Introduction to Mobile Robotics

SLAM: Simultaneous Localization and Mapping

Wolfram Burgard, Michael Krawez

UTN

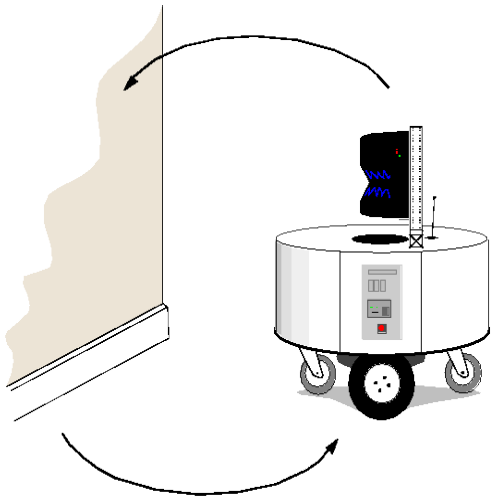
What is SLAM?

- 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

- **Localization:** inferring location given a map
- **Mapping:** inferring a map given locations
- **SLAM:** learning a map and localizing the robot simultaneously

The SLAM Problem

- SLAM has long been regarded as a **chicken-or-egg** problem:
  → a map is needed for localization and
  → a pose estimate is needed for mapping



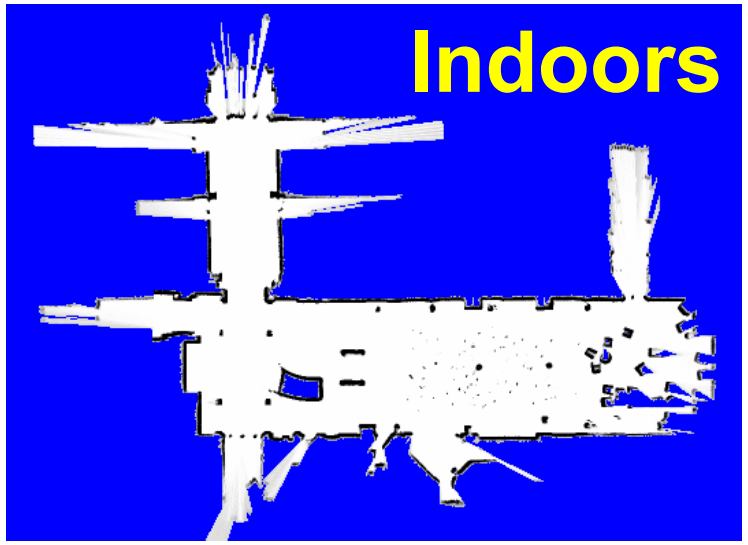
SLAM Applications

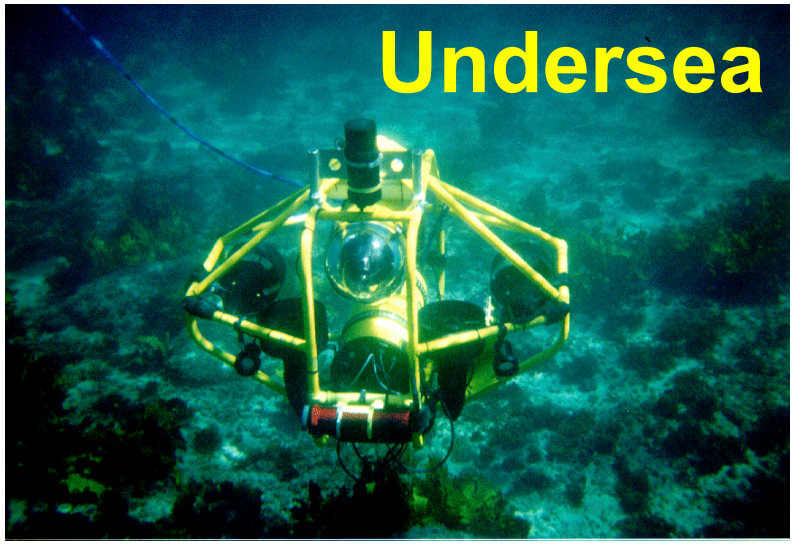
- SLAM is central to a range of indoor, outdoor, in-air and underwater applications for both manned and autonomous vehicles.

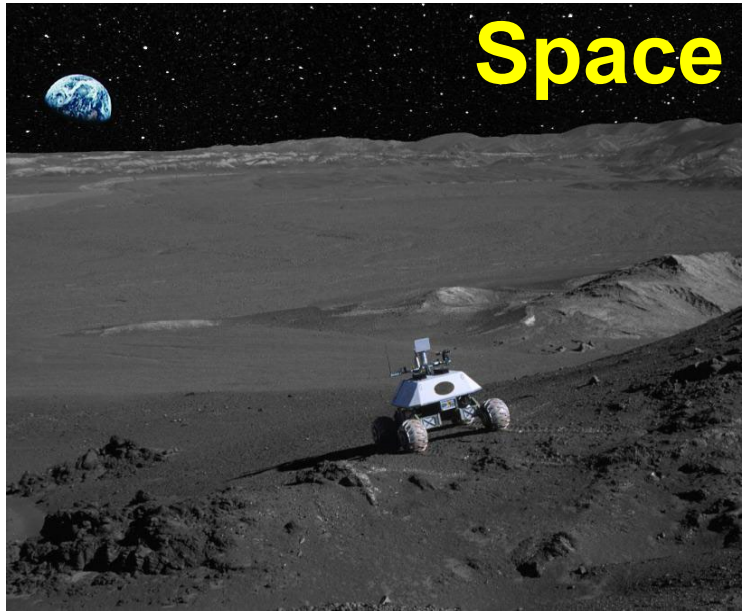
Examples:

- At home: vacuum cleaning, lawn mowing
- Air: surveillance with unmanned air vehicles
- Underwater: reef monitoring
- Underground: exploration of mines
- Space: terrain mapping for localization
- Cars: staying on the road

SLAM Applications

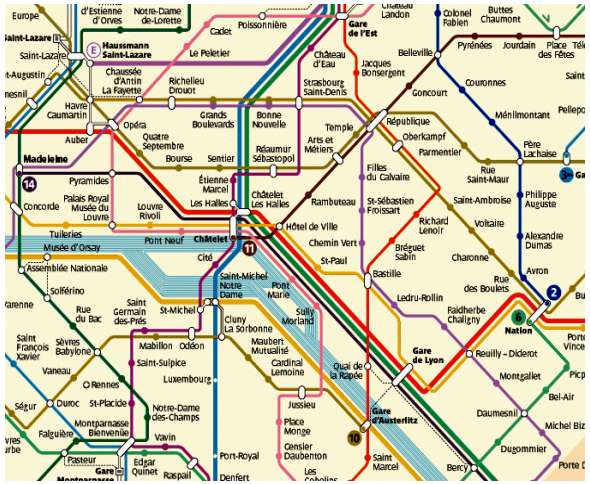


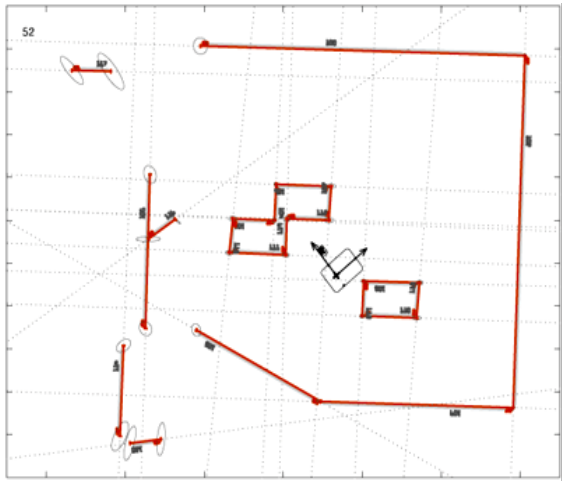




Map Representations

Examples: Subway map, city map, landmark-based map

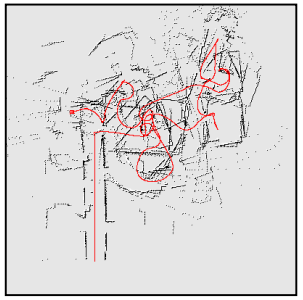


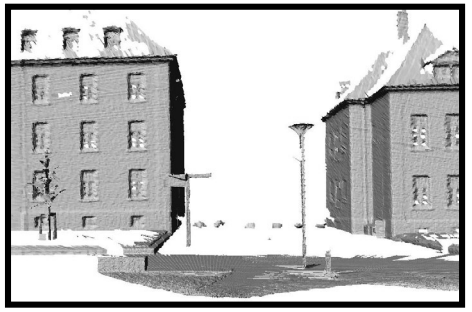


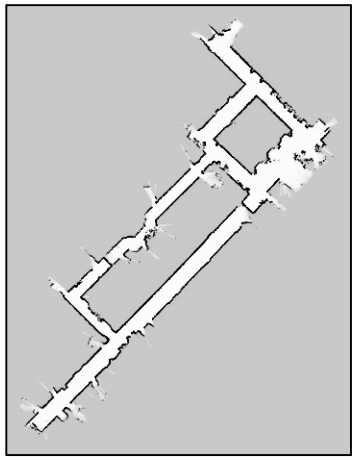
Here: Maps are **topological** and/or **metric models** of the environment

Map Representations in Robotics

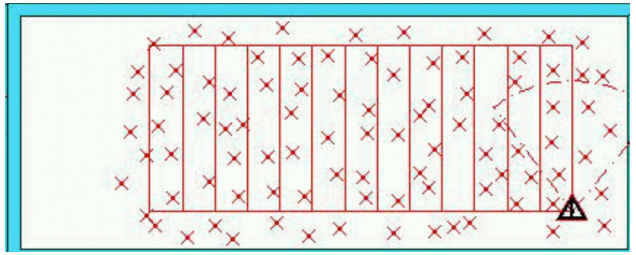
- Grid maps or scans, 2d, 3d

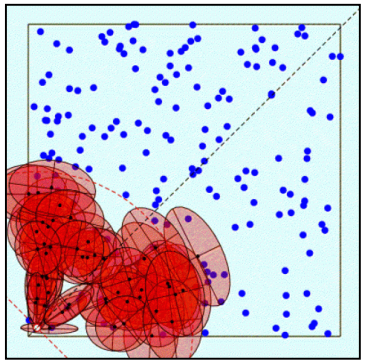






- Landmark-based





The SLAM Problem

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

Different Variants of the SLAM Problem

- Depending on the representation
  - Feature-based (landmarks or features)
  - Dense (2d/3d grids, signed distance functions, …)
- Depending on the sensor
  - Vision
  - Proximity sensors
- Depending on the sensor information
  - Range only
  - bearing only
  - Range and bearing
- Depending on the algorithm
  - Filtering-based
  - Optimization-based

In this Course

Filtering-based approaches to SLAM for

- Extended Kalman Filter (feature-based)
- Rao-Blackwellized Particle Filters (feature-based and dense)

Feature-Based SLAM

Given:

- The robot's controls

$$U_{1:k} = \{u_1, u_2, \ldots, u_k\}$$

- Relative observations

$$Z_{1:k} = \{z_1, z_2, \ldots, z_k\}$$

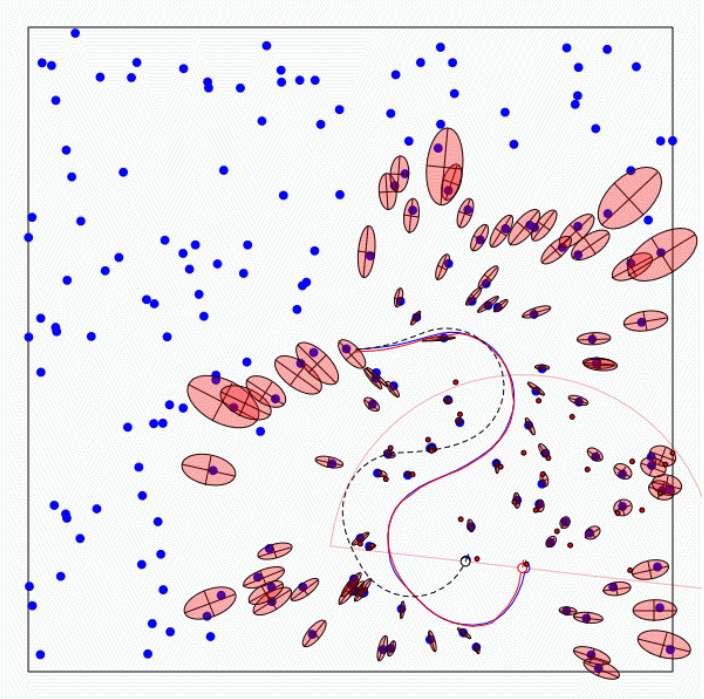
Wanted:

- Map of features

$$m = \{m_1, m_2, \ldots, m_n\}$$

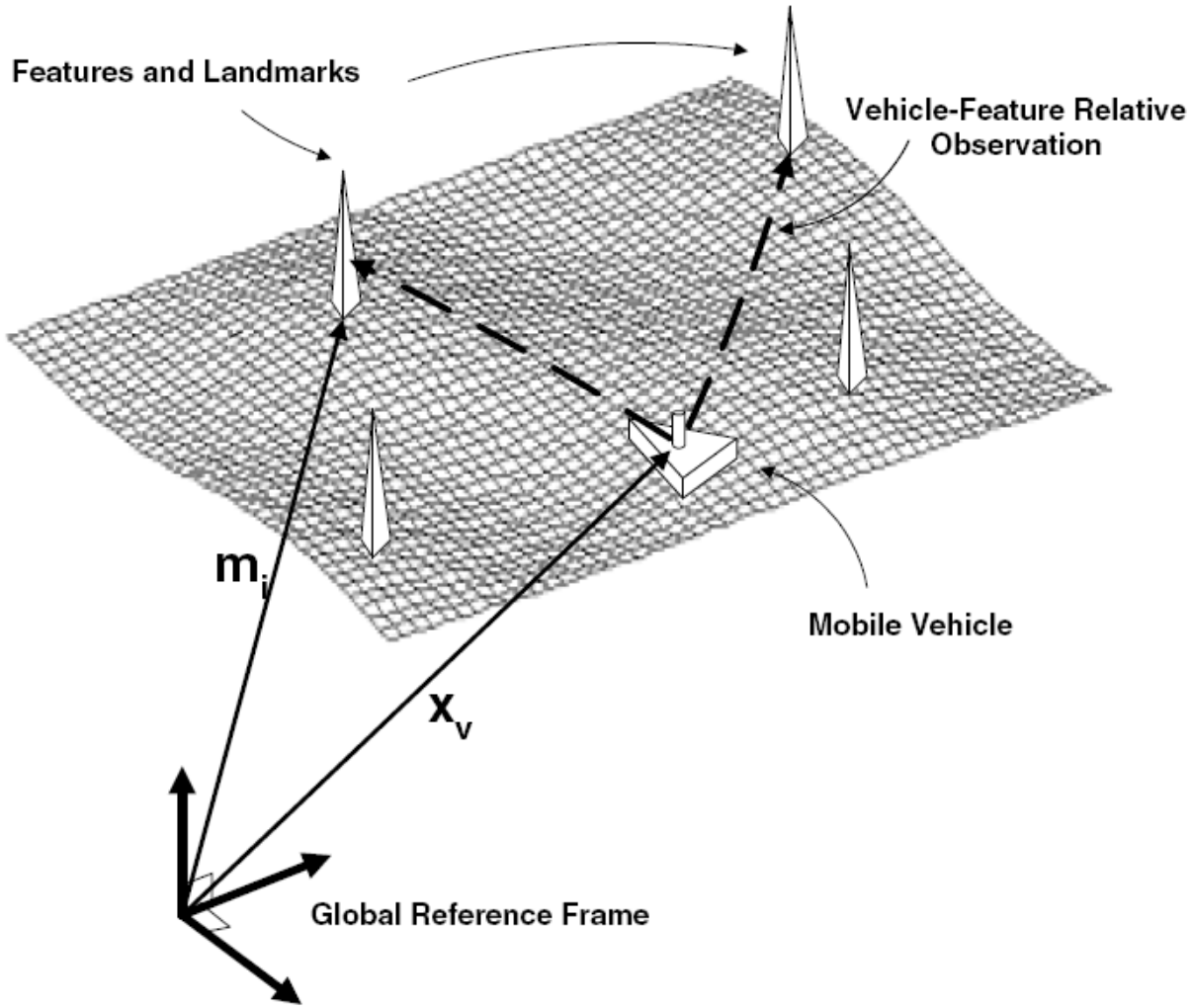
- Path of the robot

$$X_{1:k} = \{x_1, x_2, \ldots, x_k\}$$



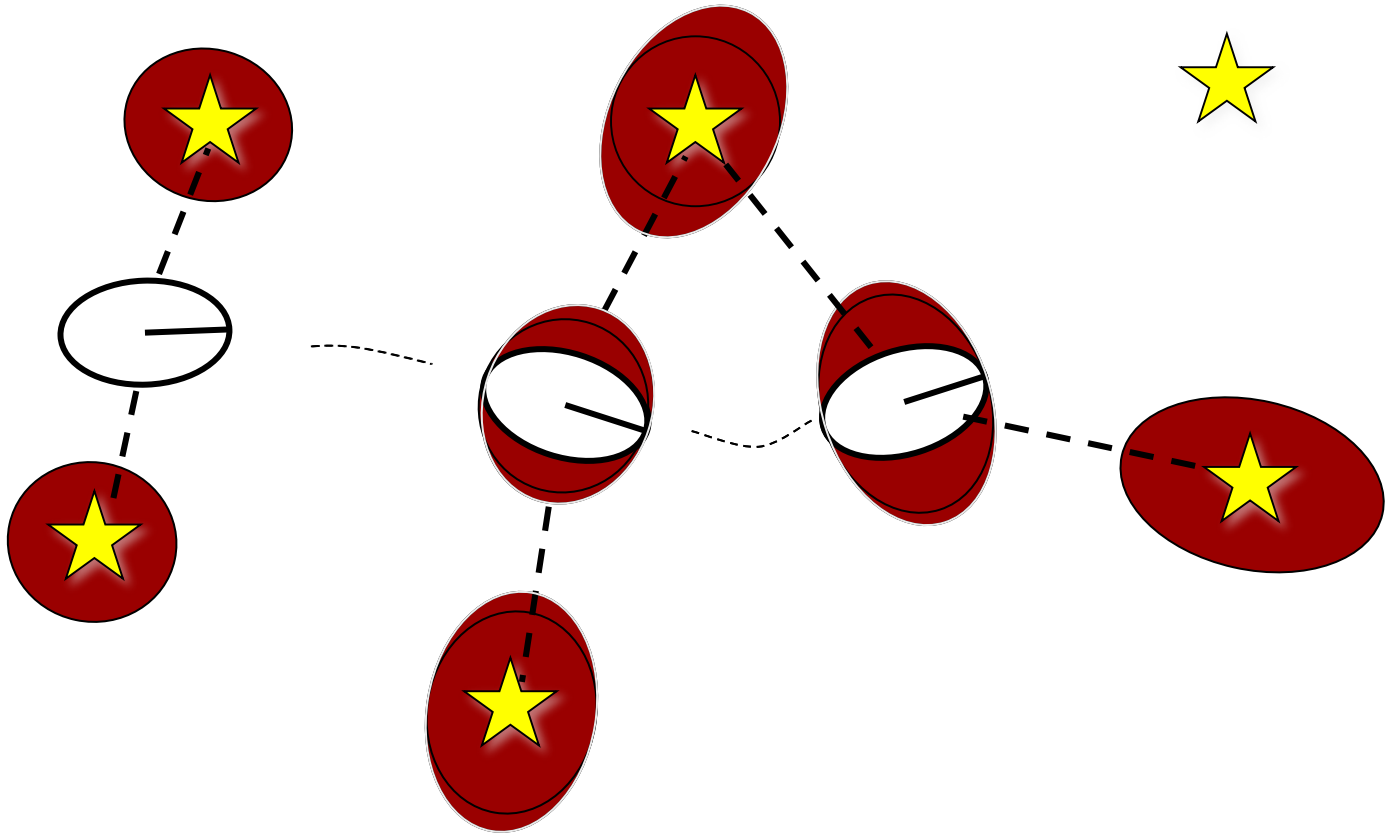
Feature-Based SLAM

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



Why this SLAM Problem is Hard

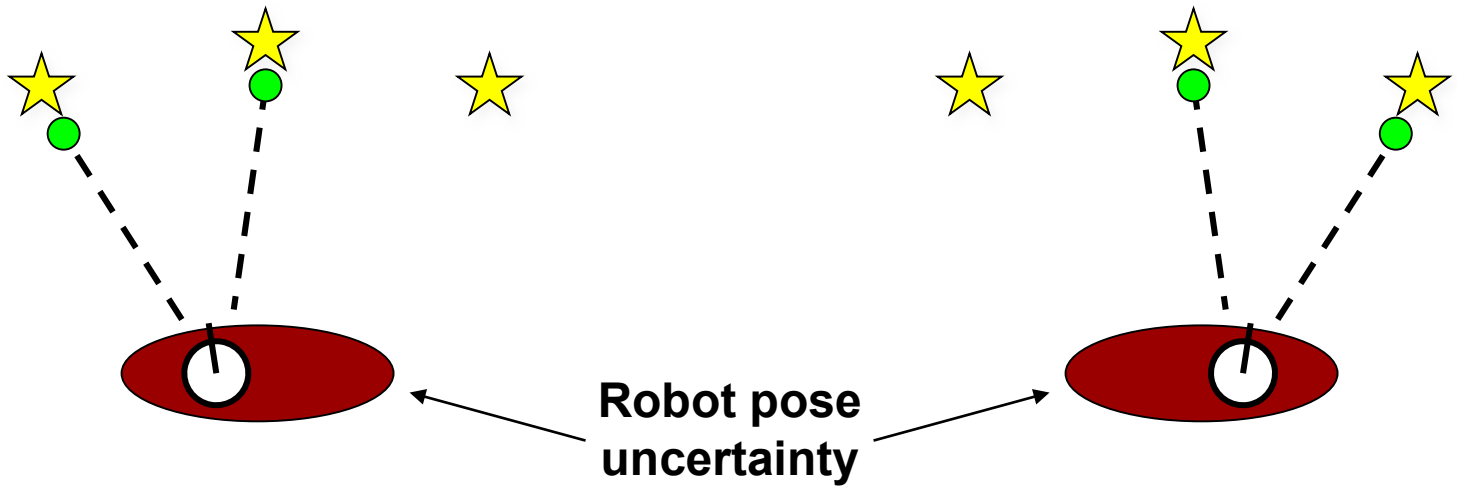
1. Robot path and map are both **unknown**



2. Errors in map and pose estimates correlated

Why this SLAM Problem is Hard

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



SLAM: Simultaneous Localization And Mapping

- Full SLAM:

$$p(x_{0:t}, m \mid z_{1:t}, u_{1:t})$$

Estimates entire path and map!

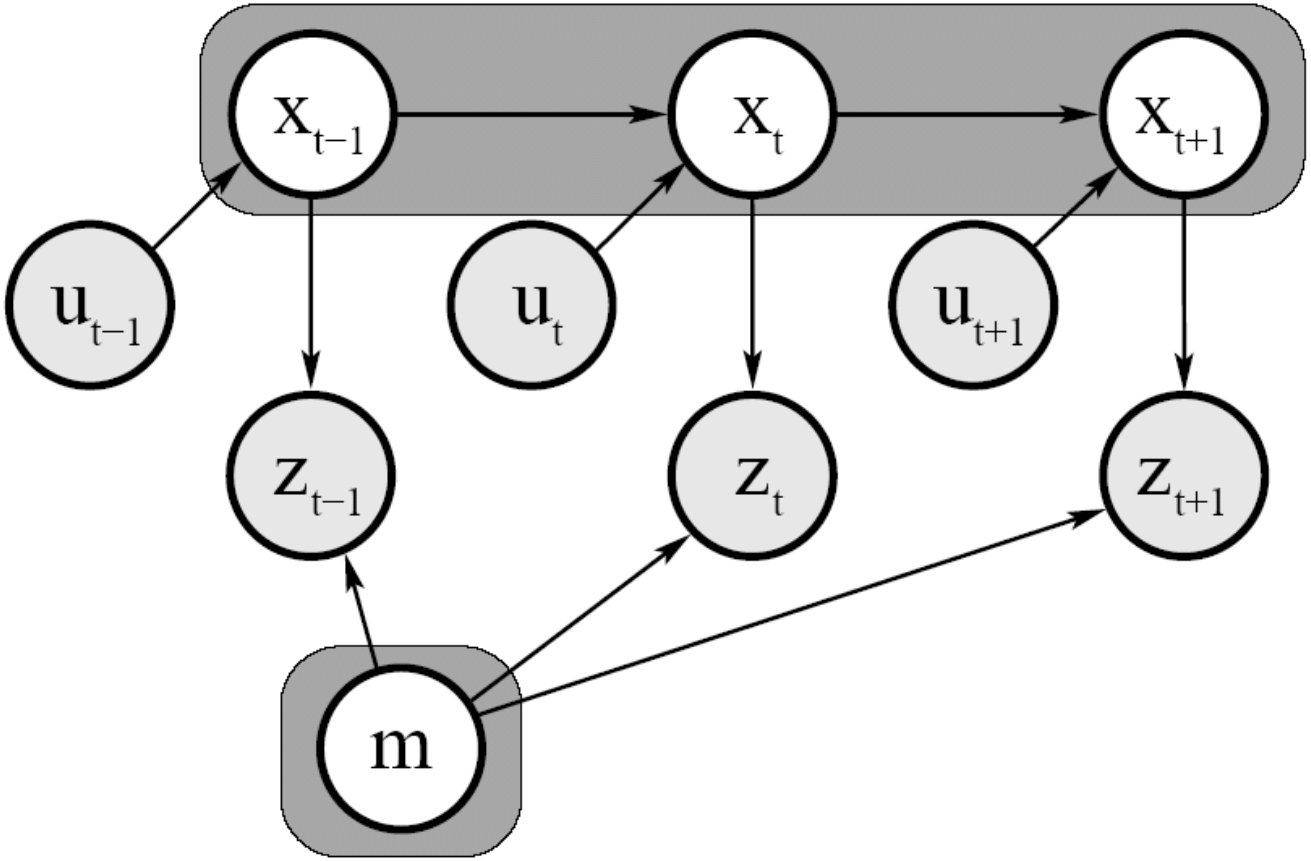
- Online SLAM:

$$p(x_t, m \mid z_{1:t}, u_{1:t}) = \int\int \ldots \int p(x_{1:t}, m \mid z_{1:t}, u_{1:t}) dx_1 dx_2 \ldots dx_{t-1}$$

Estimates most recent pose and map!

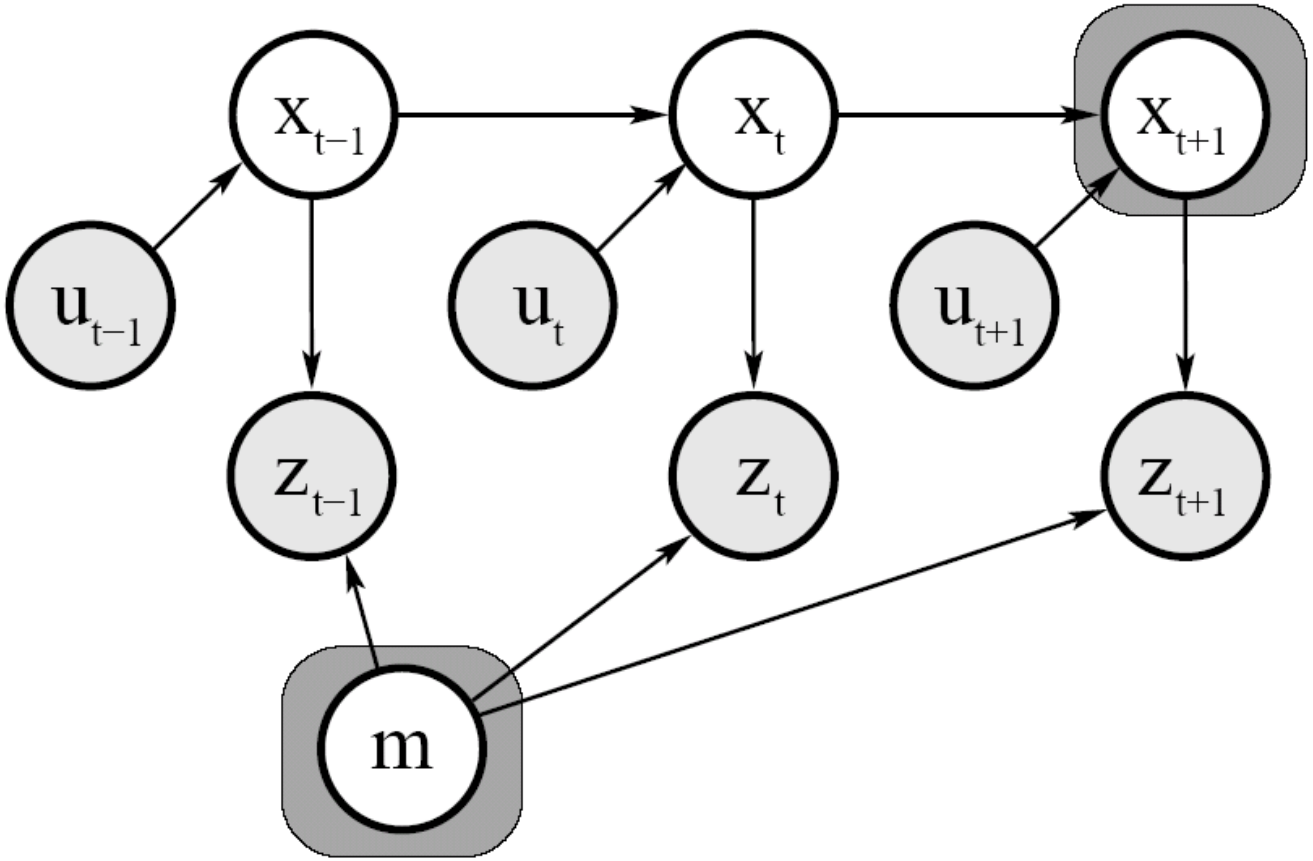
- Integrations (marginalization) typically done recursively, one at a time

Graphical Model of Full SLAM



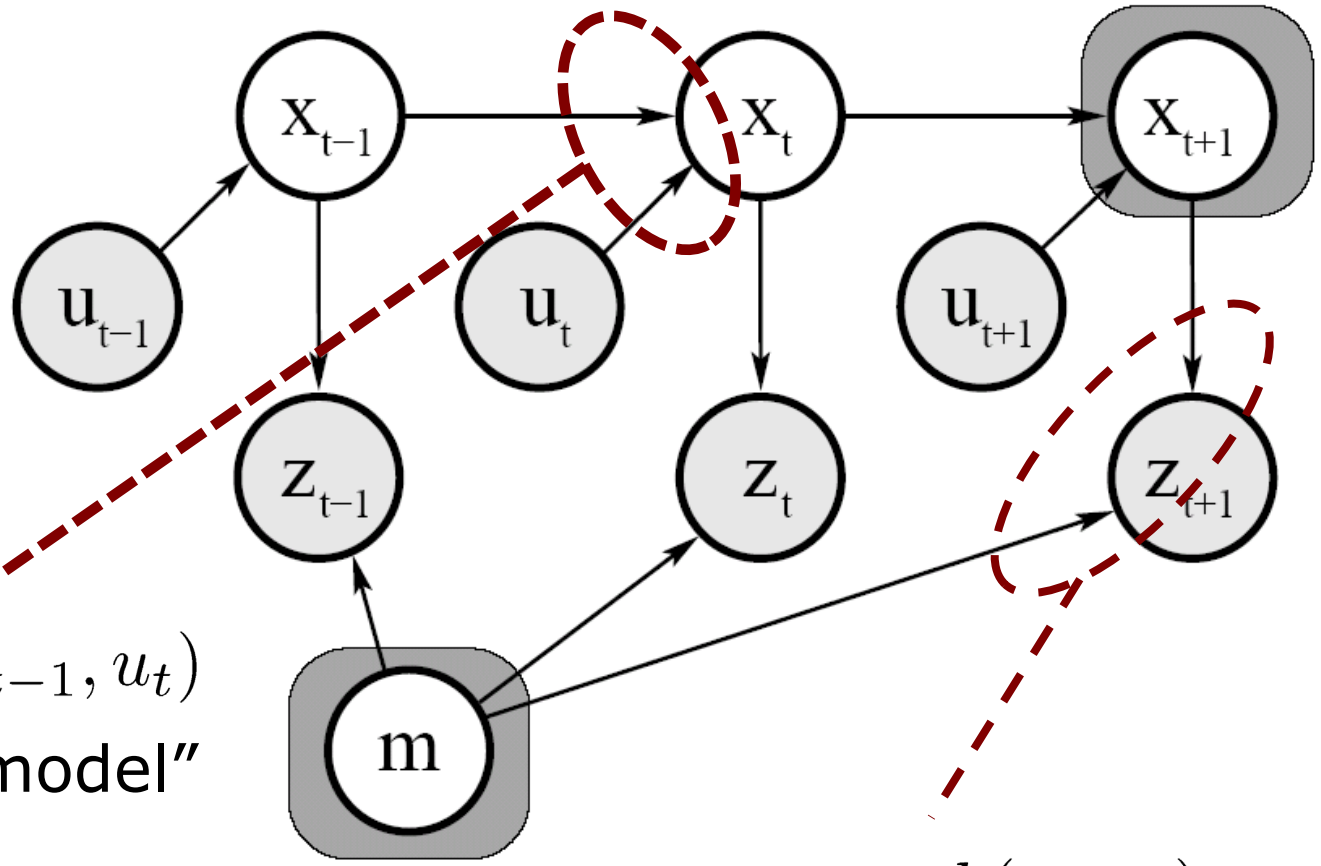
$$p(x_{1:t+1}, m \mid z_{1:t+1}, u_{1:t+1})$$

Graphical Model of Online SLAM



$$p(x_{t+1}, m \mid z_{1:t+1}, u_{1:t+1}) = \int\int \ldots \int p(x_{1:t+1}, m \mid z_{1:t+1}, u_{1:t+1}) dx_1 dx_2 \ldots dx_t$$

Motion and Observation Model



$x_t = f(x_{t-1}, u_t)$

"Motion model"

$z_t = h(x_t, m)$

"Observation model"

Remember the KF Algorithm

1. Algorithm **Kalman_filter**($\mu_{t-1}, \Sigma_{t-1}, u_t, z_t$):
2. Prediction:
3. $\bar{\mu}_t = A_t \mu_{t-1} + B_t u_t$
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. $\mu_t = \bar{\mu}_t + K_t (z_t - C_t \bar{\mu}_t)$
8. $\Sigma_t = (I - K_t C_t) \bar{\Sigma}_t$
9. Return μ_t, Σ_t

EKF SLAM: State representation

- **Localization**

  3x1 pose vector
  3x3 cov. matrix

$$\mathbf{x}_k = \begin{bmatrix} x_k \\ y_k \\ \theta_k \end{bmatrix} \quad \Sigma_k = \begin{bmatrix} \sigma_x^2 & \sigma_{xy}^2 & \sigma_{x\theta}^2 \\ \sigma_{yx}^2 & \sigma_y^2 & \sigma_{y\theta}^2 \\ \sigma_{\theta x}^2 & \sigma_{\theta y}^2 & \sigma_\theta^2 \end{bmatrix}$$

- **SLAM**

  Landmarks **simply extend** the state.
  Growing state vector and covariance matrix!

$$\mathbf{x}_k = \begin{bmatrix} \mathbf{x}_R \\ \mathbf{m}_1 \\ \mathbf{m}_2 \\ \vdots \\ \mathbf{m}_n \end{bmatrix} \quad \Sigma_k = \begin{bmatrix} \Sigma_R & \Sigma_{RM_1} & \Sigma_{RM_2} & \cdots & \Sigma_{RM_n} \\ \Sigma_{M_1R} & \Sigma_{M_1} & \Sigma_{M_1M_2} & \cdots & \Sigma_{M_1M_n} \\ \Sigma_{M_2R} & \Sigma_{M_2M_1} & \Sigma_{M_2} & \cdots & \Sigma_{M_2M_n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \Sigma_{M_nR} & \Sigma_{M_nM_1} & \Sigma_{M_nM_2} & \cdots & \Sigma_{M_n} \end{bmatrix}$$

EKF SLAM: State representation

- Map with *n* landmarks: (3+2*n*)-dimensional Gaussian

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

- Can handle a large number of dimensions

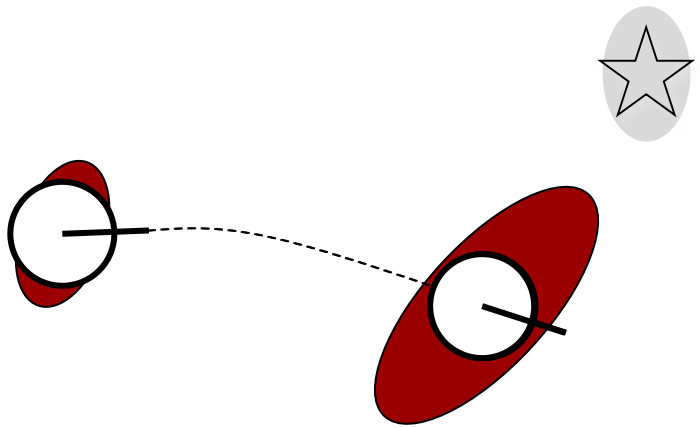
EKF SLAM: Filter Cycle

1. State prediction (odometry)
2. Measurement prediction
3. Measurement
4. Data association
5. Update
6. Integration of new landmarks

EKF SLAM: Filter Cycle

1. State prediction (odometry)
2. Measurement prediction
3. Measurement
4. Data association
5. Update
6. Integration of new landmarks

EKF SLAM: State Prediction



Odometry:

$$\hat{\mathbf{x}}_R = f(\mathbf{x}_R, \mathbf{u})$$

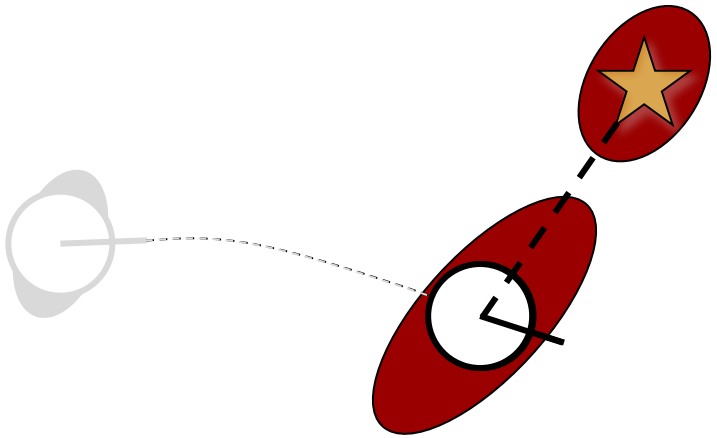
$$\hat{\Sigma}_R = F_x \Sigma_R F_x^T + F_u U F_u^T$$

Robot-landmark cross-covariance prediction:

$$\hat{\Sigma}_{RM_i} = F_x \Sigma_{RM_i}$$

$$\underbrace{\begin{bmatrix} \mathbf{x}_R \\ \mathbf{m}_1 \\ \vdots \\ \mathbf{m}_n \end{bmatrix}}_{\mu} \quad \underbrace{\begin{bmatrix} \Sigma_R & \Sigma_{RM_1} & \cdots & \Sigma_{RM_n} \\ \Sigma_{M_1 R} & \Sigma_{M_1} & \cdots & \Sigma_{M_1 M_n} \\ \vdots & \vdots & \ddots & \vdots \\ \Sigma_{M_n R} & \Sigma_{M_n M_1} & \cdots & \Sigma_{M_n} \end{bmatrix}}_{\Sigma}$$

EKF SLAM: Measurement Prediction

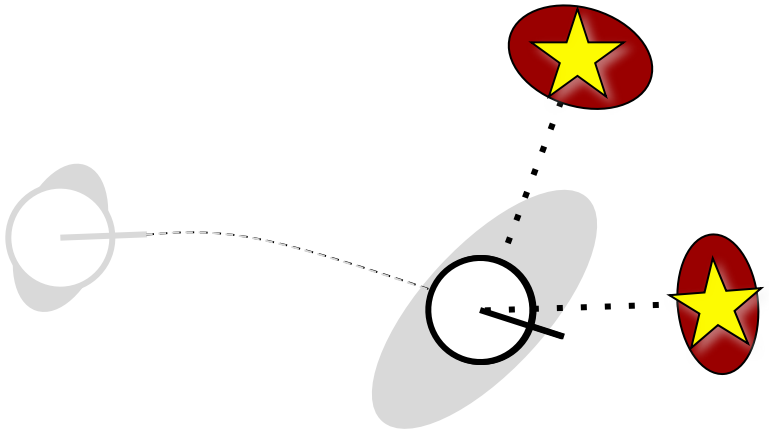


Global-to-local
frame transform h

$$\hat{\mathbf{z}}_k = h(\hat{\mathbf{x}}_k)$$

$$\underbrace{\begin{bmatrix} \mathbf{x}_R \\ \mathbf{m}_1 \\ \vdots \\ \mathbf{m}_n \end{bmatrix}}_{\mu} \quad \underbrace{\begin{bmatrix} \Sigma_R & \Sigma_{RM_1} & \cdots & \Sigma_{RM_n} \\ \Sigma_{M_1R} & \Sigma_{M_1} & \cdots & \Sigma_{M_1M_n} \\ \vdots & \vdots & \ddots & \vdots \\ \Sigma_{M_nR} & \Sigma_{M_nM_1} & \cdots & \Sigma_{M_n} \end{bmatrix}}_{\Sigma}$$

EKF SLAM: Obtained Measurement



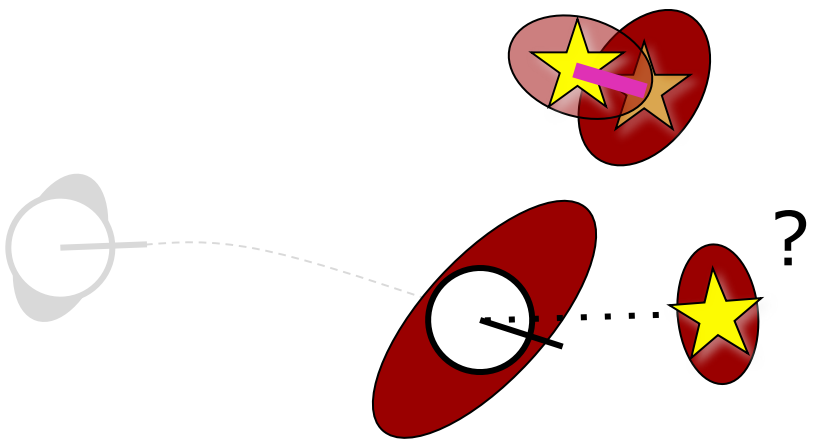
(x,y)-point landmarks

$$\mathbf{z}_k = \begin{bmatrix} x_1 \\ y_1 \\ x_2 \\ y_2 \end{bmatrix} = \begin{bmatrix} \mathbf{z}_1 \\ \mathbf{z}_2 \end{bmatrix}$$

$$R_k = \begin{bmatrix} R_1 & 0 \\ 0 & R_2 \end{bmatrix}$$

$$\underbrace{\begin{bmatrix} \mathbf{x}_R \\ \mathbf{m}_1 \\ \vdots \\ \mathbf{m}_n \end{bmatrix}}_{\mu} \quad \underbrace{\begin{bmatrix} \Sigma_R & \Sigma_{RM_1} & \cdots & \Sigma_{RM_n} \\ \Sigma_{M_1R} & \Sigma_{M_1} & \cdots & \Sigma_{M_1M_n} \\ \vdots & \vdots & \ddots & \vdots \\ \Sigma_{M_nR} & \Sigma_{M_nM_1} & \cdots & \Sigma_{M_n} \end{bmatrix}}_{\Sigma}$$

EKF SLAM: Data Association

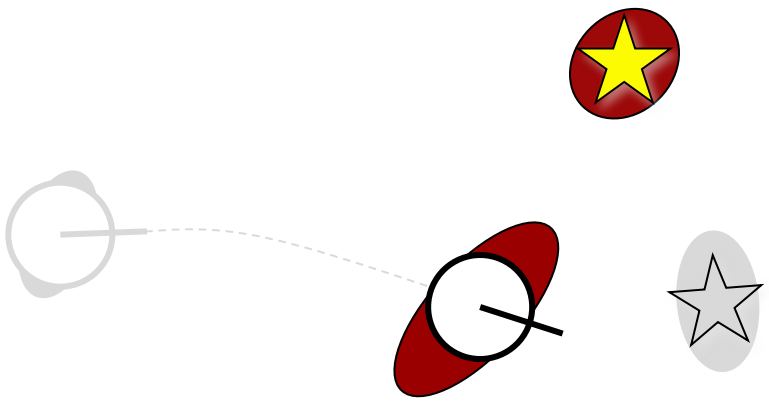


Associates predicted measurements $\hat{\mathbf{z}}_k^i$ with observation $\mathbf{z}_k^j$

$$\begin{aligned} \nu_k^{ij} &= \mathbf{z}_k^j - \hat{\mathbf{z}}_k^i \\ S_k^{ij} &= R_k^j + H^i \hat{\Sigma}_k H^{i\,T} \end{aligned}$$

$$\underbrace{\begin{bmatrix} \mathbf{x}_R \\ \mathbf{m}_1 \\ \vdots \\ \mathbf{m}_n \end{bmatrix}}_{\mu} \quad \underbrace{\begin{bmatrix} \Sigma_R & \Sigma_{RM_1} & \cdots & \Sigma_{RM_n} \\ \Sigma_{M_1R} & \Sigma_{M_1} & \cdots & \Sigma_{M_1M_n} \\ \vdots & \vdots & \ddots & \vdots \\ \Sigma_{M_nR} & \Sigma_{M_nM_1} & \cdots & \Sigma_{M_n} \end{bmatrix}}_{\Sigma}$$

EKF SLAM: Update Step



The usual Kalman filter expressions

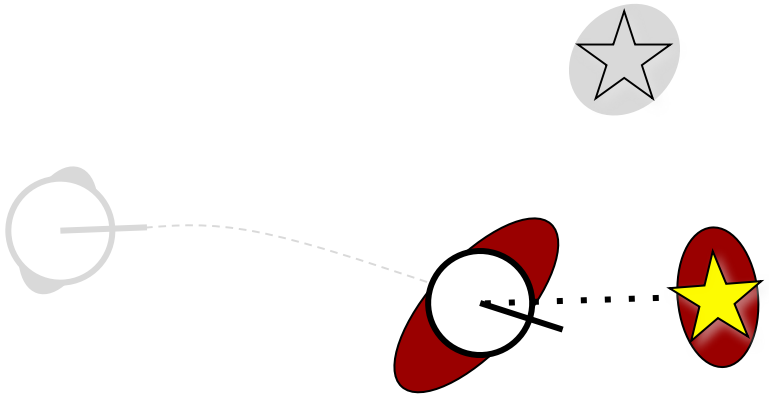
$$K_k = \hat{\Sigma}_k H^T S_k^{-1}$$

$$\mathbf{x}_k = \hat{\mathbf{x}}_k + K_k \nu_k$$

$$C_k = (I - K_k H)\hat{\Sigma}_k$$

$$\underbrace{\begin{bmatrix} \mathbf{x}_R \\ \mathbf{m}_1 \\ \vdots \\ \mathbf{m}_n \end{bmatrix}}_{\mu} \quad \underbrace{\begin{bmatrix} \Sigma_R & \Sigma_{RM_1} & \cdots & \Sigma_{RM_n} \\ \Sigma_{M_1R} & \Sigma_{M_1} & \cdots & \Sigma_{M_1M_n} \\ \vdots & \vdots & \ddots & \vdots \\ \Sigma_{M_nR} & \Sigma_{M_nM_1} & \cdots & \Sigma_{M_n} \end{bmatrix}}_{\Sigma}$$

EKF SLAM: New Landmarks



State augmented by

$$\mathbf{m}_{n+1} = g(\mathbf{x}_R, \mathbf{z}_j)$$

$$\Sigma_{M_{n+1}} = G_R \Sigma_R G_R^T + G_z R_j G_z^T$$

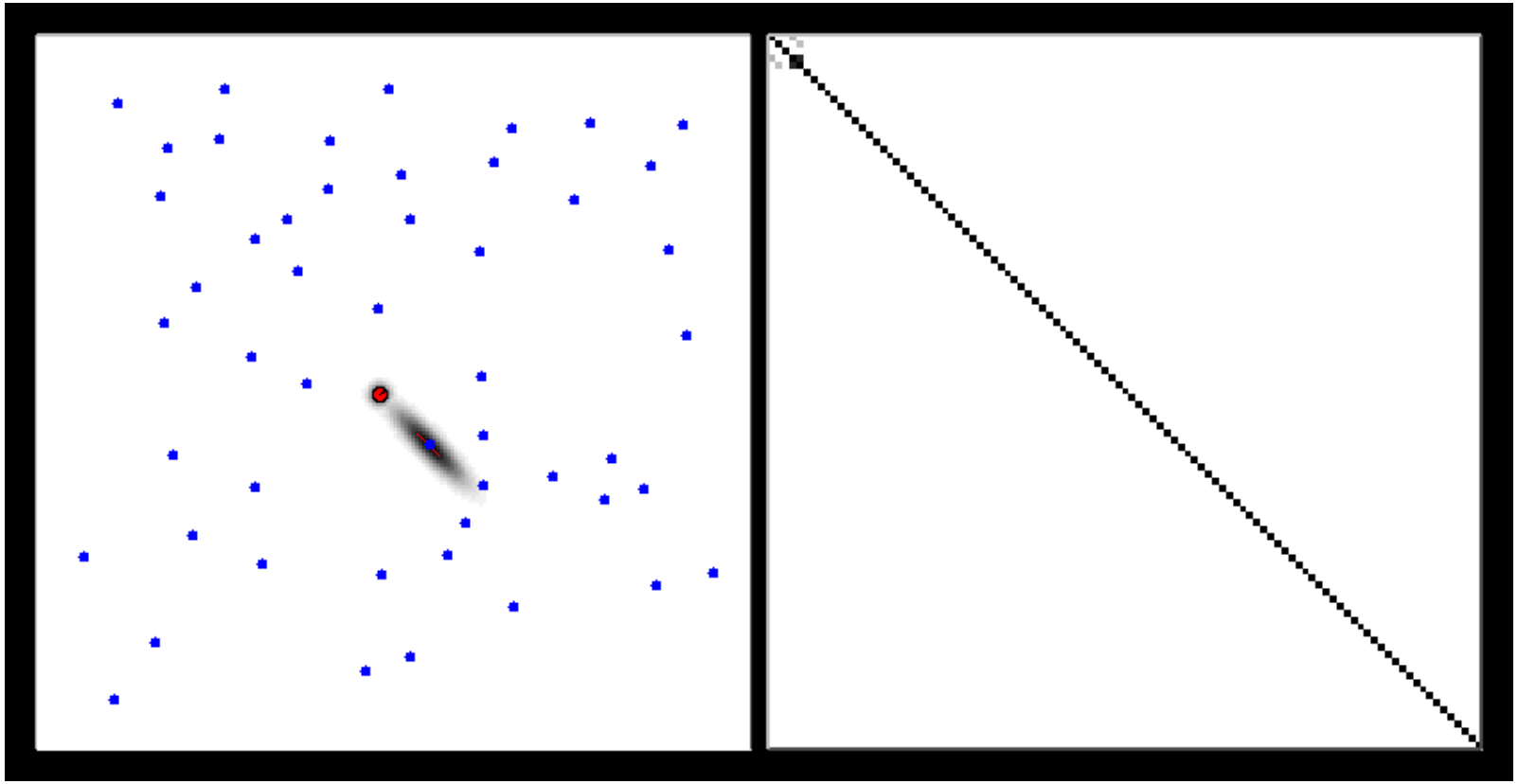
Cross-covariances:

$$\Sigma_{M_{n+1} M_i} = G_R \Sigma_{R M_i}$$

$$\Sigma_{M_{n+1} R} = G_R \Sigma_R$$

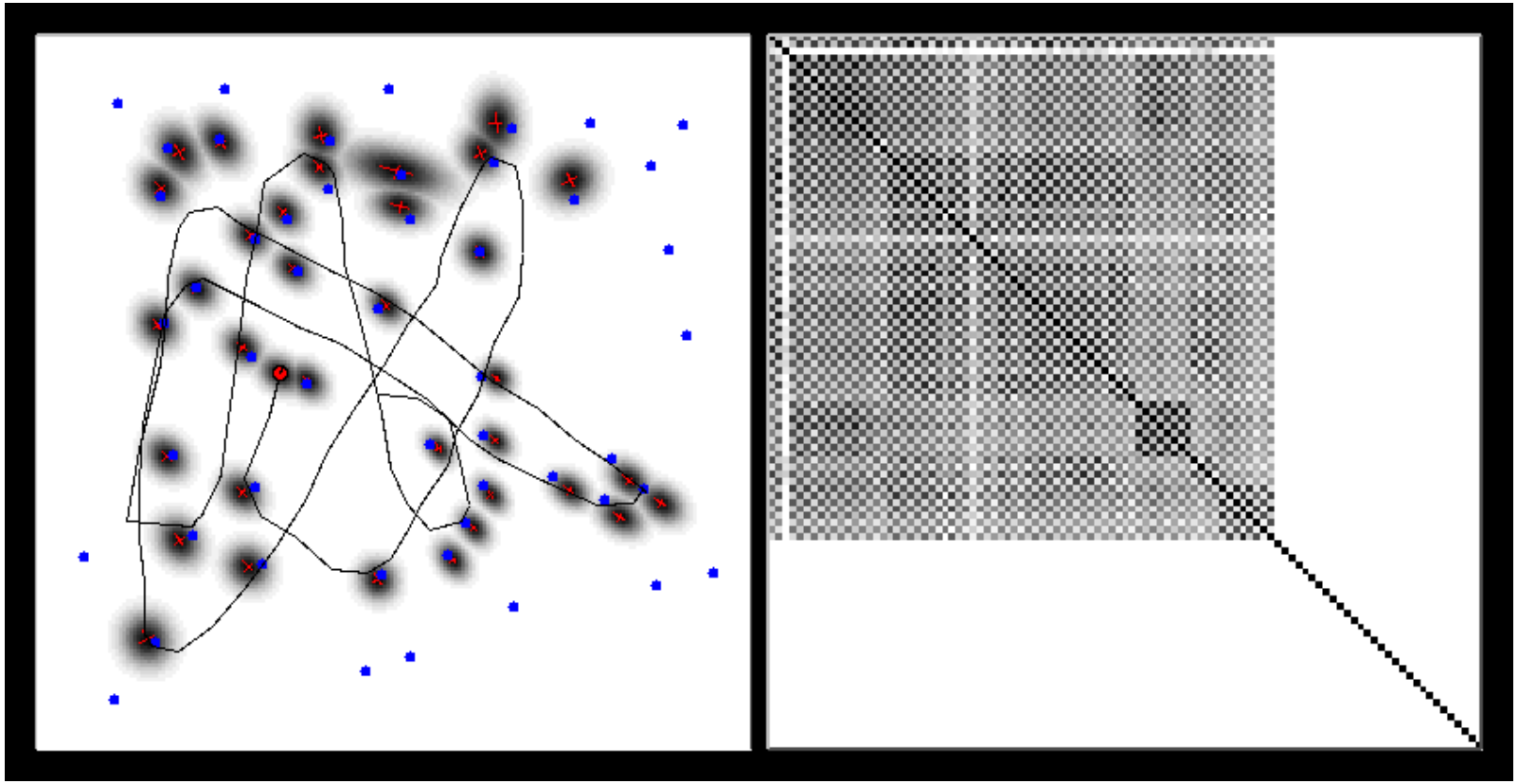
$$\underbrace{\begin{bmatrix} \mathbf{x}_R \\ \mathbf{m}_1 \\ \vdots \\ \mathbf{m}_n \\ \mathbf{m}_{n+1} \end{bmatrix}}_{\mu} \quad \underbrace{\begin{bmatrix} \Sigma_R & \Sigma_{RM_1} & \cdots & \Sigma_{RM_n} & \Sigma_{RM_{n+1}} \\ \Sigma_{M_1 R} & \Sigma_{M_1} & \cdots & \Sigma_{M_1 M_n} & \Sigma_{M_1 M_{n+1}} \\ \vdots & \vdots & \ddots & \vdots & \\ \Sigma_{M_n R} & \Sigma_{M_n M_1} & \cdots & \Sigma_{M_n} & \Sigma_{M_n M_{n+1}} \\ \Sigma_{M_{n+1} R} & \Sigma_{M_{n+1} M_1} & \cdots & \Sigma_{M_{n+1} M_n} & \Sigma_{M_{n+1}} \end{bmatrix}}_{\Sigma}$$

EKF SLAM



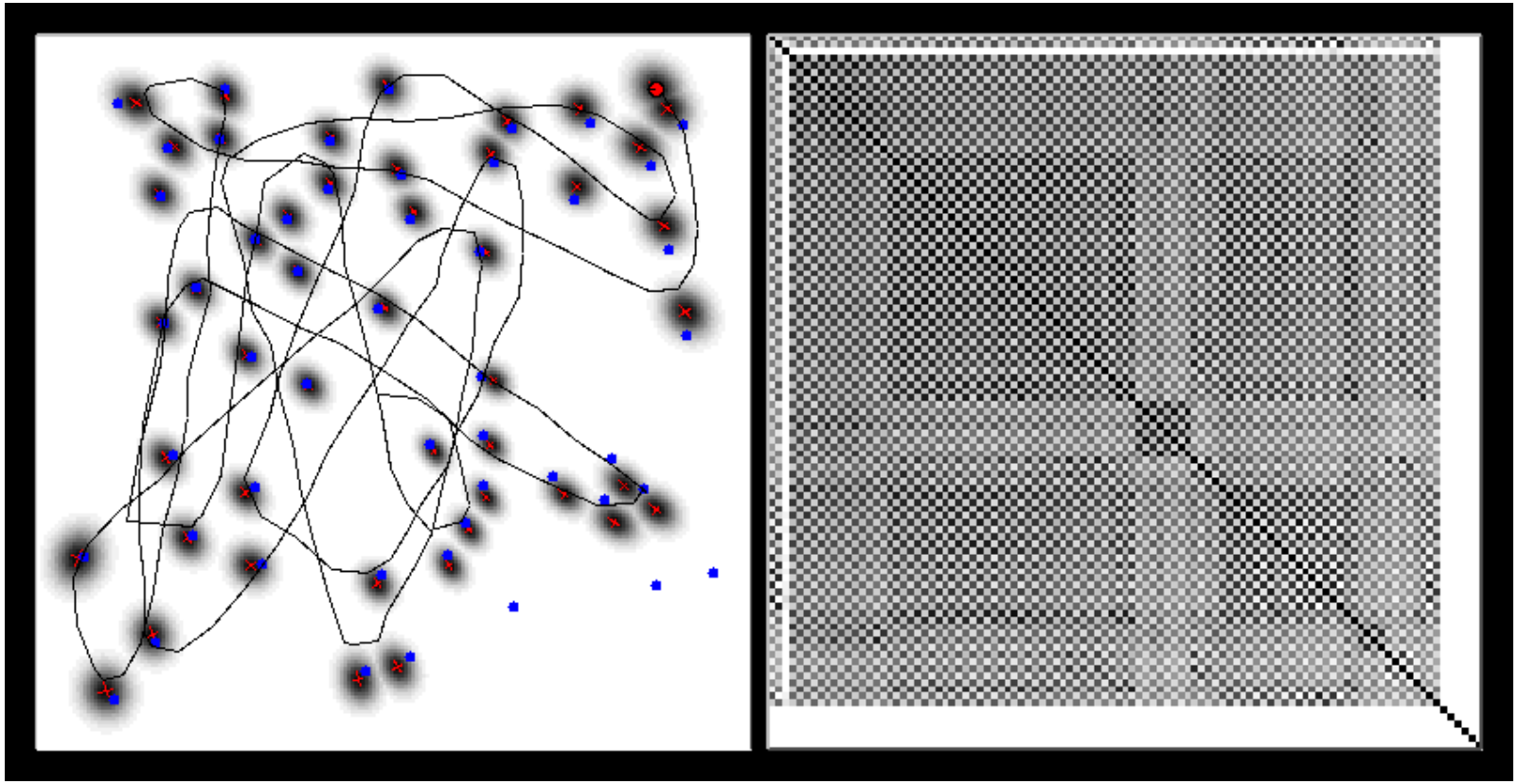
Map Correlation matrix

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?

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

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

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

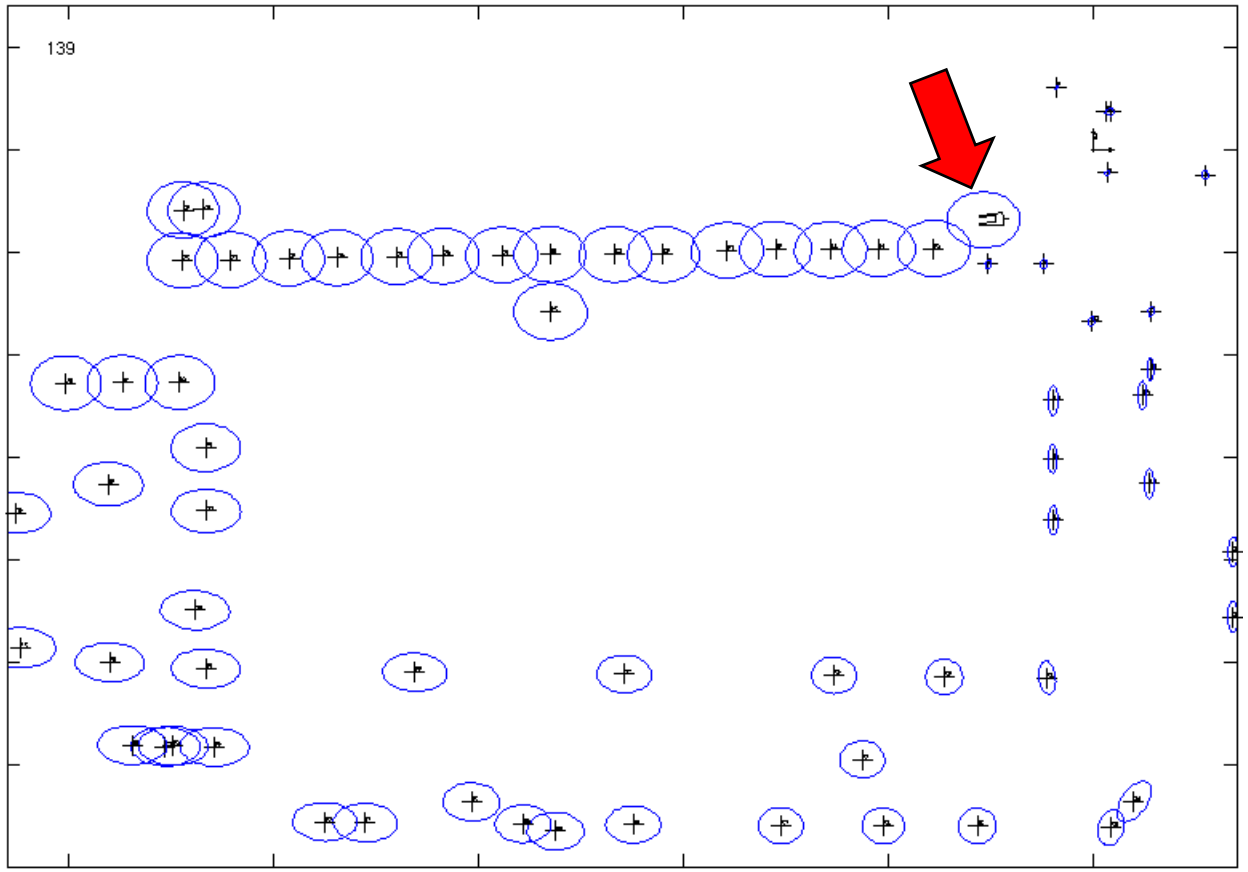
- Landmark and robot uncertainties would become overly optimistic
- Data association would fail
- Multiple map entries of the same landmark
- Inconsistent map

SLAM: Loop Closure

- **Recognizing an already mapped area**, typically after a long exploration path (the robot "closes a loop")
- Structurally identical to data association, but
  - high levels of ambiguity
  - possibly useless validation gates
  - environment symmetries
- Typically, uncertainties **collapse** after a loop closure (whether the closure was correct or not)

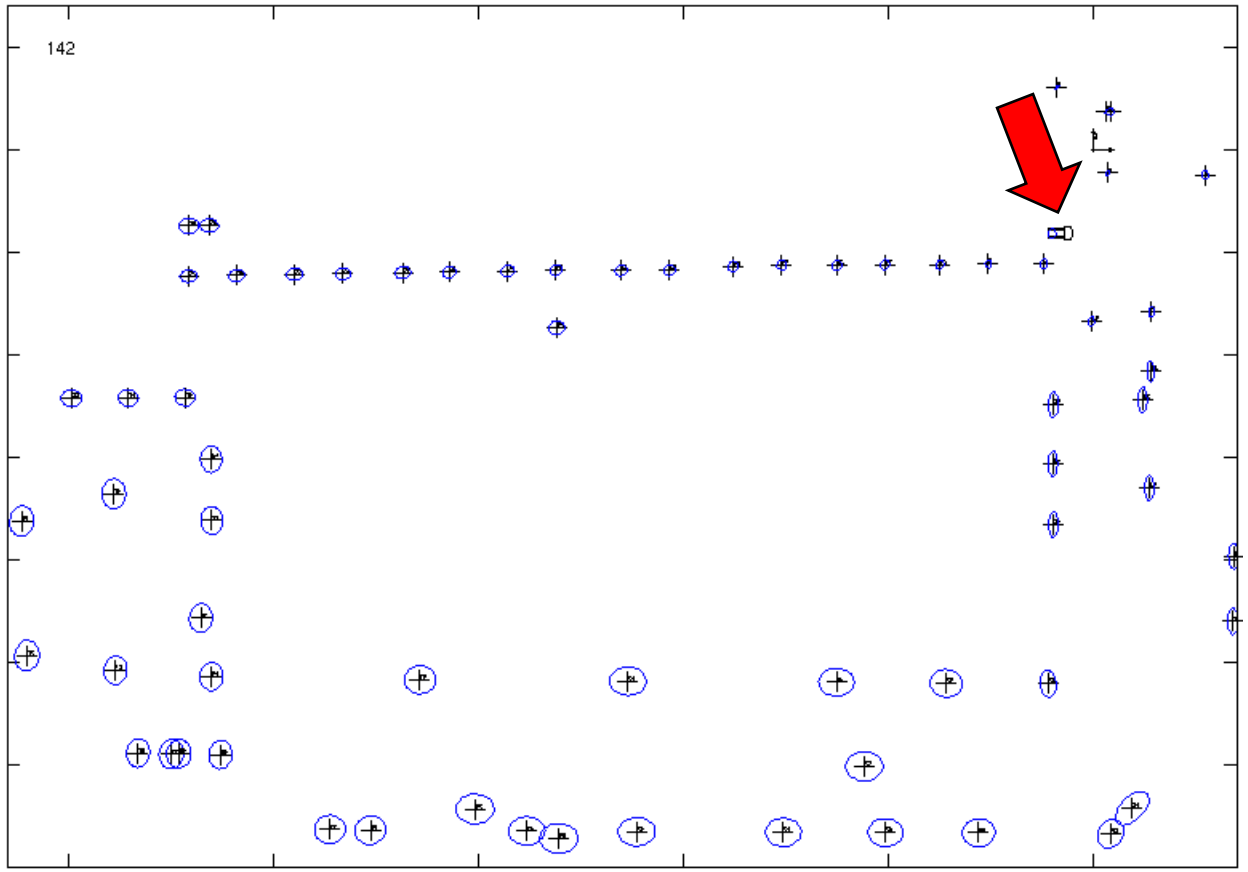
SLAM: Loop Closure

- Before loop closure



SLAM: Loop Closure

- After loop closure



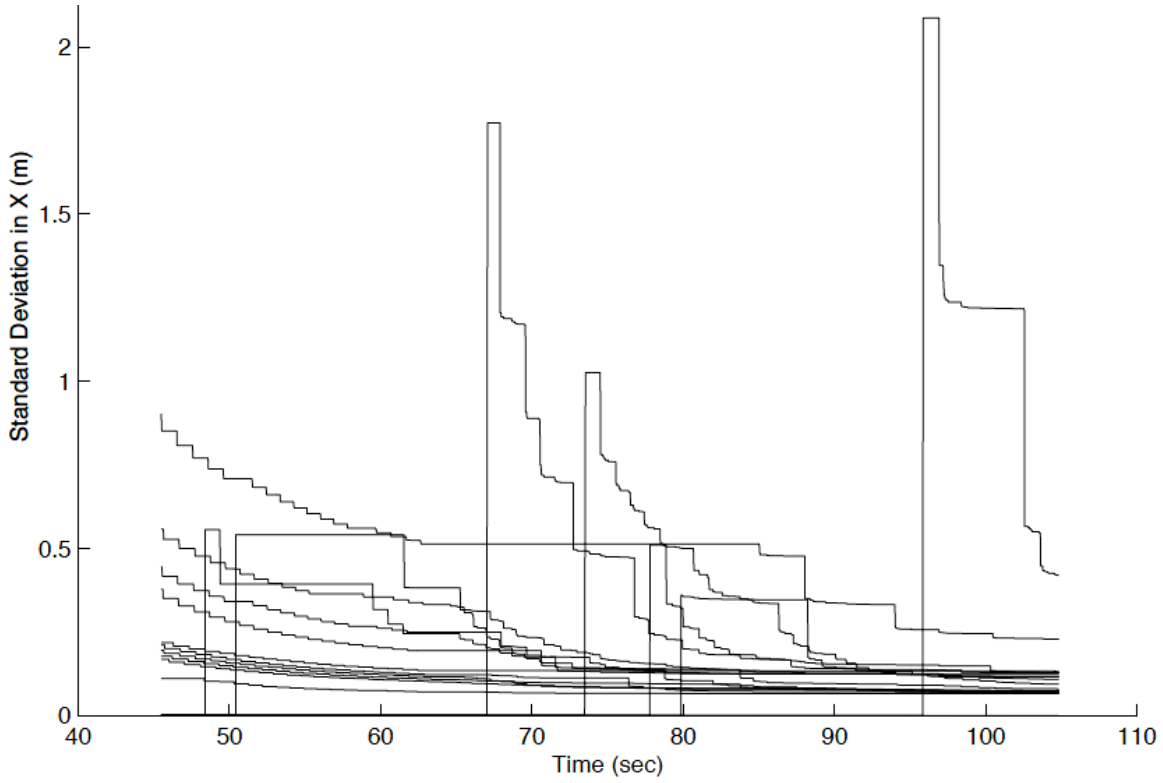
SLAM: Loop Closure

- By revisiting already mapped areas, uncertainties in robot and landmark estimates can be **reduced**
- This can be exploited when **exploring** an environment for the sake of better (e.g., more accurate) maps
- Exploration: the problem of ***where* to acquire new information**

→ See separate chapter on exploration

KF-SLAM Properties (Linear Case)

- The **determinant** of any sub-matrix of the map covariance matrix **decreases monotonically** as successive observations are made

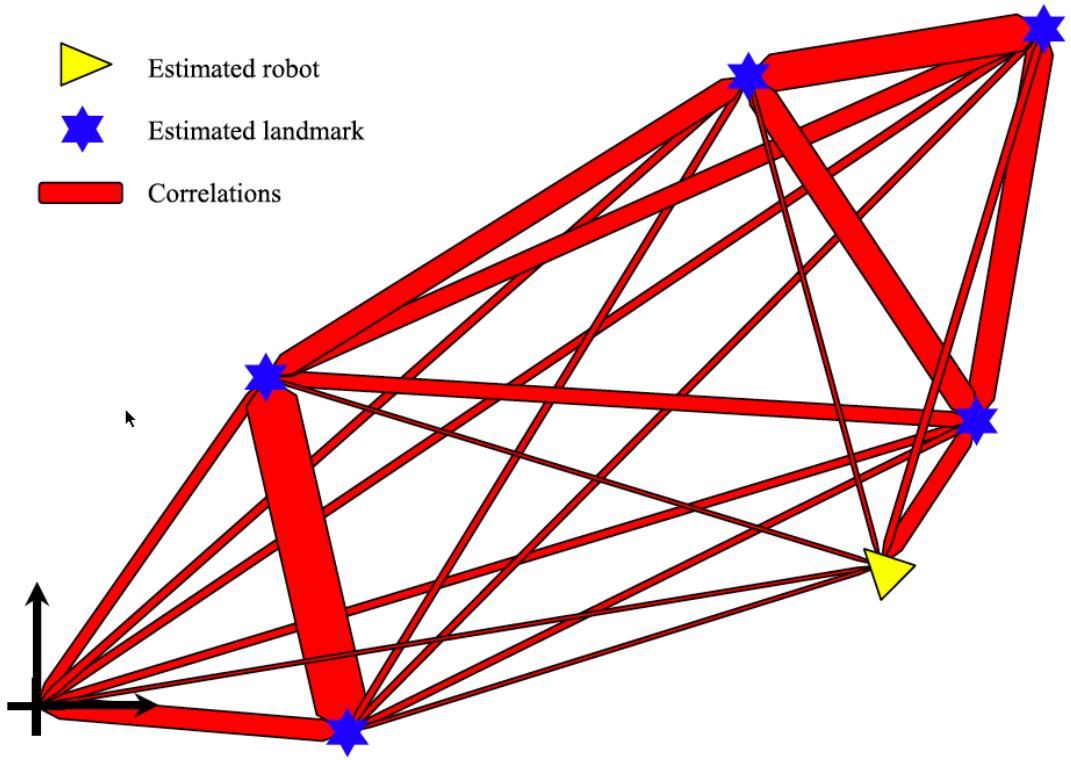


[Dissanayake et al., 2001]

- When a new land-mark is initialized, its **uncertainty** is **maximal**
- Landmark uncertainty **decreases monotonically** with each new observation

KF-SLAM Properties (Linear Case)

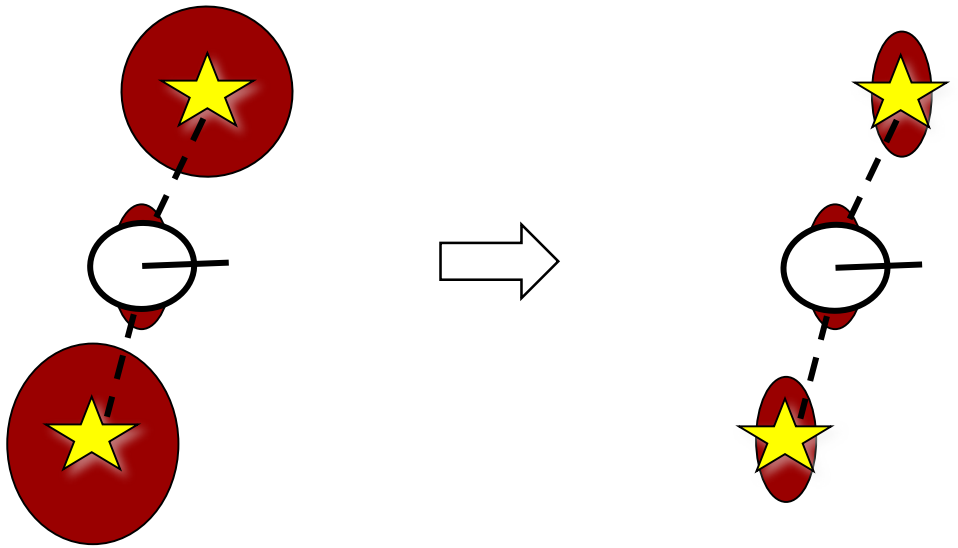
- In the limit, the landmark estimates become **fully correlated**



[Dissanayake et al., 2001]

KF-SLAM Properties (Linear Case)

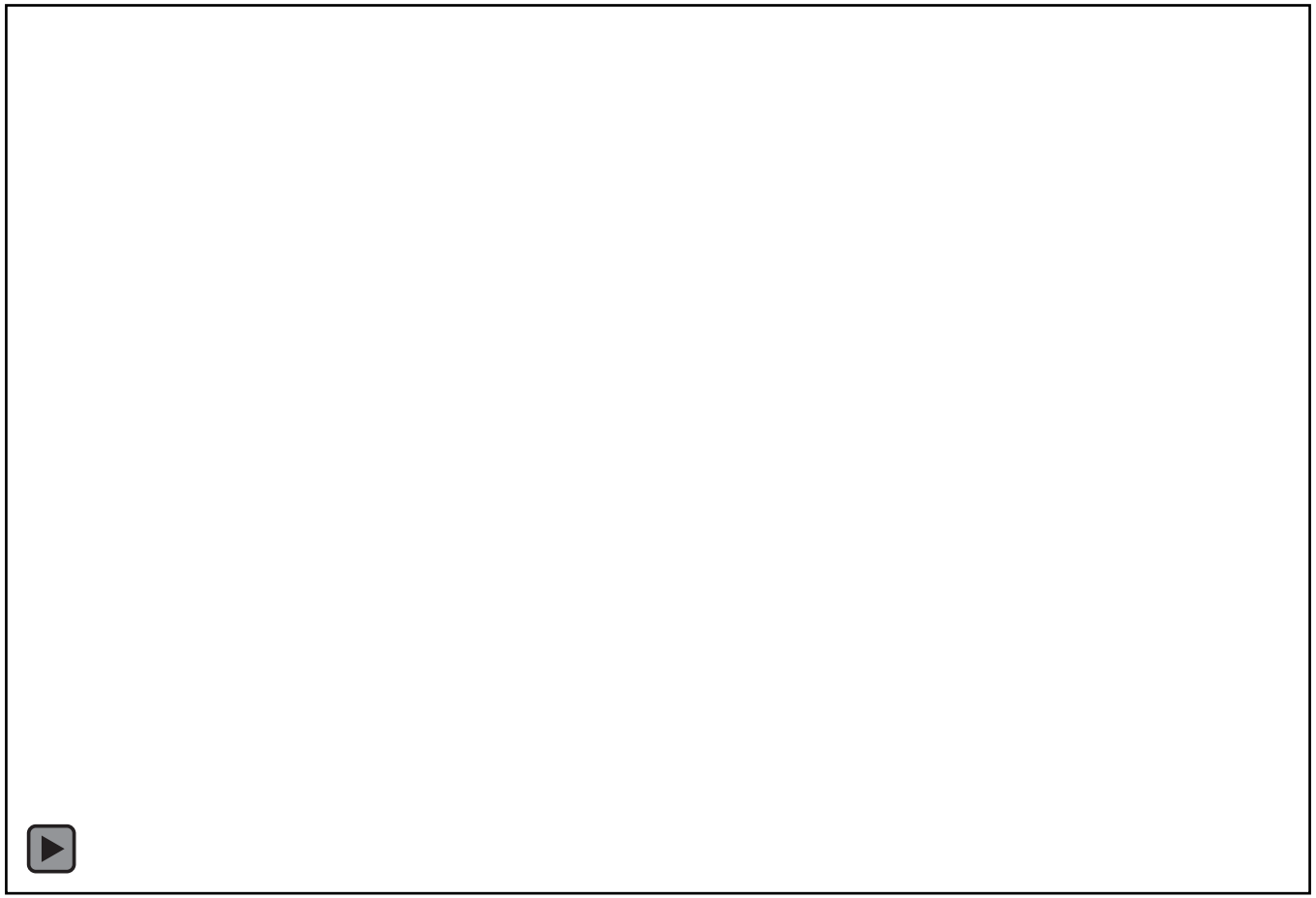
- In the limit, the **covariance** associated with any single landmark location estimate is determined only by the **initial covariance in the vehicle location estimate**.



[Dissanayake et al., 2001]

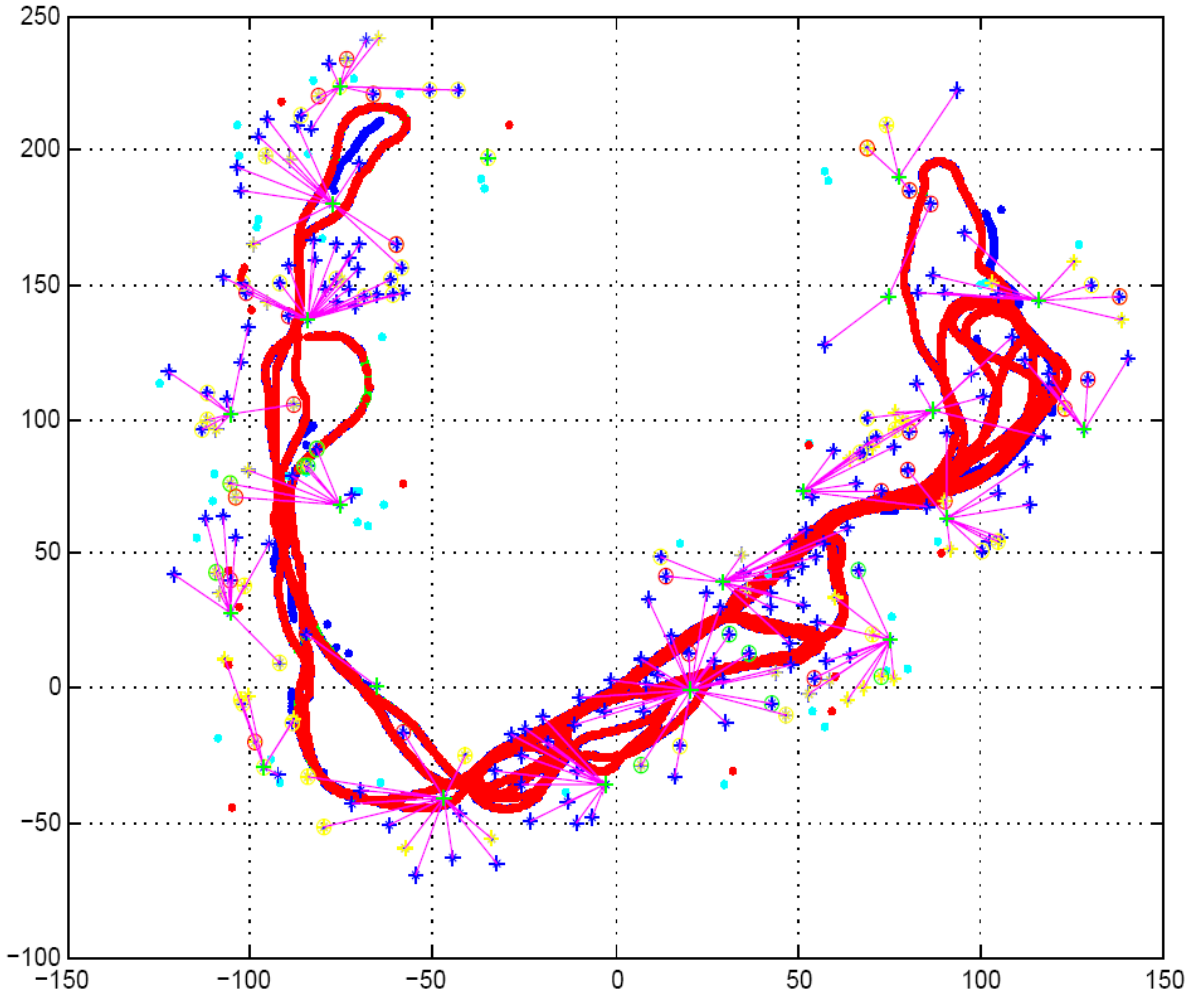
EKF SLAM Example: Victoria Park Dataset

Victoria Park: Data Acquisition



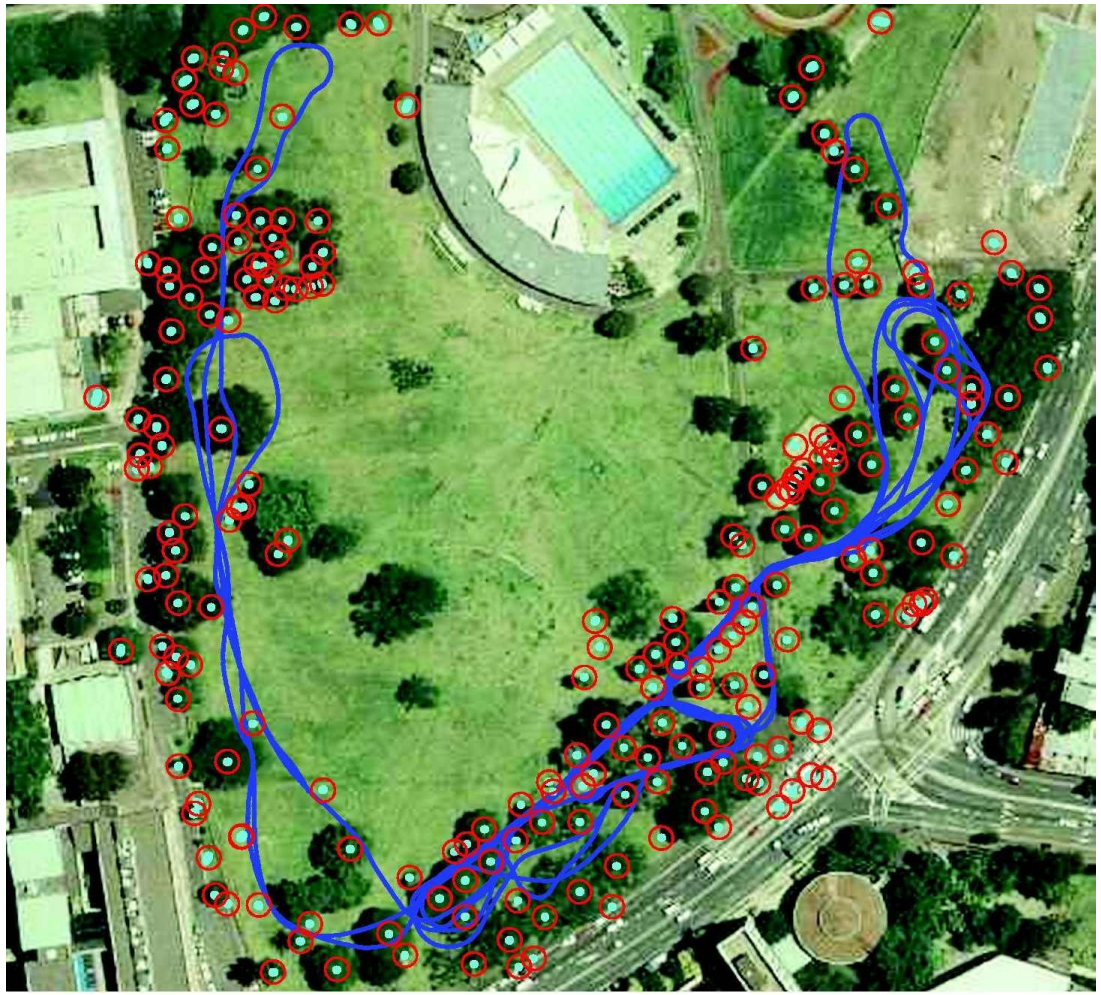
[courtesy by E. Nebot]

Victoria Park: Estimated Trajectory



[courtesy by E. Nebot]

Victoria Park: Landmarks

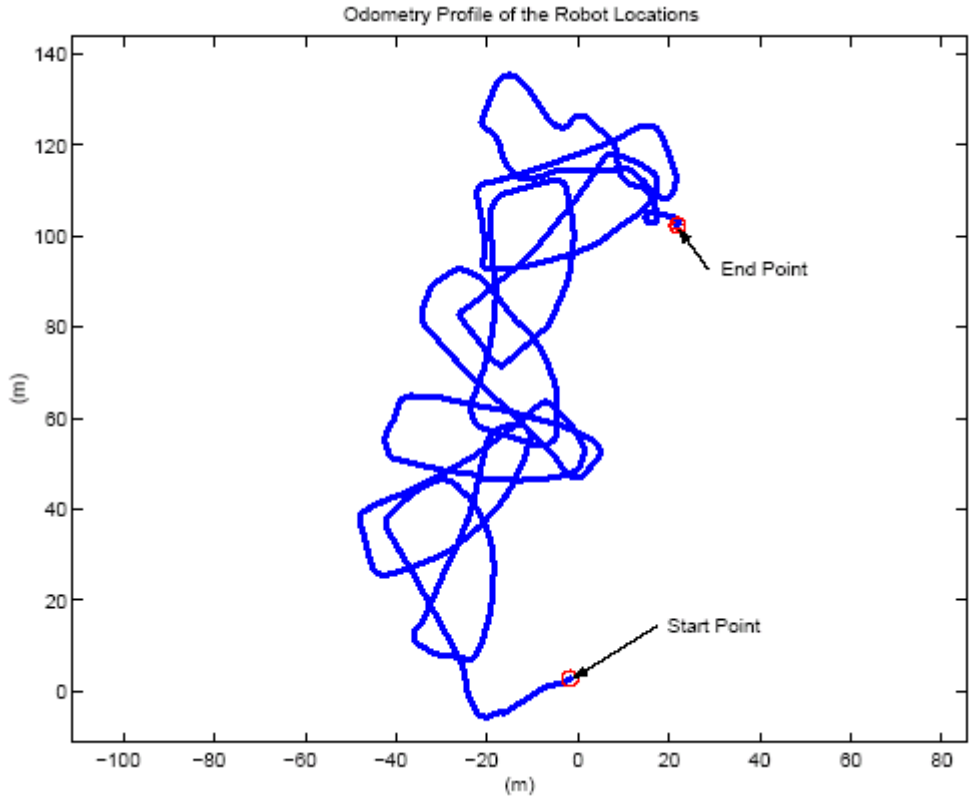


[courtesy by E. Nebot]

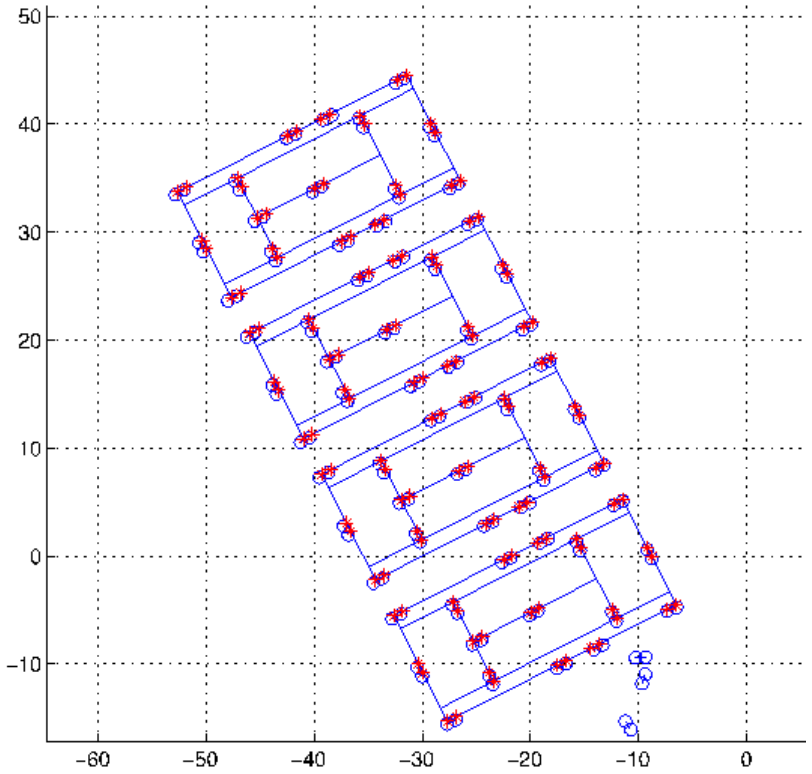
EKF SLAM Example: Tennis Court

[courtesy by J. Leonard]

EKF SLAM Example: Tennis Court



odometry

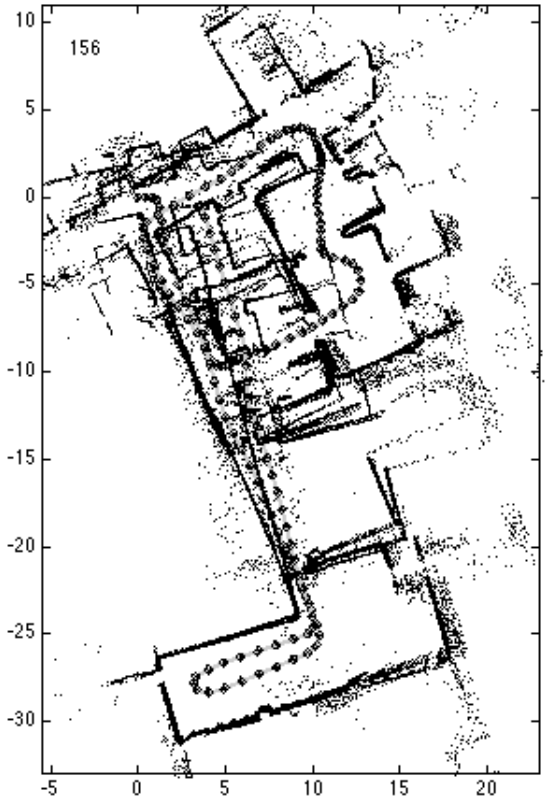


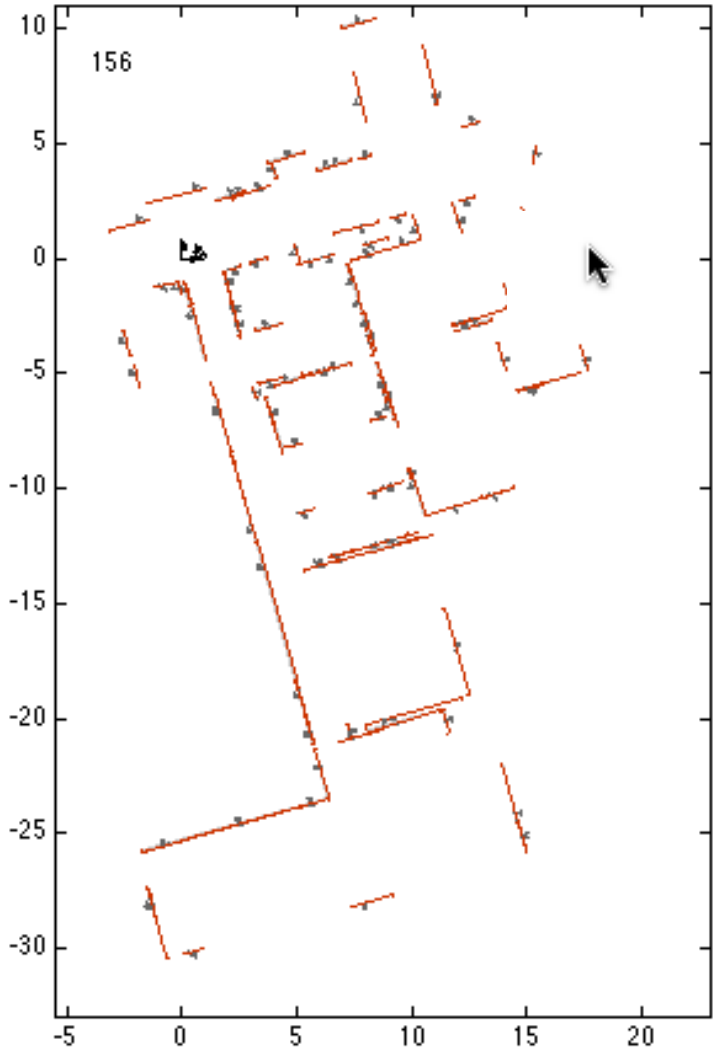
estimated map

[courtesy by John Leonard]

EKF SLAM Example: Line Features

- KTH Bakery Data Set





[Wulf et al., ICRA 04]

EKF-SLAM: Complexity

- Cost per step: quadratic in n, the number of landmarks: $O(n^2)$
- Total cost to build a map with n landmarks: $O(n^3)$
- Memory consumption: $O(n^2)$
- Problem: becomes computationally challenging for very large maps!
- There exist variants to circumvent these problems

EKF-SLAM: Summary

- The first SLAM solution
- Convergence proof for linear Gaussian case
- Can diverge if nonlinearities are large
  (and the real world is nonlinear ...)
- Can only deal with a single mode
- Successful in medium-scale scenes with landmarks
- Approximations exists to reduce the computational challenges

Particle Filters for SLAM

- Feature-based representations
- Grid-based representations

Particle Filters

- Represent belief by random samples
- Estimation of non-Gaussian, nonlinear processes

- Sampling Importance Resampling (SIR) principle
  - Draw the new generation of particles
  - Assign an importance weight to each particle
  - Resample

- Typical application scenarios are tracking, localization, …

Localization vs. SLAM

- A particle filter can be used to solve both problems
- Localization: state space $<x, y, \theta>$
- SLAM: state space $<x, y, \theta, map>$
  - for landmark maps = $<l_1, l_2, \ldots, l_m>$
  - for grid maps = $<c_{11}, c_{12}, \ldots, c_{1n}, c_{21}, \ldots, c_{nm}>$

- **Problem:** The number of particles needed to represent a posterior grows exponentially with the dimension of the state space!

Dependencies

- Is there a dependency between certain dimensions of the state space?
- If so, can we use the dependency to solve the problem more efficiently?

Dependencies

- Is there a dependency between certain dimensions of the state space?
- If so, can we use the dependency to solve the problem more efficiently?
- In the SLAM context
  - The map depends on the poses of the robot.
  - We know how to build a map given the position of the sensor is known.

Factored Posterior (Landmarks)

poses map observations & movements

$$p(x_{1:t}, l_{1:m} \mid z_{1:t}, u_{0:t-1}) =$$

$$p(x_{1:t} \mid z_{1:t}, u_{0:t-1}) \cdot p(l_{1:m} \mid x_{1:t}, z_{1:t})$$

Factorization first introduced by Murphy in 1999

Factored Posterior (Landmarks)

poses map observations & movements

$$p(x_{1:t}, l_{1:m} \mid z_{1:t}, u_{0:t-1}) =$$

$$p(x_{1:t} \mid z_{1:t}, u_{0:t-1}) \cdot p(l_{1:m} \mid x_{1:t}, z_{1:t})$$

SLAM posterior

Robot path posterior

landmark positions

Does this help to solve the problem?

Factorization first introduced by Murphy in 1999

Rao-Blackwellization

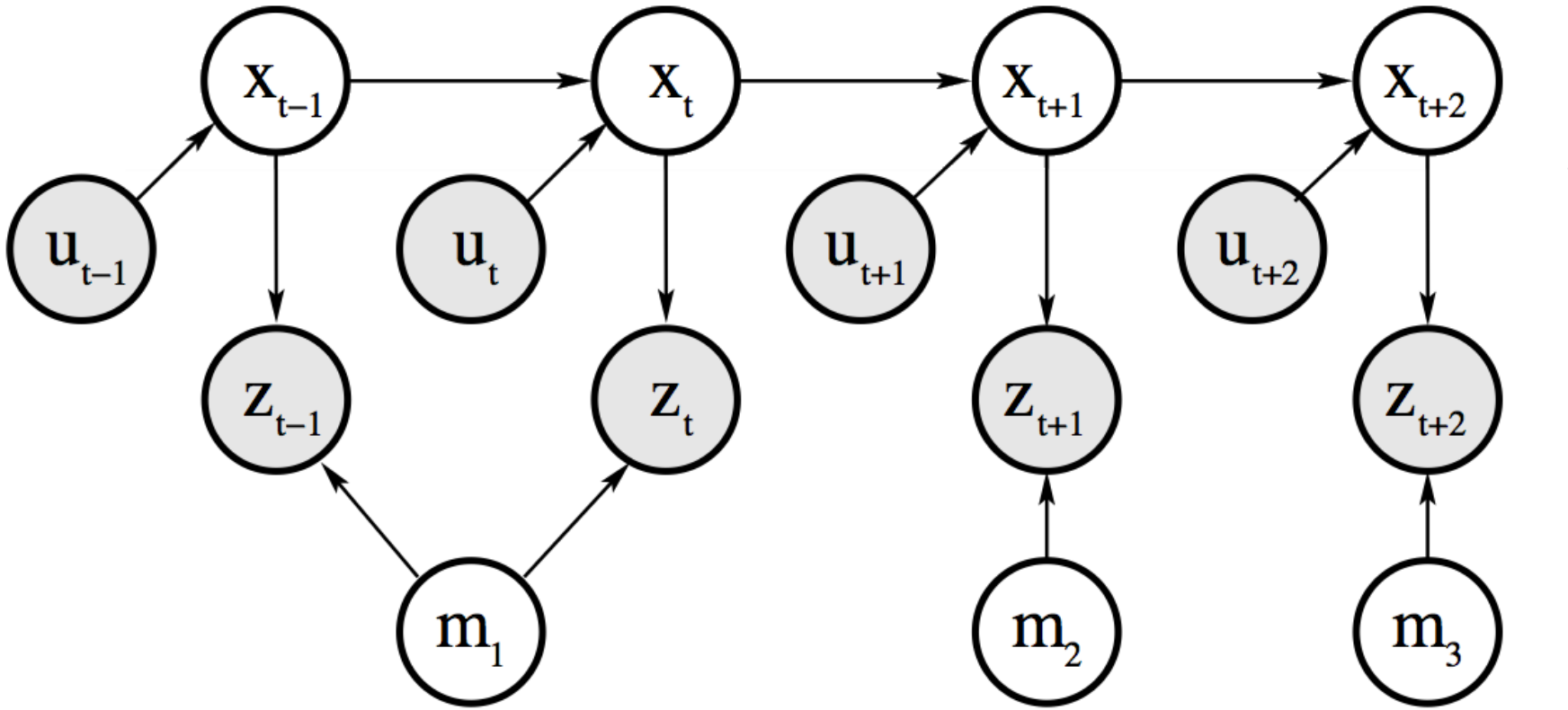
- Factorization to exploit dependencies between variables:

$$p(a, b) \;=\; p(a) \cdot p(b \mid a)$$

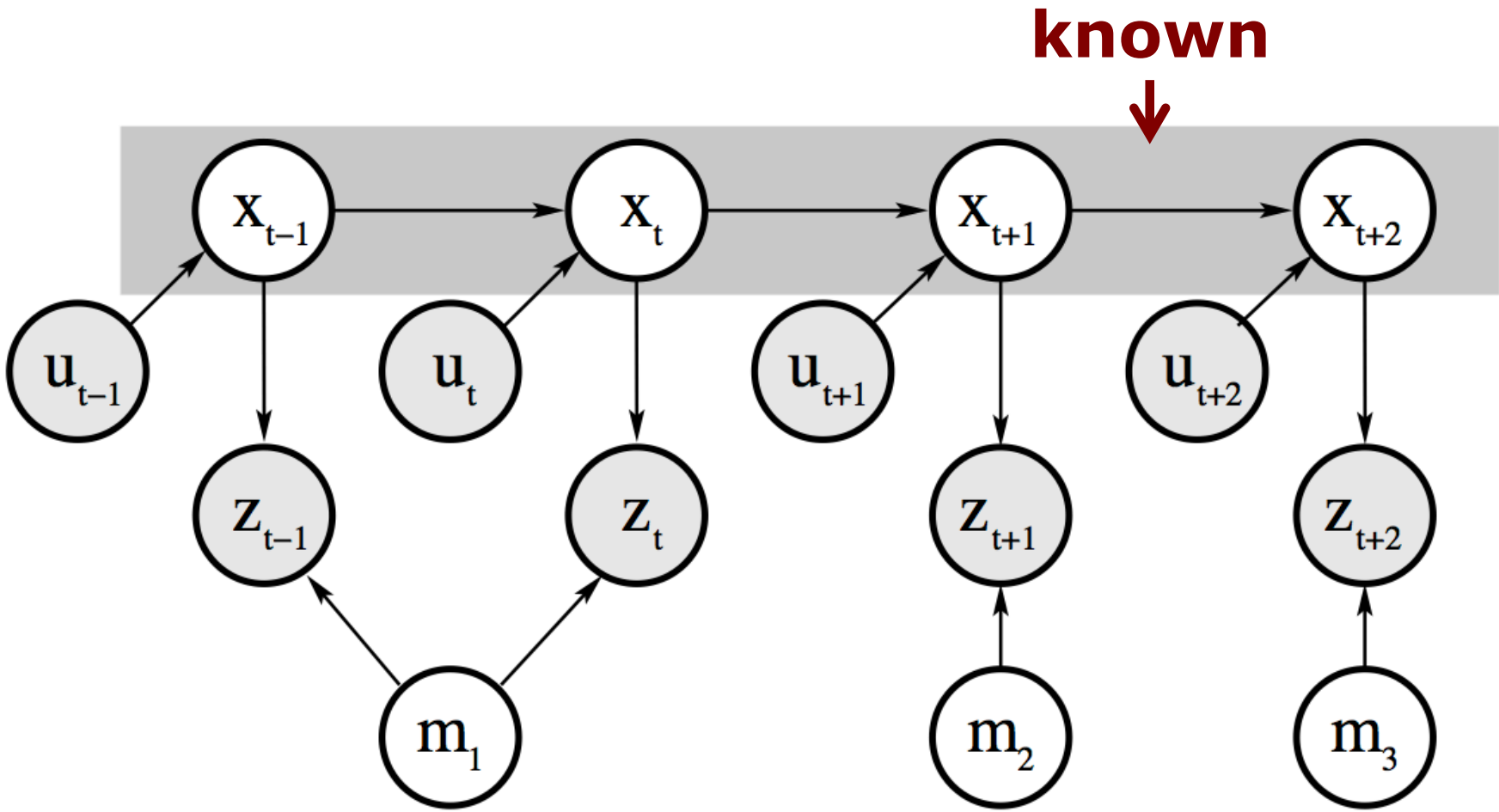
- If $p(b \mid a)$ can be computed in closed form, represent only $p(a)$ with samples and compute $p(b \mid a)$ for every sample

- It comes from the Rao-Blackwell theorem

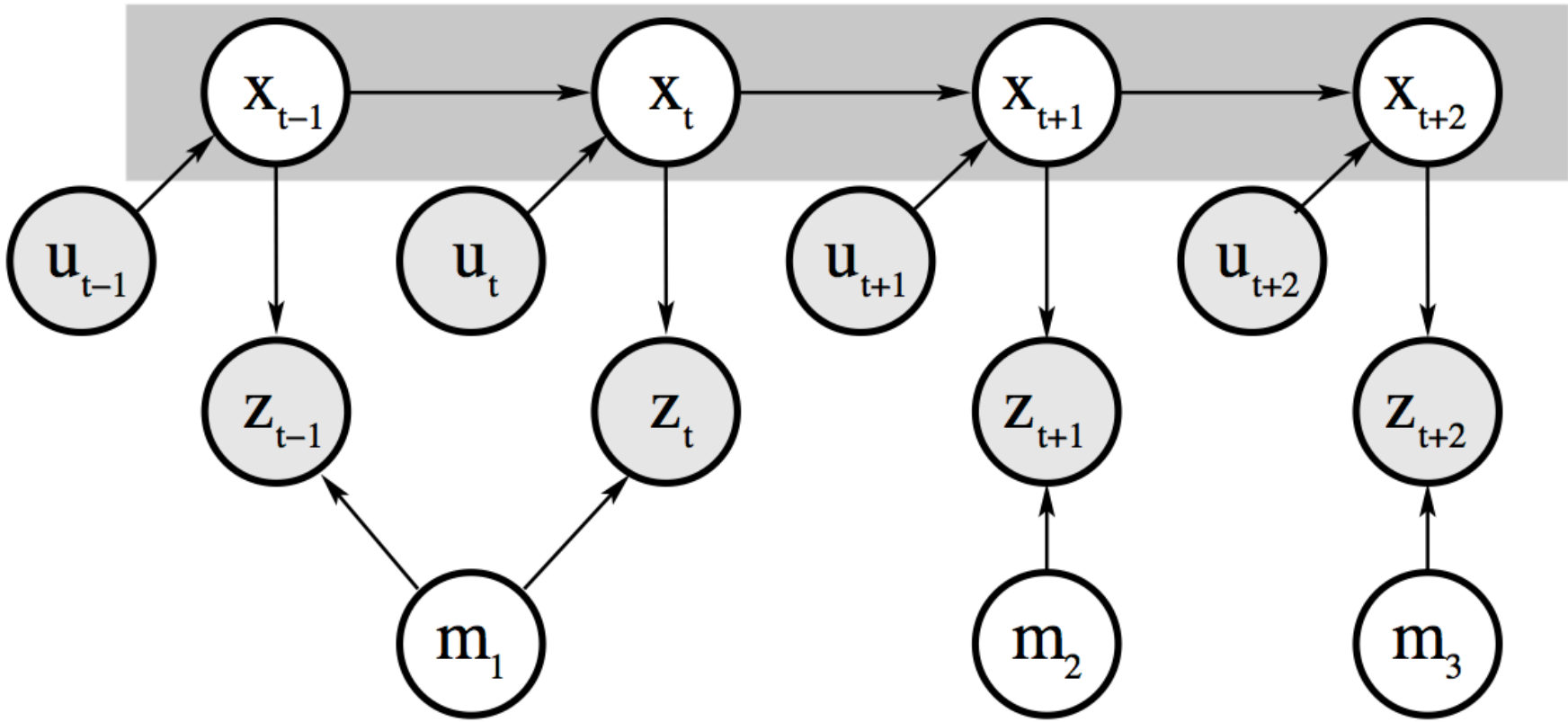
Revisit the Graphical Model



Revisit the Graphical Model



Landmarks are Conditionally Independent Given the Poses



Landmark variables are all disconnected (i.e. independent) given the robot's path

Factored Posterior

$$
\begin{aligned}
& p(x_{1:t}, l_{1:m} \mid z_{1:t}, u_{0:t-1}) \\
&= p(x_{1:t} \mid z_{1:t}, u_{0:t-1}) \cdot p(l_{1:m} \mid x_{1:t}, z_{1:t}) \\
&= p(x_{1:t} \mid z_{1:t}, u_{0:t-1}) \cdot \prod_{i=1}^{M} p(l_i \mid x_{1:t}, z_{1:t})
\end{aligned}
$$

Robot path posterior (localization problem)

Conditionally independent landmark positions

Rao-Blackwellization for SLAM

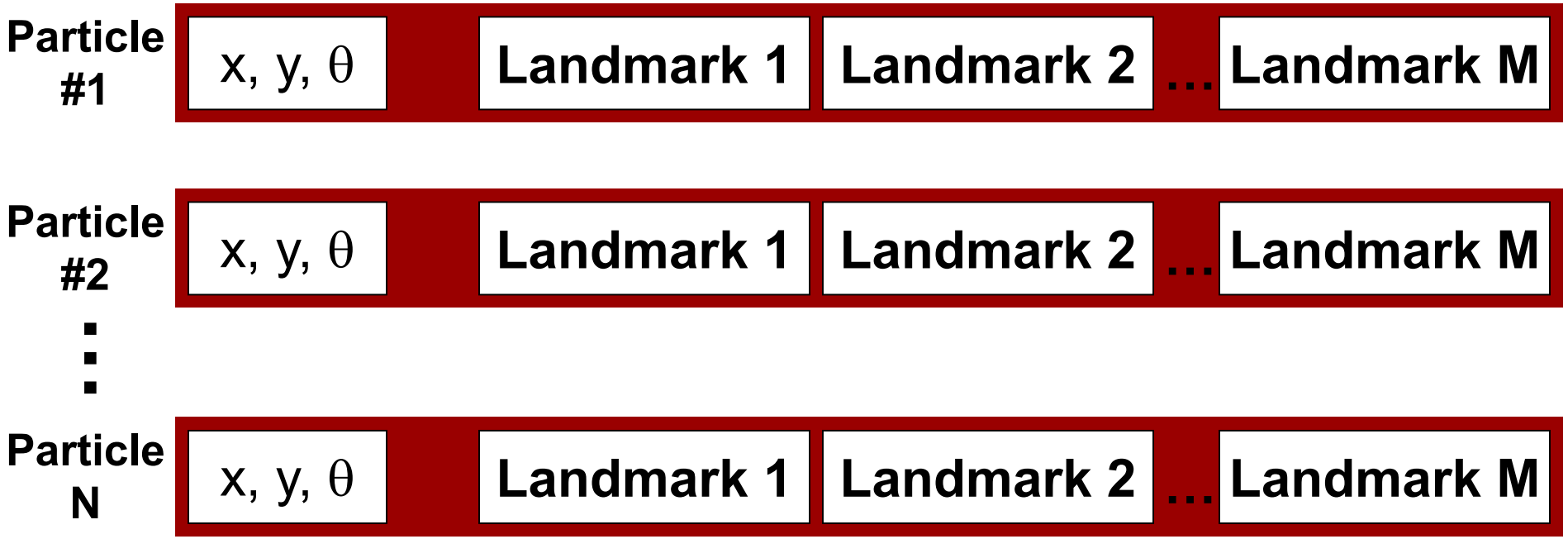
$$p(x_{1:t}, l_{1:m} \mid z_{1:t}, u_{0:t-1}) =$$

$$p(x_{1:t} \mid z_{1:t}, u_{0:t-1}) \cdot \prod_{i=1}^{M} p(l_i \mid x_{1:t}, z_{1:t})$$

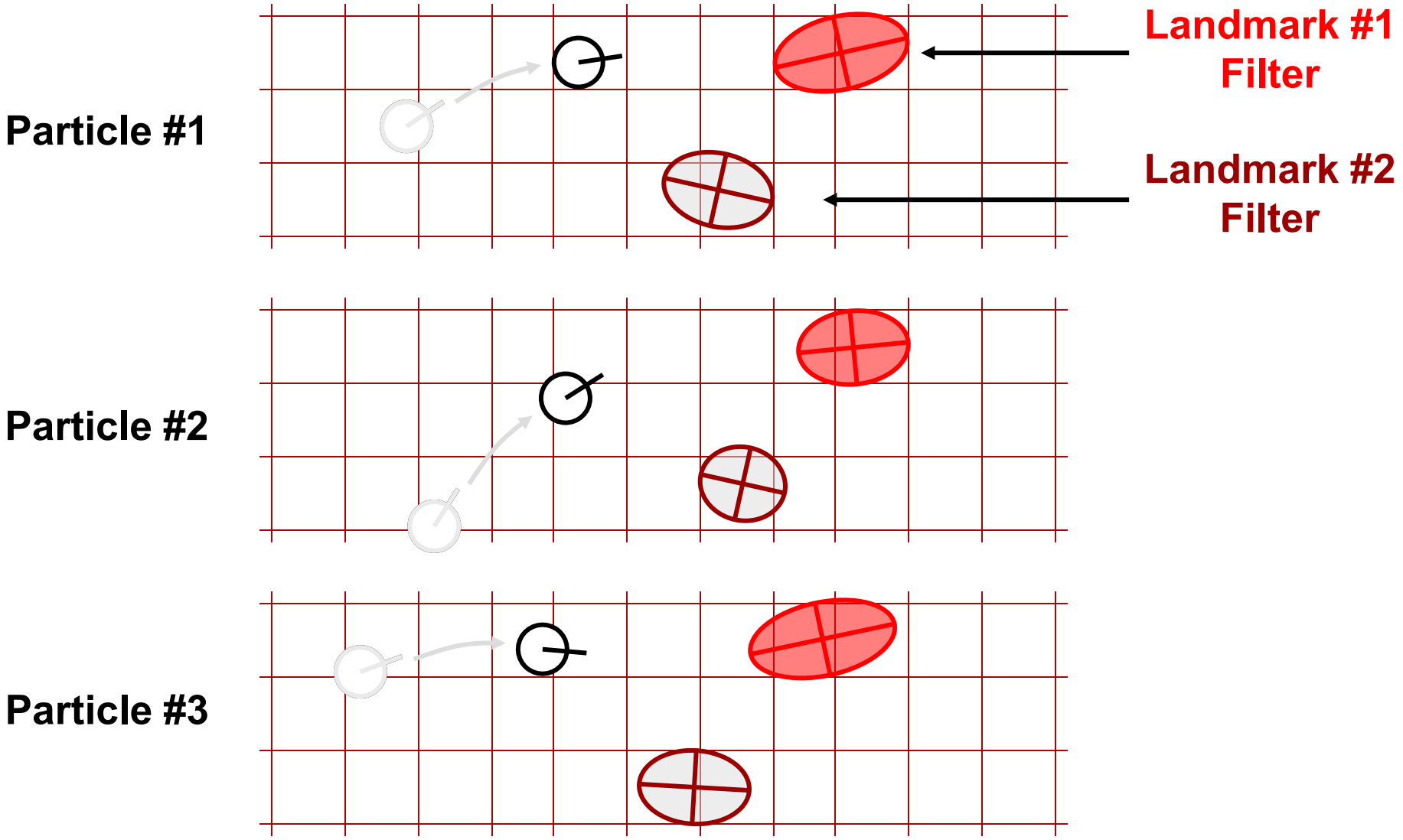
- Given that the second term can be computed efficiently, particle filtering becomes possible!

FastSLAM

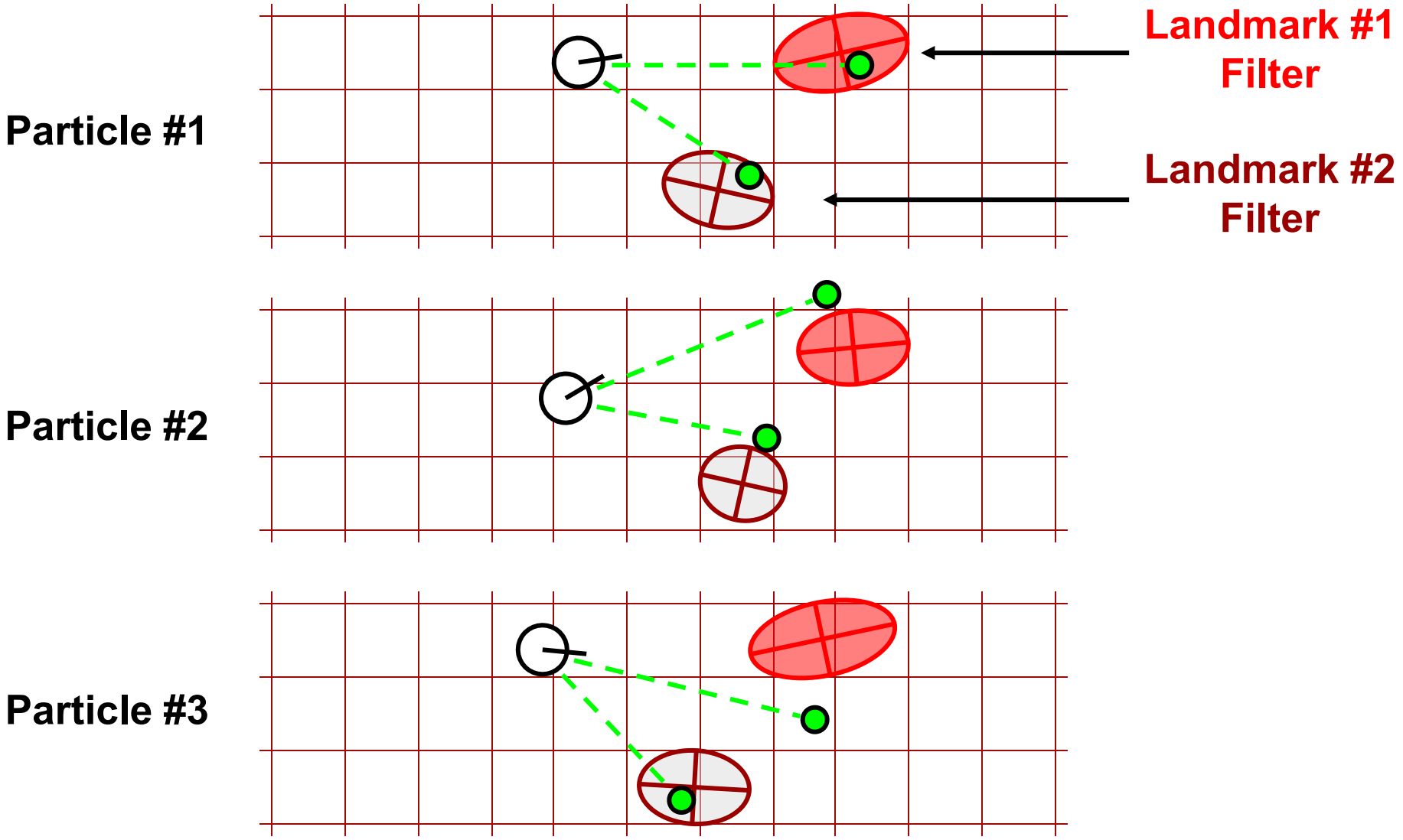
- Rao-Blackwellized particle filtering based on landmarks [Montemerlo et al., 2002]
- Each landmark is represented by a 2x2 Extended Kalman Filter (EKF)
- Each particle therefore has to maintain *M* EKFs



FastSLAM – Action Update

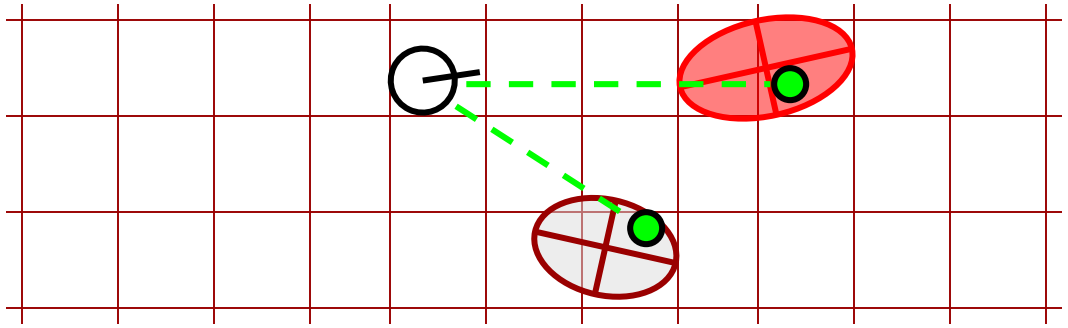


FastSLAM – Sensor Update



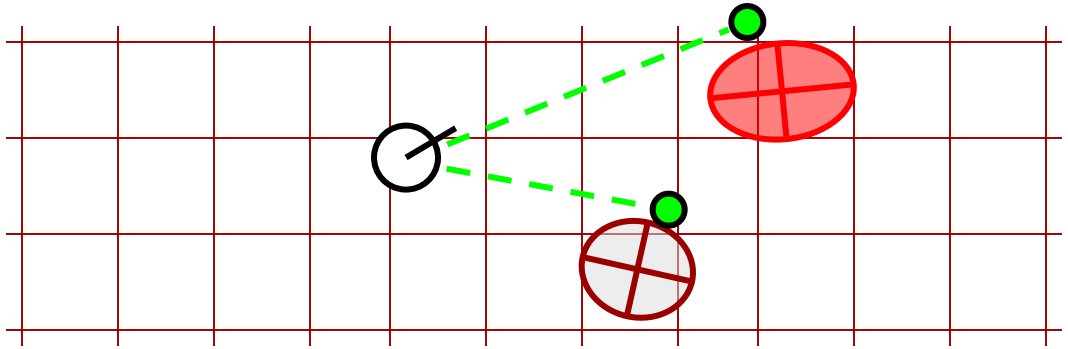
FastSLAM – Sensor Update

Particle #1



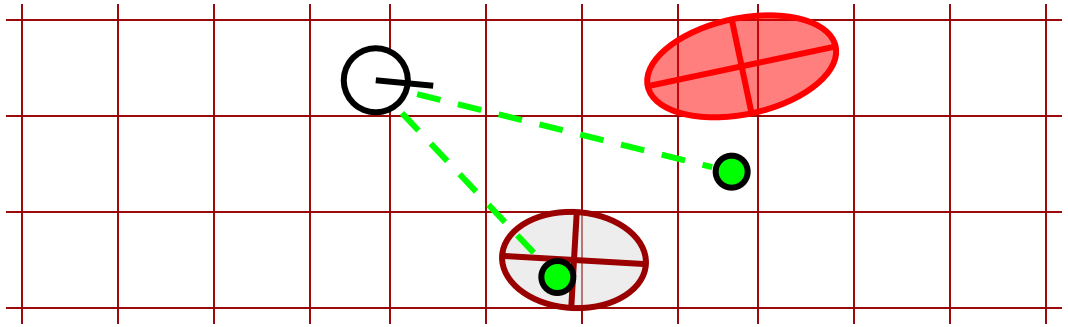
Weight = 0.8

Particle #2



Weight = 0.4

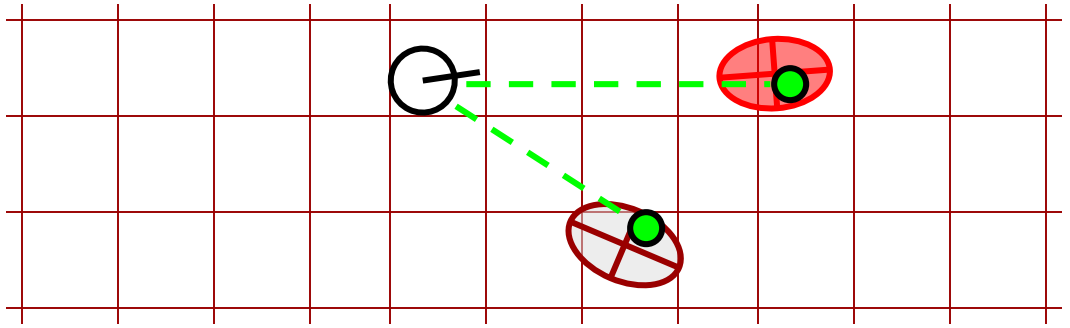
Particle #3



Weight = 0.1

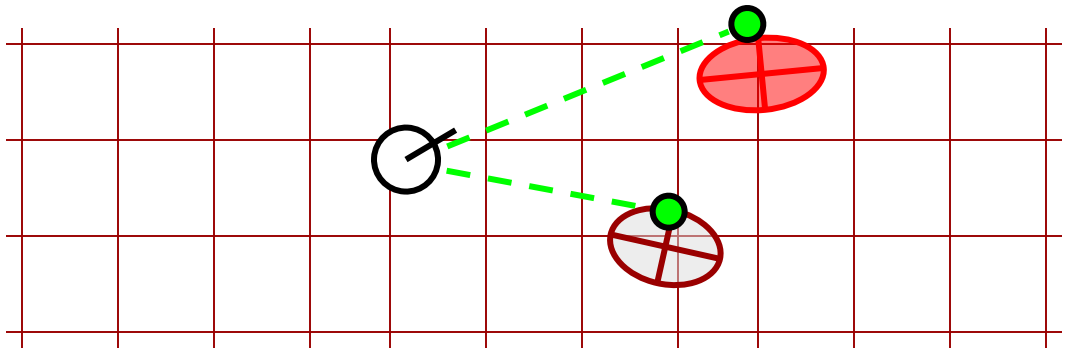
FastSLAM – Sensor Update

Particle #1



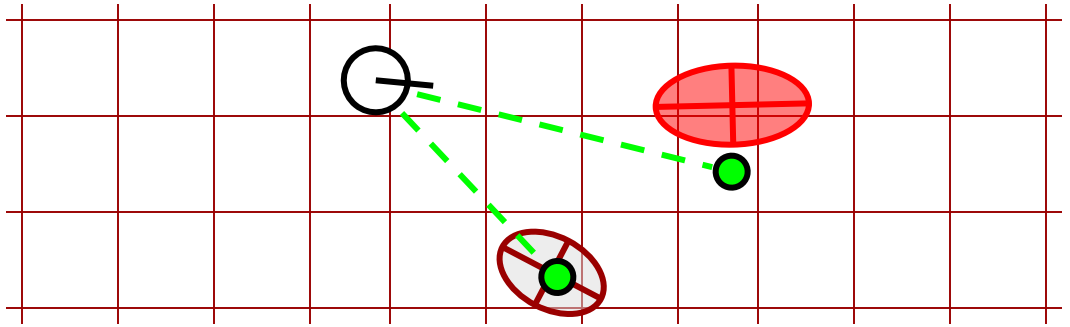
Update map of particle #1

Particle #2



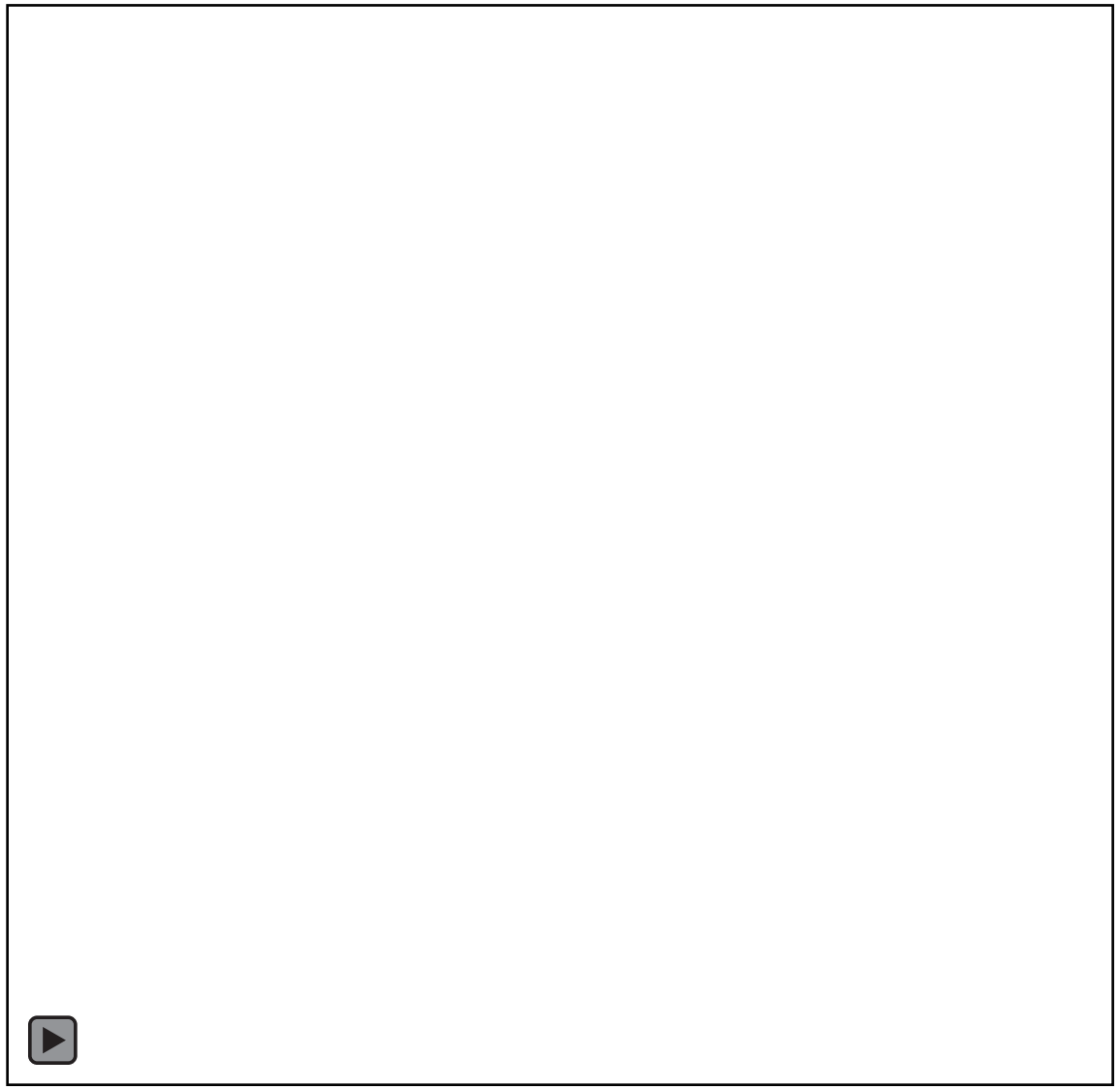
Update map of particle #2

Particle #3



Update map of particle #3

FastSLAM - Video



Odometry: dotted

Ground truth: blue

Best particle: red

FastSLAM Complexity – Naive

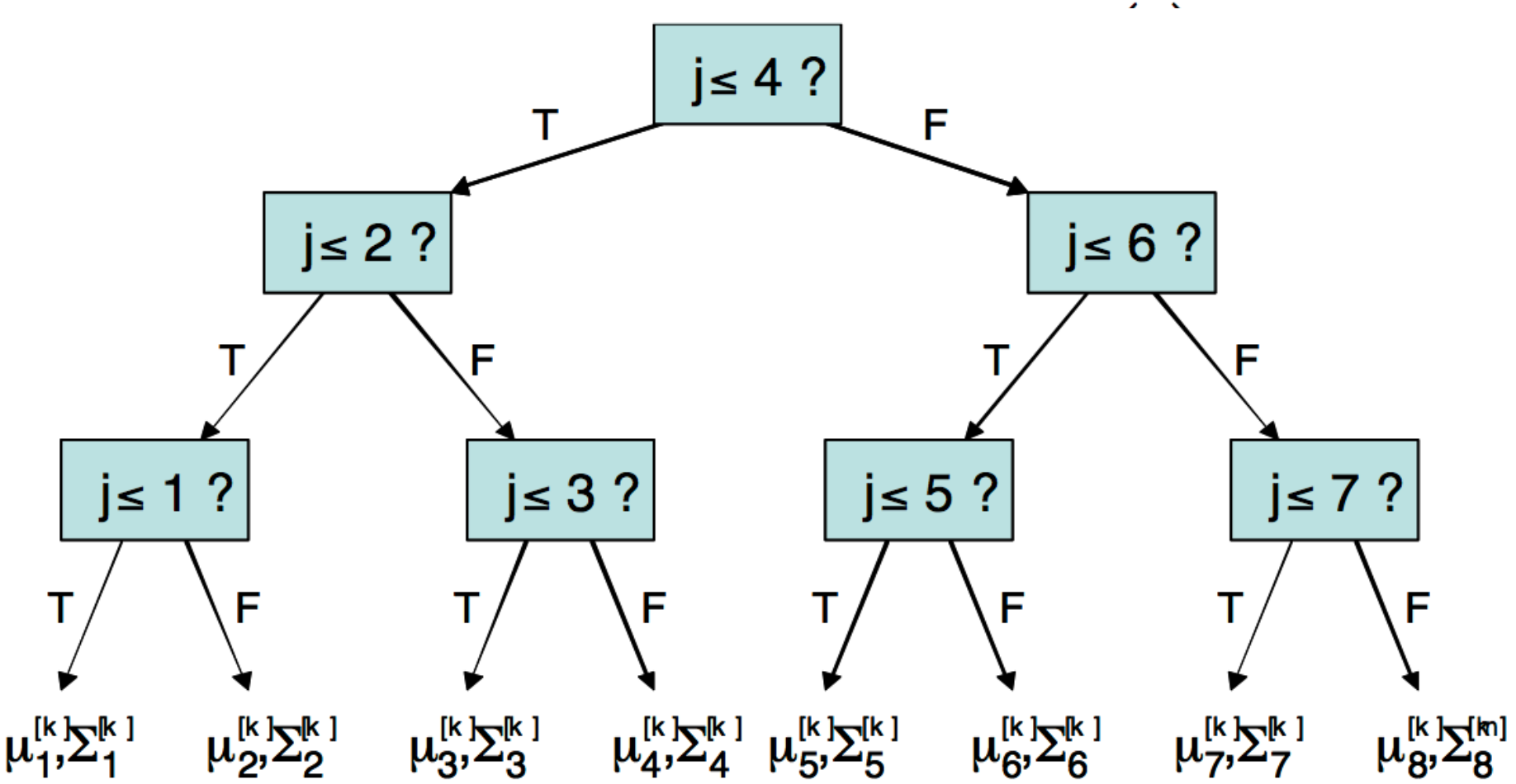
- Update robot particles based on the control $\mathcal{O}(N)$
- Incorporate an observation into the Kalman filters (given the data association) $\mathcal{O}(N)$
- Resample particle set $\mathcal{O}(NM)$

Expensive copying of memory

$\mathcal{O}(NM)$

N = Number of particles
M = Number of map features

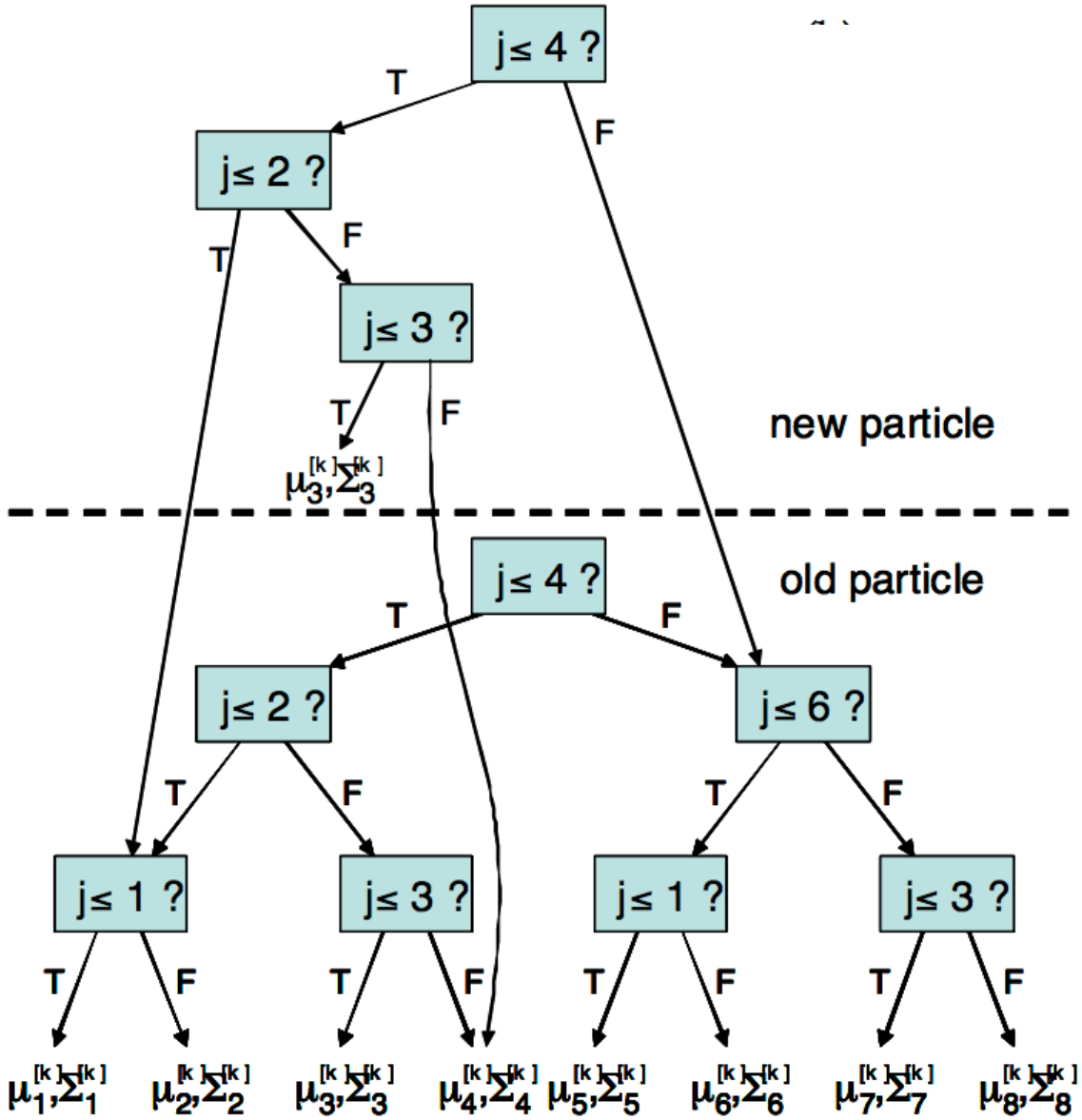
A Better Data Structure for FastSLAM



Courtesy: M. Montemerlo

A Better Data Structure for FastSLAM

Sharing of map features between particles



FastSLAM Complexity

- Update robot particles based on the control

$$\mathcal{O}(N)$$

- Incorporate an observation into the Kalman filters (given the data association)

$$\mathcal{O}(N \log M)$$

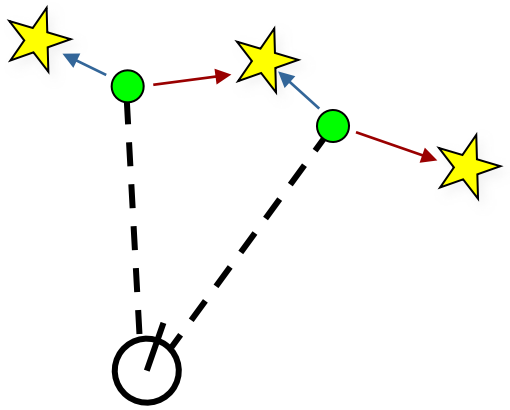
- Resample particle set

$$\frac{\mathcal{O}(N \log M)}{\mathcal{O}(N \log M)}$$

N = Number of particles
M = Number of map features

Data Association Problem

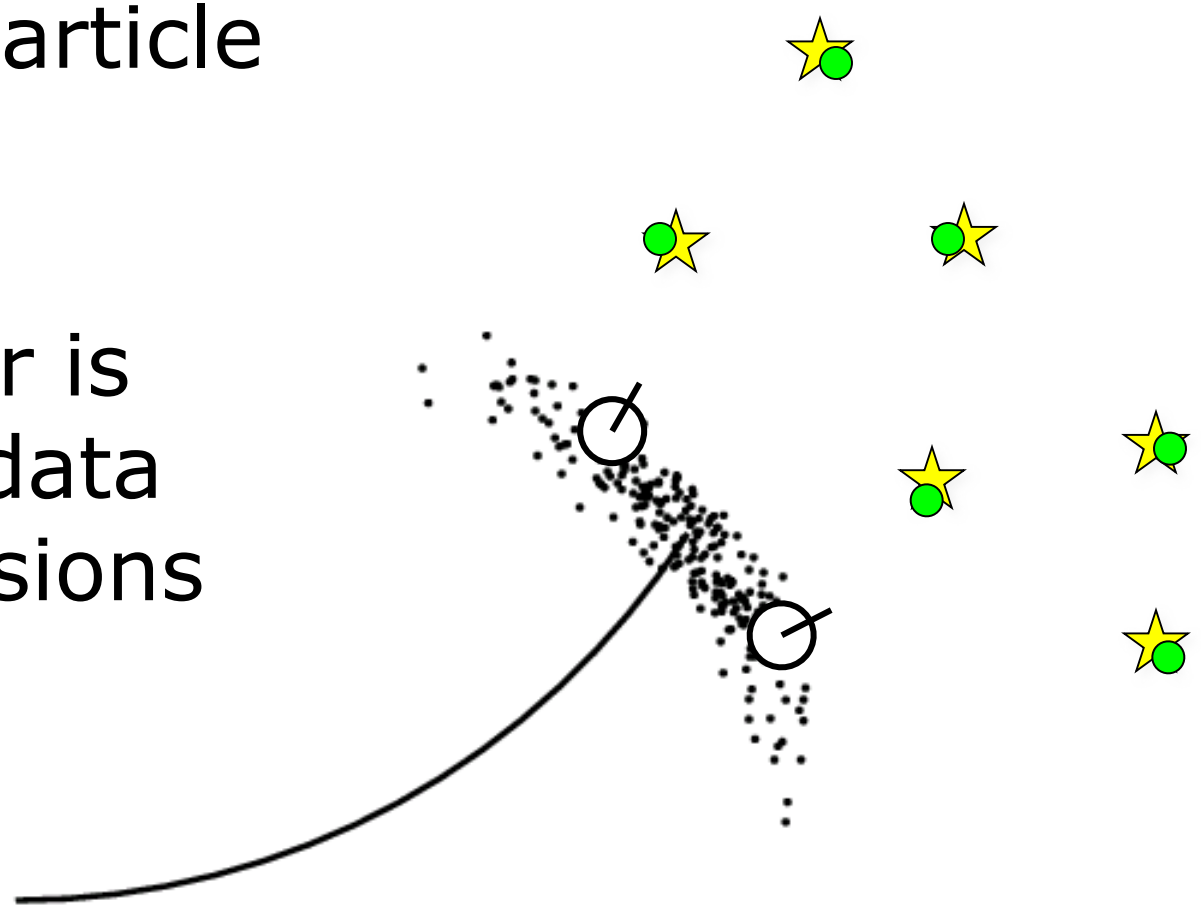
- Which observation belongs to which landmark?



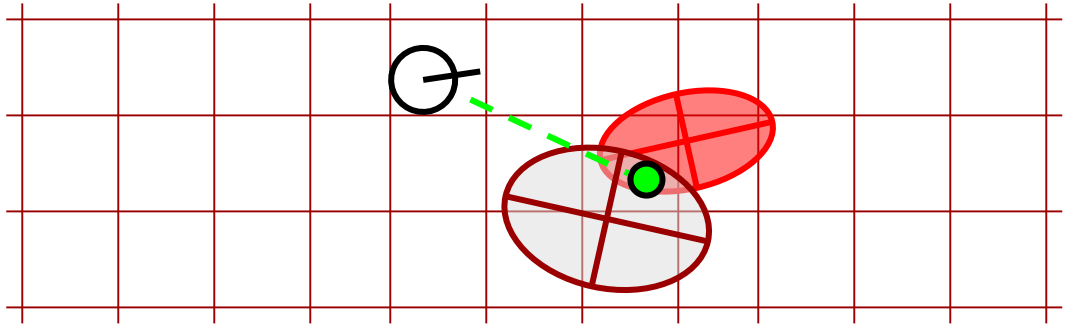
- A robust SLAM solution must consider possible data associations
- Potential data associations depend also on the pose of the robot

Multi-Hypothesis Data Association

- Data association is done on a per-particle basis
- Robot pose error is factored out of data association decisions



Per-Particle Data Association



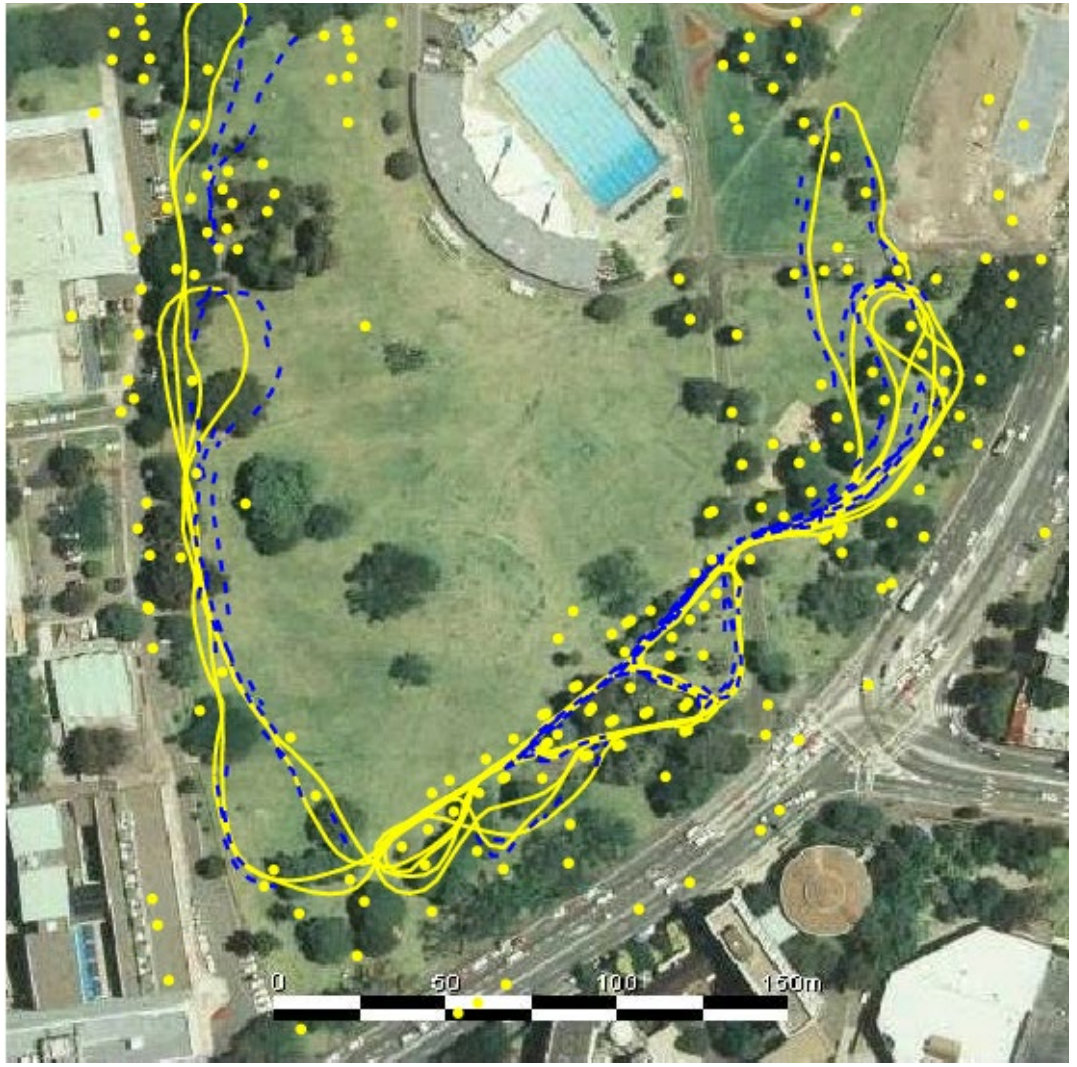
Was the observation generated by the red or the brown landmark?

P(observation|red) = 0.3 P(observation|brown) = 0.7

- Two options for per-particle data association
  - Pick the most likely match
  - Pick association randomly with probability proportional to the observation likelihoods
- If the data association probability is too low, generate a new landmark

Results – Victoria Park

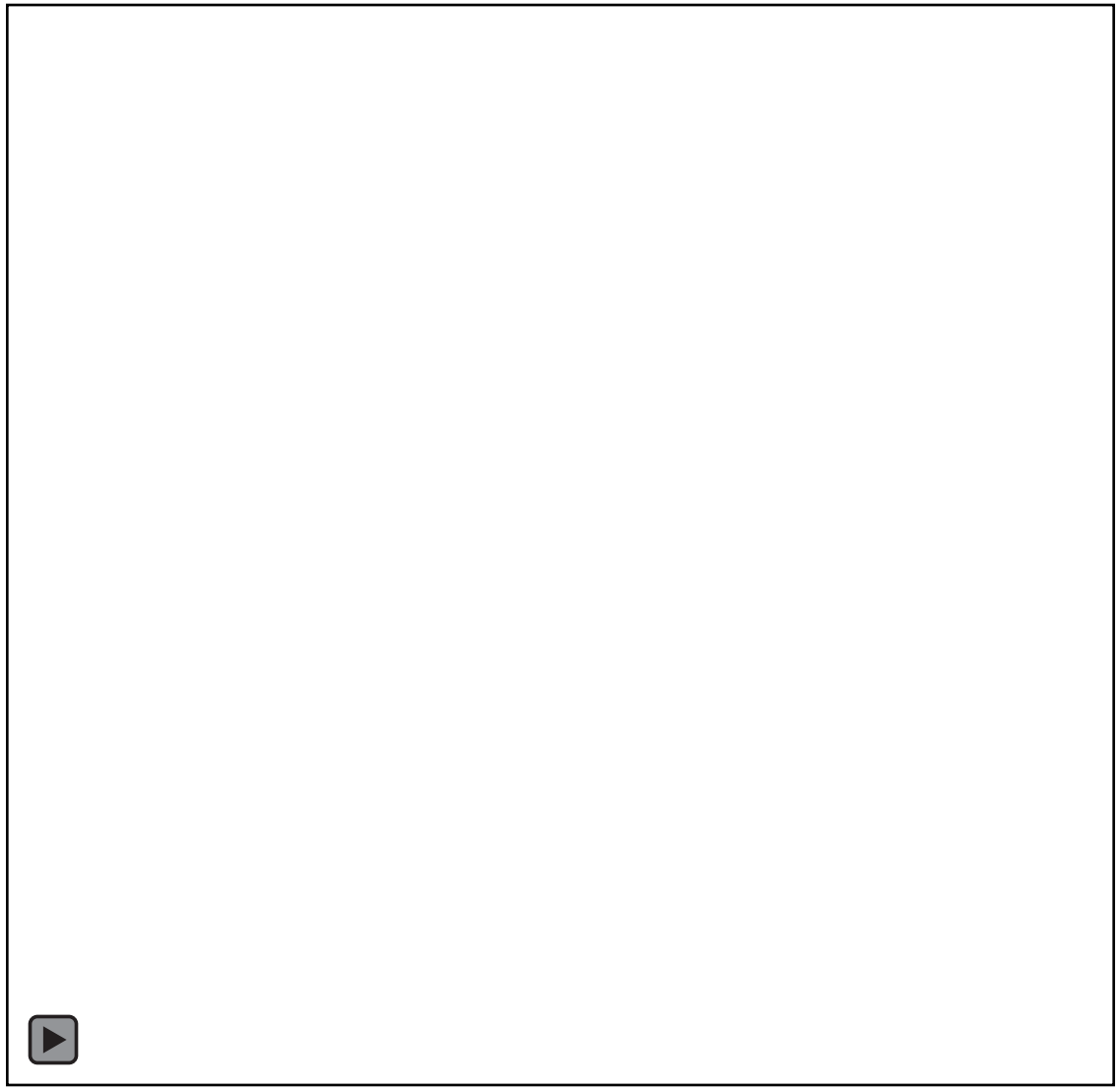
- 4 km traverse
- < 5 m RMS position error
- 100 particles



Blue = GPS
Yellow = FastSLAM

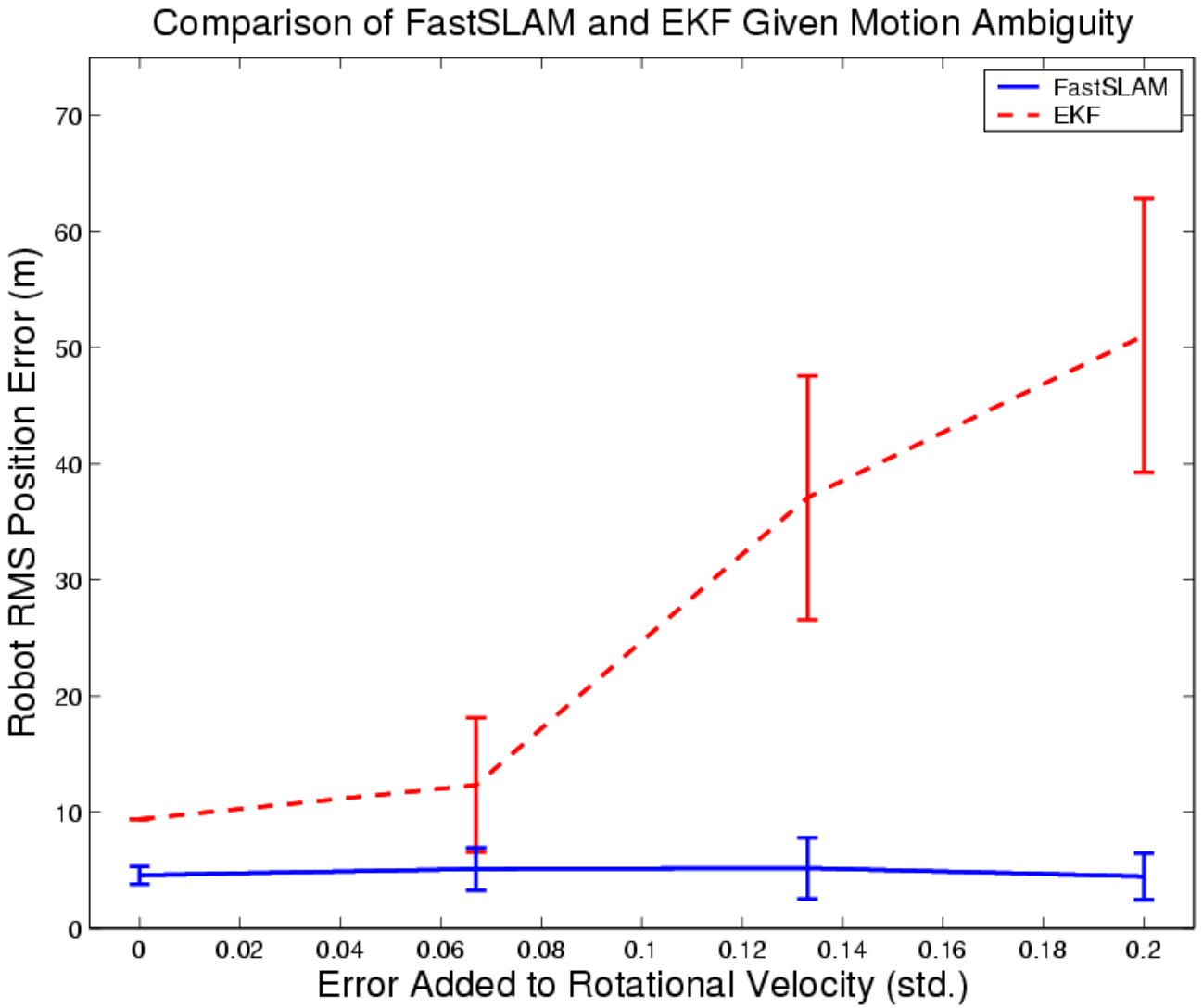
Dataset courtesy of University of Sydney

Results – Victoria Park (Video)



Dataset courtesy of University of Sydney

Results – Data Association

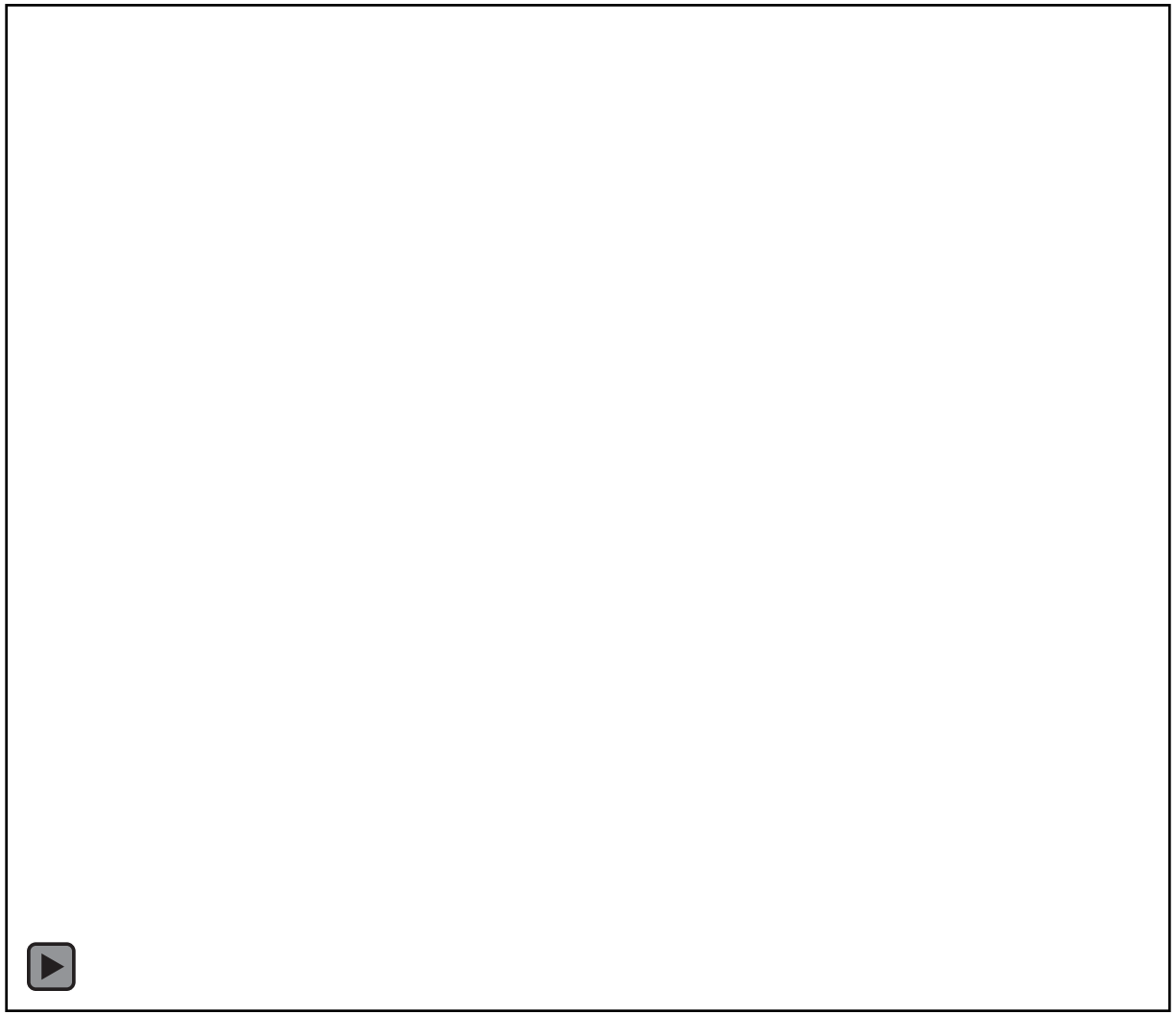


FastSLAM shows high robustness to motion errors

FastSLAM Summary

- FastSLAM factors the SLAM posterior into low-dimensional estimation problems
  - Scales to problems with over 1 million features
- FastSLAM factors robot pose uncertainty out of the data association problem
  - Robust to significant ambiguity in data association
  - Allows data association decisions to be delayed until unambiguous evidence is collected
- Advantages compared to the classical EKF approach (especially with non-linearities)
- Complexity *O(N log M)*

Grid-based SLAM using Raw Odometry



Grid-based SLAM

- Can we solve the SLAM problem if no pre-defined landmarks are available?
- Can we use the ideas of FastSLAM to build grid maps?
- As with landmarks, the map depends on the poses of the robot during data acquisition
- If the poses are known, grid-based mapping is easy ("mapping with known poses")

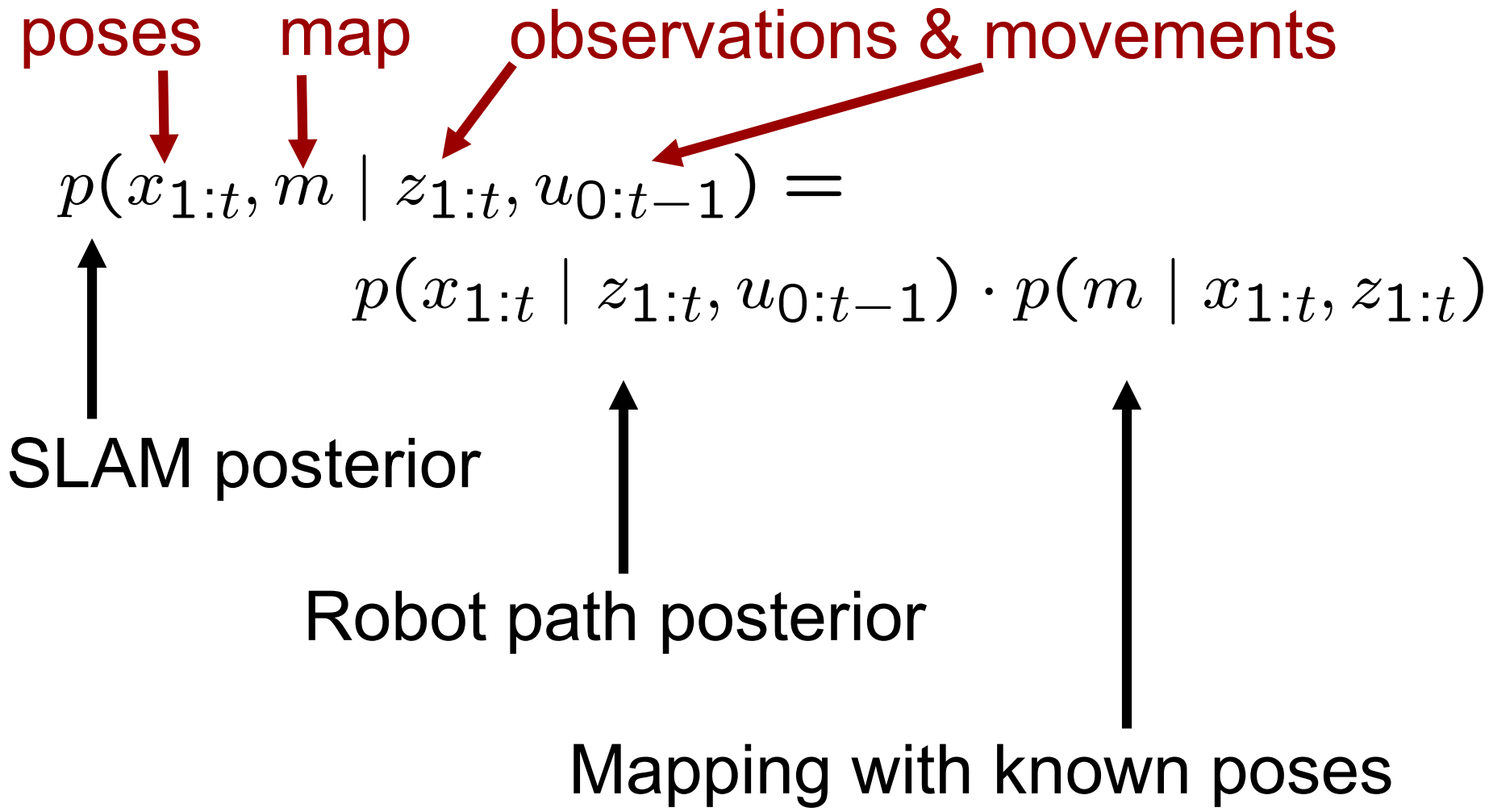
Rao-Blackwellization

$$p(x_{1:t}, m \mid z_{1:t}, u_{0:t-1}) =$$

$$p(x_{1:t} \mid z_{1:t}, u_{0:t-1}) \cdot p(m \mid x_{1:t}, z_{1:t})$$

Factorization first introduced by Murphy in 1999

Rao-Blackwellization



Factorization first introduced by Murphy in 1999

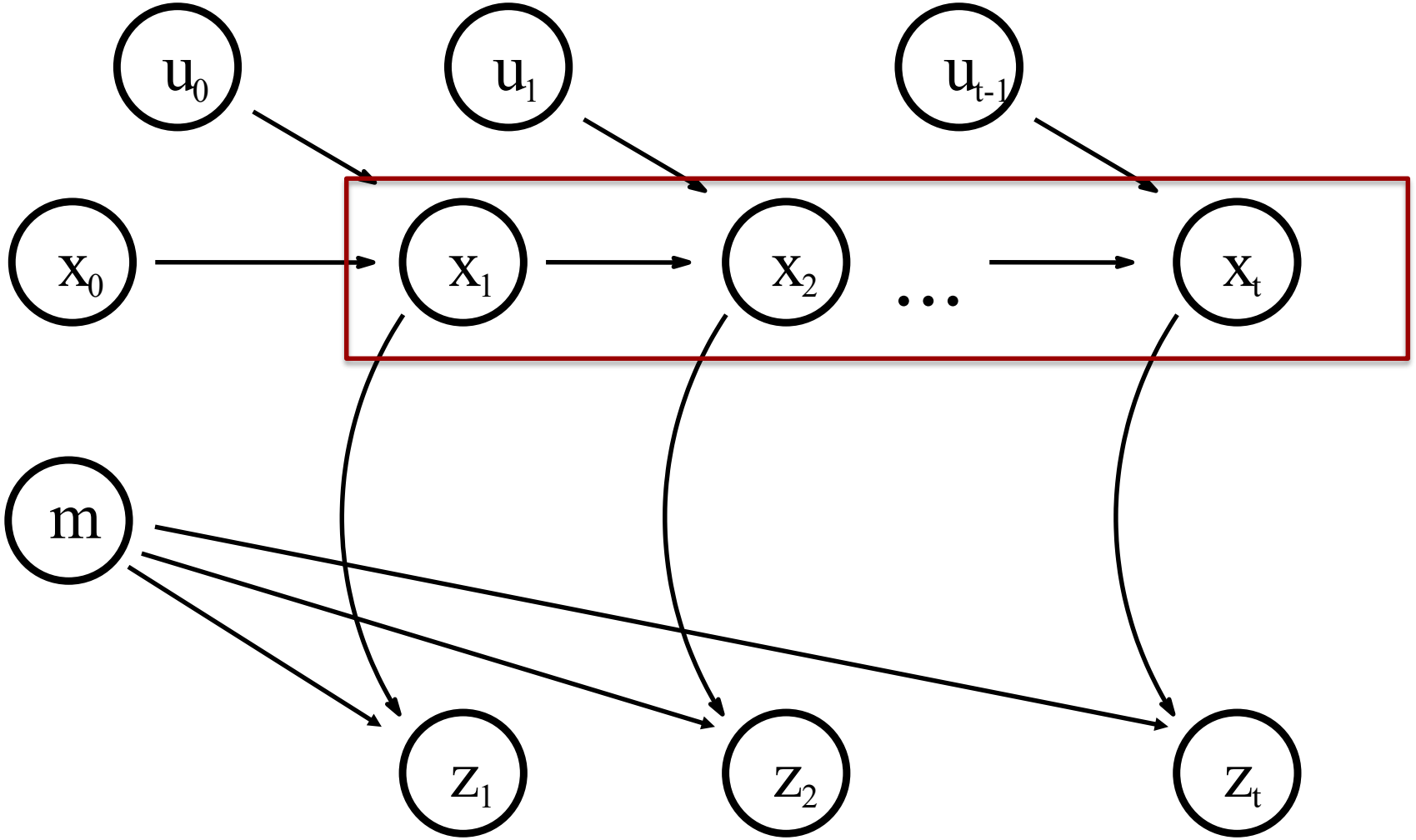
Rao-Blackwellization

$$p(x_{1:t}, m \mid z_{1:t}, u_{0:t-1}) =$$
$$p(x_{1:t} \mid z_{1:t}, u_{0:t-1}) \cdot p(m \mid x_{1:t}, z_{1:t})$$

This is localization, use MCL

Use the pose estimate from the MCL and apply mapping with known poses

A Graphical Model of Mapping with Rao-Blackwellized Particle Filters



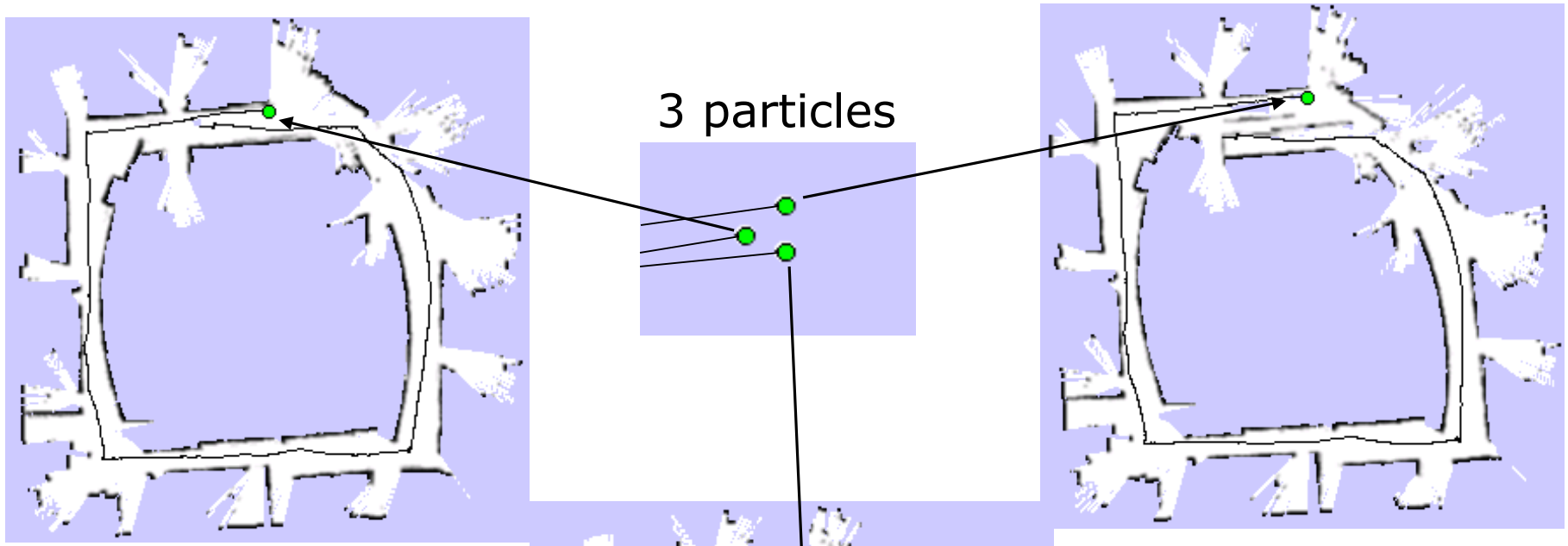
Mapping with Rao-Blackwellized Particle Filters

- Each particle represents a possible trajectory of the robot

- Each particle
  - maintains its own map and
  - updates it using “mapping with known poses”

- Each particle survives with a probability proportional to the likelihood of the observations relative to its own map

Particle Filter Example

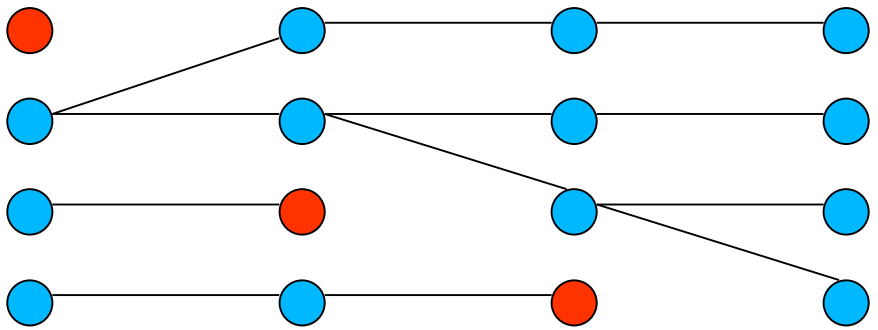


Problem

- Each map is quite big in case of grid maps
- Each particle maintains its own map, therefore, one needs to keep the number of particles small
- **Solution**:
  Compute better proposal distributions!
- **Idea**:
  Improve the pose estimate **before** applying the particle filter
- See additional literature!

Problem: Re-sampling

- However, resampling at each step limits the “memory” of our filter
- Supposed we lose 25% of the particles at each frame , in the worst case we have a memory of only 4 steps.



Goal: reduce the number of resampling actions

Selective Re-sampling

- Re-sampling is dangerous, since important samples might get lost (particle depletion problem)
- In case of suboptimal proposal distributions re-sampling is necessary to achieve convergence.
- Key question: When should we re-sample?

Number of Effective Particles

$$n_{eff} = \frac{1}{\sum_i \left(w_t^{(i)}\right)^2}$$

- Assuming normalized particle weights that sum up to 1.0:

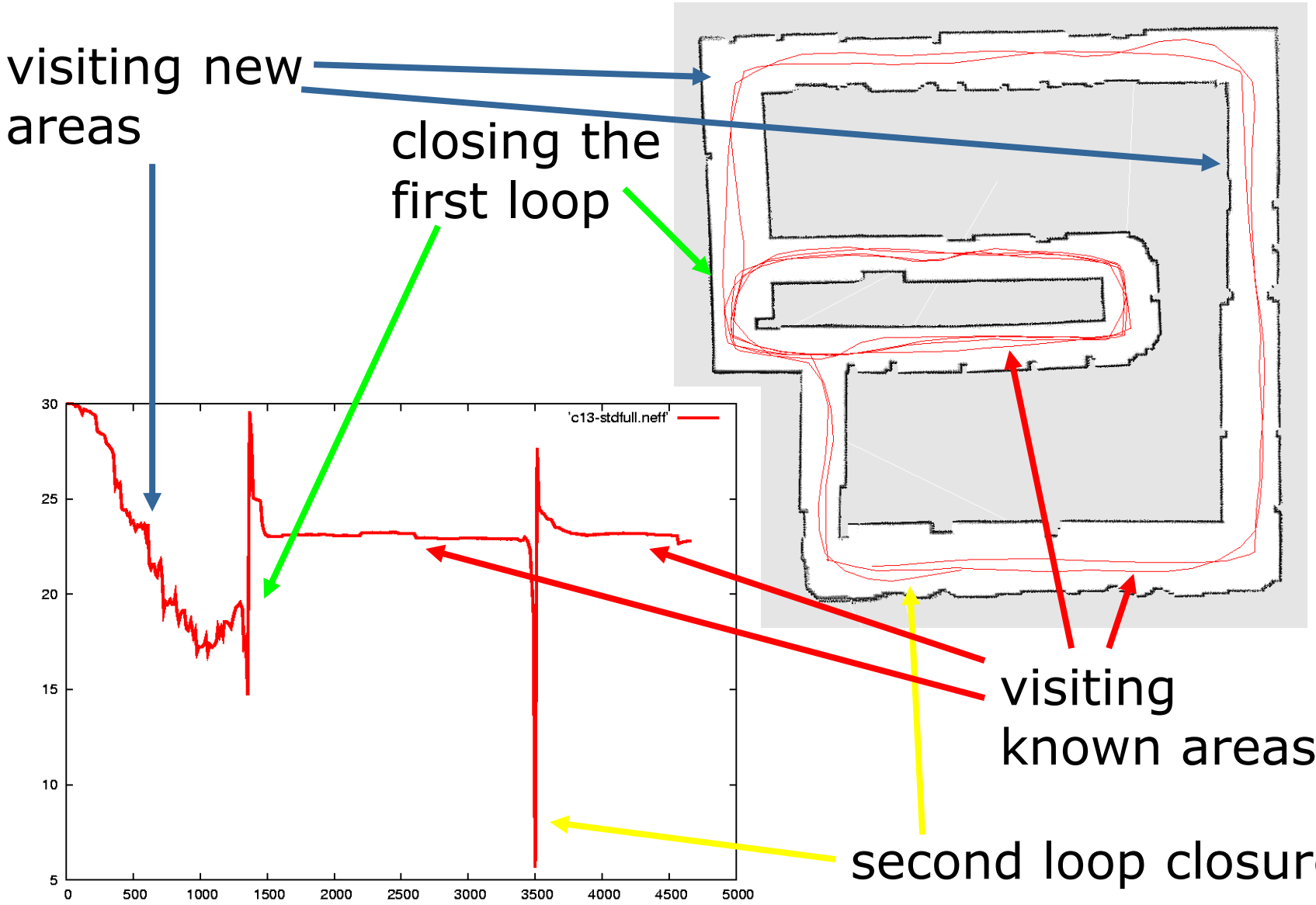
$$\sum_{i=1}^{n} w_t^{(i)} = 1 \;\Rightarrow\; n_{eff} \in [1, n]$$

- Empirical measure of how well the goal distribution is approximated by samples drawn from the proposal
- It describes "the variance of the particle weights"
- It is maximal for equal weights. In this case the distribution is close to the proposal

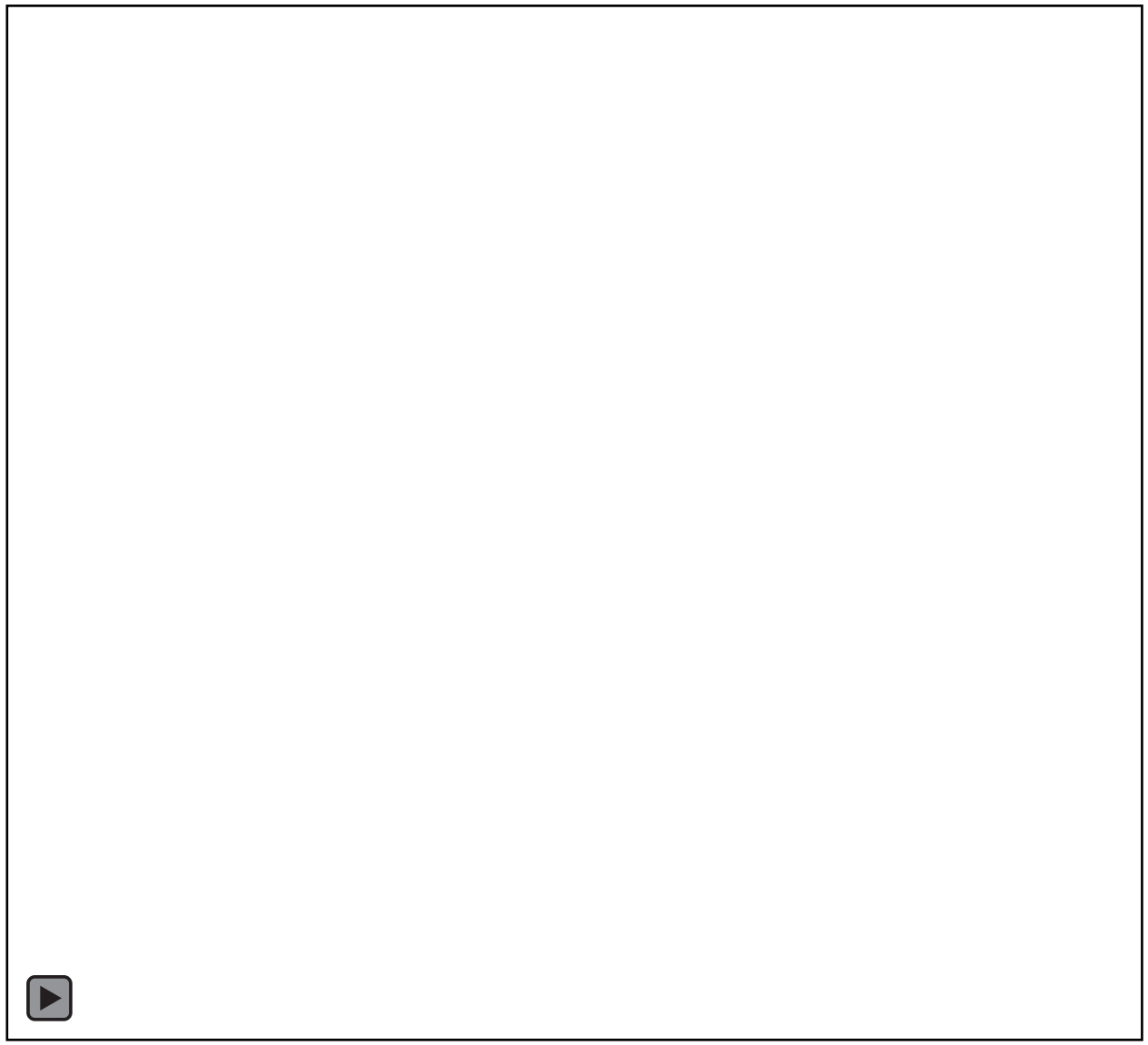
Resampling with n_{eff}

- If our approximation is close to the proposal, no resampling is needed
- We only re-sample when n_{eff} drops below a given threshold, typically $\frac{n}{2}$
- See [Doucet, '98; Arulampalam, '01]

Typical Evolution of n_{eff}



SLAM with Rao-Blackwellized Particle Filters



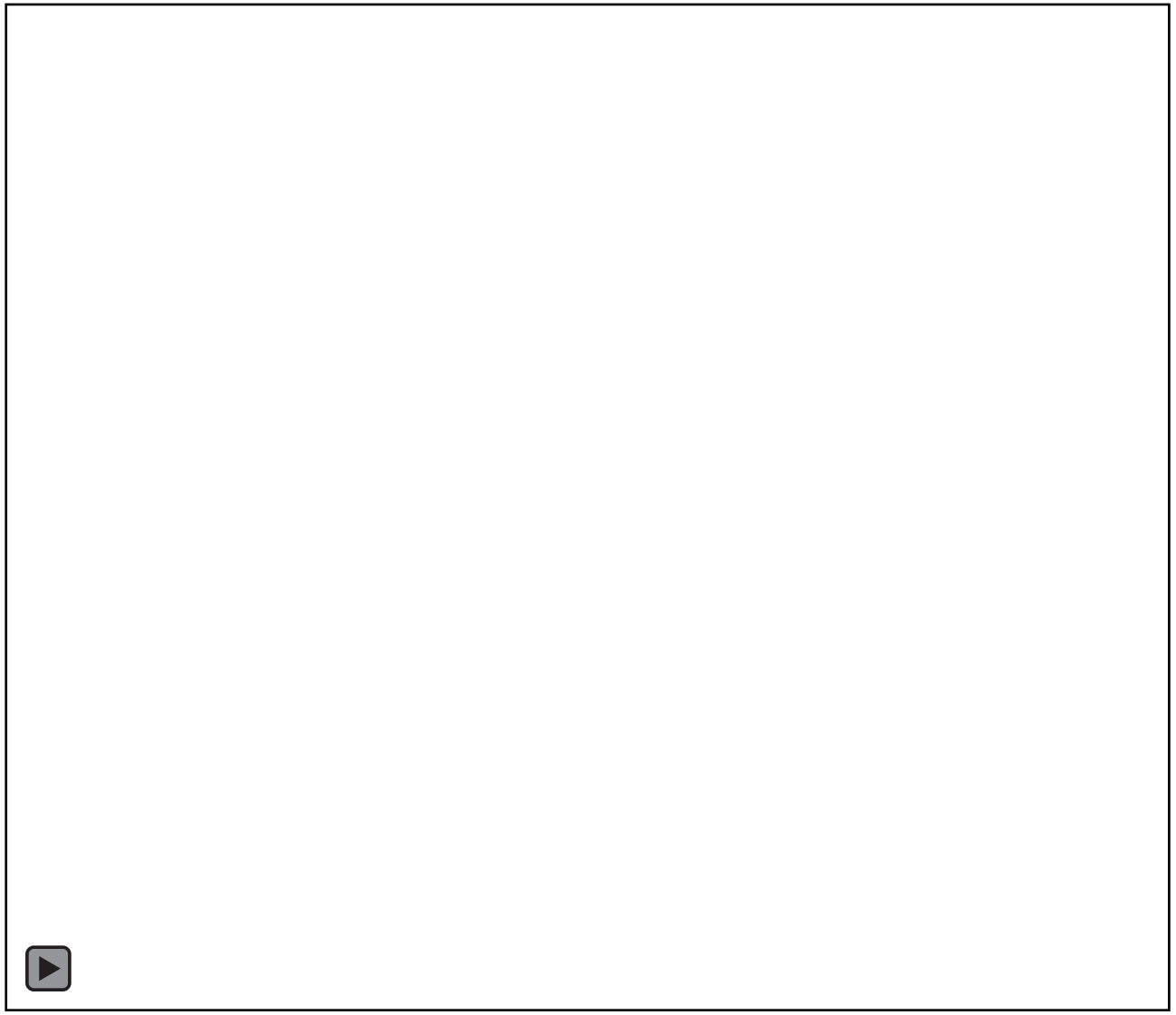
Intel Lab

- **15 particles**
- four times faster than real-time P4, 2.8GHz
- 5cm resolution during scan matching
- 1cm resolution in final map

Outdoor Campus Map

- **30 particles**
- 250x250m^2
- 1.088 miles (odometry)
- 20cm resolution during scan matching
- 30cm resolution in final map

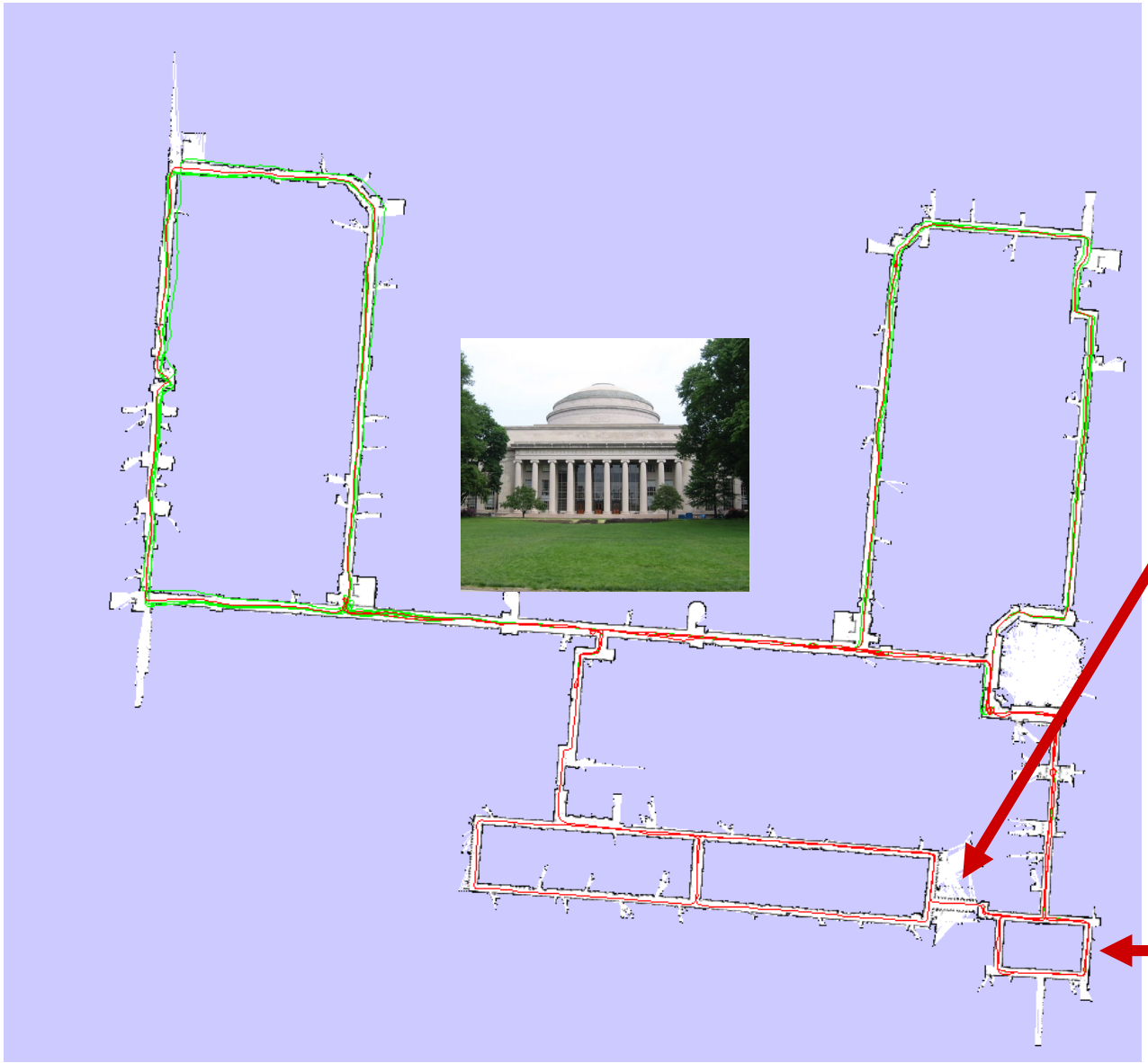
Outdoor Campus Map - Video

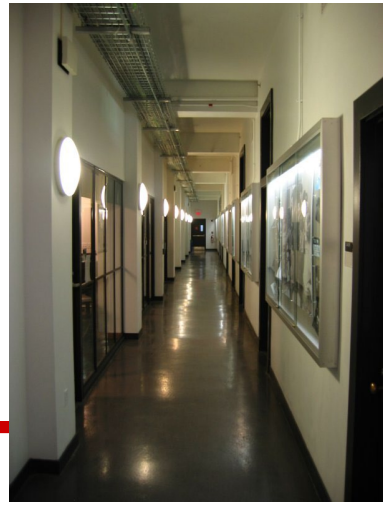


MIT Killian Court

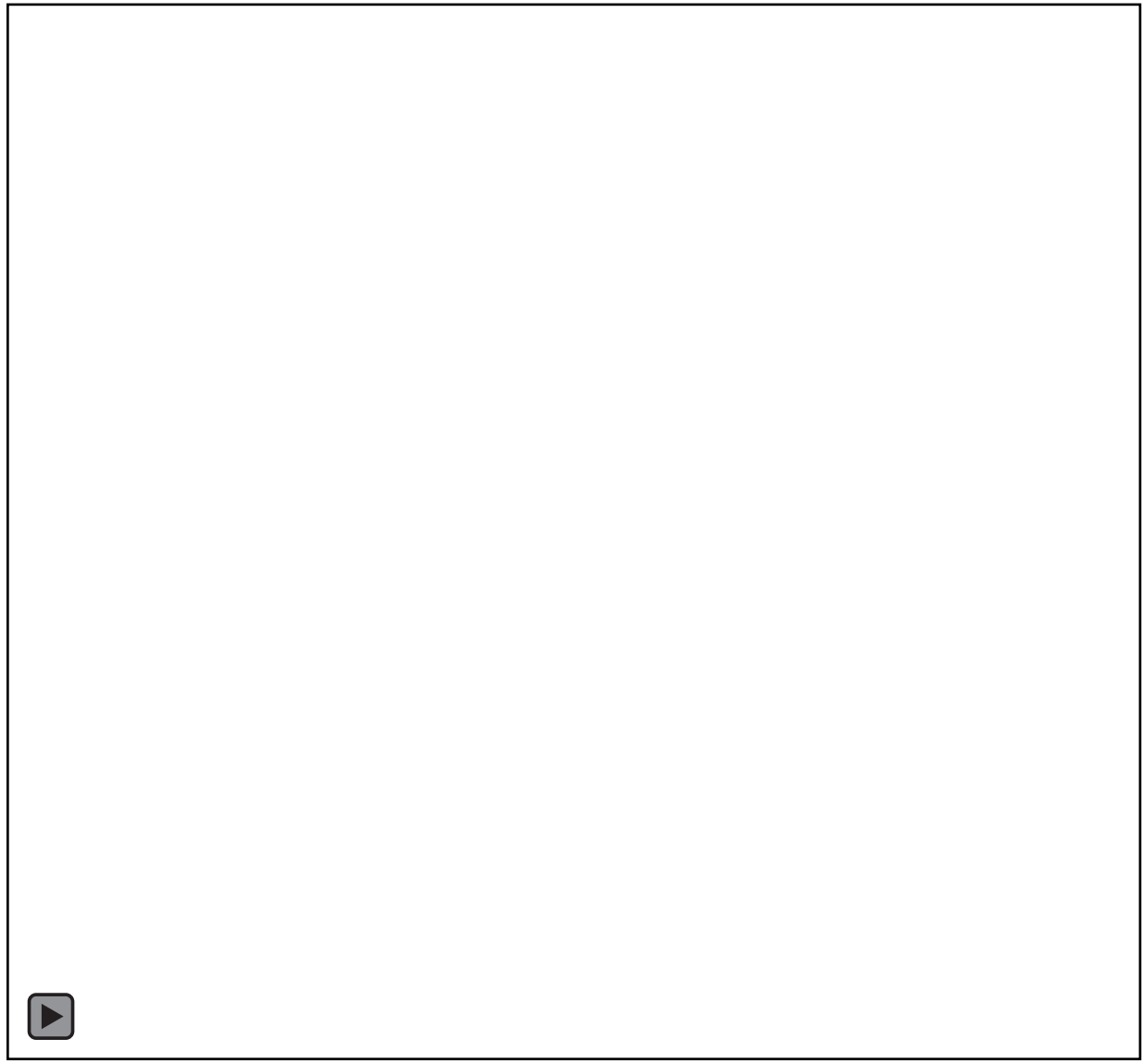
- The **“infinite-corridor-dataset”** at MIT

MIT Killian Court





Mapping the MIT Killian Court



FastSLAM with Grid Maps: Conclusions

- The ideas of FastSLAM can also be applied in the context of grid maps
- Utilizing accurate sensor observation leads to good proposals and highly efficient filters (not covered)
- It is similar to scan-matching on a per-particle base (not covered)
- The number of necessary particles and re-sampling steps can seriously be reduced
- Improved versions of grid-based FastSLAM can handle large environments in “real time”

Conclusions

- SLAM is one of the fundamental problems in robot navigation
- It entails estimating the map and the robot pose at the very same time
- The Extended Kalman Filter can effectively solve the SLAM problem for landmark-based representations
- The Particle Filter can be applied to SLAM problems with landmark-based and dense grid representations

More Details on FastSLAM

- M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit. FastSLAM: A factored solution to simultaneous localization and mapping, *AAAI02* *(The classic FastSLAM paper with landmarks)*

- D. Haehnel, W. Burgard, D. Fox, and S. Thrun. An efficient FastSLAM algorithm for generating maps of large-scale cyclic environments from raw laser range measurements, IROS03 *(FastSLAM on grid-maps using scan-matched input)*

- G. Grisetti, C. Stachniss, and W. Burgard. Improving grid-based SLAM with Rao-Blackwellized particle filters by adaptive proposals and selective resampling, ICRA05 *(Proposal using laser observation, adaptive resampling)*

- A. Eliazar and R. Parr. DP-SLAM: Fast, robust simultaneous localization and mapping without predetermined landmarks, IJCAI03 *(An approach to handle big particle sets and maps)*