

# L1: Introduction

Hao Su



- <https://www.youtube.com/watch?v=fn3KWM1kuAw>

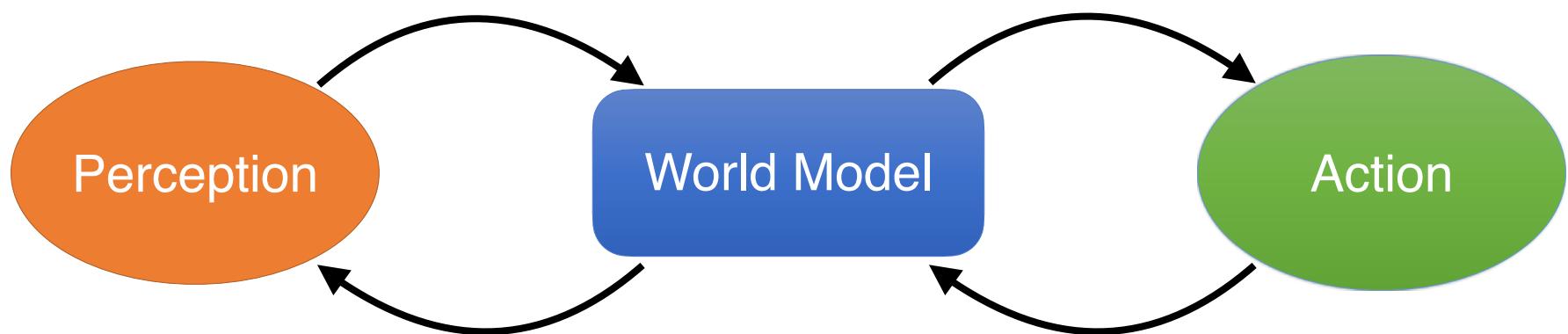
# Agenda

- Syllabus
- Logistics
- $\text{SO}(3)$

# Syllabus

Last quarter

This quarter



# Vision → Robotics

Passive AI

- We know how to fit data well (by “deep learning”)
  - e.g., computer vision, natural language processing

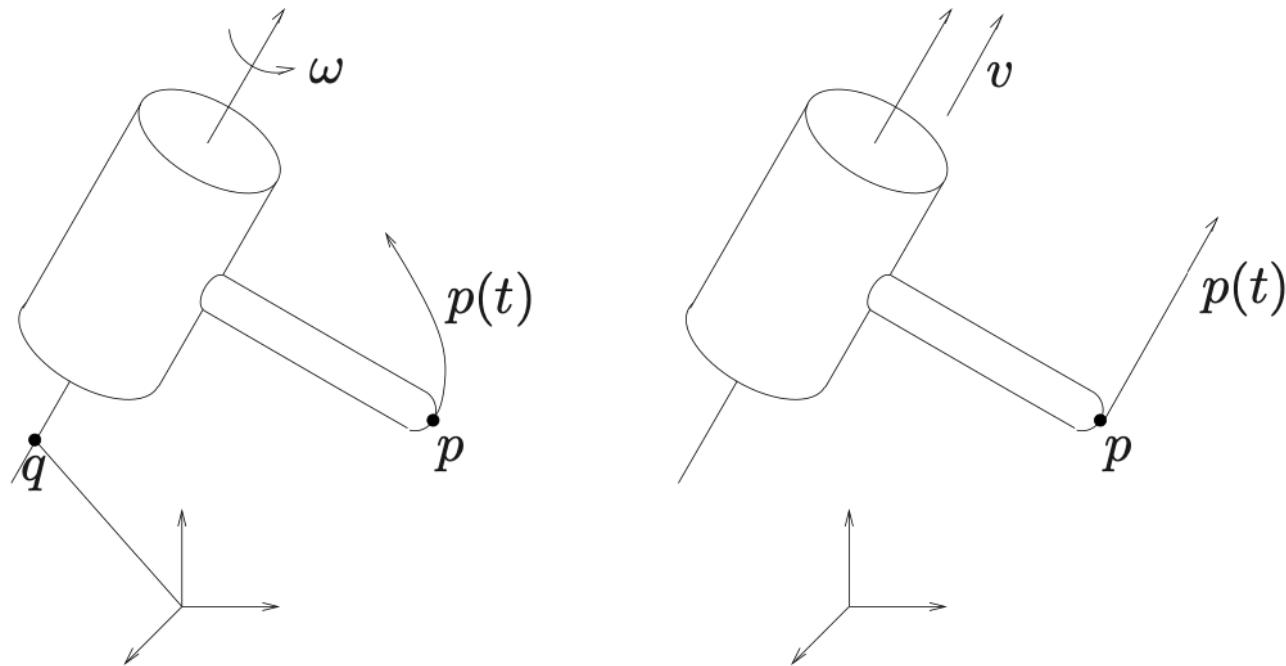
# Vision → Robotics



- We aspire that autonomous agents can perform tasks and “grow” through interaction experiences
  - Need the ability to **interact**

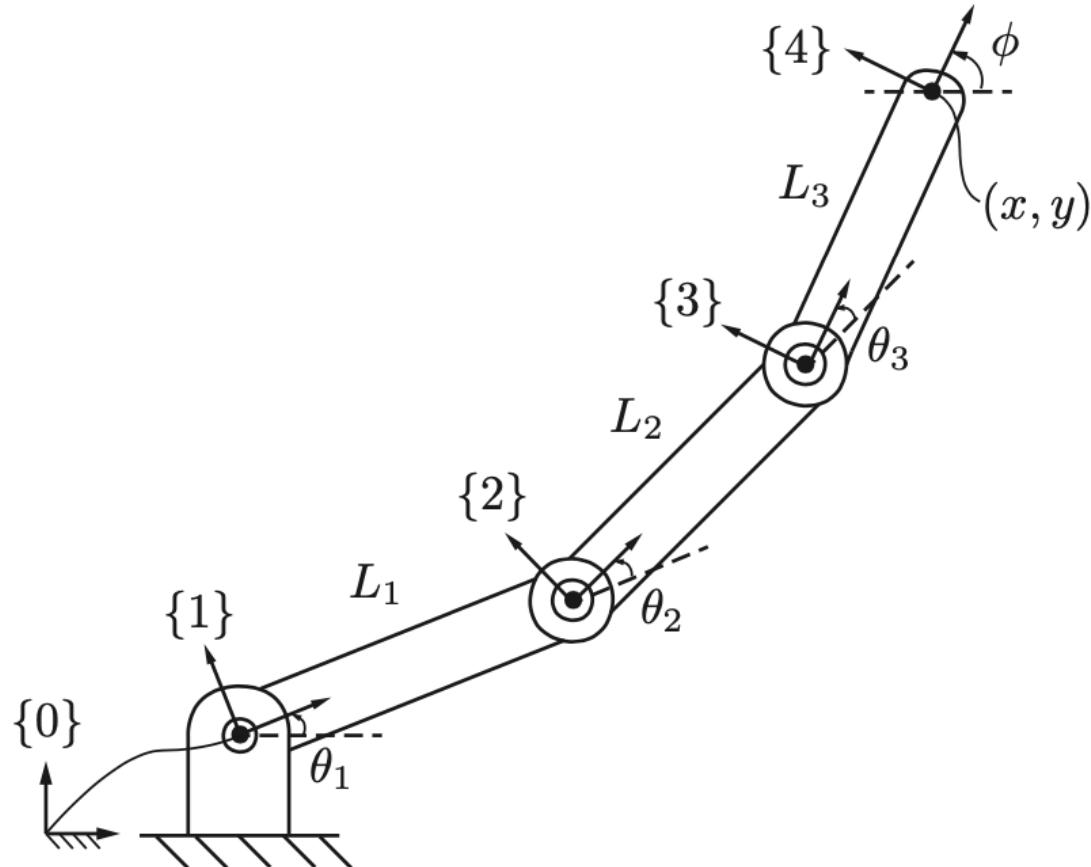
# Topics Covered in This Course

- Modeling Robots by Rigid-Body Geometry



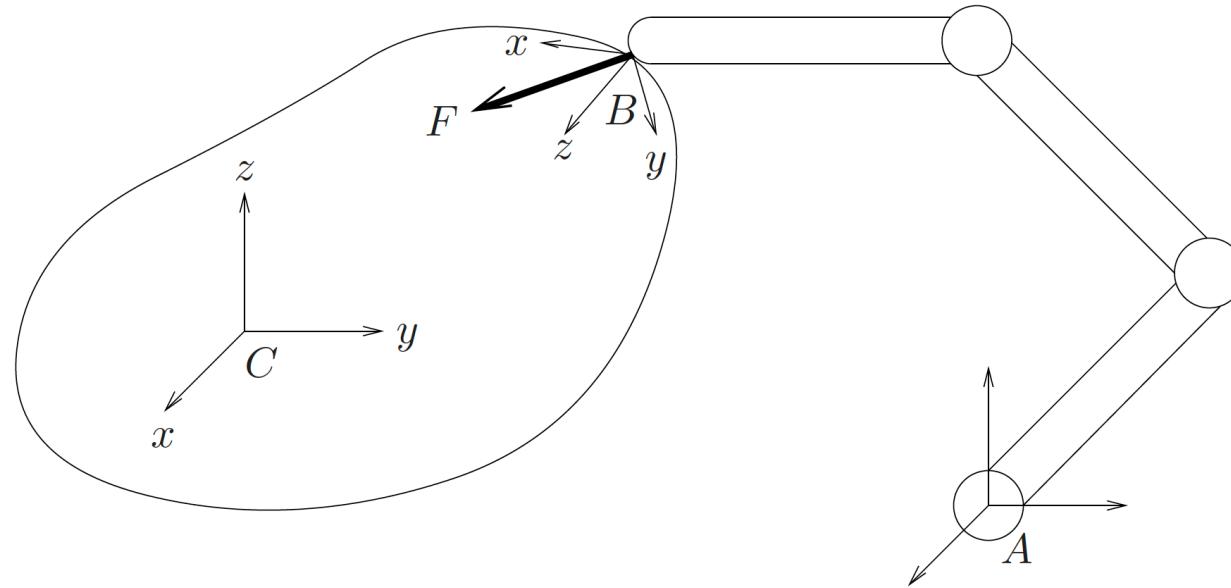
# Topics Covered in This Course

- Forward and Inverse Kinematics of Robots



# Topics Covered in This Course

- Generalized Force and Inertia



# Topics Covered in This Course

- Friction, Contact Model, and Grasp

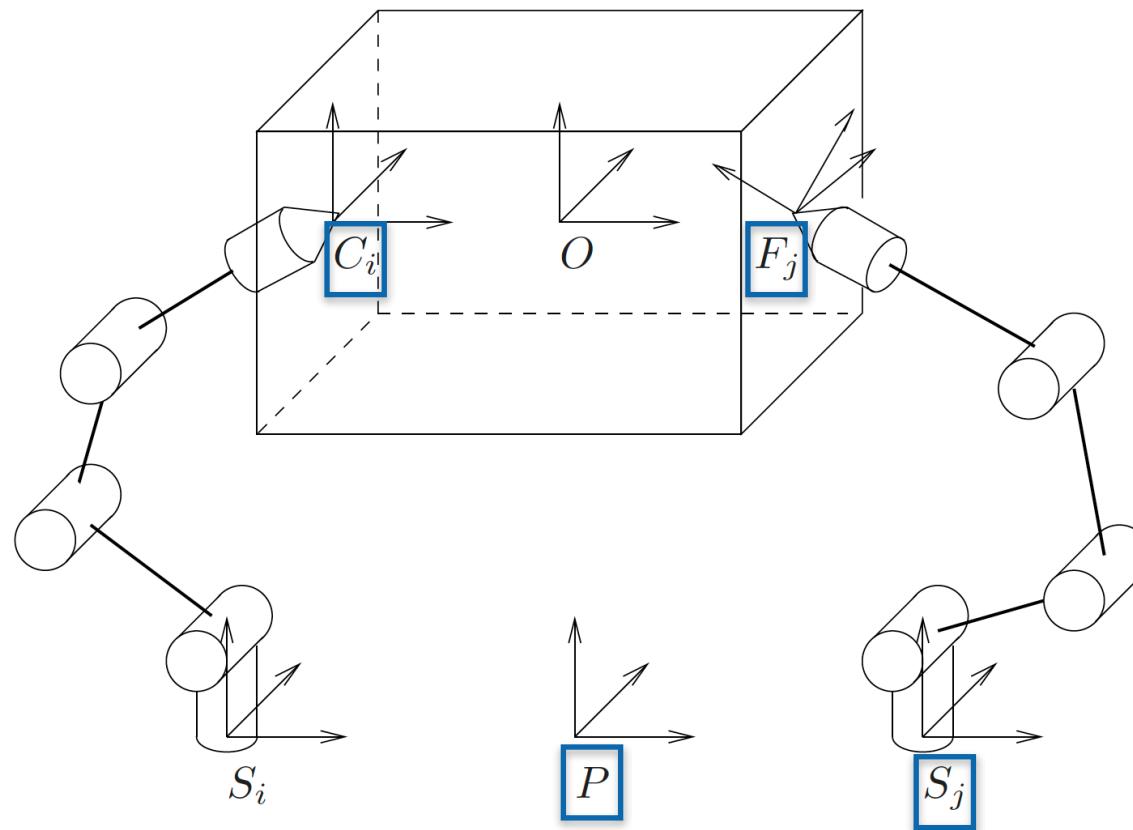
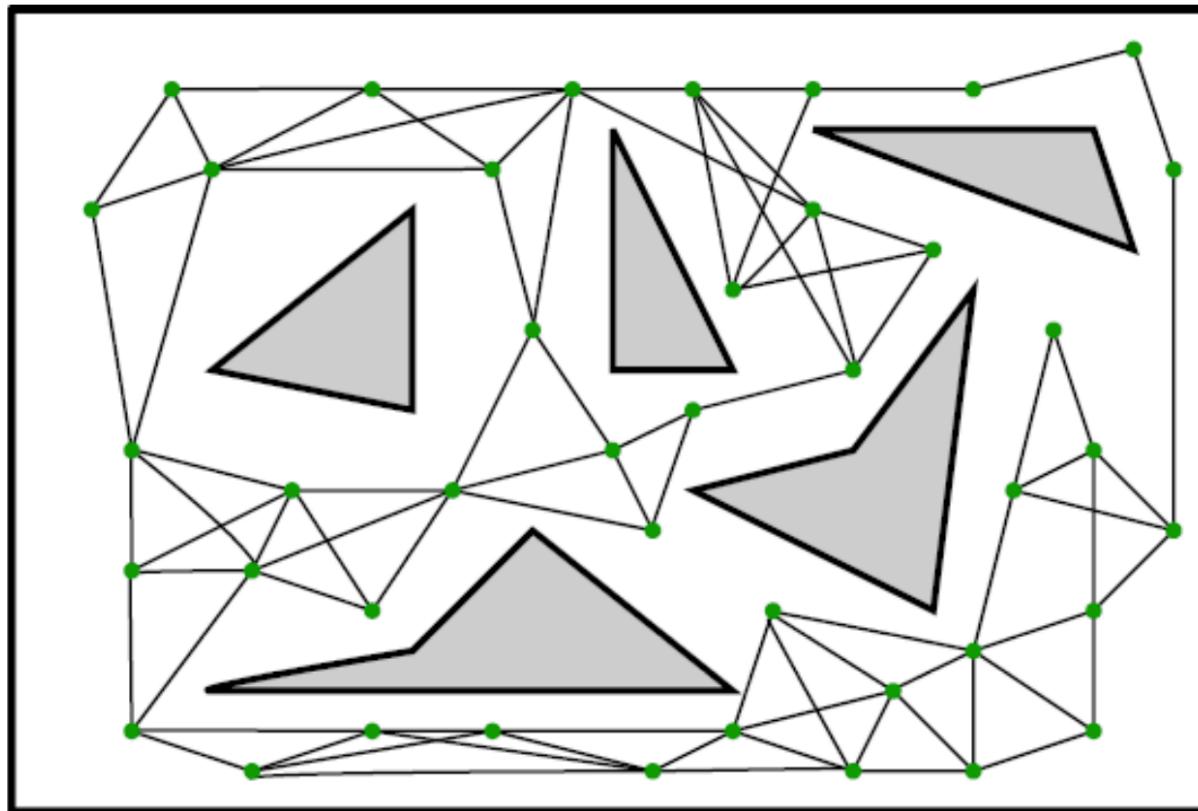


Figure 5.14: Grasp coordinate frames.

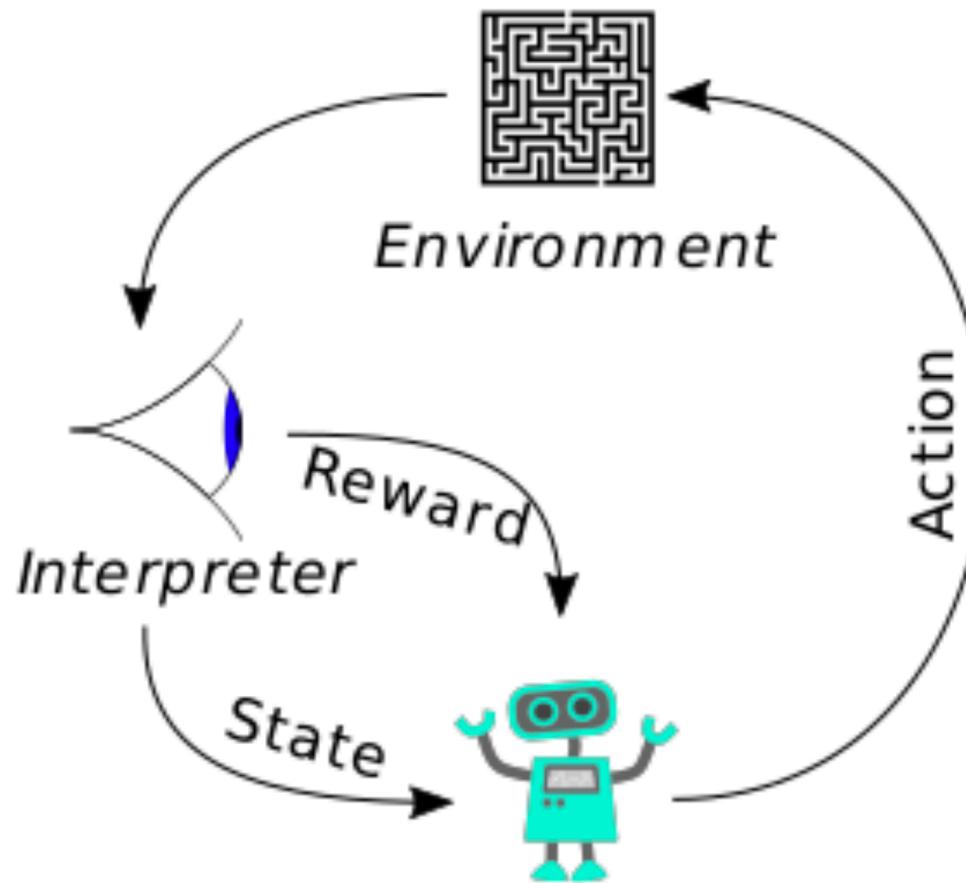
# Topics Covered in This Course

- Classical Planning and Control



# Topics Covered in This Course

- Concepts of Reinforcement Learning



# Topics Covered in This Course

- Deep RL Frameworks

# Playing Atari with Deep Reinforcement Learning

Volodymyr Mnih Kd

## Trust Region Policy Optimization

{vlad, koray, david, a

John Schulman

Sergey Levine

Philipp Moritz

Michael Jordan

Pieter Abbeel

We present the first directly from high-dimensional model is a convolutional neural network whose input is raw sensor data and rewards. We apply our model to the Mountain Car Environment, where we find that it outperforms a human expert on a task that requires planning.

Abs

We describe an iterative policies, with guarantee. By making several theoretically-justified practical algorithm, called Optimization (TRPO).

JOSCHU@EECS.BERKELEY.EDU

# Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor

Tuomas Haarnoja<sup>1</sup> Aurick Zhou<sup>1</sup> Pieter Abbeel<sup>1</sup> Sergey Levine<sup>1</sup>

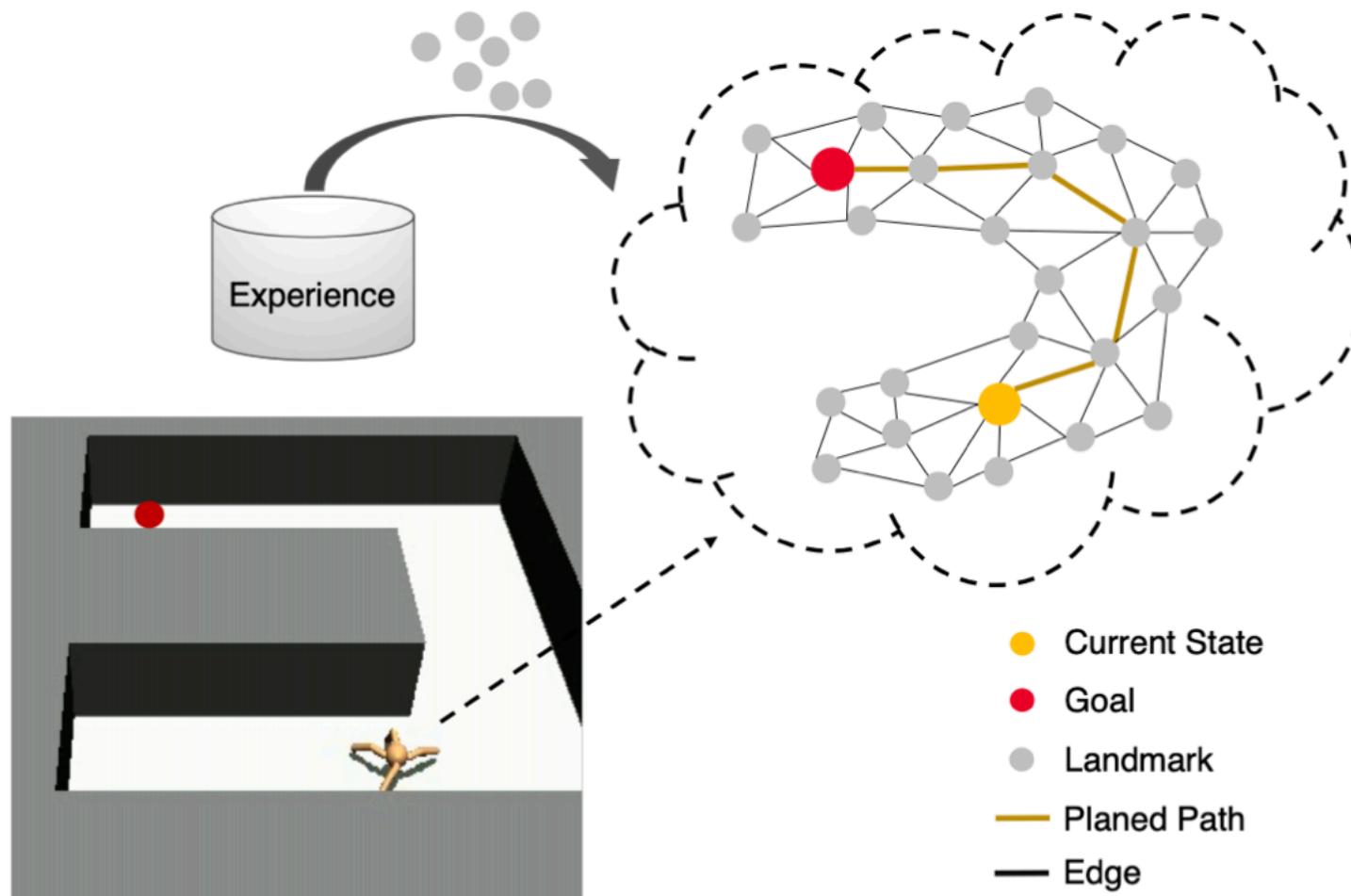
## Abstract

Model-free deep reinforcement learning (RL) algorithms have been demonstrated on a range of challenging decision making and control tasks. However, these methods typically suffer from two major challenges: very high sample complexity and brittle convergence properties, which necessitate meticulous hyperparameter tuning. Both of these challenges severely limit the applicability

of these methods in real-world domains has been hampered by two major challenges. First, model-free deep RL methods are notoriously expensive in terms of their sample complexity. Even relatively simple tasks can require millions of steps of data collection, and complex behaviors with high-dimensional observations might need substantially more. Second, these methods are often brittle with respect to their hyperparameters: learning rates, exploration constants, and other settings must be set carefully for different problem settings to achieve good results. Both of these challenges

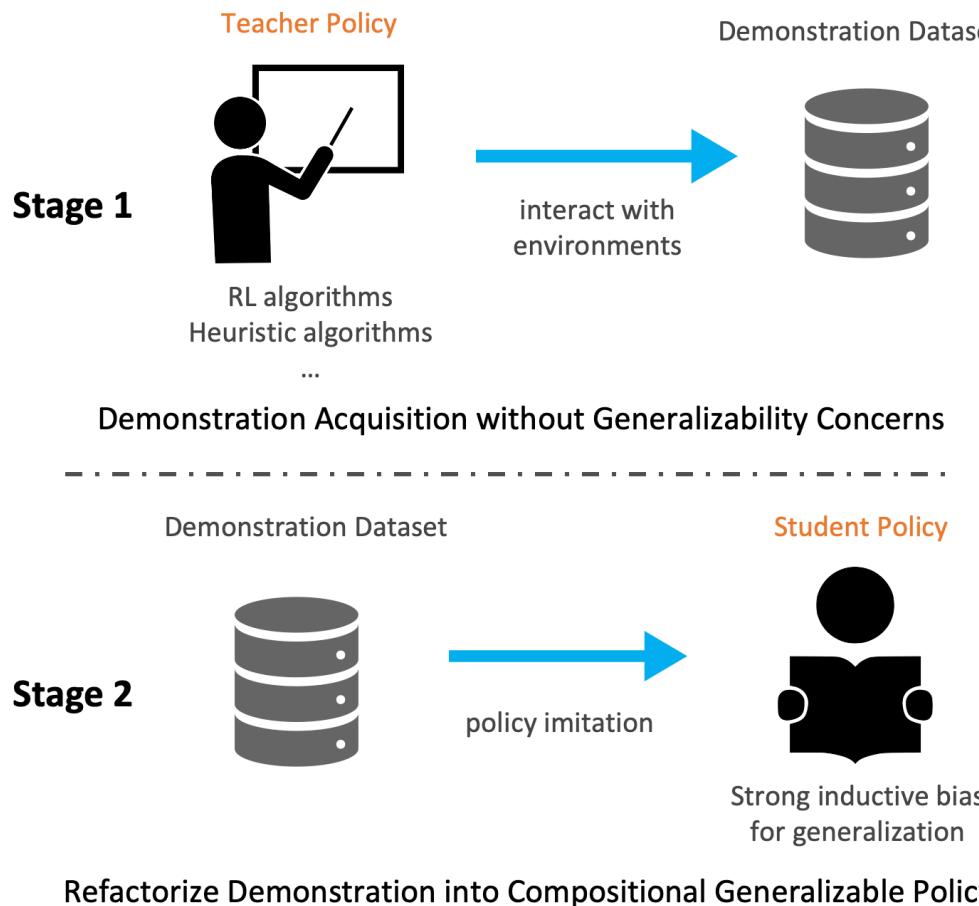
# Topics Covered in This Course

- Hierarchical RL



# Topics Covered in This Course

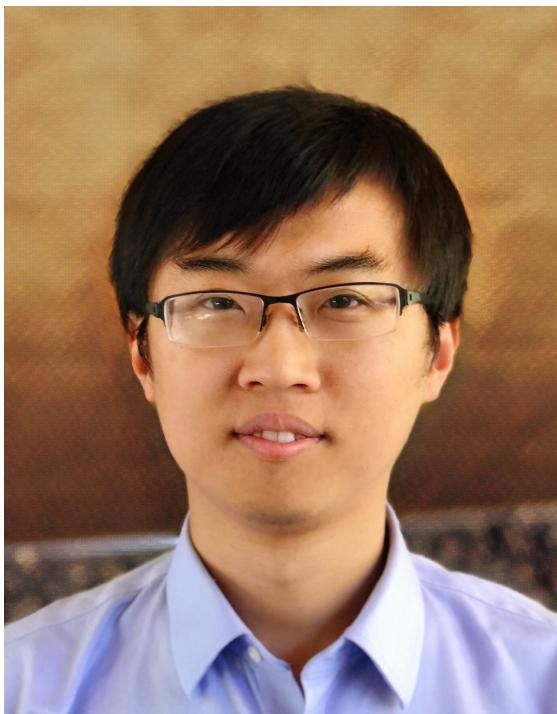
- Generalizability of RL



# Course Logistic

# Instructors

Instructor: Hao Su



TA: Minghua Liu



# Teaching Goal

- Foundational
  - Programming problems ask you to **implement low-level modules from scratch**
- Hands-on
  - **Heavy** programming assignments to exercise what are taught in class

# Pre-requisite: Technique

- **Skilled** in Linear Algebra, Multi-variable Calculus, and Deep Learning
- **Familiar** with Probability and Numerical Methods
- **Strong** programming skills
  - Familiar with Linux Toolchain
  - Familiar with python, numpy, and pytorch
- Course/project experiences in 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: 04/06/2021

# Pre-requisite: Resources

- This course requires deep learning resources (to run reinforcement learning challenges)
- 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 for deep learning

# 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)
- Late policy: 15% grade reduction for each 12 hours late. No acceptance 72 hours after the due time.

# Assignments

- HW1-HW3: practice basic concepts and algorithms; 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-for-robotics/SP21/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

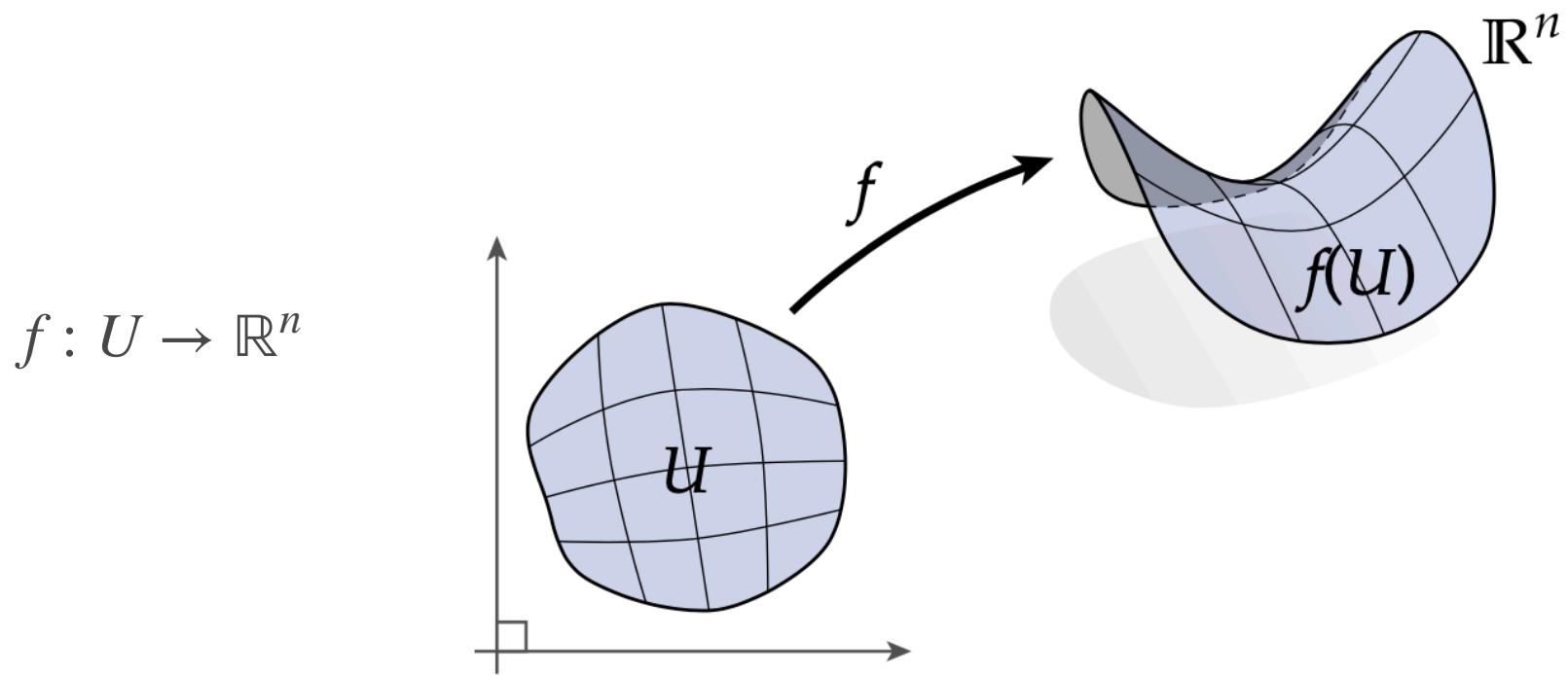
- Time to be determined.
- Please fill in our Piazza survey for information about time zone and etc.

# Questions?

# **Concepts of Differential Geometry**

# Parameterized Surfaces

A **parameterization** is a map from the domain  $U$  into  $\mathbb{R}^n$



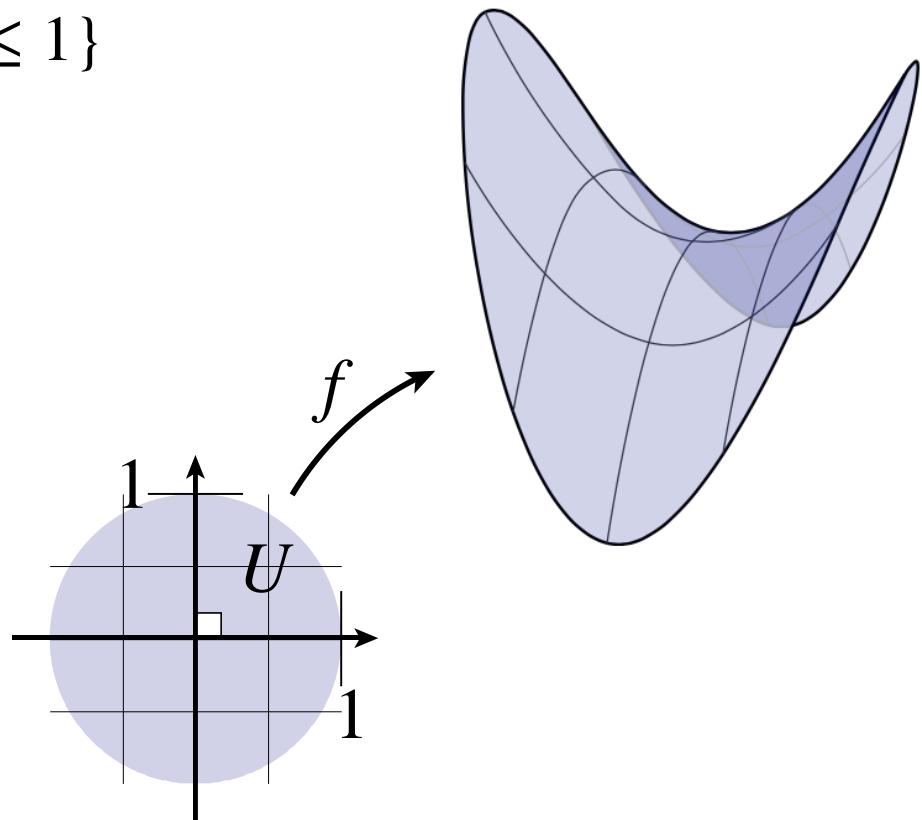
The set of points  $f(U)$  is called the **image** of the parameterization.

# Example

- Example: We can express a *saddle* as a *parameterized surface*:

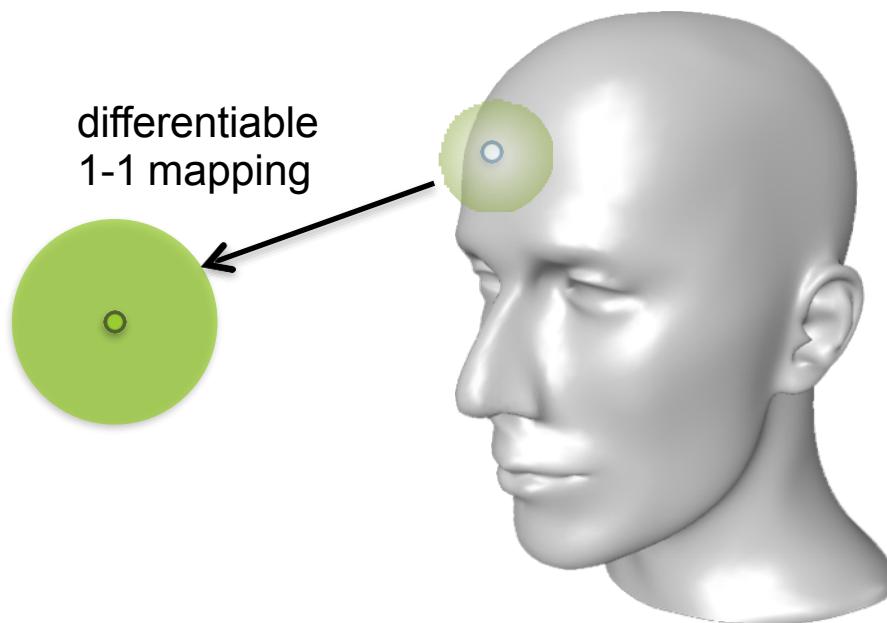
$$U := \{(u, v) \in \mathbb{R}^2 : u^2 + v^2 \leq 1\}$$

$$f(u, v) = [u, v, u^2 - v^2]^T$$



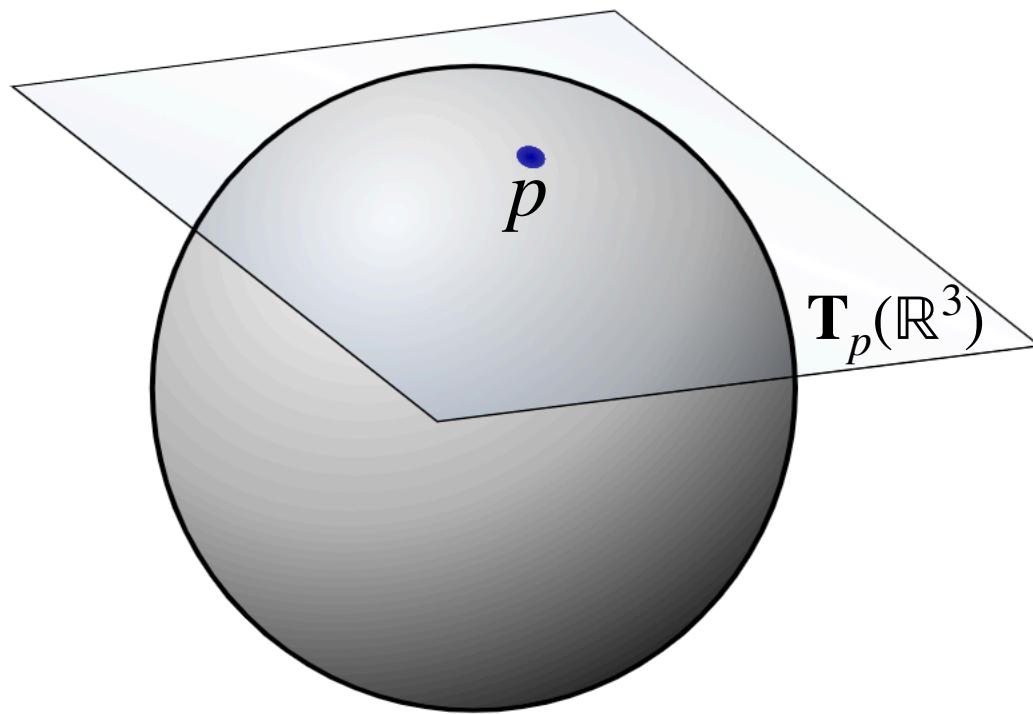
# Manifold

- Things that can be discovered by local observation:  
point + neighborhood



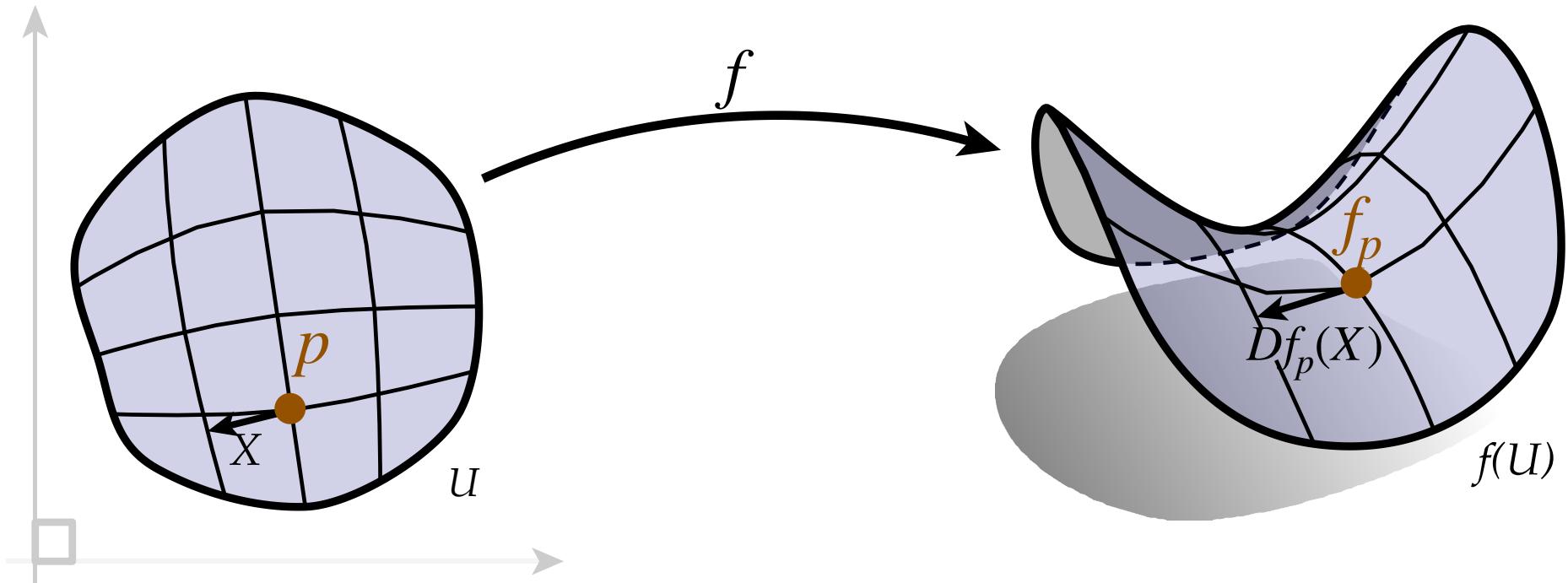
# Tangent Plane

- One can attach to every point  $p$  a tangent plane  $\mathbf{T}_p$
- Intuitively, it contains the possible directions in which one can tangentially pass through  $p$ .

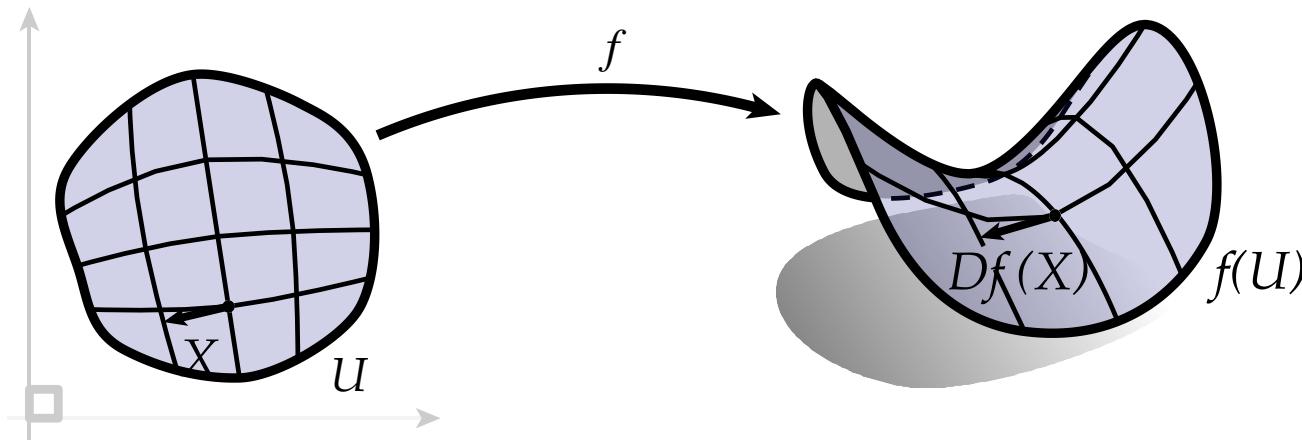


# Differential of a Surface

- Relate the movement of point in the domain and on the image



# Differential of a Surface

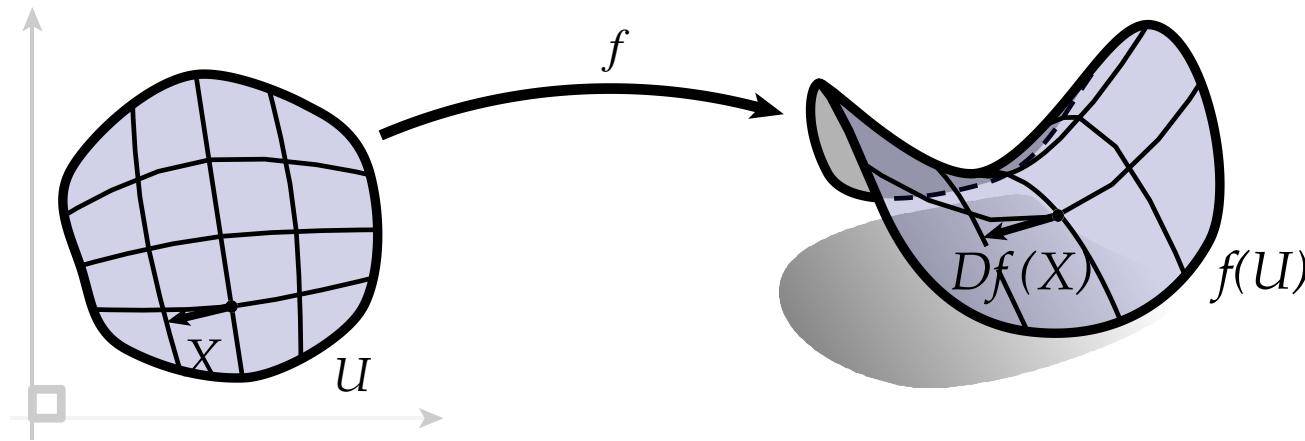


Total differential:  $df = \frac{\partial f}{\partial u} du + \frac{\partial f}{\partial v} dv \rightarrow \Delta f \approx \frac{\partial f}{\partial u} \Delta u + \frac{\partial f}{\partial v} \Delta v$

If point  $p \in \mathbb{R}^2$  moves along vector  $X = [u, v]^T$  by  $\epsilon$ , the movement of  $f_p$  is:

$$\Delta f_p \approx \frac{\partial f}{\partial u}(\epsilon u) + \frac{\partial f}{\partial v}(\epsilon v) = \epsilon \left[ \frac{\partial f}{\partial u}, \frac{\partial f}{\partial v} \right] \begin{bmatrix} u \\ v \end{bmatrix}$$

# Differential of a Surface



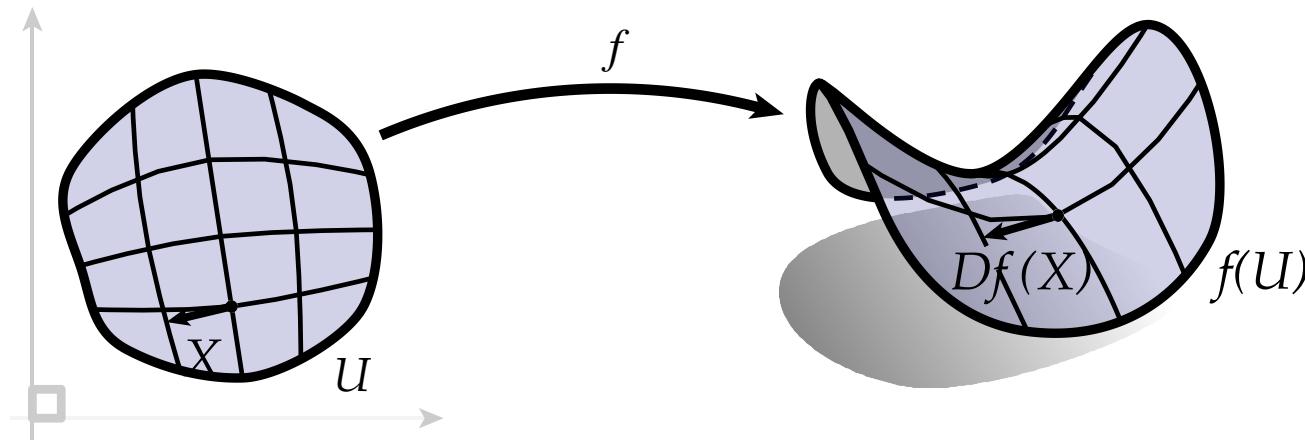
Total differential:  $df = \frac{\partial f}{\partial u} du + \frac{\partial f}{\partial v} dv$

If point  $p \in \mathbb{R}^2$  moves with velocity  $X = [u, v]^T$  by  $\epsilon$ , the movement of  $f_p$  is:

$$\Delta f_p \approx \frac{\partial f}{\partial u}(\epsilon u) + \frac{\partial f}{\partial v}(\epsilon v) = \epsilon \left[ \frac{\partial f}{\partial u}, \frac{\partial f}{\partial v} \right] \begin{bmatrix} u \\ v \end{bmatrix}$$

$$Df_p := \left[ \frac{\partial f}{\partial u}, \frac{\partial f}{\partial v} \right] \in \mathbb{R}^{3 \times 2}$$

# Differential of a Surface



Total differential:  $df = \frac{\partial f}{\partial u} du + \frac{\partial f}{\partial v} dv$

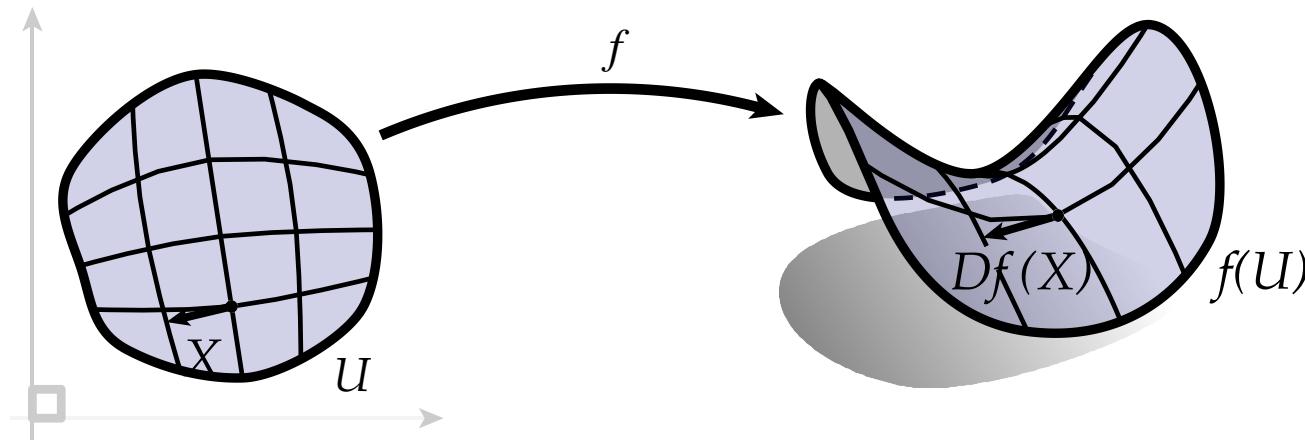
If point  $p \in \mathbb{R}^2$  moves with velocity  $X = [u, v]^T$  by  $\epsilon$ , the movement of  $f_p$  is:

$$\Delta f_p \approx \frac{\partial f}{\partial u}(\epsilon u) + \frac{\partial f}{\partial v}(\epsilon v) = \epsilon \left[ \frac{\partial f}{\partial u}, \frac{\partial f}{\partial v} \right] \begin{bmatrix} u \\ v \end{bmatrix} = \epsilon [Df_p]X$$

$$Df_p := \left[ \frac{\partial f}{\partial u}, \frac{\partial f}{\partial v} \right] \in \mathbb{R}^{3 \times 2}$$

$Df_p$ : differential (Jacobian),  
a linear map.

# Differential of a Surface



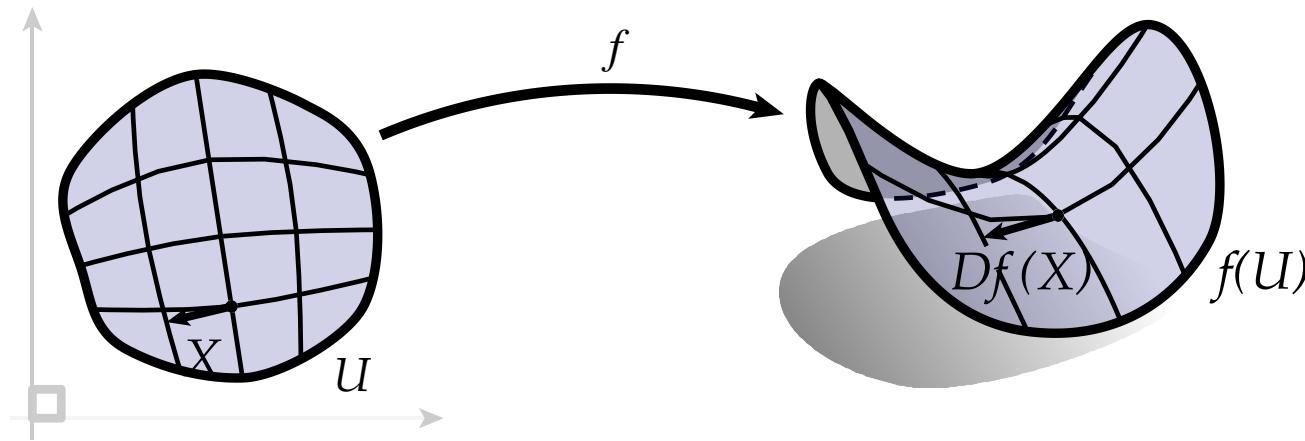
Total differential:  $df = \frac{\partial f}{\partial u} du + \frac{\partial f}{\partial v} dv$

If point  $p \in \mathbb{R}^2$  moves with velocity  $X = [u, v]^T$  by  $\epsilon$ , the movement of  $f_p$  is:

$$\Delta f_p \approx \frac{\partial f}{\partial u}(\epsilon u) + \frac{\partial f}{\partial v}(\epsilon v) = \epsilon \left[ \frac{\partial f}{\partial u}, \frac{\partial f}{\partial v} \right] \begin{bmatrix} u \\ v \end{bmatrix} = \epsilon [Df_p] \boxed{X}$$

$$Df_p := \left[ \frac{\partial f}{\partial u}, \frac{\partial f}{\partial v} \right] \in \mathbb{R}^{3 \times 2} \quad \text{velocity in the 2D domain}$$

# Differential of a Surface



Total differential:  $df = \frac{\partial f}{\partial u} du + \frac{\partial f}{\partial v} dv$

If point  $p \in \mathbb{R}^2$  moves with velocity  $X = [u, v]^T$  by  $\epsilon$ , the movement of  $f_p$  is:

velocity in 3D space

$$\Delta f_p \approx \frac{\partial f}{\partial u}(\epsilon u) + \frac{\partial f}{\partial v}(\epsilon v) = \epsilon \left[ \frac{\partial f}{\partial u}, \frac{\partial f}{\partial v} \right] \begin{bmatrix} u \\ v \end{bmatrix} = \epsilon [Df_p] \boxed{X}$$

$$Df_p := \left[ \frac{\partial f}{\partial u}, \frac{\partial f}{\partial v} \right] \in \mathbb{R}^{3 \times 2}$$

velocity in 2D domain

# Rigid Transformation

# Describe the Pose of Agent and Objects



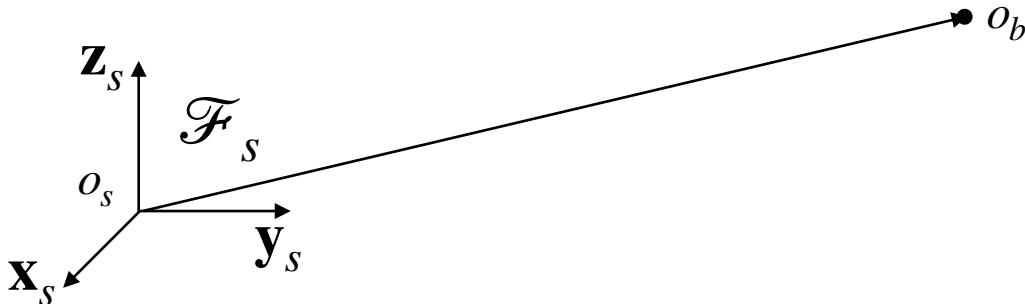
# Describe the Pose of Agent and Objects



Position &  
Orientation



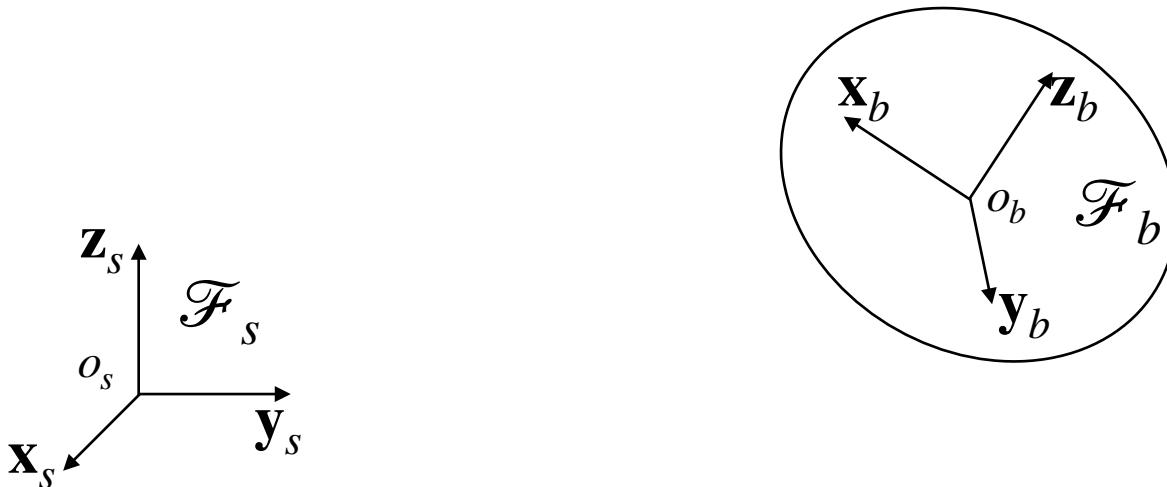
# Notation Convention



- An observer **records** the position of any point in the space **using a frame**  $\mathcal{F}_s$
- We use ordinary letters to denote points (e.g.,  $p$ ), and bold letters to denote **vectors** (e.g.,  $\mathbf{v}$ )
- When **writing equations**, we add a superscript to symbols to denote the recording frame, e.g.,

$$o_b^s = o_s^s + \mathbf{t}_{s \rightarrow b}^s$$

# Rigid Transformation



- There is a rigid object, to which we bind a frame  $\mathcal{F}_b$  (body frame) tightly, so that  $\mathcal{F}_b$  moves along with the object

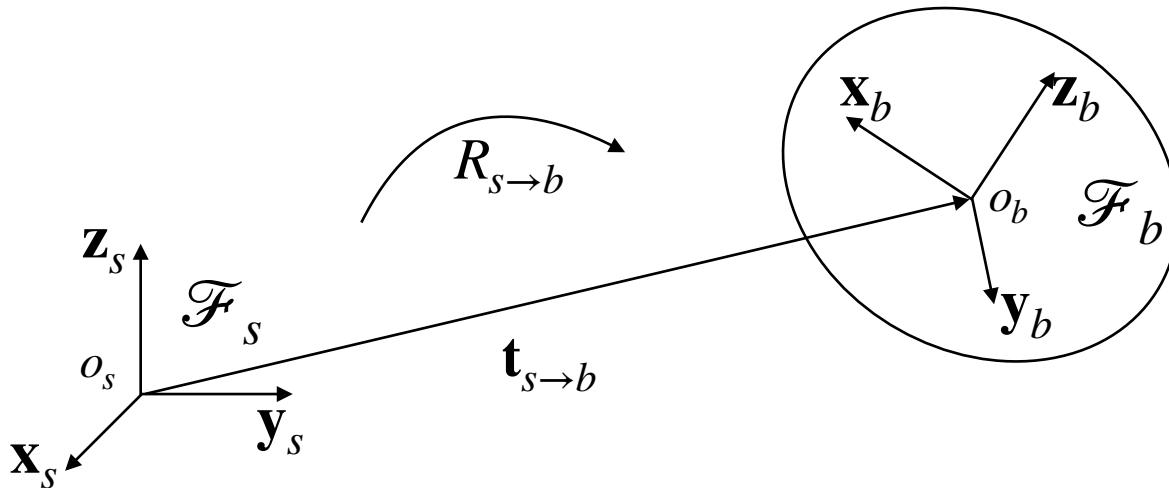
# Rigid Transformation



- When talking about the pose of the *rigid* object, we ask:

How to **transform**  $\mathcal{F}_s$  so that it overlaps with  $\mathcal{F}_b$ ?

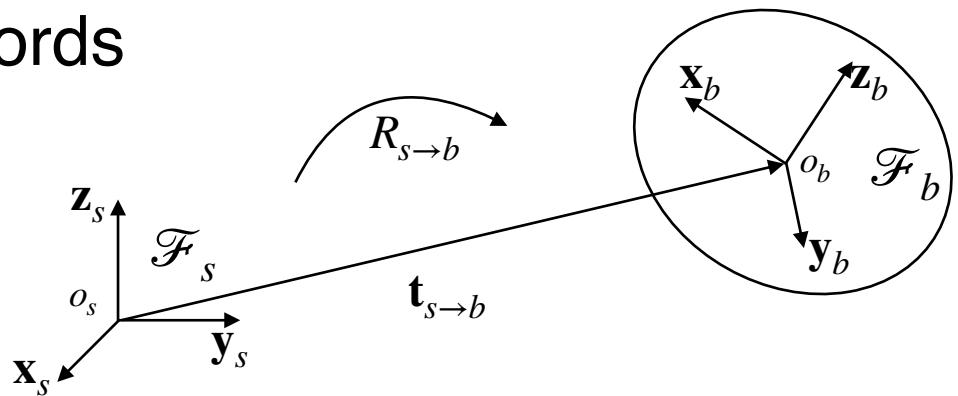
# Rigid Transformation



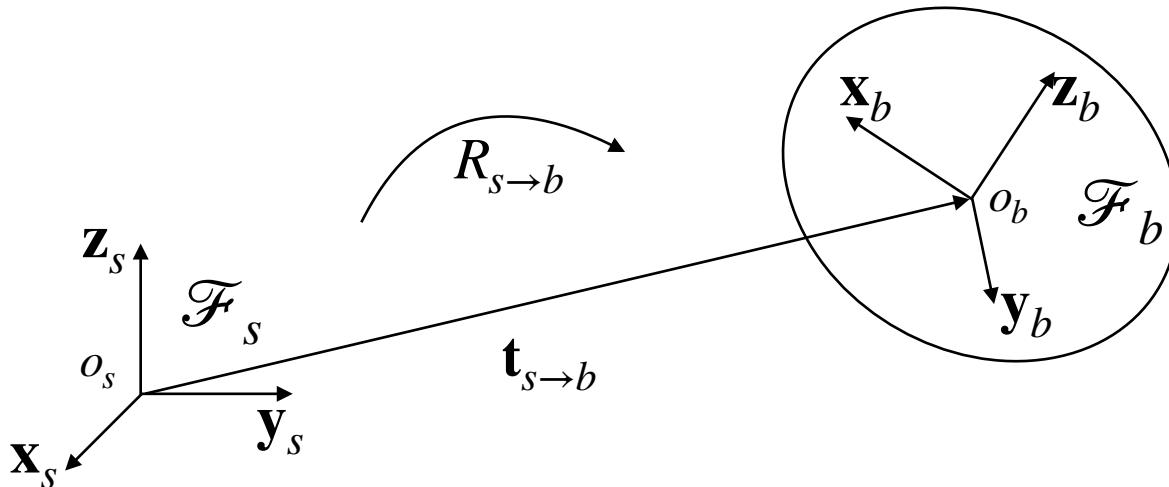
- We first translate  $\mathcal{F}_s$  by  $\mathbf{t}_{s \rightarrow b}$  to align  $o_s$  and  $o_b$
- And then rotate by  $R_{s \rightarrow b}$  to align  $\{\mathbf{x}_i, \mathbf{y}_i, \mathbf{z}_i\}$  ( $i = s$  or  $b$ )

# Rigid Transformation

- Formally,
  - $o_b^s = o_s^s + \mathbf{t}_{s \rightarrow b}^s$
  - $[\mathbf{x}_b^s, \mathbf{y}_b^s, \mathbf{z}_b^s] = R_{s \rightarrow b}^s [\mathbf{x}_s^s, \mathbf{y}_s^s, \mathbf{z}_s^s]$
- Since the observer records everything using  $\mathcal{F}_s$ ,
  - $o_s^s = 0$
  - $[\mathbf{x}_s^s, \mathbf{y}_s^s, \mathbf{z}_s^s] = I_{3 \times 3}$
- Therefore,
  - $\mathbf{t}_{s \rightarrow b}^s = o_b^s$
  - $R_{s \rightarrow b}^s = [\mathbf{x}_b^s, \mathbf{y}_b^s, \mathbf{z}_b^s] \in \mathbb{R}^{3 \times 3}$

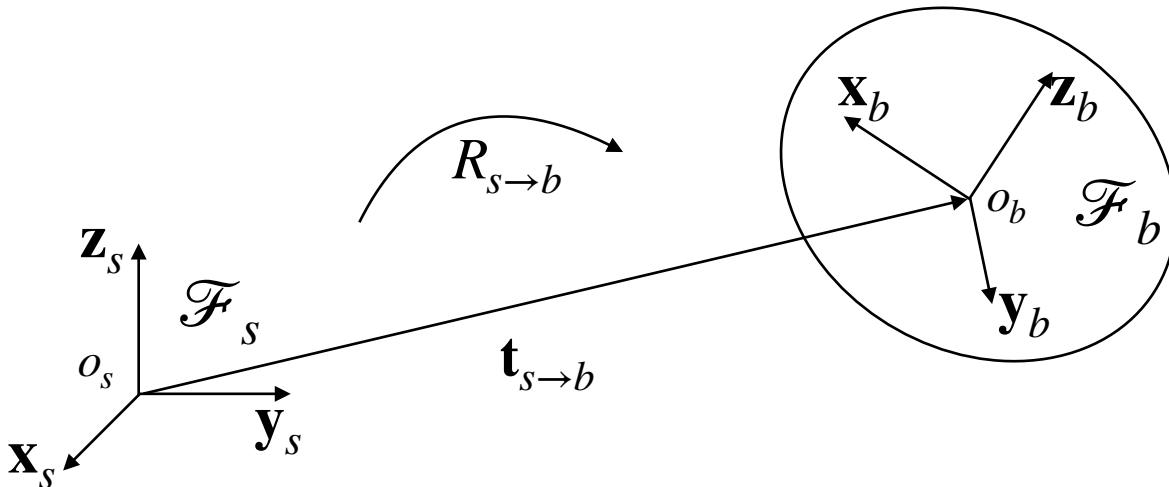


# $(R_{s \rightarrow b}, t_{s \rightarrow b})$ for Coordinate Transformation



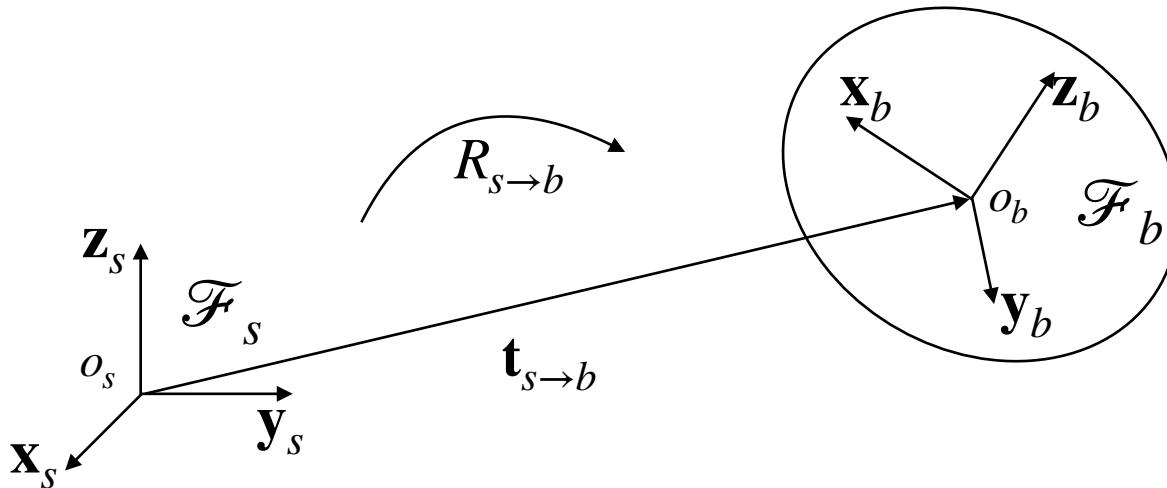
- Assume a second observer that records coordinates by  $\mathcal{F}_b$
- Assume a point  $p$  on the body. Since  $\mathcal{F}_b$  moves along the body, its coordinate recorded in  $\mathcal{F}_b$ , denoted as  $p^b$ , should not change.

# $(R_{s \rightarrow b}, t_{s \rightarrow b})$ for Coordinate Transformation



- Imagine a process:  $\mathcal{F}_b$  moves from  $\mathcal{F}_s$  to the current location. This is how we define  $(R_{s \rightarrow b}^s, t_{s \rightarrow b}^s)$ .
- Since  $p$  moves along  $\mathcal{F}_b$ , it is moved from the initial position,  $p^s = p^b$ , to the current location:  
$$p^s = R_{s \rightarrow b}^s p^b + t_{s \rightarrow b}^s$$

# $(R_{s \rightarrow b}, \mathbf{t}_{s \rightarrow b})$ as a Linear Transformation



- $(R_{s \rightarrow b}, \mathbf{t}_{s \rightarrow b})$  transforms any **point** in the *whole space* by the following equation:

$$x'^s = R_{s \rightarrow b}^s x^s + \mathbf{t}_{s \rightarrow b}^s$$

# $(R_{s \rightarrow b}, t_{s \rightarrow b})$ as a Linear Transformation

- Suppose  $\mathcal{F}_p^s = \{p^s, (\mathbf{x}_p^s, \mathbf{y}_p^s, \mathbf{z}_p^s)\}$  is a frame at an arbitrary point  $p^s$
- **Then, the new origin is:**  $p'^s = ?$

# $(R_{s \rightarrow b}, \mathbf{t}_{s \rightarrow b})$ as a Linear Transformation

- Suppose  $\mathcal{F}_p^s = \{p^s, (\mathbf{x}_p^s, \mathbf{y}_p^s, \mathbf{z}_p^s)\}$  is a frame at an arbitrary point  $p^s$
- **Then, the new origin is:**  $p'^s = R_{s \rightarrow b}^s p^s + \mathbf{t}_{s \rightarrow b}^s$

# $(R_{s \rightarrow b}, \mathbf{t}_{s \rightarrow b})$ as a Linear Transformation

- Suppose  $\mathcal{F}_p^s = \{p^s, (\mathbf{x}_p^s, \mathbf{y}_p^s, \mathbf{z}_p^s)\}$  is a frame at an arbitrary point  $p^s$
- **Then, the new origin is:**  $p'^s = R_{s \rightarrow b}^s p^s + \mathbf{t}_{s \rightarrow b}^s$
- How about the bases vectors of the frame?
  - Assume three curves,  $\gamma_x, \gamma_y, \gamma_z$ , passing  $O_p$  at  $t = 0$  with tangents  $\mathbf{x}_p, \mathbf{y}_p, \mathbf{z}_p$

# $(R_{s \rightarrow b}, \mathbf{t}_{s \rightarrow b})$ as a Linear Transformation

- Suppose  $\mathcal{F}_p^s = \{p^s, (\mathbf{x}_p^s, \mathbf{y}_p^s, \mathbf{z}_p^s)\}$  is a frame at an arbitrary point  $p^s$
- **Then, the new origin is:**  $p'^s = R_{s \rightarrow b}^s p^s + \mathbf{t}_{s \rightarrow b}^s$
- How about the bases vectors of the frame?
  - Assume three curves,  $\gamma_x, \gamma_y, \gamma_z$ , passing  $p^s$  at  $t = 0$  with tangents  $\mathbf{x}_p, \mathbf{y}_p, \mathbf{z}_p$
  - Then, the new tangents after transformation are:
$$\frac{d}{dt} R_{s \rightarrow b}^s \gamma_x^s(0), \frac{d}{dt} R_{s \rightarrow b}^s \gamma_y^s(0), \frac{d}{dt} R_{s \rightarrow b}^s \gamma_z^s(0)$$

# $(R_{s \rightarrow b}, \mathbf{t}_{s \rightarrow b})$ as a Linear Transformation

- Suppose  $\mathcal{F}_p^s = \{p^s, (\mathbf{x}_p^s, \mathbf{y}_p^s, \mathbf{z}_p^s)\}$  is a frame at an arbitrary point  $p^s$
- **Then, the new origin is:**  $p'^s = R_{s \rightarrow b}^s p^s + \mathbf{t}_{s \rightarrow b}^s$
- How about the bases vectors of the frame?
  - Assume three curves,  $\gamma_x, \gamma_y, \gamma_z$ , passing  $p^s$  at  $t = 0$  with tangents  $\mathbf{x}_p, \mathbf{y}_p, \mathbf{z}_p$
  - Then, the new tangents after transformation are:
$$\frac{d}{dt} R_{s \rightarrow b}^s \gamma_x^s(0), \frac{d}{dt} R_{s \rightarrow b}^s \gamma_y^s(0), \frac{d}{dt} R_{s \rightarrow b}^s \gamma_z^s(0)$$
- **So the new frame is:**  $\mathcal{F}_{p'}^s = \{p'^s, R_{s \rightarrow b}^s [\mathbf{x}_p^s, \mathbf{y}_p^s, \mathbf{z}_p^s]\}$