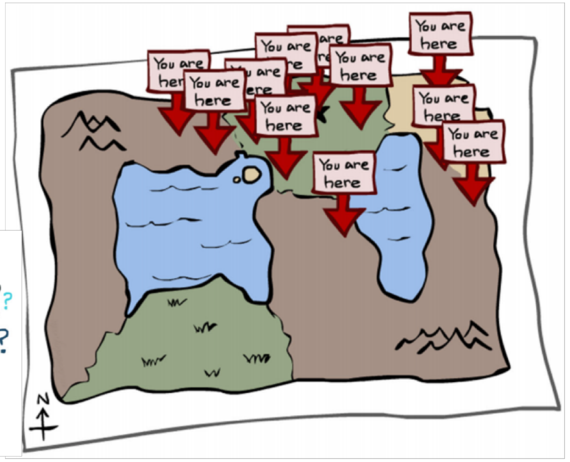
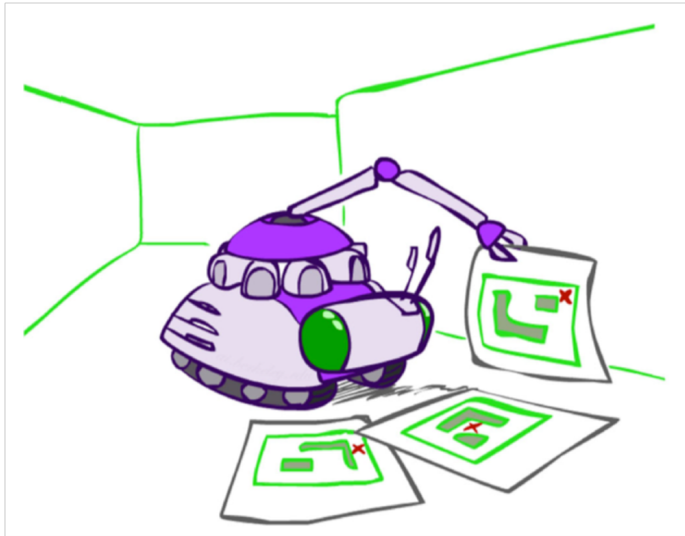
CS 1501: Intro to Robotics

Autonomy, AI, and Applications



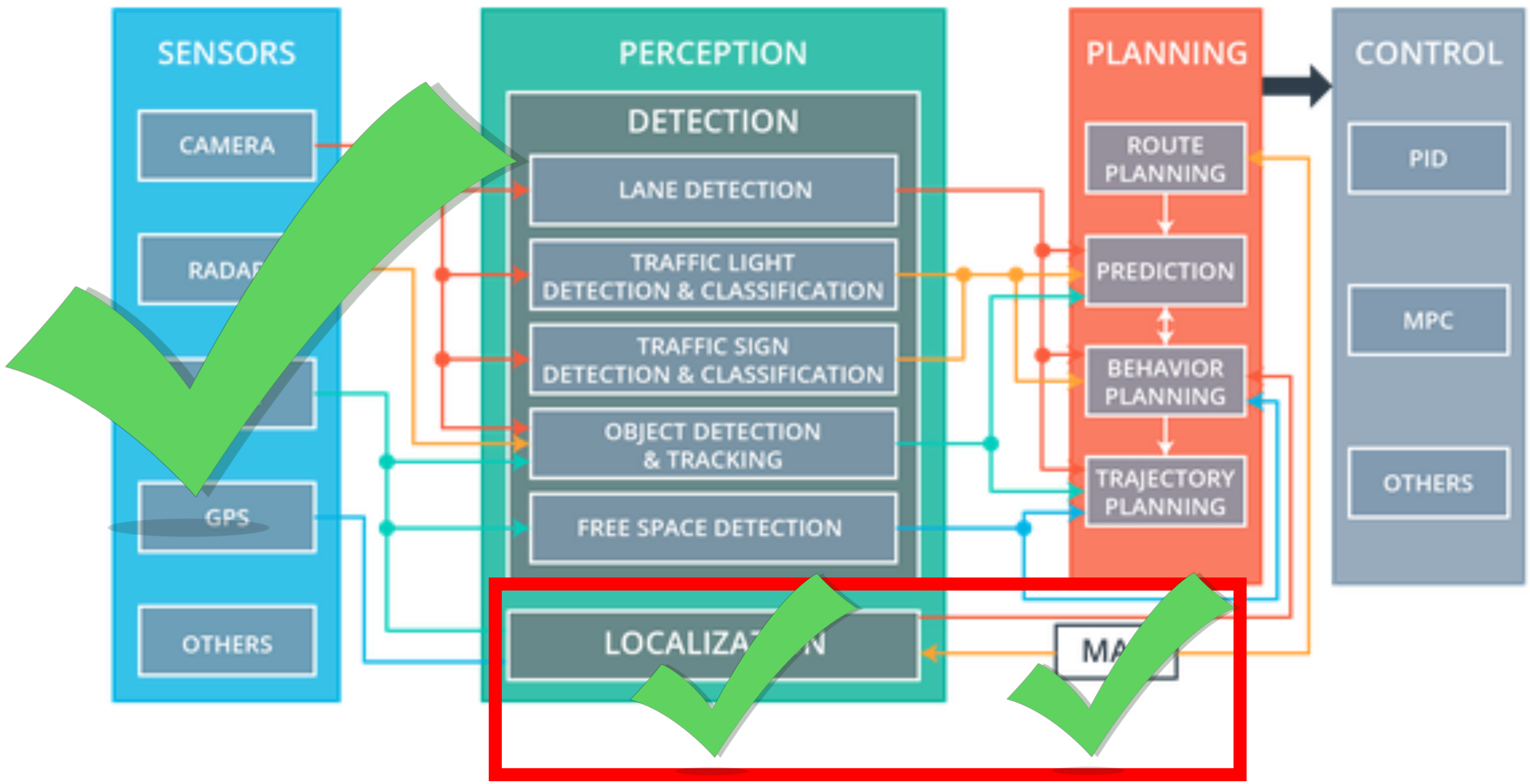
Localization II



Rohan Raval

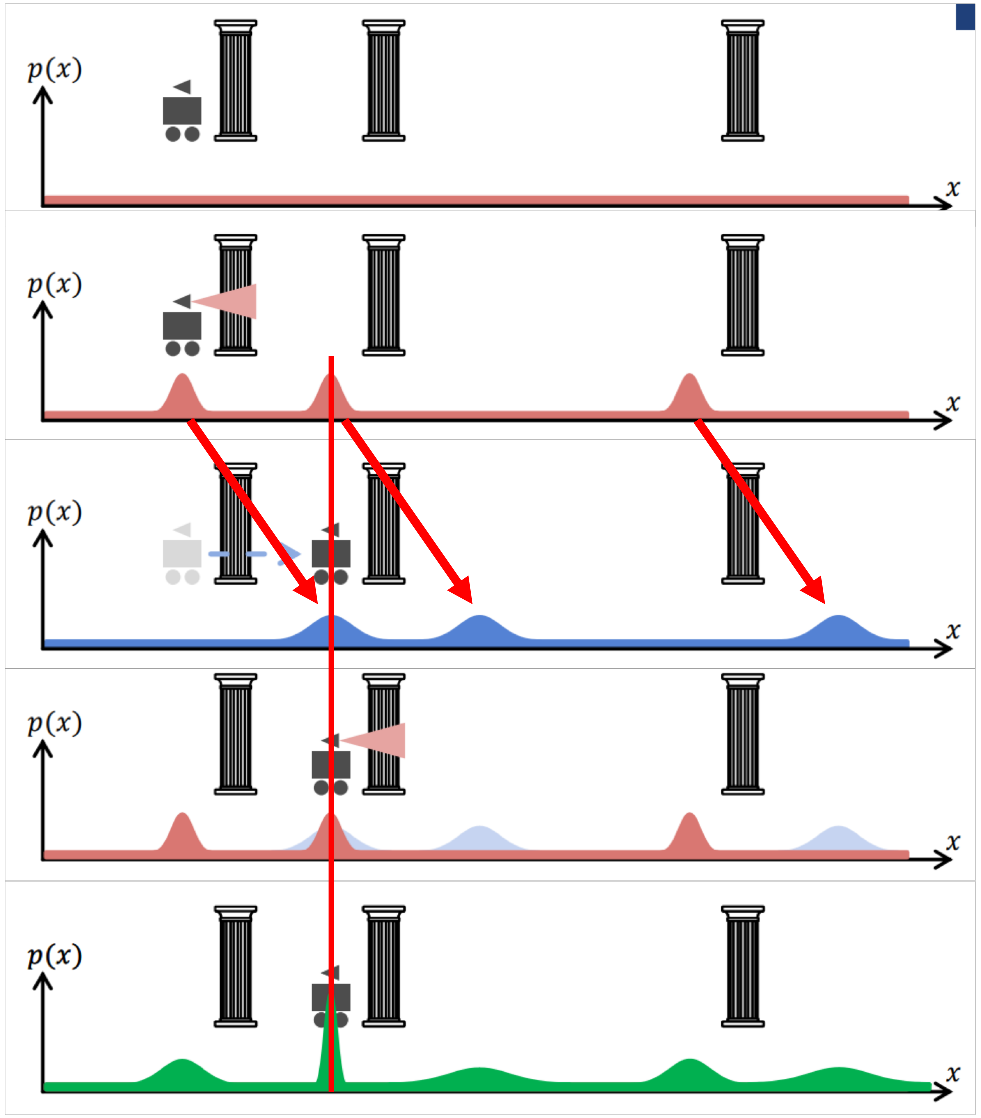
Monday 1-1:50pm, MEC 213

Recap: See-Think-Act



Recap: Probabilistic Localization

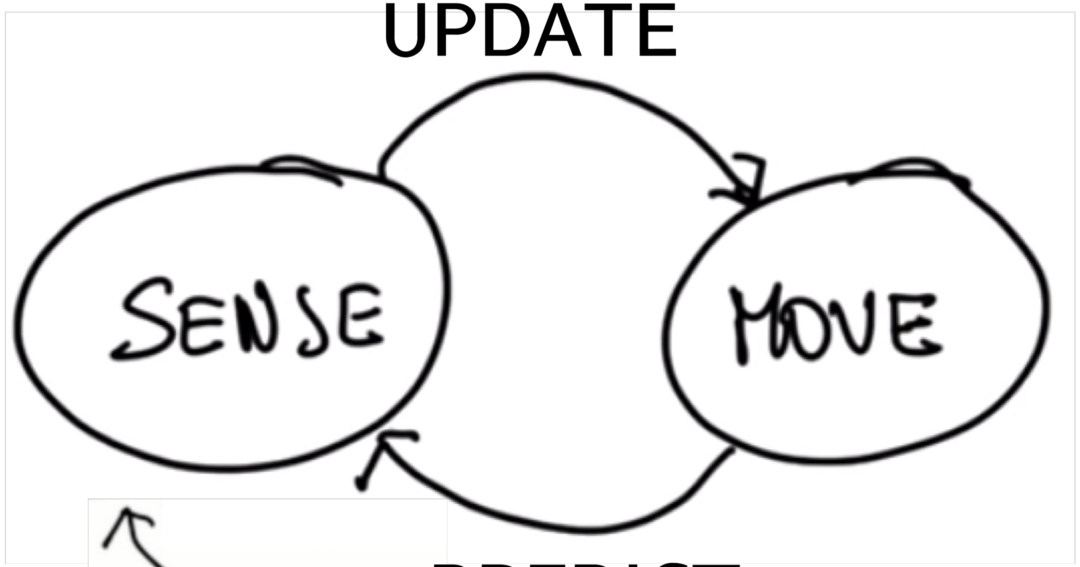
1. Initial Belief (Prior)
2. Measurement → Update
3. Move → Predict
4. Measurement → Update
5. Final Belief (Posterior)



Predict-Update Cycle

UPDATE

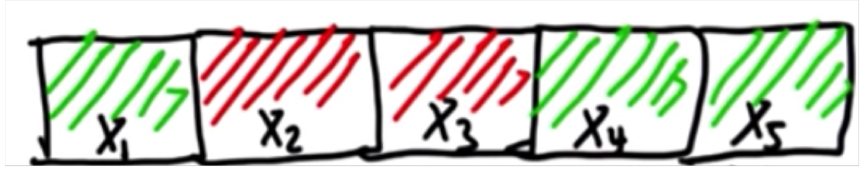
Gains Information



Loses Information

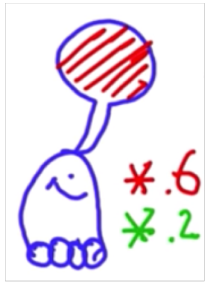
PREDICT

Histogram Filter: Exercise



1. Initial Belief (Prior)

0.2	0.2	0.2	0.2	0.2

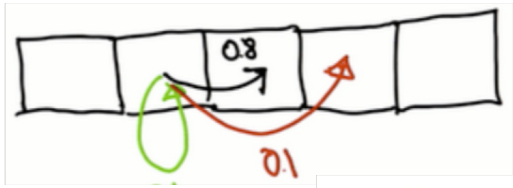


2. Measurement → Update
   - Normalize so all probabilities add to 1

0.04	0.12	0.12	0.04	0.04

0.11	0.33	0.33	0.11	0.11

3. Move → Predict
   - Cyclic World!



0.11	0.13	0.31	0.31	0.13

4. Measurement → Update
   - Normalize so all probabilities add to 1

0.022	0.078	0.186	0.062	0.026

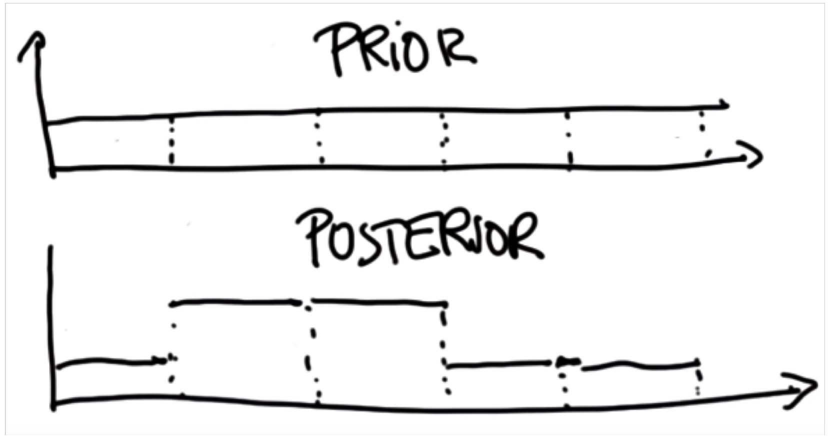
0.059	0.201	0.497	0.166	0.069

5. Final Belief (Posterior)

6%	**20%**	**50%**	**17%**	**7%**

Histogram Filter

- Pros
  - Easy to Implement
  - Simple – just a histogram!
  - Multimodal belief (job security term!)
    - We can represent belief for many locations

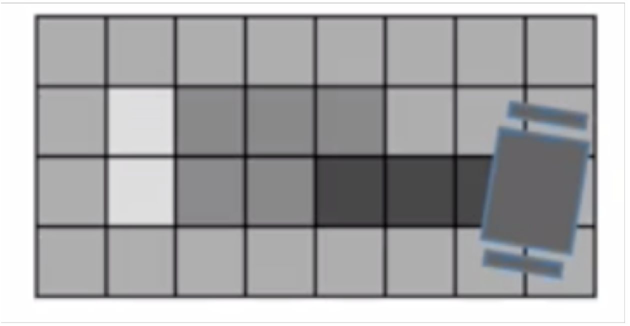


- Cons
  - One-dimensional
    - How does it scale for higher dimensions?
  - Discrete
  - Computationally expensive

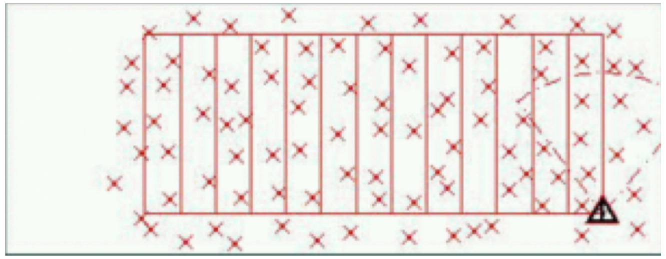
Quick Note on Landmarks...

Two Types of Mapping:

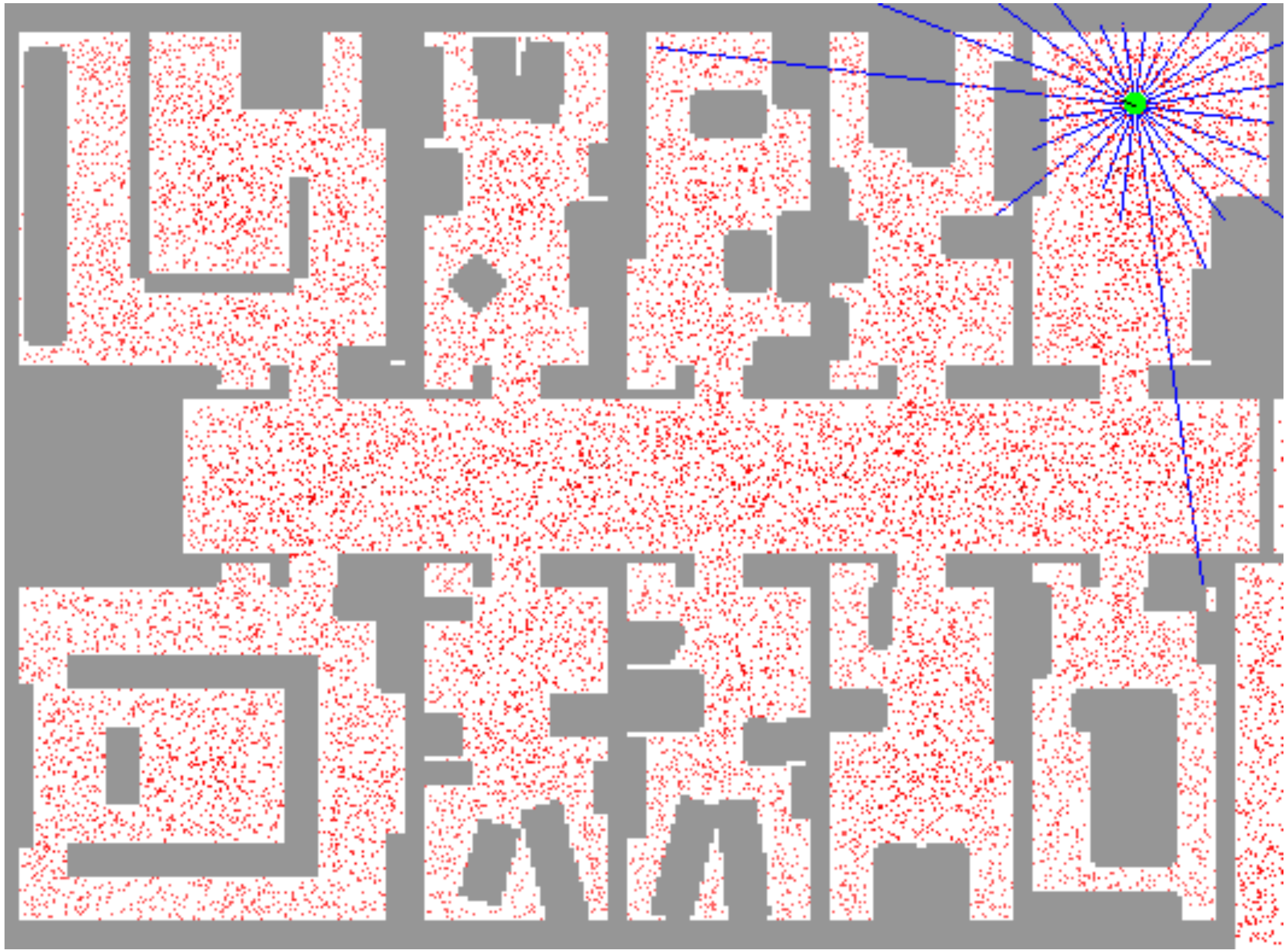
1. Occupancy Mapping



2. Landmark/Feature Mapping
   - Landmarks are *literally* your map!



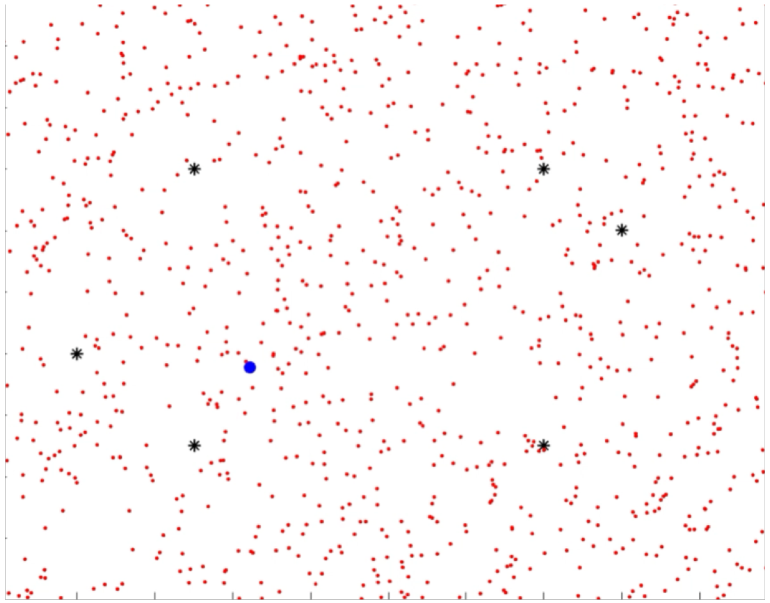
Particle Filter

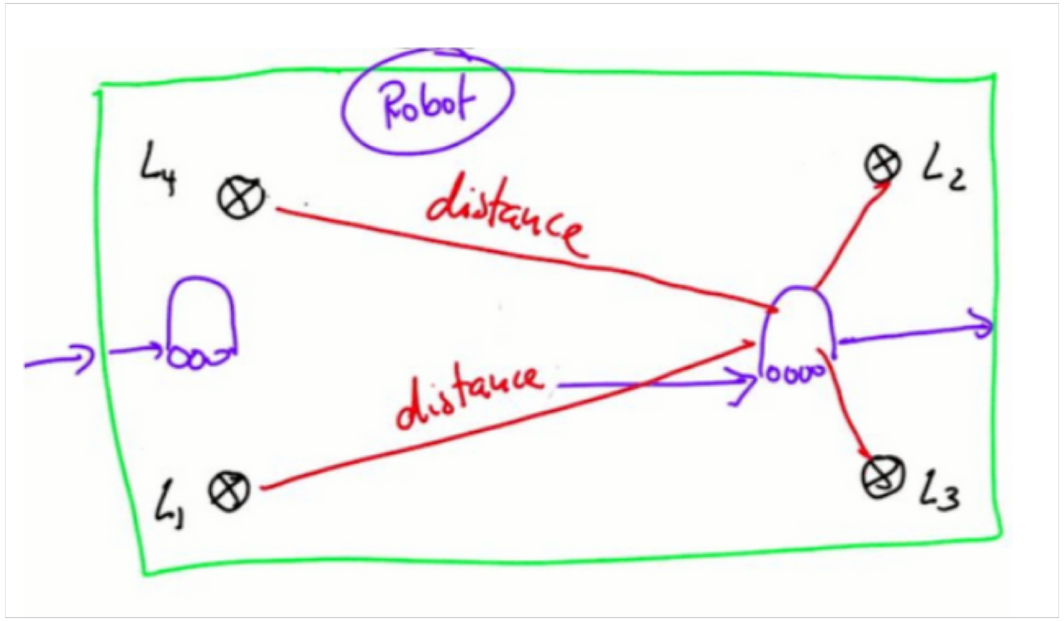


Particle Filter

1. Start with N randomly-scattered particles
   - Each particle is a "guess" of robot's position
   - Particle $p_i = (x_i, y_i, \theta_i)$

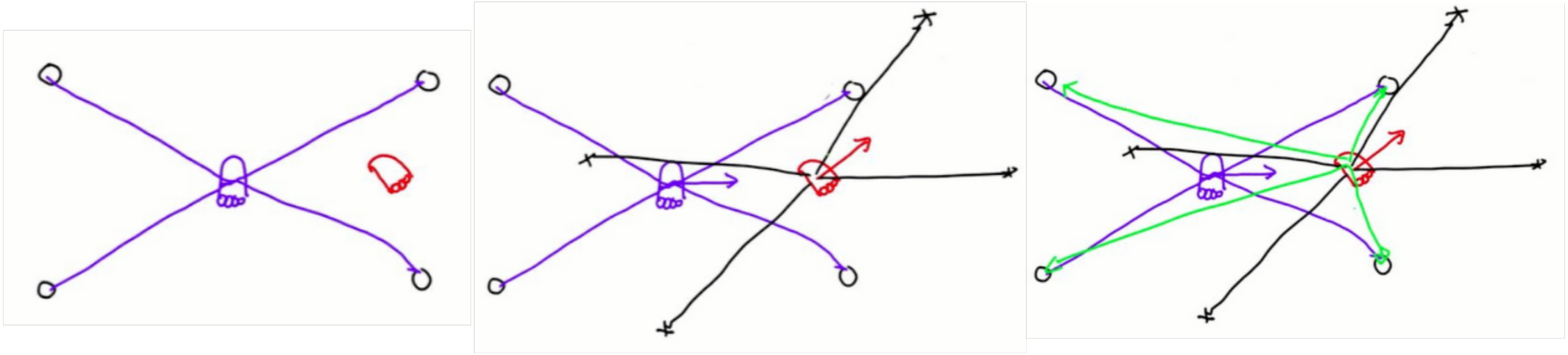
2. Motion





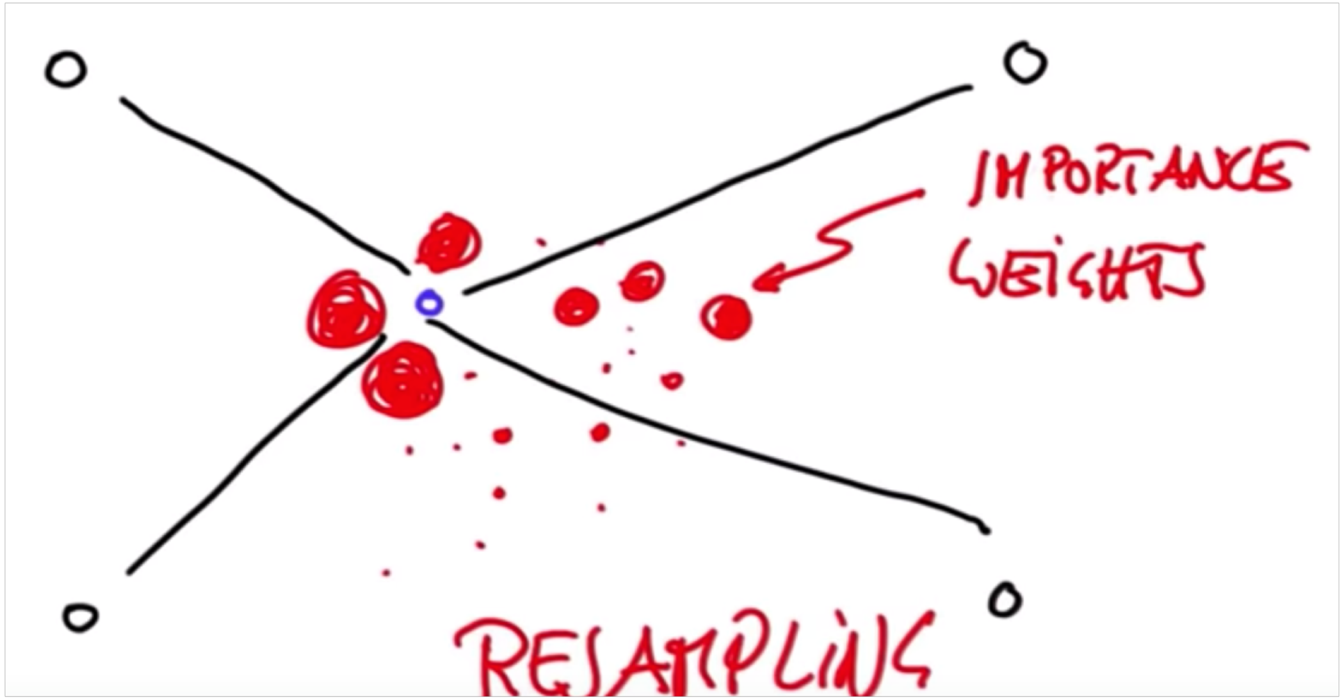
Particle Filter

3. Measure **each** particle's distance to **all** landmarks
    - Compare observed measurement (black) to expected measurement (green)



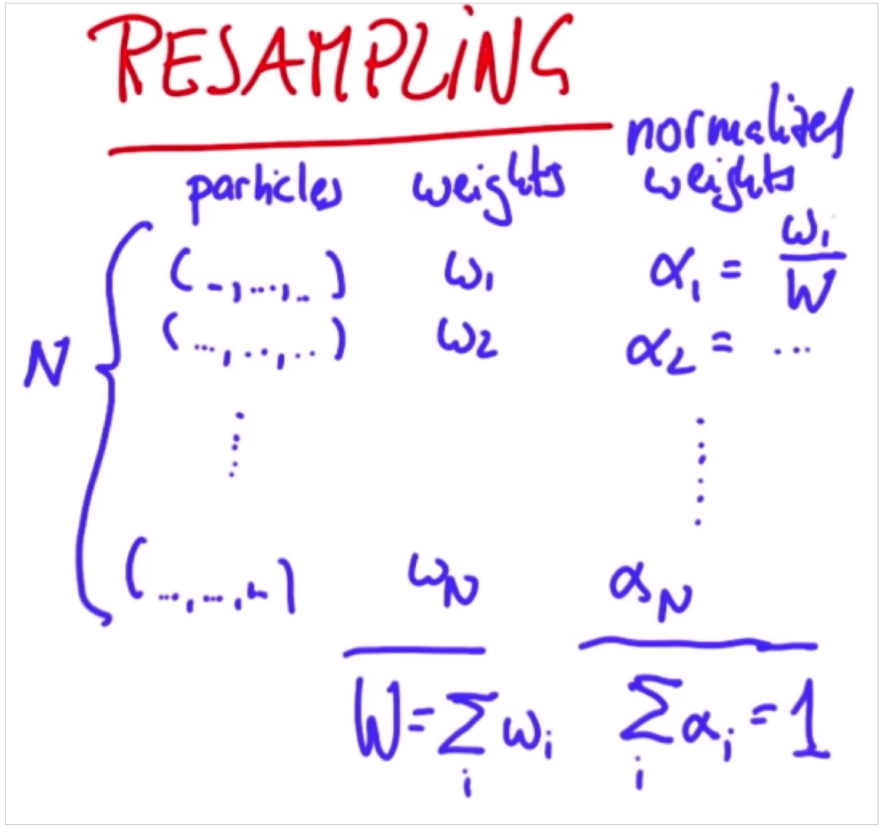
Particle Filter

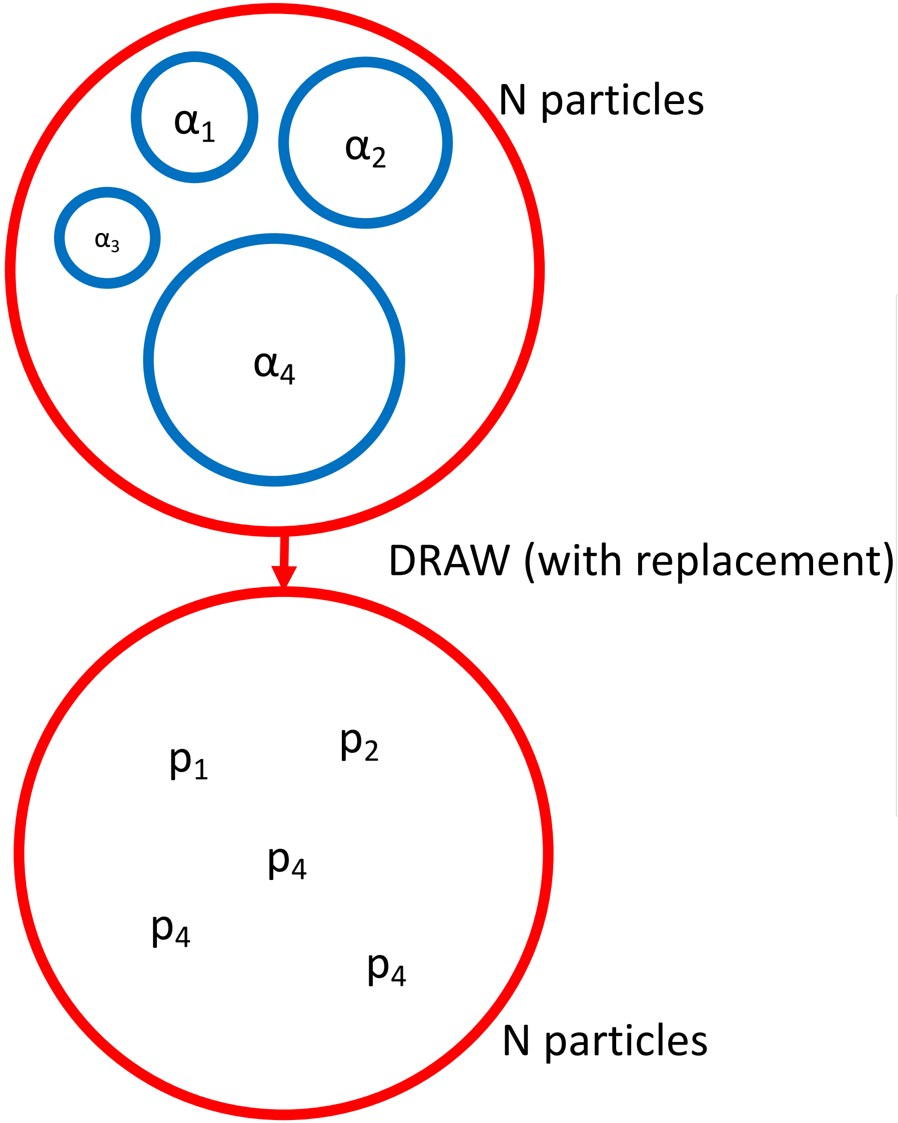
4. Calculate Importance Weights
   - Importance Weights = difference between actual and expected measurement

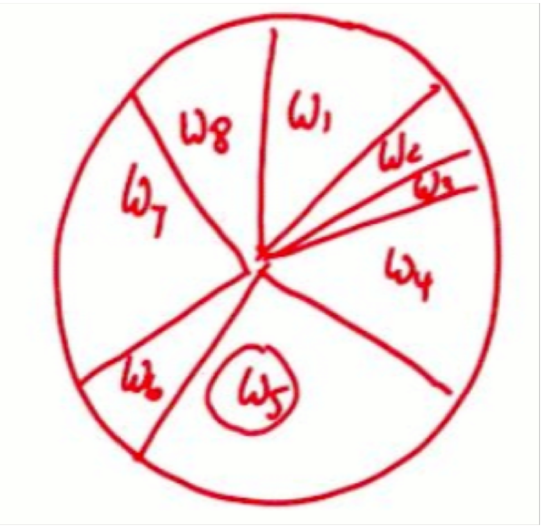


Particle Filter

5. Resampling
   - "Survival of the Fittest"



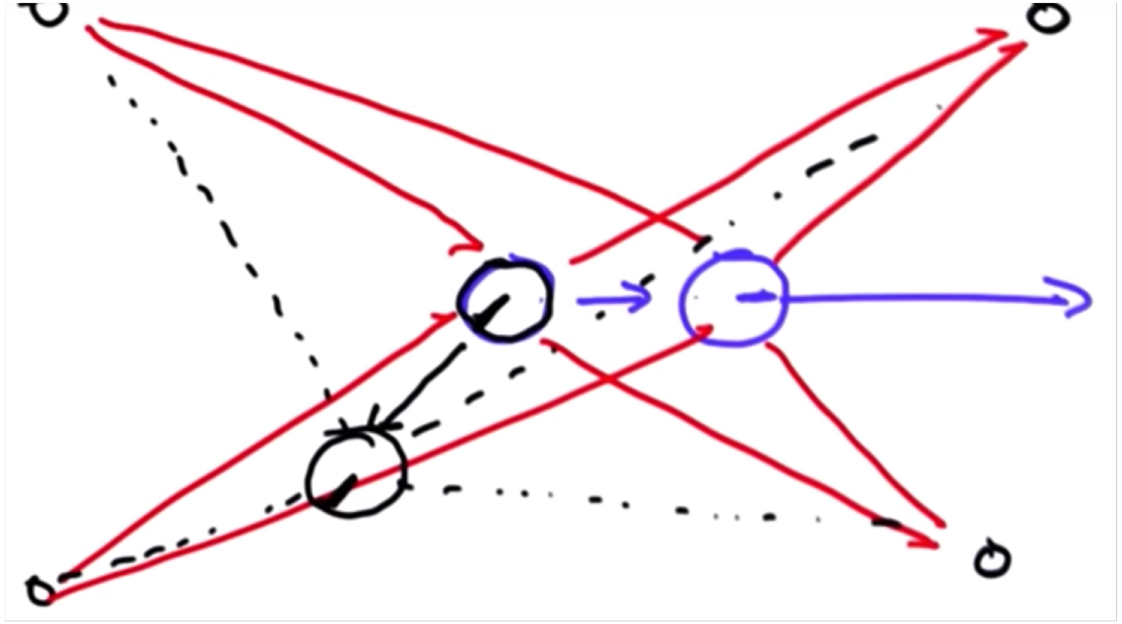




Particle Filter

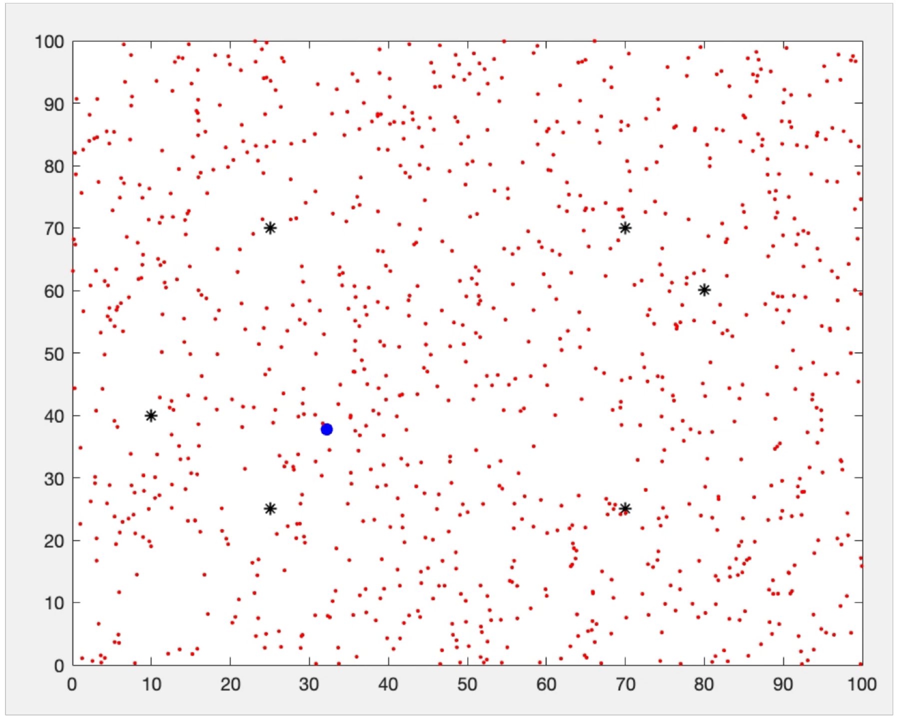
6. Move
   - Orientation now matters!

Note: Noise (motion and measurement) = all particles different



Particle Filter

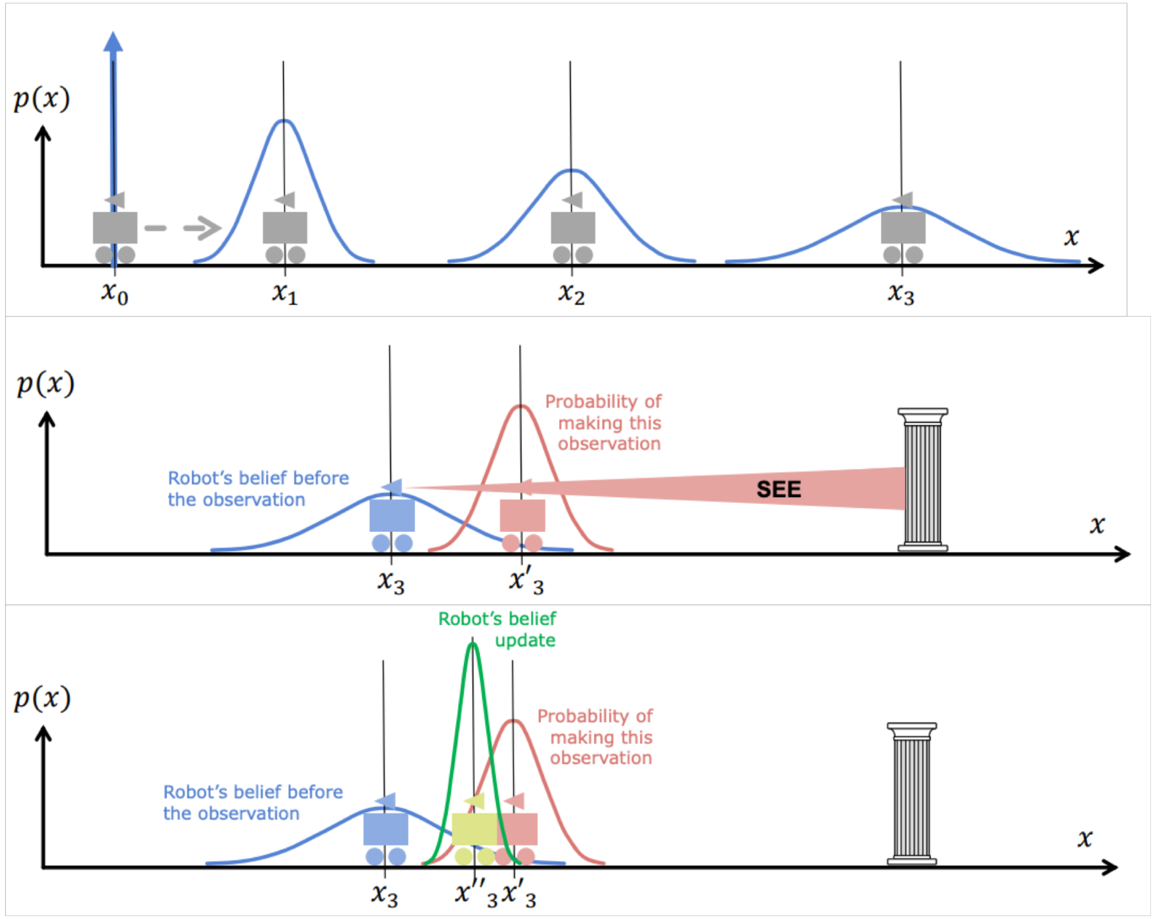
- **Pros**
  - Easy to Implement
  - Still relatively simple – just a bunch of particles!
  - Multimodal belief (job security term!)
    - We can represent belief for many locations
  - Many-dimensional
  - Continuous
  - Robust



- **Cons**
  - How does it scale for higher dimensions?
    - Not well!
  - Computationally expensive

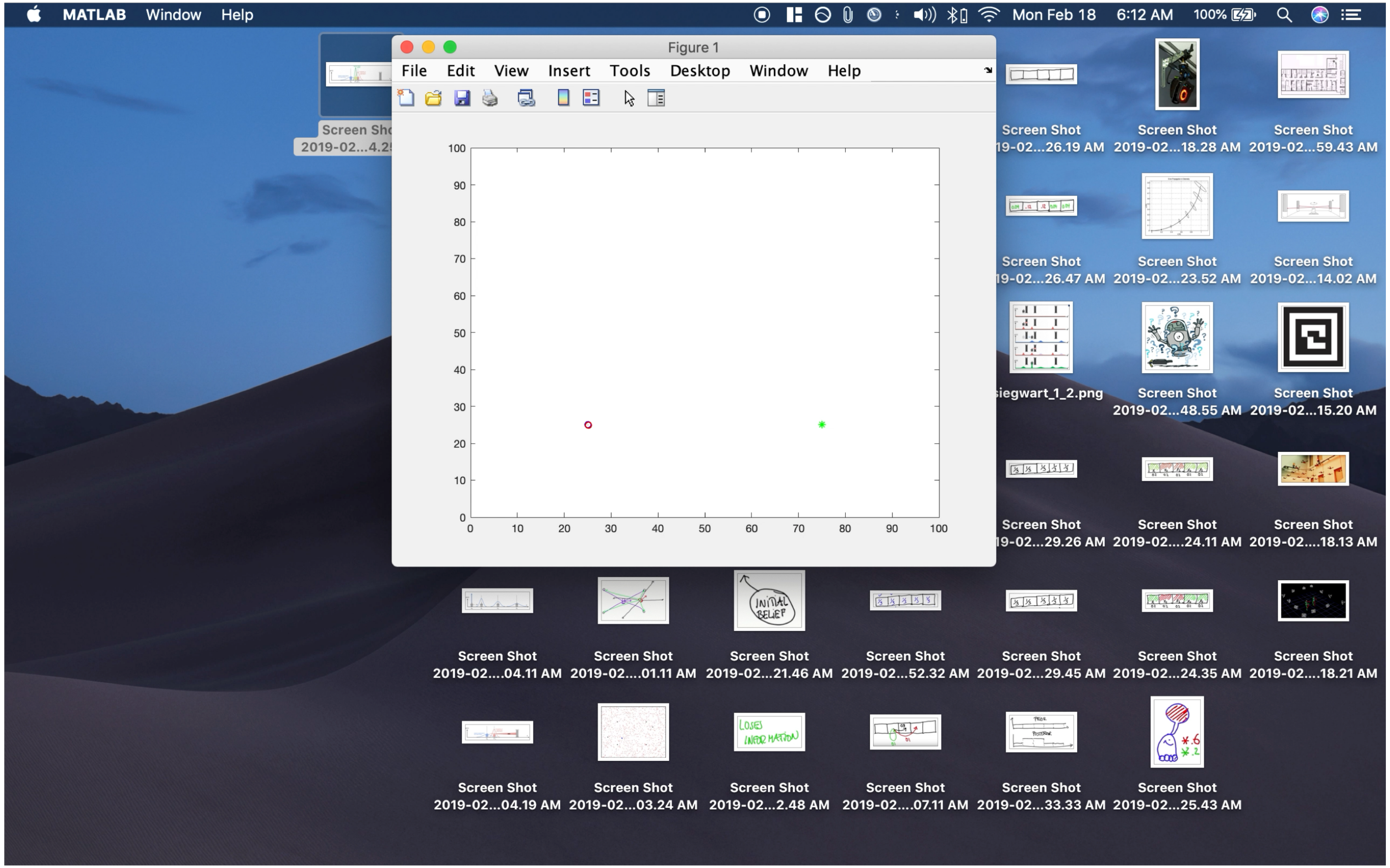
Kalman Filter

- Continuous pose representation (as opposed to discrete)
- Unimodal Gaussian (one peak = one estimate of location!)



Kalman Filter

https://home.wlu.edu/~levys/kalman_tutorial/kalman_14.html



Kalman Filter

- **Pros**
  - Many-dimensional
  - Continuous
  - Robust
    - Really good at dealing with *noise*!
  - **Sensor Fusion**
  - Computationally **in**expensive!
    - Just store the Gaussian and update that (Gaussian = mean & covariance)

- **Cons**
  - Complex, very "math-y"
  - Not the best for all use cases
  - Unimodal (single-peak)

Localization Summary

- Solve the problem of **State Estimation**
  - **State Estimation** = Process of determining the best value of some physical quantity (in this case, pose) from different noisy measurements

- All are instances of **Recursive Bayes Filter**

- Discrete vs Continuous
- Unimodal vs Multimodal
- Computational Efficiency
  - Storage and Compute

- Different use cases, combinations also used