

Cognitive Robotics

09. FastSLAM

AbdElMoniem Bayoumi, PhD

Recap: Drawbacks of EKF SLAM

- Can only deal with a single mode
- Successful only in medium-scale scenes
- Computationally intractable for large maps

- **Idea:** Apply the particle filter principle to solve both problems simultaneously, localization and mapping

Particle Representation

- A set of weighted samples

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

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

$$x = \left(\underbrace{x_{1:t}}_{\text{poses}}, \underbrace{m_{1,x}, m_{1,y}, \dots, m_{M,x}, m_{M,y}}_{\text{landmarks}} \right)^T$$

Dimensionality Problem

- PF are effective in low-dimensional spaces
- The number of particles needed to represent a posterior grows exponentially with the dimension of the state space!

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

high-dimensional!

Dependencies

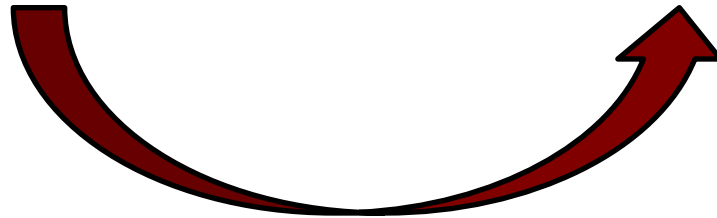
- Are there dependencies between the dimensions of the state space?
- Yes: The map depends on the robot poses from which measurements were obtained
- Use this dependency to solve the state estimation problem more efficiently

Exploit Dependencies Between the Different Dimensions of the State Space

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

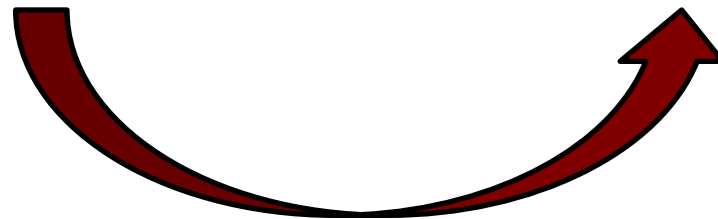
If We Know the Robot Poses, Mapping is Easy!

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



Key Idea

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



- Use the particle set only to model the robot's path
- For each sample, compute an individual map of landmarks

Rao-Blackwellization

- Use factorization to exploit dependencies between variables:


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

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

Rao-Blackwellization for SLAM

Factorization of the SLAM posterior

poses map observations & movements


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

Rao-Blackwellization for SLAM

Factorization of the SLAM posterior

poses map observations & movements

↓ ↓ ↙ ↘

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

↑ ↑

path posterior map posterior

The diagram illustrates the factorization of the SLAM posterior. At the top, three labels in red: 'poses', 'map', and 'observations & movements'. Red arrows point from 'poses' and 'map' down to the variables $x_{0:t}$ and $m_{1:M}$ in the joint probability expression. A red arrow points from 'observations & movements' to the conditioning variables $z_{1:t}, u_{1:t}$. The joint probability is shown as $p(x_{0:t}, m_{1:M} \mid z_{1:t}, u_{1:t}) =$ followed by the product of two terms: $p(x_{0:t} \mid z_{1:t}, u_{1:t})$ and $p(m_{1:M} \mid x_{0:t}, z_{1:t})$. Below these terms, two more red labels in red: 'path posterior' and 'map posterior'. Red arrows point from 'path posterior' up to the first term and from 'map posterior' up to the second term.

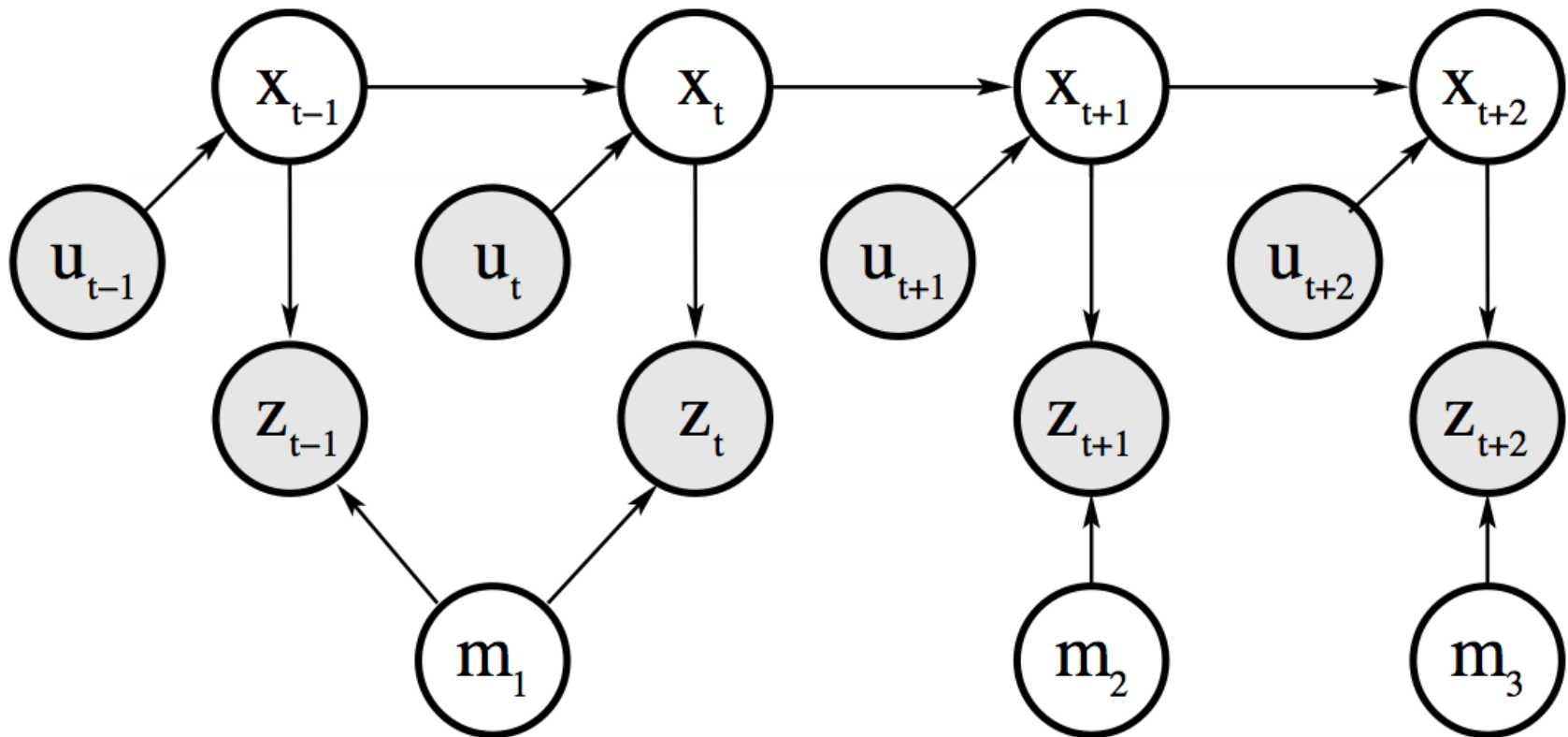
Rao-Blackwellization for SLAM

Factorization of the SLAM posterior

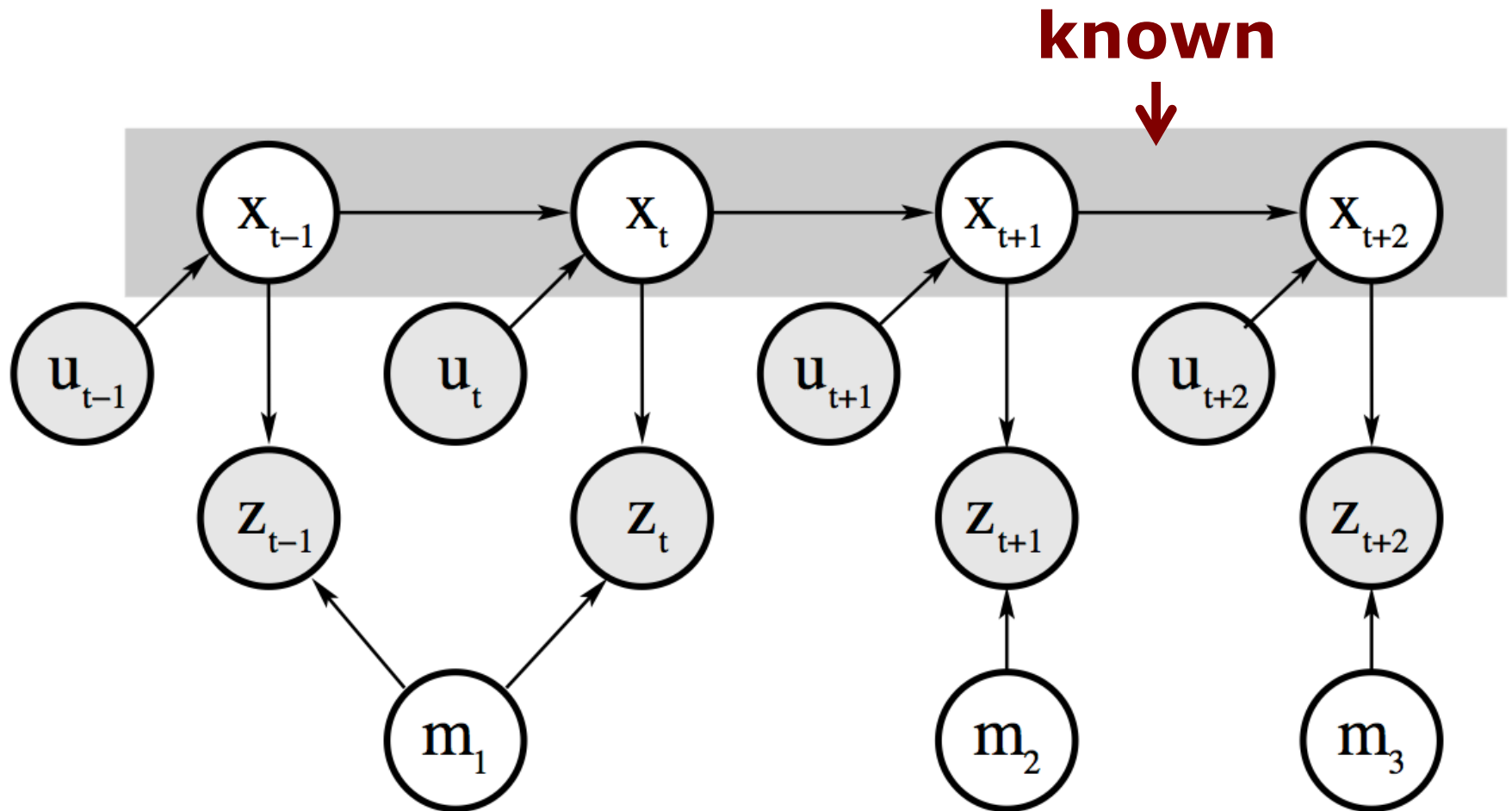
$$p(x_{0:t}, m_{1:M} \mid z_{1:t}, u_{1:t}) = p(x_{0:t} \mid z_{1:t}, u_{1:t}) \underbrace{p(m_{1:M} \mid x_{0:t}, z_{1:t})}$$

How to compute this term efficiently?

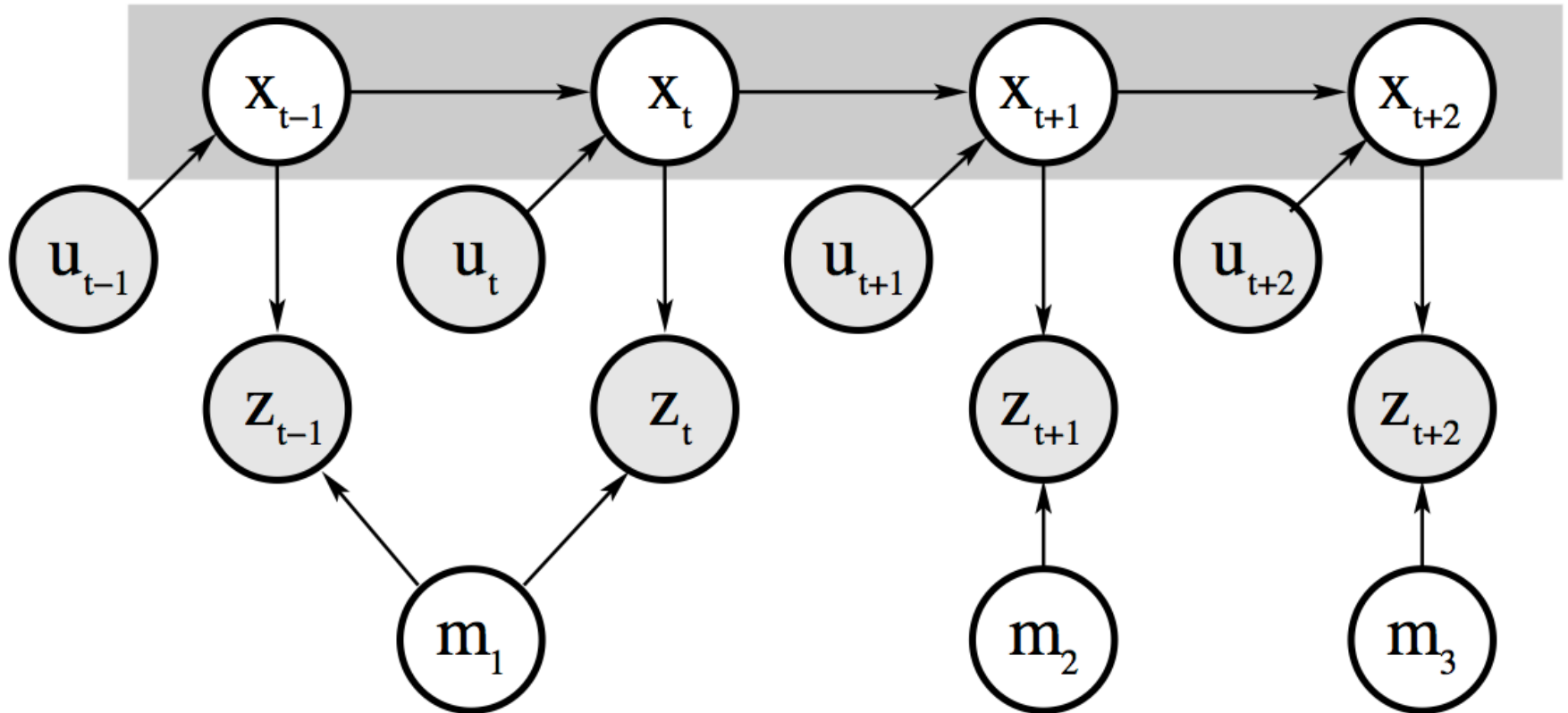
Revisit the Graphical Model



Revisit the Graphical Model



Landmarks are Conditionally Independent Given the Poses



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

Rao-Blackwellization for SLAM

Factorization of the SLAM posterior

$$p(x_{0:t}, m_{1:M} \mid z_{1:t}, u_{1:t}) = p(x_{0:t} \mid z_{1:t}, u_{1:t}) \underbrace{p(m_{1:M} \mid x_{0:t}, z_{1:t})}$$



Landmarks are conditionally independent given the poses

Rao-Blackwellization for SLAM

Factorization of the SLAM posterior

$$\begin{aligned} p(x_{0:t}, m_{1:M} \mid z_{1:t}, u_{1:t}) = \\ p(x_{0:t} \mid z_{1:t}, u_{1:t}) p(m_{1:M} \mid x_{0:t}, z_{1:t}) \\ p(x_{0:t} \mid z_{1:t}, u_{1:t}) \prod_{i=1}^M p(m_i \mid x_{0:t}, z_{1:t}) \end{aligned}$$

Rao-Blackwellization for SLAM

Factorization of the SLAM posterior

$$p(x_{0:t}, m_{1:M} \mid z_{1:t}, u_{1:t}) =$$
$$p(x_{0:t} \mid z_{1:t}, u_{1:t}) p(m_{1:M} \mid x_{0:t}, z_{1:t})$$
$$p(x_{0:t} \mid z_{1:t}, u_{1:t}) \prod_{i=1}^M \underbrace{p(m_i \mid x_{0:t}, z_{1:t})}$$

2-dimensional EKFs!

Rao-Blackwellization for SLAM

Factorization of the SLAM posterior

$$p(x_{0:t}, m_{1:M} \mid z_{1:t}, u_{1:t}) =$$
$$p(x_{0:t} \mid z_{1:t}, u_{1:t}) p(m_{1:M} \mid x_{0:t}, z_{1:t})$$
$$\frac{p(x_{0:t} \mid z_{1:t}, u_{1:t})}{\text{particle filter similar to MCL}} \prod_{i=1}^M \frac{p(m_i \mid x_{0:t}, z_{1:t})}{\text{2-dimensional EKF!}}$$

particle filter similar to MCL

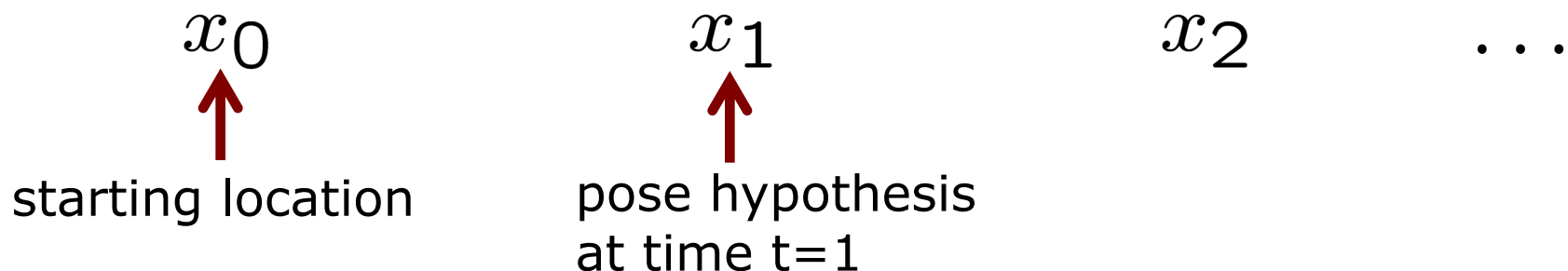
2-dimensional EKFs!

Modeling the Robot's Path

- Sample-based representation for

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

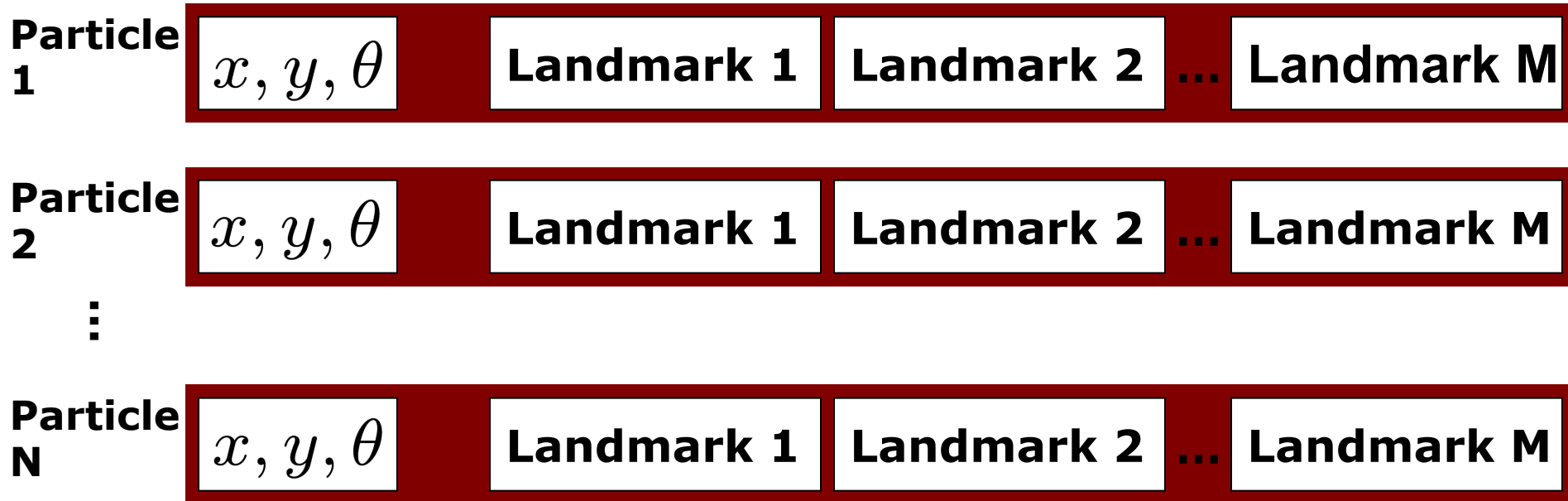
- Each sample represents a path hypothesis



- But: Past poses of a sample are not revised
- Thus, no need to maintain past poses in the sample set

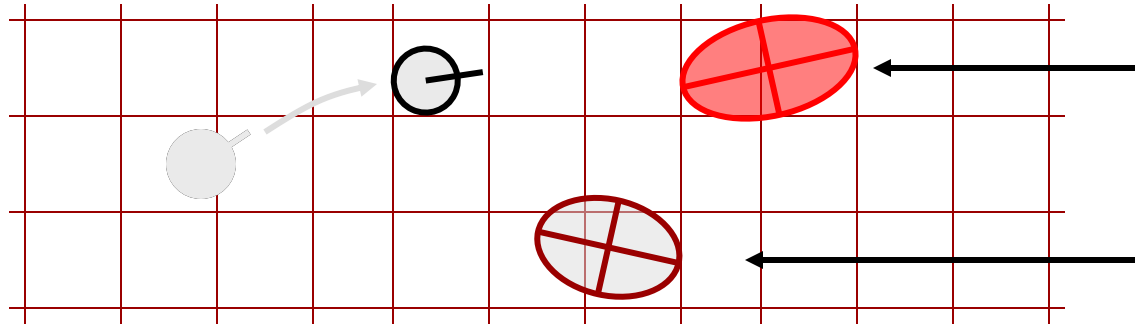
FastSLAM

- Each landmark is represented by a 2x2 EKF
- Thus, each particle has to maintain M individual EKFs



FastSLAM – Motion Update

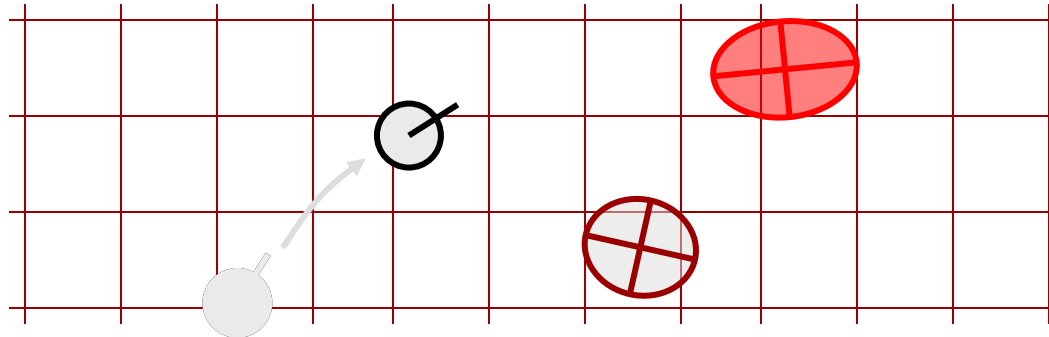
Particle #1



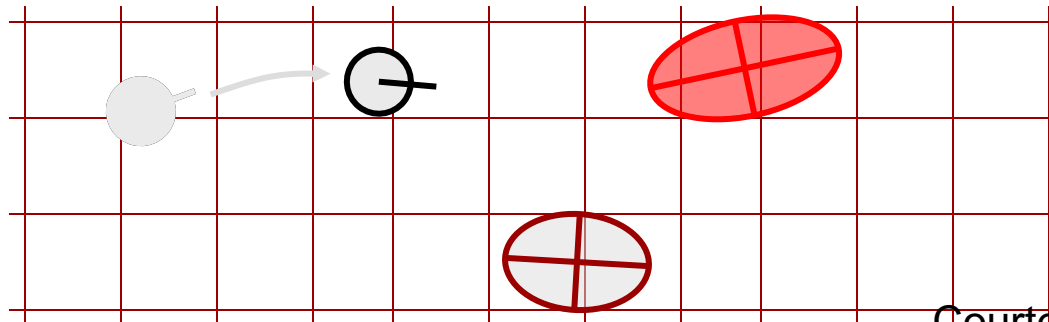
Landmark 1
2x2 EKF

Landmark 2
2x2 EKF

Particle #2

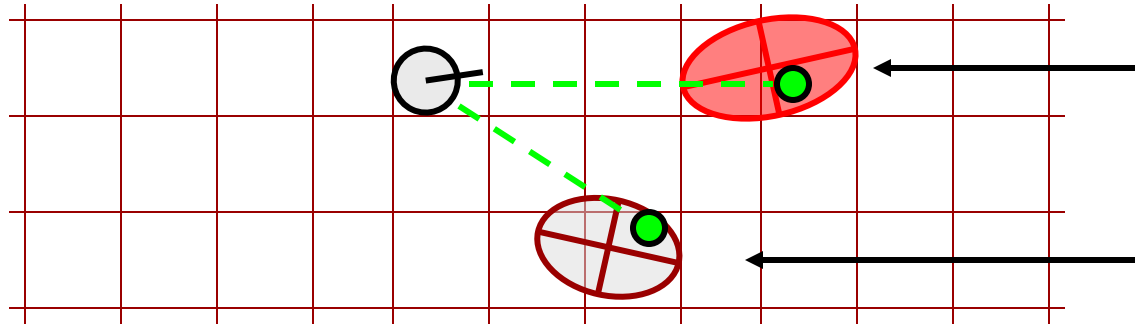


Particle #3



FastSLAM – Sensor Update

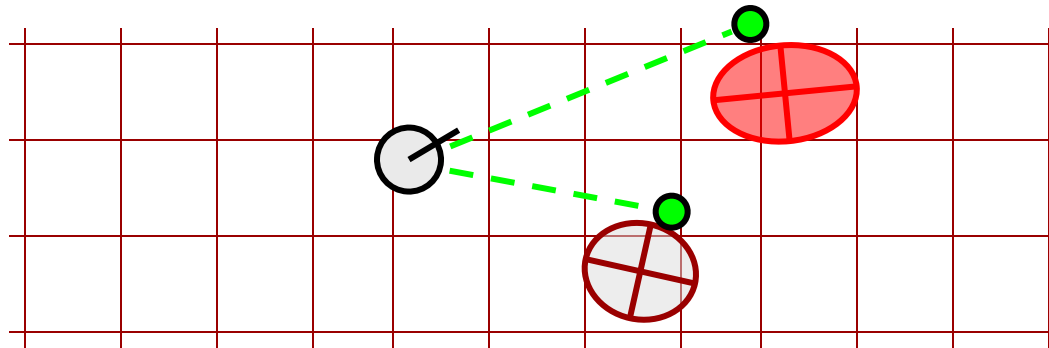
Particle #1



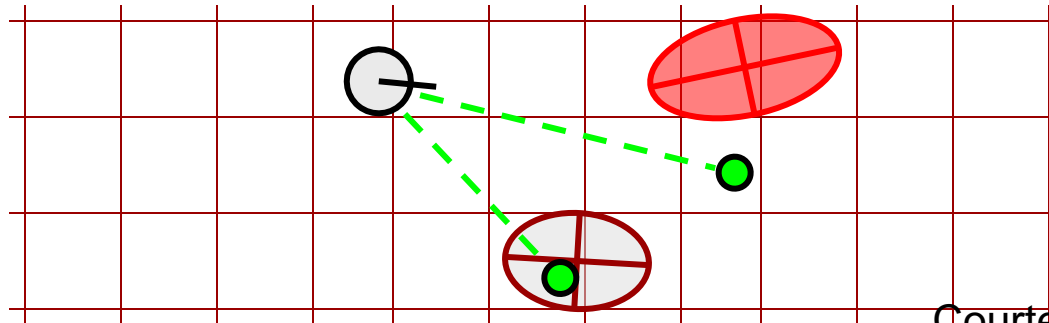
Landmark 1
2x2 EKF

Landmark 2
2x2 EKF

Particle #2

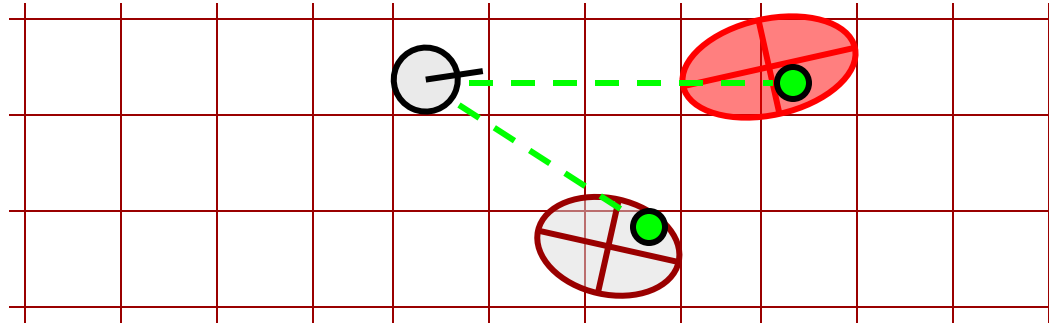


Particle #3



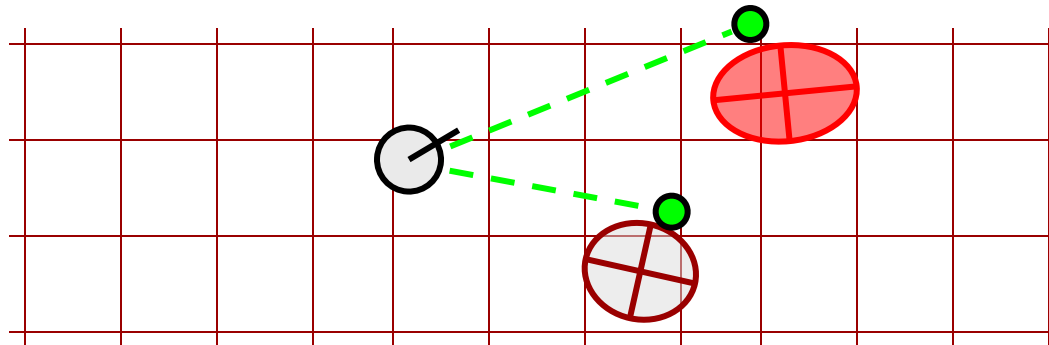
FastSLAM – Sensor Update

Particle #1



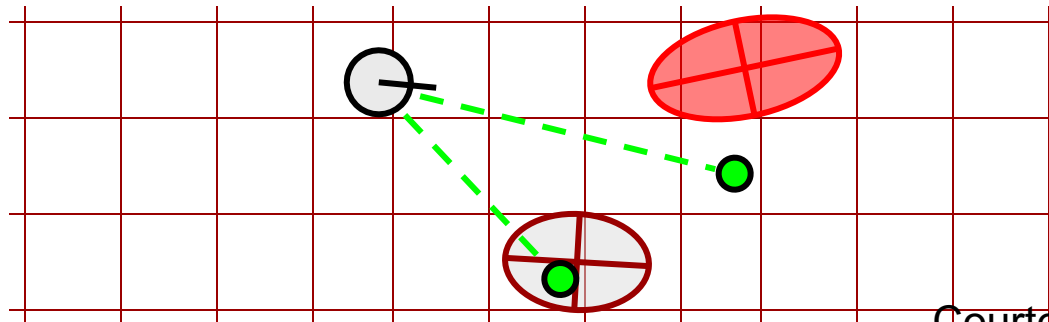
Weight = 0.8

Particle #2



Weight = 0.4

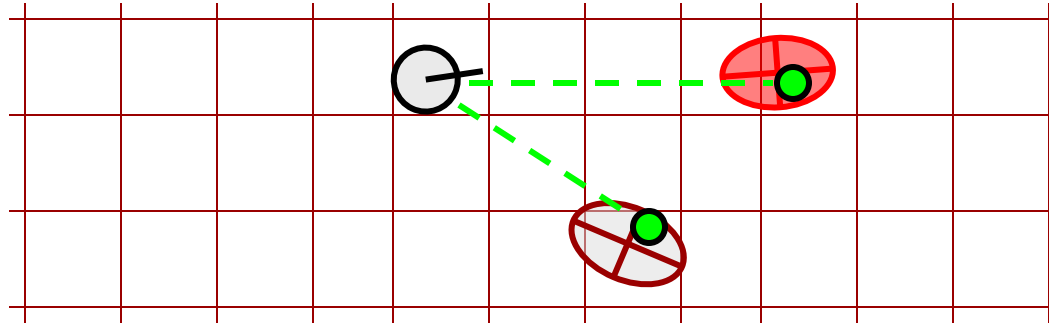
Particle #3



Weight = 0.1

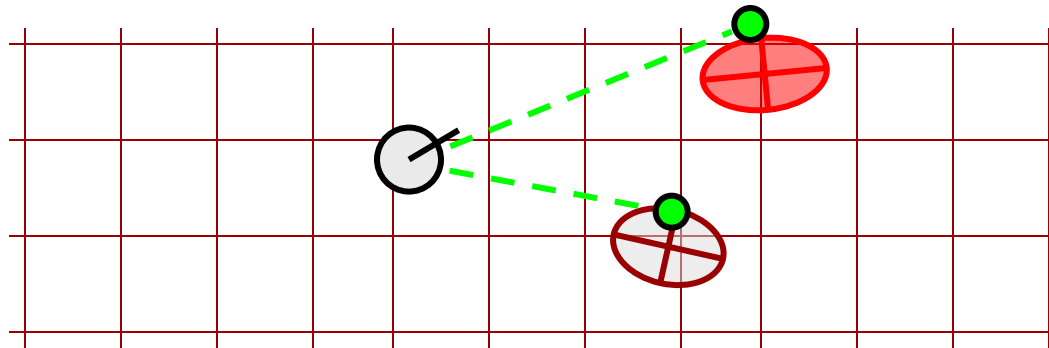
FastSLAM – Sensor Update

Particle #1



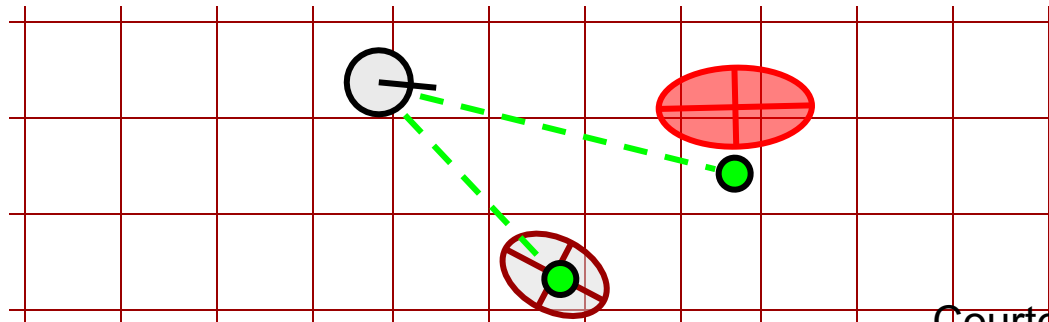
Update map
of particle 1

Particle #2



Update map
of particle 2

Particle #3



Update map
of particle 3

Key Steps of FastSLAM 1.0

- Sample a new pose for each particle

$$x_t^{[k]} \sim p(x_t \mid x_{t-1}^{[k]}, u_t)$$

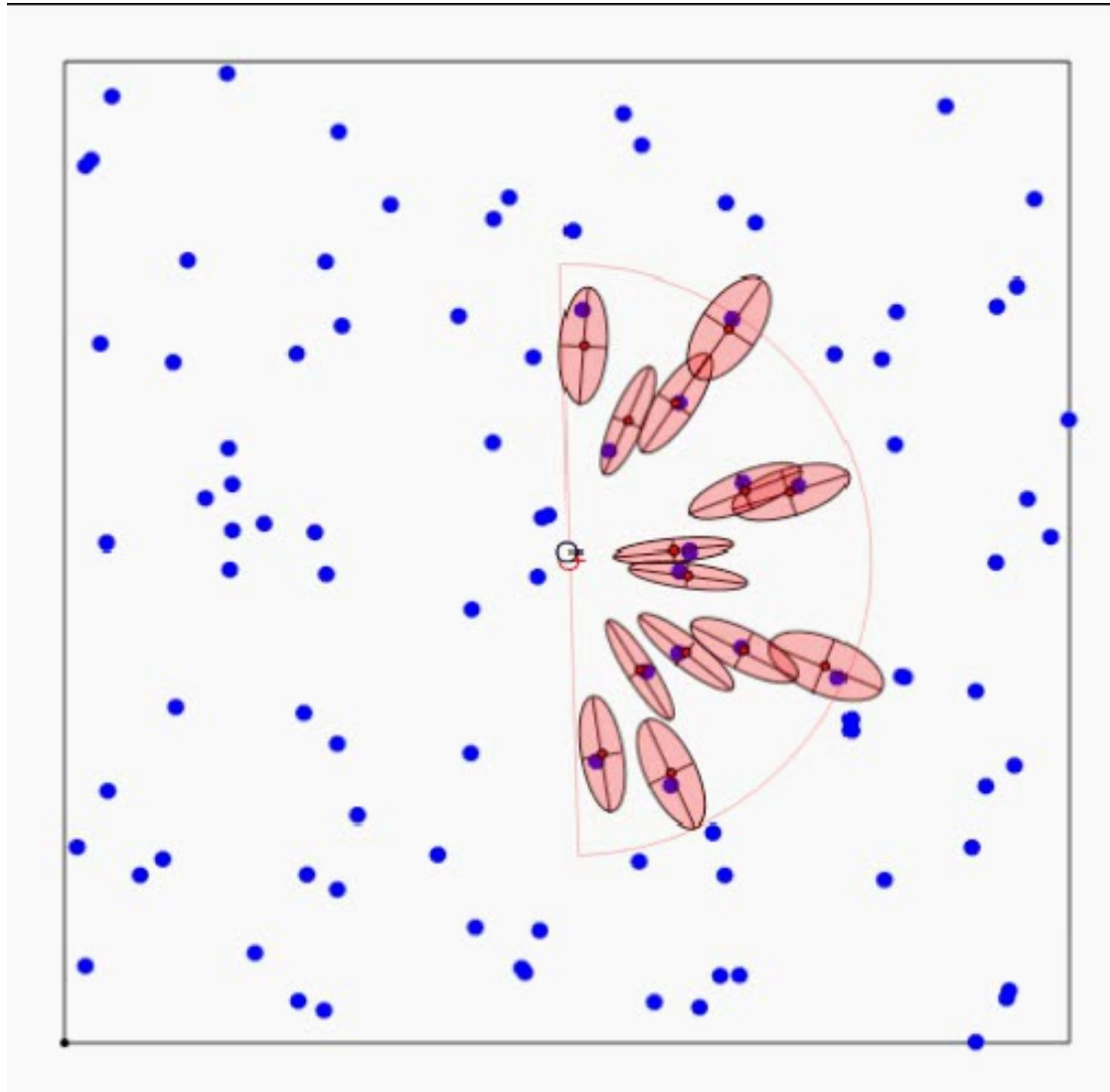
- Compute the particle weights **exp. observation**

$$w^{[k]} = |2\pi Q|^{-\frac{1}{2}} \exp \left\{ -\frac{1}{2} (z_t - \hat{z}^{[k]})^T Q^{-1} (z_t - \hat{z}^{[k]}) \right\}$$

measurement covariance

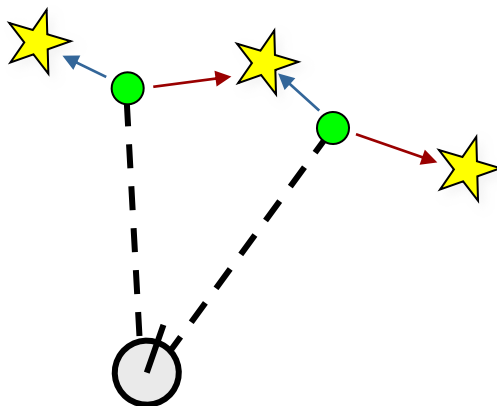
- Update belief of observed landmarks (EKF update rule)
- Resample

FastSLAM in Action



Data Association Problem

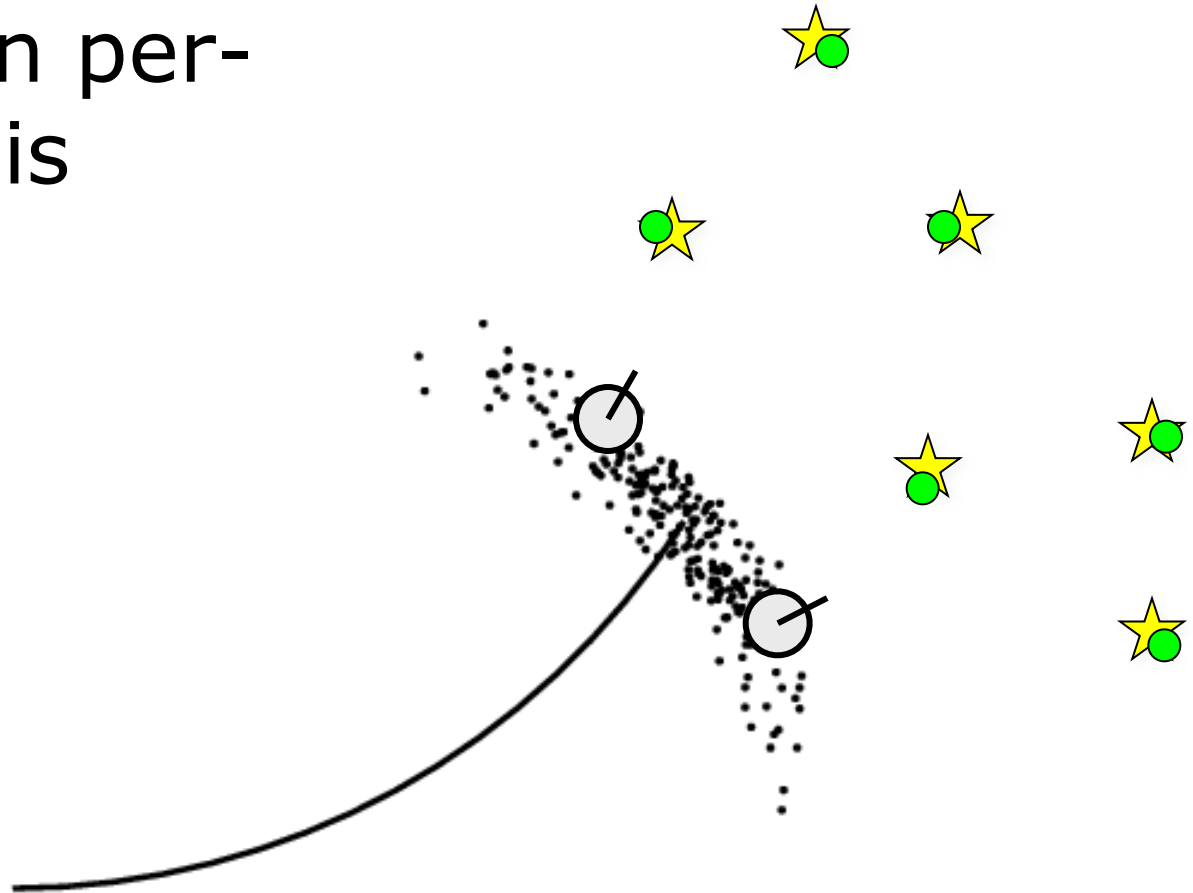
- Which observation belongs to which landmark?



- More than one possible association
- **Potential data associations depend on the pose of the robot**

Particles Support Multi-Hypotheses Data Association

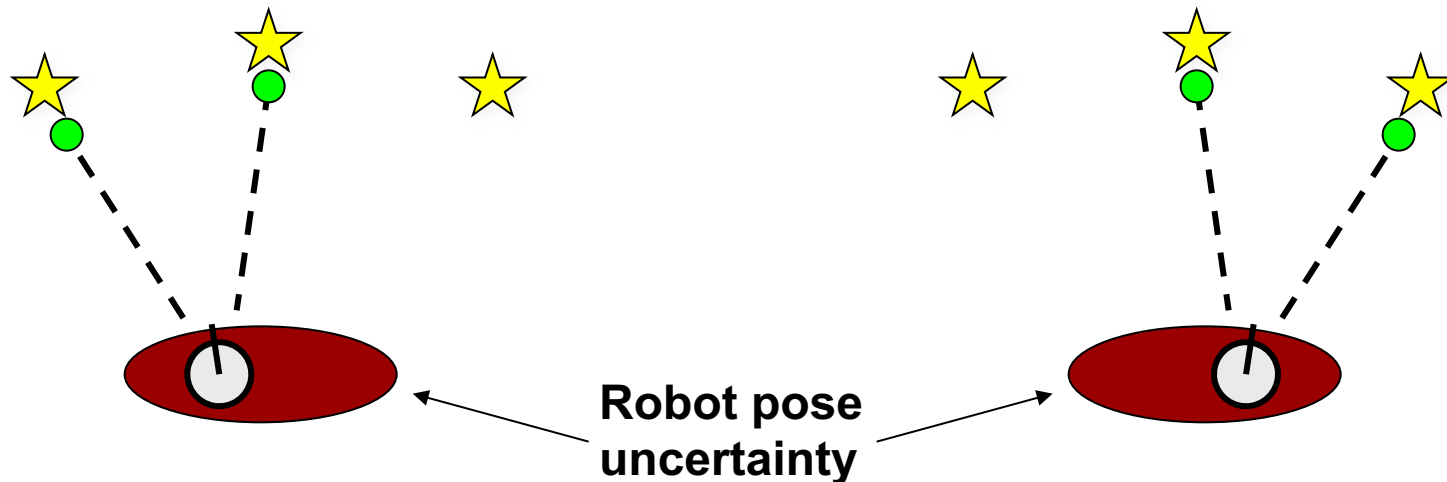
Decisions on per-particle basis



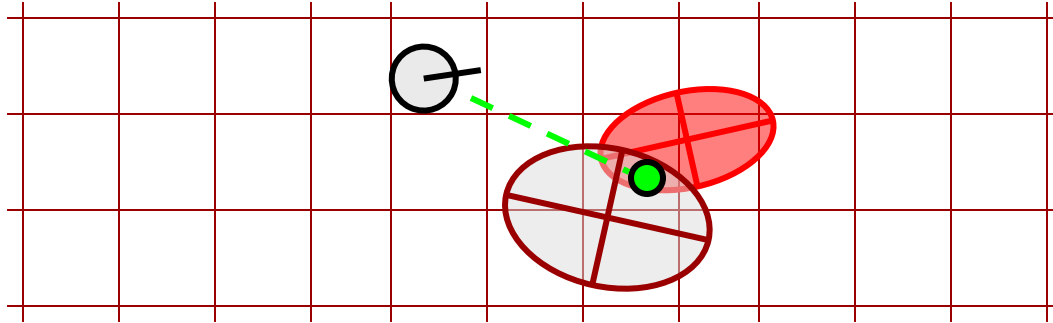
Recap:

Data Association in EKF SLAM

The best data association is not obvious even within the error ellipse of the pose estimate



Per-Particle Data Association

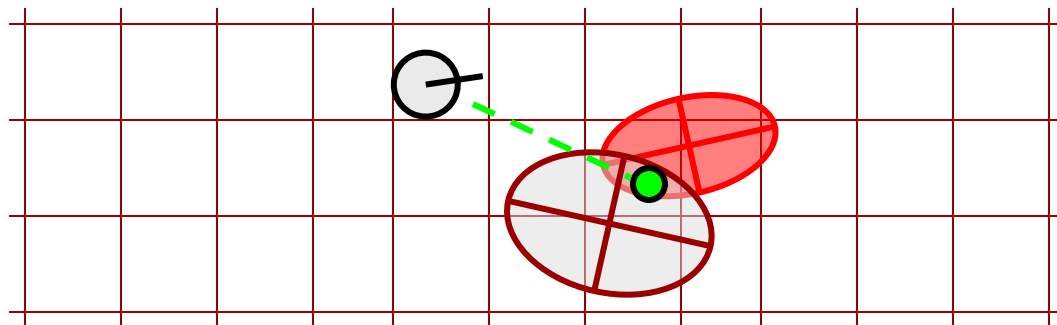


Is the observation generated by the **red** or by the **brown** landmark?

$$P(\text{observation}|\text{red}) = 0.3$$

$$P(\text{observation}|\text{brown}) = 0.7$$

Per-Particle Data Association



Is the observation generated by the **red** or by the **brown** landmark?

$$P(\text{observation}|\text{red}) = 0.3$$

$$P(\text{observation}|\text{brown}) = 0.7$$

- Options:
 - Pick the most probable match
 - Pick a random association weighted by the observation likelihoods
- If the probability for an assignment is too low, generate a new landmark

Per-Particle Data Association

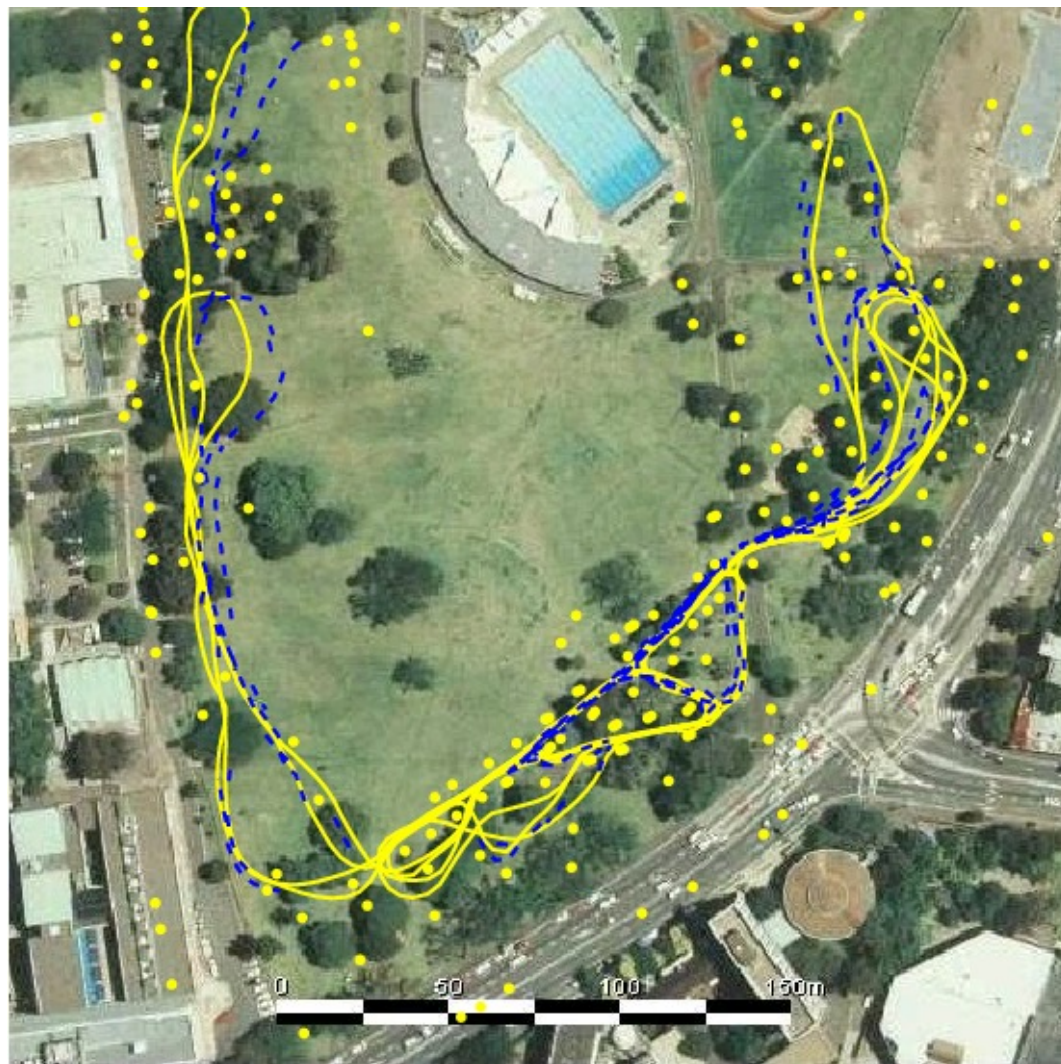
- Multi-modal belief supports multi-hypotheses data association
- Simple but effective data association applicable
- **Big advantage of FastSLAM over EKF**

Results – Victoria Park

- 4 km traverse
- < 2.5 m RMS position error
- 100 particles

Blue = GPS

Yellow = FastSLAM



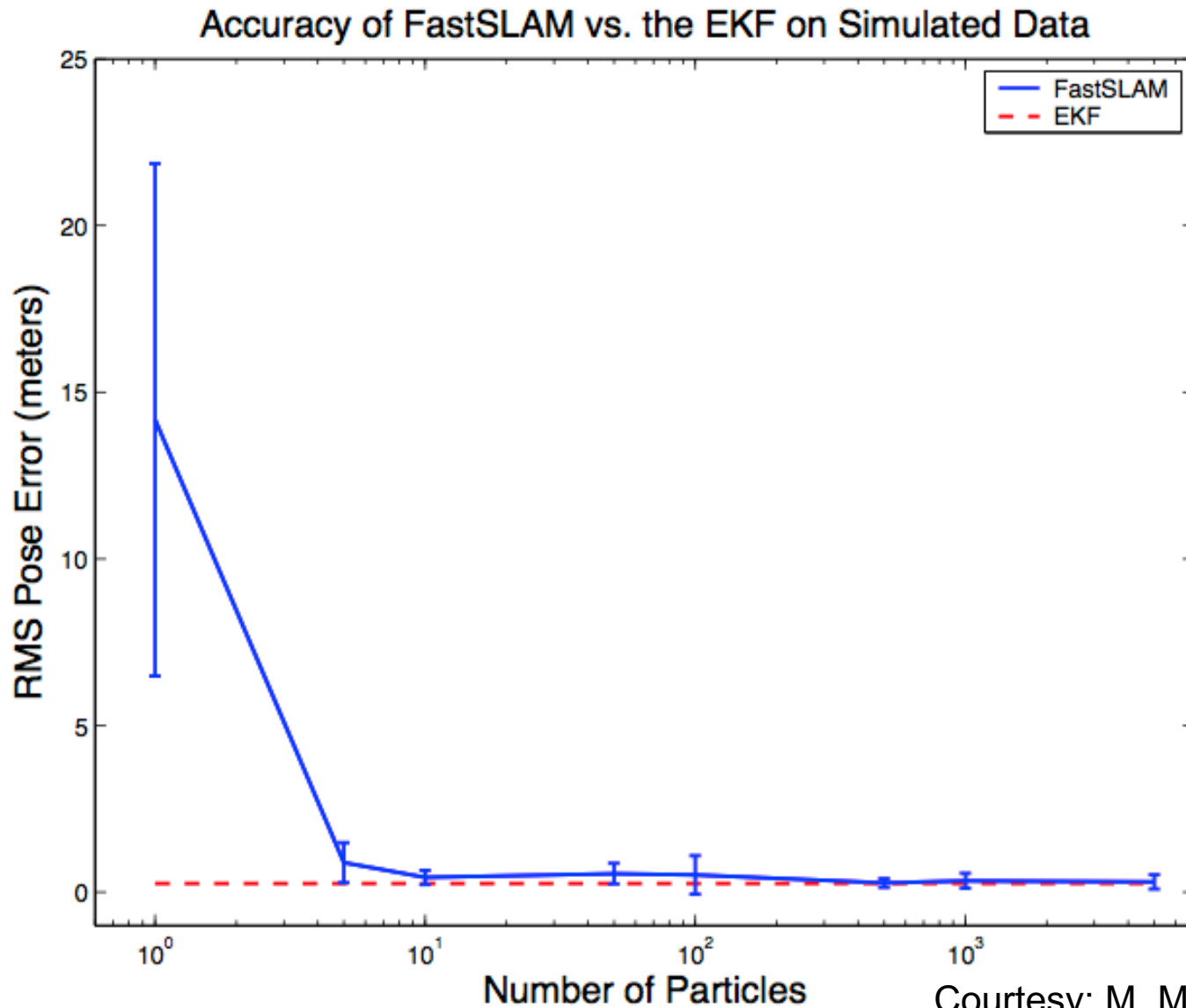
Courtesy: M. Montemerlo

Results – Victoria Park (Video)

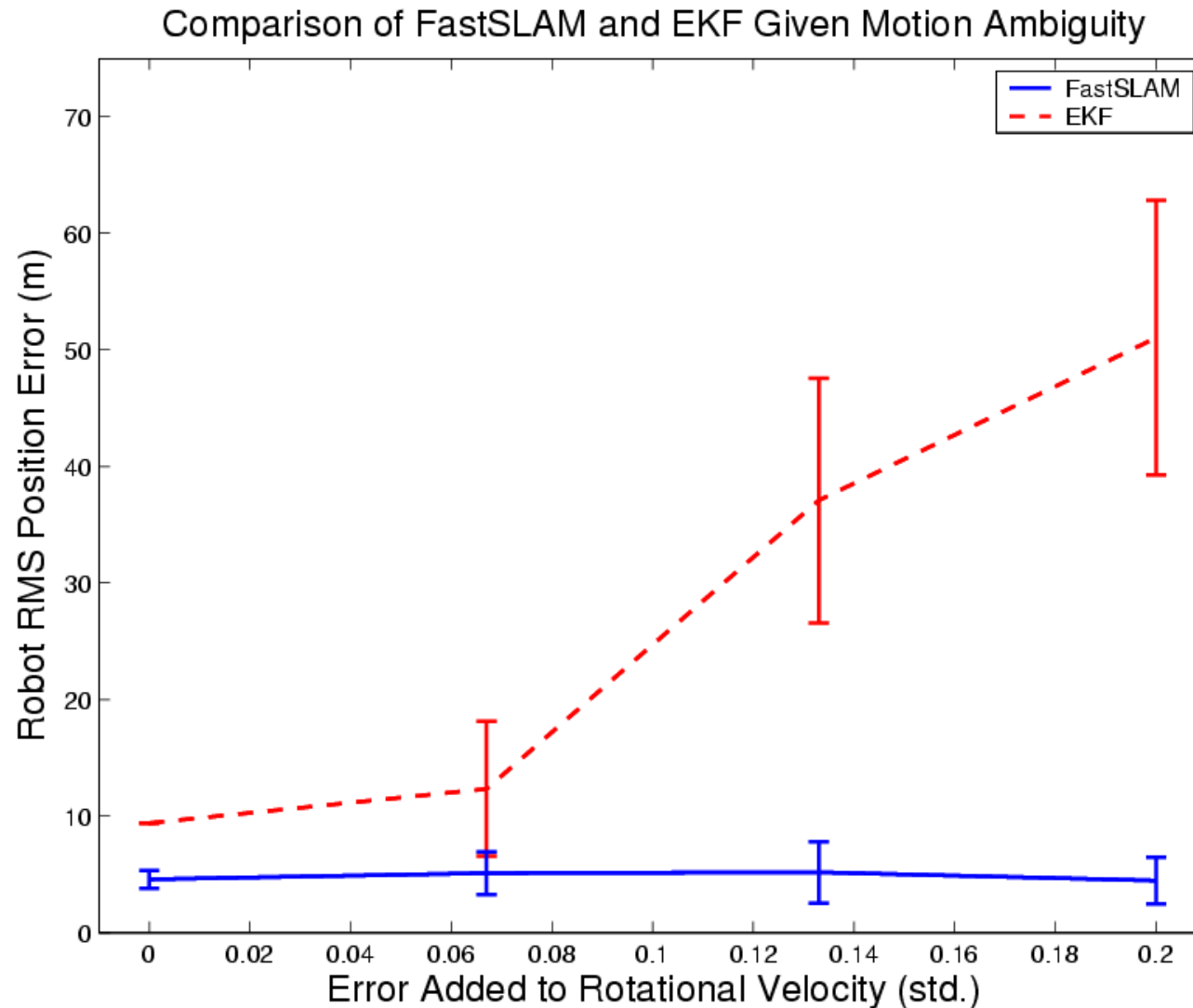


Courtesy: M. Montemerlo

Results (Sample Size)



Results (w. Motion Uncertainty)



Feature-Based FastSLAM 1.0

- Factors the SLAM posterior into low-dimensional estimation problems
- Models the robot's path by samples
- Computes the landmarks given particle pose
- Per-particle data association
- No robot pose uncertainty in the data association

FastSLAM Complexity – Simple Implementation

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

N = Number of particles

M = Number of map features

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

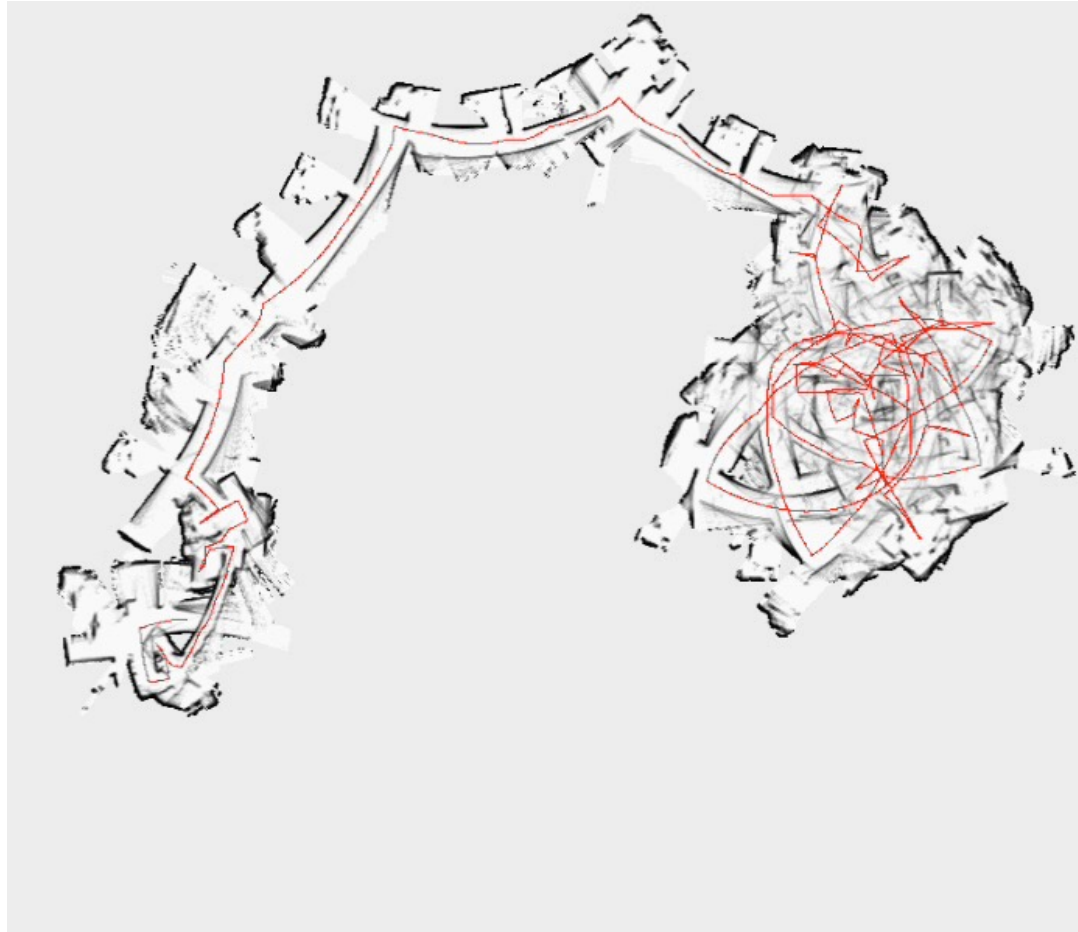
Summary: FastSLAM so far

- Feature-Based SLAM with particle filter
- Rao-Blackwellization: model the robot's path by sampling and compute the landmarks given the robot poses
- Data association on per-particle basis
- Robust to ambiguities in the data association
- Scales well (1 million+ features)
- Advantages compared to EFK SLAM

Now: FastSLAM for Grid Maps

- So far, we addressed landmark-based SLAM (KF-based SLAM, FastSLAM)
- We learned how to build grid maps assuming “known poses”

Grid-Based Mapping With Raw Odometry



Courtesy: Dirk Hähnel

Grid-Based Mapping

- Assuming known poses fails

Questions

- Can we solve the SLAM problem if no pre-defined landmarks are available?
- Can we use the ideas of FastSLAM to build grid maps?

Recap: Rao-Blackwellization for SLAM

Factorization of the SLAM posterior

poses map observations movements

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

Recap: Rao-Blackwellization for SLAM

Factorization of the SLAM posterior

poses map observations movements

↓ ↓ ↙ ↘

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

↑ ↑

path posterior map posterior
(particle filter) (given the path)

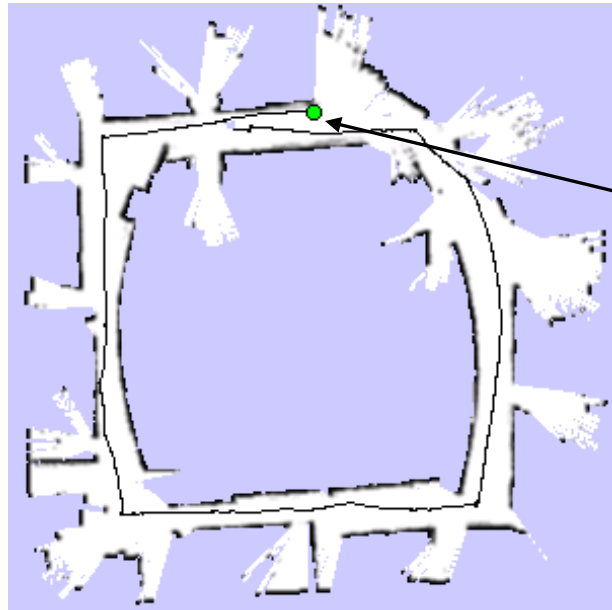
Grid-Based SLAM

- 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

Grid-Based Mapping with Rao-Blackwellized Particle Filters

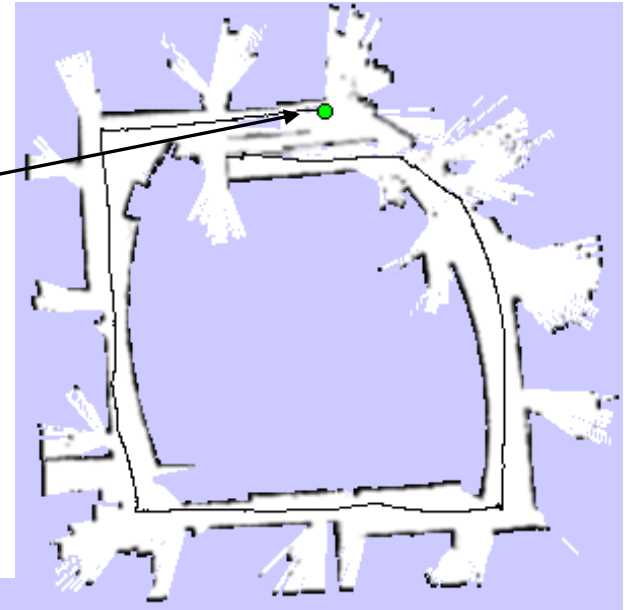
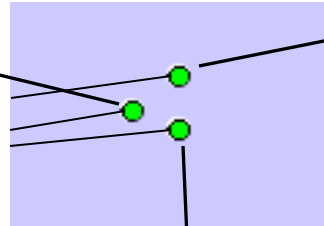
- Each particle represents a possible trajectory of the robot
- Each particle maintains its own map
- Each particle updates it upon “mapping with known poses”

Particle Filter Example

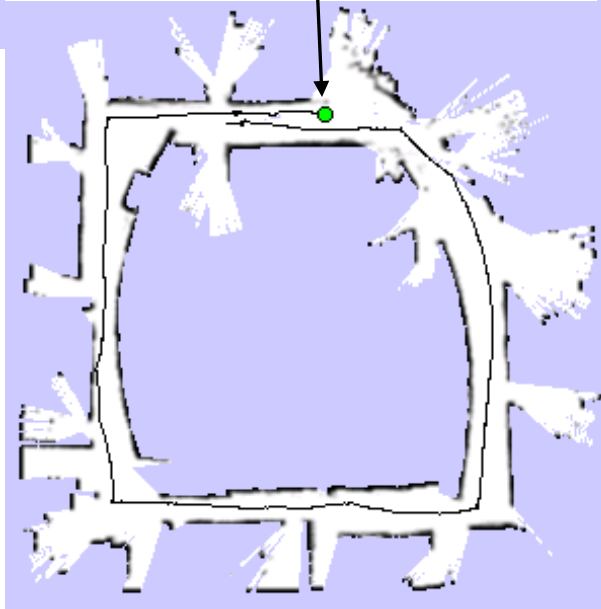


map of particle 1

3 particles

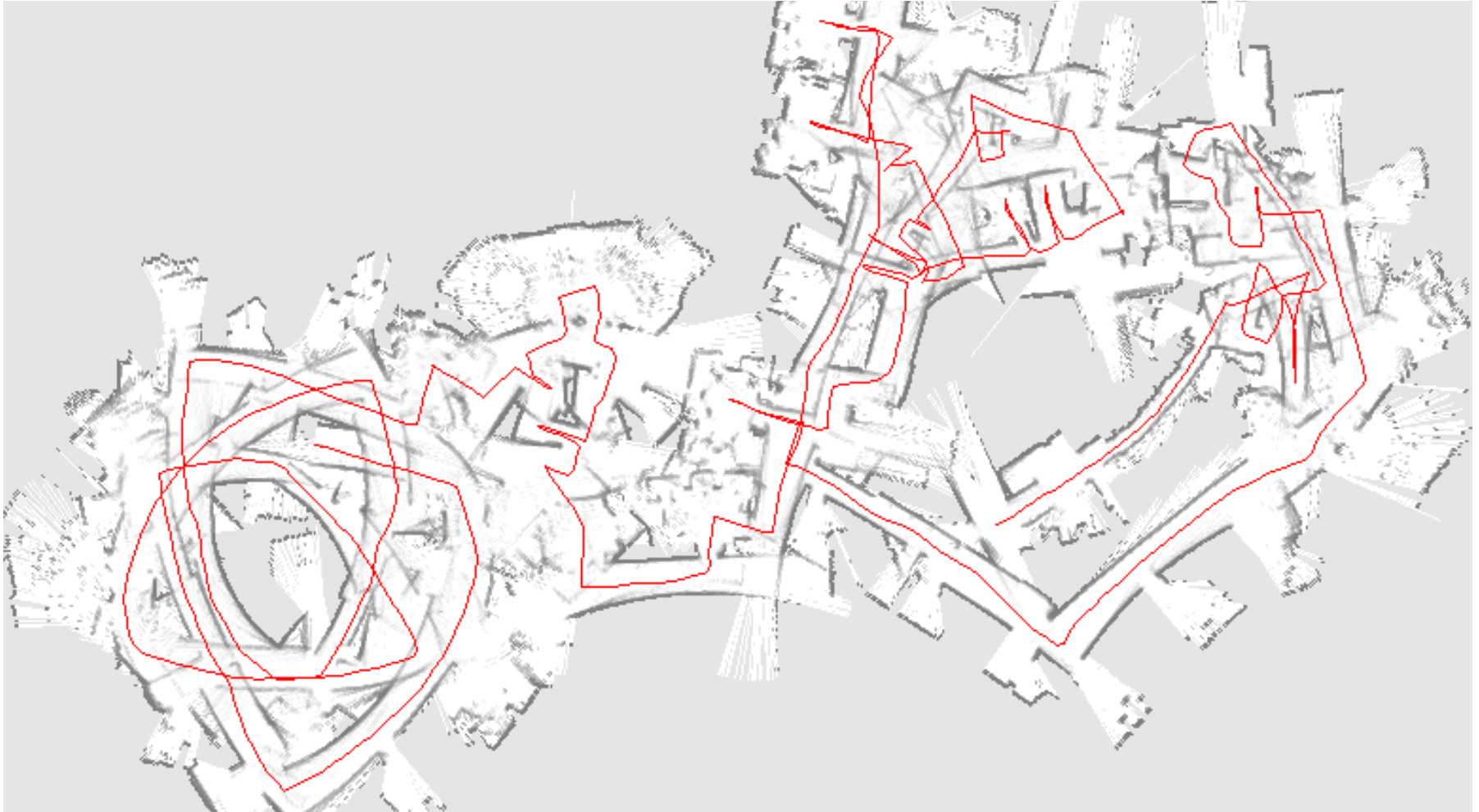


map of particle 3



map of particle 2

Performance of Grid-Based FastSLAM 1.0



Problem

- Too many samples are needed to sufficiently model the motion noise
- Increasing the number of samples is difficult as each map is quite large
- **Idea:** Improve the pose estimate **before** applying the particle filter

Recap: Pose Correction Using Scan Matching

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

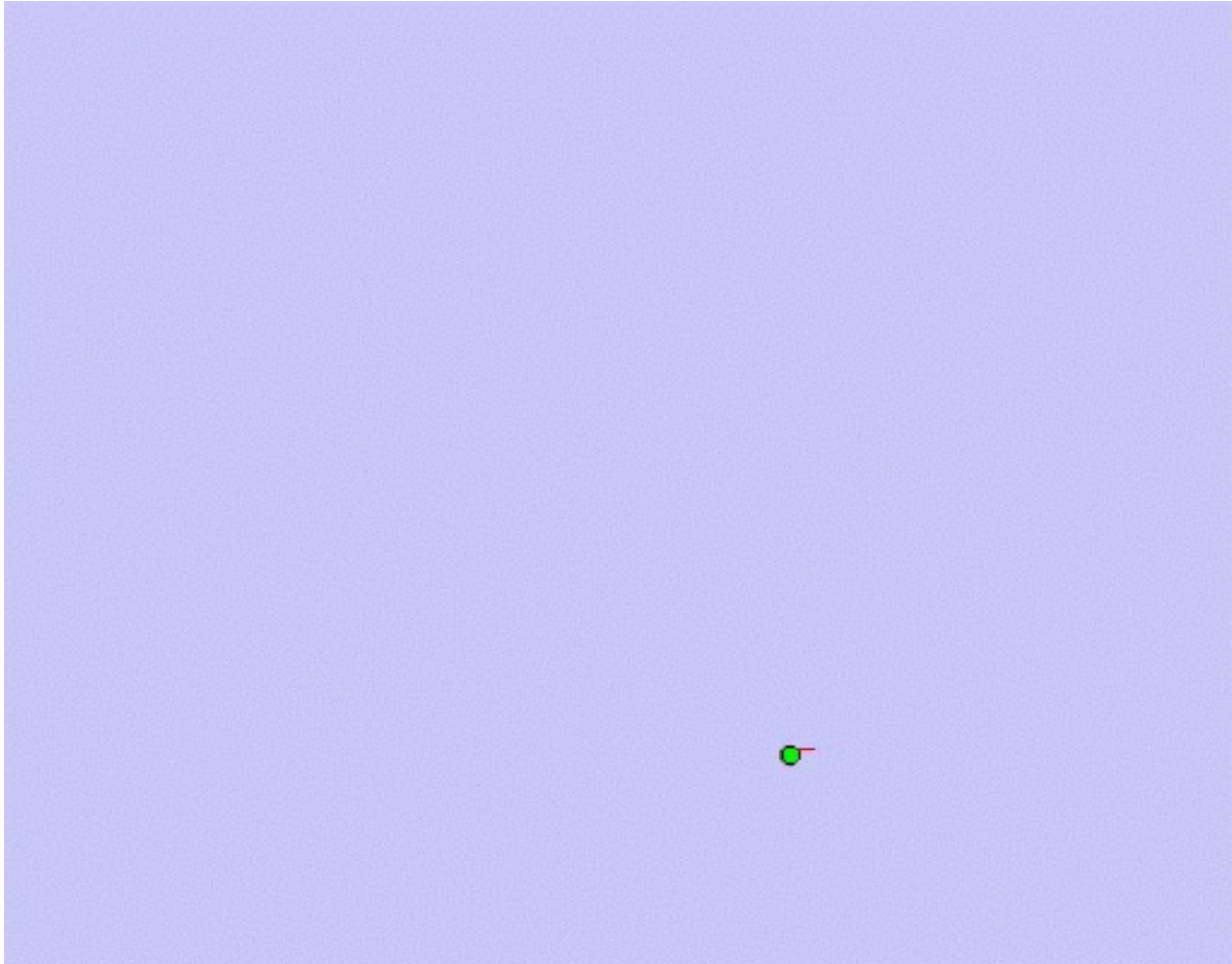
$$x_t^* = \underset{x_t}{\operatorname{argmax}} \left\{ p(z_t \mid x_t, m_{t-1}) p(x_t \mid u_t, x_{t-1}^*) \right\}$$

current measurement

robot motion

map constructed so far

Mapping Using Scan Matching



Courtesy: Dirk Hähnel

Grid-Based FastSLAM with Improved Odometry

- Scan matching provides a **locally consistent** pose correction
- **Idea:** Pre-correct short odometry sequences using scan matching and use them as input to FastSLAM
- Fewer particles are needed, since the error in the input is smaller

Acknowledgment

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