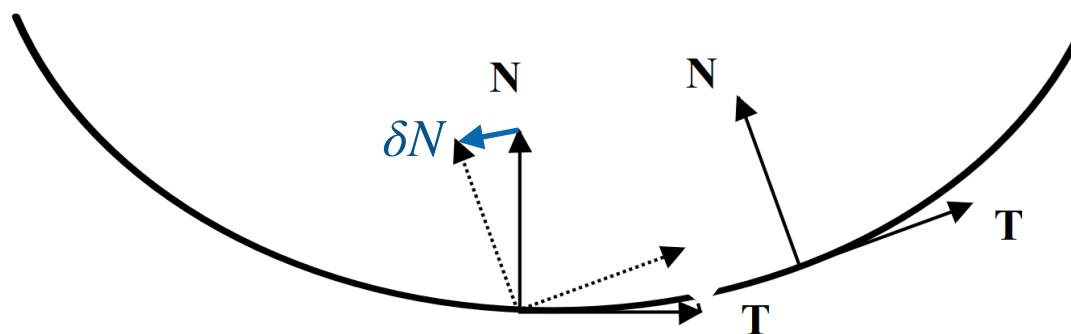
$$\frac{d}{ds} \begin{pmatrix} T(s) \\ N(s) \end{pmatrix} := \begin{pmatrix} 0 & k(s) \\ -k(s) & 0 \end{pmatrix} \begin{pmatrix} T(s) \\ N(s) \end{pmatrix}$$

(See notes)

# Perspective of Normal Change

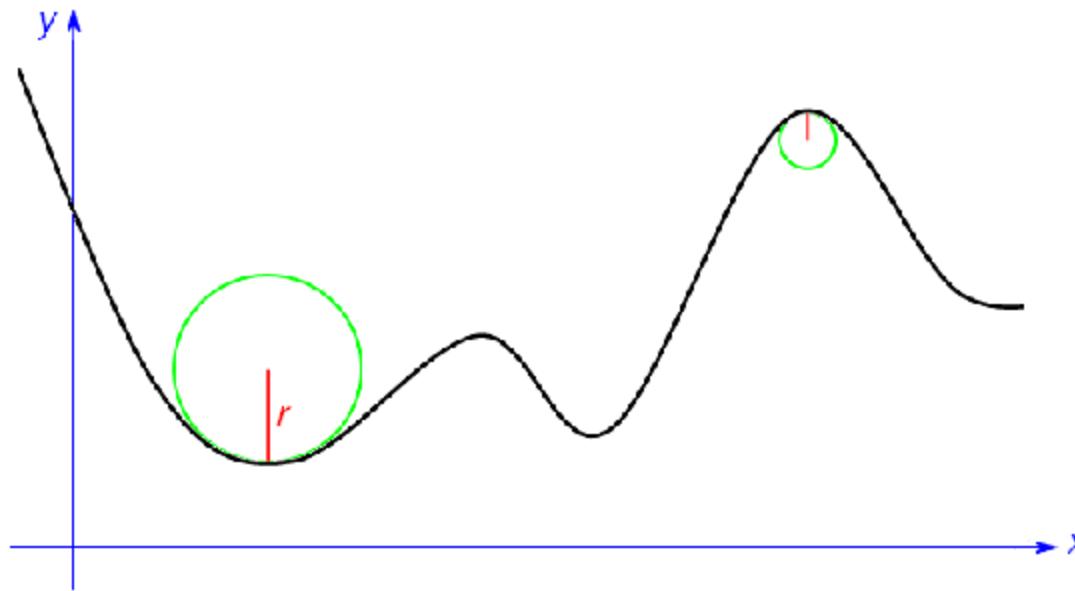
$$\mathbf{N}'(s) = -\kappa(s)\mathbf{T}(s)$$

- Curvature indicates how much the **normal** changes in the direction **tangent to the curve**



- Curvature is always positive

# Radius of Curvature



$$r(s) := \frac{1}{k(s)}$$

# Invariance is Important

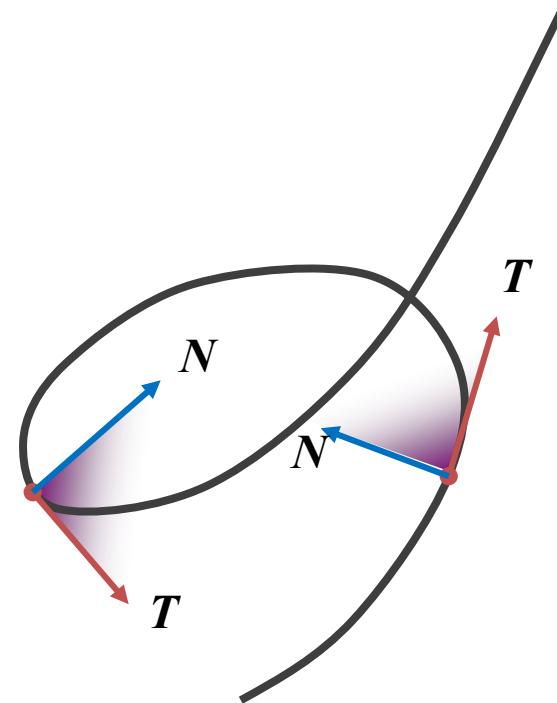
Fundamental theorem of the local theory of plane curves:

$\kappa(s)$  characterizes a **planar curve** up to rigid motion.

# 3D Curves

- Osculating Plane

The plane determined by the unit tangent and normal vectors  $T(s)$  and  $N(s)$  is called the *osculating plane* at  $s$

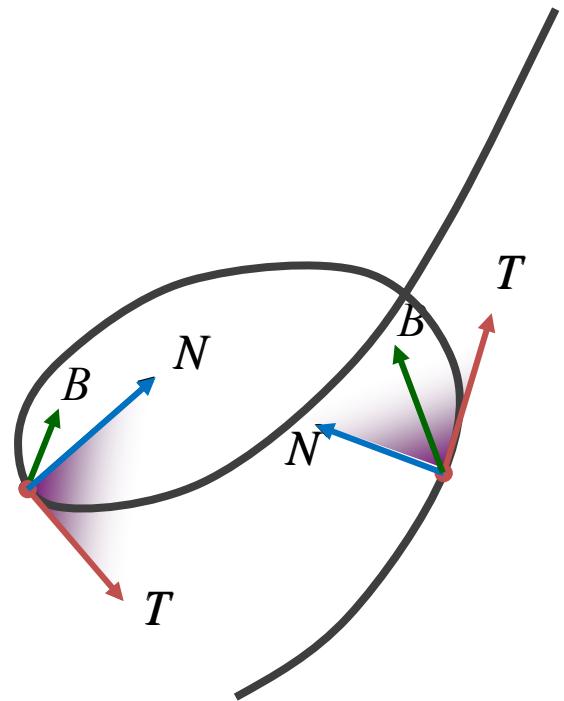


# The Binormal Vector

For points  $s$ , s.t.  $\kappa(s) \neq 0$ , the *binormal vector*  $B(s)$  is defined as:

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

The binormal vector defines the osculating plane



$$\mathbf{T}'(s)$$

- Already used it to define the curvature:

$$\mathbf{T}'(s) = \kappa(s) \boxed{\mathbf{N}(s)}$$

↑  
Unit vector

- Orthogonal to  $\mathbf{T}(s)$  (the same derivation as 2D curve)
- Since along the direction of  $\mathbf{N}(s)$ , also orthogonal to  $\mathbf{B}(s)$

$$\mathbf{N}'(s)$$

We know:  $\langle \mathbf{N}(s), \mathbf{N}(s) \rangle = 1$

From the lemma  $\longrightarrow \langle \mathbf{N}'(s), \mathbf{N}(s) \rangle = 0$

(Derivative orthogonal to itself)

We know:  $\langle \mathbf{N}(s), \mathbf{T}(s) \rangle = 0$

From the lemma  $\longrightarrow \langle \mathbf{N}'(s), \mathbf{T}(s) \rangle = \langle -\mathbf{N}(s), \mathbf{T}'(s) \rangle$

From the definition  $\longrightarrow \kappa(s) = \langle \mathbf{N}(s), \mathbf{T}'(s) \rangle$

$\longrightarrow \langle \mathbf{N}'(s), \mathbf{T}(s) \rangle = -\kappa(s)$

# The Torsion

- From previous slide:

$$\langle \mathbf{N}'(s), \mathbf{N}(s) \rangle = 0$$

$$\langle \mathbf{N}'(s), \mathbf{T}(s) \rangle = -\kappa(s)$$

The remaining component of  $\mathbf{N}'(s)$  is along  $\mathbf{B}(s)$  direction:

$$\langle \mathbf{N}'(s), \mathbf{B}(s) \rangle = \tau(s)$$

Now we can express  $N'(s)$  as

$$\mathbf{N}'(s) = -\kappa(s)\mathbf{T}(s) + \tau(s)\mathbf{B}(s)$$

# Perspective of Normal Change

$$\mathbf{N}'(s) = -\kappa(s)\mathbf{T}(s) + \tau(s)\mathbf{B}(s)$$

- Curvature indicates how much the normal changes in the direction tangent to the curve
- Torsion indicates how much normal changes in the direction orthogonal to the osculating plane of the curve
- Curvature is always positive but torsion can be negative

$$\mathbf{B}'(s)$$

We know:  $\langle \mathbf{B}(s), \mathbf{B}(s) \rangle = 1$

From the lemma  $\rightarrow \langle \mathbf{B}'(s), \mathbf{B}(s) \rangle = 0$

We know:  $\langle \mathbf{B}(s), \mathbf{T}(s) \rangle = 0, \langle \mathbf{B}(s), \mathbf{N}(s) \rangle = 0$

From the lemma  $\rightarrow$

$$\langle \mathbf{B}'(s), \mathbf{T}(s) \rangle = \langle -\mathbf{B}(s), \mathbf{T}'(s) \rangle = \langle -\mathbf{B}(s), \kappa(s)\mathbf{N}(s) \rangle = 0$$

From the lemma  $\rightarrow$

$$\langle \mathbf{B}'(s), \mathbf{N}(s) \rangle = \langle -\mathbf{B}(s), \mathbf{N}'(s) \rangle = -\tau(s)$$

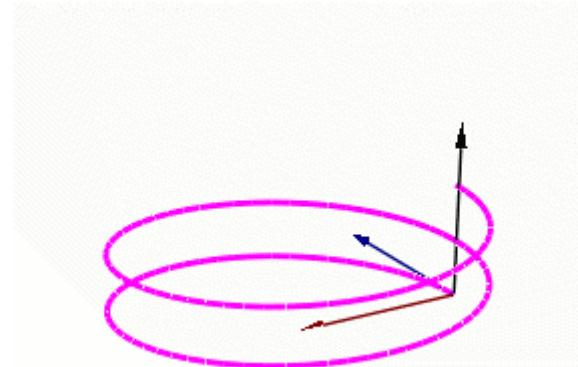
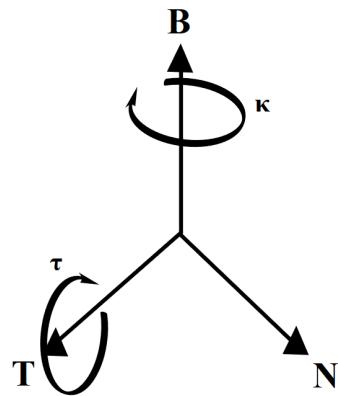
Now we express  $\mathbf{B}'(s)$  as:

$$\mathbf{B}'(s) = -\tau(s)\mathbf{N}(s)$$

# Frenet Frame: Curves in $\mathbb{R}^3$

- Binormal:
  - **Curvature:** In-plane motion
  - **Torsion:** Out-of-plane motion

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



**Self-reading**

# Fundamental theorem of the local theory of space curves:

Curvature and torsion  
characterize a 3D curve up to  
rigid motion.

# Summary

- Curve is a map from an interval to  $\mathbb{R}^n$
- Tangent describes the moving direction
- The derivative of tangent under arc-length parameterization is normal
- Curvature (and torsion) both characterize the change of normal direction, uniquely describing the shape of a curve (up to rigid transformation)
- Tangent, normal, and binormal form a moving frame (Frenet frame)