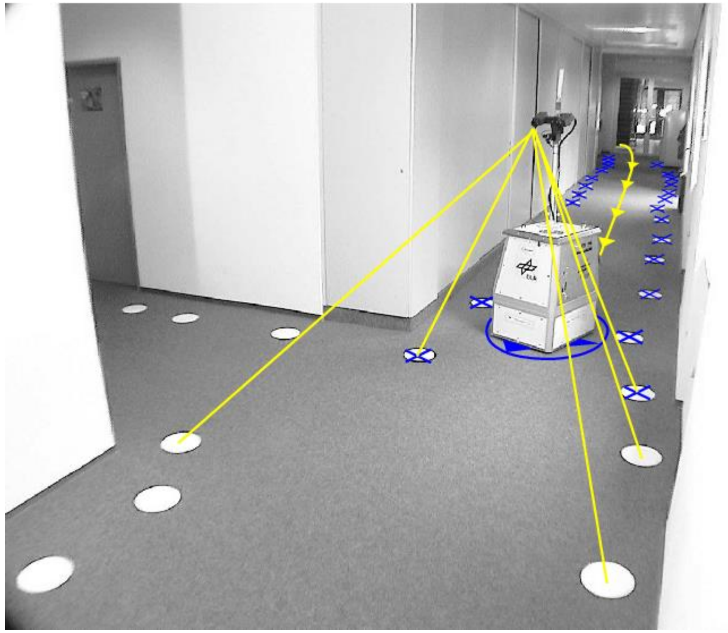
Robust Mechatronics

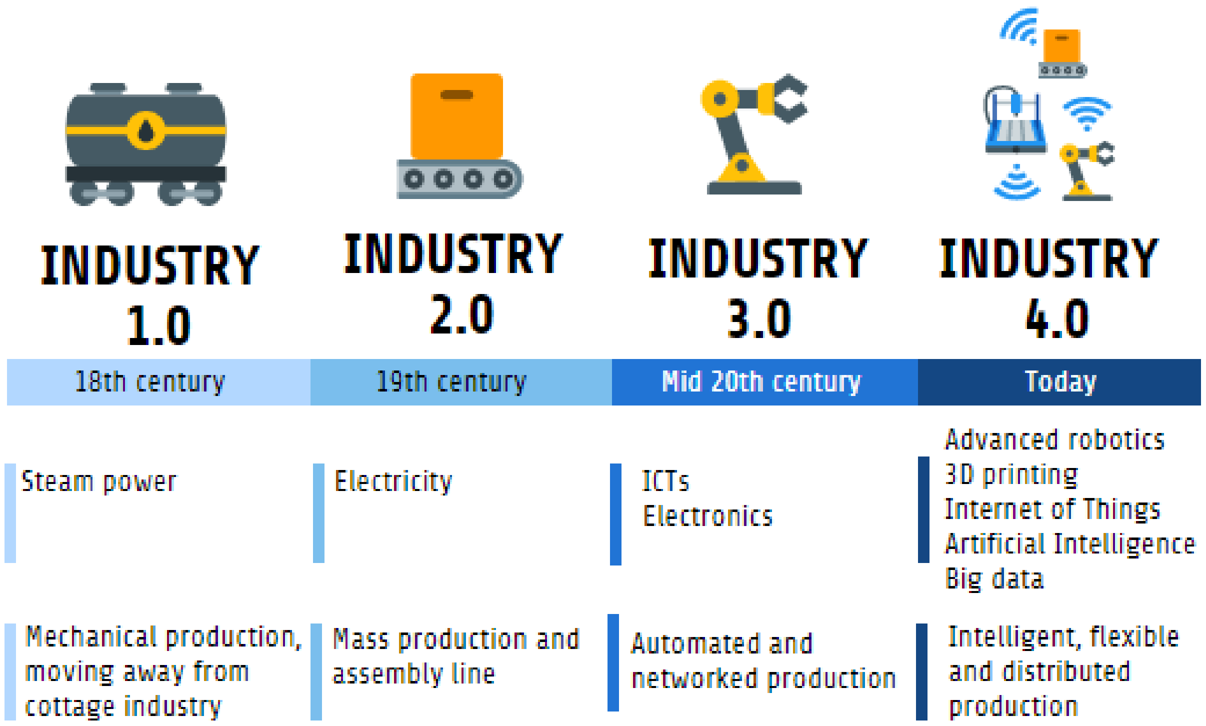
Localization and Mapping for Autonomous Mobile Systems

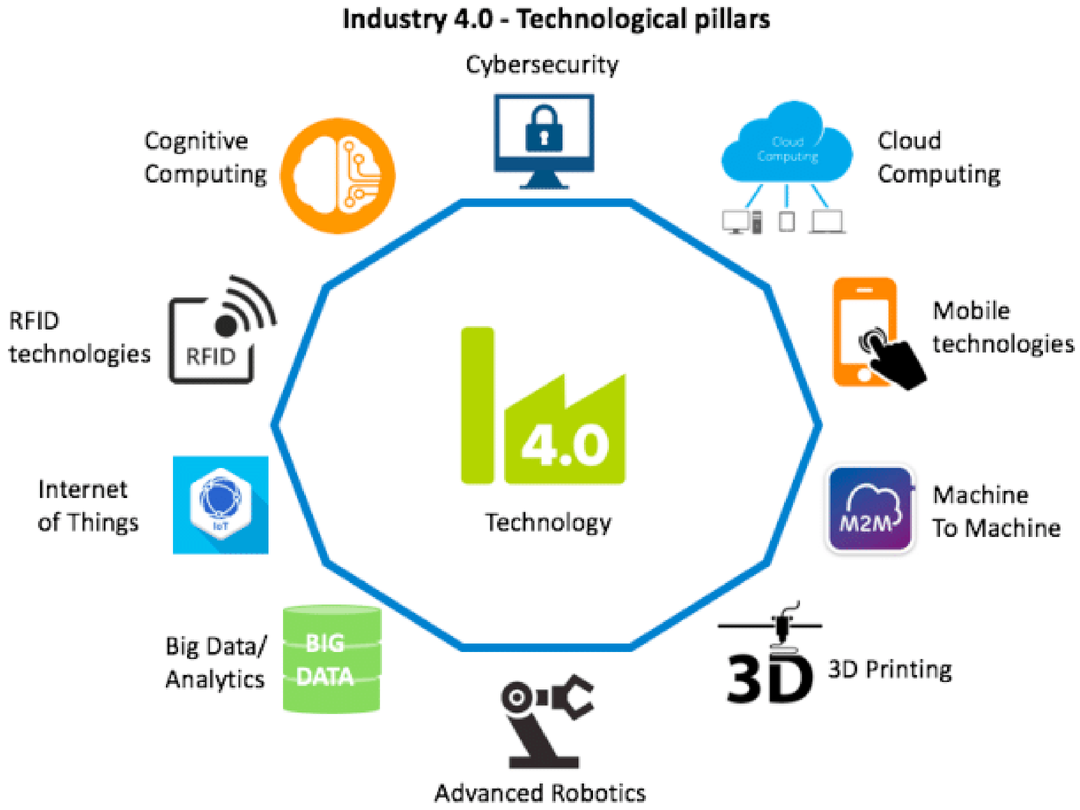


Dr Loukas Bampis, Assistant Professor
Mechatronics & Systems Automation Lab

Localization and Mapping for Autonomous Mobile Systems

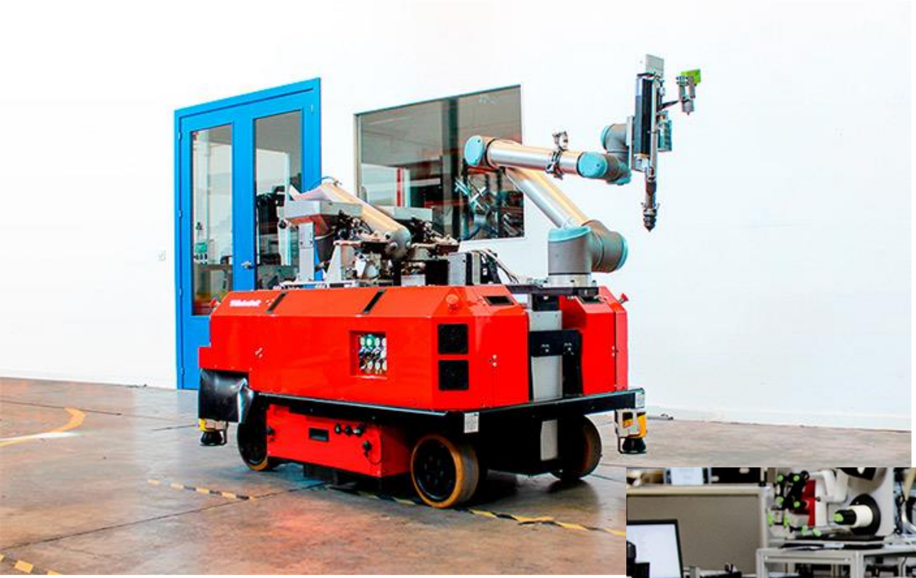
Industry 4.0 and the Age of Mobile Robotics

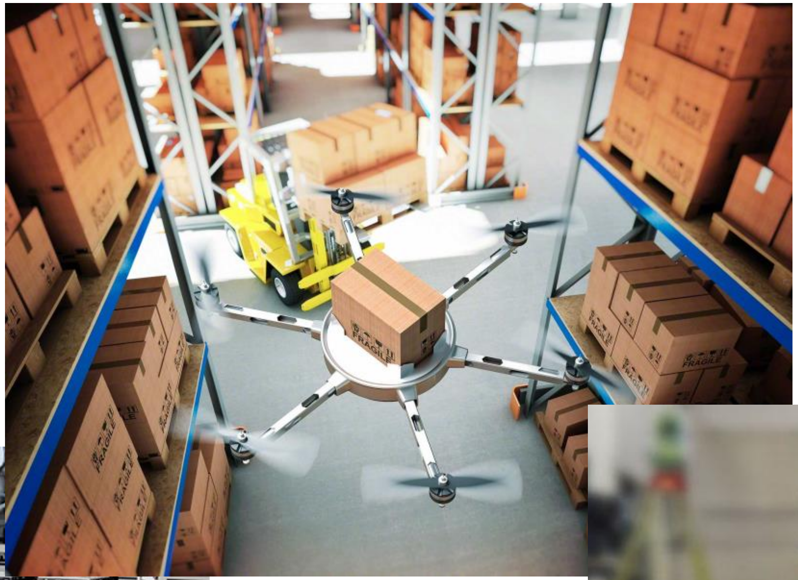




Localization and Mapping for Autonomous Mobile Systems

Industry 4.0 and the Age of Mobile Robotics



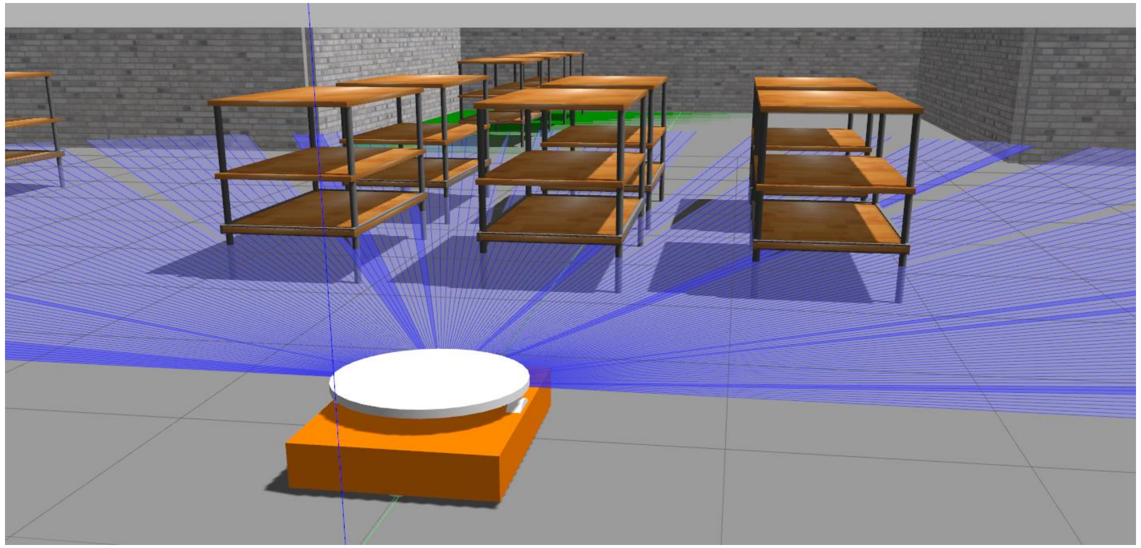


Localization and Mapping for Autonomous Mobile Systems

Simultaneous Localization And Mapping

Provide the means for a mobile robot to

- map an unknown environment and
- identify its location within it



Today’s lecture in simple terms

- Evolution of Localization and Mapping approaches
- Basic theory and methods
- Re-evaluation and correction of the output

Now, lets define some terms for this lecture

What is a mobile robot?

- An autonomous machine equipped with a motion system and a set of sensors

What is an unknown environment?

- Our knowledge of the world's structure is zero or the environment is constantly changing

What is localization and mapping for a robot?

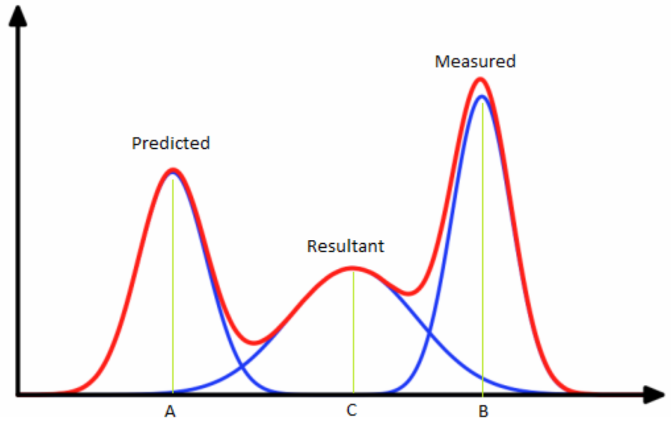
- Localization: Identify the robot's pose within a specified environment
- Mapping: Measure the environment's structure given the robot's pose
- Chicken-or-egg problem: Simultaneous Localization and Mapping (*SLAM*)

SLAM – Simultaneous Localization And Mapping

- Iterative process
  - Given your best knowledge about the environment, measure your location
  - Given your best localization estimate, update your knowledge about the environment

- Means to perceive the world
  - Noisy sensory inputs

- In its core: Estimation theory
  - Probabilistic Models
  - Most recently: Deep Learning



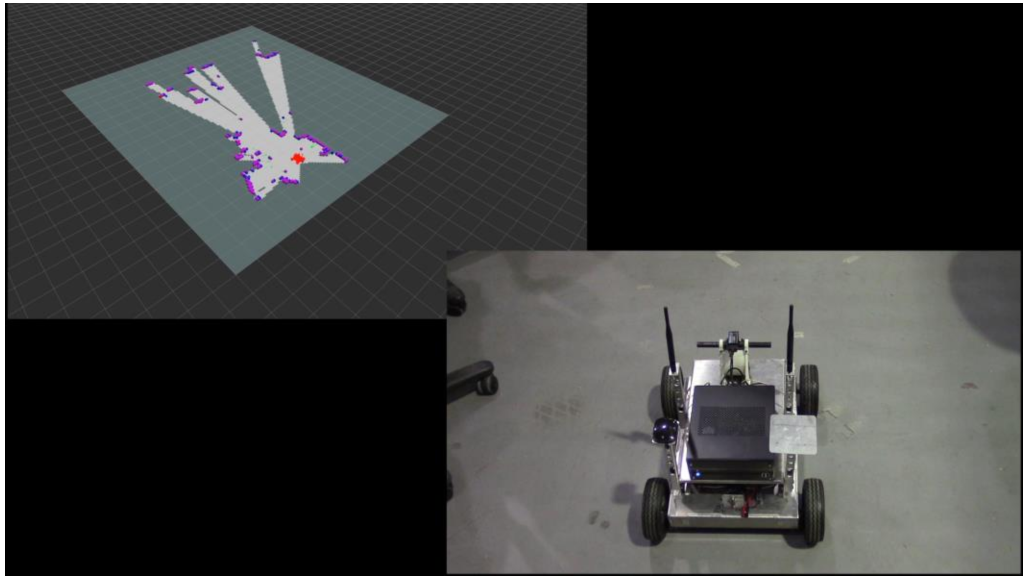
SLAM VS Deep Learning

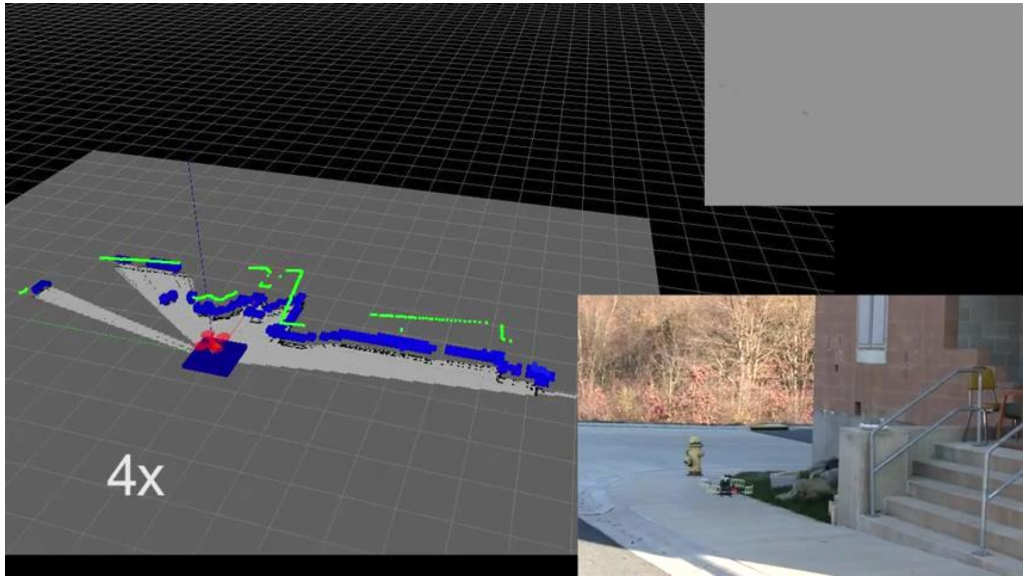
- Typical SLAM focuses on geometric problems
- Deep Learning is the master of perception (recognition) problems

- If you want a robot to go towards your refrigerator without hitting a wall, use SLAM
- If you want the robot to identify the items inside your fridge, use Deep Learning

- Most institutes split their graduate level curriculums into:
  - Learning-based Methods -> Deep Learning
  - Geometry-Based Methods -> SLAM

A video is worth a thousand words





Perception systems: The means for obtaining information

What kind of sensors do we need?

- Robot’s state
- World’s geometry

Mimic the living

- Humans do not have 5 senses
- At least 9 according to neuroscience

Perception systems: The means for obtaining information

Robot’s state

- Position:
  - Absolute: e.g., Global Navigation Satellite System (GNSS)
  - Relative: e.g., Wheel encoders

- Inertia:
  - Acceleration: Accelerometers
  - Orientation and Angular velocity: Gyroscopes
  - Earth frame alignment: Magnetometers

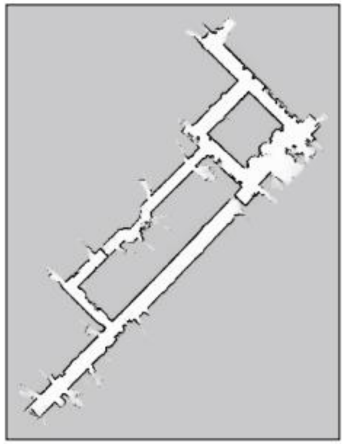
Perception systems: The means for obtaining information

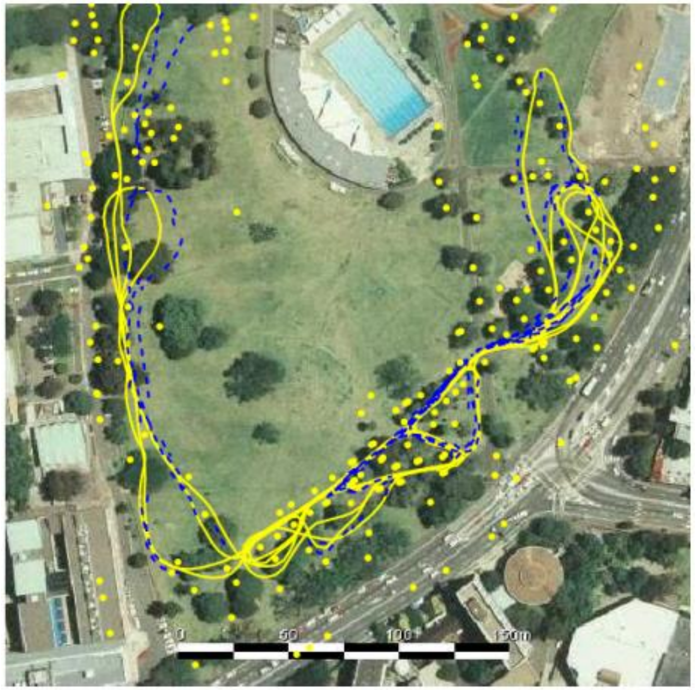
World's geometry

- Range-finders: Proximity
  - Ultrasonic
  - LIDAR
- Cameras: Appearance
  - Monocular
  - Stereo
  - RGBD
    - Infrared
    - Time of flight

The basics of SLAM: Taxonomy

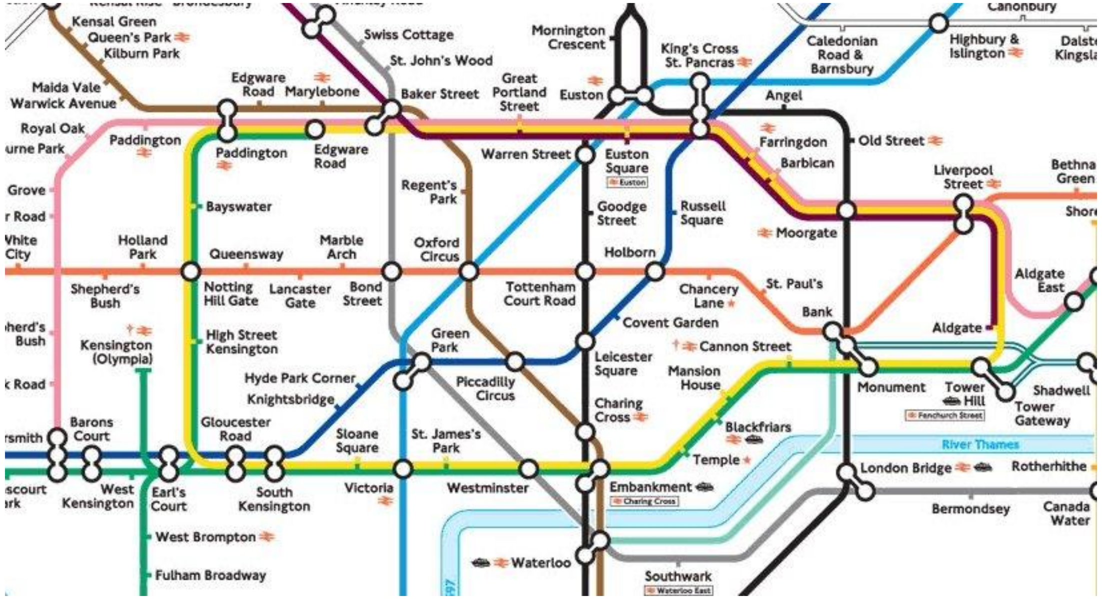
Volumetric VS Feature-Based SLAM

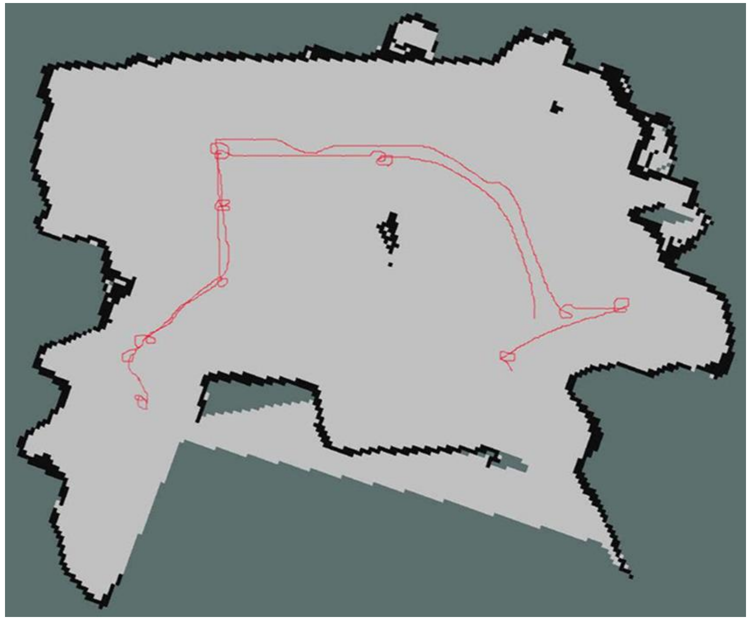




The basics of SLAM: Taxonomy

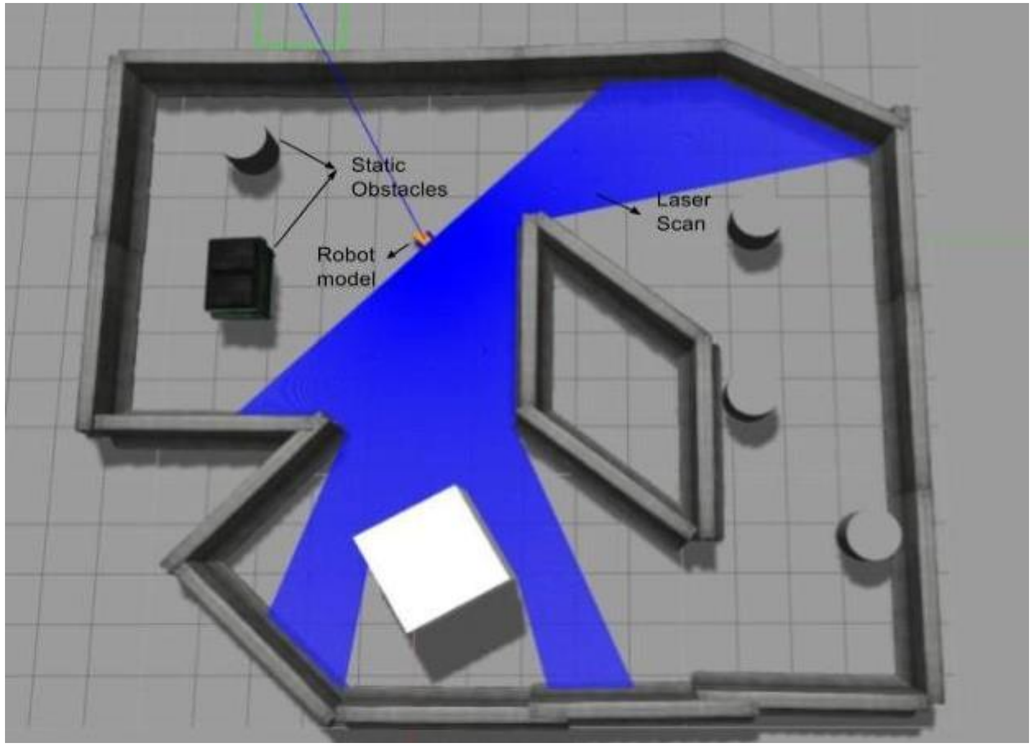
Topologic VS Geometric Maps





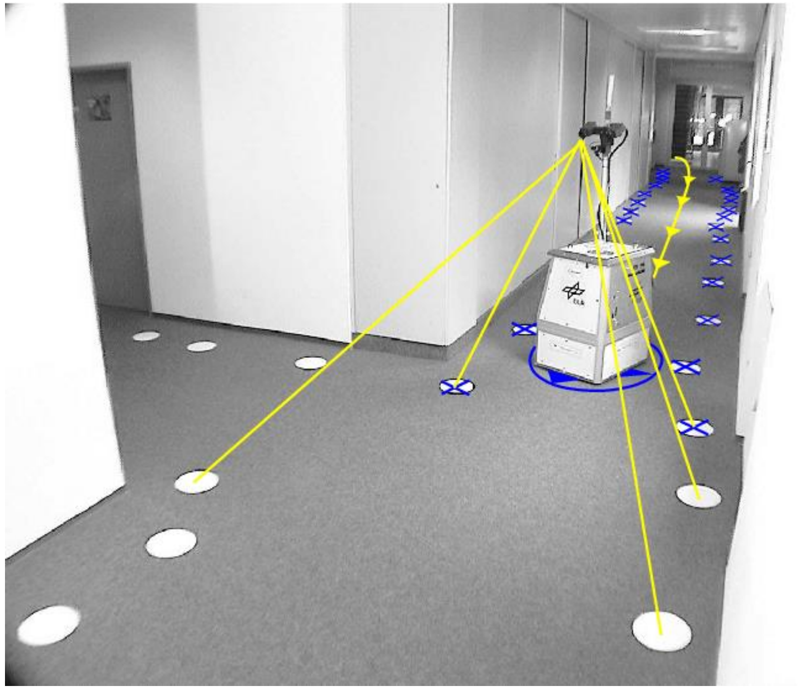
The basics of SLAM: Taxonomy

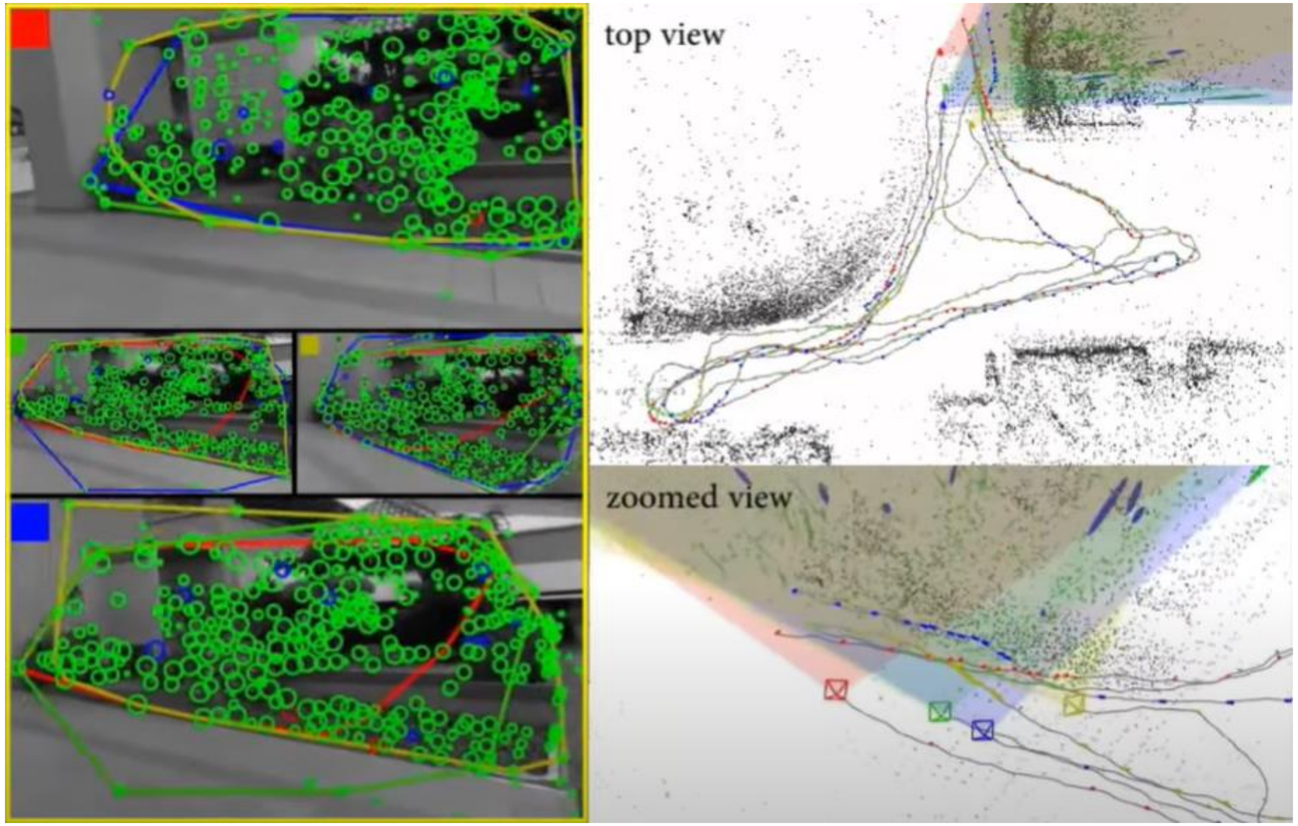
Static VS Dynamic Environments



The basics of SLAM: Taxonomy

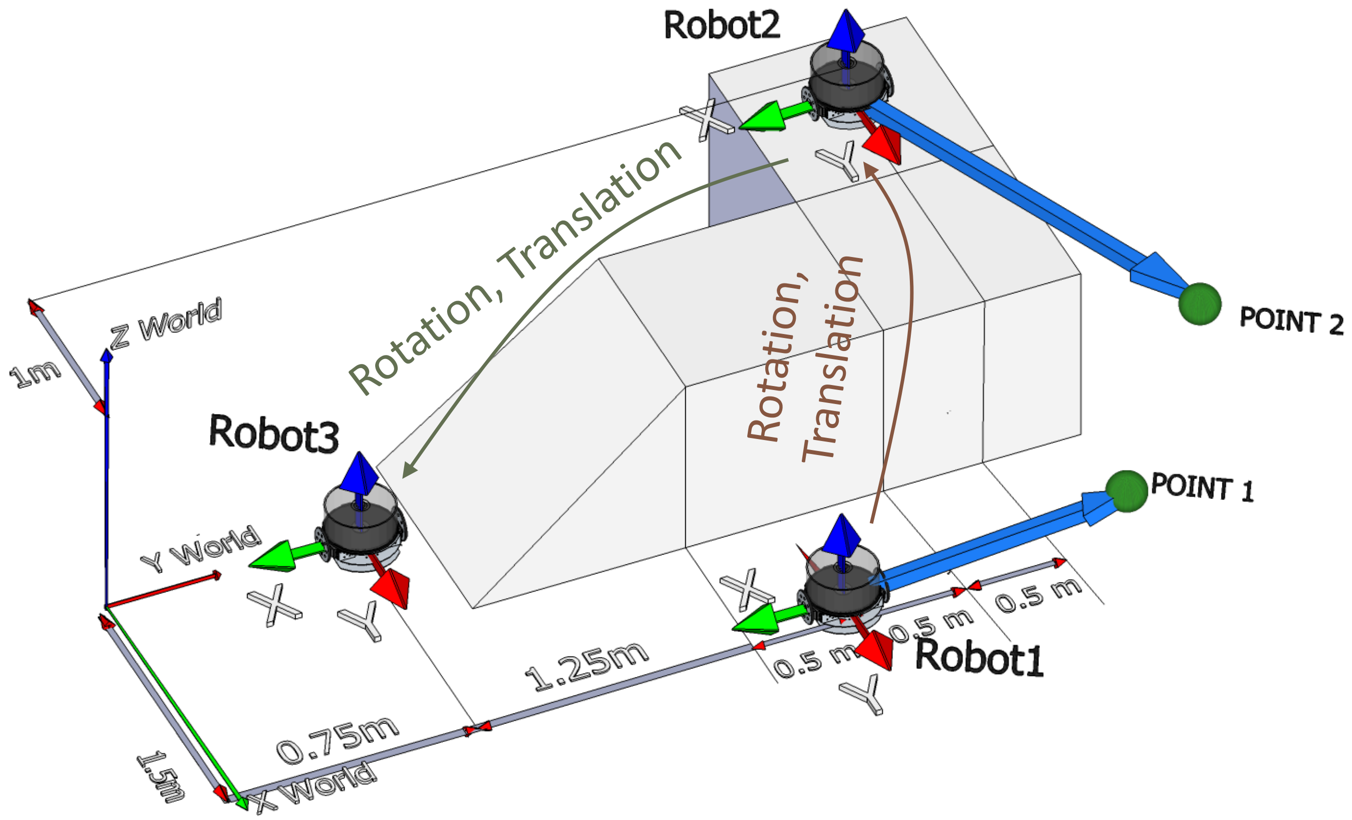
Single-Robot VS Multi-Robot





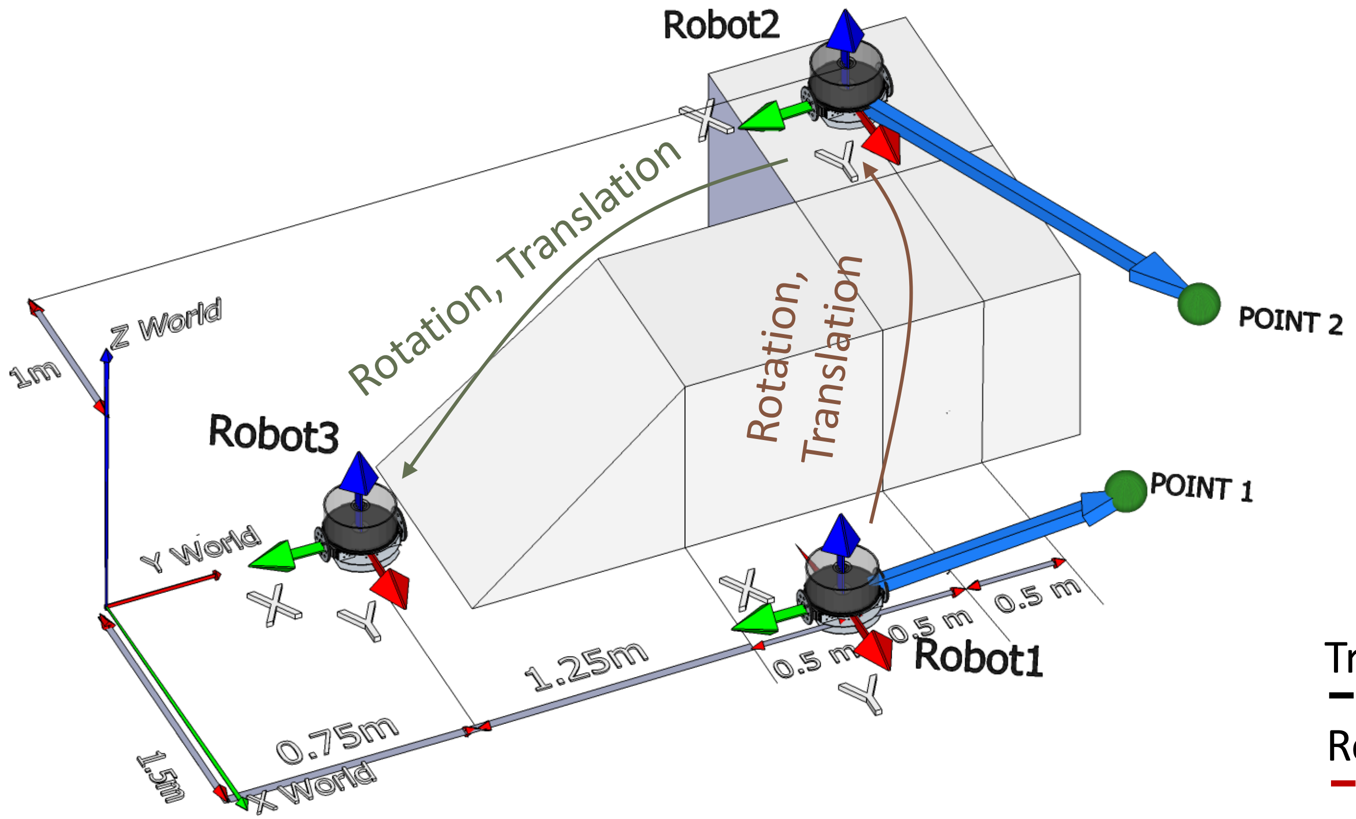
Localization and Mapping for Autonomous Mobile Systems

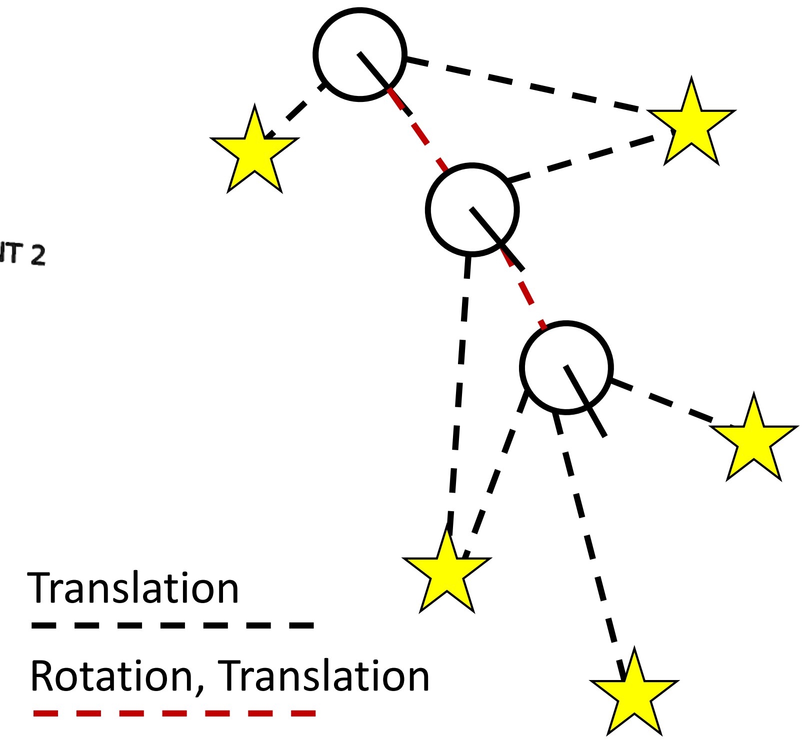
A common representation from SLAM (Graph SLAM)



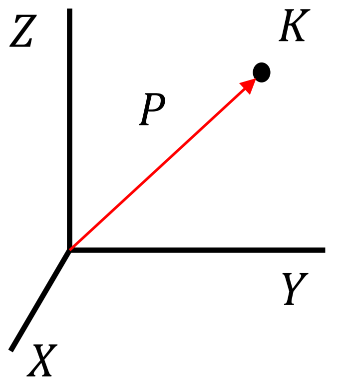
- Points are represented w.r.t frames of reference placed on each robot pose
- Robot poses are associated though their relative transformations

A common representation from SLAM (Graph SLAM)



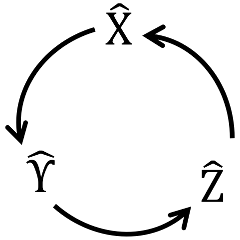


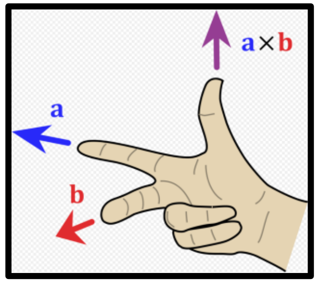
Frames of reference:: what?



A frame of reference holds:

- Specific position
- Specific orientation
- Specific axis arrangement





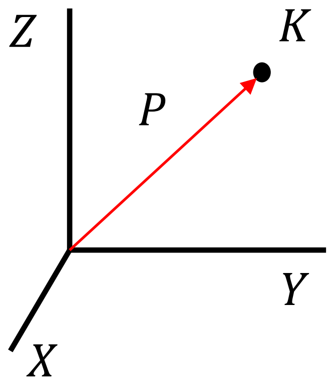
$$\hat{X} \times \hat{Y} = \hat{Z}$$

Frames of reference:: why?

Elements w/o dimensions (points):

- Definition of location:

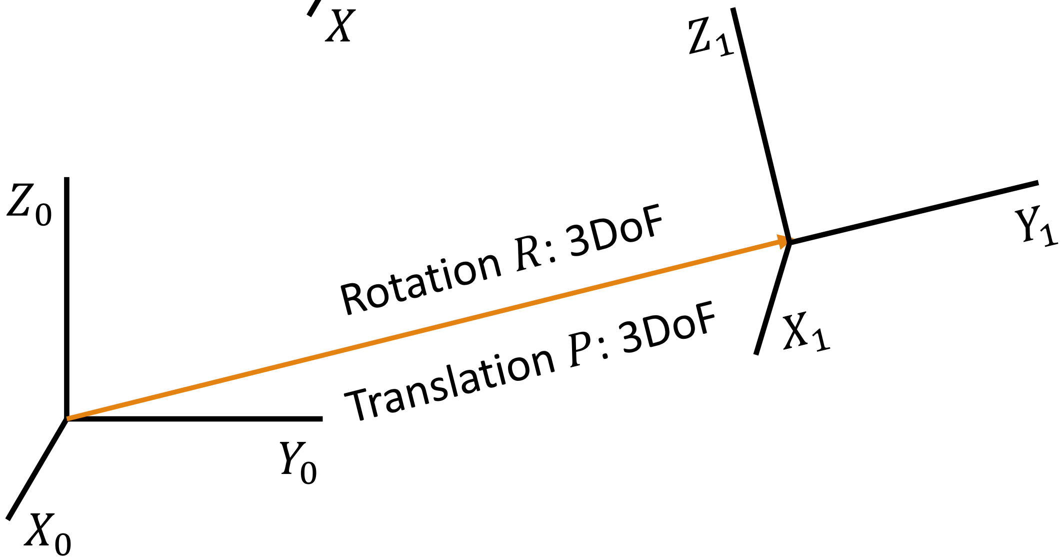
$$K = \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}$$



Elements w/ dimensions (objects):

- Definition of relative pose:

$${}^{0}_{1}T = \begin{bmatrix} & {}^{0}_{1}R & & P \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

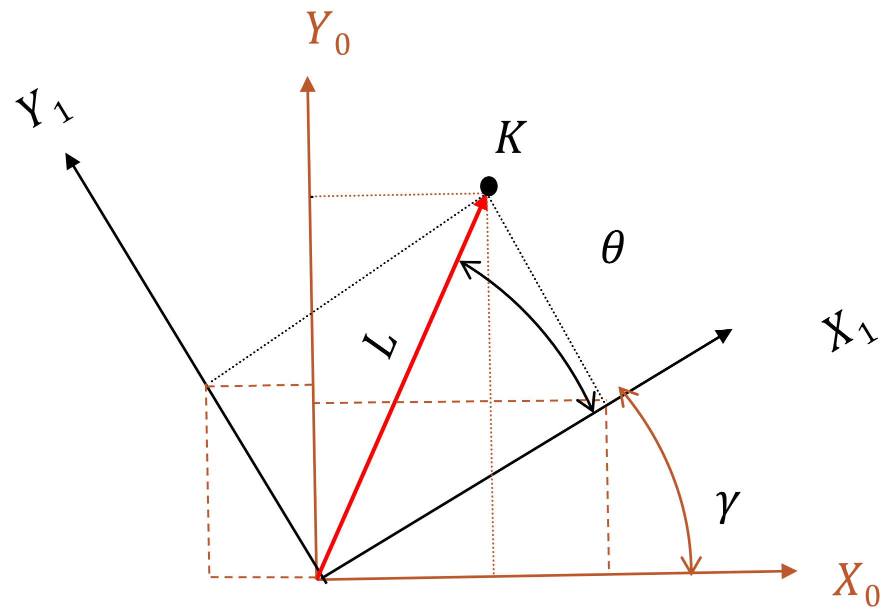


Frames of reference:: How?

Rotation Matrix: 2D

- Knowing the location of K w.r.t. frame 1: $[L \text{ and } \theta]$
- Knowing the rotation of frame 1 w.r.t. frame 0: $[\gamma]$
- Find the location of K w.r.t. frame 0

$$x_1 = L\cos(\theta)$$
$$y_1 = L\,sin(\theta)$$



$$x_0 = x_1 \cdot \cos(\gamma) - y_1 \cdot \sin(\gamma)$$
$$y_0 = x_1 \cdot \sin(\gamma) + y_1 \cdot \cos(\gamma)$$

$$\Rightarrow \begin{bmatrix} x_0 \\ y_0 \end{bmatrix} = \begin{bmatrix} \cos(\gamma) & -\sin(\gamma) \\ \sin(\gamma) & \cos(\gamma) \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ y_1 \end{bmatrix}$$

$$\Rightarrow \begin{bmatrix} x_0 \\ y_0 \end{bmatrix} = {}^0_1R \cdot \begin{bmatrix} x_1 \\ y_1 \end{bmatrix}$$

${}^0_1R$: 2D Rotation matrix from frame 1 to frame 0

Frames of reference:: How?

Rotation Matrix: 2D

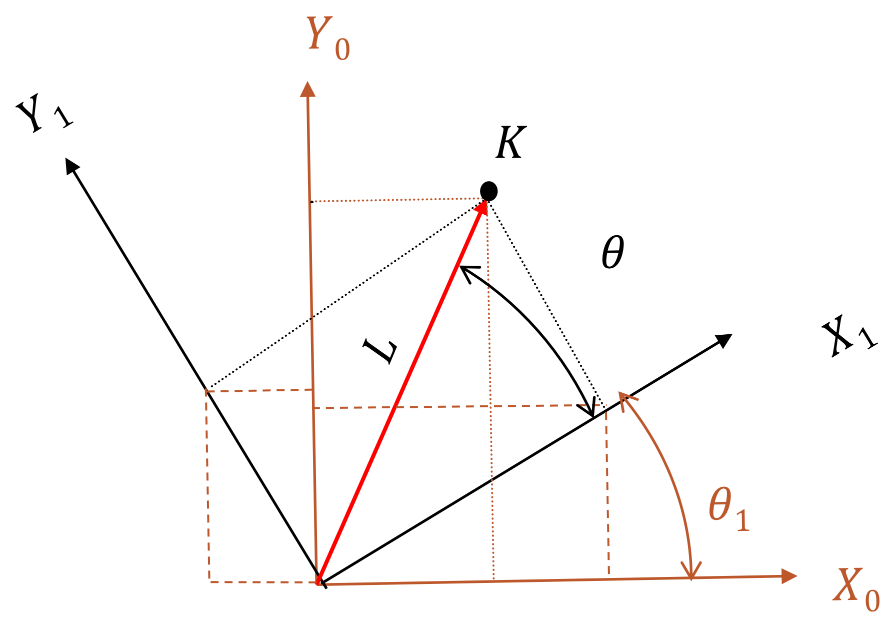
- Rotation around Z

$$ {}^0_1R = \begin{bmatrix} \cos(\gamma) & -\sin(\gamma) \\ \sin(\gamma) & \cos(\gamma) \end{bmatrix} $$

Rotation Matrix: 3D

- Rotation around Z

$$ {}^0_1R = \begin{bmatrix} \cos(\gamma) & -\sin(\gamma) & 0 \\ \sin(\gamma) & \cos(\gamma) & 0 \\ 0 & 0 & 1 \end{bmatrix} $$



$$ Z_1 = Z_0 \quad \rightarrow \quad \begin{bmatrix} X_0 \\ Y_0 \\ Z_0 \end{bmatrix} = \begin{bmatrix} \cos(\gamma) & -\sin(\gamma) & 0 \\ \sin(\gamma) & \cos(\gamma) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X_1 \\ Y_1 \\ Z_1 \end{bmatrix} $$

Frames of reference:: How?

Similarly for the rest of the axes' rotation.

- Rotation around Z

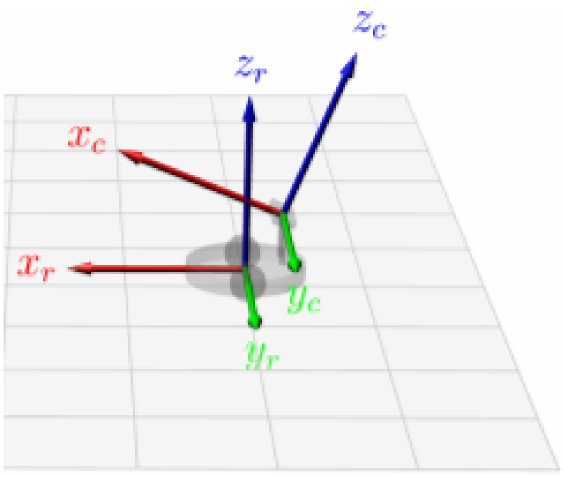
$$ {}^{B}_{A}R_Z = \begin{bmatrix} cos(\alpha) & -sin(\alpha) & 0 \\ sin(\alpha) & cos(\alpha) & 0 \\ 0 & 0 & 1 \end{bmatrix} $$

- Rotation around X

$$ {}^{B}_{A}R_X = \begin{bmatrix} 1 & 0 & 0 \\ 0 & cos(\gamma) & -sin(\gamma) \\ 0 & sin(\gamma) & cos(\gamma) \end{bmatrix} $$

- Rotation around Y

$$ {}^{B}_{A}R_Y = \begin{bmatrix} cos(\beta) & 0 & sin(\beta) \\ 0 & 1 & 0 \\ -sin(\beta) & 0 & cos(\beta) \end{bmatrix} $$



- Rotation around all 3 axes

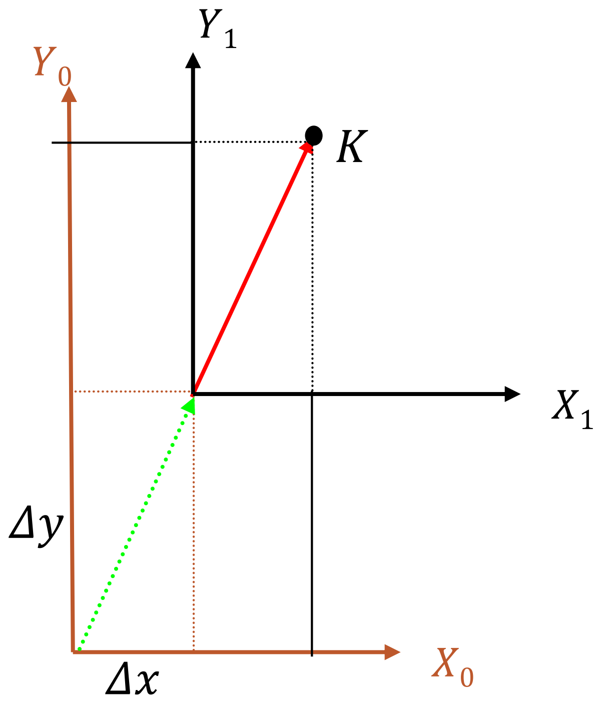
$$ {}^{B}_{A}R = R_Z(\alpha)R_Y(\beta)R_X(\gamma) $$

Frames of reference:: How?

Translation Matrix P: 2D

$$x_0 = x_1 + \Delta x \qquad \Rightarrow \quad \begin{bmatrix} x_0 \\ y_0 \end{bmatrix} = \begin{bmatrix} x_1 \\ y_1 \end{bmatrix} + \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix}$$
$$y_0 = y_1 + \Delta y$$

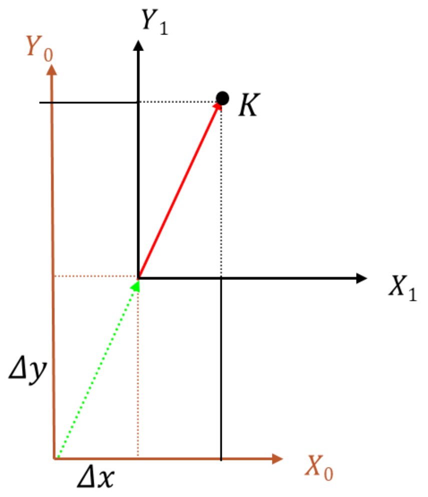
$$\Rightarrow \quad \begin{bmatrix} x_0 \\ y_0 \end{bmatrix} = \begin{bmatrix} x_1 \\ y_1 \end{bmatrix} + P$$



Frames of reference:: How?

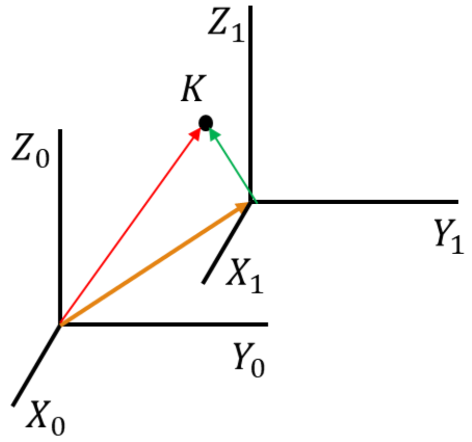
Translation Matrix P: 2D

$$\begin{bmatrix} x_0 \\ y_0 \end{bmatrix} = \begin{bmatrix} x_1 \\ y_1 \end{bmatrix} + \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} \Rightarrow \begin{bmatrix} x_0 \\ y_0 \end{bmatrix} = \begin{bmatrix} x_1 \\ y_1 \end{bmatrix} + P$$



Translation Matrix P: 3D

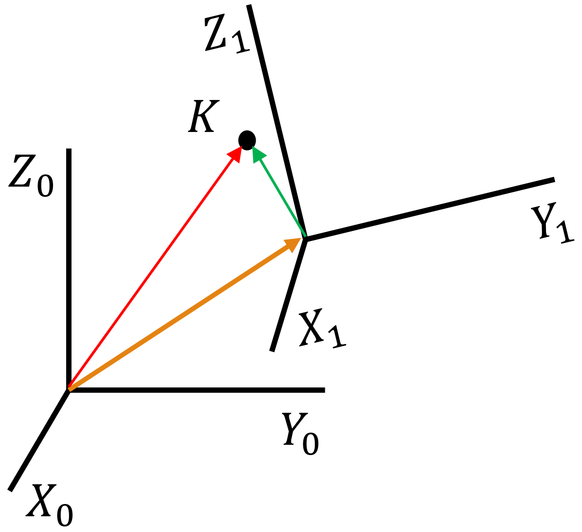
$$\begin{bmatrix} x_0 \\ y_0 \\ z_0 \end{bmatrix} = \begin{bmatrix} x_1 \\ y_1 \\ z_1 \end{bmatrix} + \begin{bmatrix} \Delta x \\ \Delta y \\ \Delta z \end{bmatrix} \Rightarrow \begin{bmatrix} x_0 \\ y_0 \\ z_0 \end{bmatrix} = \begin{bmatrix} x_1 \\ y_1 \\ z_1 \end{bmatrix} + P$$



Frames of reference:: How?

Combining R and P $\rightarrow$ Transformation Matrix T

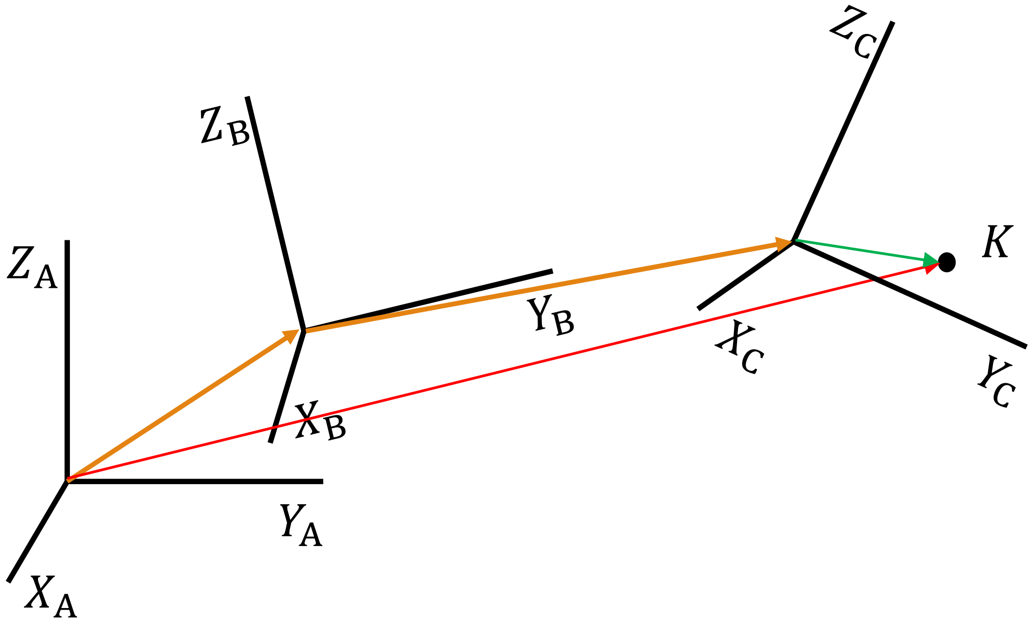
$$\left.\begin{array}{l} {}^{0}K = {}^{0}_{1}\mathrm{R}\ {}^{1}K + P \\ {}^{0}_{1}\mathrm{T} = \left[\begin{array}{ccc|c} & {}^{0}_{1}\mathrm{R} & & P \\ \hline 0 & 0 & 0 & 1 \end{array}\right] \end{array}\right\} \begin{bmatrix} {}^{0}K \\ 1 \end{bmatrix} = \left[\begin{array}{ccc|c} & {}^{0}_{1}\mathrm{R} & & P \\ \hline 0 & 0 & 0 & 1 \end{array}\right] \begin{bmatrix} {}^{1}K \\ 1 \end{bmatrix}$$



$${}^{0}K = {}^{0}_{1}\mathrm{T}\ {}^{1}K$$

Frames of reference:: Kinematic Chain

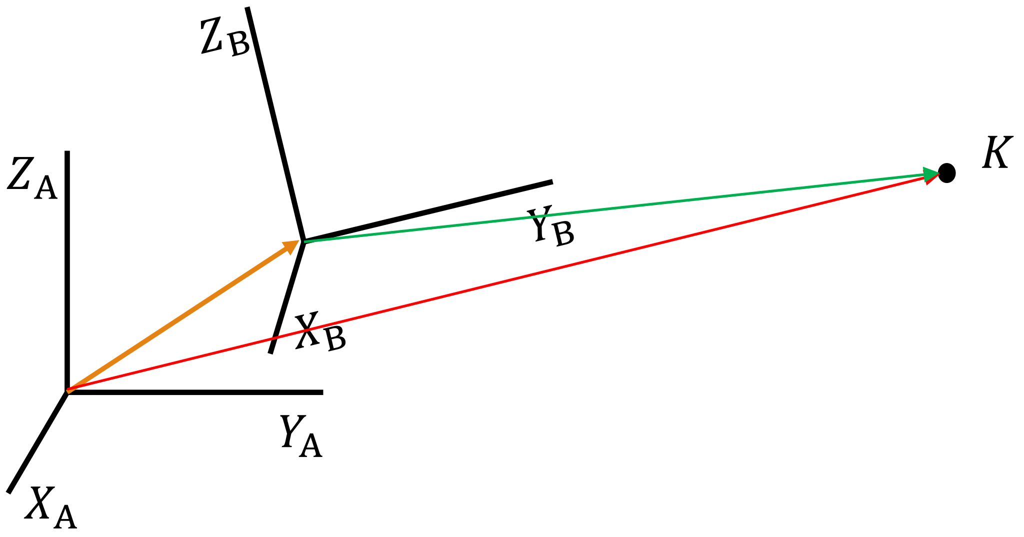
- Gradual description of a point from one frame of reference to its previous one



$$ {}^{B}K = {}^{B}_{C}T \, {}^{C}K $$

Frames of reference:: Kinematic Chain

- Gradual description of a point from one frame of reference to its previous one



$$^{B}K = {}^{B}_{C}T\, {}^{C}K$$

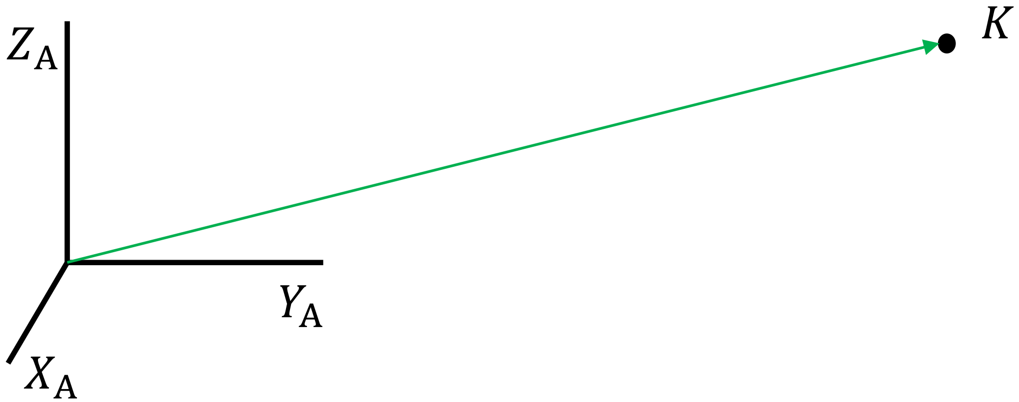
$$^{A}K = {}^{A}_{B}T\, {}^{B}K$$

Frames of reference:: Kinematic Chain

- Gradual description of a point from one frame of reference to its previous one

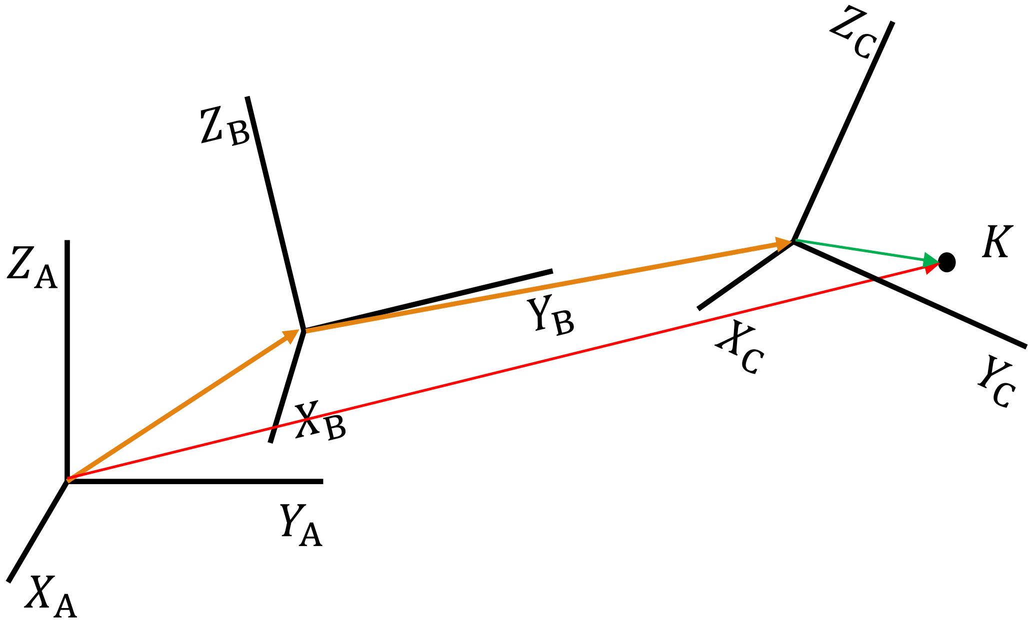
$$^{B}K = {}^{B}_{C}T\ {}^{C}K$$

$$^{A}K = {}^{A}_{B}T\ {}^{B}K$$



Frames of reference:: Kinematic Chain

- Gradual description of a point from one frame of reference to its previous one



$${}^{B}K = {}^{B}_{C}T\,{}^{C}K$$

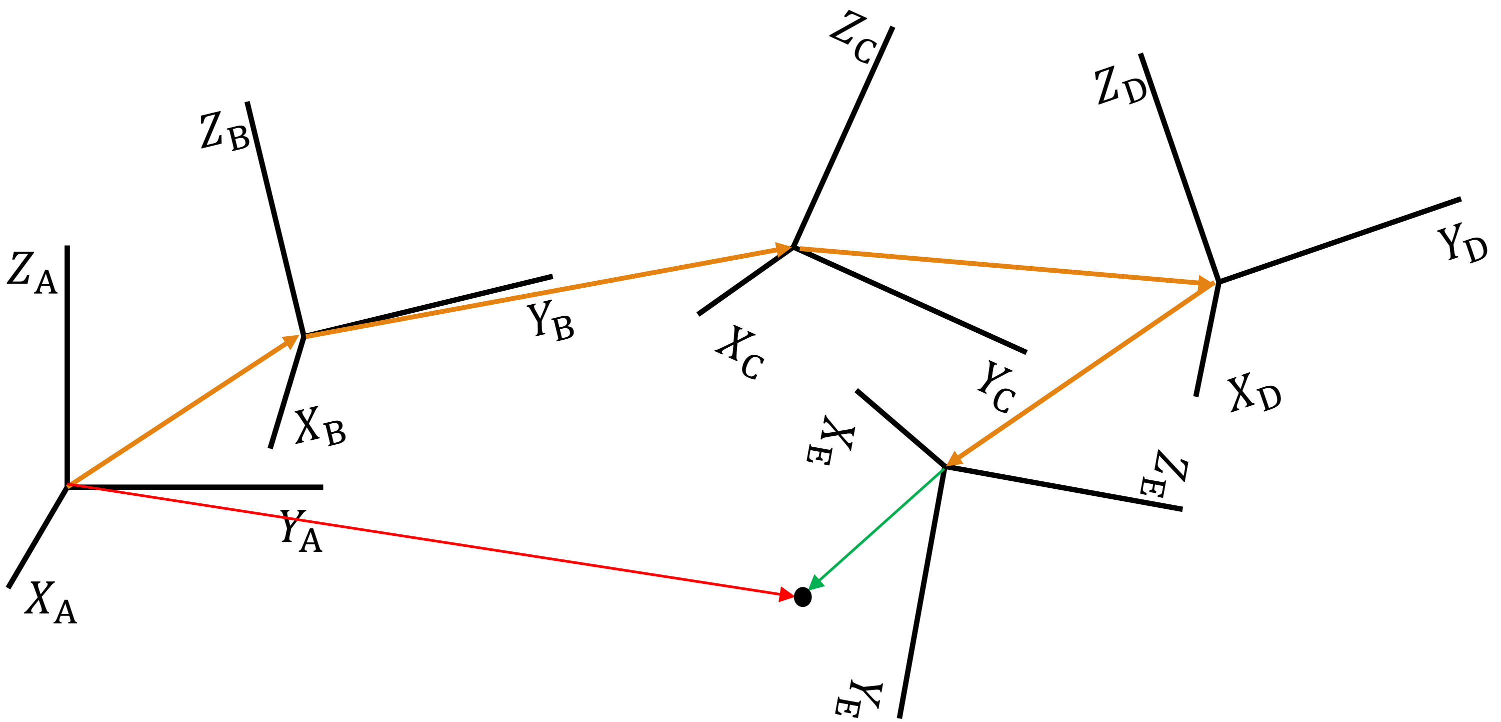
$${}^{A}K = {}^{A}_{B}T\,{}^{B}K$$

$${}^{A}K = {}^{A}_{B}T\,{}^{B}_{C}T\,{}^{C}K$$

$${}^{A}_{C}T = {}^{A}_{B}T\,{}^{B}_{C}T$$

Frames of reference:: Kinematic Chain

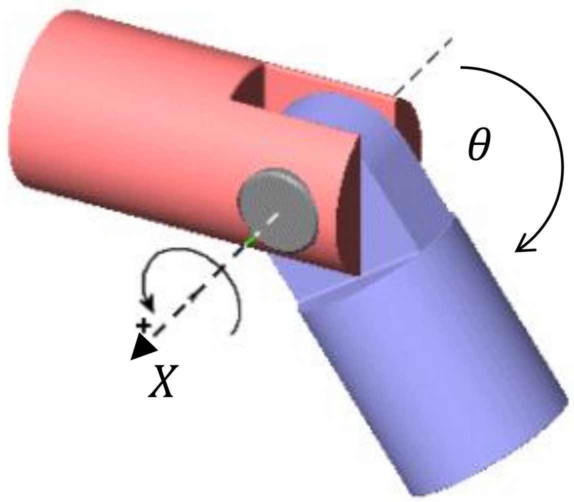
- Gradual description of a point from one frame of reference to its previous one



$$ {}^{A}_{E}T = {}^{A}_{B}T \; {}^{B}_{C}T \; {}^{C}_{D}T \; {}^{D}_{E}T $$

Frames of reference:: Applications

Actuators: Every joint can be defined by a Transformation Matrix



Revolute joint

Rotation around X axis with negative rotation direction

$$R = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\gamma) & -\sin(\gamma) \\ 0 & \sin(\gamma) & \cos(\gamma) \end{bmatrix}$$

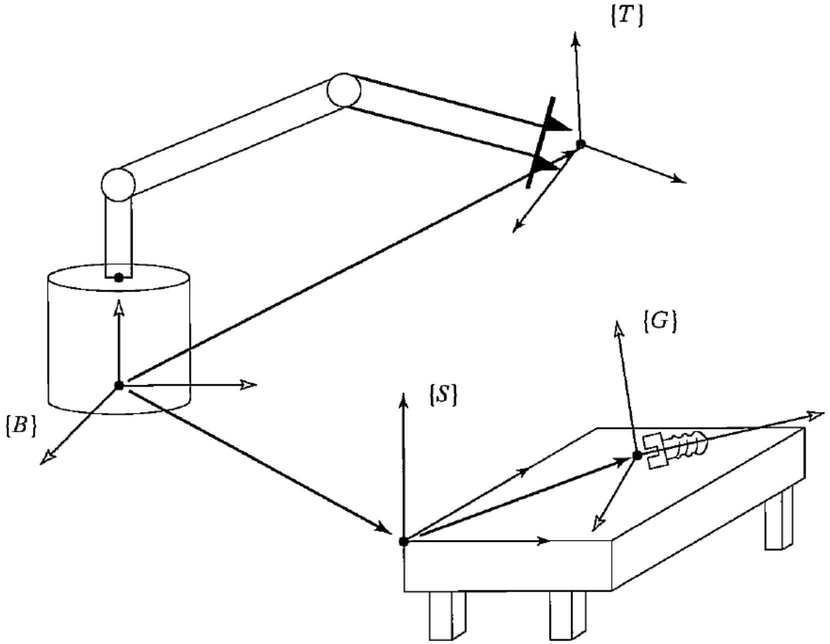
$$T = \left[\begin{array}{ccc|c} & R & & P \\ \hline 0 & 0 & 0 & 1 \end{array}\right]$$

$$T = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & cos(-\theta) & -sin(-\theta) & 0 \\ 0 & sin(-\theta) & cos(-\theta) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Frames of reference:: Applications

Actuators: Every joint can be defined by a Transformation Matrix

This way, the relative transformation between the end-effector and the target can be computed ⇒ Manipulation



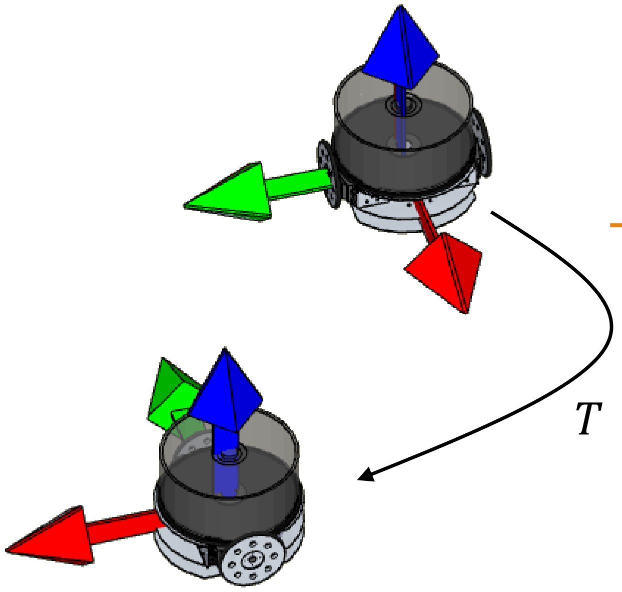
Unknowns: ${}^{T}_{G}T$

Knowns: ${}^{B}_{T}T,\ {}^{B}_{S}T,\ {}^{S}_{G}T$

Solution: ${}^{T}_{G}T = \left[{}^{B}_{T}T\right]^{-1} {}^{B}_{S}T\, {}^{S}_{G}T$

Frames of reference:: Applications

AGVs: Every robot movement can be defined by a Transformation Matrix



Rotation around Z axis with negative rotation direction and Translation on X and Y

$$R = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\gamma) & -\sin(\gamma) \\ 0 & \sin(\gamma) & \cos(\gamma) \end{bmatrix}$$

$$P = \begin{bmatrix} \Delta x \\ \Delta y \\ 0 \end{bmatrix}$$

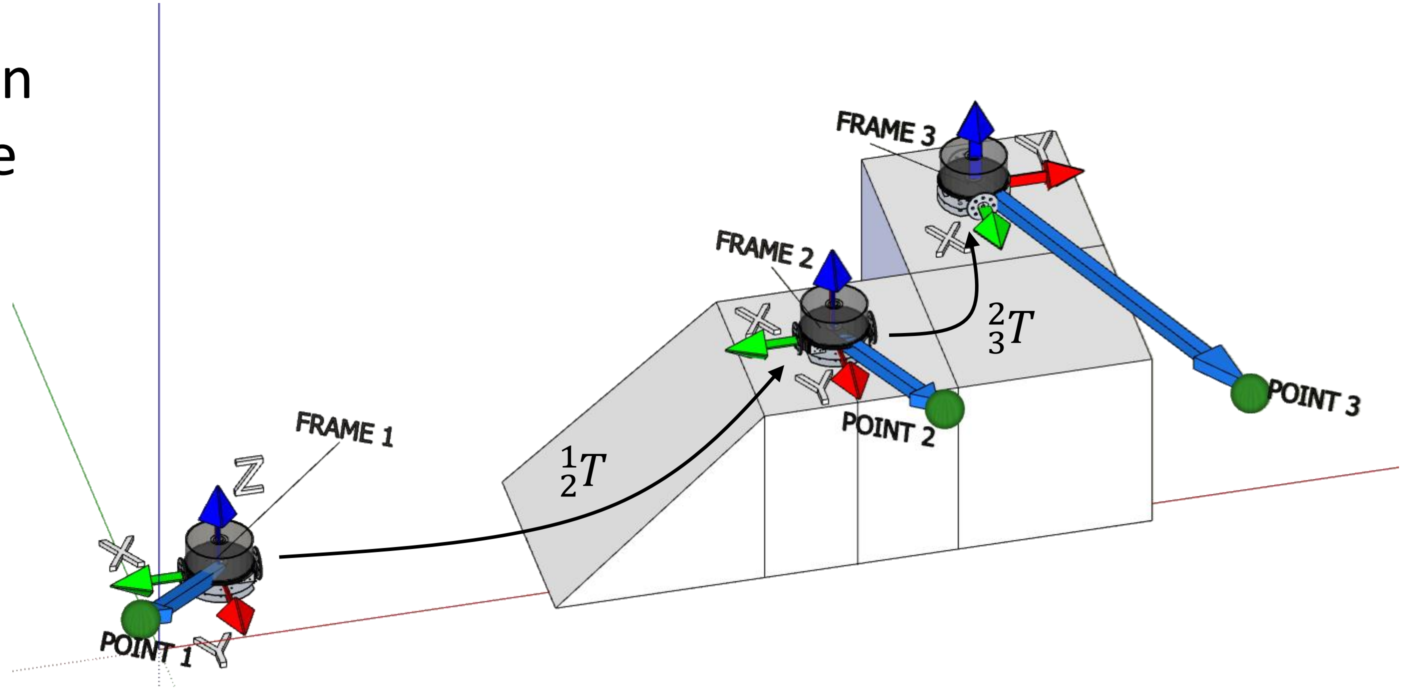
$$T = \left[\begin{array}{ccc|c} & R & & P \\ \hline 0 & 0 & 0 & 1 \end{array}\right]$$

$$T = \begin{bmatrix} 1 & 0 & 0 & \Delta x \\ 0 & cos(-\theta) & -sin(-\theta) & \Delta y \\ 0 & sin(-\theta) & cos(-\theta) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Frames of reference:: Applications

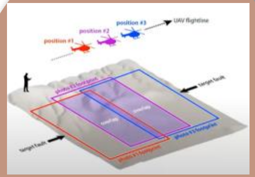
AGVs: Every robot movement can be defined by a Transformation Matrix

This way, the observed points can be projected to a common frame of reference ⇒ Map



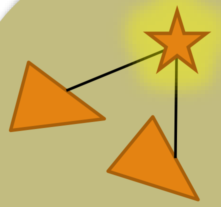
Localization and Mapping for Autonomous Mobile Systems

Evolution of SLAM



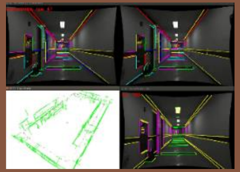
Photogrammetry
(expensive and manual)

Structure from Motion
(typically off-line)

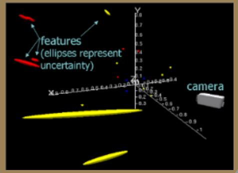


Localization or Mapping
(treated separately)

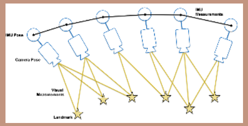
SLAM
(w/o cameras)



Visual SLAM
(Cameras as the main sensor)

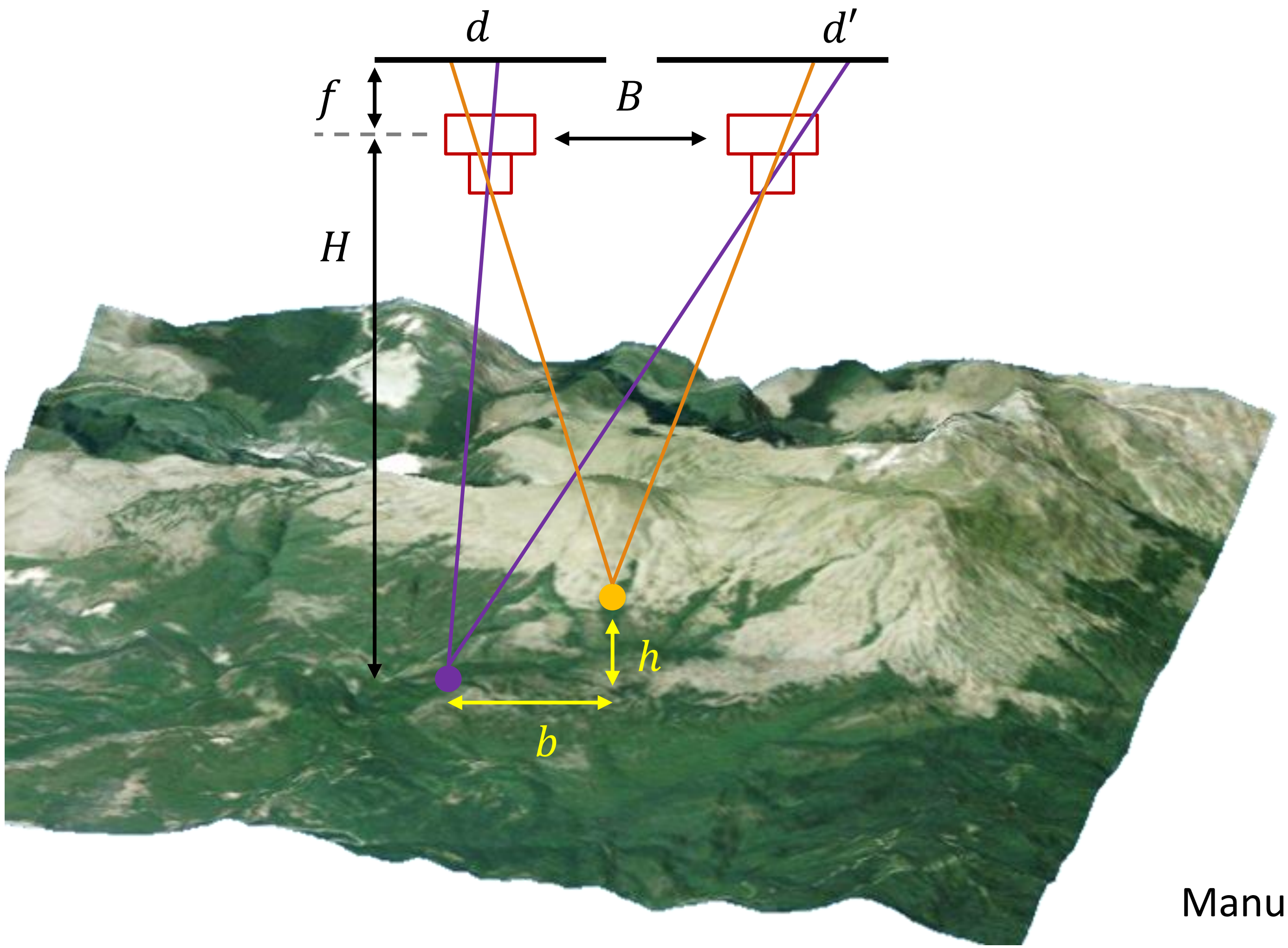


Monocular SLAM
(Single camera instead of stereo)



Visual-Inertial SLAM
(Camera + IMU)

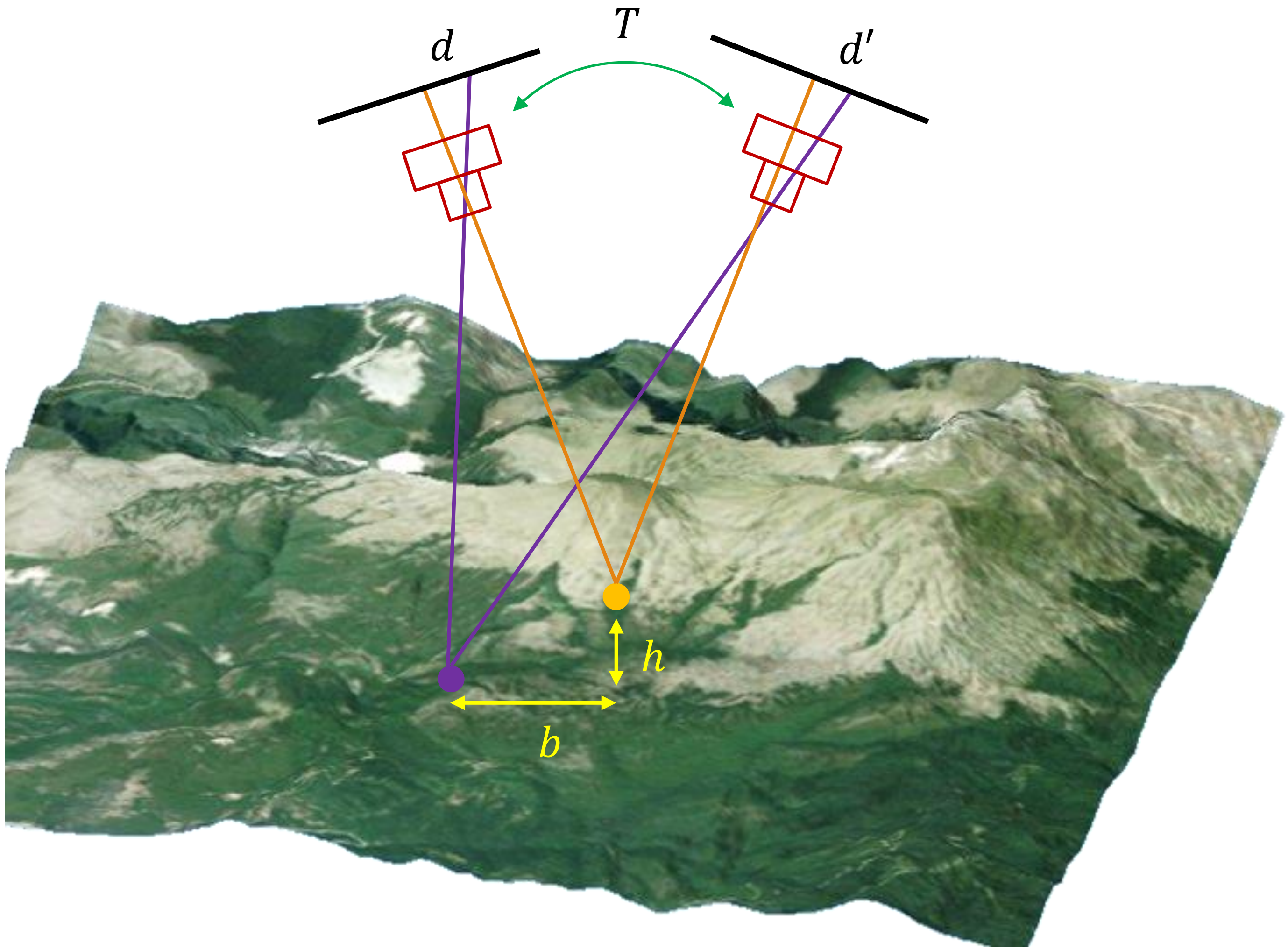
It all started with stereo-photogrammetry



- Required: H, f, B
- Measured: d, d'
- Computed: b, h

Manual measurements: Mirror Stereoscope

Structure from Motion – SfM



- Required: Multiple points associations among different frames (automated)
- Measured: d, d' (automated)
- Computed: T, b, h

Structure from Motion – SfM

Falls into the "Shape from X" problem

Methods:

- Stereo
- Shading
- Photometric Stereo
- Texture
- Contours
- Silhouettes
- Motion

SLAM and SfM

SLAM is using tools from SfM

- SfM is traditionally performed offline
  - Recovery of 3D shape from 2D images
  - Depending on the scale, reconstruction can take hours or days
  - It is typically performed on high-performance computers
  - Google Maps and Google Street View were built using SfM

- SLAM mostly refers to online applications
  - Mapping and localization on-the-fly
  - Real-time
  - Low-power sensors (e.g., a single RGB camera and an IMU)
  - It is typically performed on-board using low-power processing units

Structure from Motion – SfM

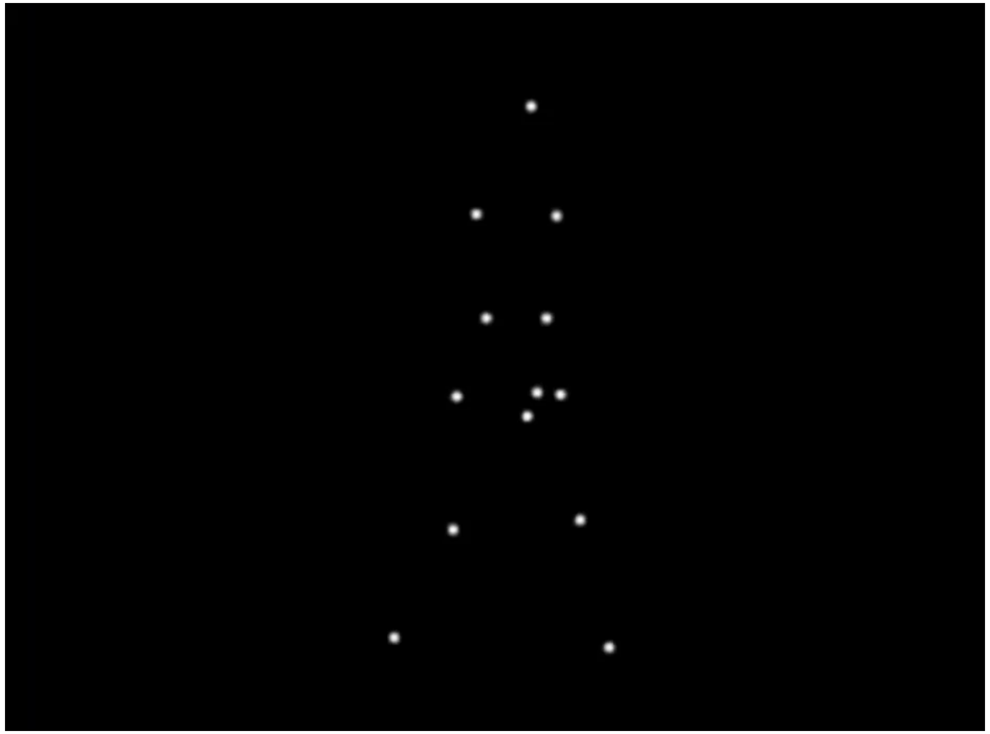
Falls into the "Shape from X" problem

Methods:

- Stereo
- Shading
- Photometric Stereo
- Texture
- Contours
- Silhouettes
- Motion

Structure from Motion – SfM

Shape from X - Shape from Motion



- Humans are able to recover 3D from motion