Lecture: Localisation 1

Mathematical Basics & Dead-Reckoning

MOBILE ROBOT CONTROL 2023

Koen de Vos

Department of Mechanical Engineering – Control Systems Technology

EINDHOVEN UNIVERSITY OF TECHNOLOGY

How does our Robot know where it is?
(and why does it need to know it?)

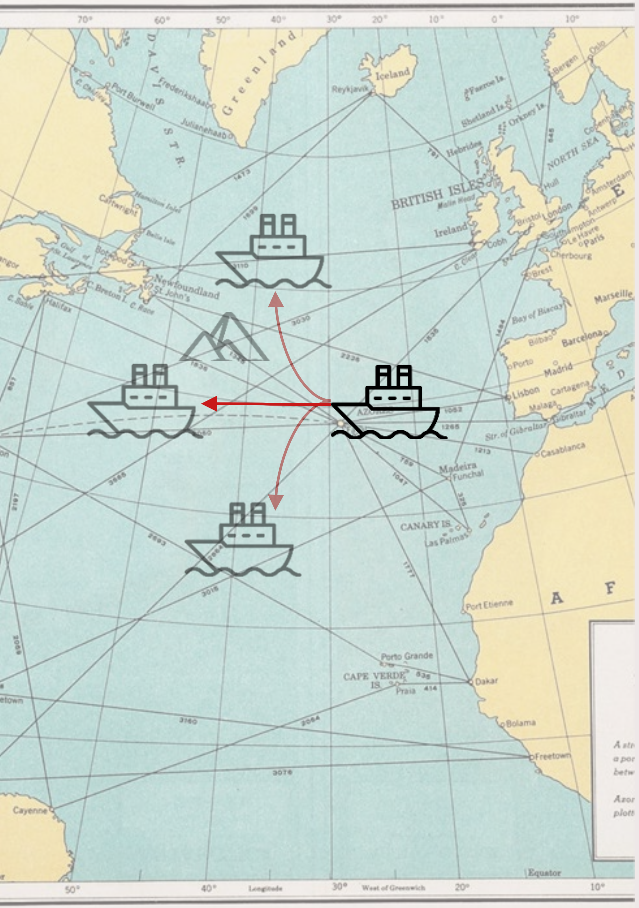
Let’s step in our time machine

- Imagine: You’re on a ship, at night, on the Atlantic ocean in the 1800s.

- All you have is:

  - a Compass,
  - a Map,
  - a Clock,
  - the Sun to estimate your longitude at Noon,
  - the stars to estimate your Latitude,
  - a rough estimate of your velocity

How do we know where we are and how do we get to our destination?



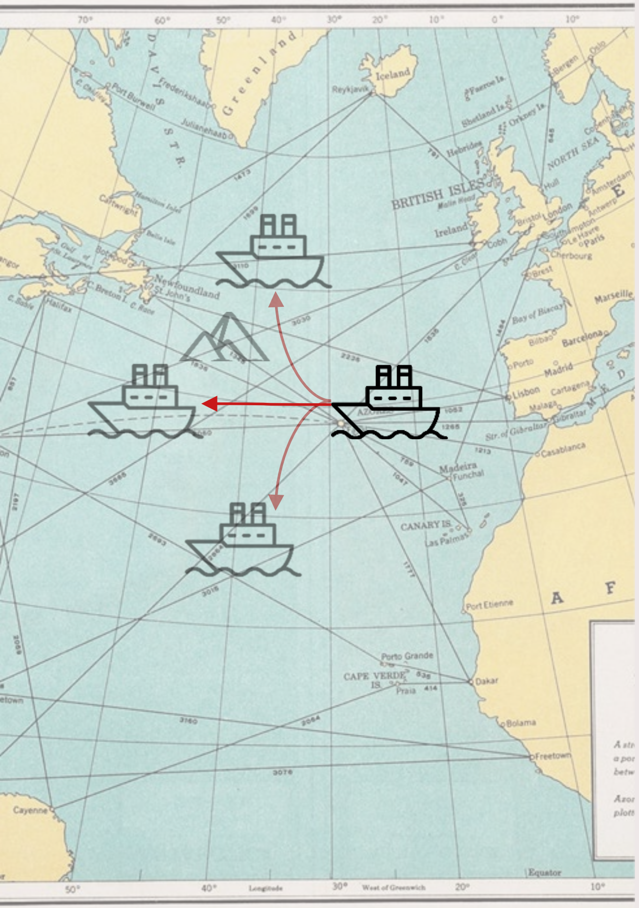
Let’s step in our time machine

- Imagine: You’re on a ship, at night, on the Atlantic ocean in the 1800s.

- All you have is:

  - a Compass,
  - a Map,
  - a Clock,
  - the Sun to estimate your longitude at Noon,
  - the stars to estimate your Latitude,
  - an estimate of your velocity

How do we know where we are and how do we get to our destination?



Let’s step in our time machine

- Imagine: You’re on a ship, at night, on the Atlantic ocean in the 1800s.

- We could:
  - Estimate our latitude at night
  - Estimate our longitude at noon

  - Keep updating our position given our velocity, time and our heading

- What can we say about its accuracy.

- How much accuracy do we need?

Why is this relevant to Robotics?

Our robot is not a ship, right?

Why is this relevant to Robotics?

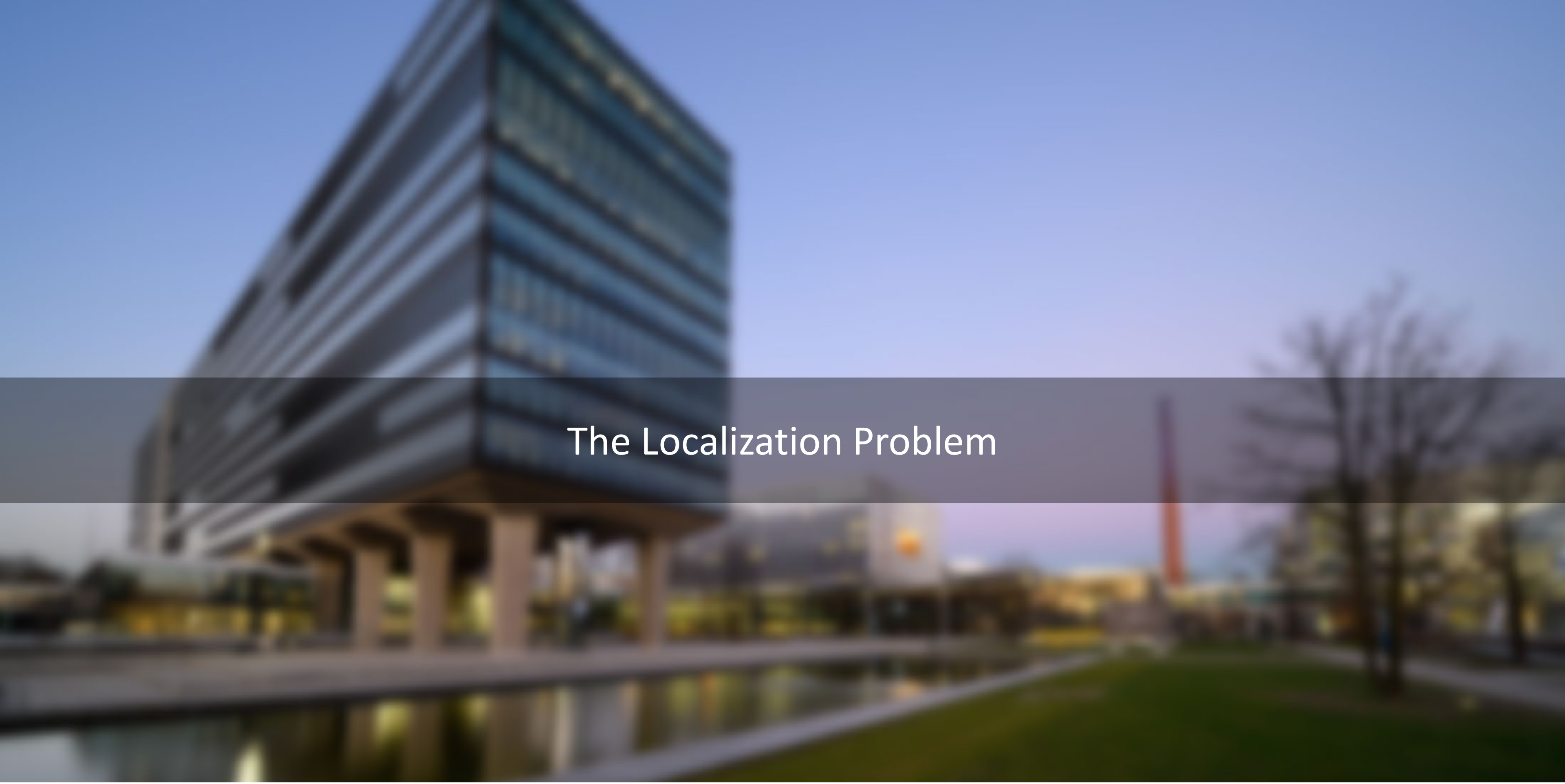
Our robots also deal with partial and imperfect information.

- We don’t have an absolute position sensor
- But we do have multiple sources of information we can use to infer our location

Goal of this Lecture

Why do we need, and what is, Robot Localization?

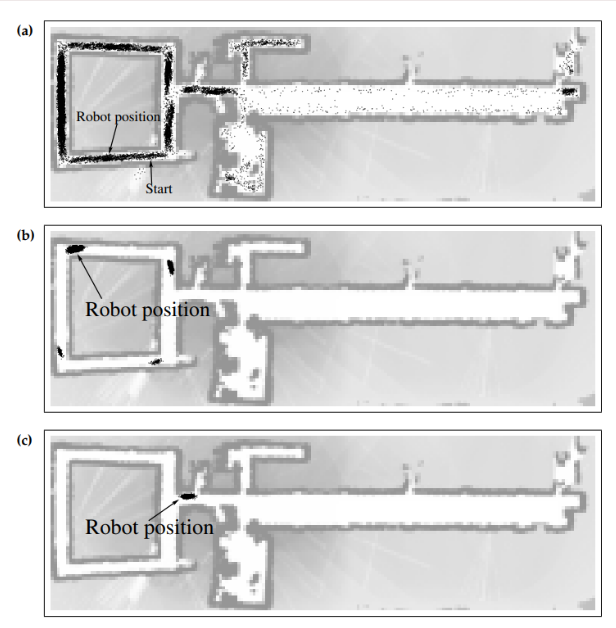
How do we solve the Localization
Problem using a dead-reckoning approach?



The Localization Problem

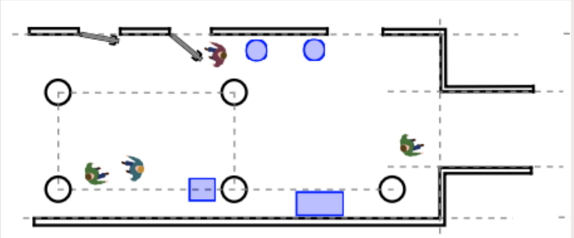
Different types of Localization problems

- For instance, depending on your **Prior Information** or Enviroment
  - Position Tracking
  - Global Localization
  - Kidnapped Robot Problem



Different types of Localization problems

- For instance, depending on your Prior Information or **Enviroment**
    - Static Enviroment
    - Dynamic Enviroment
    - Semi-Dynamic Enviroment



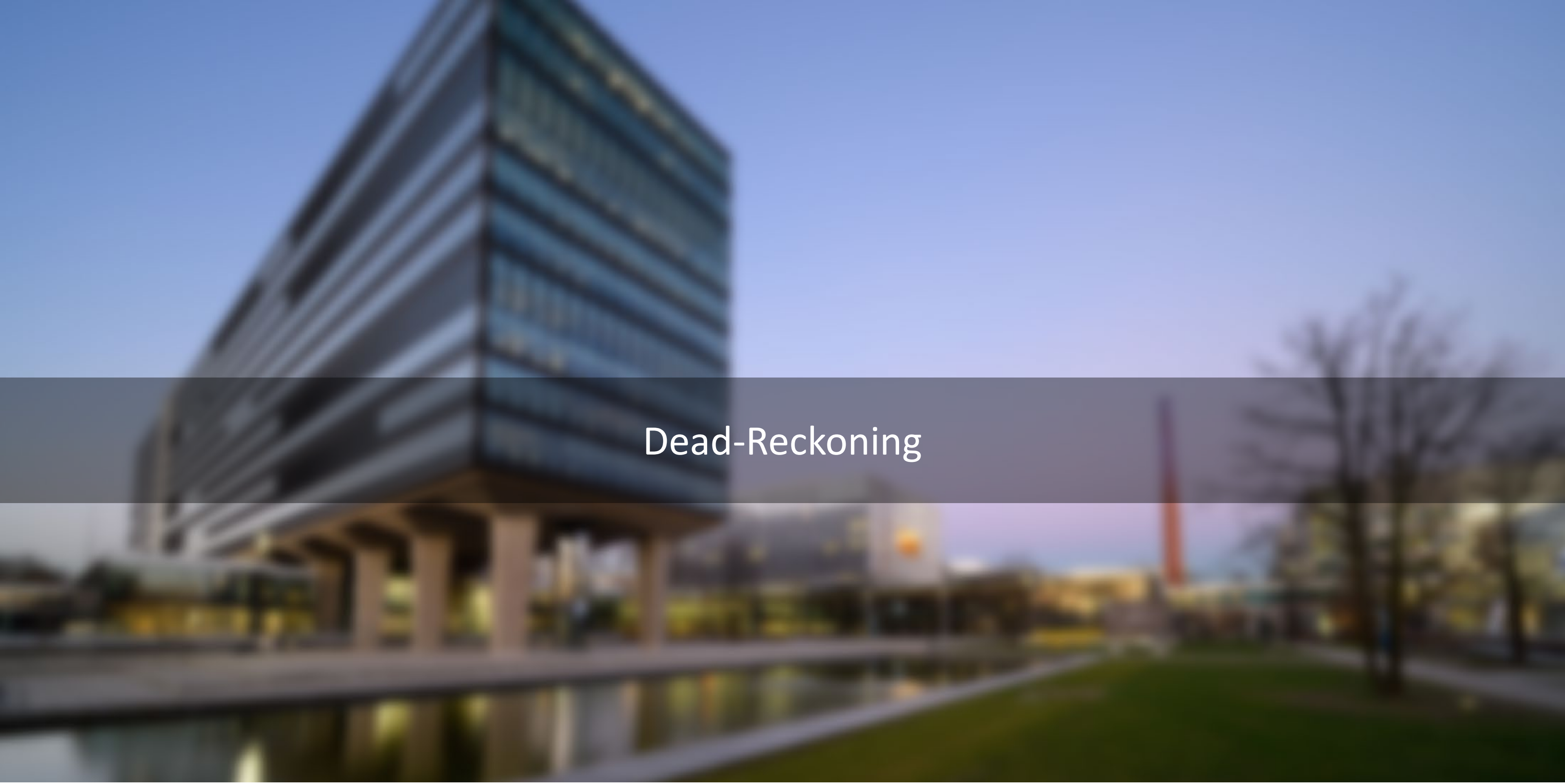
Hendrikx, R. W. M. (2023). *Object and Pattern Association for Robot Localization*. [Phd Thesis 1 (Research TU/e / Graduation TU/e), Mechanical Engineering]. Eindhoven University of Technology.

Large Variety of Approaches

- Your approach to solving the problem may vary depending on

  - Your enviroment
  - The type of problem you are solving
  - The available sensor modalities and their reliability
  - Your computational resources
  - The availability of a map
  - …..

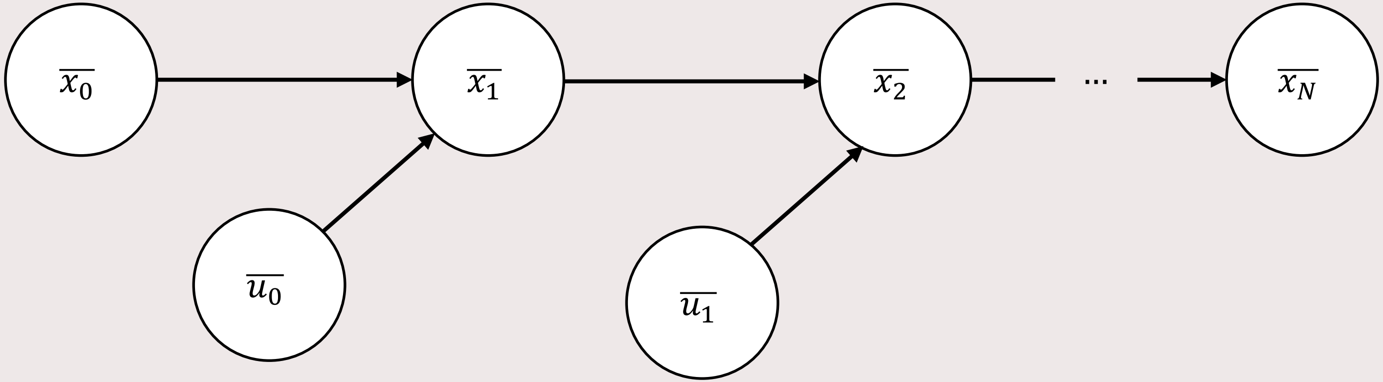
  - Today we are focusing on **dead-reckoning**



Dead-Reckoning

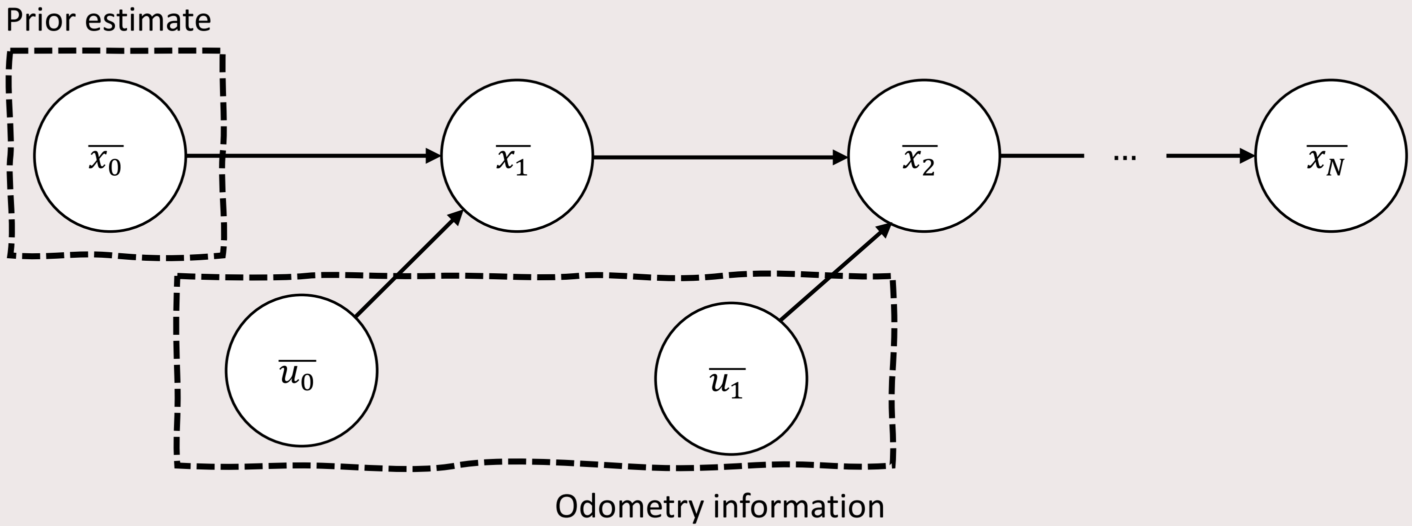
Dead-reckoning

Idea

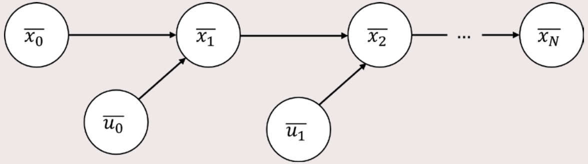


Dead-reckoning

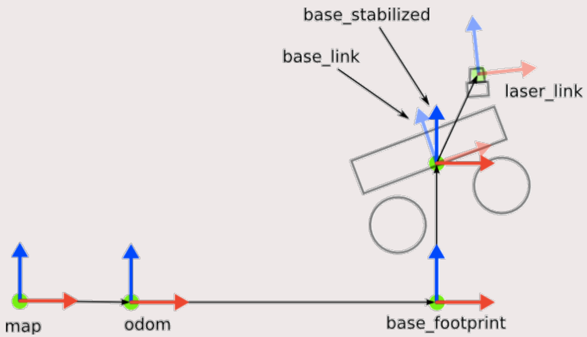
Idea



Dead-reckoning
Coordinate-Frames

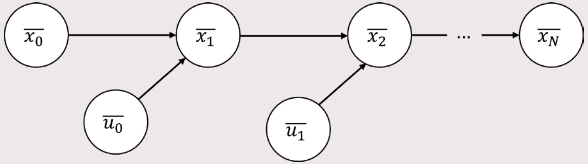


- We, most likely, have information in different coordinate frames
  - Odometry in odometry-frame
  - Prior estimate (ang goal) in map-frame
- Measurement both translated and rotated w.r.t eachother
- How do we convert between them?



GIANNI A. DI CARO 16-311-Q INTRODUCTION TO ROBOTICS LAB NOTES: ODOMETRY, ROS REFERENCE FRAMES

GIANNI A. DI CARO 16-311-Q INTRODUCTION TO ROBOTICS LAB NOTES: ODOMETRY, ROS REFERENCE FRAMES



Dead-reckoning
Coordinate-Frames

- Homogenous transformations!
- For instance, we have the 2D homogenous transformation between robot and map frame

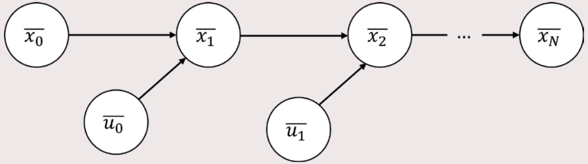
$$T_R^m = \begin{bmatrix} R & T \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta & x_t \\ \sin\theta & \cos\theta & y_t \\ 0 & 0 & 1 \end{bmatrix}$$

- Then

$$\begin{bmatrix} x_o \\ 1 \end{bmatrix} = T_R^o \begin{bmatrix} x_R \\ 1 \end{bmatrix}$$

- For further details, or a recap:
  - http://ais.informatik.uni-freiburg.de/teaching/ws22/mapping/ -> Homogenous Coordinates

Dead-reckoning
Coordinate-Frames



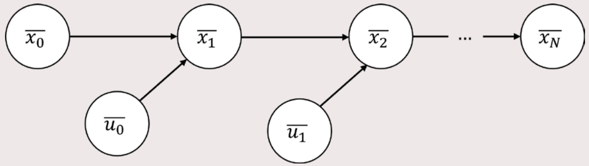
- And

$$T_m^R = (T_R^M)^{-1} = \begin{bmatrix} R^{-1} & -R^{-1}T \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} \cos\theta & \sin\theta & -\cos\theta\, x_t - \sin\theta\, y_t \\ -\sin\theta & \cos\theta & \sin\theta\, x_t - \cos\theta\, y_t \\ 0 & 0 & 1 \end{bmatrix}$$

- For further details, or a recap:
  - http://ais.informatik.uni-freiburg.de/teaching/ws22/mapping/ -> Homogenous Coordinates

Dead-reckoning

How?



Given a prior estimate (in map frame)

While true:

odom_update ← new odom message
Transform odom_update into map frame
Add the odometry update to your prior estimate

What do we expect?

How does it perform given imperfect information?

Dead-reckoning

Typical Results

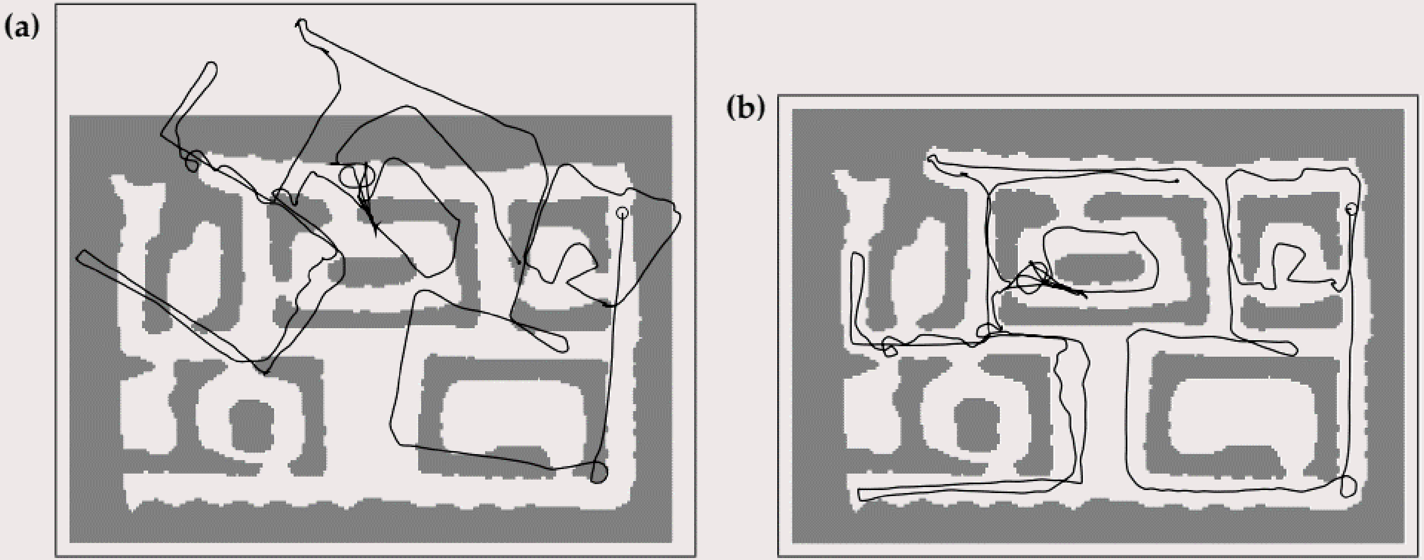


Figure 8.10 (a) Odometry information and (b) corrected path of the robot.

From: Thrun, Sebastian. "Probabilistic robotics." Communications of the ACM 45.3 (2002): 52-57.

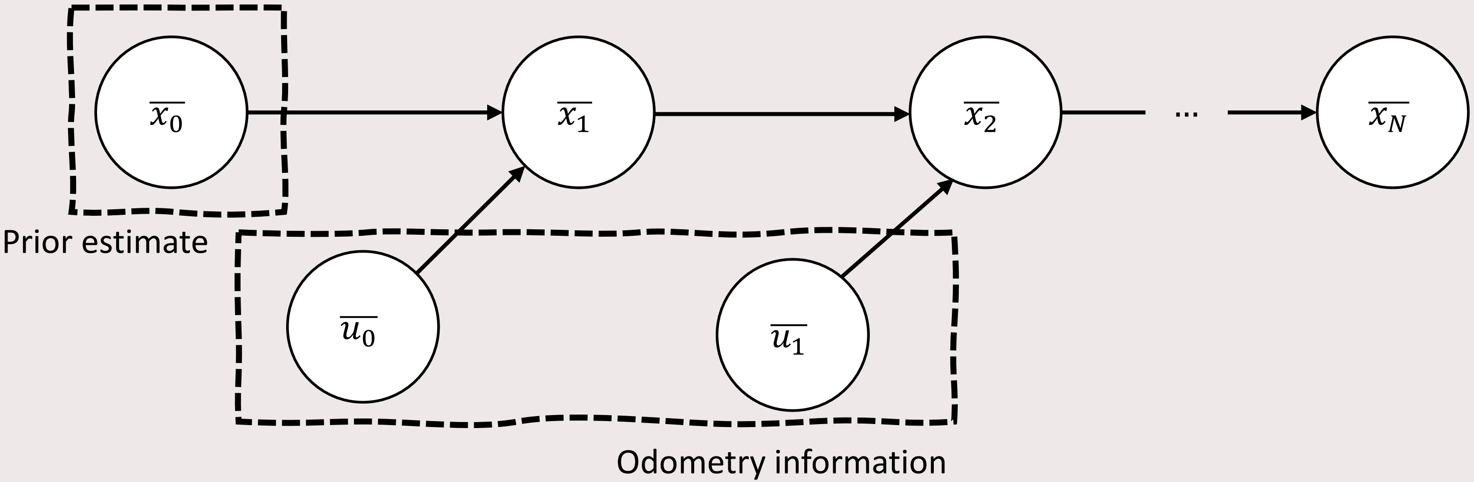
Can we do better?

We have more than one source of Information!

A First Look at
Recursive State Estimation

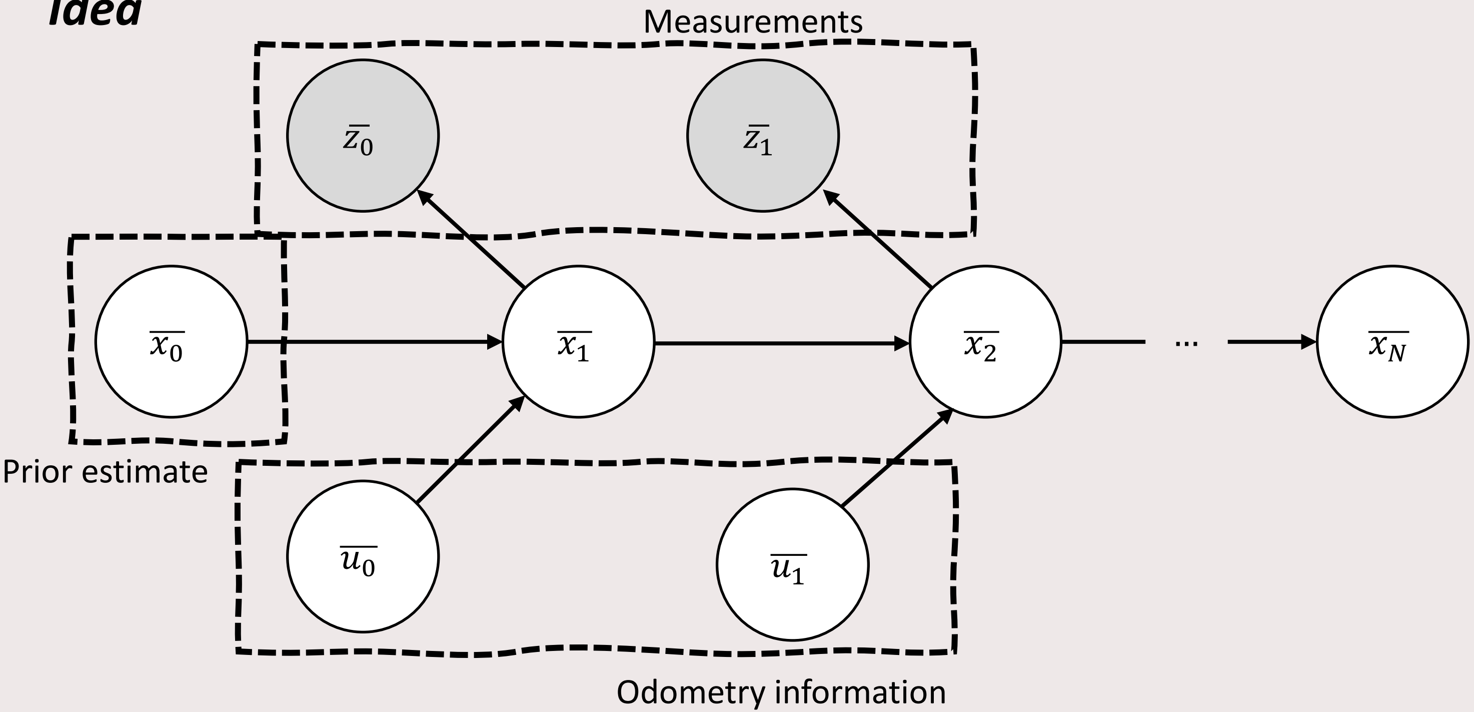
Recursive State-Estimation

Idea

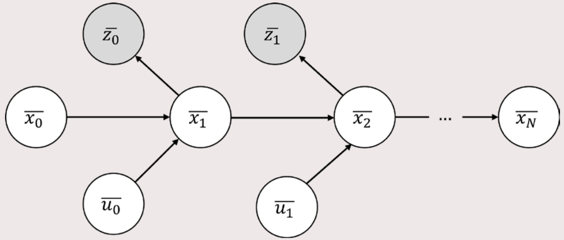


Recursive State-Estimation

Idea



Recursive State Estimation
Probabilistic approach

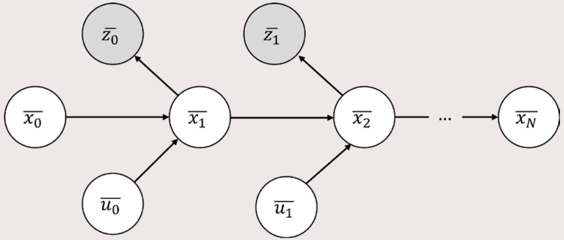


Core idea:

- Combine the uncertain information to obtain a more certain view,
- Incorporate measurements over multiple time steps.

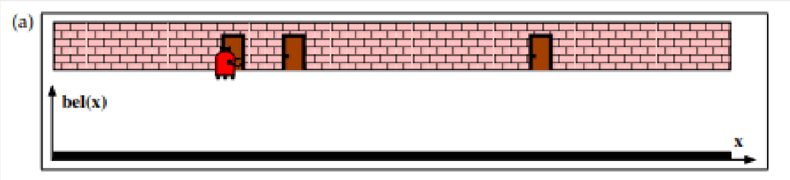
→ Bayes Filter

Recursive State Estimation
Probabilistic approach

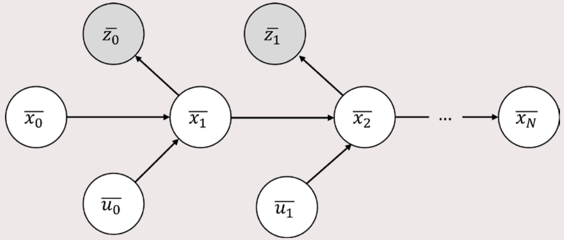


Our belief of the current state:

$$bel(x_t) \coloneqq p(x_t | z_{1:t-1}, u_{1:t-1})$$



Recursive State Estimation
Probabilistic approach

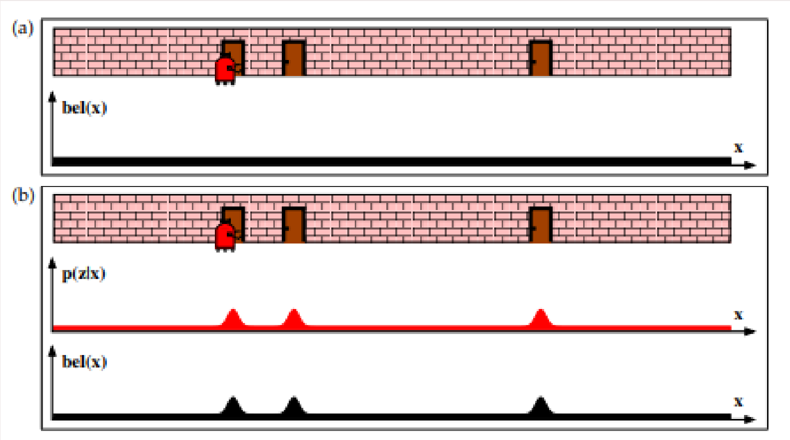


Our belief of the current state:

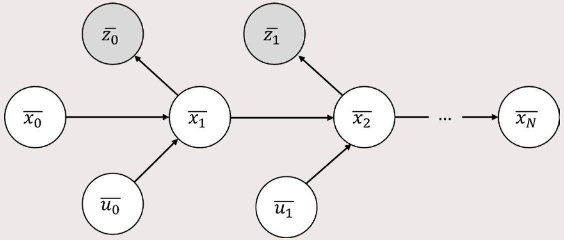
$$bel(x_t) \coloneqq p(x_t|z_{1:t-1}, u_{1:t-1})$$

Measurement update:

$$bel(x_{t+1}) = p(z_t|x_t)\overline{bel}(x_t)$$

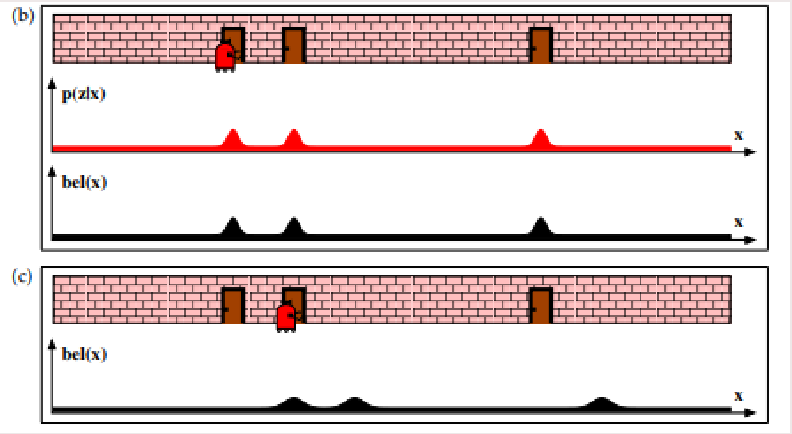


Recursive State Estimation
Probabilistic approach



Our belief of the current state:

$$bel(x_t) \coloneqq p(x_t | z_{1:t-1}, u_{1:t-1})$$



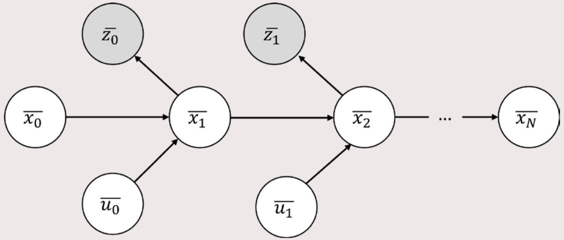
Measurement update:

$$bel(x_t) = p(z_t | x_t)\overline{bel}(x_t)$$

Control (dead-reckoning) update:

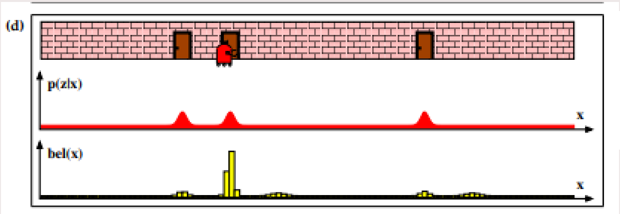
$$\overline{bel}(x_t) = \int p(x_t | u_t, x_{t-1})\, bel(x_{t-1})\, dx_{t-1}$$

Recursive State Estimation
Probabilistic approach

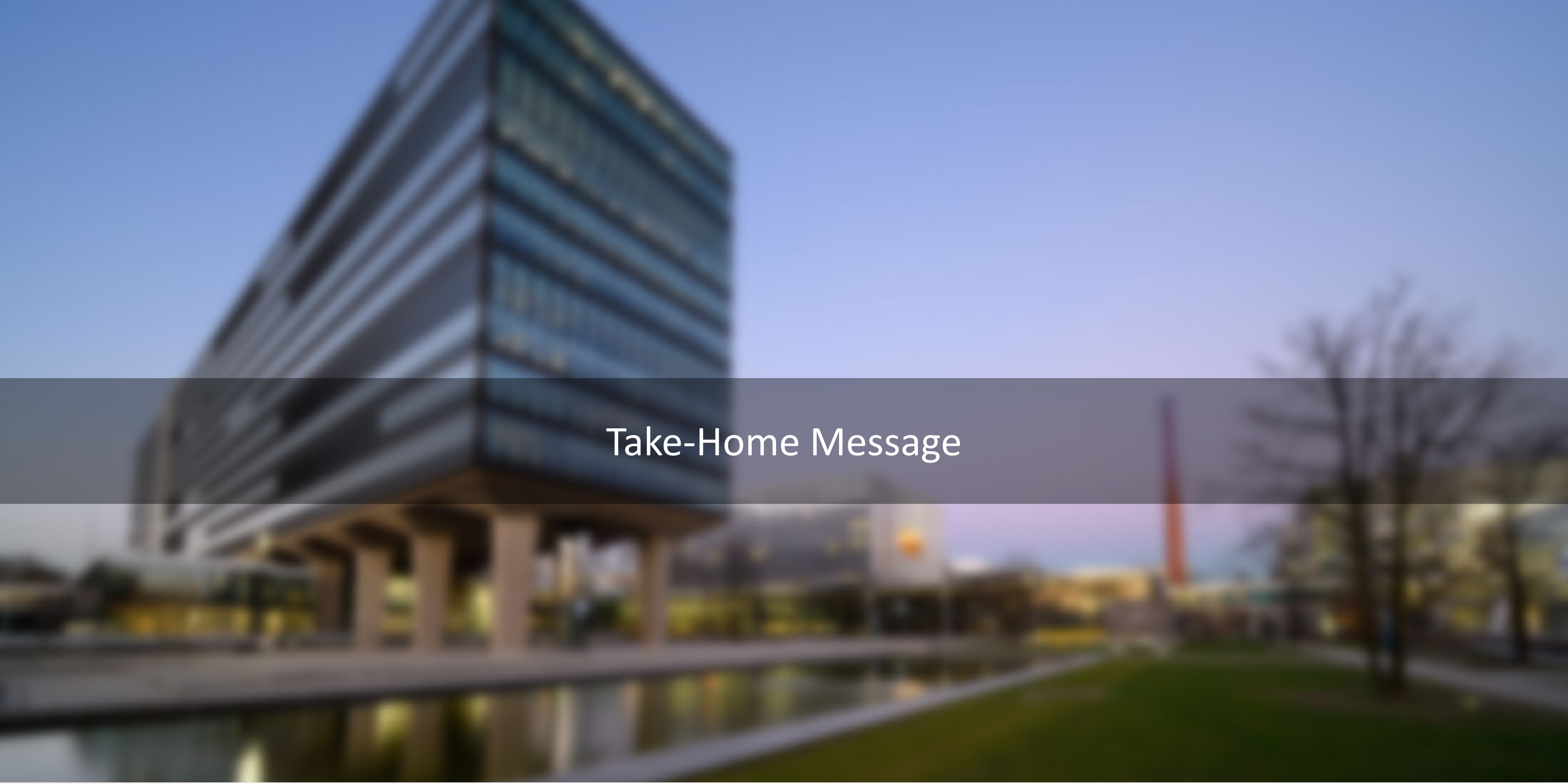


However:

- The Bayes filter is generally intractable
- We cannot compute a solution in real time



- Next week:
    - The particle filter, state-of-practice approximation of the Bayes Filter



Take-Home Message

Localization is an integral part of the robot navigation problem

Dead-reckoning is easy but has its flaws.

Using multiple sensor modalities could allow you to achieve more accurate localization performance