上海科技大学
ShanghaiTech University

CS289: **Mobile Manipulation Fall 2024**

Sören Schwertfeger

ShanghaiTech University

STAR CENTER

Outline

- Mobile Manipulation Intro & Definition
- Course Overview
- Software
- Brief History
- 3D Geometry

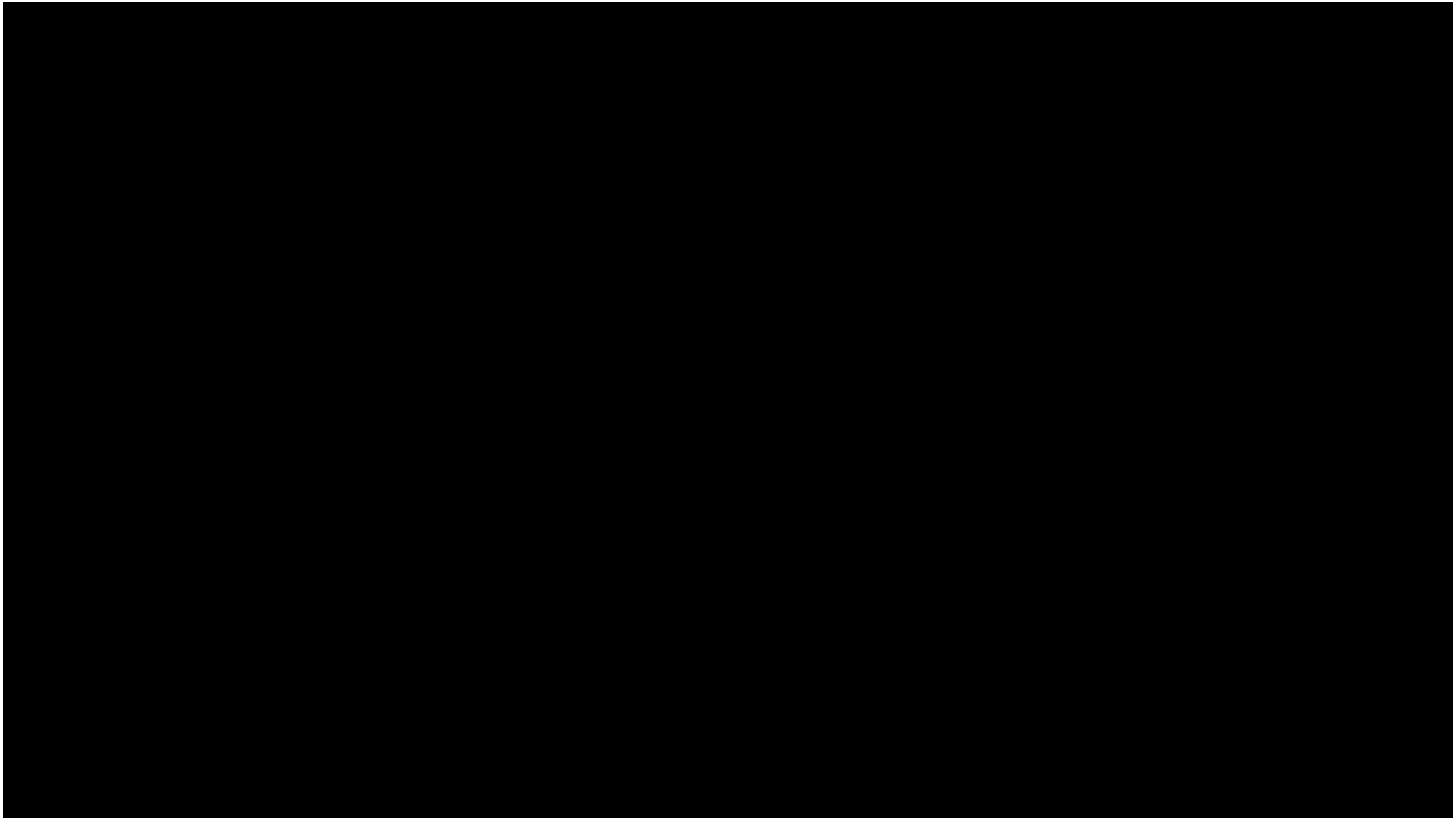
Mobile Manipulation

- A mobile robot …
  - A mobile base that can move in the environment
- that can change something in the environment
  - Typically with a robot arm/ robot manipulator

- Complex problem
  - This course: how to program/ control such a robot!

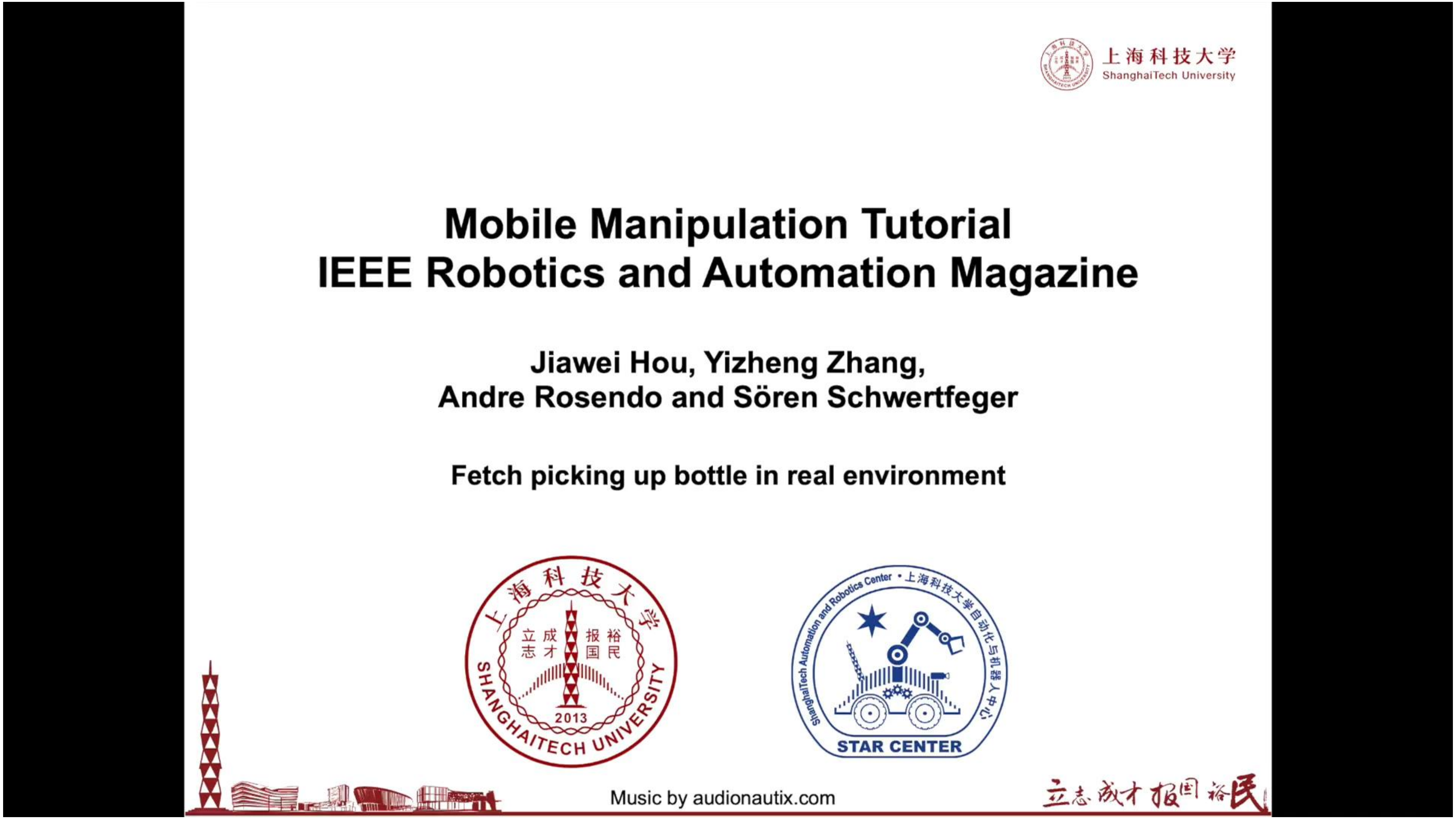
- Many different applications …


ROBLOG
SEVENTH FRAMEWORK PROGRAMME
RobLog Industrial Demonstrator
MS4 Video Trailer
ESB
Qubiqa
BIBA

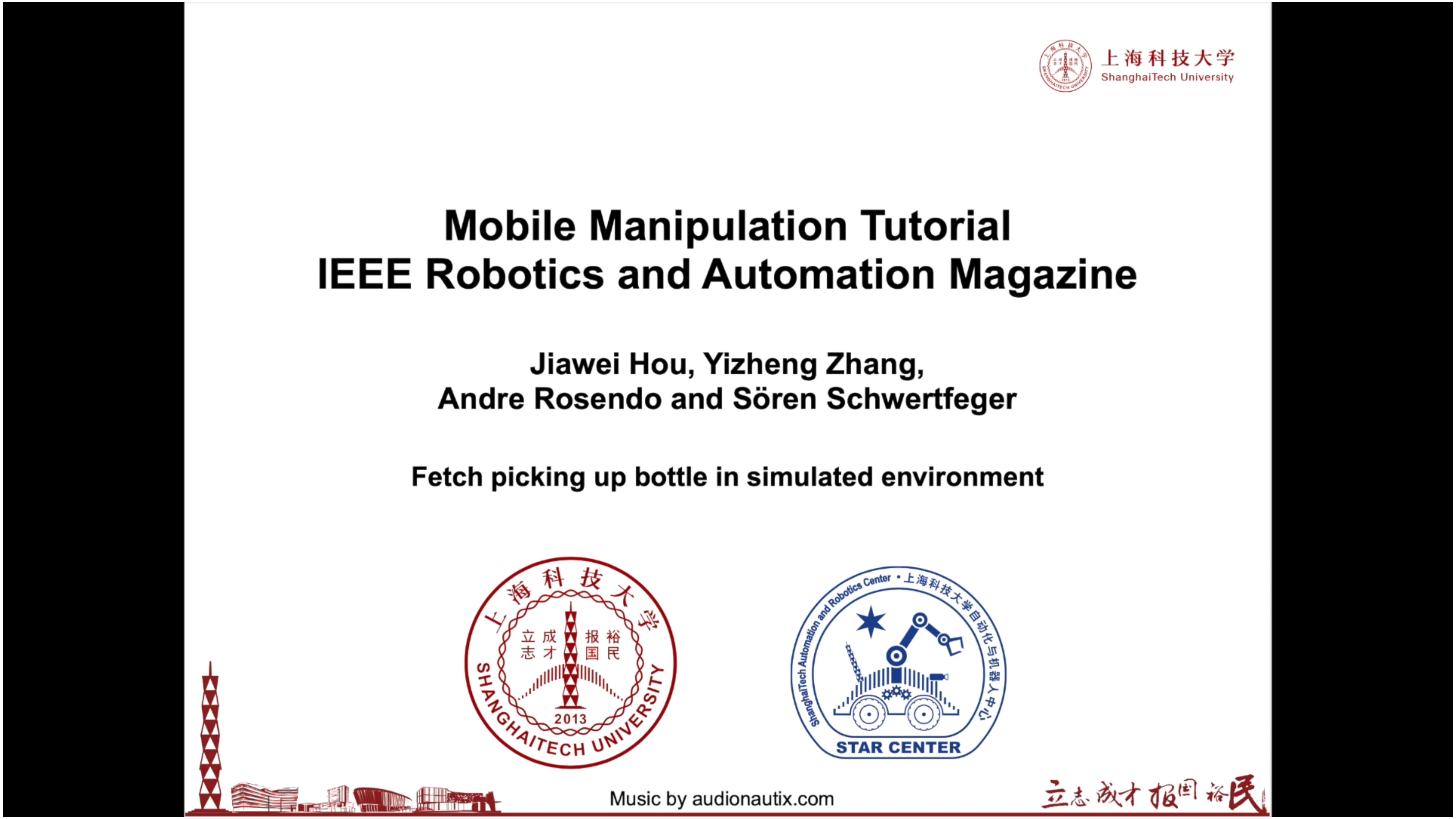



HIROTEC
OTTO


Boston Dynamics

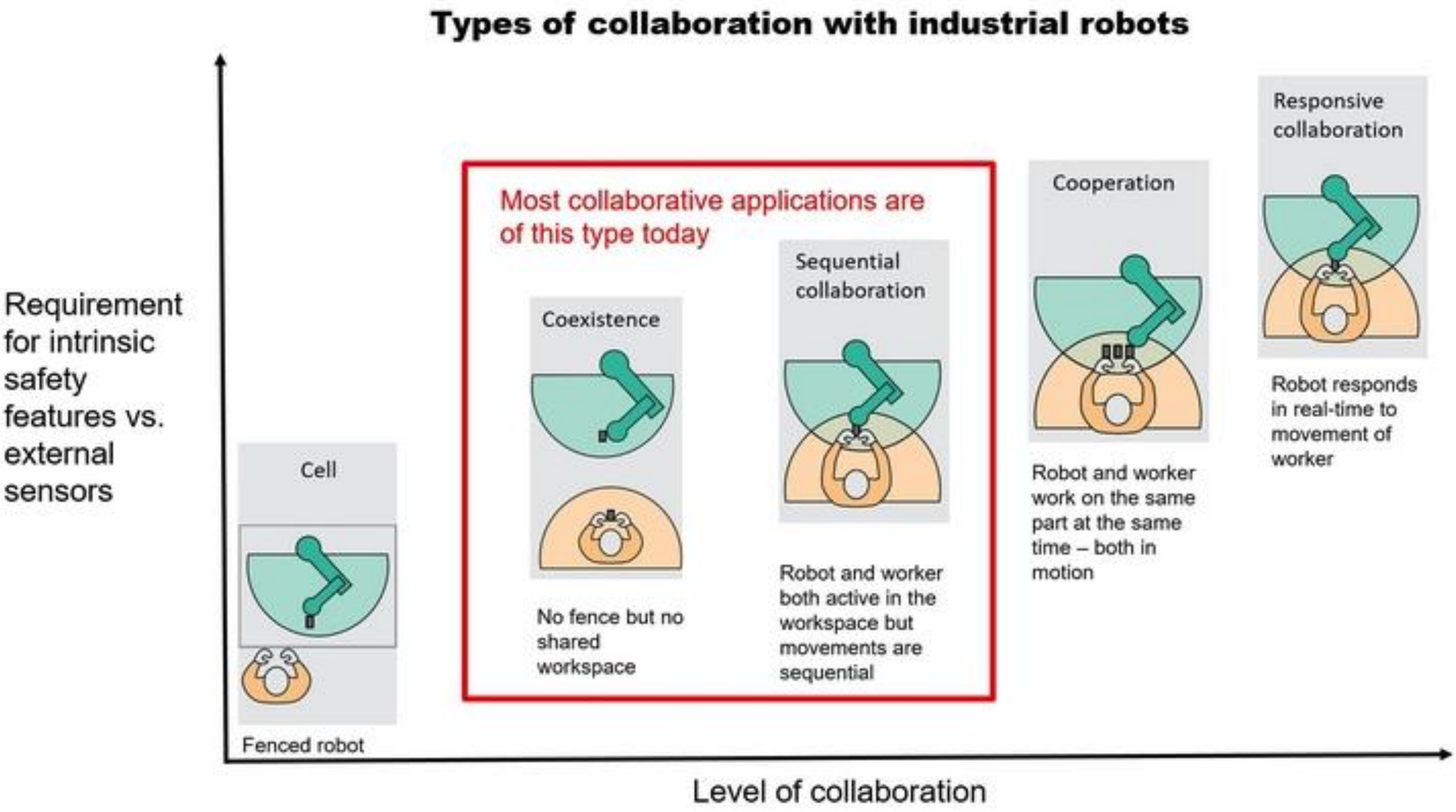

上海科技大学
ShanghaiTech University
Mobile Manipulation Tutorial
IEEE Robotics and Automation Magazine
Jiawei Hou, Yizheng Zhang,
Andre Rosendo and Sören Schwertfeger
Fetch picking up bottle in real environment
SHANGHAITECH UNIVERSITY
2013
STAR CENTER
Music by audionautix.com
立志成才 报国裕民

https://momantu.github.io/



International Organization for Standardization: ISO 8373 Definition

- https://www.iso.org/obp/ui/#iso:std:iso:8373:ed-2:v1:en
- Robot
  - actuated mechanism programmable in two or more axes with a degree of autonomy, moving within its environment, to perform intended tasks
- Industrial Robot
  - automatically controlled,
  - reprogrammable,
  - multipurpose
  - manipulator,
  - programmable in three or more axes
  - which can be either fixed in place or mobile for use in industrial automation applications
- Service Robot
  - robot that performs useful tasks for humans or equipment excluding industrial automation applications



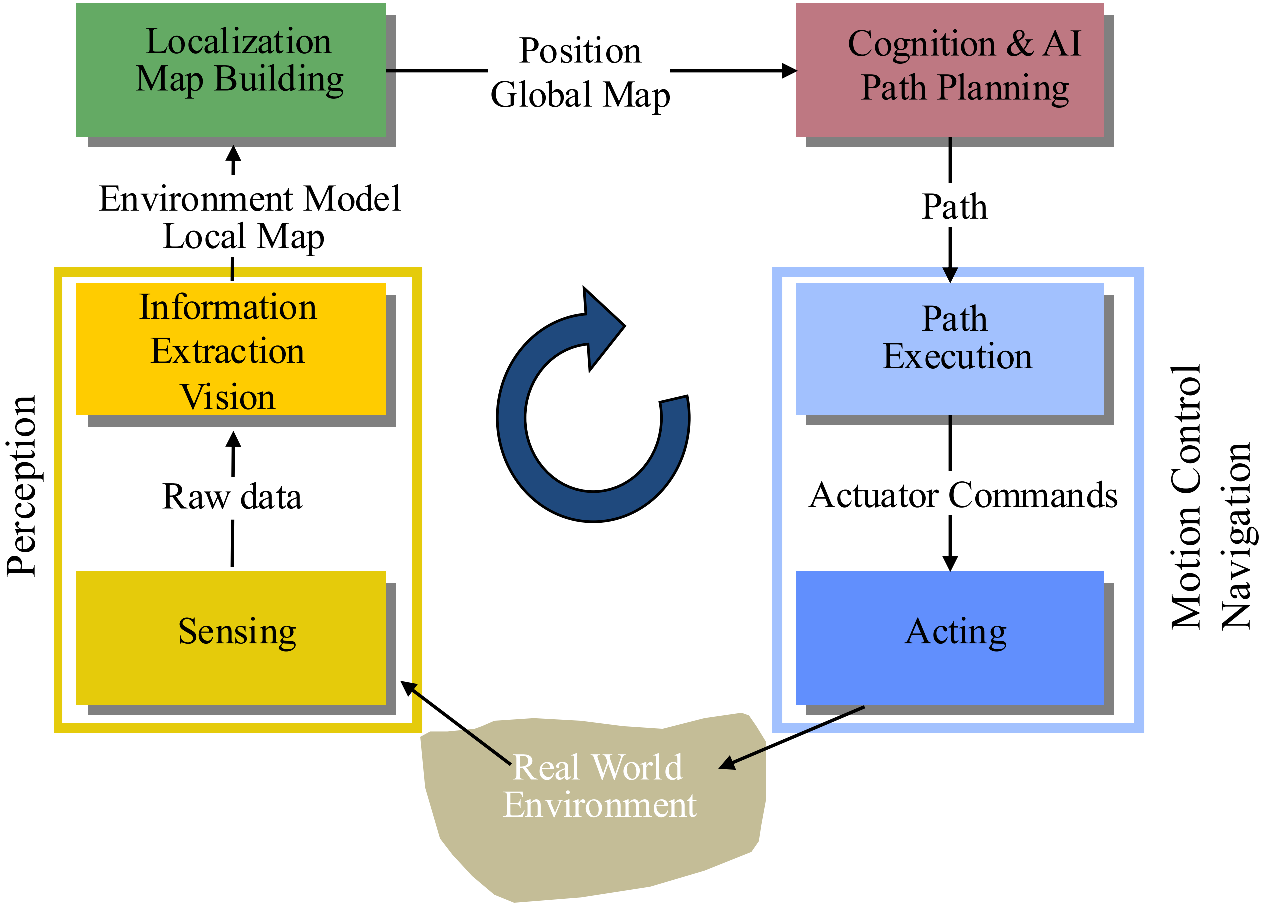
Green area: robot's workspace; yellow area: worker's workspace
Source: IFR (classification), adapted and modified from Bauer et al. (2016).

Mobile Manipulation

- Two problems
  - Mobile Robotics – move Robot from place A to place B
  - Manipulation – use a robotic manipulator to … manipulate something in the environment

- Two problems:
  - Can be solved sequentially:
    - First move the robot base to a goal position
    - Now use the robot arm to do something
    - easier
  - Can be solved together:
    - Plan for the task with moving the base and arm at the same time
    - Harder to do, more flexible solutions possible

General Control Scheme for **Mobile** Robot Systems Emphasis of CS283 Robotics



With material from Roland Siegwart and Davide Scaramuzza, ETH Zurich

Manipulation: Grasp an Object: Steps

1. Startup robot and sensors
2. Detect object & its pose
3. Select grasping points on the object
4. Scan the scene and environment (for collision checking later)
5. Use IK to check if grasping point can be reached – checks for collisions – may try thousands of possibilities (before concluding that there is always a collision)
6. Use motion planning to plan from current pose to goal pose: Lots of collision checks! Might realize that it is impossible after a long time
7. Execute that trajectory: Check if we reached the intermediate pose (within the time constraint) and command the next
8. Controller: take dynamics into account to move to the next intermediate pose
9. Once goal is reached close fingers.
10. Check if object is in fingers
11. Add the object to the collision description of the robot
12. Plan the path to the goal pose…

ADMINISTRIVIA

Teaching Plan

- Lectures
- Homework
  - Presentation about robotics paper (related to your project)
- Final Exam
- Project…

Project

- 1 credit point!
- Work in groups, min 2 students, max 3 students – maybe 4 students.

- Some topics will be proposed later…
  - You can also do your own topic, but only after approval of Prof. Schwertfeger
    - Prepare a short, written proposal till next Tuesday!
  - Choose/ suggest a topic that is in line with your graduate research!
  - We are flexible w.r.t. the topics – as long as they are involved with manipulation or complex mobile robotics.

- One graduate student from my group will co-supervise your project
- Weekly project meetings!

- Oral ”exams” to evaluate the contributions of each member
- No work on project => bad grade of fail

Grading

- Grading scheme is not 100% fixed
- Approximately:
  - Lecture: 60%
    - Homework: 30%
      - One HW will be to present a paper…
    - Final: 30%

  - Project: 40%
    - Project Proposal: 5%
    - Weekly project meetings: 10%
    - Final Report: 10%
    - Final Demo: 10%
    - Final Webpage: 5%

Getting Help

- Piazza:
  - For discussions and announcements
  - https://piazza.com/shanghaitech.edu.cn/summer2024/cs289
  - Ask questions regarding your reading assignments and homework
  - You are not allowed to give the solutions – just guidance
- Ask questions during the lecture!
- Upon request we can organize a tutorial session
- Ask general questions in our wechat group:
- Only if everything else fails: write e-mails
- Office Hours Prof. Schwertfeger: Tuesday afternoon
- Office Hours TA: make appointment via email

Group: Mobile Manipulation 2024

TA: Jiajie Zhang
zhangjj2023@shanghaitech.edu.cn

Policy on Plagiarism

- The homework are individual tasks!
- You may discuss the ideas and algorithms of homework with others but:
  - At no time should you read the source code or possess the source files of any other person, including people outside this course.
  - We will **detect plagiarism** using automated tools and will **prosecute** all violations to the fullest extent of the university regulations, including failing this course, academic probation, and expulsion from the university.

- Homework, project submissions, etc. will be submitted through git – using gitlab. We will create accounts on our gitlab for you.

Mobile Manipulation

- Topic: Mobile Manipulation Robots and how to program them

- **Literature:**
- P. Corke, Robotics, Vision & Control.
  3rd edition!
  “Fundamental Algorithms in Python” !

  Available as open access:
  https://link.springer.com/book/10.1007/978-3-031-06469-2

Material

- Webpage
  - https://robotics.shanghaitech.edu.cn/teaching/moma2024
  - Slides will be available on the webpage
- Piazza
  - https://piazza.com/shanghaitech.edu.cn/summer2024/cs289
  - Where to find us:
    Office: SIST 1D 201.A
    Lab: SIST 1D 203
- E-Mail:
  - soerensch@ShanghaiTech.edu.cn

Schedule

- Rough plan: Lectures: October & November; Project: December & January
- May change – take a look at webpage for most recent version!

Chapter	Teaching Contents	Week
Introduction	History; Definition; General Control Scheme; Programming	1
Kinematics	Differential kinematics, trajectories, optimization	2
Perception	Perception for manipulation, ICP, DL	3
Grasping	Antipodal Grasping	4
Behaviors	State machines, Behavior trees	4
Motion Planning	Optimization-based, sampling-based, global optimization	5
Control	Force Control, Manipulator Control	6
Visual Servoing	Visual Servoing	7
Reinforcement Learning	Reinforcement Learning	7-8
Task & Motion Planning	Task & Motion Planning	8
Collaboration	Robot-Robot collaboration; Human robot collaboration	9
Final Exam		10
Project	Proposal; Final Report; Demo; Webpage	8-14

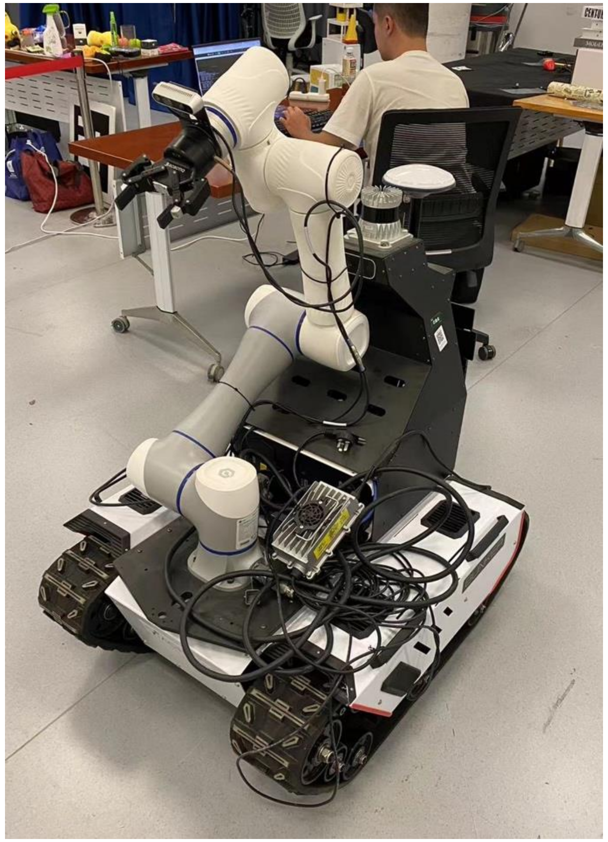
ToDo:

- Join the lecture on piazza
- Organize access to the text book
- For your project you quite likely need ROS – ROS 1 or ROS 2, depending on robot…
  - => review ROS …

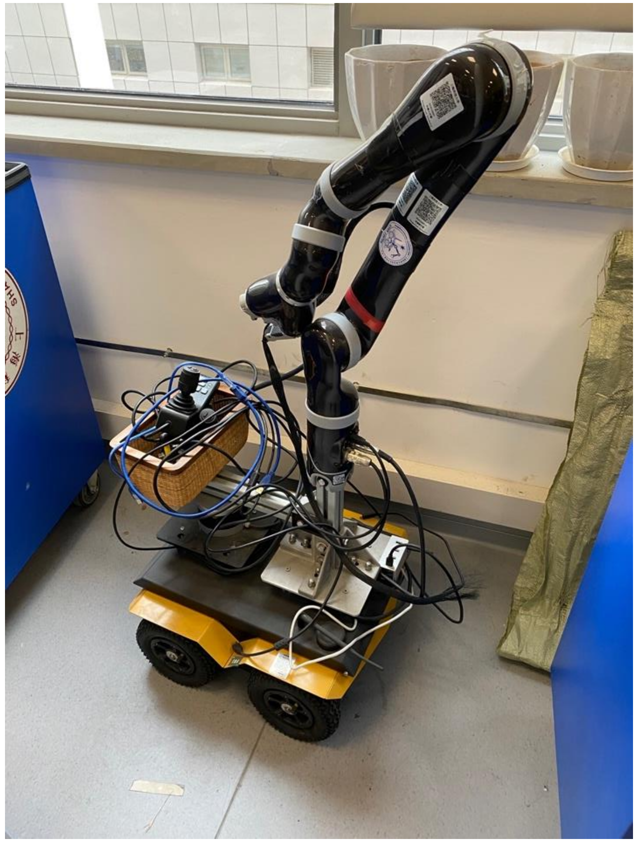
Think about projects – with real robots…

2x Fetch Robots

Think about projects – with real outdoor robots…



Agile X Bunker Tracked Mobile Robot with Dobot Arm

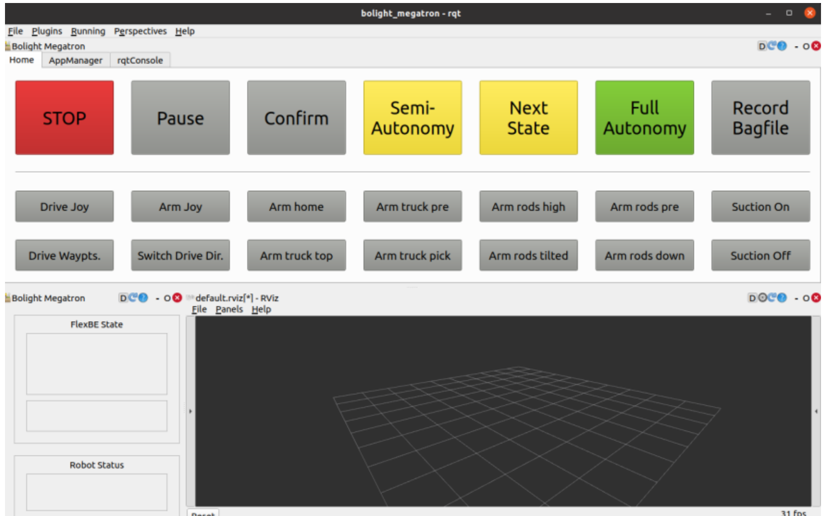


Jackal Mobile Platform with Kinova Robot Arm

Husky Mobile Platform with AUBO Robot Arm

Some of last year's projects

- Kiwi fruit picking
- Human Robot Interaction with Fetch
- MoManTu testing
- Fetch 3D Scanning
- Face Manipulation with Sophia Robot
- Use water brush to paint characters on the ground
- Tabel-balancing
- Industrial project…



Quick Introduction Round!

- Name
- Graduate or Undergraduate student
- Did you take CS283 Robotics or CS183 Introduction to Robotics or any other Robotics course?
- CS student?
- Do you know ROS?
- Graduate Instructor
- Research Topic

SOFTWARE

Robot Software: Tasks/ Modules/ Programs (ROS: node)

Support

- Communication with Micro controller
- Sensor drivers
- Networking
  - With other PCs, other Robots, Operators
- Data storage
  - Store all data for offline processing and simulation and testing
- Monitoring/ Watchdog

Robotics

- Control
- Navigation
- Planning
- Sensor data processing
  - e.g. Stereo processing, Image rectification
- Mapping
- Localization
- Object Recognition
- Mission Execution
- Task specific computing, e.g.:
  - View planning, Victim search, Planning for robot arm, …

Software Design

- Modularization:
  - Keep different software components separated
  - ☺ Keep complexity low
  - ☺ Easily exchange a component (with a different, better algorithm)
  - ☺ Easily exchange multiple components with simulation
  - ☺ Easily exchange dada from components with replay from hard disk instead of live sensor data
  - ☺ Multiple programming teams working on different components easier
  - Need: Clean definition of interfaces or exchange messages!
  - Allows: Multi-Process (vs. Single-Process, Multi-Thread) robot software system
  - Allows: Distributing computation over multiple computers

Programming techniques for you to review!

- Process vs. Thread
- C++ Object Orientation
- Constant Variables
  - const-correctness
- C++ Templates
- Shared Pointer

- Objective:
  - Prerequisites for understanding ROS.
  - Understand how we can efficiently retrieve and transfer data in ROS.

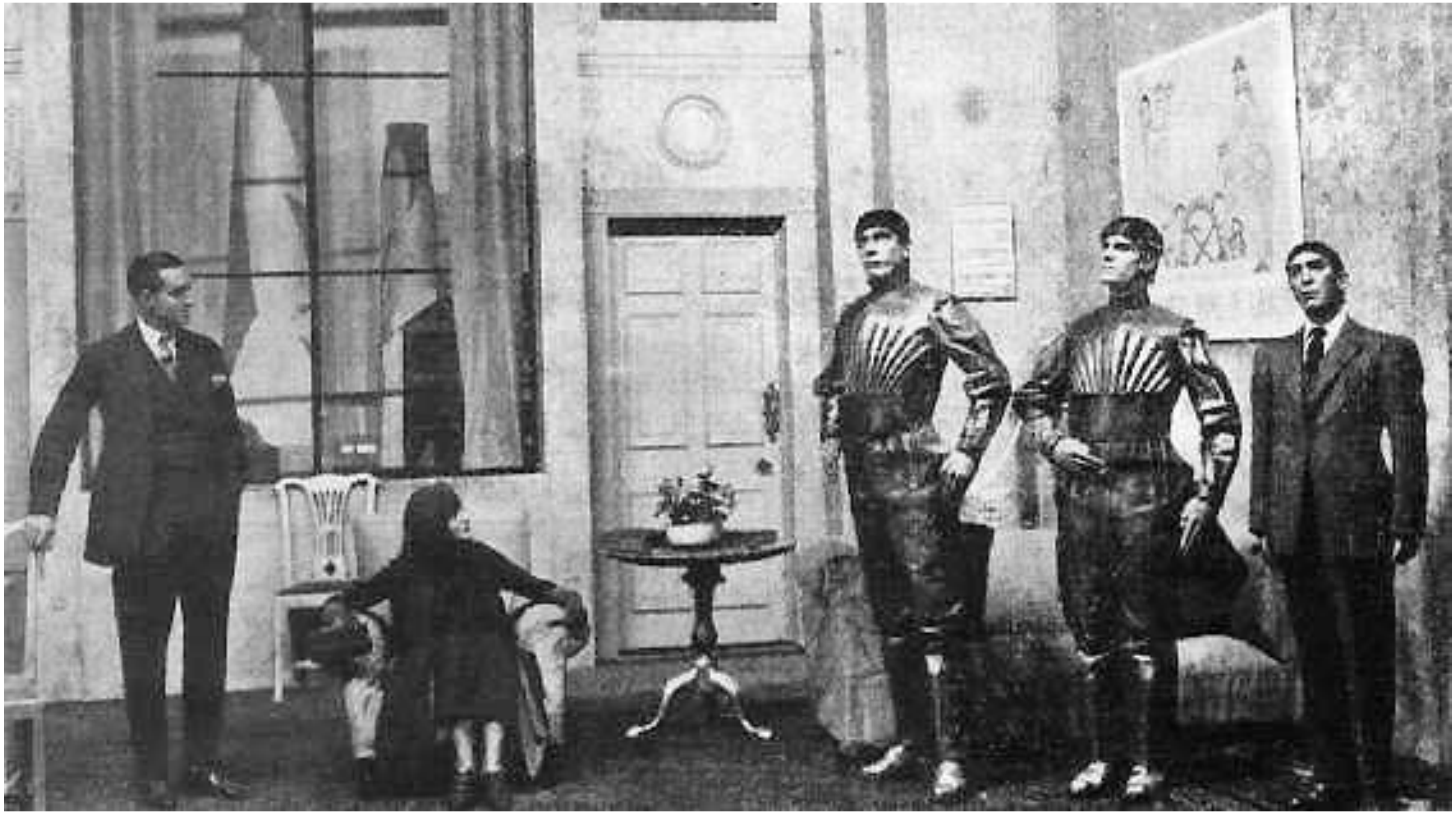
  - We may use ROS 1 and ROS 2
  - We may use C++ and Python

  - Homework will be in Python & ROS 2!

BRIEF HISTORY

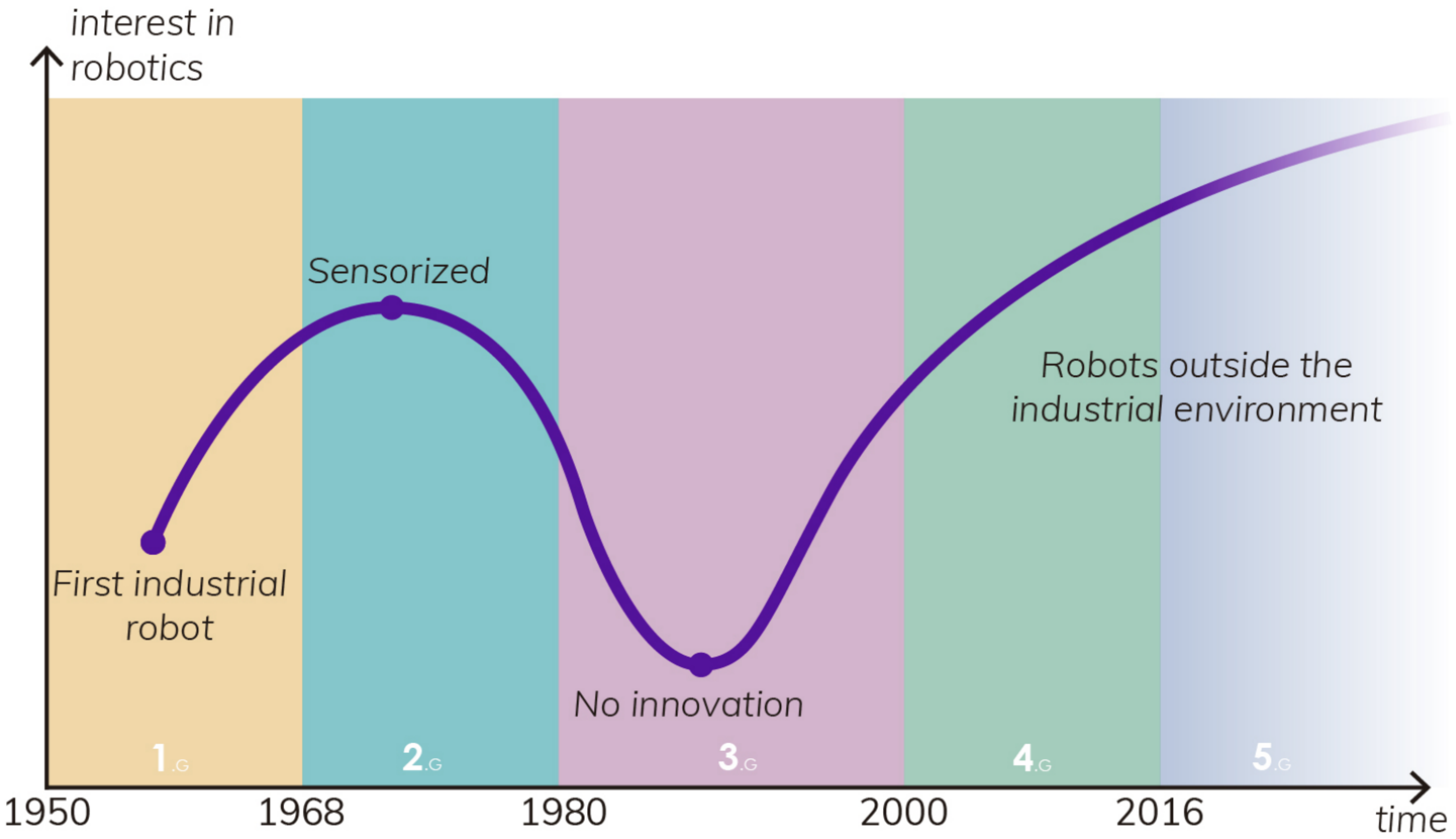
Brief History

Robota “forced labor”: Czech, Karel Čapek R.U.R. 'Rossum's Universal Robots' (1920).



History

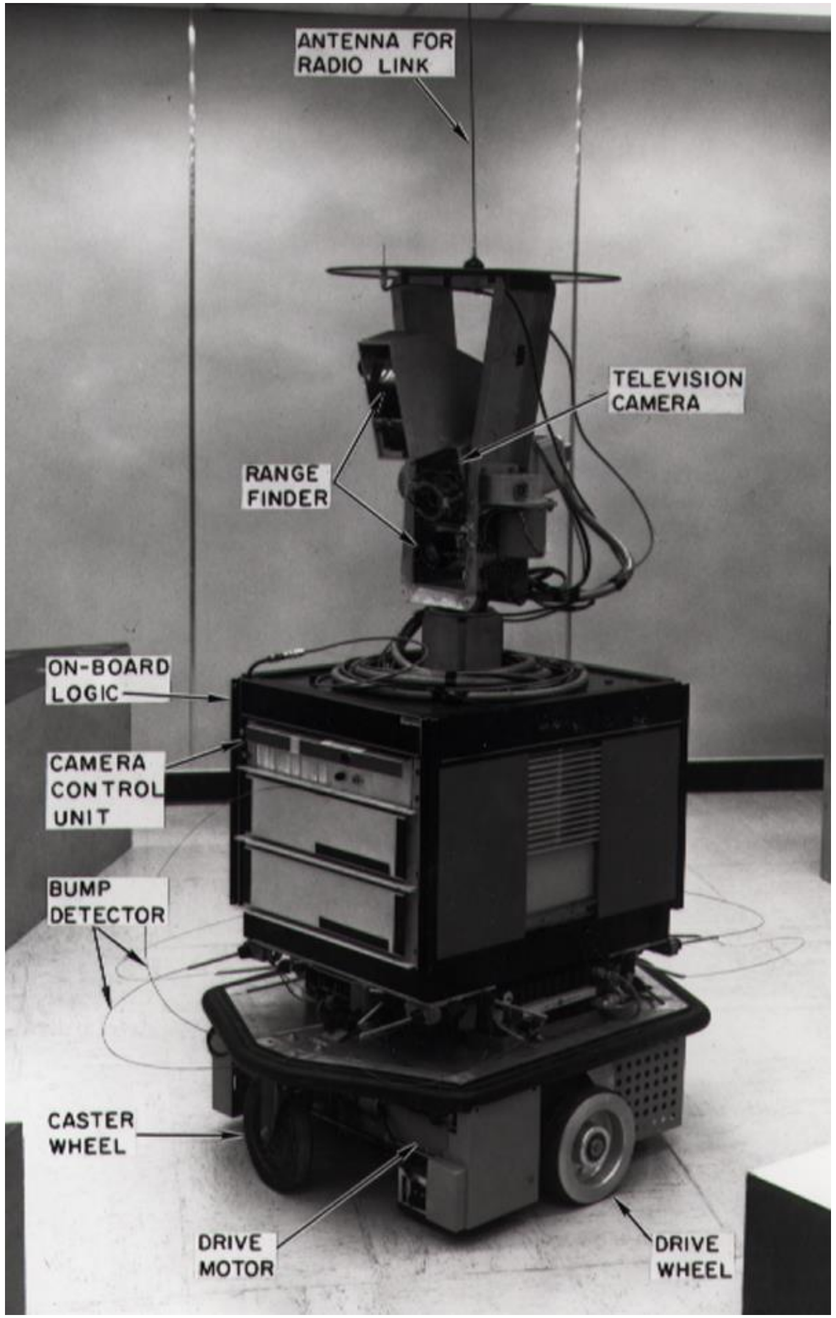
- First electronic autonomous robots 1949 in England (William Grey Walter, Burden Neurological Institute at Bristol)
  - three-wheeled robots: drive to recharging station using light source (phototaxis)
- Turing Test: 1950 (British mathematician Alan Turing)
- Unimate: 1961 lift hot pieces of metal from a die casting machine and stack them. First industry robot. Inventor: George Devol, user: General Motors.
- Lunokhod 1: 1970, lunar vehicle on the moon (Soviet Union)
- Shakey the robot: 1970
- 1989: Chess programs from Carnegie Mellon University defeat chess masters
- Aibo: 1999 Sony Robot Dog
- ASIMO: 2000 Honda (humanoid robot)



https://arxiv.org/pdf/1704.08617.pdf

Shakey the robot (1970)

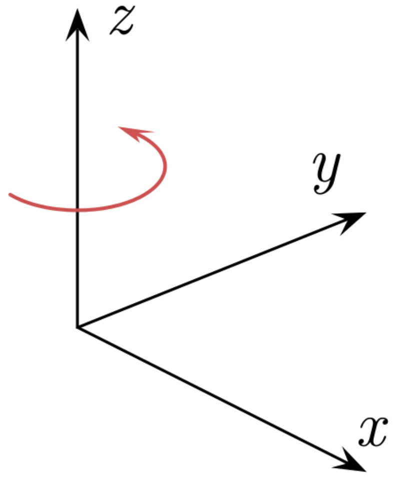
- First general-purpose mobile robot to be able to reason about its own actions
- Advanced hardware:
  - radio communication
  - sonar range finders
  - television camera
  - on-board processors
  - bump detectors
- Advanced software:
  - Sensing and reasoning
- Very big impact
- https://robotics.shanghaitech.edu.cn/static/videos/Shakey.mp4

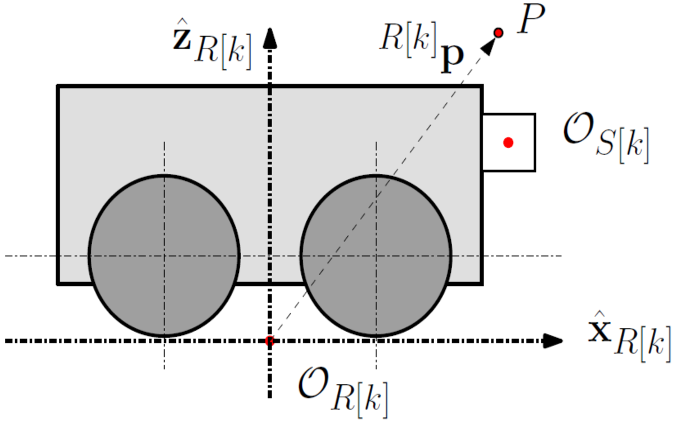


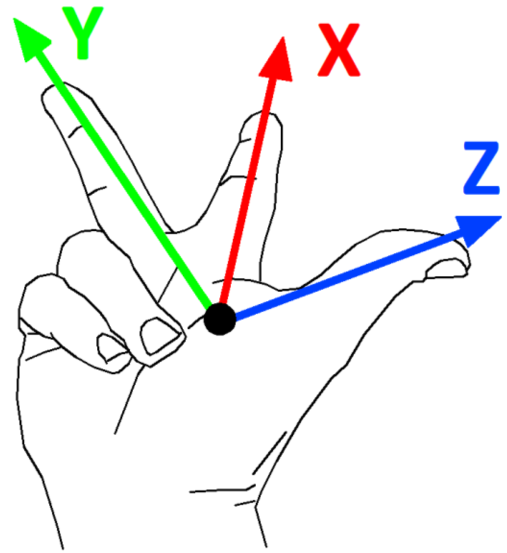
COORDINATE SYSTEM

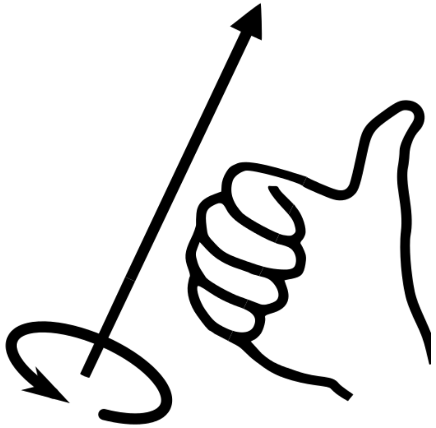
Right Hand Coordinate System

- Standard in Robotics
- Positive rotation around X is anti-clockwise
- Right-hand rule mnemonic:
  - Thumb: z-axis
  - Index finger: x-axis
  - Second finger: y-axis
  - Rotation: Thumb = rotation axis, positive rotation in finger direction
- Robot Coordinate System:
  - X front
  - Z up (Underwater: Z down)
  - Y ???



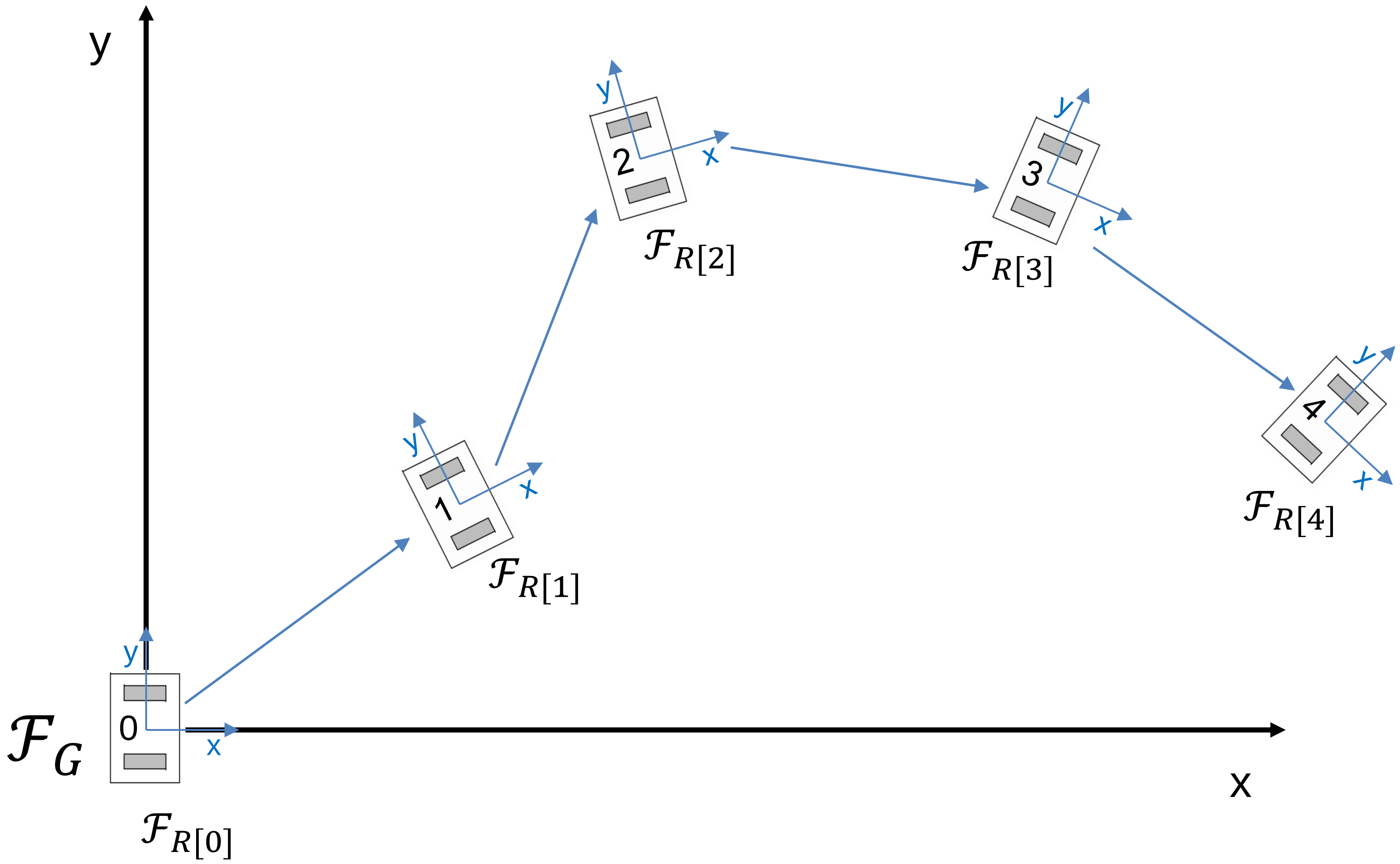






Right Hand Rule http://en.wikipedia.org/wiki/Right-hand_rule

Odometry



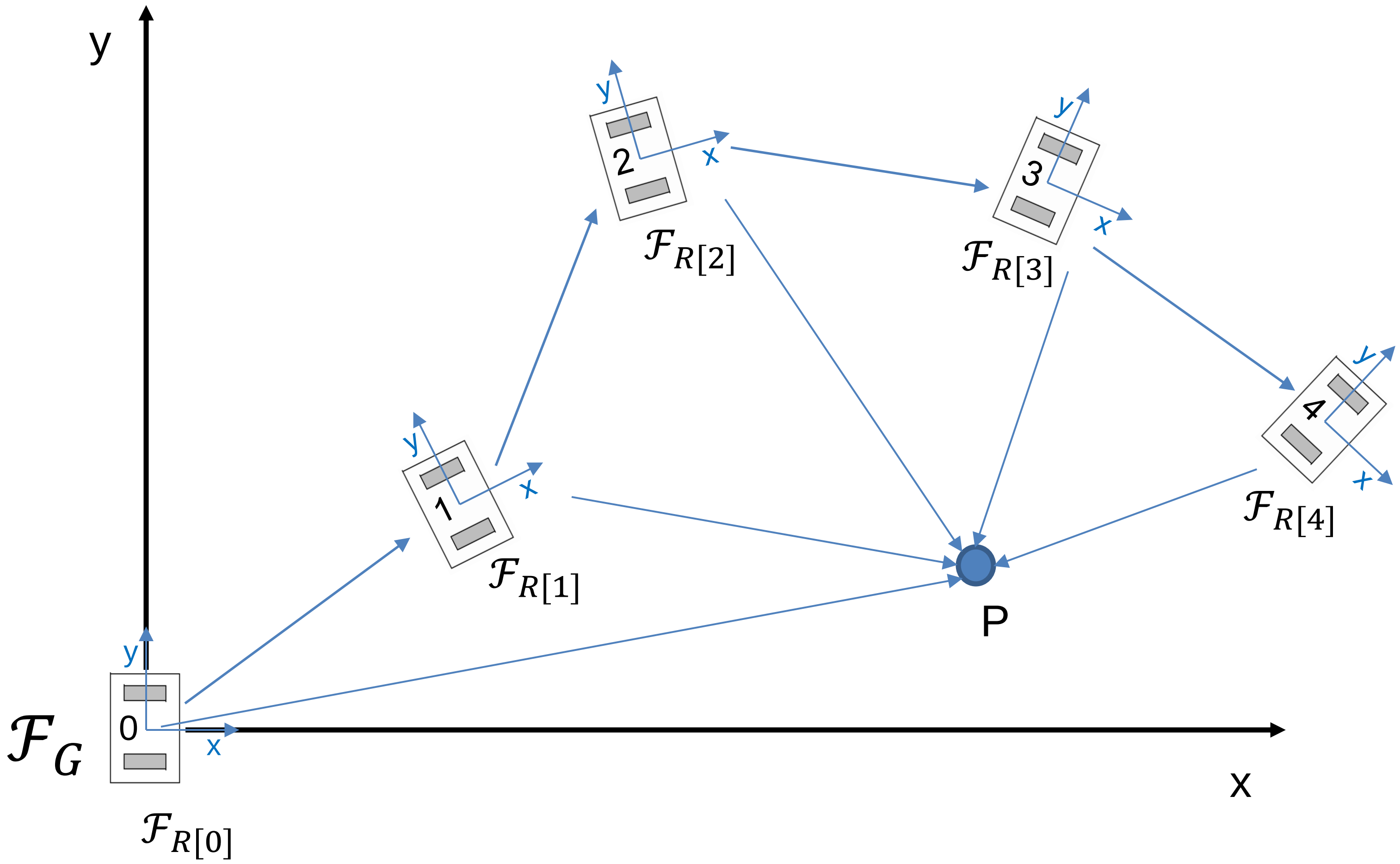
With respect to the robot start pose:
Where is the robot now?

Two approaches – same result:

- Geometry (easy in 2D)
- Transforms (better for 3D)

$\mathcal{F}_{R[X]}$: The ***F***rame of reference (the local coordinate system) of the ***R***obot at the time ***X***

Use of robot frames $\mathcal{F}_{R[X]}$

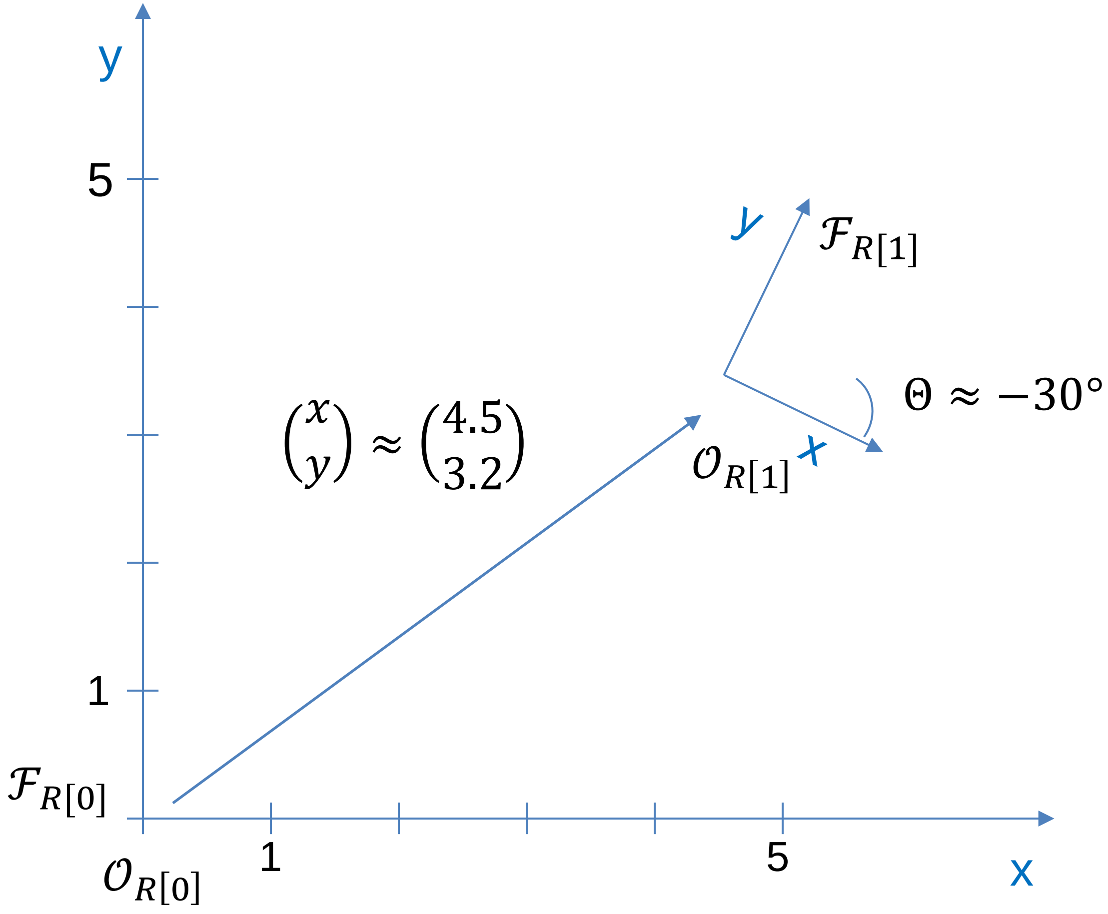


$\mathcal{O}_{R[X]}$: Origin of $\mathcal{F}_{R[X]}$ (coordinates (0, 0))

$\overrightarrow{\mathcal{O}_{R[X]}\mathrm{P}}$: position vector from $\mathcal{O}_{R[X]}$ to point P - $\binom{x}{y}$

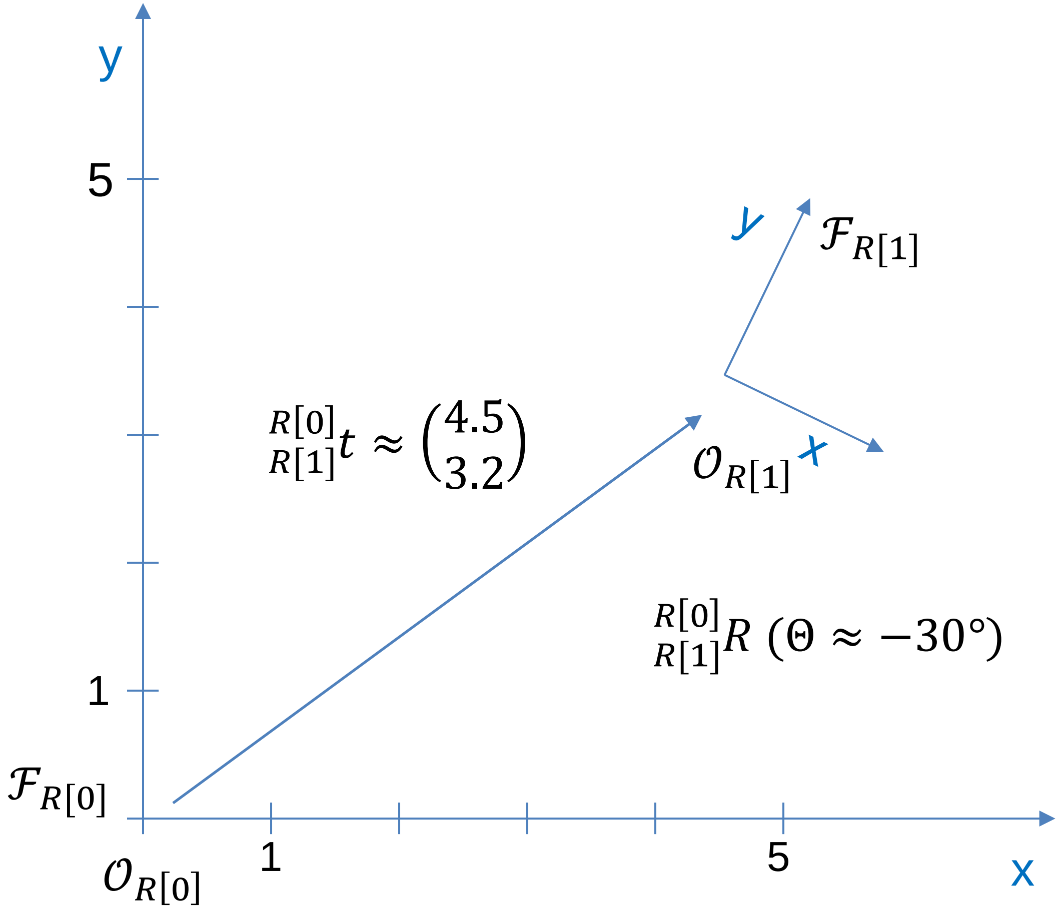
- Object P is observed at times 0 to 4
- Object P is static (does not move)
- The Robot moves (e.g. $\mathcal{F}_{R[0]} \neq \mathcal{F}_{R[1]}$)
- => (x, y) coordinates of P are different in all frames, for example:
  - $\overrightarrow{\mathcal{O}_{R[0]}\mathrm{P}} \neq \overrightarrow{\mathcal{O}_{R[1]}\mathrm{P}}$

Position, Orientation & Pose



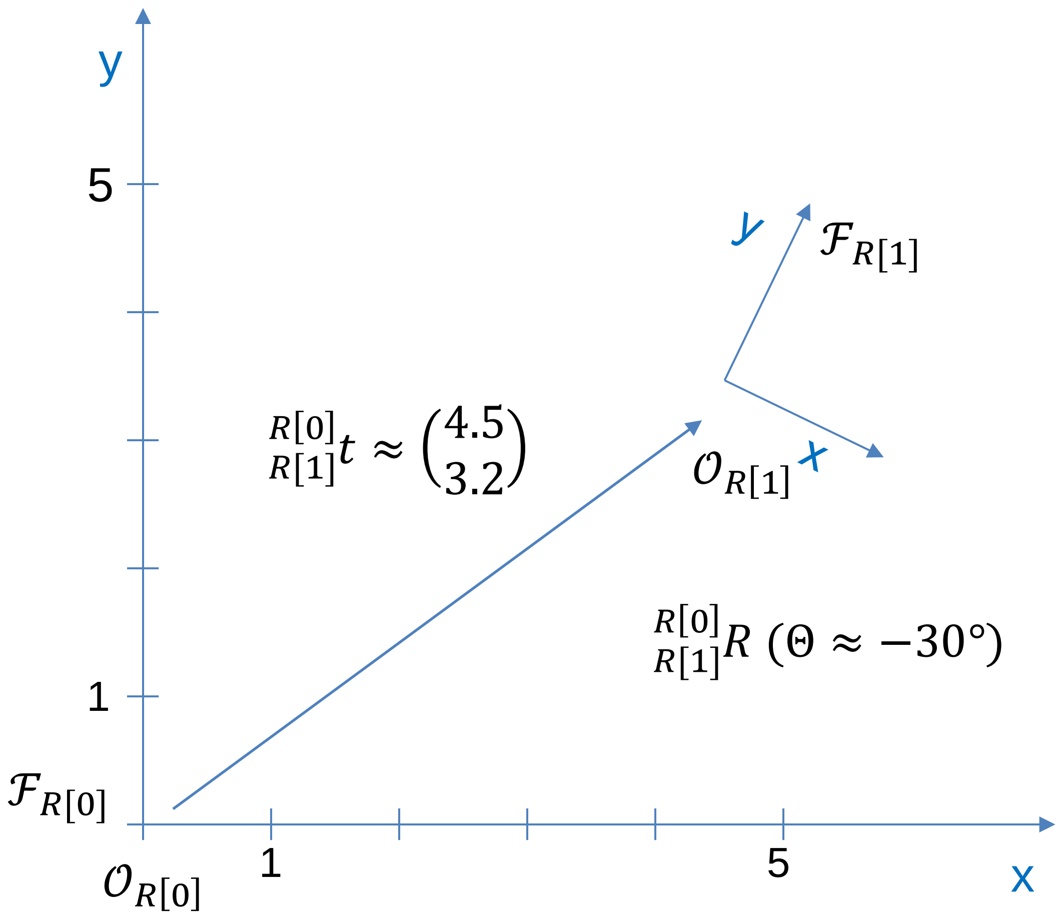
- **Position**:
  - $\begin{pmatrix} x \\ y \end{pmatrix}$ coordinates of any object or point (or another frame)
  - with respect to (wrt.) a specified frame
- **Orientation**:
  - (Θ) angle of any oriented object (or another frame)
  - with respect to (wrt.) a specified frame
- **Pose**:
  - $\begin{pmatrix} x \\ y \\ \Theta \end{pmatrix}$ position and orientation of any oriented object
  - with respect to (wrt.) a specified frame

Translation, Rotation & Transform



- **Translation**:
  - $\begin{pmatrix} x \\ y \end{pmatrix}$ difference, change, motion from one reference frame to another reference frame
- **Rotation**:
  - (Θ) difference in angle, rotation between one reference frame and another reference frame
- **Transform**:
  - $\begin{pmatrix} x \\ y \\ \Theta \end{pmatrix}$ difference, motion between one reference frame and another reference frame

Position & Translation, Orientation & Rotation



- $\mathcal{F}_{R[X]}$: Frame of reference of the robot at time X
- Where is that frame $\mathcal{F}_{R[X]}$?
  - Can only be expressed with respect to (wrt.) another frame (e.g. global Frame $\mathcal{F}_G$) =>
  - Pose of $\mathcal{F}_{R[X]}$ wrt. $\mathcal{F}_G$

- $\mathcal{O}_{R[X]}$: Origin of $\mathcal{F}_{R[X]}$
  - $\overrightarrow{\mathcal{O}_{R[X]}\mathcal{O}_{R[X+1]}}$: **Position** of $\mathcal{F}_{R[X+1]}$ wrt. $\mathcal{F}_{R[X]}$

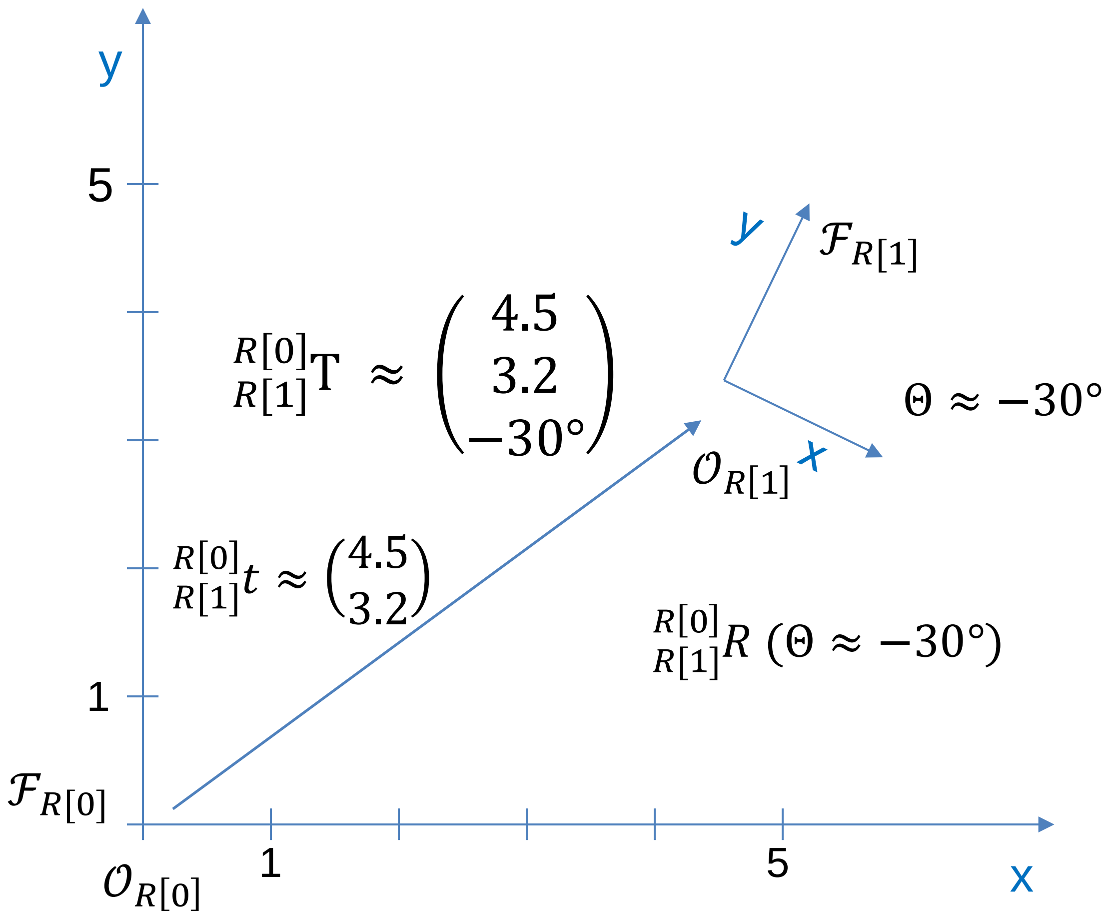
    so $\mathcal{O}_{R[X+1]}$ wrt. $\mathcal{F}_{R[X]}$

    $\triangleq\ {}^{R[X]}_{R[X+1]}t$: **Translation**

- The angle Θ between the x-Axes:
  - **Orientation** of $\mathcal{F}_{R[X+1]}$ wrt. $\mathcal{F}_{R[X]}$

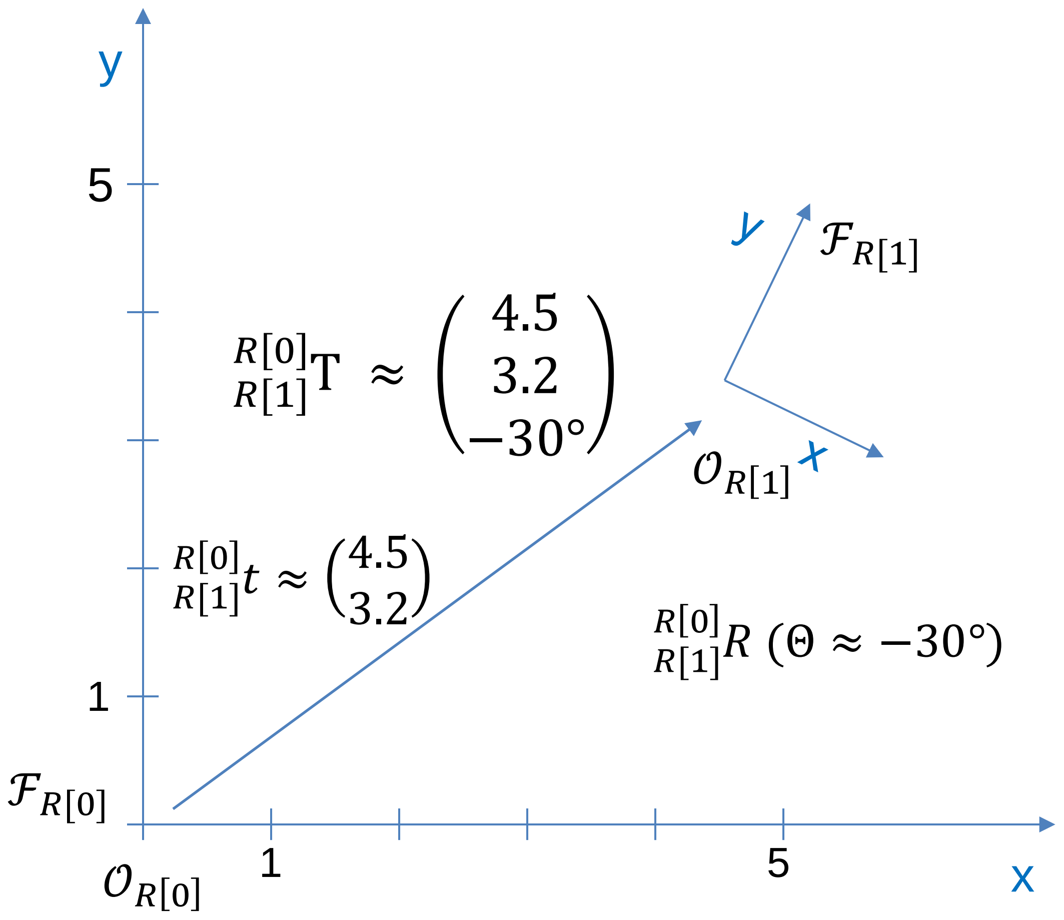
    $\triangleq\ {}^{R[X]}_{R[X+1]}R$: **Rotation** of $\mathcal{F}_{R[X+1]}$ wrt. $\mathcal{F}_{R[X]}$

Transform



- ${}^{R[X]}_{R[X+1]}t$: **Translation**
  - Position vector (x, y) of $R[X+1]$ wrt. $R[X]$
- ${}^{R[X]}_{R[X+1]}R$: **Rotation**
  - Angle (Θ) of $R[X+1]$ wrt. $R[X]$
- **Transform:** $${}^{R[X]}_{R[X+1]}\mathrm{T} \equiv \begin{Bmatrix} {}^{R[X]}_{R[X+1]}t \\ {}^{R[X]}_{R[X+1]}R \end{Bmatrix}$$

Geometry approach to Odometry



We want to know:

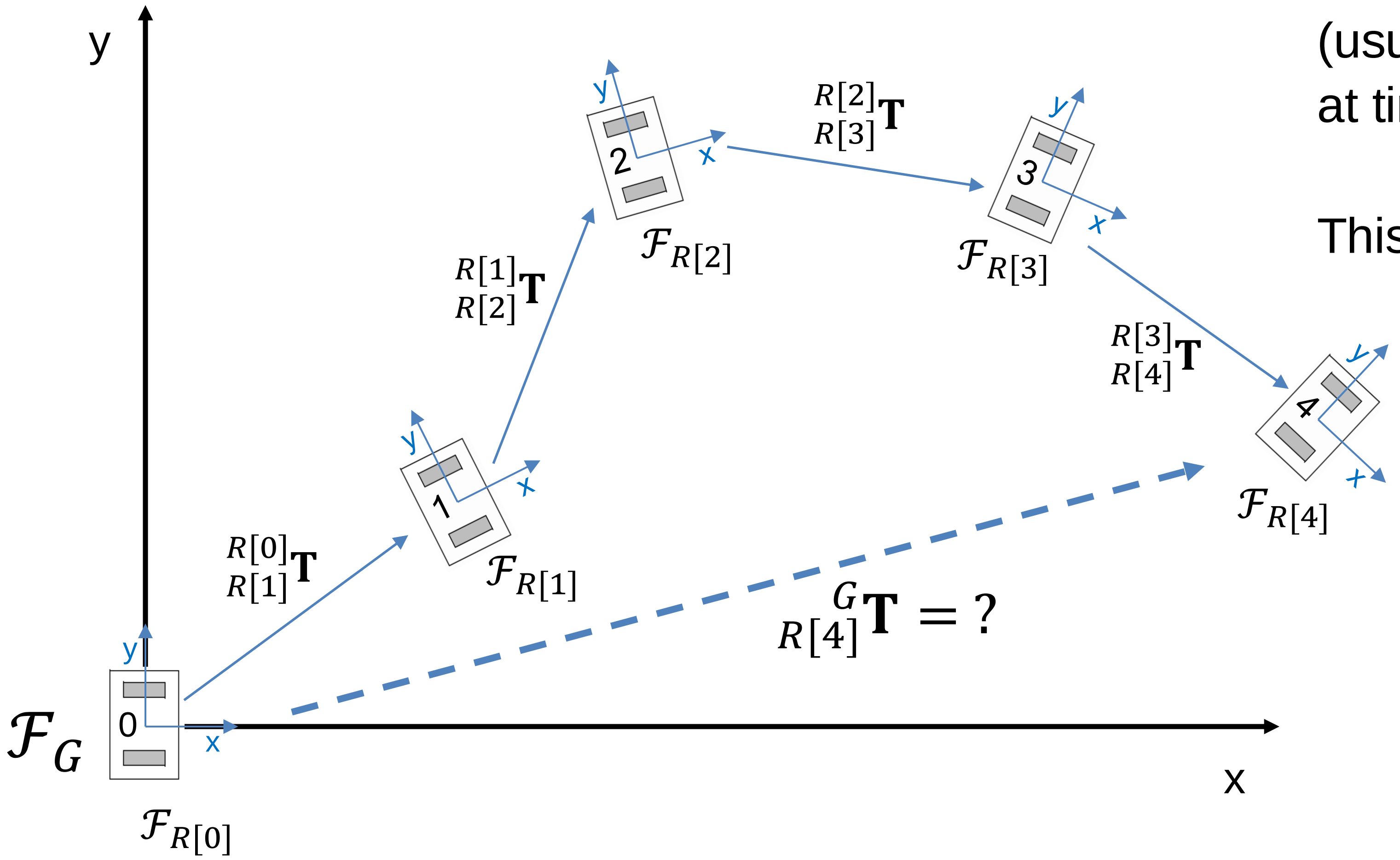
- Position of the robot (x, y)
- Orientation of the robot (Θ)
- => together: Pose $\begin{pmatrix} x \\ y \\ \Theta \end{pmatrix}$

With respect to (wrt.) $\mathcal{F}_G$: The global frame; global coordinate system

$$\mathcal{F}_{R[0]} = \mathcal{F}_G \Rightarrow \ {}^{G}\mathcal{F}_{R[0]} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$$

$${}^{G}\mathcal{F}_{R[1]} = {}^{R[0]}_{R[1]}\mathrm{T} \approx \begin{pmatrix} 4.5 \\ 3.2 \\ 30° \end{pmatrix}$$

Mathematical approach: Transforms



Where is the Robot now?

The pose of $\mathcal{F}_{R[X]}$ with respect to $\mathcal{F}_G$ (usually = $\mathcal{F}_{R[0]}$) is the pose of the robot at time X.

This is equivalent to ${}^{G}_{R[X]}\mathbf{T}$

Chaining of Transforms

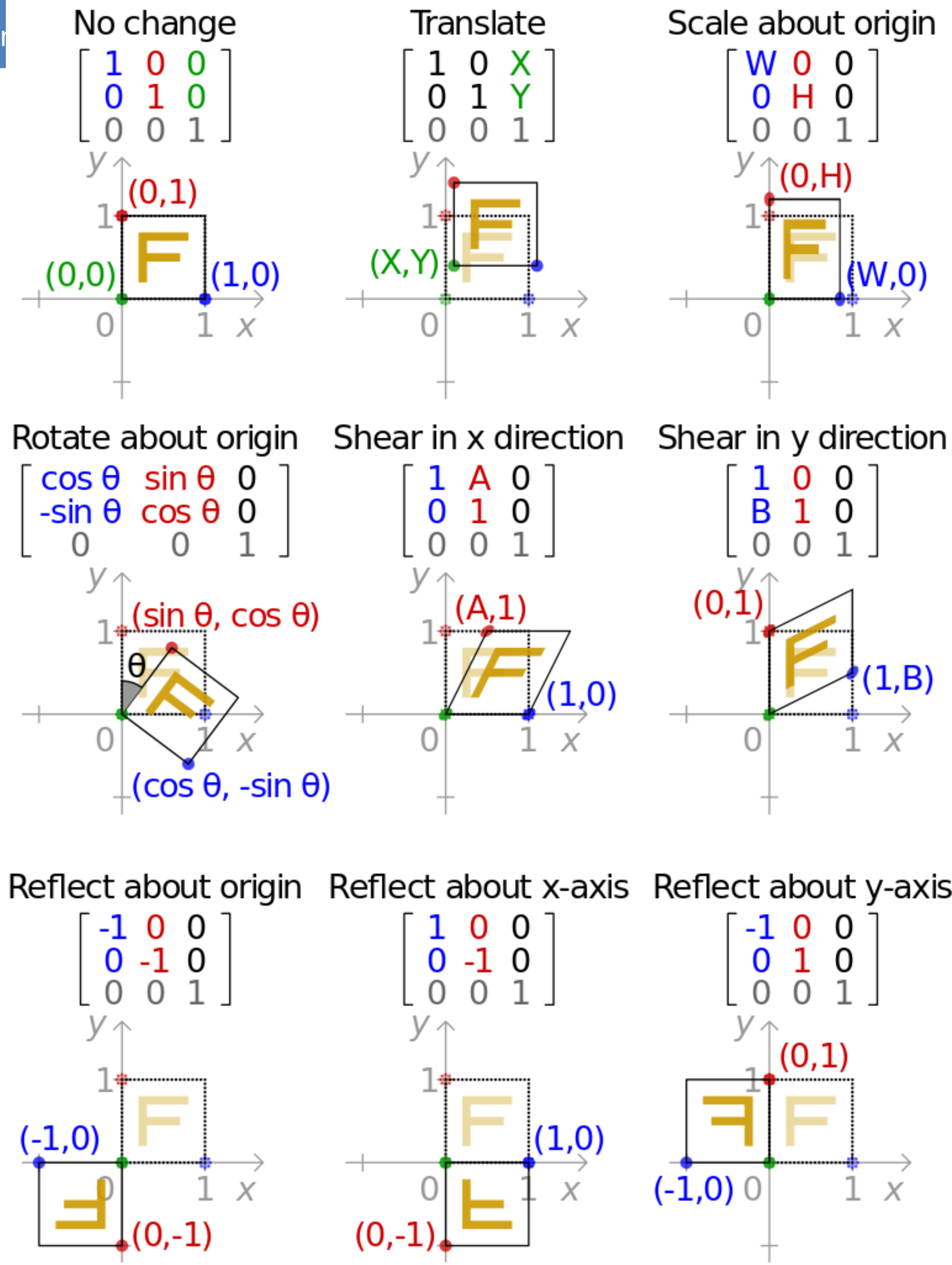
$$ {}^{G}_{R[X+1]}\mathbf{T} = {}^{G}_{R[X]}\mathbf{T} \; {}^{R[X]}_{R[X+1]}\mathbf{T} $$

often: $\mathcal{F}_G \equiv \mathcal{F}_{R[0]} \Rightarrow {}^{G}_{R[0]}\mathbf{T} = id$

TRANSFORMS & STUFF ☺

Affine Transformation

- Function between affine spaces. Preserves:
  - points,
  - straight lines
  - planes
  - sets of parallel lines remain parallel
- Allows:
  - Interesting for Robotics: translation, rotation, (scaling), and chaining of those
  - Not so interesting for Robotics: reflection, shearing, homothetic transforms
- Rotation and Translation: $\begin{bmatrix} \cos\theta & \sin\theta & X \\ -\sin\theta & \cos\theta & Y \\ 0 & 0 & 1 \end{bmatrix}$



Math: Rigid Transformation

- Geometric transformation that preserves Euclidean distance between pairs of points.
- Includes reflections (i.e. change from right-hand to left-hand coorinate system and back)
- Just rotation & translation: rigid motions or proper rigid transformations:
  - Decomposed to rotation and translation
  - => subset of Affine Transofrmations

- In Robotics: Just use term **Transform** or **Transformation** for rigid motions (without reflections)

Lie groups for transformations

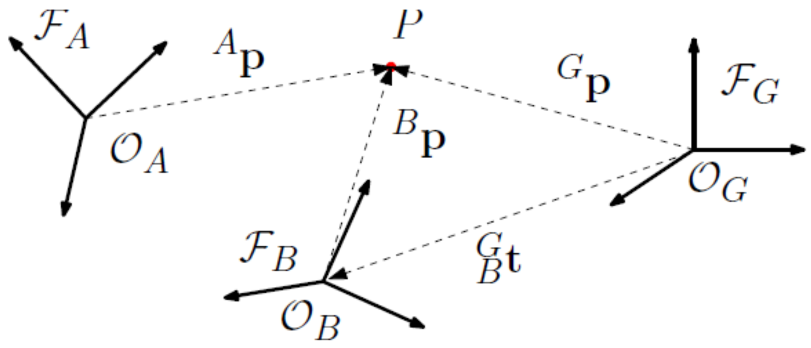
- Smoothly differentiable Group
- No singularities
- Good interpolation

- SO: Special Orthorgonal group
- SE: Special Euclidian group
- Sim_ilarity transform group

Group	Description	Dim.	Matrix Representation
SO(3)	3D Rotations	3	3D rotation matrix
SE(3)	3D Rigid transformations	6	Linear transformation on homogeneous 4-vectors
SO(2)	2D Rotations	1	2D rotation matrix
SE(2)	2D Rigid transformations	3	Linear transformation on homogeneous 3-vectors
Sim(3)	3D Similarity transformations (rigid motion + scale)	7	Linear transformation on homogeneous 4-vectors

http://ethaneade.com/lie.pdf

Transform



Notation	Meaning
$\mathcal{F}_{\mathrm{R}[k]}$	Coordinate frame attached to object 'R' (usually the robot) at sample time-instant k.
$\mathcal{O}_{\mathrm{R}[k]}$	Origin of $\mathcal{F}_{\mathrm{R}[k]}$.
${}^{\mathrm{R}[k]}\mathbf{p}$	For any general point P, the position vector $\overrightarrow{\mathcal{O}_{\mathrm{R}[k]}P}$ resolved in $\mathcal{F}_{\mathrm{R}[k]}$.
${}^{\mathrm{H}}\hat{\mathbf{x}}_{\mathrm{R}}$	The x-axis direction of $\mathcal{F}_{\mathrm{R}}$ resolved in $\mathcal{F}_{\mathrm{H}}$. Similarly, ${}^{\mathrm{H}}\hat{\mathbf{y}}_{\mathrm{R}}$, ${}^{\mathrm{H}}\hat{\mathbf{z}}_{\mathrm{R}}$ can be defined. Obviously, ${}^{\mathrm{R}}\hat{\mathbf{x}}_{\mathrm{R}} = \hat{\mathbf{e}}_1$. Time indices can be added to the frames, if necessary.
${}^{\mathrm{R}[k]}_{\mathrm{S}[k']}\mathbf{R}$	The rotation-matrix of $\mathcal{F}_{\mathrm{S}[k']}$ with respect to $\mathcal{F}_{\mathrm{R}[k]}$.
${}^{\mathrm{R}}_{\mathrm{S}}\mathbf{t}$	The translation vector $\overrightarrow{\mathcal{O}_{\mathrm{R}}\mathcal{O}_{\mathrm{S}}}$ resolved in $\mathcal{F}_{\mathrm{R}}$.

Transform between two coordinate frames

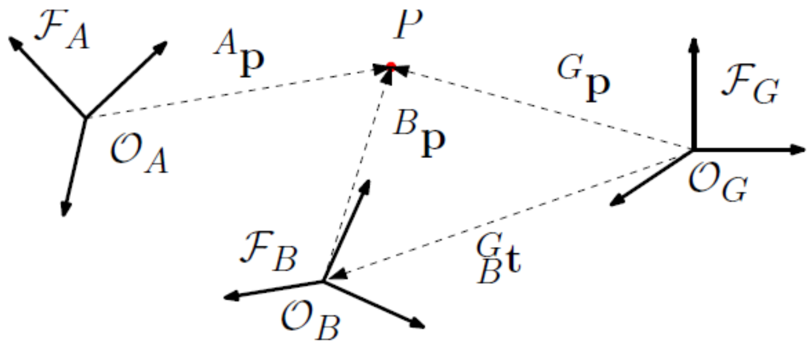
$${}^{\mathrm{G}}_{\mathrm{A}}\mathbf{t} \triangleq \overrightarrow{\mathcal{O}_{\mathrm{G}}\mathcal{O}_{\mathrm{A}}} \text{ resolved in } \mathcal{F}_{\mathrm{G}}$$

$$\begin{pmatrix} {}^{\mathrm{G}}\mathbf{p} \\ 1 \end{pmatrix} \equiv \begin{pmatrix} {}^{\mathrm{G}}_{\mathrm{A}}\mathbf{R} & {}^{\mathrm{G}}_{\mathrm{A}}\mathbf{t} \\ \mathbf{0}_{1\times[2,3]} & 1 \end{pmatrix} \begin{pmatrix} {}^{\mathrm{A}}\mathbf{p} \\ 1 \end{pmatrix} \qquad {}^{\mathrm{G}}_{\mathrm{A}}\mathbf{T} \equiv \left\{ \begin{matrix} {}^{\mathrm{G}}_{\mathrm{A}}\mathbf{t} \\ {}^{\mathrm{G}}_{\mathrm{A}}\mathbf{R} \end{matrix} \right\}$$

$$\begin{aligned} {}^{\mathrm{G}}\mathbf{p} &= {}^{\mathrm{G}}_{\mathrm{A}}\mathbf{R}\ {}^{\mathrm{A}}\mathbf{p} + {}^{\mathrm{G}}_{\mathrm{A}}\mathbf{t} \\ &\triangleq {}^{\mathrm{G}}_{\mathrm{A}}\mathbf{T}(\ {}^{\mathrm{A}}\mathbf{p}). \end{aligned}$$

$$\begin{bmatrix} \cos\theta & -\sin\theta & {}^{\mathrm{G}}_{\mathrm{A}}t_x \\ \sin\theta & \cos\theta & {}^{\mathrm{G}}_{\mathrm{A}}t_y \\ 0 & 0 & 1 \end{bmatrix}$$

Transform: Operations



Transform between two coordinate frames (chaining, compounding):

$$ {}^G_B\mathbf{T} = {}^G_A\mathbf{T}\,{}^A_B\mathbf{T} \equiv \left\{ \begin{matrix} {}^G_A\mathbf{R}\,{}^A_B\mathbf{t} + {}^G_A\mathbf{t} \\ {}^G_A\mathbf{R}\,{}^A_B\mathbf{R} \end{matrix} \right\} $$

Inverse of a Transform :

$$ {}^B_A\mathbf{T} = {}^A_B\mathbf{T}^{-1} \equiv \left\{ \begin{matrix} -{}^A_B\mathbf{R}^\top\,{}^A_B\mathbf{t} \\ {}^A_B\mathbf{R}^\top \end{matrix} \right\} $$

Relative (Difference) Transform :

$$ {}^B_A\mathbf{T} = {}^G_B\mathbf{T}^{-1}\,{}^G_A\mathbf{T} $$

See: **Quick Reference to Geometric Transforms in Robotics** by Kaustubh Pathak on the webpage!

Chaining :

$$ {}^{G}_{R[X+1]}\mathbf{T} = {}^{G}_{R[X]}\mathbf{T}\ {}^{R[X]}_{R[X+1]}\mathbf{T} \equiv \begin{Bmatrix} {}^{G}_{R[X]}\mathrm{R}\ {}^{R[X]}_{R[X+1]}t + {}^{G}_{R[X]}t \\ {}^{G}_{R[X]}\mathrm{R}\ {}^{R[X]}_{R[X+1]}\mathrm{R} \end{Bmatrix} = \begin{Bmatrix} {}^{G}_{R[X+1]}t \\ {}^{G}_{R[X+1]}\mathrm{R} \end{Bmatrix} $$

In 2D Translation:

$$ \begin{bmatrix} {}^{G}_{R[X+1]}t_x \\ {}^{G}_{R[X+1]}t_y \\ 1 \end{bmatrix} = \begin{bmatrix} \cos {}^{G}_{R[X]}\theta & -\sin {}^{G}_{R[X]}\theta & {}^{G}_{R[X]}\mathrm{t_x} \\ \sin {}^{G}_{R[X]}\theta & \cos {}^{G}_{R[X]}\theta & {}^{G}_{R[X]}\mathrm{t_y} \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} {}^{R[X]}_{R[X+1]}\mathrm{t_x} \\ {}^{R[X]}_{R[X+1]}t_y \\ 1 \end{bmatrix} $$

In 2D Rotation:

$$ {}^{G}_{R[X+1]}\mathrm{R} = \begin{bmatrix} \cos {}^{G}_{R[X+1]}\theta & -\sin {}^{G}_{R[X+1]}\theta \\ \sin {}^{G}_{R[X+1]}\theta & \cos {}^{G}_{R[X+1]}\theta \end{bmatrix} = \begin{bmatrix} \cos {}^{G}_{R[X]}\theta & -\sin {}^{G}_{R[X]}\theta \\ \sin {}^{G}_{R[X]}\theta & \cos {}^{G}_{R[X]}\theta \end{bmatrix} \begin{bmatrix} \cos {}^{R[X]}_{R[X+1]}\theta & -\sin {}^{R[X]}_{R[X+1]}\theta \\ \sin {}^{R[X]}_{R[X+1]}\theta & \cos {}^{R[X]}_{R[X+1]}\theta \end{bmatrix} $$

In 2D Rotation (simple): $ {}^{G}_{R[X+1]}\theta = {}^{G}_{R[X]}\theta + {}^{R[X]}_{R[X+1]}\theta $