Localization & Odometry

EEL 4930/5934: Autonomous Robots

Spring 2023

Md Jahidul Islam

Lecture 7

ECE | Florida
Electrical & Computer Engineering

UF | UNIVERSITY of FLORIDA

Robot Localization

⇒ The "*Where Am I?*" problem

Estimating robot pose in a given map
(or global coordinates)

Types:

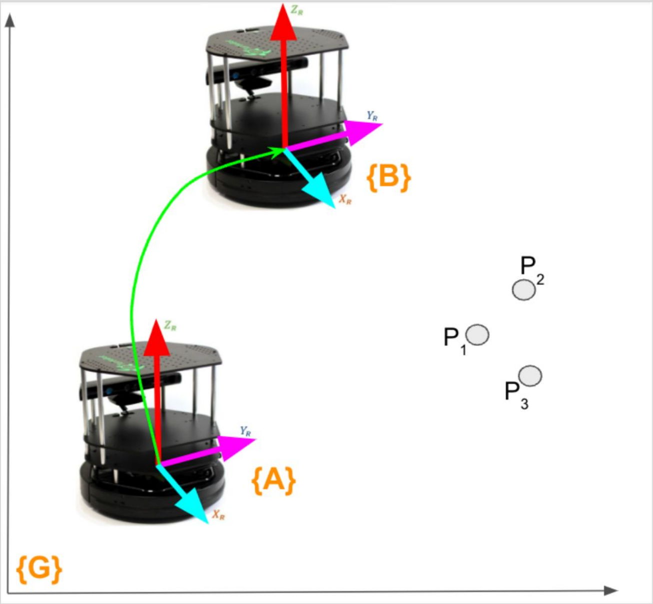
- Relative vs absolute localization
- Static map-based localization
- Dynamic environment without a map

Creating map on the go:

- SLAM: Simultaneous localization & mapping
- Popular SLAM literature
  - Visual SLAM
  - LiDAR-based SLAM
  - Laser-based SLAM (mainly 2D robots)

awabot

Localization in 2D from Landmarks



Known:

$$^GP_1 = \begin{bmatrix} ^Gx_1 \\ ^Gy_1 \\ ^Gz_1 \end{bmatrix},\ ^GP_2 = \begin{bmatrix} ^Gx_2 \\ ^Gy_2 \\ ^Gz_2 \end{bmatrix},\ \text{and}\ ^GP_3 = \begin{bmatrix} ^Gx_3 \\ ^Gy_3 \\ ^Gz_3 \end{bmatrix}$$

$$^AP_1 = \begin{bmatrix} ^Ax_1 \\ ^Ay_1 \\ ^Az_1 \end{bmatrix},\ ^AP_2 = \begin{bmatrix} ^Ax_2 \\ ^Ay_2 \\ ^Az_2 \end{bmatrix},\ \text{and}\ ^AP_3 = \begin{bmatrix} ^Ax_3 \\ ^Ay_3 \\ ^Az_3 \end{bmatrix}$$

Need to **localize** robot at {A}:

Find $^G\mathbf{P}_A$ and ϴ

$$^GP_1 = {}^GP_A + {}^G_AR \cdot {}^AP_1$$
$$^GP_2 = {}^GP_A + {}^G_AR \cdot {}^AP_2$$
$$^GP_3 = {}^GP_A + {}^G_AR \cdot {}^AP_3$$

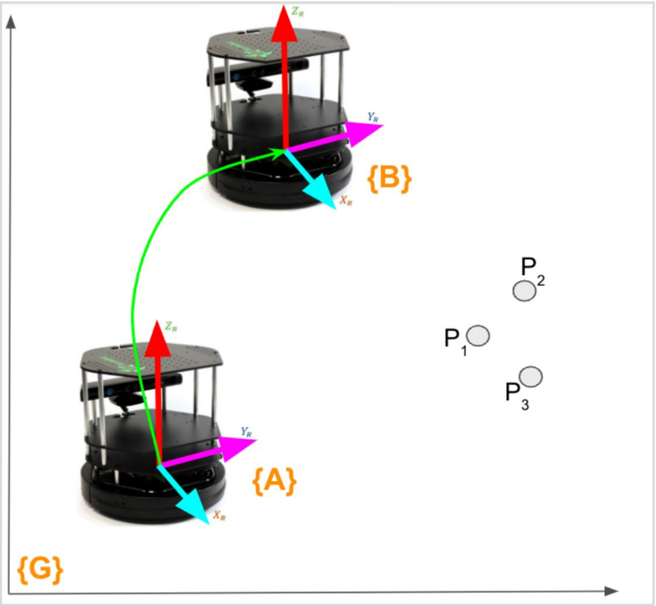
→

$$^GP_1 - {}^GP_2 = {}^G_AR \cdot ({}^AP_1 - {}^AP_2)$$
$$^GP_2 - {}^GP_3 = {}^G_AR \cdot ({}^AP_2 - {}^AP_3)$$
$$^GP_3 - {}^GP_1 = {}^G_AR \cdot ({}^AP_3 - {}^AP_1)$$

$$^G_A\mathbf{T} = \left[\begin{array}{c|c} ^G_AR & ^GP_A \\ \hline 0\ 0\ 0 & 1 \end{array}\right] = \begin{bmatrix} \cos\theta_a & -\sin\theta_a & 0 & x_a \\ \sin\theta_a & \cos\theta_a & 0 & y_a \\ 0 & 0 & 1 & z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$^GP_1 + {}^GP_2 = 2\,{}^GP_A + {}^G_AR \cdot ({}^AP_1 + {}^AP_2)$$
$$^GP_A = \frac{1}{2}\left({}^GP_1 + {}^GP_2 - {}^G_AR \cdot ({}^AP_1 + {}^AP_2)\right)$$

Solving For ϴ



$$
\begin{aligned}
{}^{G}P_1 &= {}^{G}P_A + {}^{G}_{A}R \cdot {}^{A}P_1 \\
{}^{G}P_2 &= {}^{G}P_A + {}^{G}_{A}R \cdot {}^{A}P_2 \\
{}^{G}P_3 &= {}^{G}P_A + {}^{G}_{A}R \cdot {}^{A}P_3
\end{aligned}
\;\longrightarrow\;
\begin{aligned}
{}^{G}P_1 - {}^{G}P_2 &= {}^{G}_{A}R \cdot ({}^{A}P_1 - {}^{A}P_2) \\
{}^{G}P_2 - {}^{G}P_3 &= {}^{G}_{A}R \cdot ({}^{A}P_2 - {}^{A}P_3) \\
{}^{G}P_3 - {}^{G}P_1 &= {}^{G}_{A}R \cdot ({}^{A}P_3 - {}^{A}P_1)
\end{aligned}
$$

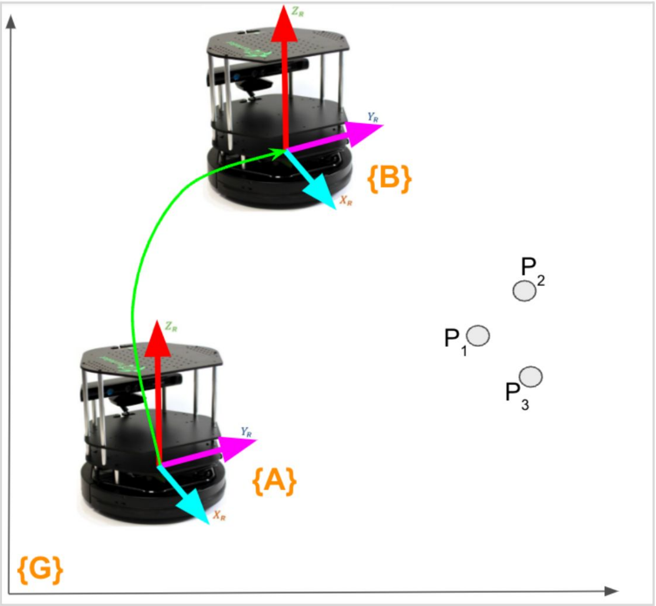
$$
\begin{bmatrix} {}^{G}x_1 - {}^{G}x_2 \\ {}^{G}y_1 - {}^{G}y_2 \end{bmatrix} = \begin{bmatrix} \cos\theta_a & -\sin\theta_a \\ \sin\theta_a & \cos\theta_a \end{bmatrix} \cdot \begin{bmatrix} {}^{A}x_1 - {}^{A}x_2 \\ {}^{A}y_1 - {}^{A}y_2 \end{bmatrix}
$$

$$
\begin{bmatrix} {}^{G}x_2 - {}^{G}x_3 \\ {}^{G}y_2 - {}^{G}y_3 \end{bmatrix} = \begin{bmatrix} \cos\theta_a & -\sin\theta_a \\ \sin\theta_a & \cos\theta_a \end{bmatrix} \cdot \begin{bmatrix} {}^{A}x_2 - {}^{A}x_3 \\ {}^{A}y_2 - {}^{A}y_3 \end{bmatrix}
$$

$$
\begin{bmatrix} {}^{G}x_3 - {}^{G}x_1 \\ {}^{G}y_3 - {}^{G}y_1 \end{bmatrix} = \begin{bmatrix} \cos\theta_a & -\sin\theta_a \\ \sin\theta_a & \cos\theta_a \end{bmatrix} \cdot \begin{bmatrix} {}^{A}x_3 - {}^{A}x_1 \\ {}^{A}y_3 - {}^{A}y_1 \end{bmatrix}
$$

$$
{}^{G}_{A}\mathbf{T} = \left[\begin{array}{ccc|c} & {}^{G}_{A}R & & {}^{G}P_A \\ \hline 0 & 0 & 0 & 1 \end{array}\right] = \begin{bmatrix} \cos\theta_a & -\sin\theta_a & 0 & x_a \\ \sin\theta_a & \cos\theta_a & 0 & y_a \\ 0 & 0 & 1 & z \\ 0 & 0 & 0 & 1 \end{bmatrix}
$$

Solving For Θ



$$\begin{bmatrix} {}^Gx_1 - {}^Gx_2 \\ {}^Gy_1 - {}^Gy_2 \end{bmatrix} = \begin{bmatrix} \cos\theta_a & -\sin\theta_a \\ \sin\theta_a & \cos\theta_a \end{bmatrix} \cdot \begin{bmatrix} {}^Ax_1 - {}^Ax_2 \\ {}^Ay_1 - {}^Ay_2 \end{bmatrix}$$

$$\begin{bmatrix} {}^G\Delta x1 \\ {}^G\Delta y1 \end{bmatrix} = \begin{bmatrix} \cos\theta_a & -\sin\theta_a \\ \sin\theta_a & \cos\theta_a \end{bmatrix} \cdot \begin{bmatrix} {}^A\Delta x1 \\ {}^A\Delta y1 \end{bmatrix}$$

$$= \begin{bmatrix} {}^A\Delta x1 & -{}^A\Delta y1 \\ {}^A\Delta y1 & {}^A\Delta x1 \end{bmatrix} \cdot \begin{bmatrix} \cos\theta_a \\ \sin\theta_a \end{bmatrix}$$

$$= A \cdot \begin{bmatrix} \cos\theta_a \\ \sin\theta_a \end{bmatrix}$$

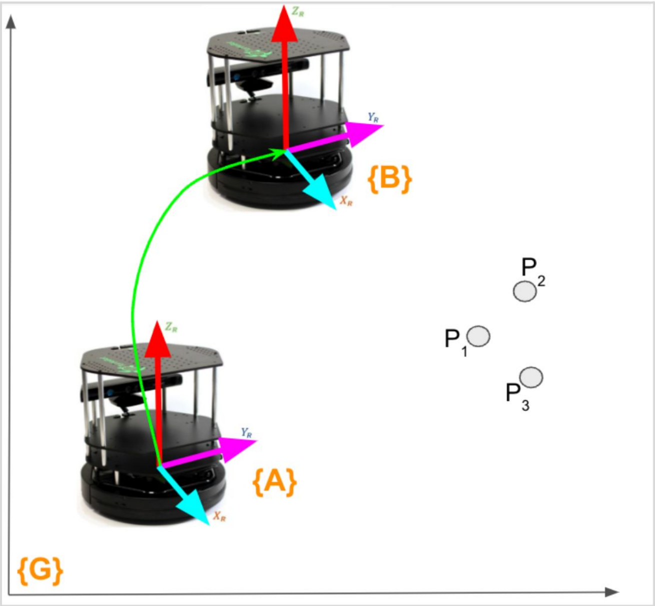
$$\begin{bmatrix} \cos\theta_a \\ \sin\theta_a \end{bmatrix} = A^{-1} \cdot \begin{bmatrix} {}^G\Delta x1 \\ {}^G\Delta y1 \end{bmatrix} \quad \text{if } \det(A) = ({}^A\Delta x1)^2 + ({}^A\Delta y1)^2 \neq 0$$

$$\theta = arctan2(\sin\theta_a, \cos\theta_a)$$

What happens in |A| = 0 ?

$${}^G_A\mathbf{T} = \left[\begin{array}{c:c} {}^G_AR & {}^GP_A \\ \hdashline 0\ 0\ 0 & 1 \end{array}\right] = \begin{bmatrix} \cos\theta_a & -\sin\theta_a & 0 & x_a \\ \sin\theta_a & \cos\theta_a & 0 & y_a \\ 0 & 0 & 1 & z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Solving For $^G P_A$



$$^G P_1 = {^G P_A} + {^G_A R} \cdot {^A P_1}$$
$$^G P_2 = {^G P_A} + {^G_A R} \cdot {^A P_2}$$
$$^G P_3 = {^G P_A} + {^G_A R} \cdot {^A P_3}$$

$$^G P_1 - {^G P_2} = {^G_A R} \cdot ({^A P_1} - {^A P_2})$$
$$^G P_2 - {^G P_3} = {^G_A R} \cdot ({^A P_2} - {^A P_3})$$
$$^G P_3 - {^G P_1} = {^G_A R} \cdot ({^A P_3} - {^A P_1})$$

Find ɵ as shown, for the case |A| ≠ 0
Or use LS formulation

$$\begin{bmatrix} {^G\Delta x1} \\ {^G\Delta y1} \end{bmatrix} = \begin{bmatrix} \cos\theta_a & -\sin\theta_a \\ \sin\theta_a & \cos\theta_a \end{bmatrix} \cdot \begin{bmatrix} {^A\Delta x1} \\ {^A\Delta y1} \end{bmatrix} = \begin{bmatrix} {^A\Delta x1} & -{^A\Delta y1} \\ {^A\Delta y1} & {^A\Delta x1} \end{bmatrix} \cdot \begin{bmatrix} \cos\theta_a \\ \sin\theta_a \end{bmatrix} = A \cdot \begin{bmatrix} \cos\theta_a \\ \sin\theta_a \end{bmatrix}$$

Then find $^G\mathbf{P}_A$

$$^G P_1 + {^G P_2} = 2\,{^G P_A} + {^G_A R} \cdot ({^A P_1} + {^A P_2})$$
$$^G P_A = \frac{1}{2}\left({^G P_1} + {^G P_2} - {^G_A R} \cdot ({^A P_1} + {^A P_2})\right)$$

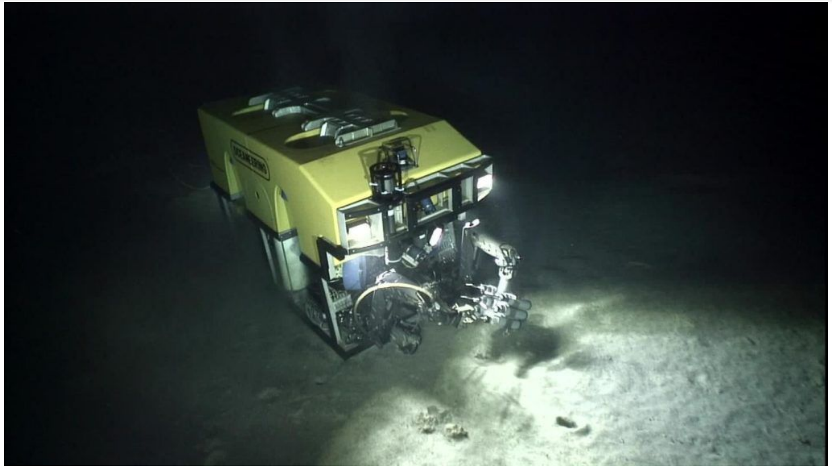
$$^G_A\mathbf{T} = \left[\begin{array}{c|c} {^G_A R} & {^G P_A} \\ \hline 0\ 0\ 0 & 1 \end{array}\right] = \begin{bmatrix} \cos\theta_a & -\sin\theta_a & 0 & x_a \\ \sin\theta_a & \cos\theta_a & 0 & y_a \\ 0 & 0 & 1 & z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Localization in 3D from Landmarks

Given: coordinates of 3 non-planar points

$$^{G}P_1 = \begin{bmatrix} ^{G}x_1 \\ ^{G}y_1 \\ ^{G}z_1 \end{bmatrix}, \; ^{G}P_2 = \begin{bmatrix} ^{G}x_2 \\ ^{G}y_2 \\ ^{G}z_2 \end{bmatrix}, \text{ and } ^{G}P_3 = \begin{bmatrix} ^{G}x_3 \\ ^{G}y_3 \\ ^{G}z_3 \end{bmatrix}$$

$$^{A}P_1 = \begin{bmatrix} ^{A}x_1 \\ ^{A}y_1 \\ ^{A}z_1 \end{bmatrix}, \; ^{A}P_2 = \begin{bmatrix} ^{A}x_2 \\ ^{A}y_2 \\ ^{A}z_2 \end{bmatrix}, \text{ and } ^{A}P_3 = \begin{bmatrix} ^{A}x_3 \\ ^{A}y_3 \\ ^{A}z_3 \end{bmatrix}$$

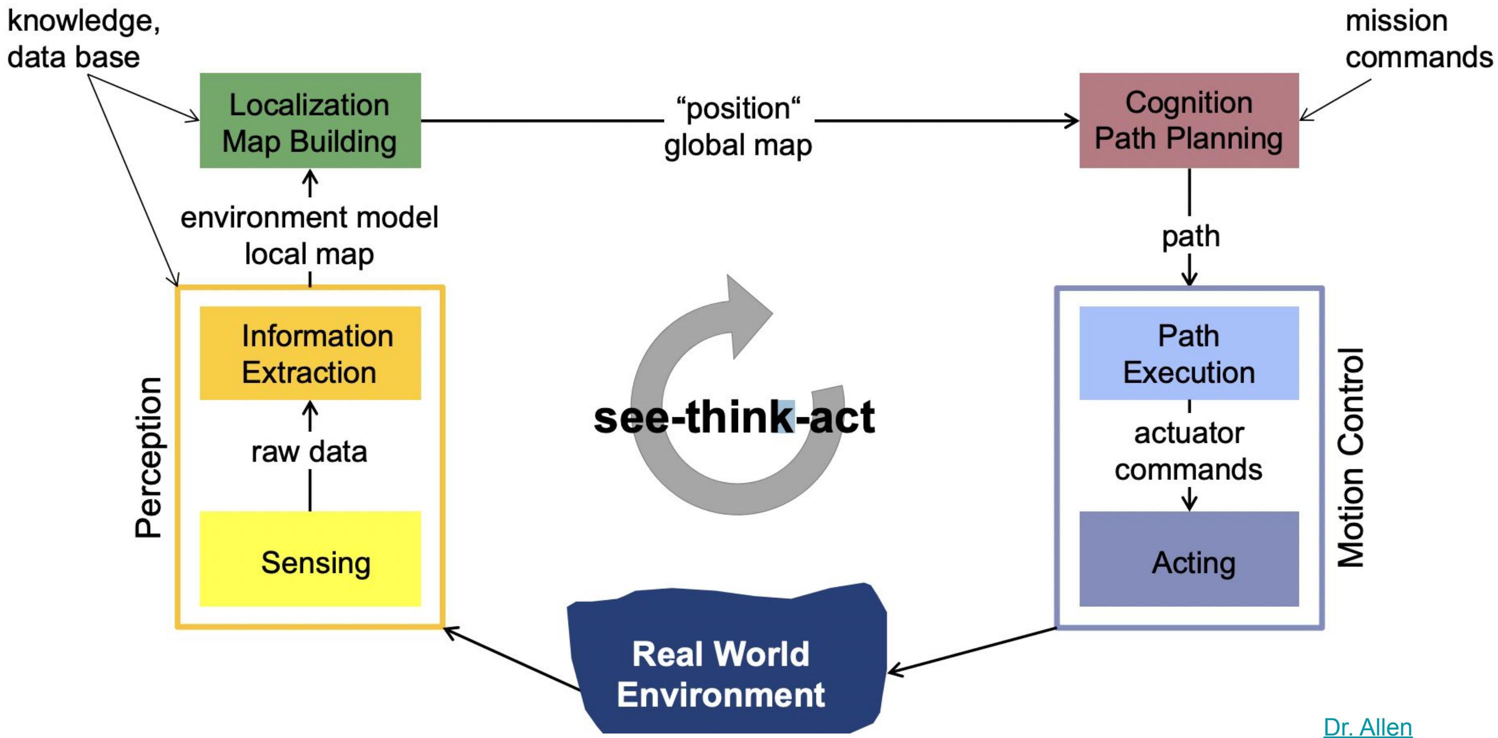


Need to **localize** robot at {A}: $^{G}_{A}\mathbf{T} = \left[\begin{array}{ccc|c} & ^{G}_{A}R & & ^{G}P_A \\ \hline 0 & 0 & 0 & 1 \end{array}\right]$

Homework 5A

So, find the position $^{G}\mathbf{P}_{A}$ and rotation matrix $^{G}\mathbf{R}_{A}$

Map-based Localization



Dr. Allen

Probabilistic Map-based Localization

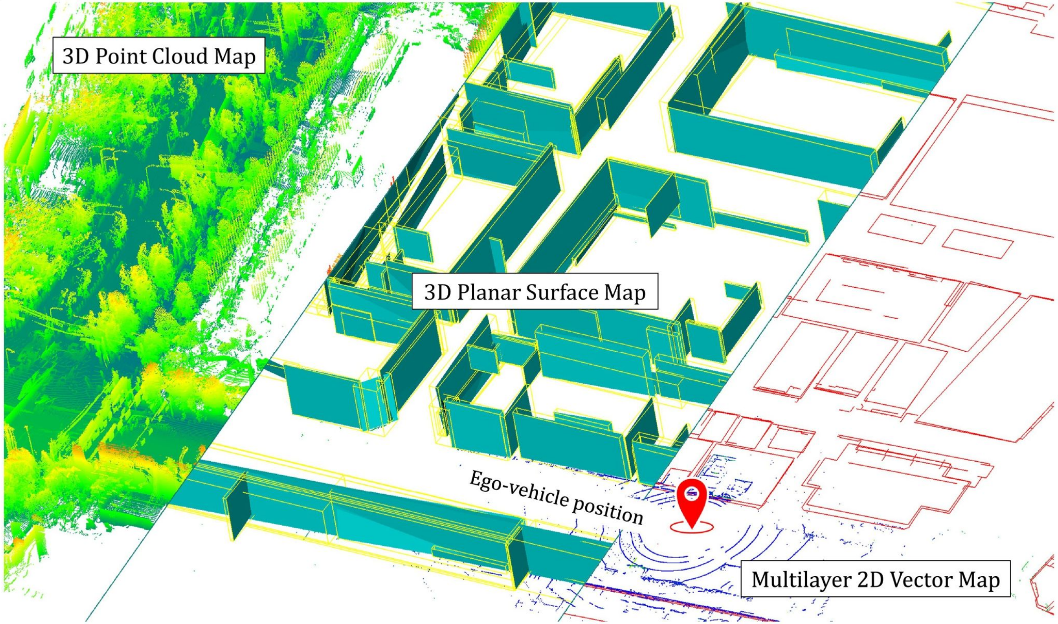
⇒ Estimating robot pose using the perceived sensory information given a global map

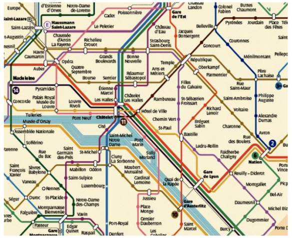
<u>Challenges</u>

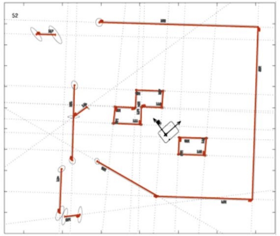
- Measurements and the map are inherently error prone
- Thus the robot has to deal with uncertain information

<u>Probabilistic map-base localization</u>

- The robot estimates the belief state about its position
- Through an ACT and SEE cycle







Ehsan et al

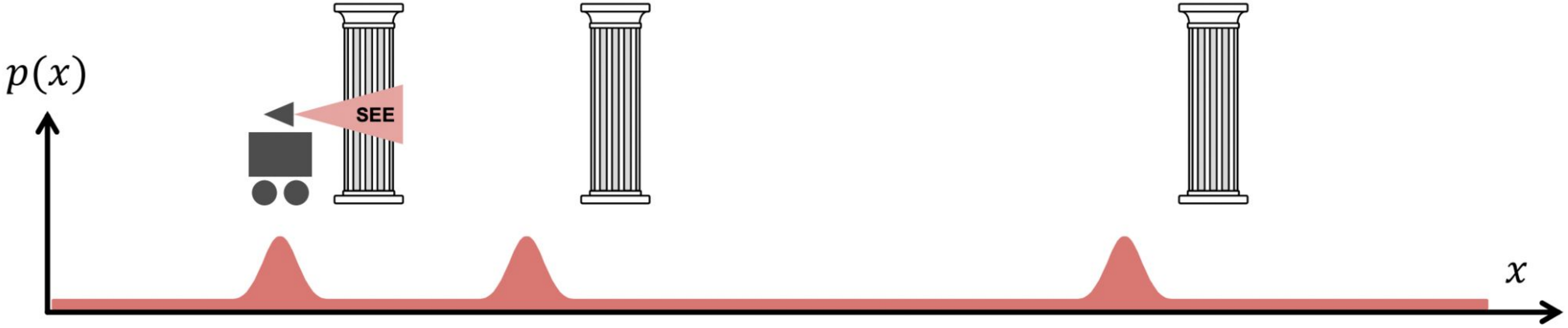
See – Act – Update

- Robot is placed somewhere in the environment → location unknown
- SEE: The robot queries its sensors → finds itself next to a pillar
- ACT: Robot moves one meter forward
  - motion estimated by wheel encoders
  - accumulation of uncertainty
- SEE: The robot queries its sensors again → finds itself next to a pillar
- Belief updates (information fusion)

Dr. Allen

See – Act – Update

- Robot is placed somewhere in the environment → location unknown
- SEE: The robot queries its sensors → finds itself next to a pillar
- ACT: Robot moves one meter forward
  - motion estimated by wheel encoders
  - accumulation of uncertainty
- SEE: The robot queries its sensors again → finds itself next to a pillar
- Belief updates (information fusion)

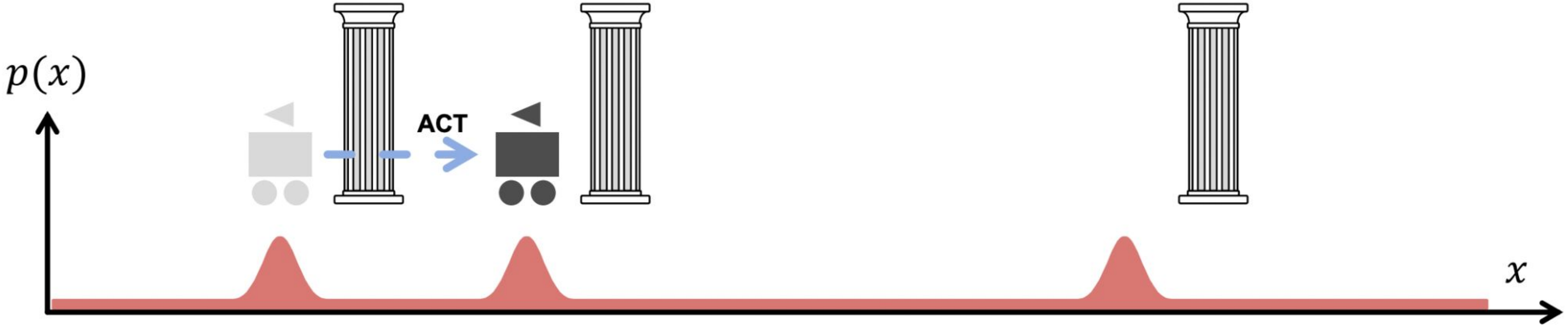


Dr. Allen

See – Act – Update

- Robot is placed somewhere in the environment → location unknown
- SEE: The robot queries its sensors → finds itself next to a pillar

- ACT: Robot moves one meter forward
  - motion estimated by wheel encoders
  - accumulation of uncertainty
- SEE: The robot queries its sensors again → finds itself next to a pillar
- Belief updates (information fusion)

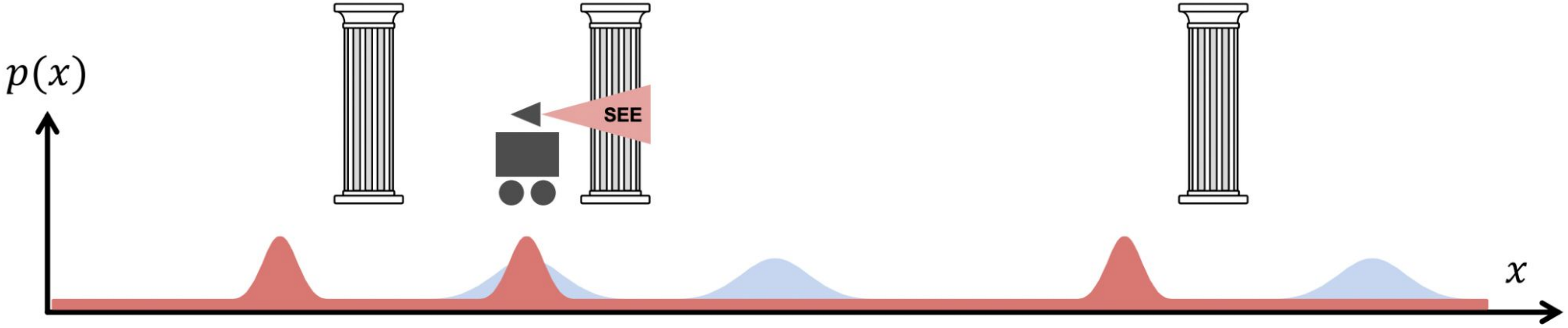


Dr. Allen

See – Act – Update

- Robot is placed somewhere in the environment → location unknown
- SEE: The robot queries its sensors → finds itself next to a pillar

- ACT: Robot moves one meter forward
  - motion estimated by wheel encoders
  - accumulation of uncertainty
- SEE: The robot queries its sensors again → finds itself next to a pillar
- Belief updates (information fusion)

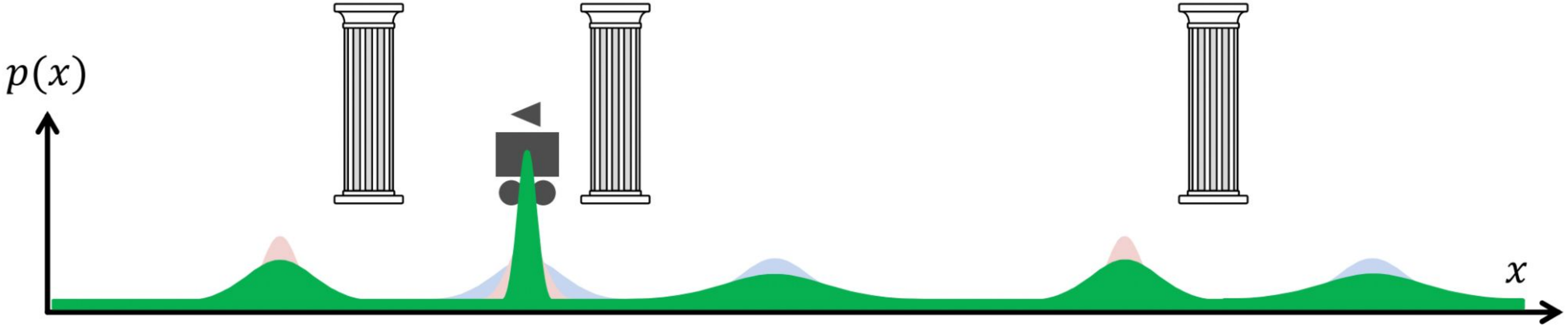


Dr. Allen

See – Act – Update

- Robot is placed somewhere in the environment → location unknown
- SEE: The robot queries its sensors → finds itself next to a pillar

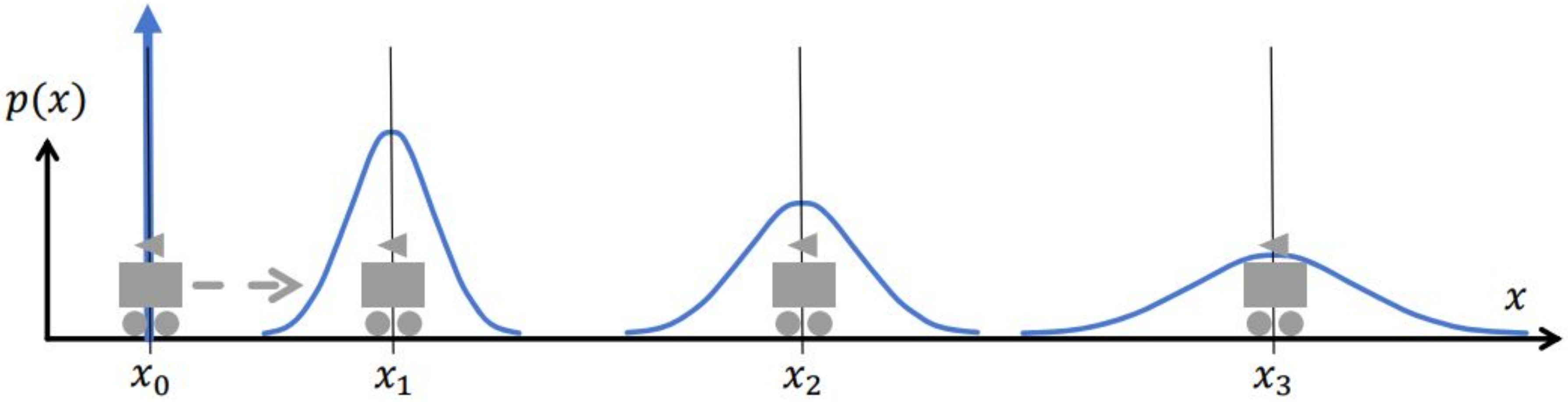
- ACT: Robot moves one meter forward
  - motion estimated by wheel encoders
  - accumulation of uncertainty
- SEE: The robot queries its sensors again → finds itself next to a pillar
- Belief update (information fusion)



Dr. Allen

Motion Model: Uncertainties

- The robot moves and estimates its position through its proprioceptive sensors
  - Wheel Encoder (Odometry)
- During this step, the robot's state uncertainty grows
  - The uncertainty will go unbounded if we are only 'dead reckoning'
  - Need to update the belief state to keep the uncertainties bounded

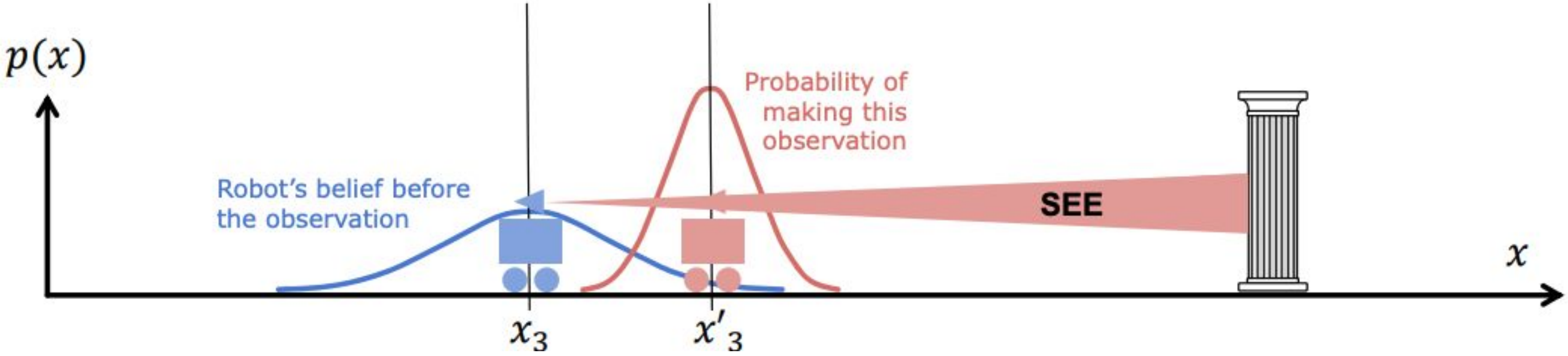


Dr. Allen

Motion Model: Observations

⇒ The <u>state update</u> step requires sensory observation

- The robot makes an observation using its exteroceptive sensors
- This results in a second estimation of the current position

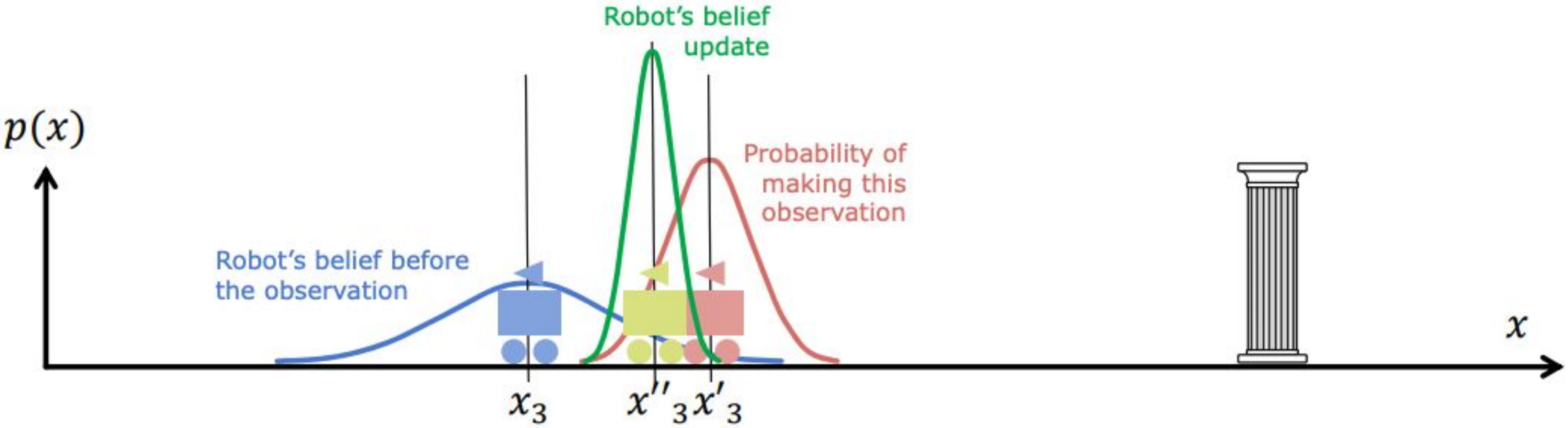


Dr. Allen

Motion Model: Belief Updates

⇒ The update step keeps the uncertainties bounded

- The robot corrects its position by combining its belief before the observation
- with the probability of making exactly that observation
- During this step, the robot's state uncertainty shrinks



Dr. Allen

Belief Representation

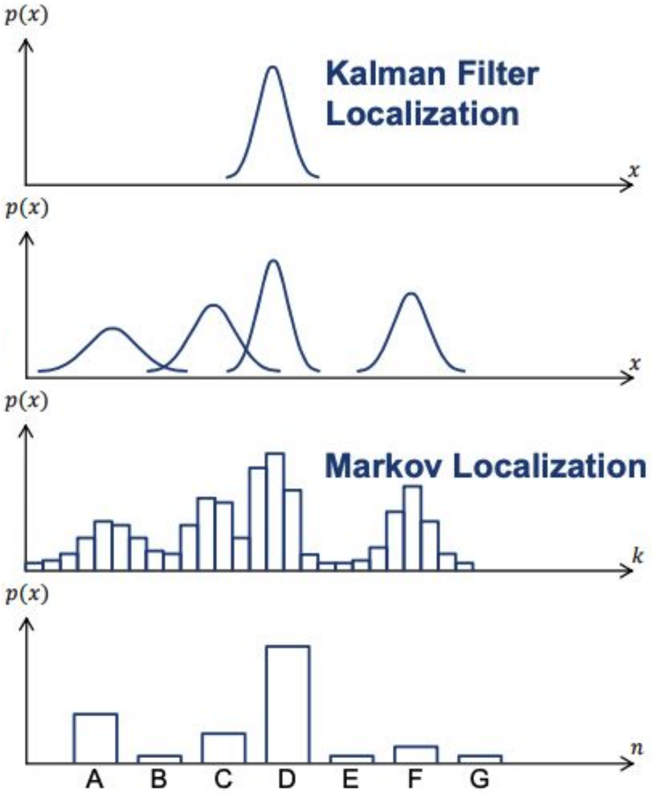
⇒ The Kalman Filtering way

- Continuous map with a probability distribution *p(x)*
- Can have
  - Singe hypothesis
  - Multiple hypothesis
- The state is usually represented by x ∈ X: a multivariate Gaussian random variable

⇒ For low-dimensional discrete spaces

- Discretized metric map (grid *k*) is visualized with a probability distribution *p(k)*
- Often, topological maps (with *n* nodes) are considered with a probability distribution *p(n)*

We will cover filtering algorithms in the last lecture

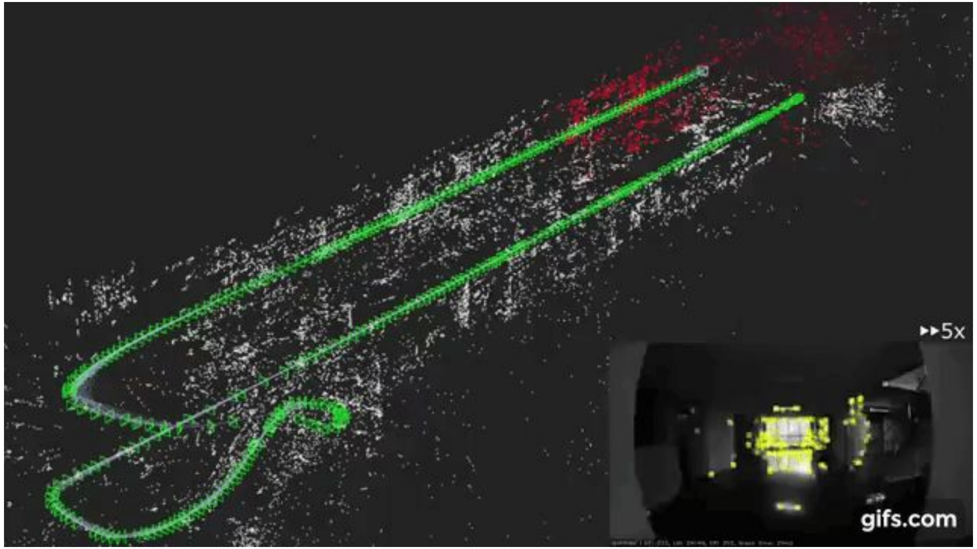


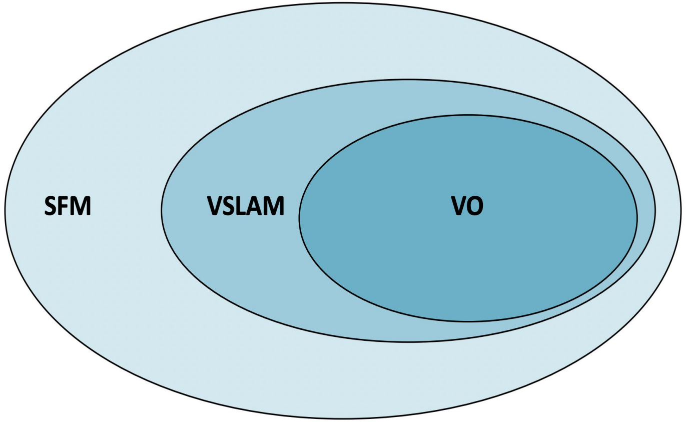
Dr. Allen

Localization Without a Map

⇒ Create map on the go

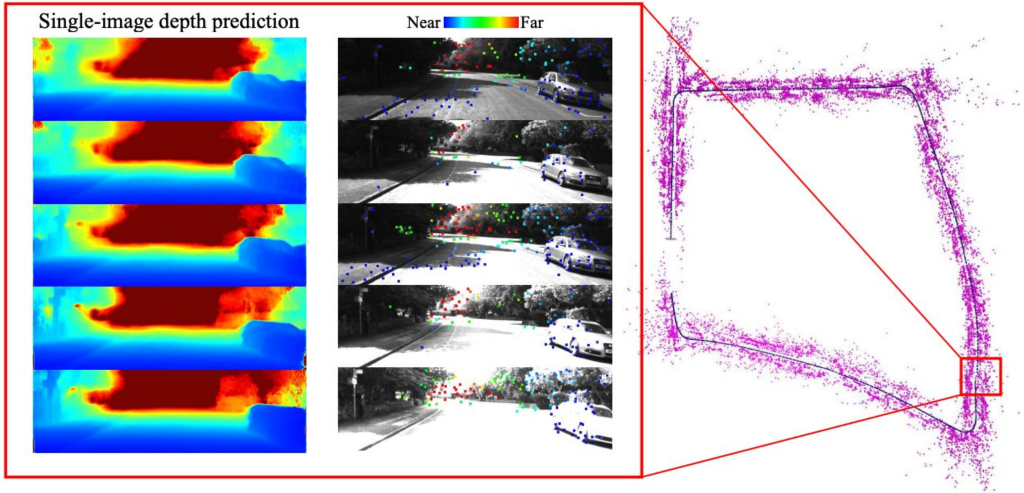
- SLAM: Simultaneous localization & mapping
- Popular SLAM literature
  - Visual SLAM
  - LiDAR-based SLAM
  - Laser-based SLAM (mainly 2D robots)



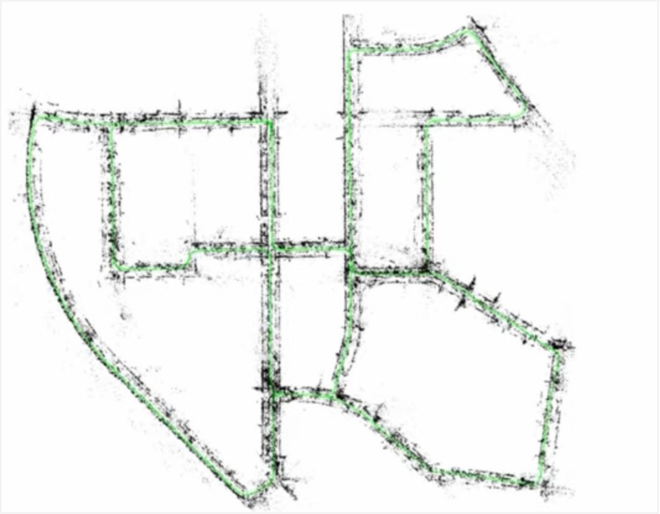


VO: pose recovery from motion of a calibrated camera
VSLAM: VO + place recognition (loop closure) + optimization for global consistency
SfM: Recovers scene structure from unordered cameras at different viewpoints (often uncalibrated cameras)

Visual Odometry (VO)



Loo et al



Kittie odometry benchmark

⇒ **VO:** Estimates camera pose from motion by tracking features across images (aka keyframes)

- Map representation: sparse / dense / semi-dense
- Major types:
  - Direct methods
  - Indirect (feature-based) methods

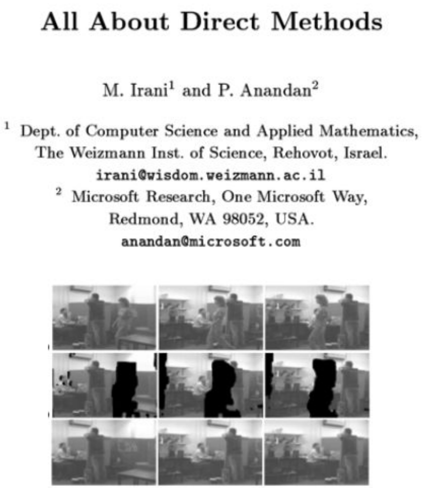
VO: Direct and Semi-Indirect Methods

⇒ Direct (featureless) VO

- Estimate structure and motion directly
  - By minimizing photometric error
  - Based on the image's pixel-level intensities
- The local intensity gradients are used in the optimization
- Issues:
  - Have brightness constancy assumption
  - Dependent on a good initialization
  - Low computational speed and no optimality guarantees
- Examples: DSO, LSD-SLAM

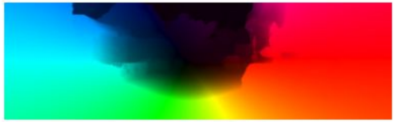
⇒ SVO: Semi-Direct VO

- Tracks and triangulates pixels that are characterized by high image gradients, but also relies on feature-based methods for a joint optimization for recovering structure and motion


All About Direct Methods

M. Irani[1] and P. Anandan[2]

[1] Dept. of Computer Science and Applied Mathematics,
The Weizmann Inst. of Science, Rehovot, Israel.
irani@wisdom.weizmann.ac.il
[2] Microsoft Research, One Microsoft Way,
Redmond, WA 98052, USA.
anandan@microsoft.com

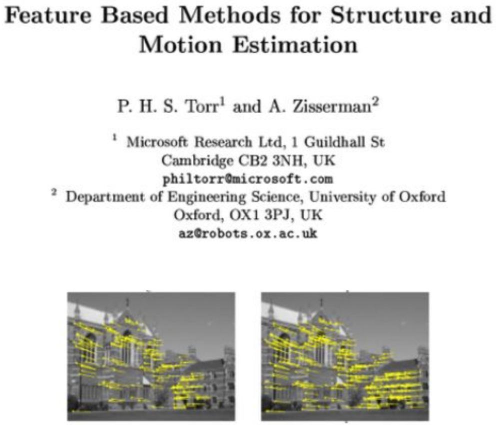


- All about direct methods
- SVO: Semi-direct VO methods

VO: Indirect (Feature-based) Methods

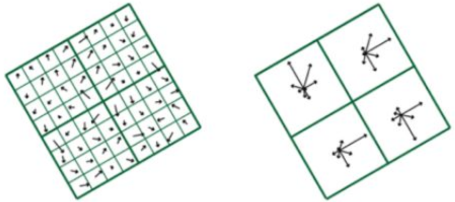
⇒ Indirect (feature-based) methods

- Follows the traditional SfM idea
  - Extract and match a sparse set of salient image features
  - Then recover both camera motion and structure using the epipolar geometry
  - Establish new feature correspondences with old landmarks when closing loops, which increases both the accuracy of the trajectory after bundle adjustment
- Issues:
  - Low speed due to feature extraction/matching at each frame
  - The necessity for robust estimation techniques that deal with erroneous correspondences (*eg*, RANSAC)
- Example: Mono-SLAM, ORB-SLAM


Feature Based Methods for Structure and Motion Estimation

P. H. S. Torr[1] and A. Zisserman[2]

[1] Microsoft Research Ltd, 1 Guildhall St
Cambridge CB2 3NH, UK
philtorr@microsoft.com
[2] Department of Engineering Science, University of Oxford
Oxford, OX1 3PJ, UK
az@robots.ox.ac.uk



- All about direct methods
- SVO: Semi-direct VO methods

DSO: Direct Sparse Odometry

⇒ **Sparse + Indirect**

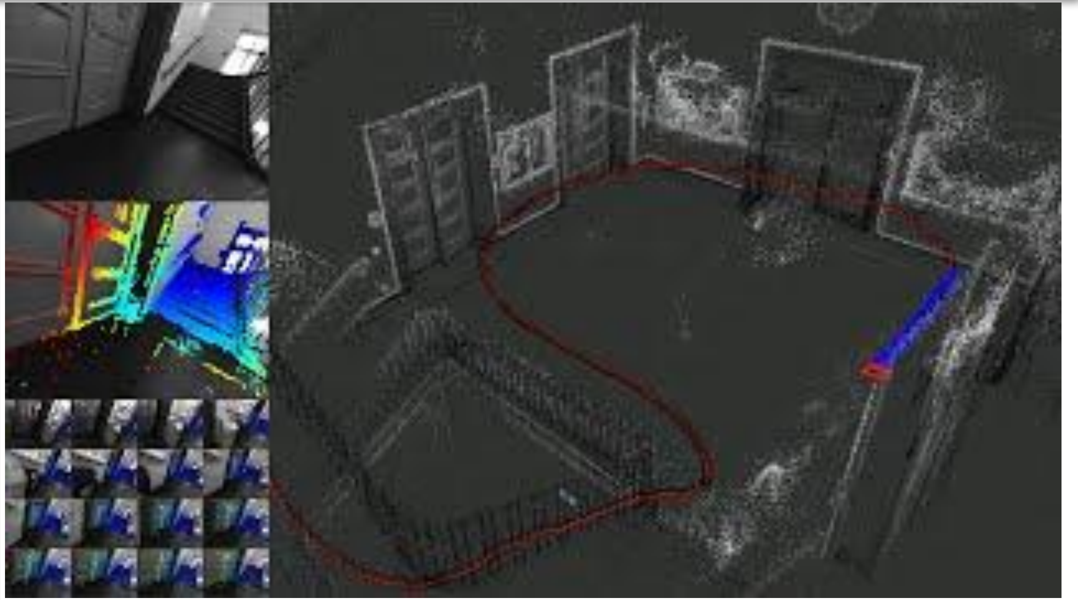
- Traditional SfM + BA pipeline
- Ex: Mono-SLAM, ORB-SLAM

⇒ **Dense + Indirect**

- Estimates 3D geometry from/with a dense, regularized optical flow field
- Combines a geometric error (from flow field) with a prior (smoothness)
- Ranftl et al

⇒ **Dense + Direct**

- Employs a photometric error and a geometric prior to estimate dense or semi-dense geometry
- Ex: DTAM, LSD-SLAM



https://youtu.be/C6-xwSOOdqQ

⇒ **DSO:** Direct + Sparse

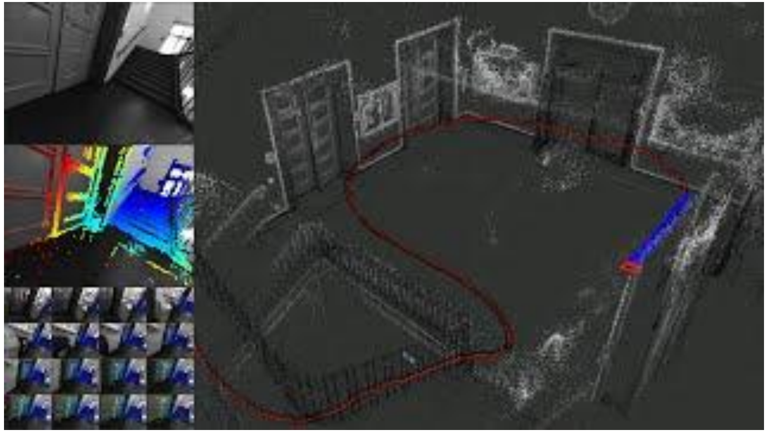
- Optimizes a photometric error defined directly on the images, without using a geometric prior

Important Papers: DSO

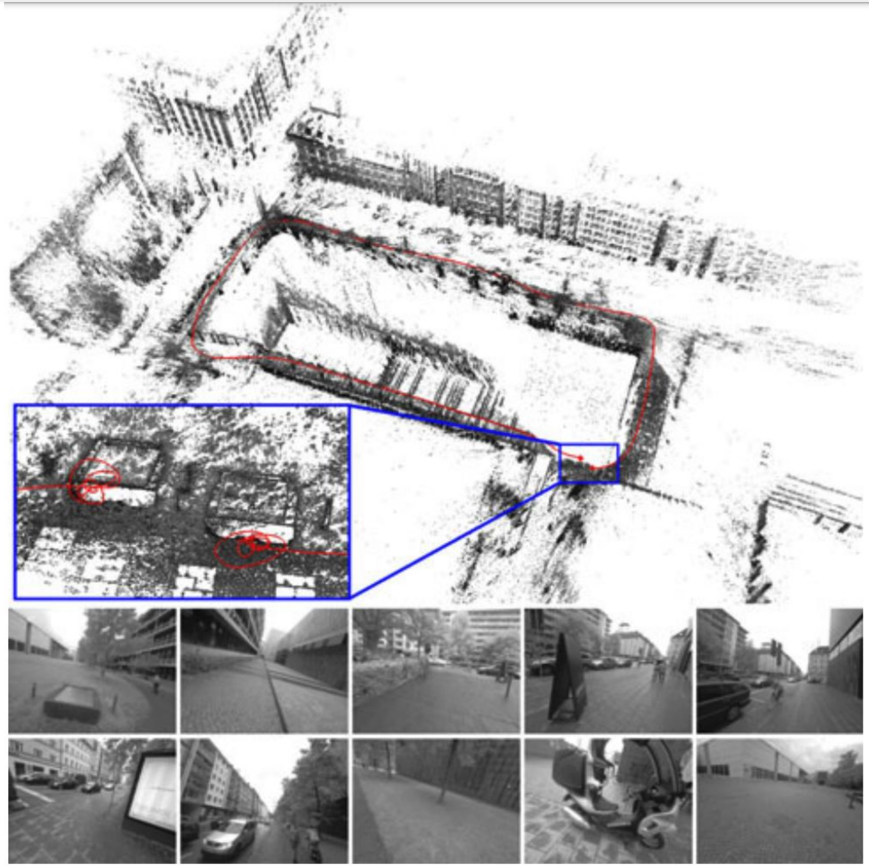
Direct Sparse Odometry

Jakob Engel, Vladlen Koltun, and Daniel Cremers

Abstract—Direct Sparse Odometry (DSO) is a visual odometry method based on a novel, highly accurate sparse and direct structure and motion formulation. It combines a fully direct probabilistic model (minimizing a photometric error) with consistent, joint optimization of all model parameters, including geometry-represented as inverse depth in a reference frame-and camera motion. This is achieved in real time by omitting the smoothness prior used in other direct methods and instead sampling pixels evenly throughout the images. Since our method does not depend on keypoint detectors or descriptors, it can naturally sample pixels from across all image regions that have intensity gradient, including edges or smooth intensity variations on essentially featureless walls. The proposed model integrates a full photometric calibration, accounting for exposure time, lens vignetting, and non-linear response functions. We thoroughly evaluate our method on three different datasets comprising several hours of video. The experiments show that the presented approach significantly outperforms state-of-the-art direct and indirect methods in a variety of real-world settings, both in terms of tracking accuracy and robustness.



https://youtu.be/C6-xwSOOdqQ

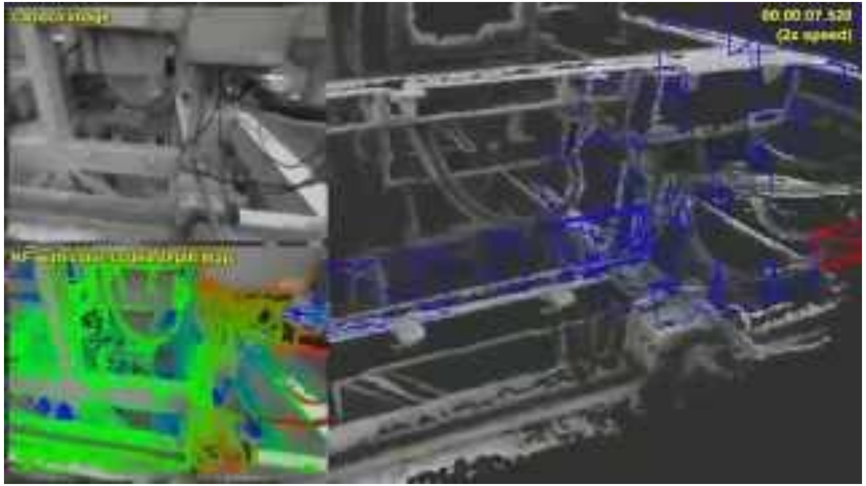


Important Papers: LSD-SLAM

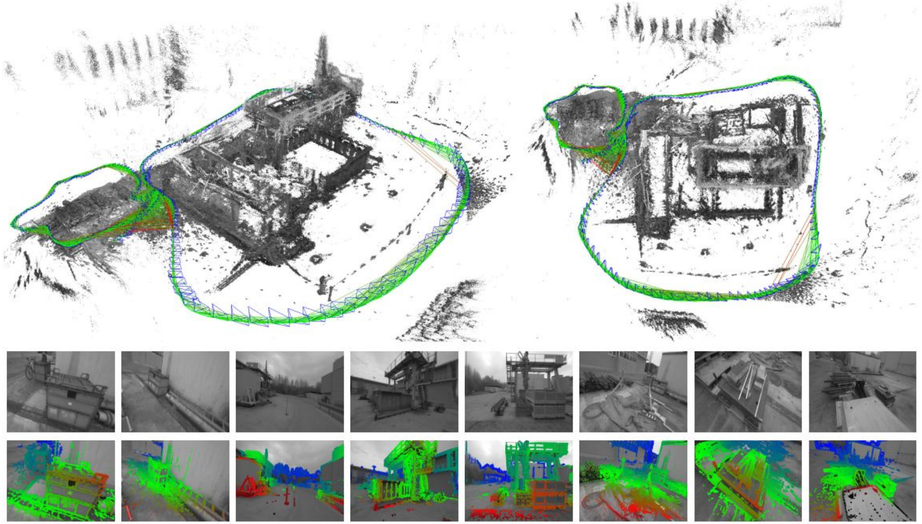
LSD-SLAM: Large-Scale Direct Monocular SLAM

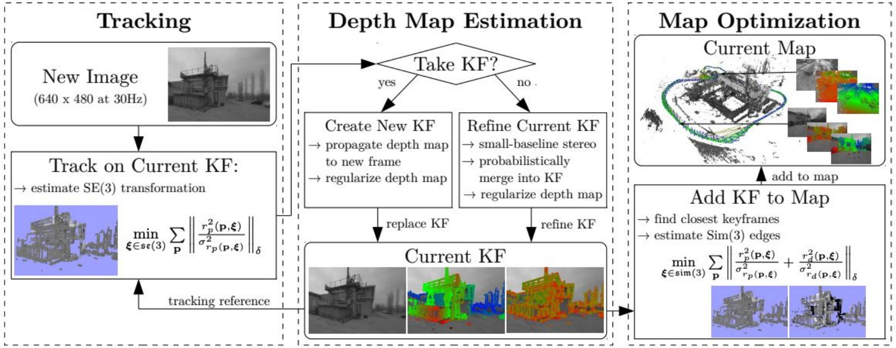
Jakob Engel and Thomas Schöps and Daniel Cremers

Technical University Munich



[Project page](#) https://youtu.be/GnuQzP3gty4

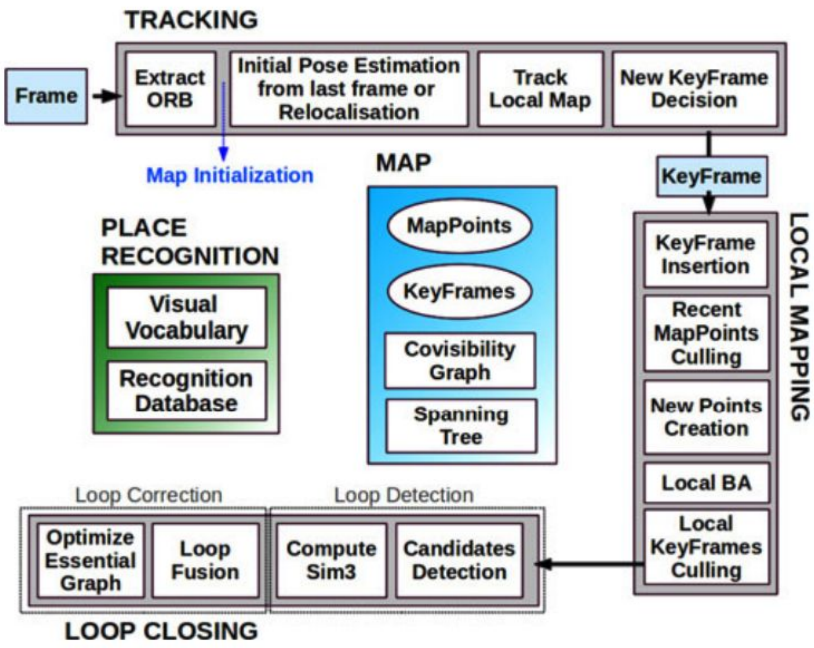


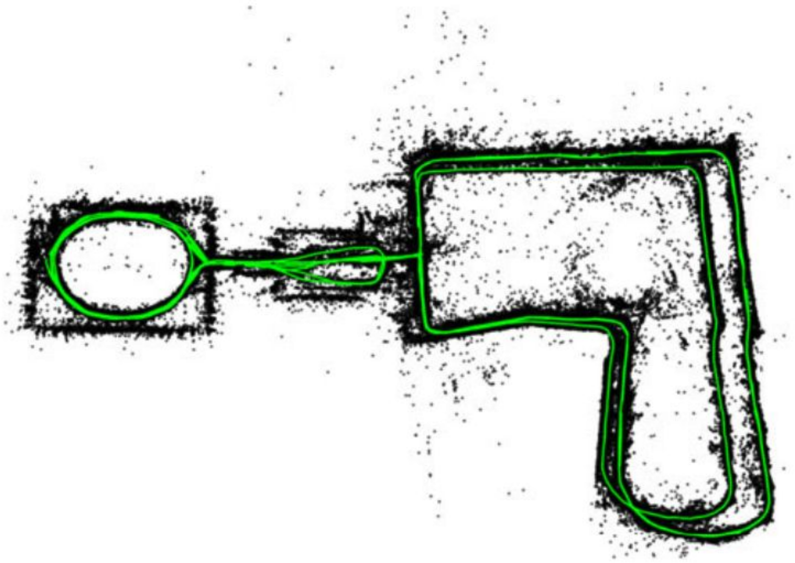


Important Papers: ORB-SLAM

ORB-SLAM: A Versatile and Accurate Monocular SLAM System

Raúl Mur-Artal, J. M. M. Montiel, *Member, IEEE*, and Juan D. Tardós, *Member, IEEE*

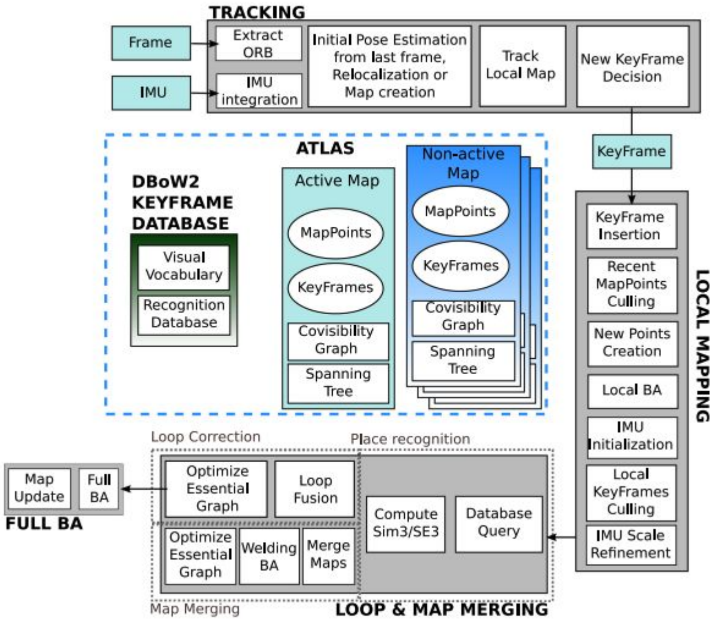


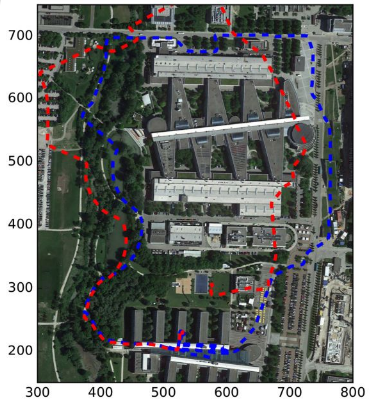


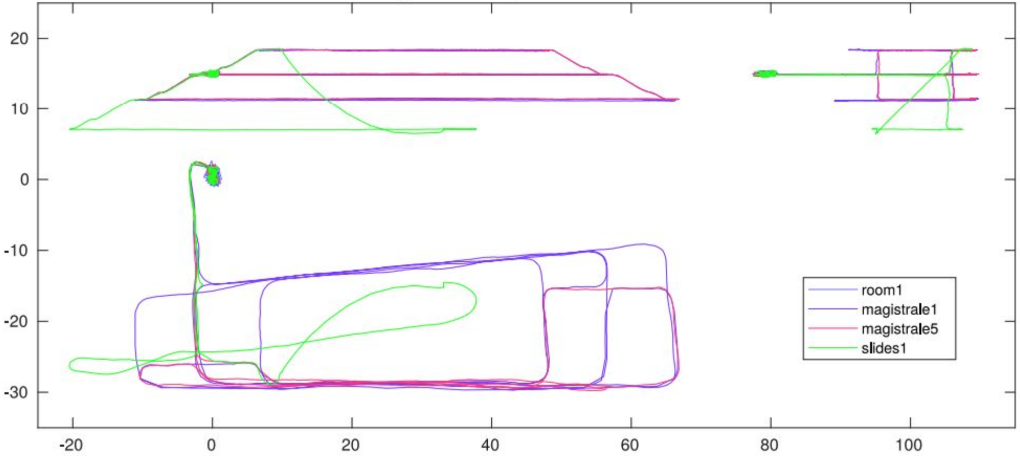
Important Papers: ORB-SLAM3

ORB-SLAM3: An Accurate Open-Source Library for Visual, Visual–Inertial, and Multimap SLAM

Carlos Campos, Richard Elvira, Juan J. Gómez Rodríguez, *Graduate Student Member, IEEE*, José M. M. Montiel, *Member, IEEE*, and Juan D. Tardós, *Senior Member, IEEE*

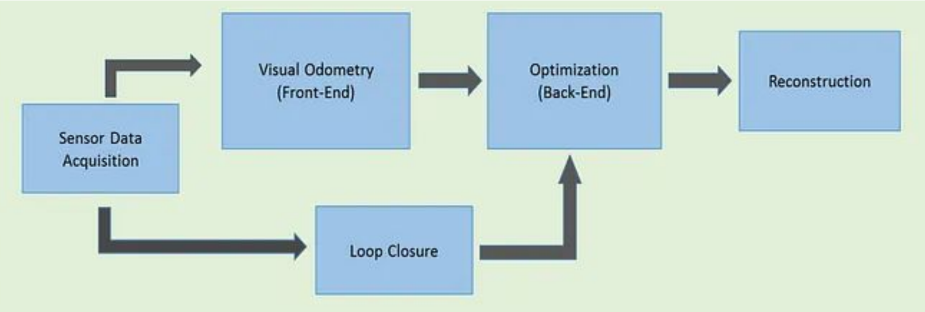






VO and VSLAM and Lidar-based SLAM

⇒ VSLAM = VO + Loop closure + Global optimization



blog

- VO provides only local/relative estimates, and the path is refined online with windowed optimization.
- VSLAM provides a global and consistent estimate
  - The detection of loop closure reduces the drift in both the map and the trajectory estimates
  - By performing bundle adjustment (BA)

SLAM intro in 5 min: https://youtu.be/BuRCJ2fegcc

https://youtu.be/5O8VmDiab3w

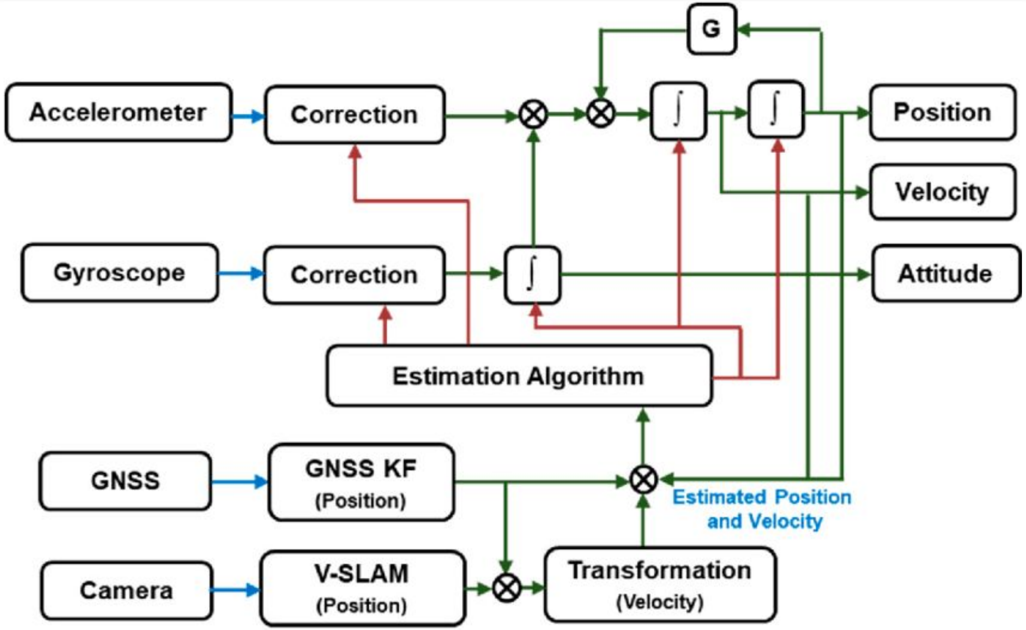
Visual Inertial Odometry (VIO)

⇒ **VIO:** VO (pose estimates) + IMU (error correction / scale)

- Uses VO to estimate camera pose from motion
- Inertial measurements from the IMU are used for error corrections associated with rapid motion

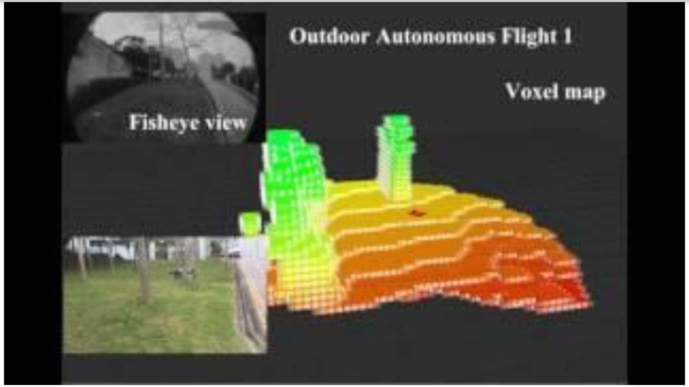
⇒ Backbone of **VINs (Visual Inertial Navigation Systems)**

- Uses camera+IMU sensory fusion to estimate robot pose in real-time
- VINs review paper
- ICRA-2019 workshop tutorial

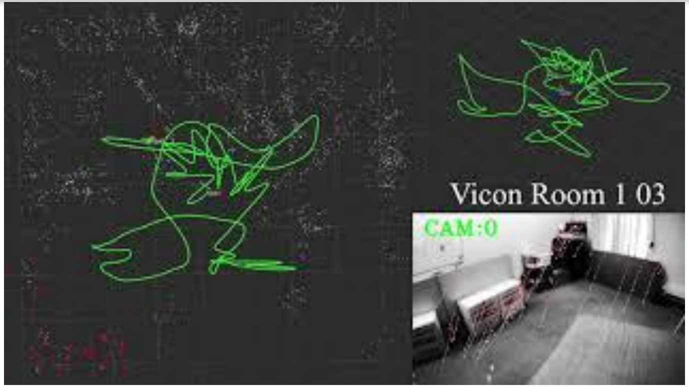


Chiang et al

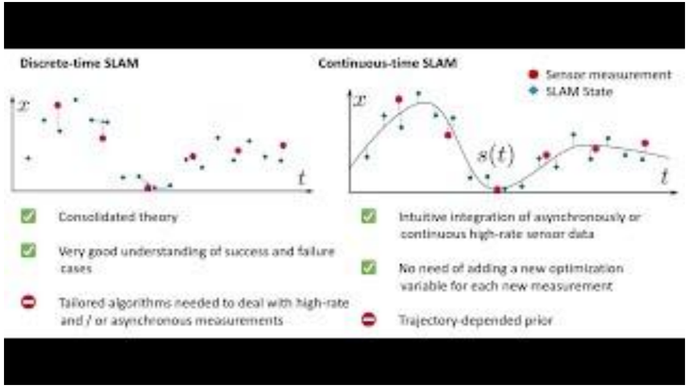
Intro to VINs



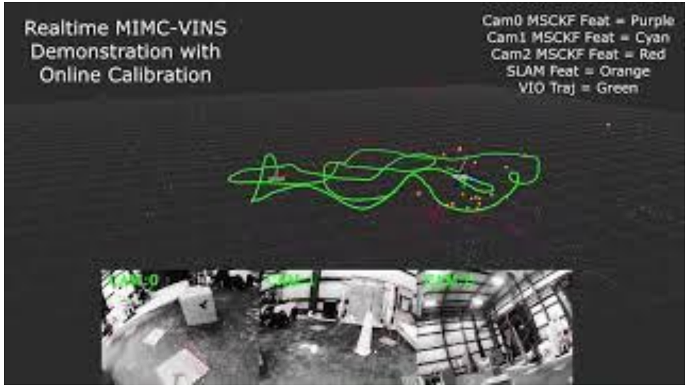
Monocular VINS: https://youtu.be/FalwEI0C2Y8



OpenVINS: https://youtu.be/187AXuuGNNw



[VIO and SLAM Playlist by UZH (RAL 2022)]()



Multi-Camera Multi-IMU: https://youtu.be/my1ljdJ4irY