Cognitive Robotics

08. Simultaneous Localization and Mapping (SLAM)

AbdElMoniem Bayoumi, PhD

Spring 2022

Recap: Mapping so far

- Mapping with **known poses** for a grid map representation (easy)
- **Occupancy grids**: each cell is a binary random variable estimating whether the cell is occupied
- Static state binary Bayes filter per cell
- **Reflection Maps:** store in each cell the probability that a beam is reflected by this cell
- Given the discussed sensor model, counting yields the maximum likelihood model

Difference between Occupancy Grid Maps and Reflection Maps

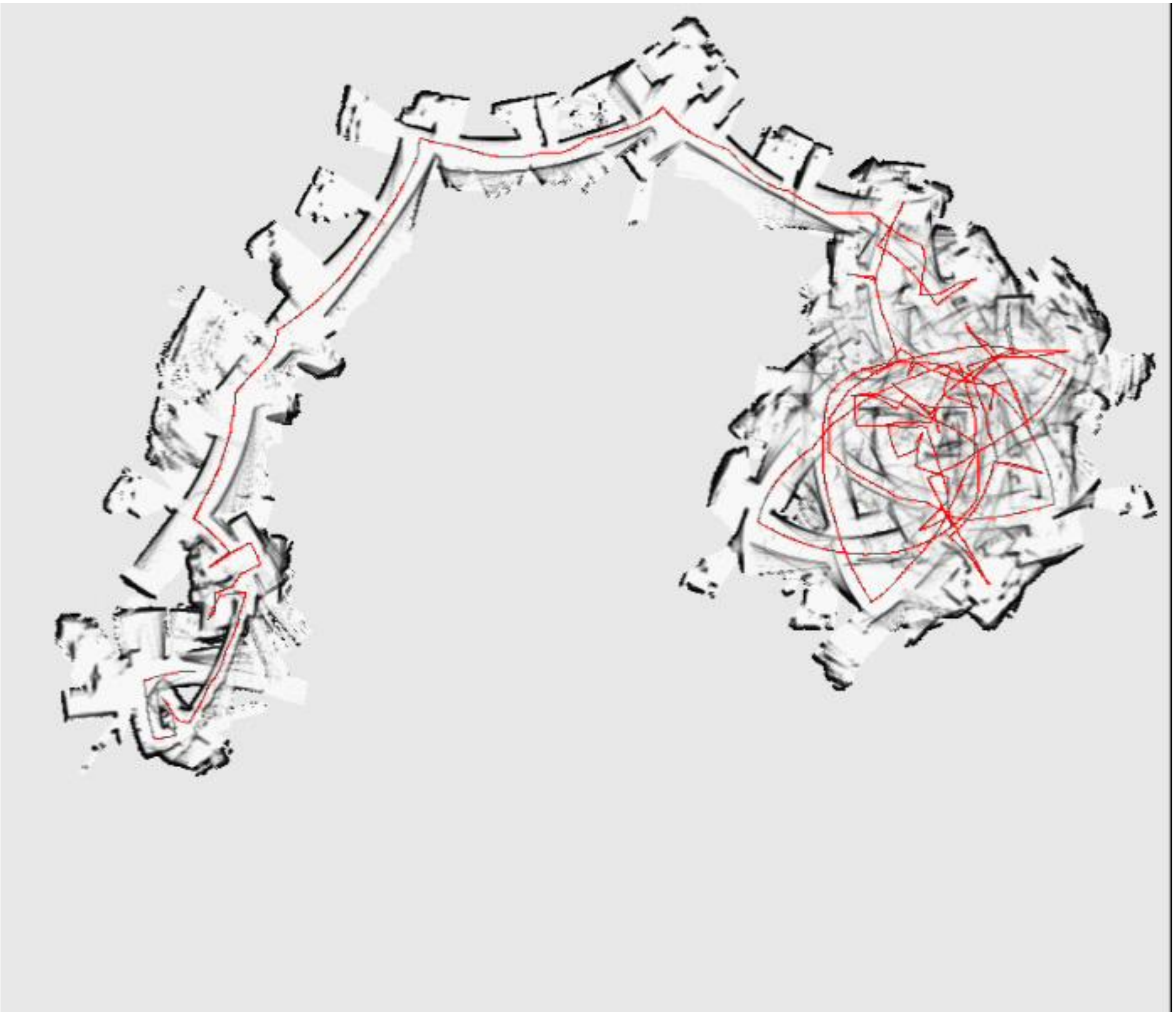
- The counting model determines how often a cell reflects a beam
- The occupancy model represents whether or not a cell is occupied by an object
- Although a cell might be occupied by an object, the reflection probability of this object might be very small

Recap: Example Occupancy Map

Recap: Example Reflection Map

Grid Mapping Meets Reality…

Mapping With Raw Odometry



Courtesy: D. Hähnel

Possible Solution: Incremental Scan Alignment

- Motion is noisy, we cannot ignore it
- In reality, the robot poses are not known
- Often, the sensor is rather precise (laser)
- Scan matching: **incrementally align two scans** or **a scan to a map**

Pose Correction Using Scan Matching

Maximize the likelihood of the **current** pose relative to the **previous** pose and map

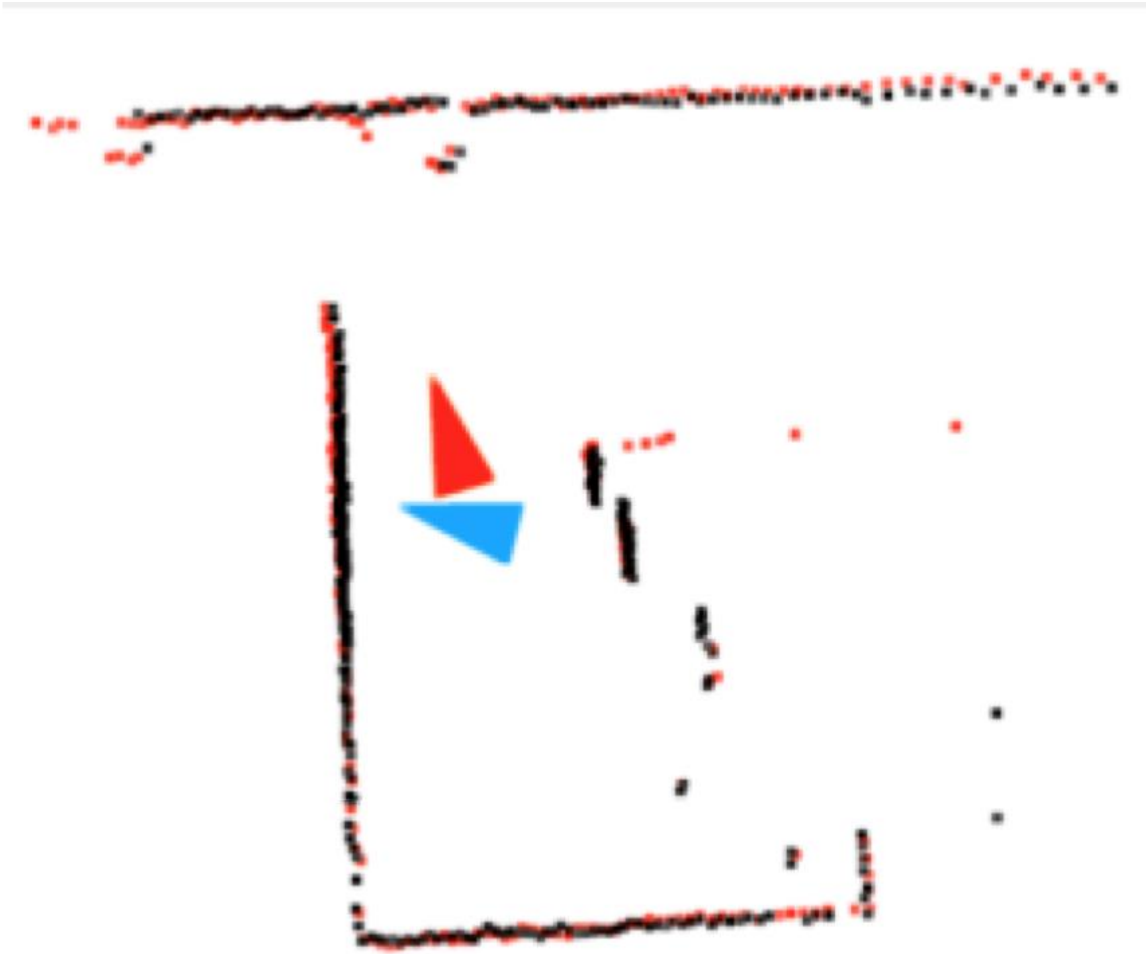
$$x_t^* = \underset{x_t}{\operatorname{argmax}} \left\{ p(z_t \mid x_t, m_{t-1}) \, p(x_t \mid u_{t-1}, x_{t-1}^*) \right\}$$

current measurement

robot motion

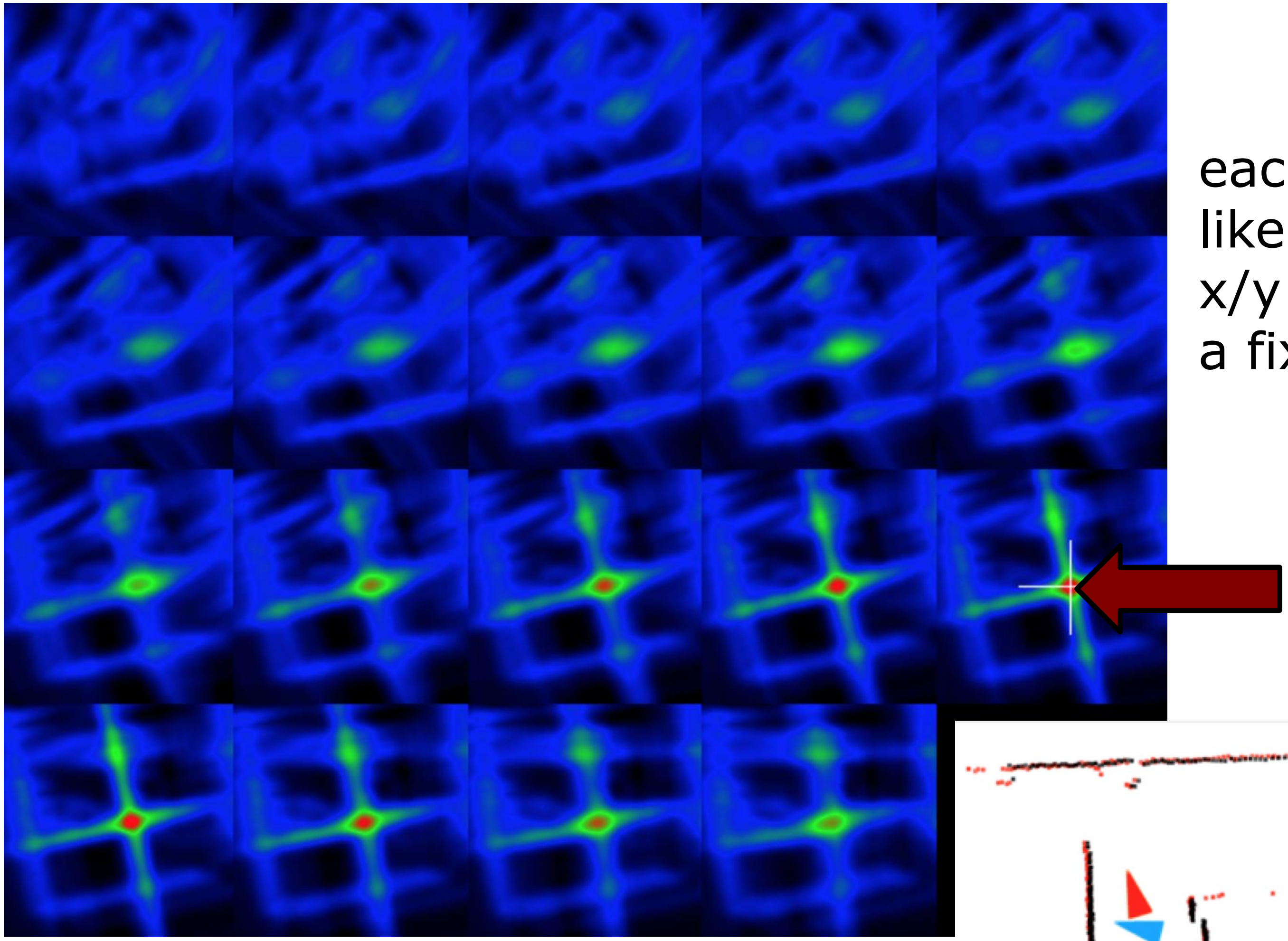
map constructed so far

Incremental Alignment

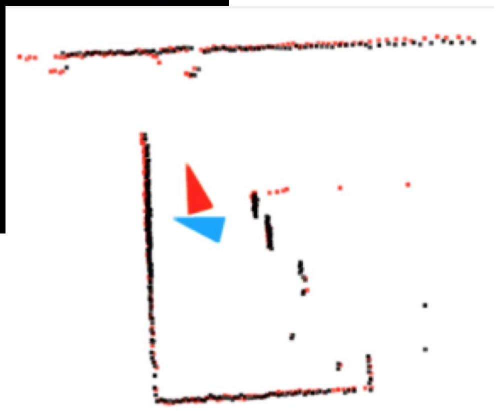


Courtesy: E. Olson

Incremental Alignment



each tile shows the likelihood for the x/y translation for a fixed orientation

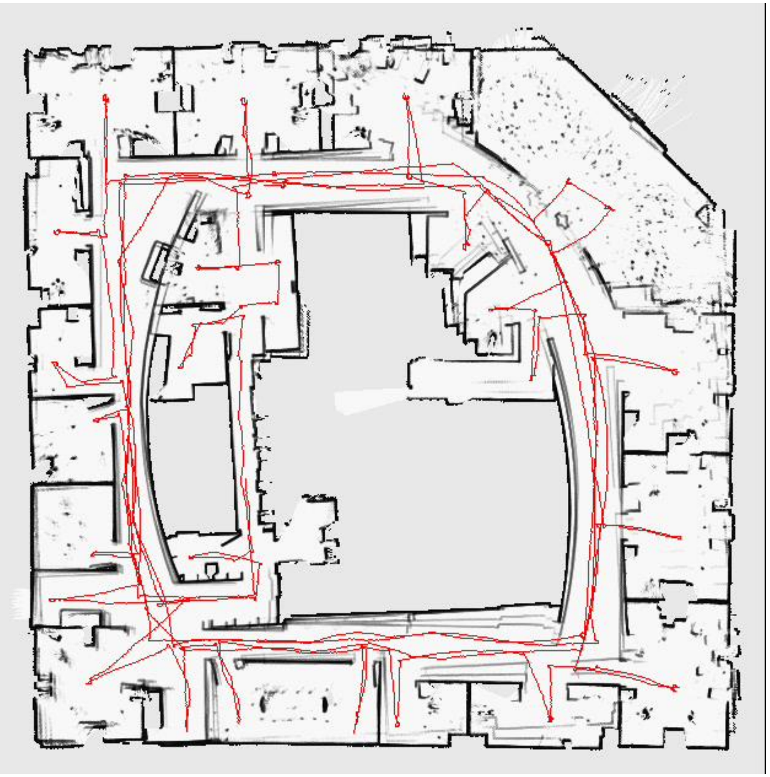


Courtesy: E. Olson

Various Different Ways to Realize Scan Matching

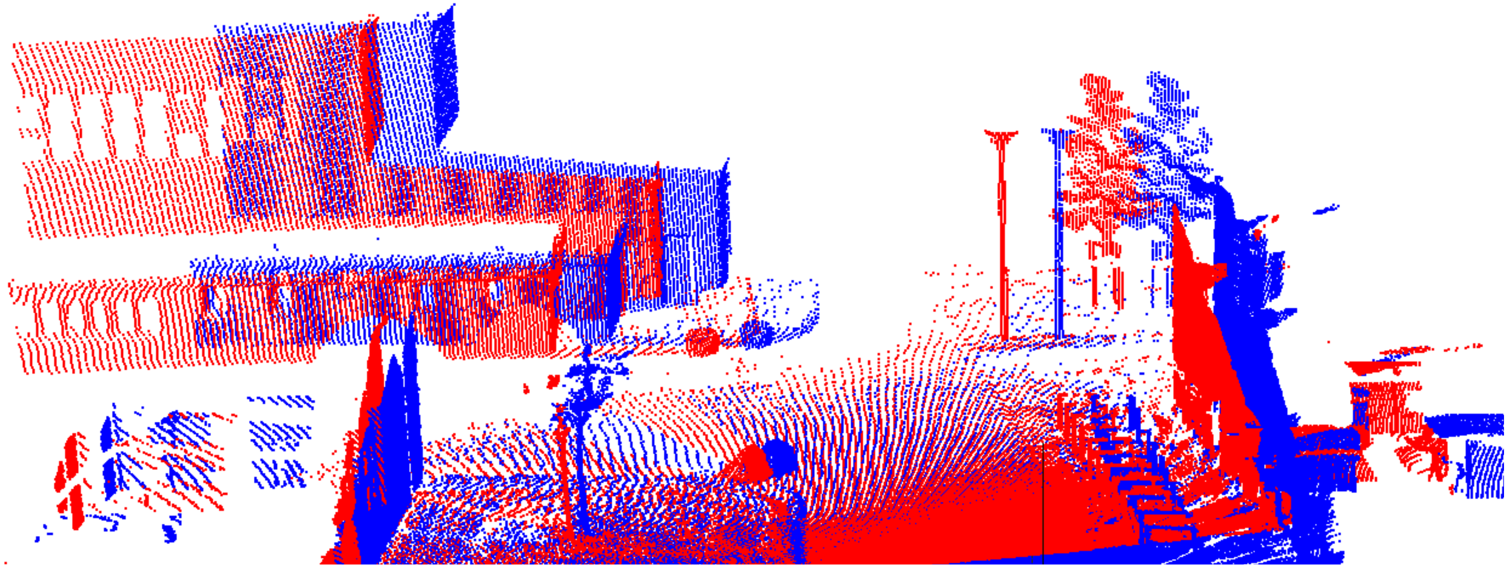
- Scan-to-scan
- Scan-to-map
- Map-to-map
- Iterative closest point (ICP)
- Feature-based
- …

With and Without Scan Matching



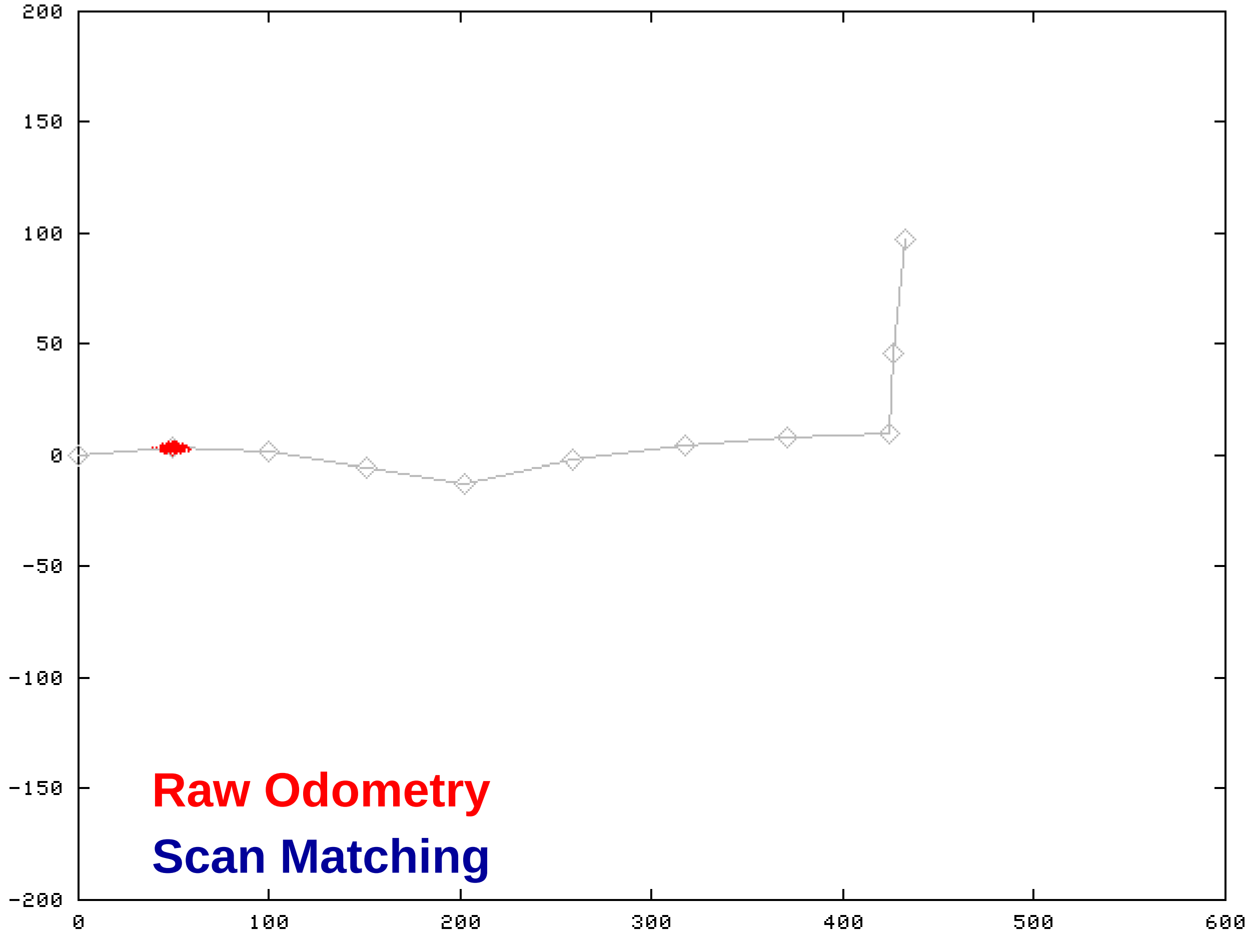
Courtesy: D. Hähnel

Example: Aligning in 3D



Courtesy: P. Pfaff

Motion Model for Scan Matching



Courtesy: D. Hähnel

Summary: Scan Matching

- Scan matching improves the pose estimate (and thus mapping) substantially
- Locally consistent estimates
- But: Often scan matching is not sufficient to build large consistent maps

SLAM

What is SLAM?

- Estimate the pose of a robot and the map of the environment at the same time
- SLAM is hard, because
  - a map is needed for localization and
  - a good pose estimate is needed for mapping

What is SLAM?

- Localization: inferring the robot's location within a given a map
- Mapping: inferring a map given sensor data from known robot locations
- **SLAM: learning a map and locating the robot simultaneously**

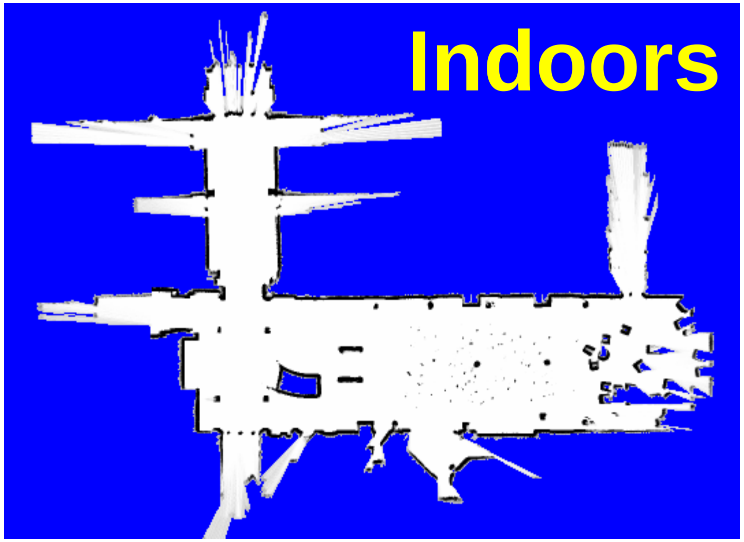
SLAM Applications

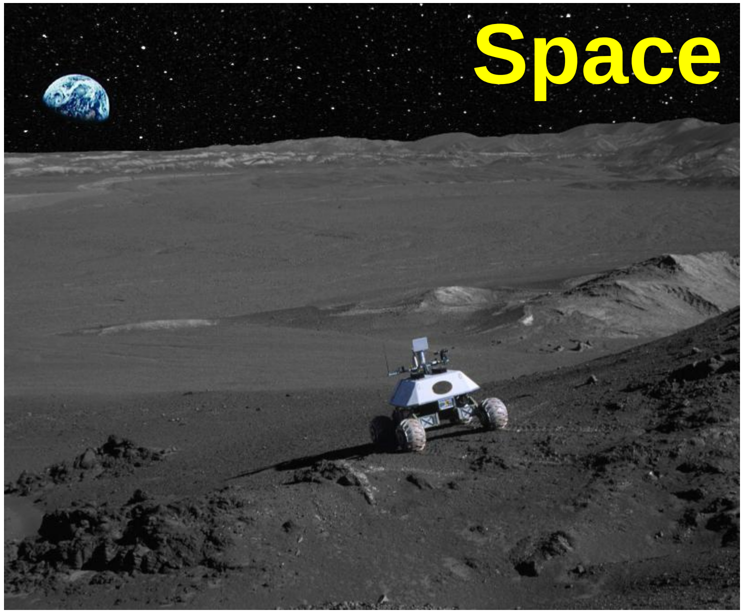
- SLAM is central to a range of indoor, outdoor, in-air, and underwater applications

Examples:

- At home: vacuum cleaner, lawn mower
- Surveillance with unmanned air vehicles
- Underwater: reef monitoring
- Underground: exploration of mines
- Space: terrain mapping

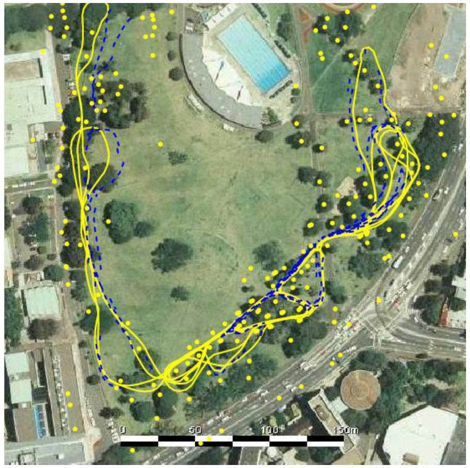
SLAM Applications

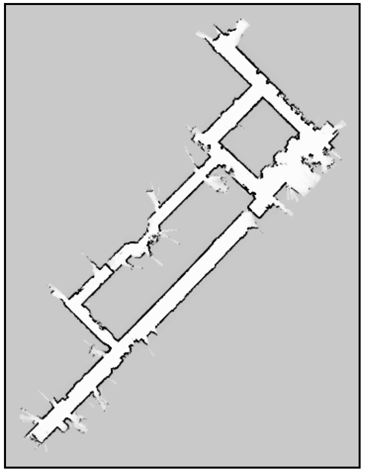


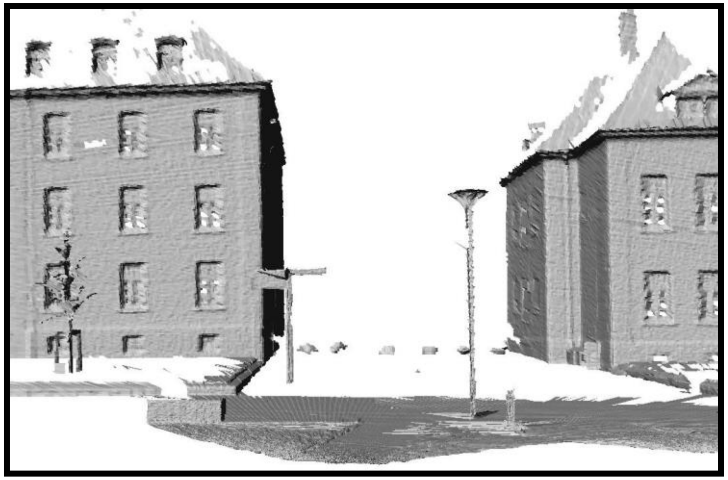


Typical Map Representations

Landmark-based or grid-based (2D or 3D) representations of the environment







The SLAM Problem

- SLAM is considered a fundamental problem for robots to become truly autonomous
- Large variety of SLAM approaches have been developed
- The majority uses probabilistic concepts
- History of SLAM dates back to the mid-eighties

Definition of the SLAM Problem

Given

- The robot's controls

$$u_{1:T} = \{u_1, u_2, u_3, \ldots, u_T\}$$

- Observations

$$z_{1:T} = \{z_1, z_2, z_3, \ldots, z_T\}$$

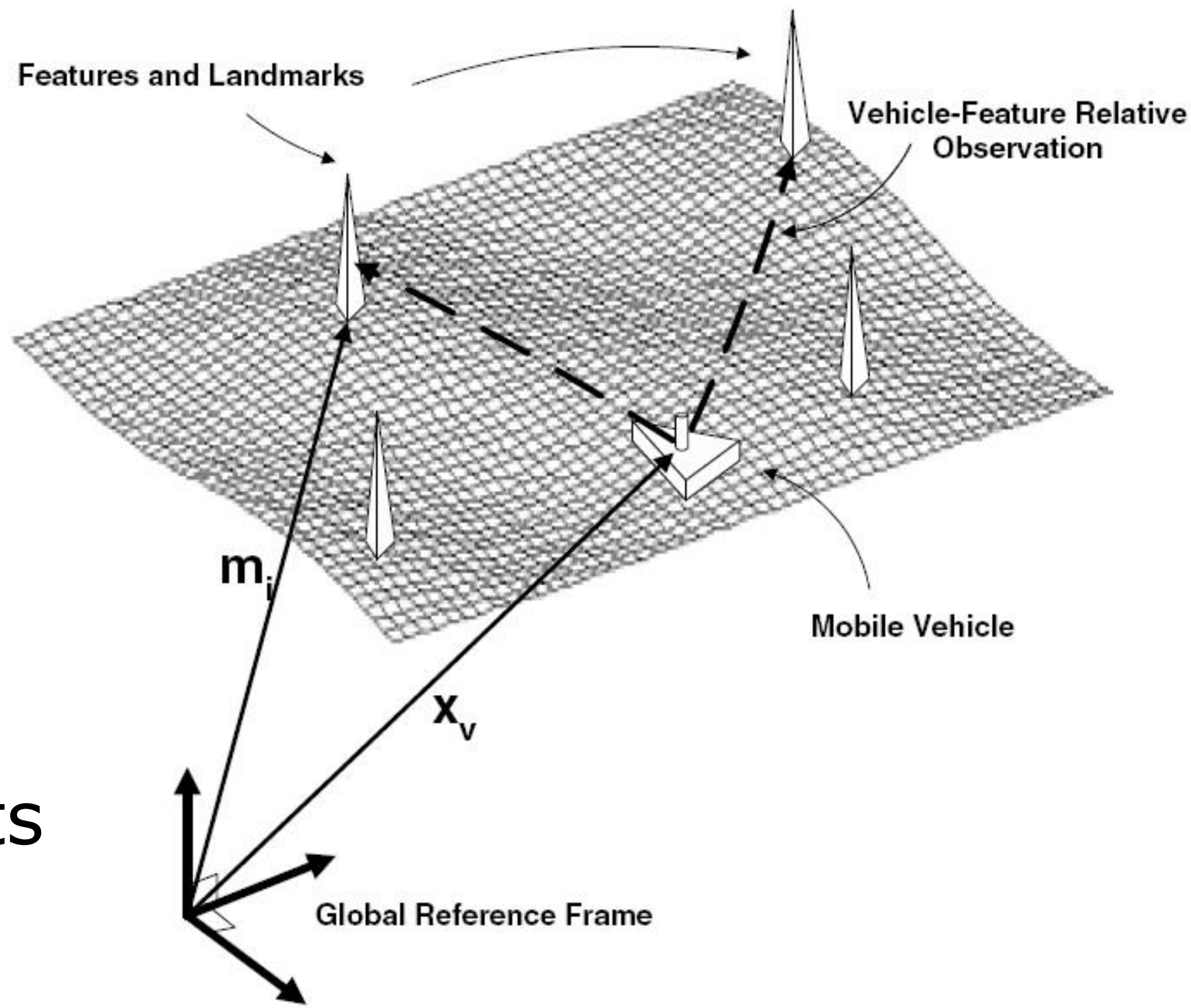
Wanted

- Map of the environment m
- Path of the robot

$$x_{0:T} = \{x_0, x_1, x_2, \ldots, x_T\}$$

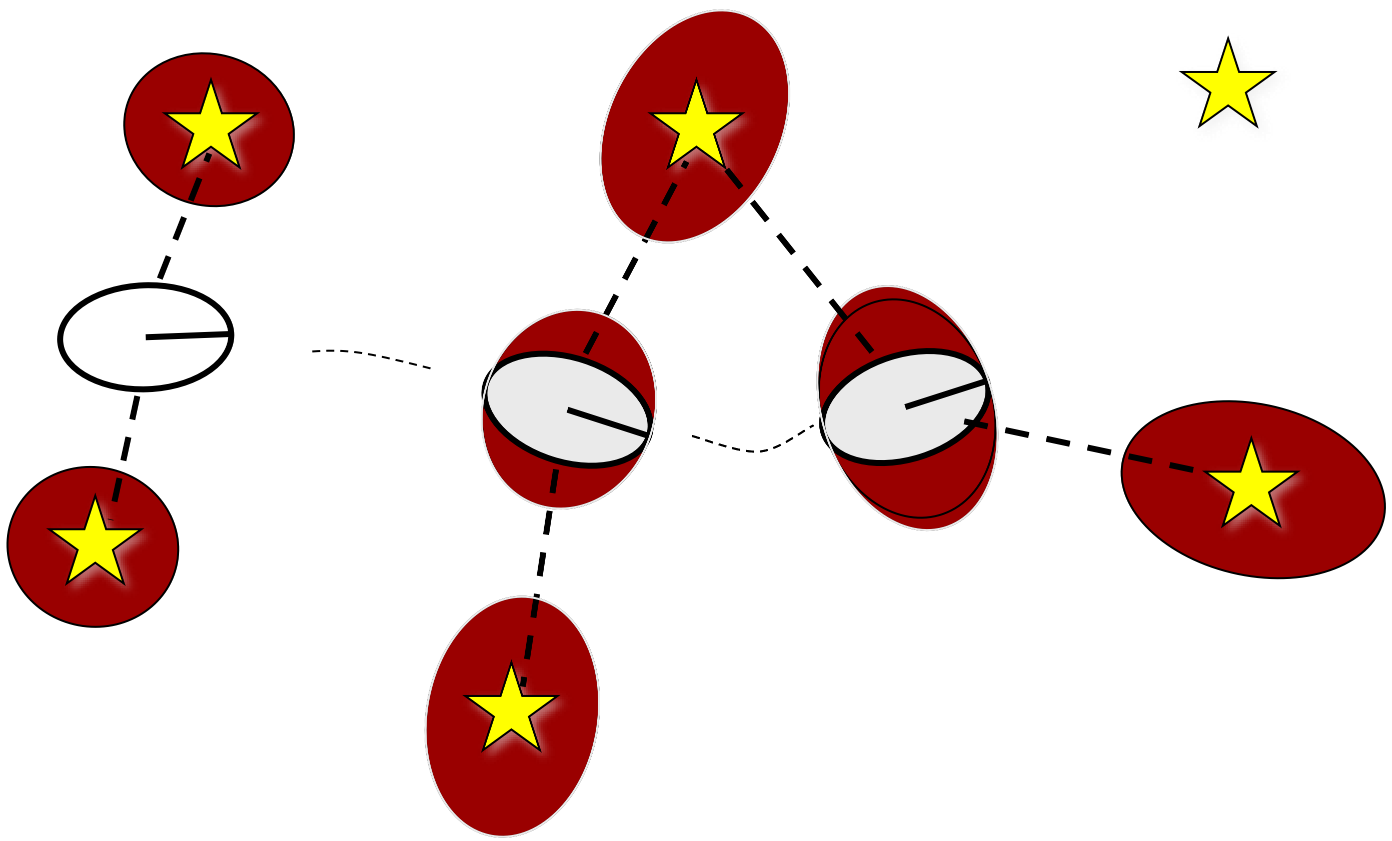
Feature-Based SLAM

- **Absolute** robot pose
- **Absolute** landmark positions
- But only **relative** measurements of landmarks



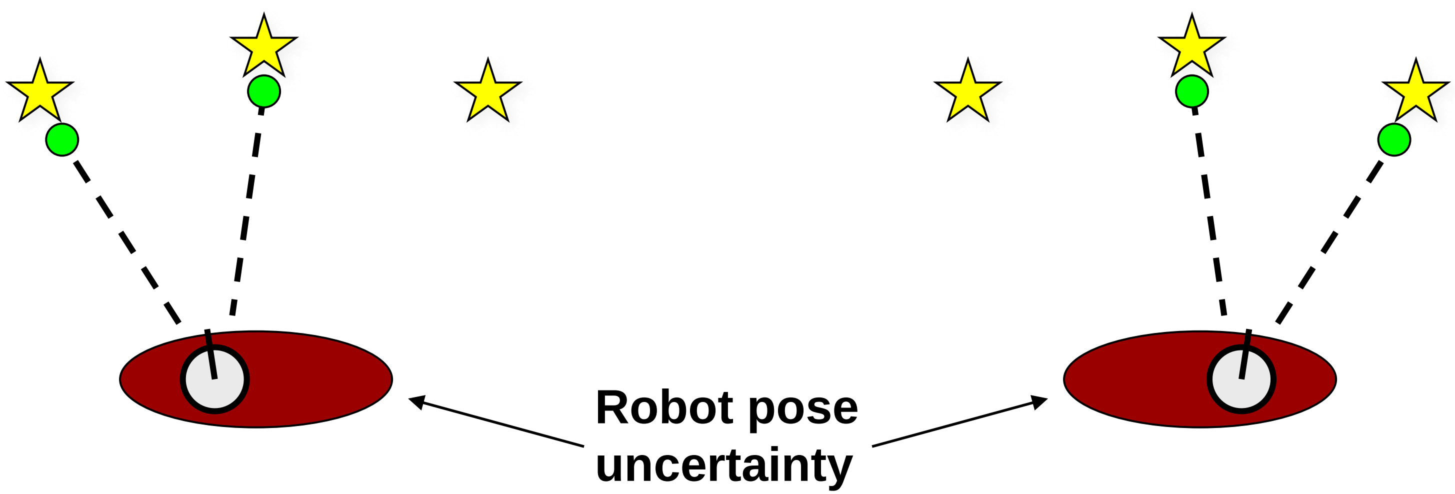
Why is SLAM a Hard Problem?

- Robot path and map are both **unknown**
- Errors in map and pose estimates **correlated**



Data Association

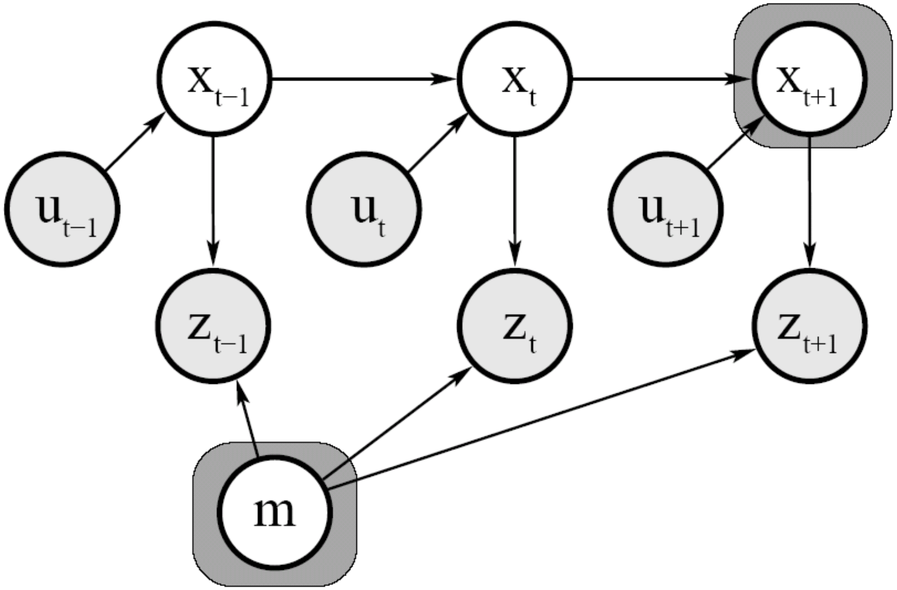
- The **mapping between observations and landmarks is unknown**
- Picking **wrong** data associations can have **catastrophic** consequences (divergence)



EKF for Online SLAM

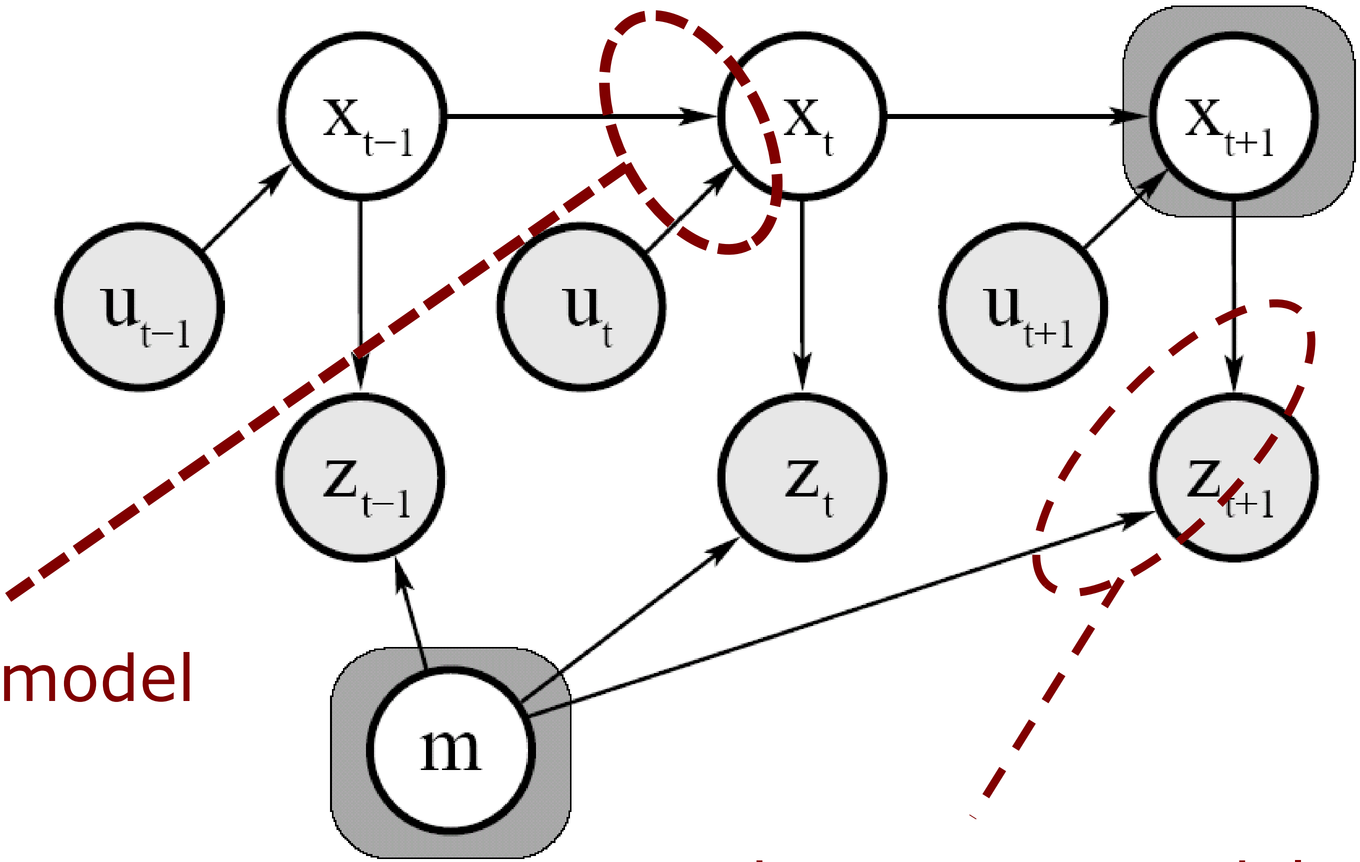
- Kalman filter as a solution to the online SLAM problem
- Estimate the most recent pose and map

$$p(x_t, m \mid z_{1:t}, u_{1:t})$$



Courtesy: Thrun, Burgard, Fox

Motion and Observation Model



motion model

observation model

Recap: KF Algorithm

1. Algorithm **Kalman_filter**(μ_{t-1}, Σ_{t-1}, u_t, z_t):
2. Prediction:
3. $\bar{\mu}_t = A_t \mu_{t-1} + B_t u_t$ motion model
4. $\bar{\Sigma}_t = A_t \Sigma_{t-1} A_t^T + R_t$
5. Correction:
6. $K_t = \bar{\Sigma}_t C_t^T (C_t \bar{\Sigma}_t C_t^T + Q_t)^{-1}$
7. $\mu_t = \bar{\mu}_t + K_t (z_t - C_t \bar{\mu}_t)$ sensor model
8. $\Sigma_t = (I - K_t C_t) \bar{\Sigma}_t$
9. Return μ_t, Σ_t

EKF SLAM

- Application of the EKF to SLAM
- Estimate robot's pose **and landmark locations**
- Assumption: known correspondences
- State space (for the 2D plane):

$$x_t = (\underbrace{x, y, \theta}_{\text{robot's pose}}, \underbrace{m_{1,x}, m_{1,y}}_{\text{landmark 1}}, \ldots, \underbrace{m_{n,x}, m_{n,y}}_{\text{landmark n}})^T$$

EKF SLAM: State Representation

- Map with *n* landmarks: (3+2*n*)-dimensional Gaussian
- Belief is represented by

$$\underbrace{\begin{pmatrix} x \\ y \\ \theta \\ m_{1,x} \\ m_{1,y} \\ \vdots \\ m_{n,x} \\ m_{n,y} \end{pmatrix}}_{\mu} \quad \underbrace{\begin{pmatrix} \sigma_{xx} & \sigma_{xy} & \sigma_{x\theta} & \sigma_{xm_{1,x}} & \sigma_{xm_{1,y}} & \cdots & \sigma_{xm_{n,x}} & \sigma_{xm_{n,y}} \\ \sigma_{yx} & \sigma_{yy} & \sigma_{y\theta} & \sigma_{ym_{1,x}} & \sigma_{ym_{1,y}} & \cdots & \sigma_{m_{n,x}} & \sigma_{m_{n,y}} \\ \sigma_{\theta x} & \sigma_{\theta y} & \sigma_{\theta\theta} & \sigma_{\theta m_{1,x}} & \sigma_{\theta m_{1,y}} & \cdots & \sigma_{\theta m_{n,x}} & \sigma_{\theta m_{n,y}} \\ \sigma_{m_{1,x}x} & \sigma_{m_{1,x}y} & \sigma_{\theta} & \sigma_{m_{1,x}m_{1,x}} & \sigma_{m_{1,x}m_{1,y}} & \cdots & \sigma_{m_{1,x}m_{n,x}} & \sigma_{m_{1,x}m_{n,y}} \\ \sigma_{m_{1,y}x} & \sigma_{m_{1,y}y} & \sigma_{\theta} & \sigma_{m_{1,y}m_{1,x}} & \sigma_{m_{1,y}m_{1,y}} & \cdots & \sigma_{m_{1,y}m_{n,x}} & \sigma_{m_{1,y}m_{n,y}} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ \sigma_{m_{n,x}x} & \sigma_{m_{n,x}y} & \sigma_{\theta} & \sigma_{m_{n,x}m_{1,x}} & \sigma_{m_{n,x}m_{1,y}} & \cdots & \sigma_{m_{n,x}m_{n,x}} & \sigma_{m_{n,x}m_{n,y}} \\ \sigma_{m_{n,y}x} & \sigma_{m_{n,y}y} & \sigma_{\theta} & \sigma_{m_{n,y}m_{1,x}} & \sigma_{m_{n,y}m_{1,y}} & \cdots & \sigma_{m_{n,y}m_{n,x}} & \sigma_{m_{n,y}m_{n,y}} \end{pmatrix}}_{\Sigma}$$

EKF SLAM: State Representation

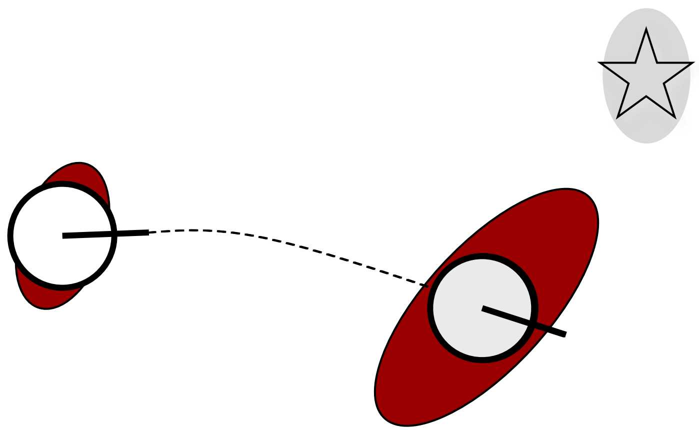
More compactly:

$$\underbrace{\begin{pmatrix} x_R \\ m_1 \\ \vdots \\ m_n \end{pmatrix}}_{\mu} \quad \underbrace{\begin{pmatrix} \Sigma_{x_R x_R} & \Sigma_{x_R m_1} & \dots & \Sigma_{x_R m_n} \\ \Sigma_{m_1 x_R} & \Sigma_{m_1 m_1} & \dots & \Sigma_{m_1 m_n} \\ \vdots & \vdots & \ddots & \vdots \\ \Sigma_{m_n x_R} & \Sigma_{m_n m_1} & \dots & \Sigma_{m_n m_n} \end{pmatrix}}_{\Sigma}$$

EKF SLAM: Filter Cycle

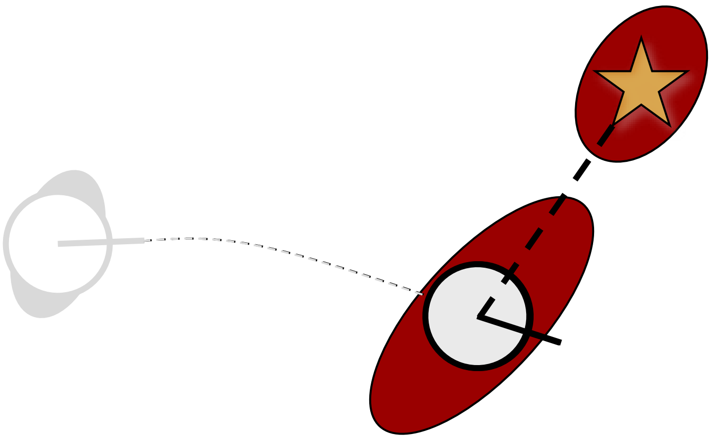
1. State prediction
2. Measurement prediction
3. Measurement
4. Data association
5. Update

EKF SLAM: State Prediction



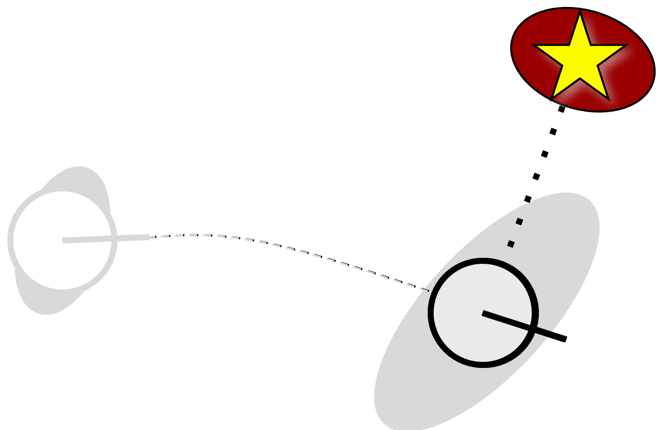
$$\underbrace{\begin{pmatrix} x_R \\ m_1 \\ \vdots \\ m_n \end{pmatrix}}_{\mu} \quad \underbrace{\begin{pmatrix} \Sigma_{x_R x_R} & \Sigma_{x_R m_1} & \cdots & \Sigma_{x_R m_n} \\ \Sigma_{m_1 x_R} & \Sigma_{m_1 m_1} & \cdots & \Sigma_{m_1 m_n} \\ \vdots & \vdots & \ddots & \vdots \\ \Sigma_{m_n x_R} & \Sigma_{m_n m_1} & \cdots & \Sigma_{m_n m_n} \end{pmatrix}}_{\Sigma}$$

EKF SLAM: Measurement Prediction



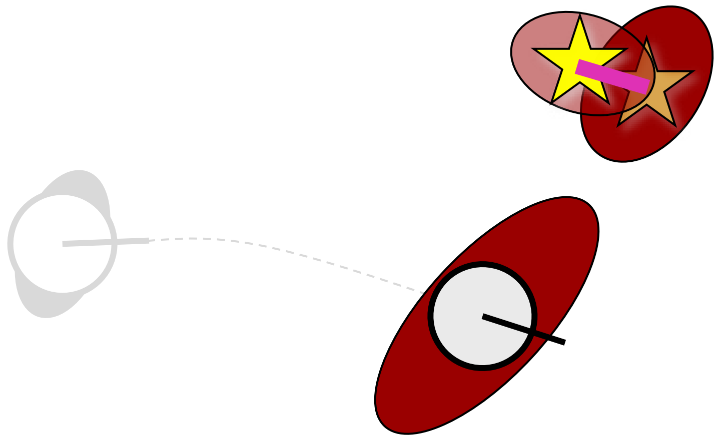
$$\underbrace{\begin{pmatrix} x_R \\ m_1 \\ \vdots \\ m_n \end{pmatrix}}_{\mu} \quad \underbrace{\begin{pmatrix} \Sigma_{x_R x_R} & \Sigma_{x_R m_1} & \cdots & \Sigma_{x_R m_n} \\ \Sigma_{m_1 x_R} & \Sigma_{m_1 m_1} & \cdots & \Sigma_{m_1 m_n} \\ \vdots & \vdots & \ddots & \vdots \\ \Sigma_{m_n x_R} & \Sigma_{m_n m_1} & \cdots & \Sigma_{m_n m_n} \end{pmatrix}}_{\Sigma}$$

EKF SLAM: Obtained Measurement



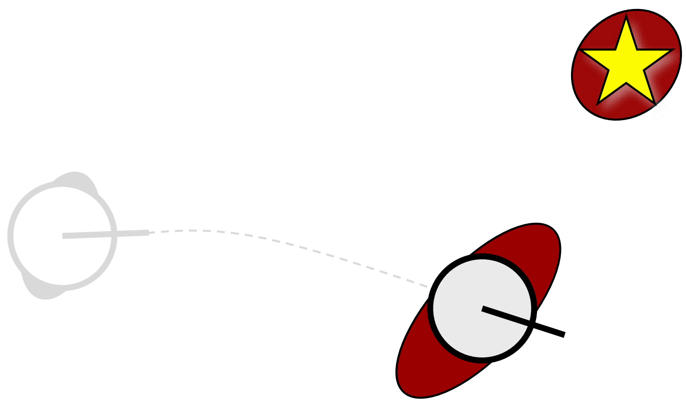
$$\underbrace{\begin{pmatrix} x_R \\ m_1 \\ \vdots \\ m_n \end{pmatrix}}_{\mu} \quad \underbrace{\begin{pmatrix} \Sigma_{x_R x_R} & \Sigma_{x_R m_1} & \cdots & \Sigma_{x_R m_n} \\ \Sigma_{m_1 x_R} & \Sigma_{m_1 m_1} & \cdots & \Sigma_{m_1 m_n} \\ \vdots & \vdots & \ddots & \vdots \\ \Sigma_{m_n x_R} & \Sigma_{m_n m_1} & \cdots & \Sigma_{m_n m_n} \end{pmatrix}}_{\Sigma}$$

EKF SLAM: Data Association and Difference to Predicted Observ.



$$\underbrace{\begin{pmatrix} x_R \\ m_1 \\ \vdots \\ m_n \end{pmatrix}}_{\mu} \quad \underbrace{\begin{pmatrix} \Sigma_{x_R x_R} & \Sigma_{x_R m_1} & \cdots & \Sigma_{x_R m_n} \\ \Sigma_{m_1 x_R} & \Sigma_{m_1 m_1} & \cdots & \Sigma_{m_1 m_n} \\ \vdots & \vdots & \ddots & \vdots \\ \Sigma_{m_n x_R} & \Sigma_{m_n m_1} & \cdots & \Sigma_{m_n m_n} \end{pmatrix}}_{\Sigma}$$

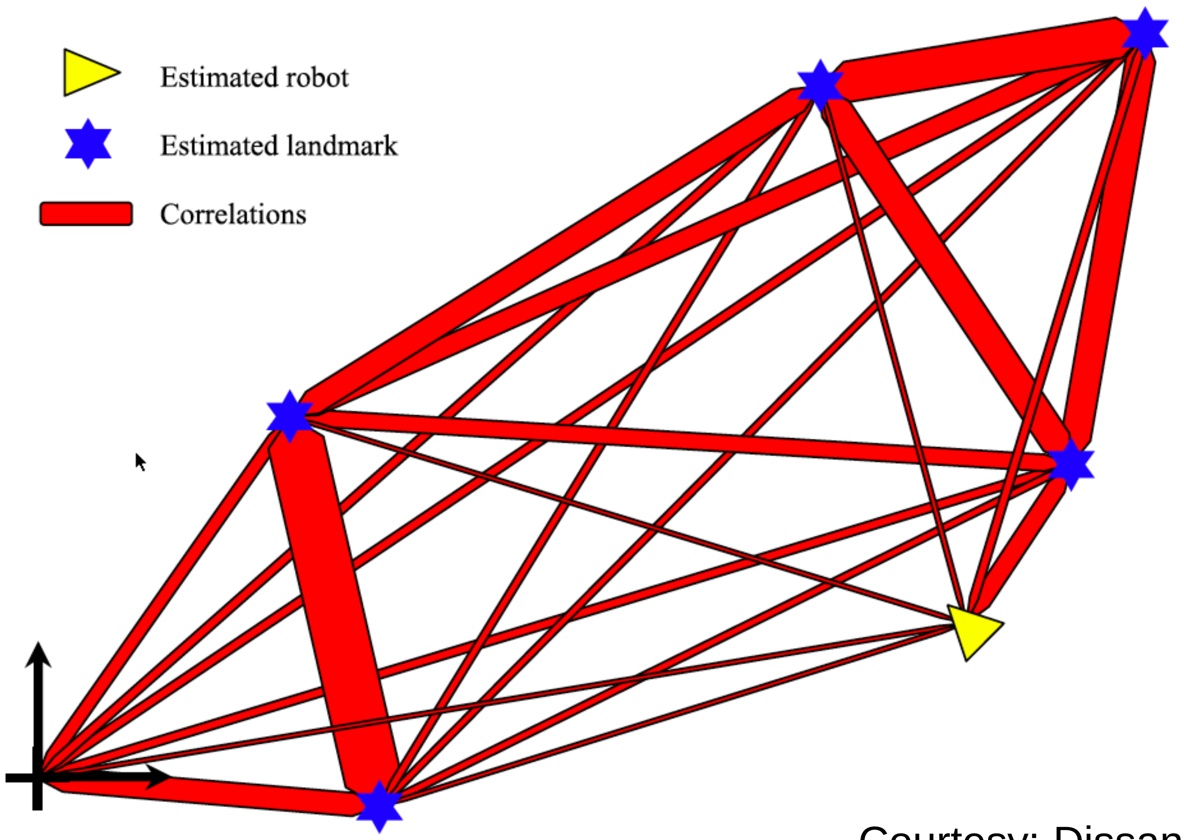
EKF SLAM: Update Step



$$\underbrace{\begin{pmatrix} x_R \\ m_1 \\ \vdots \\ m_n \end{pmatrix}}_{\mu} \quad \underbrace{\begin{pmatrix} \Sigma_{x_R x_R} & \Sigma_{x_R m_1} & \cdots & \Sigma_{x_R m_n} \\ \Sigma_{m_1 x_R} & \Sigma_{m_1 m_1} & \cdots & \Sigma_{m_1 m_n} \\ \vdots & \vdots & \ddots & \vdots \\ \Sigma_{m_n x_R} & \Sigma_{m_n m_1} & \cdots & \Sigma_{m_n m_n} \end{pmatrix}}_{\Sigma}$$

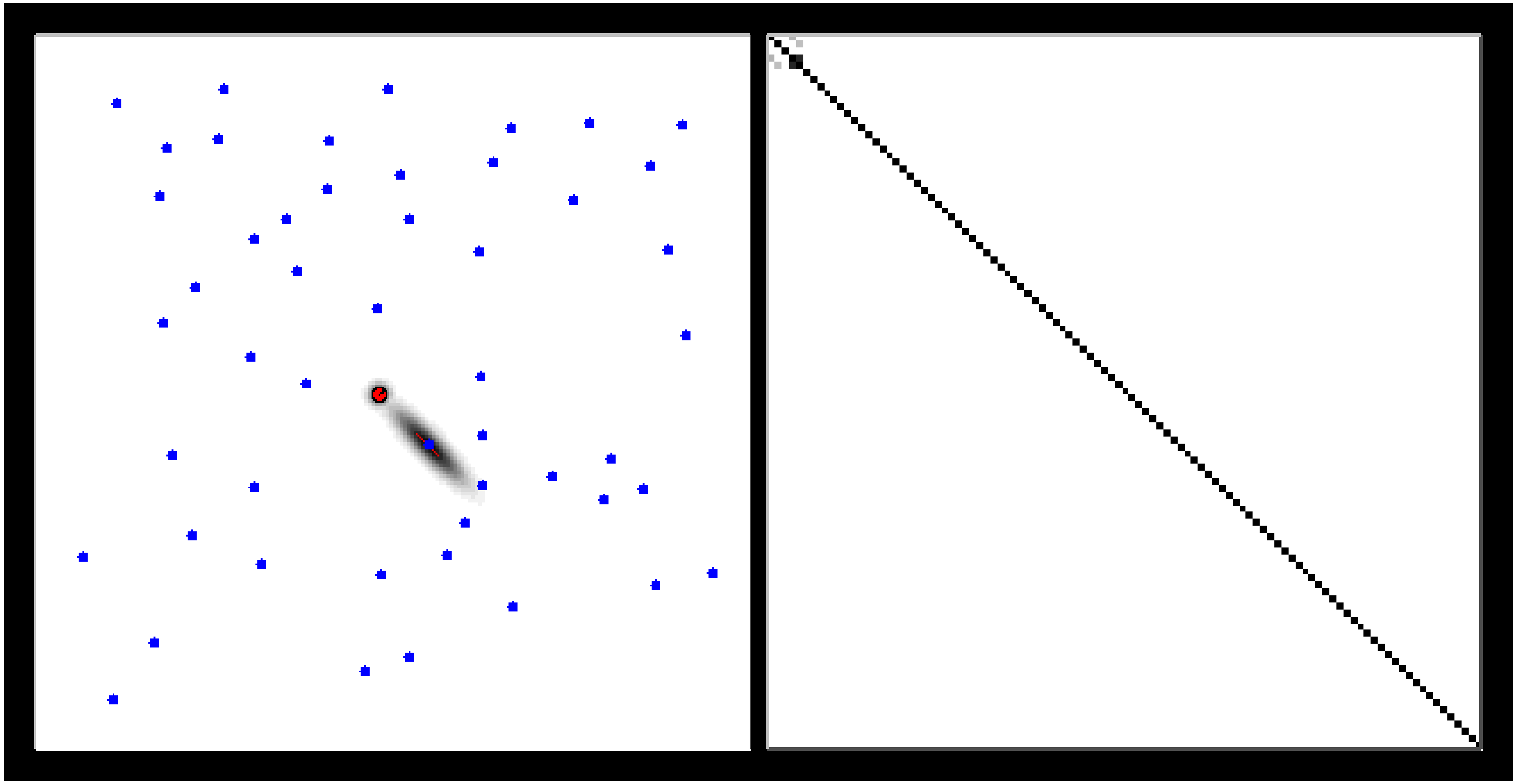
EKF SLAM Correlations

Over time, the landmark estimates become **fully correlated**



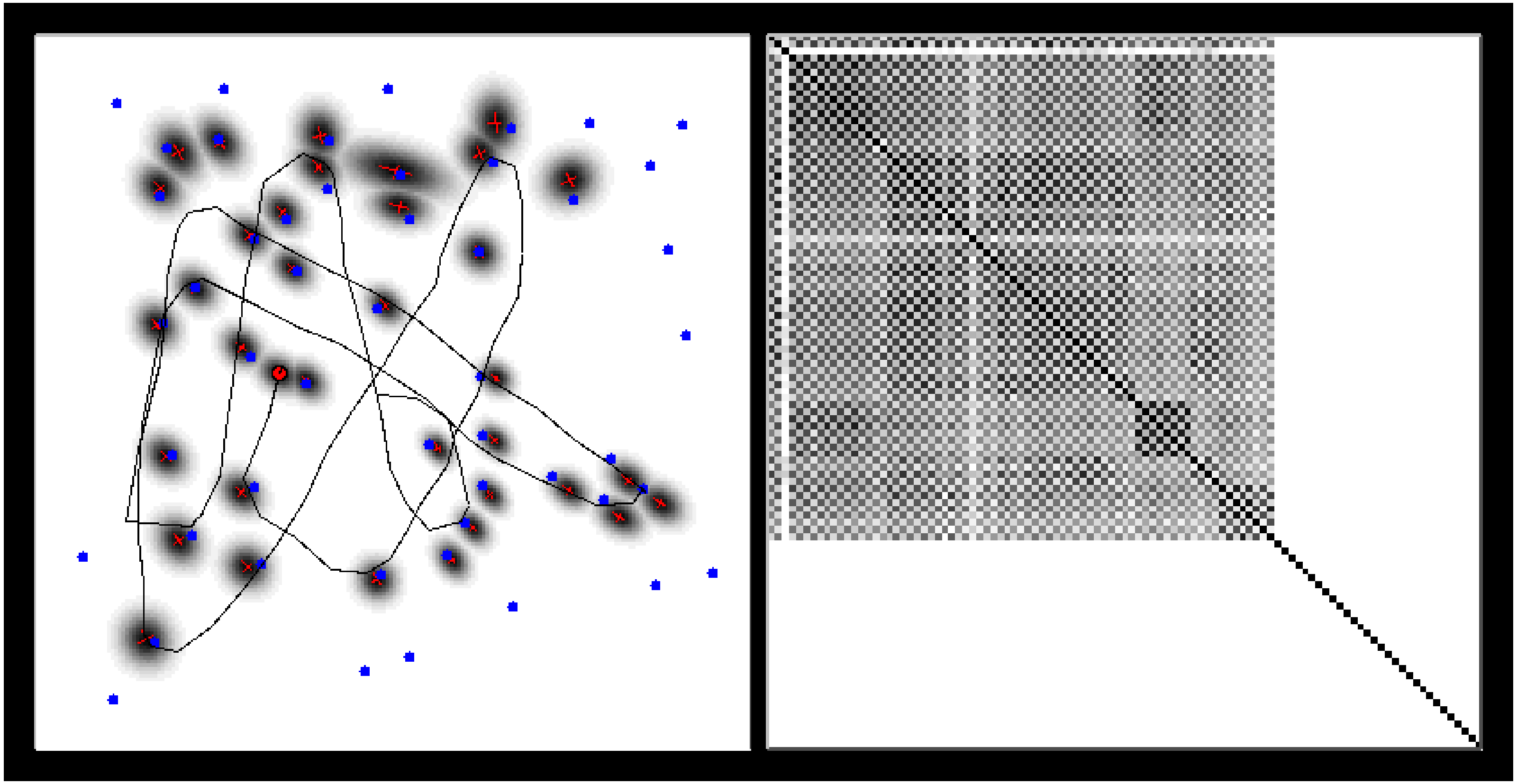
Courtesy: Dissanayake

EKF SLAM



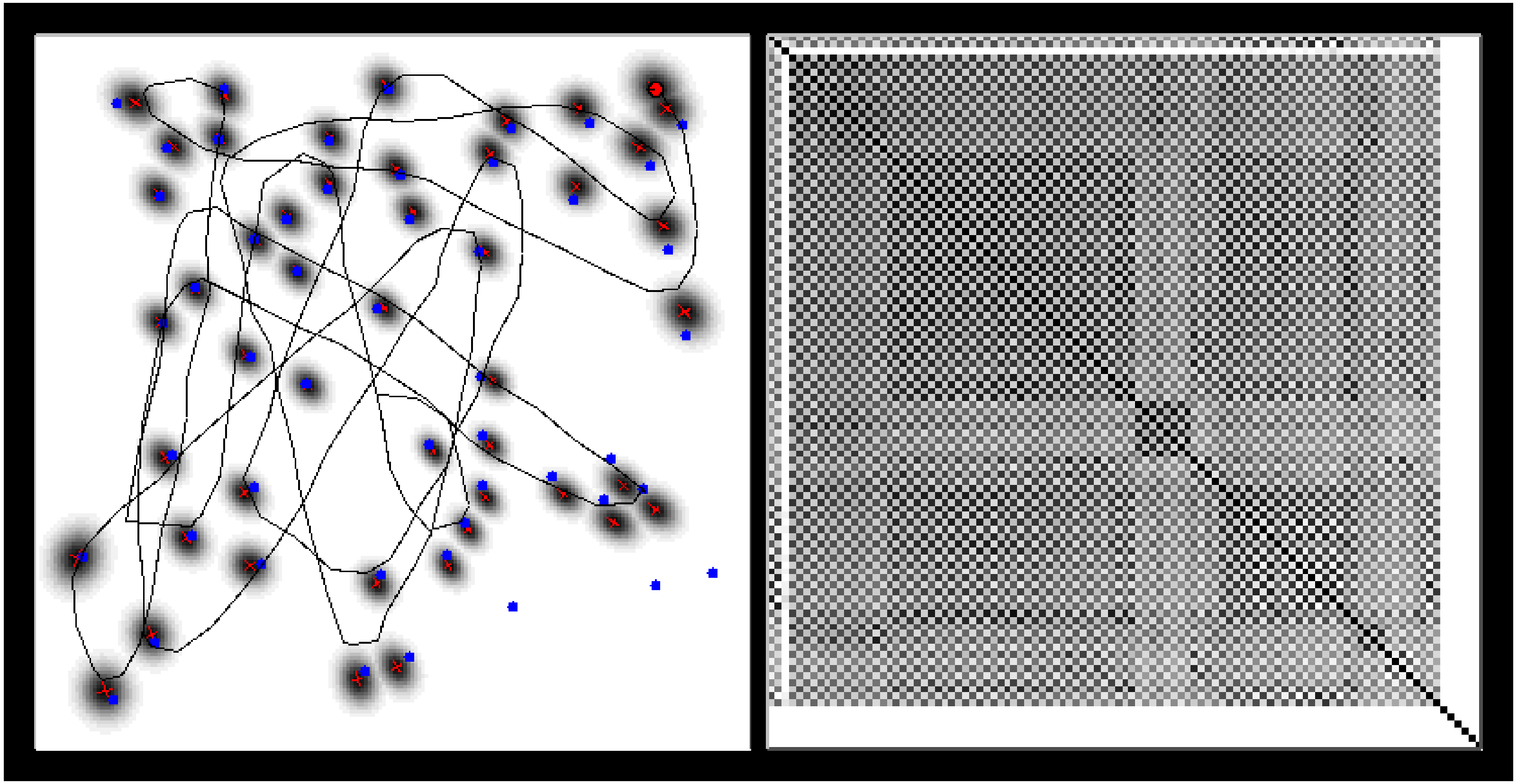
Map Correlation matrix

EKF SLAM



Map Correlation matrix

EKF SLAM



Map Correlation matrix

EKF SLAM: Correlations Matter

- What if we neglected cross-correlations?

$$\Sigma_k = \begin{bmatrix} \Sigma_R & 0 & \cdots & 0 \\ 0 & \Sigma_{M_1} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \Sigma_{M_n} \end{bmatrix}$$

$$\Sigma_{RM_i} = \mathbf{0}_{3\times 2}$$

$$\Sigma_{M_i M_{i+1}} = \mathbf{0}_{2\times 2}$$

EKF SLAM: Correlations Matter

- What if we neglected cross-correlations?

$$\Sigma_k = \begin{bmatrix} \Sigma_R & 0 & \cdots & 0 \\ 0 & \Sigma_{M_1} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \Sigma_{M_n} \end{bmatrix}$$

$$\Sigma_{RM_i} = \mathbf{0}_{3\times 2}$$

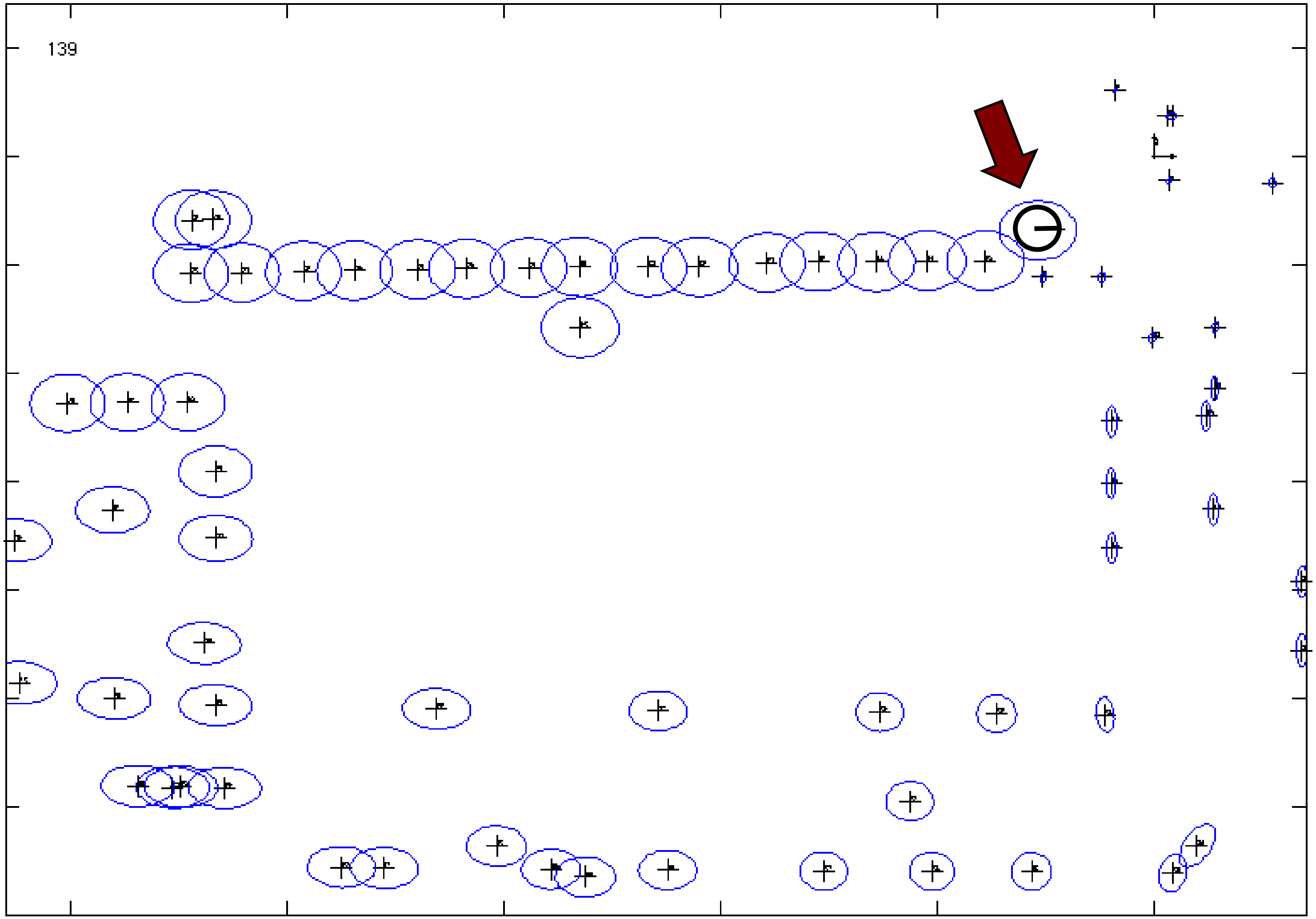
$$\Sigma_{M_i M_{i+1}} = \mathbf{0}_{2\times 2}$$

- Landmark and robot uncertainties would become overly optimistic
- Data association would fail
- As a result, multiple map entries of the same landmark
- Inconsistent map

Loop Closing

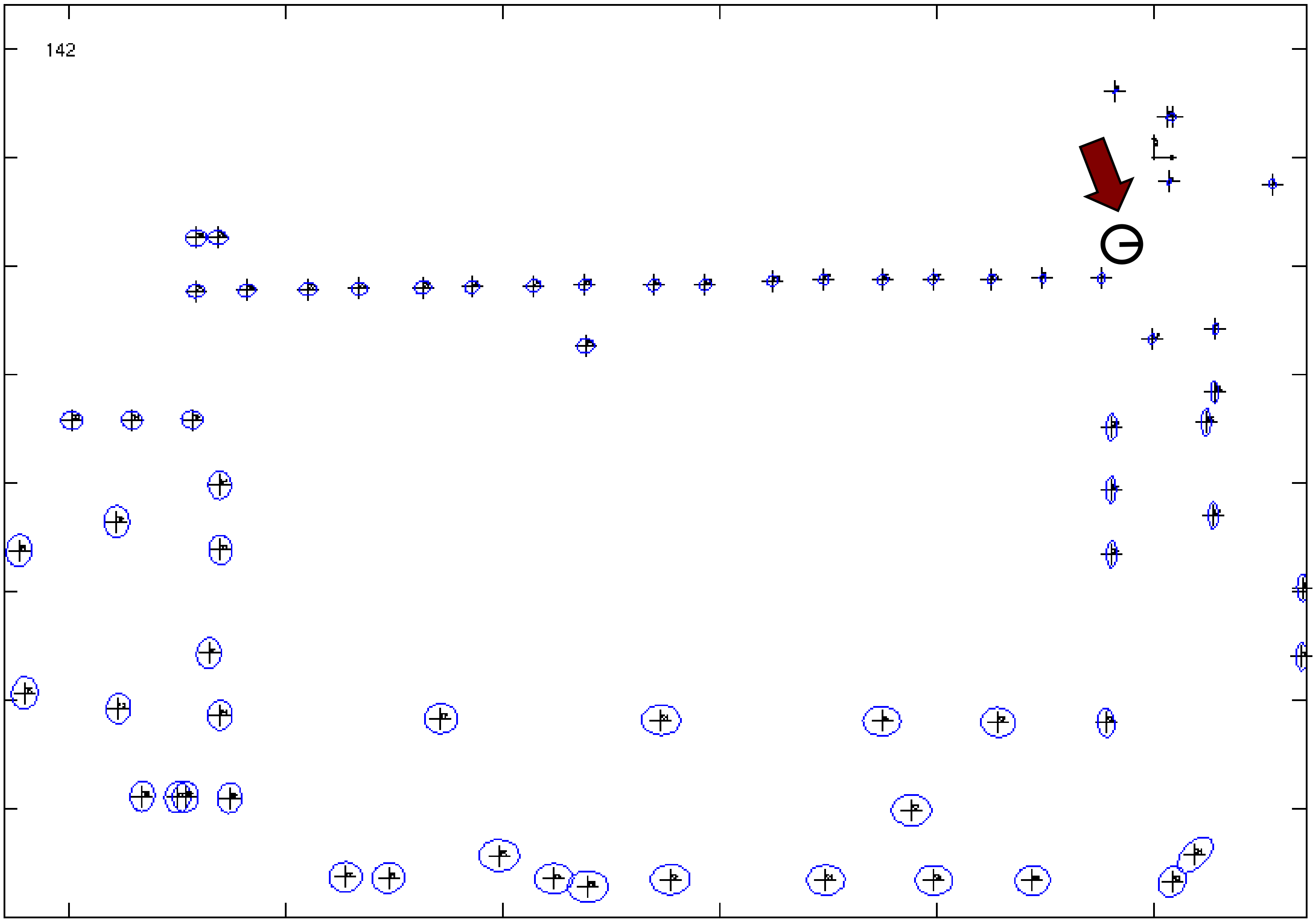
- Recognizing an already mapped area
- Data association under
  - high ambiguity
  - possible environment symmetries
- **Uncertainties collapse after a loop closure** (whether the closure was correct or not)

Before the Loop Closure



Courtesy: K. Arras

After the Loop Closure



Courtesy: K. Arras

Loop Closures in SLAM

- **Loop closing reduces the uncertainty in robot and landmark estimates**
- This can be exploited when exploring an environment
- **However, wrong loop closures lead to filter divergence**

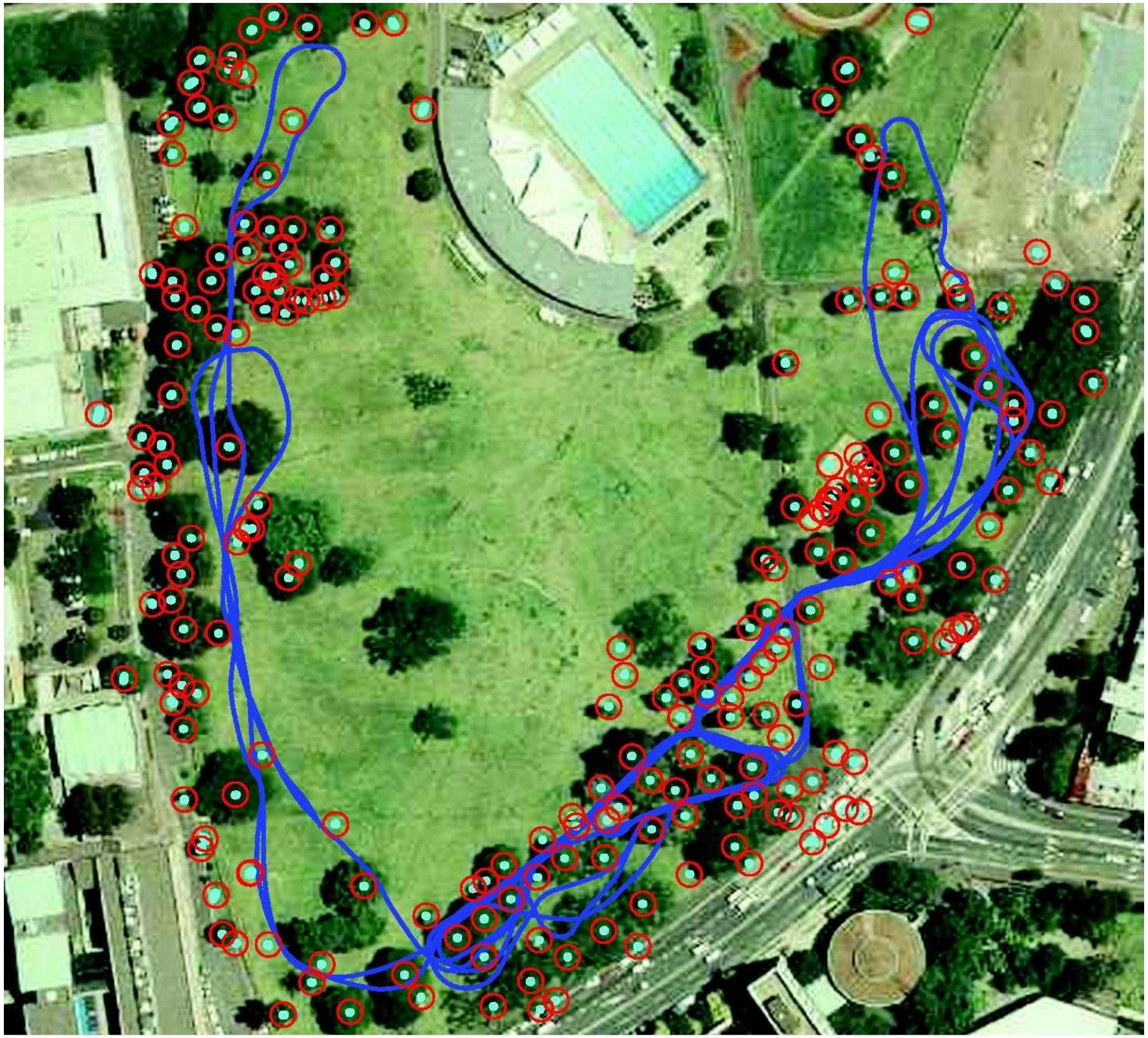
Example: Victoria Park Dataset

Courtesy: E. Nebot

Victoria Park: Data Acquisition

Courtesy: E. Nebot

Victoria Park: Landmarks

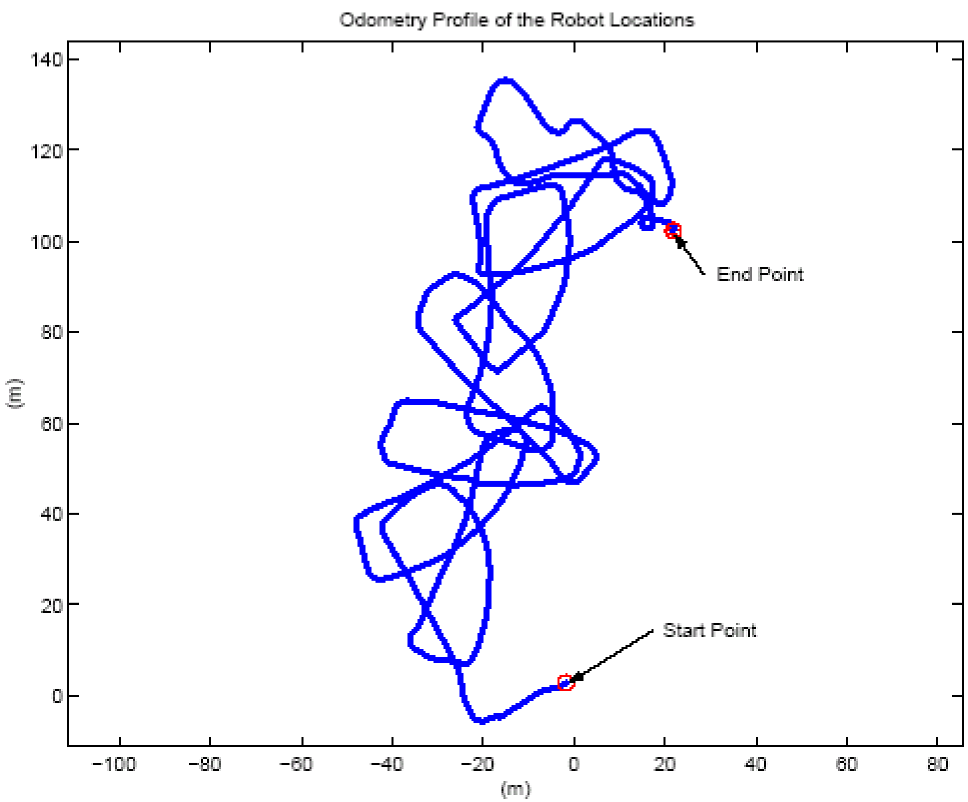


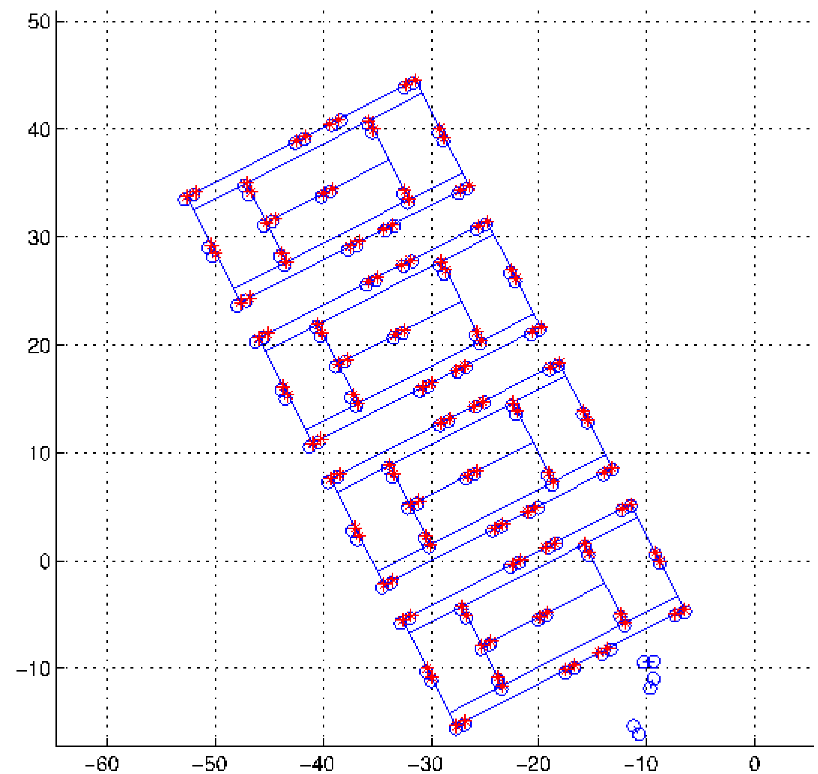
Courtesy: E. Nebot

Example: Tennis Court Dataset

Courtesy: J. Leonard and M. Walter

EKF SLAM on a Tennis Court





odometry

estimated map

Courtesy: J. Leonard and M. Walter

EKF SLAM Complexity

- Cubic complexity depends only on the measurement dimensionality
- Cost per step: dominated by the number of landmarks: $O(n^2)$
- Memory consumption: $O(n^2)$
- The EKF becomes computationally intractable for large maps!

Summary: EKF SLAM

- The first SLAM solution
- Convergence proof for the linear Gaussian case
- Can diverge if non-linearities are large (and the reality is non-linear...)
- Can deal only with a single mode
- Successful in medium-scale scenes
- Approximations exists to reduce the computational complexity
- Data association has to be solved

Particle Filter

- Non-parametric recursive Bayes filter
- Posterior is represented by a set of weighted samples
- Can model arbitrary distributions
- Works well in low-dimensional spaces
- Three steps
  - Sampling from proposal
  - Importance weighting
  - Resampling

Particle Representation

- A set of weighted samples

$$\mathcal{X} = \left\{ \left\langle x^{[i]}, w^{[i]} \right\rangle \right\}_{i=1,\ldots,N}$$

- Each sample is a hypothesis about the state
- For feature-based SLAM:

$$x = (\underbrace{x_{1:t}}_{\textbf{poses}}, \underbrace{m_{1,x}, m_{1,y}, \ldots, m_{M,x}, m_{M,y}}_{\textbf{landmarks}})^T$$

Dimensionality Problem

- Particle filters are effective in low-dimensional spaces
- The likely regions of the state space need to be covered with samples

$$x = (x_{1:t}, m_{1,x}, m_{1,y}, \ldots, m_{M,x}, m_{M,y})^T$$

high-dimensional!

Can We Exploit Dependencies Between the Different Dimensions of the State Space?

$$x_{1:t}, m_1, \ldots, m_M$$

If We Know the Poses of the Robot, Mapping is Easy!

$$\underline{x_{1:t}}, \underline{m_1, \ \ldots, m_M}$$

Key Idea

$$\underline{x_{1:t}}, \underline{m_1, \ \ldots, m_M}$$

- If we use the particle set only to model the robot's path, each sample is a path hypothesis
- For each sample, we can compute an individual map of landmarks

Acknowledgment

- These slides have been created by Wolfram Burgard, Dieter Fox, Cyrill Stachniss and Maren Bennewitz