CSE-571
Robotics

SLAM: Simultaneous Localization and Mapping

Many slides courtesy of Ryan Eustice, Cyrill Stachniss, John Leonard, Dieter Fox

The SLAM Problem

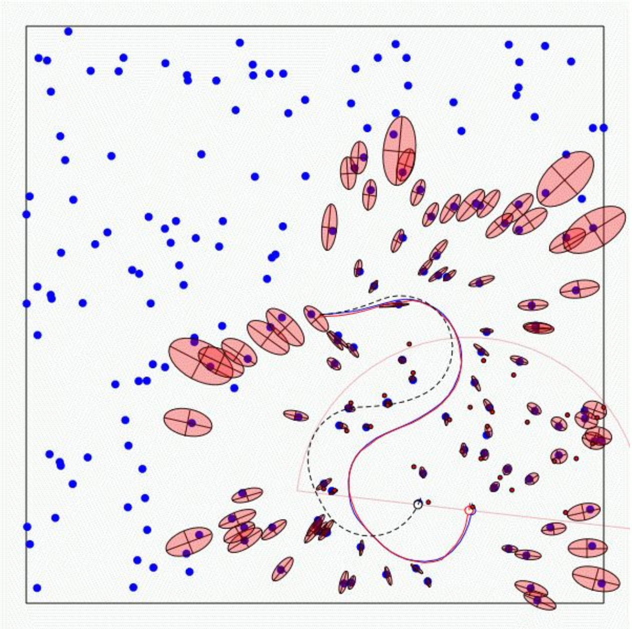
A robot is exploring an unknown, static environment.

Given:

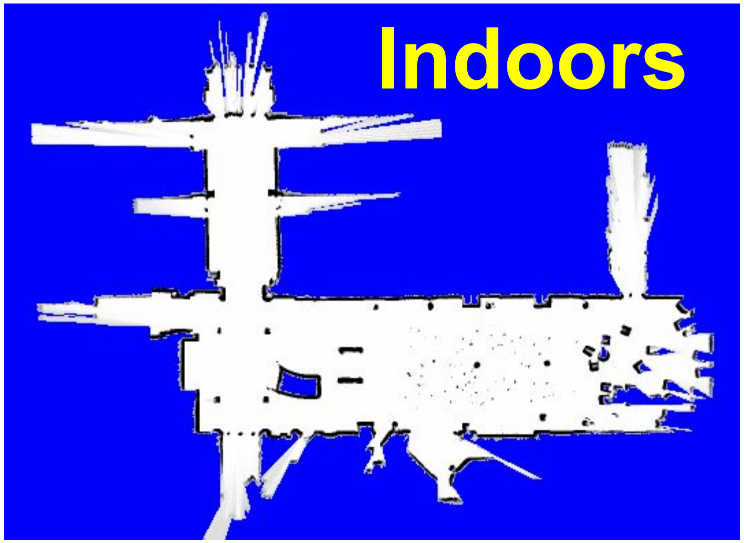
- The robot's controls
- Observations of nearby features

Estimate:

- Map of features
- Path of the robot



SLAM Applications



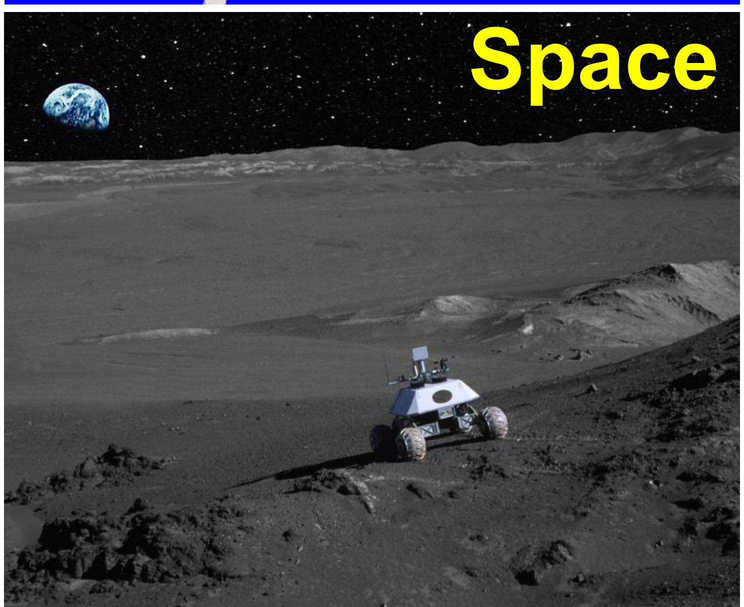
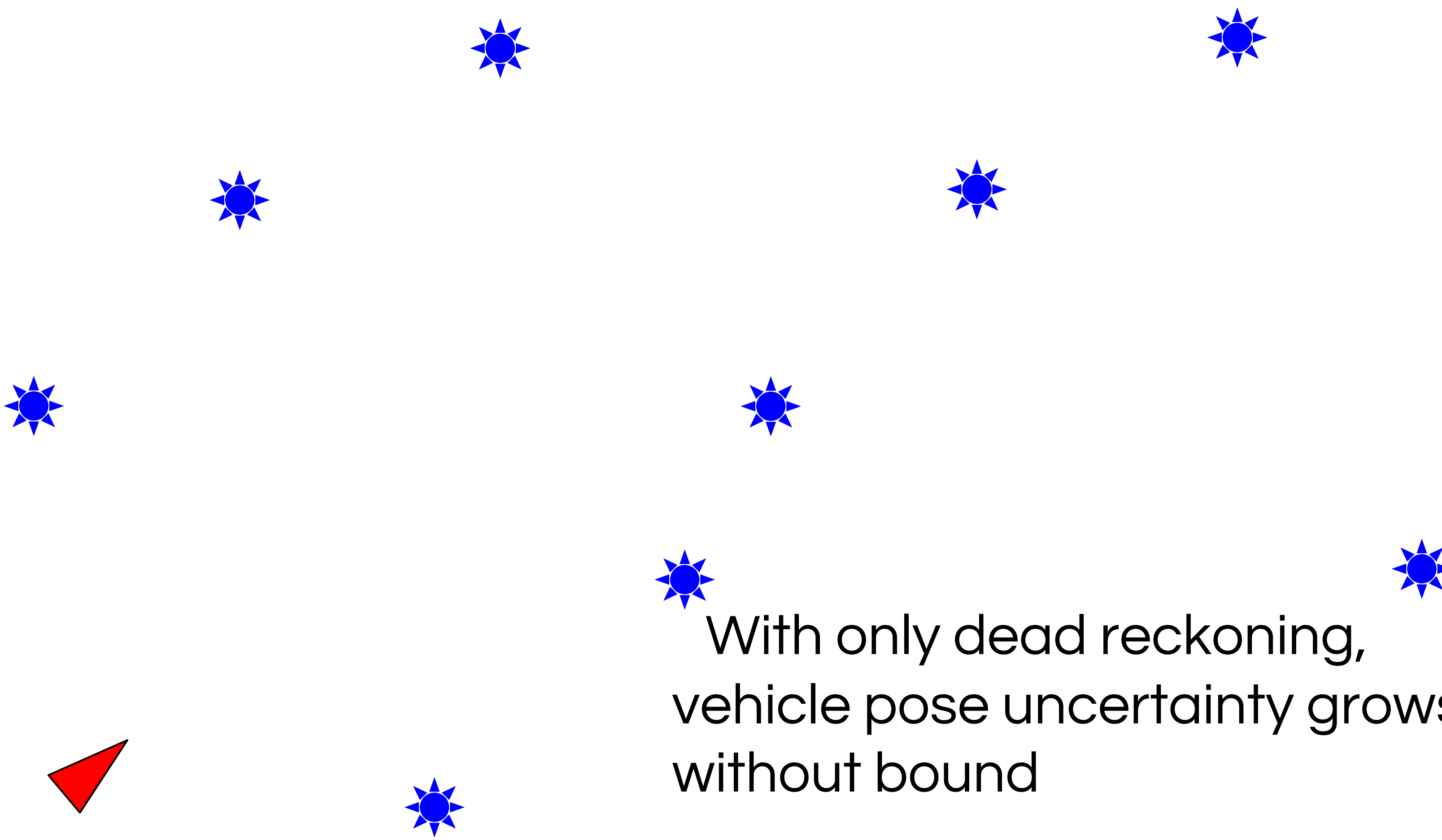
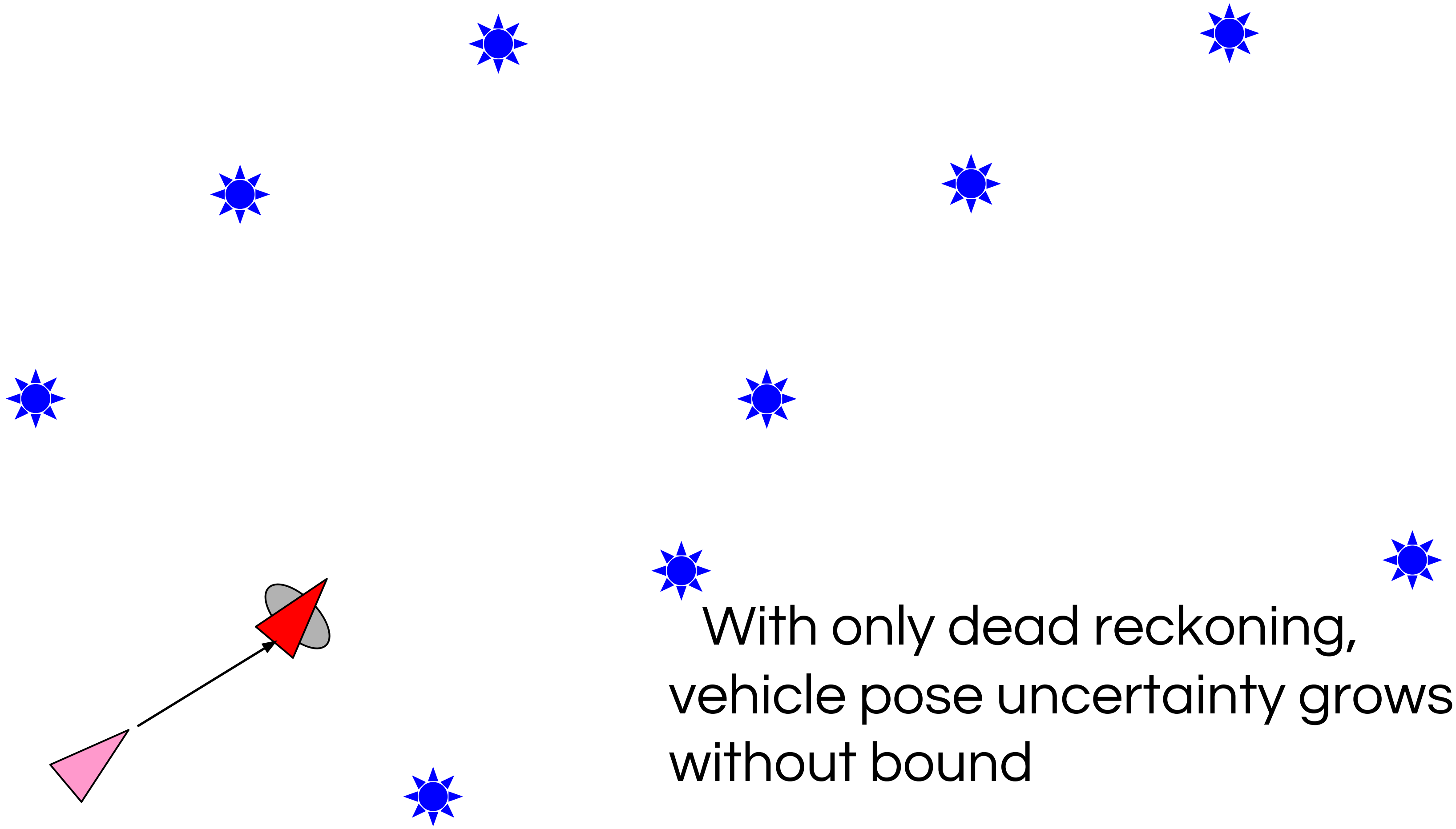


Illustration of SLAM without Landmarks



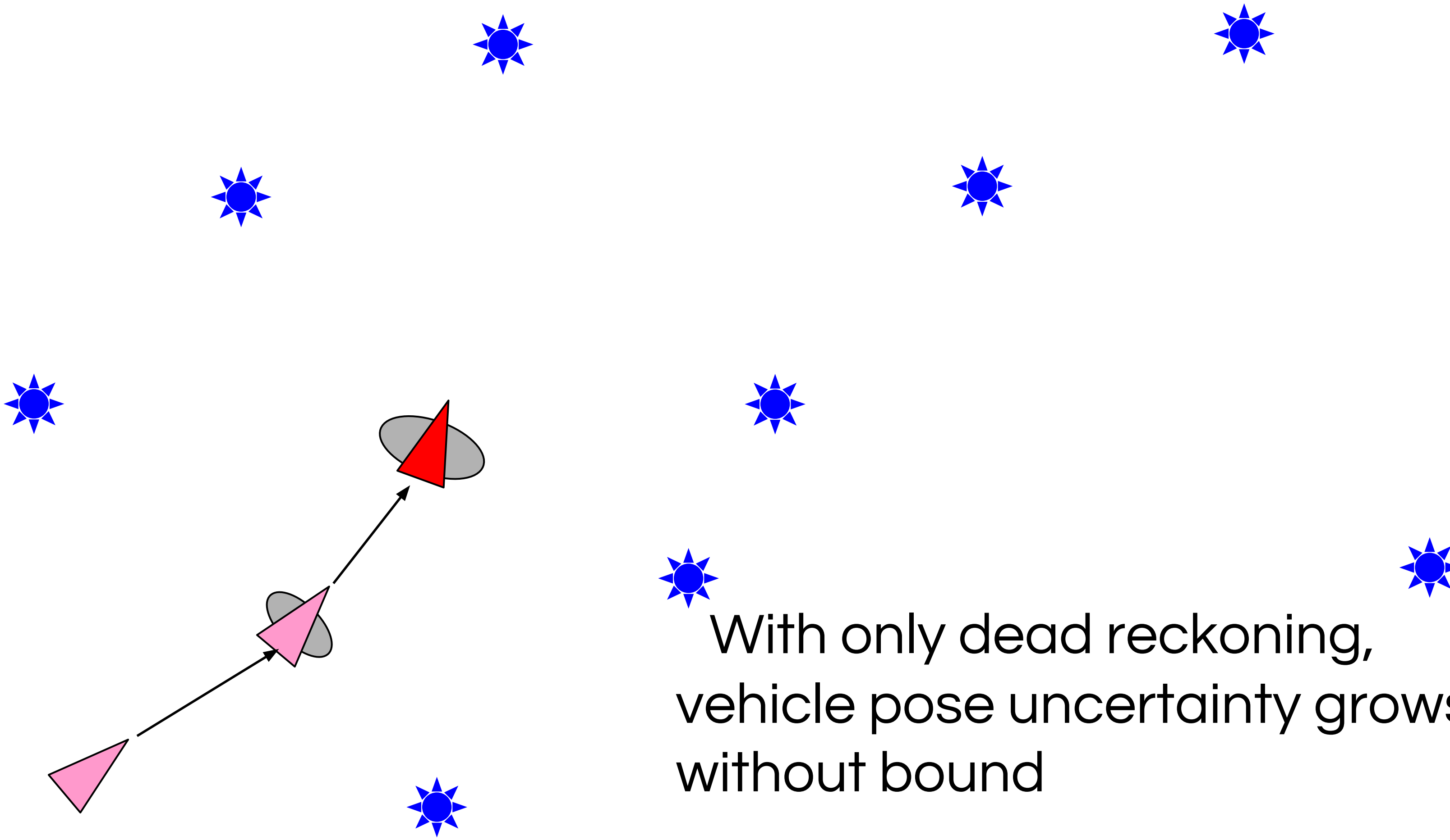
With only dead reckoning, vehicle pose uncertainty grows without bound

Illustration of SLAM without Landmarks



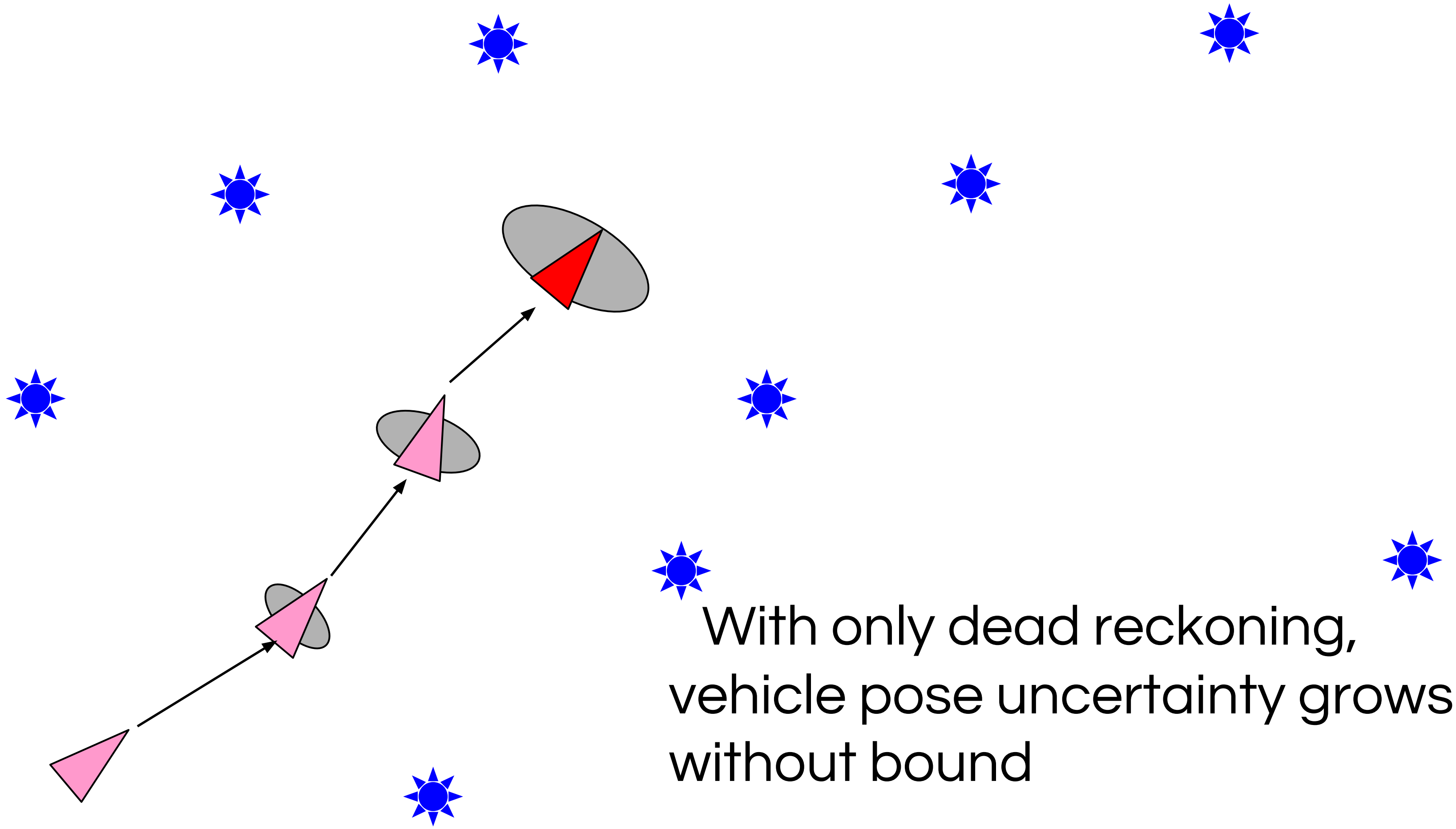
With only dead reckoning, vehicle pose uncertainty grows without bound

Illustration of SLAM without Landmarks



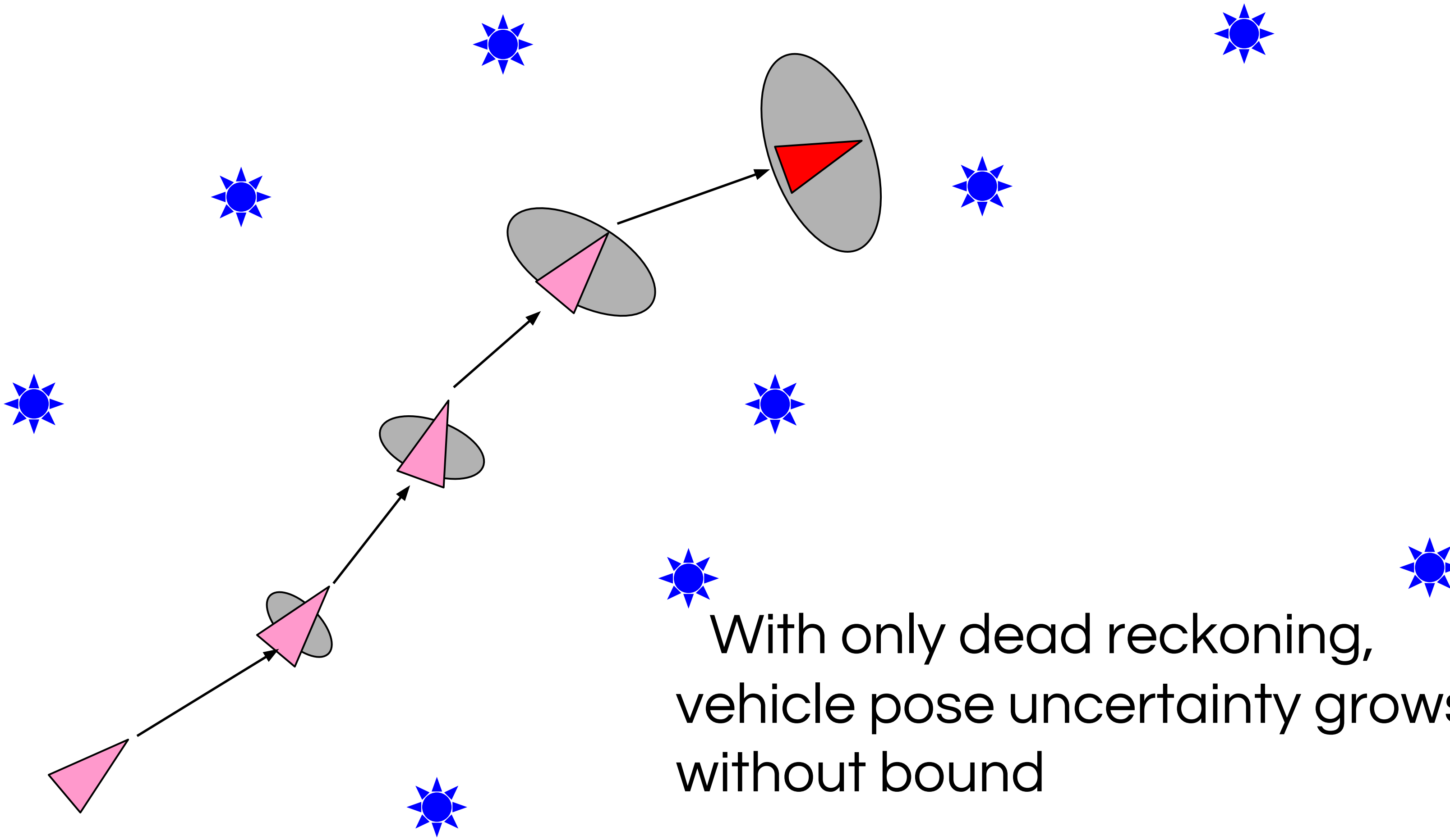
With only dead reckoning, vehicle pose uncertainty grows without bound

Illustration of SLAM without Landmarks



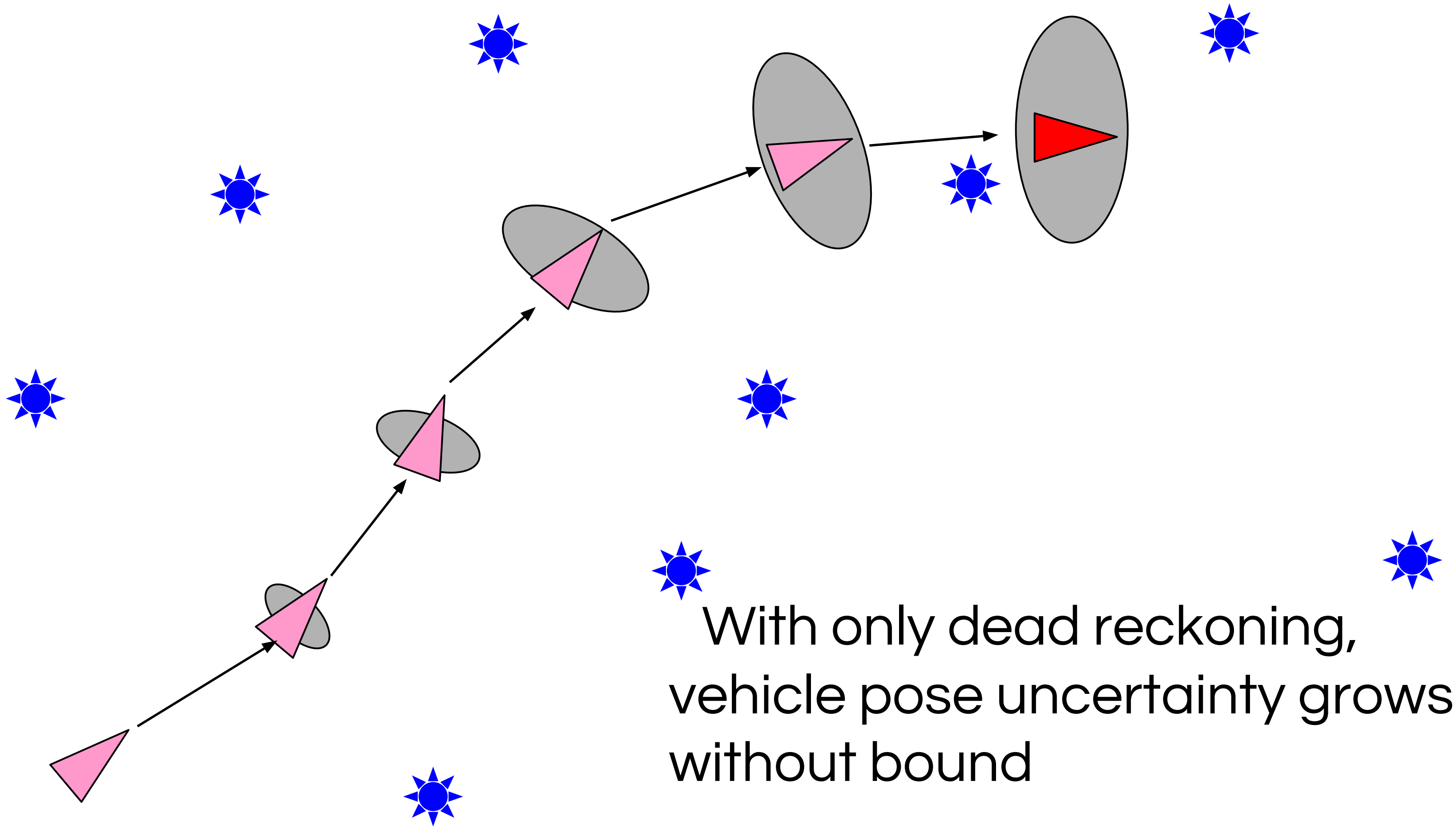
With only dead reckoning, vehicle pose uncertainty grows without bound

Illustration of SLAM without Landmarks



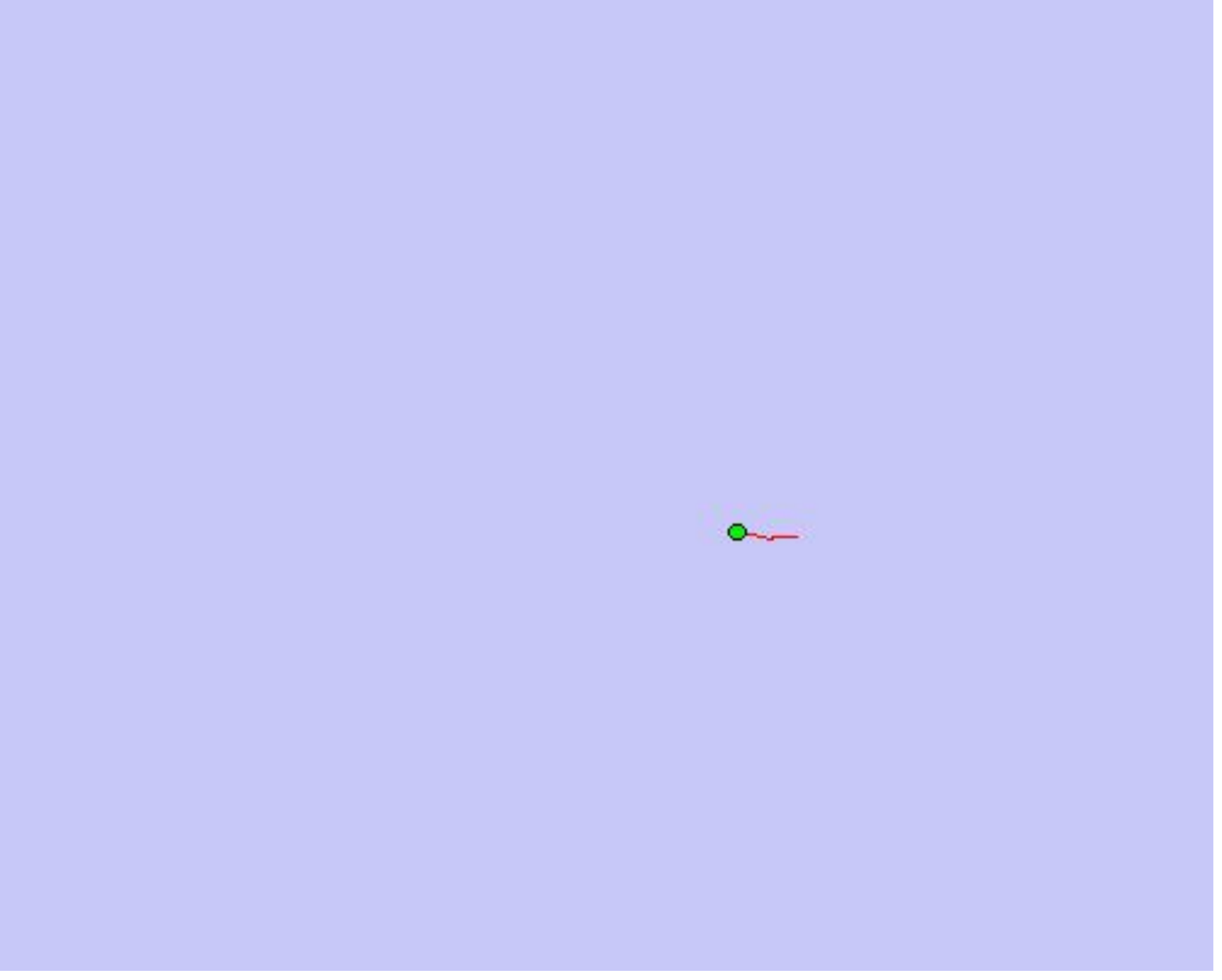
With only dead reckoning, vehicle pose uncertainty grows without bound

Illustration of SLAM without Landmarks

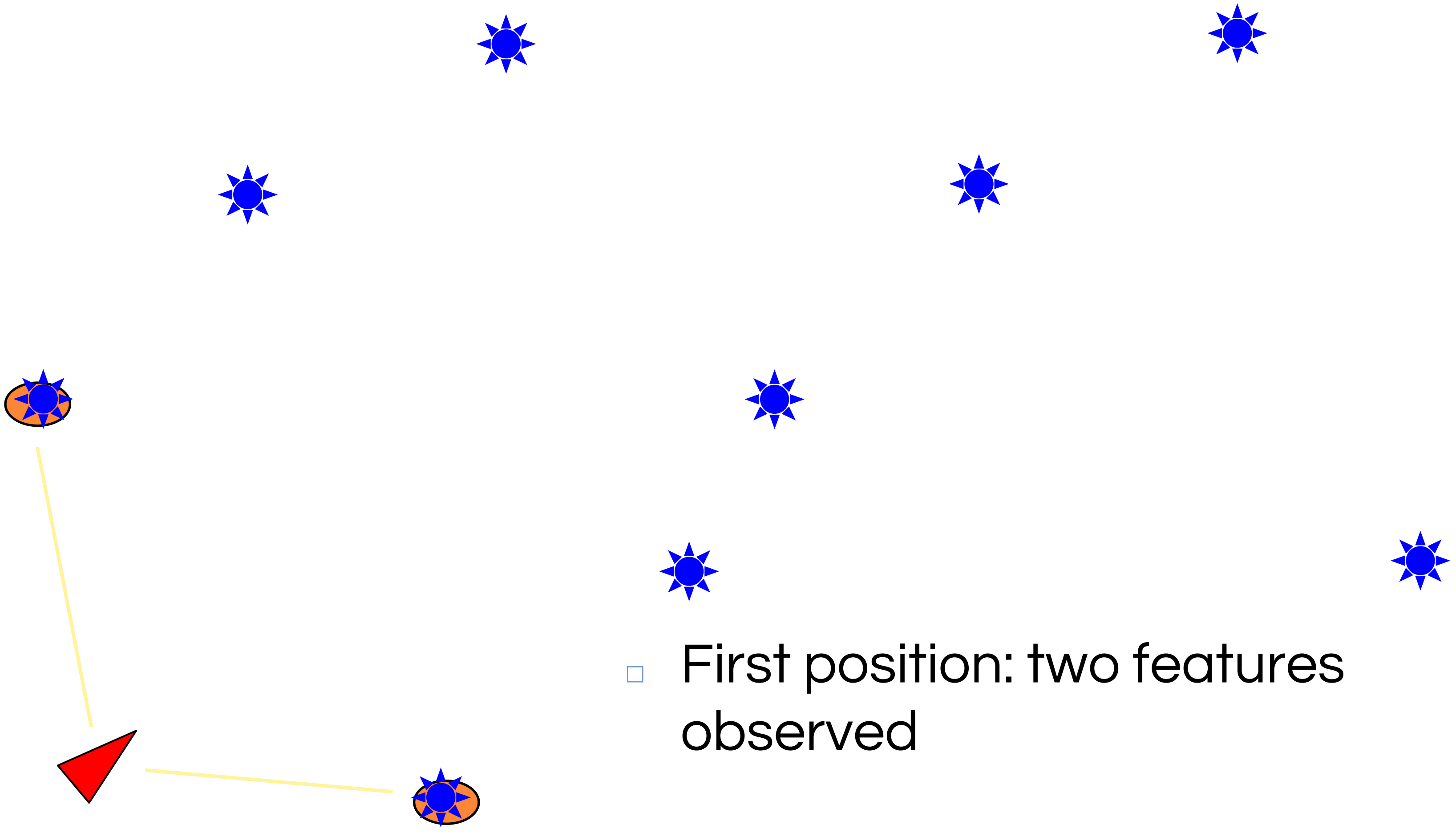


With only dead reckoning, vehicle pose uncertainty grows without bound

Mapping with Raw Odometry

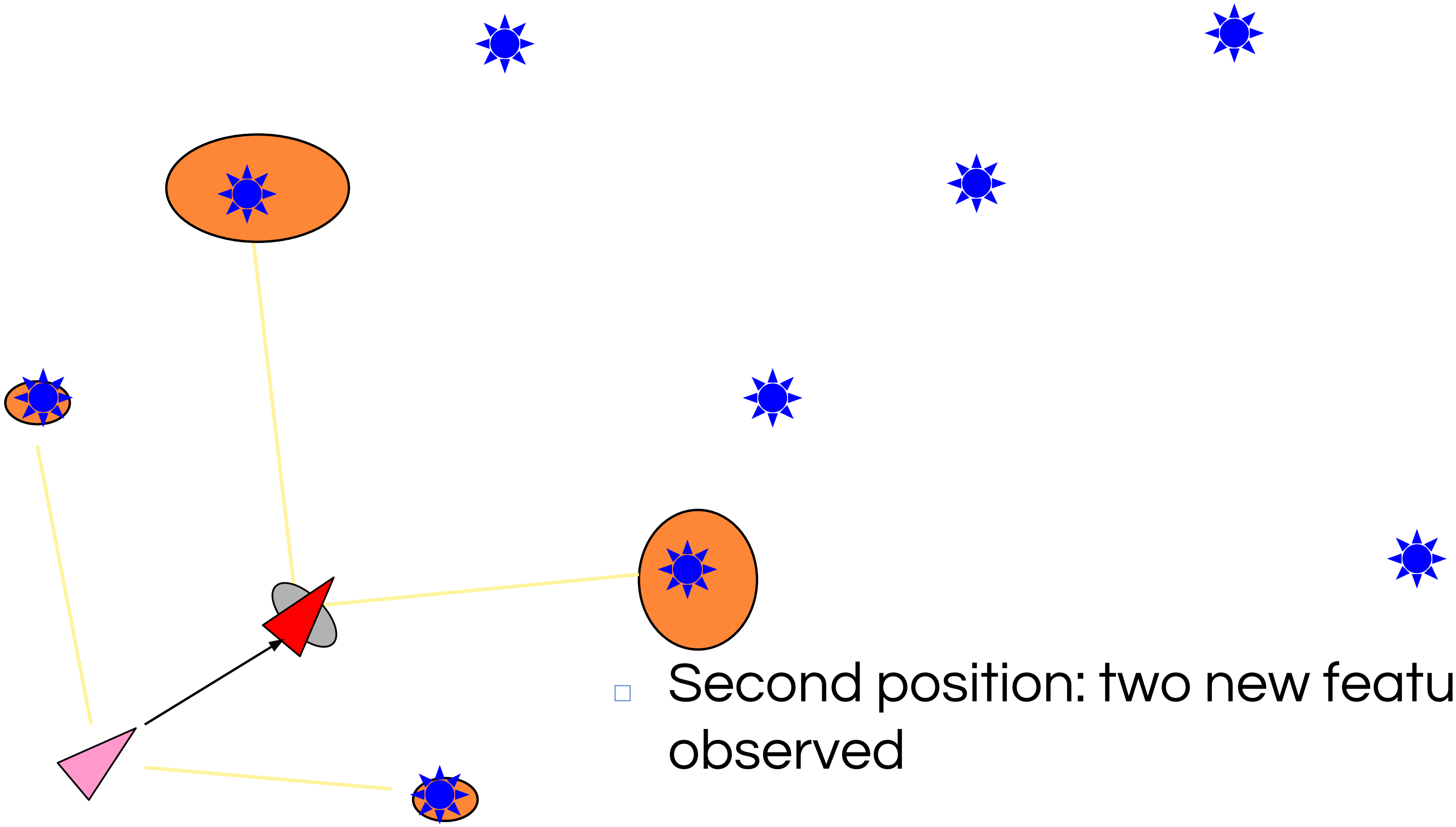


Repeat, with Measurements of Landmarks



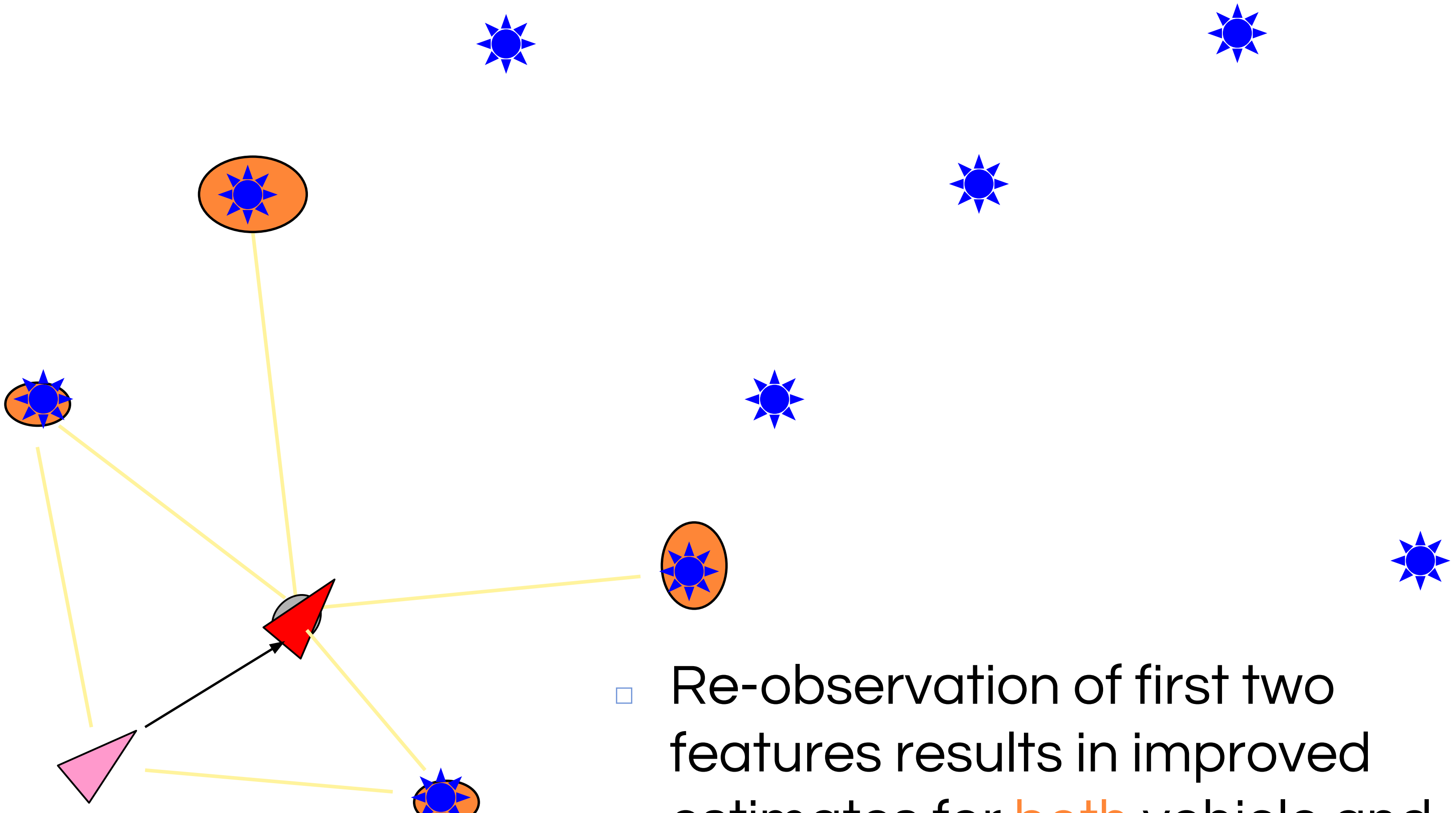
- First position: two features observed

Illustration of SLAM with Landmarks



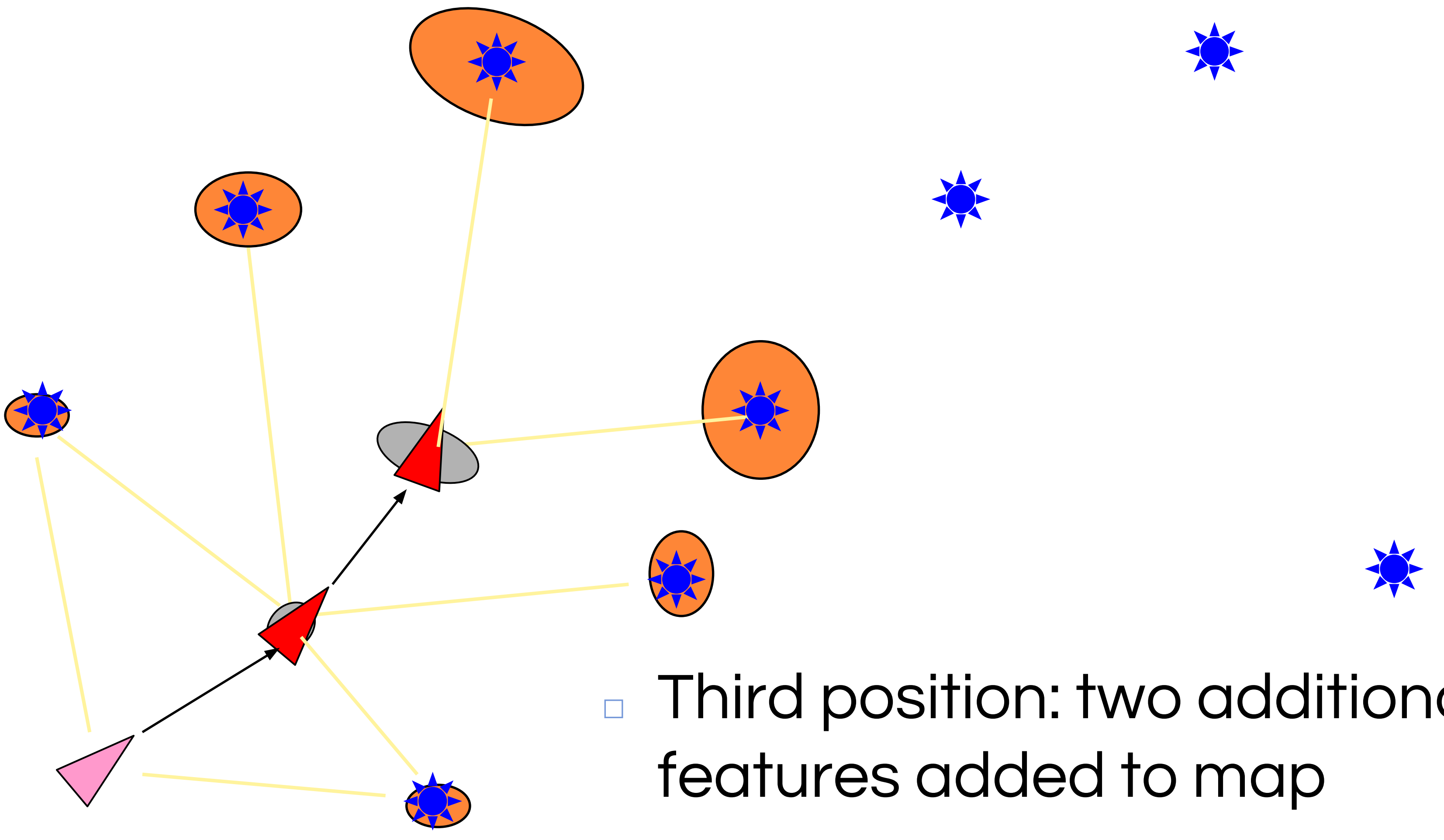
- Second position: two new features observed

Illustration of SLAM with Landmarks



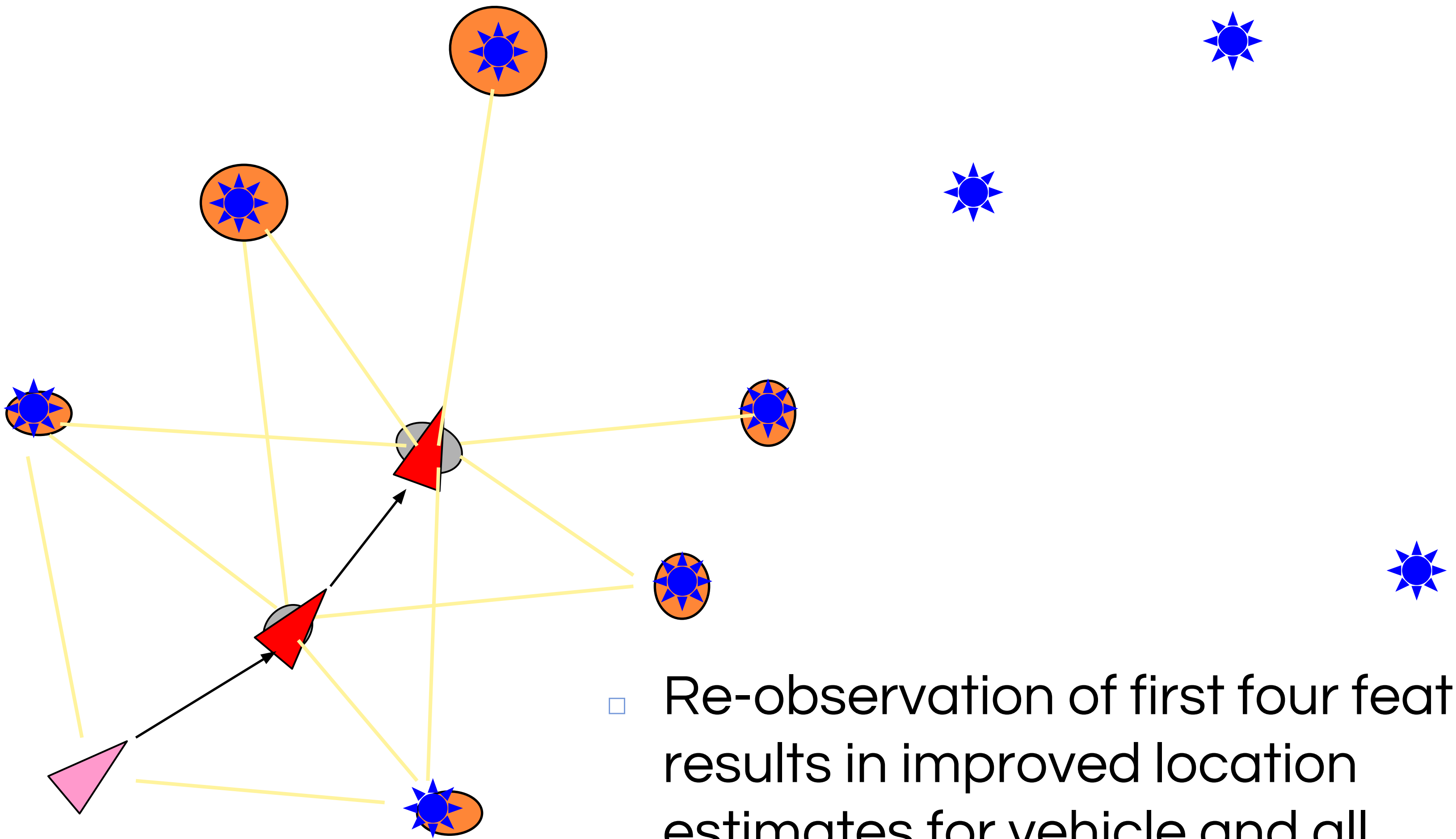
- Re-observation of first two features results in improved estimates for both vehicle <u>and</u> feature

Illustration of SLAM with Landmarks



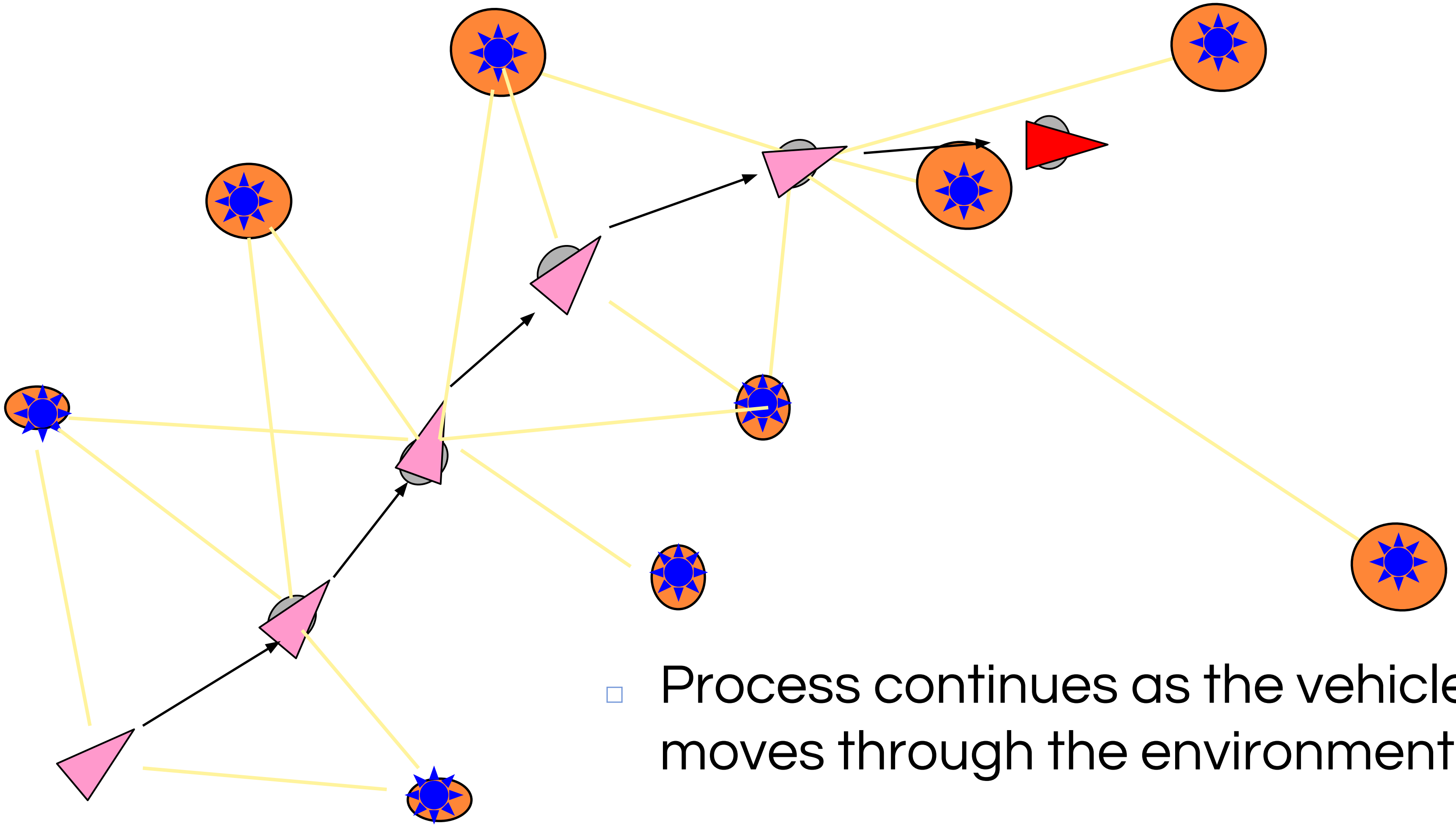
- Third position: two additional features added to map

Illustration of SLAM with Landmarks



- Re-observation of first four features results in improved location estimates for vehicle and all features

Illustration of SLAM with Landmarks



- Process continues as the vehicle moves through the environment

What is SLAM?

Localization: Estimate current pose, given map, controls, and observations

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

Mapping: Build map given poses and observations

$$p(m | x_{1:t}, z_{1:t})$$

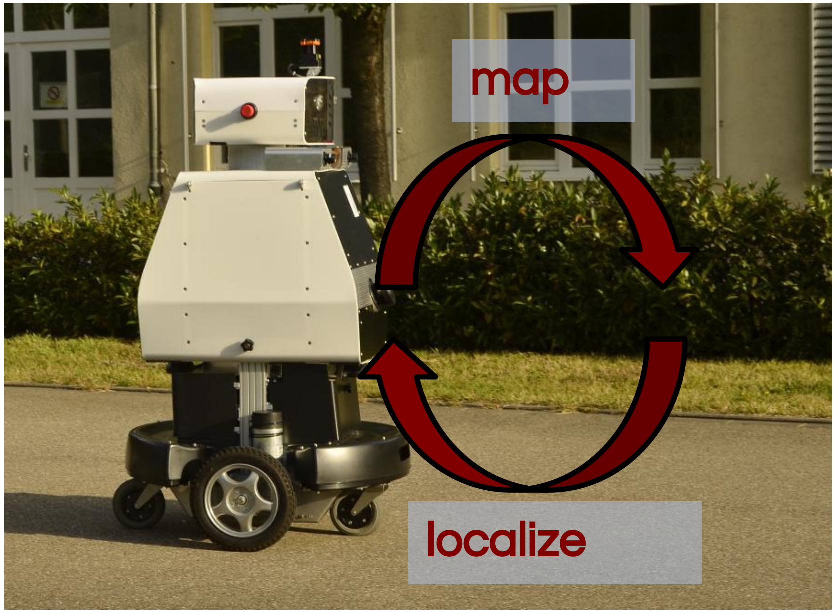
Simultaneous Mapping and Localization (SLAM):

Find poses and map given controls and observations

$$p(x_{1:t}, m | u_{1:t}, z_{1:t})$$

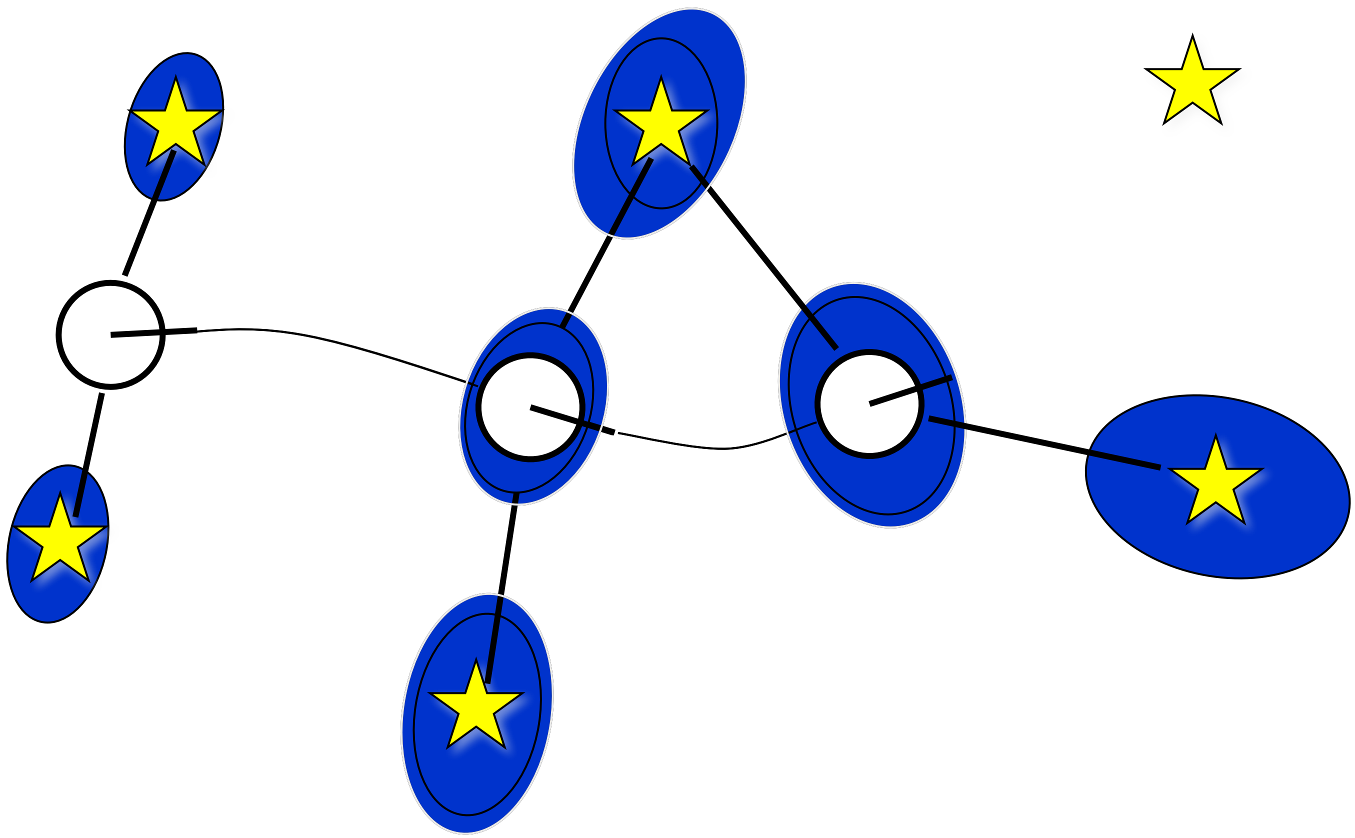
Simultaneous Localization and Mapping (SLAM)

- Building a map and locating the robot in the map at the same time
- Chicken-and-egg problem



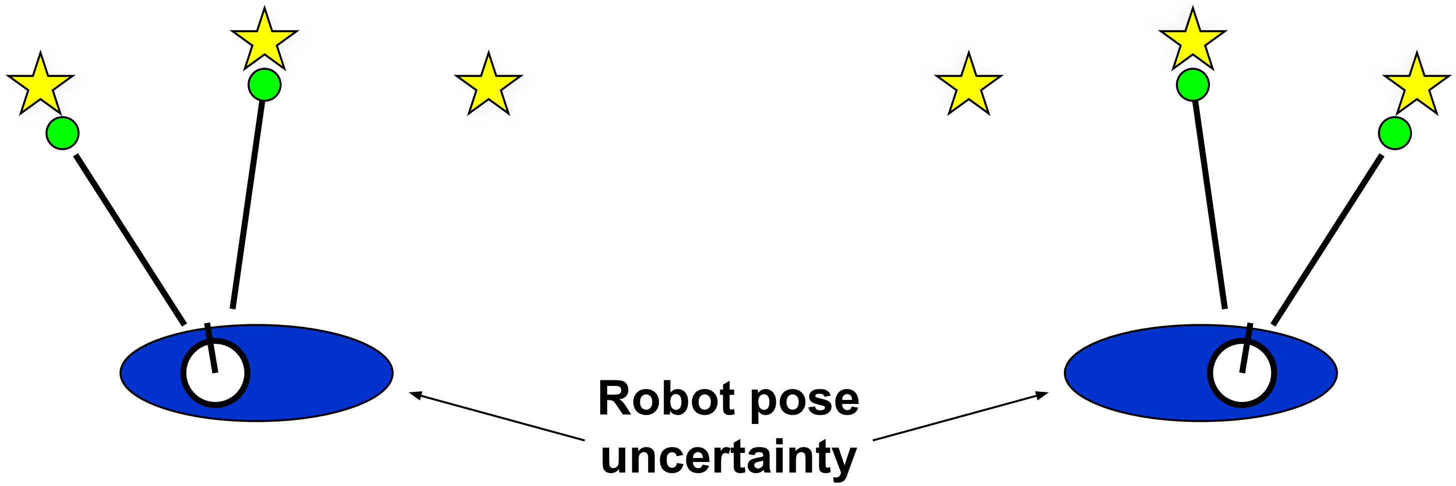
Why is SLAM a hard problem?

SLAM: robot path and map are both **unknown**



Robot path error correlates errors in the map

Why is SLAM a hard problem?



- In the real world, the mapping between observations and landmarks is unknown
- Picking wrong data associations can have catastrophic consequences
- Pose error correlates data associations

SLAM:

Simultaneous Localization and Mapping

- Full SLAM:

Estimates entire path and map!

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

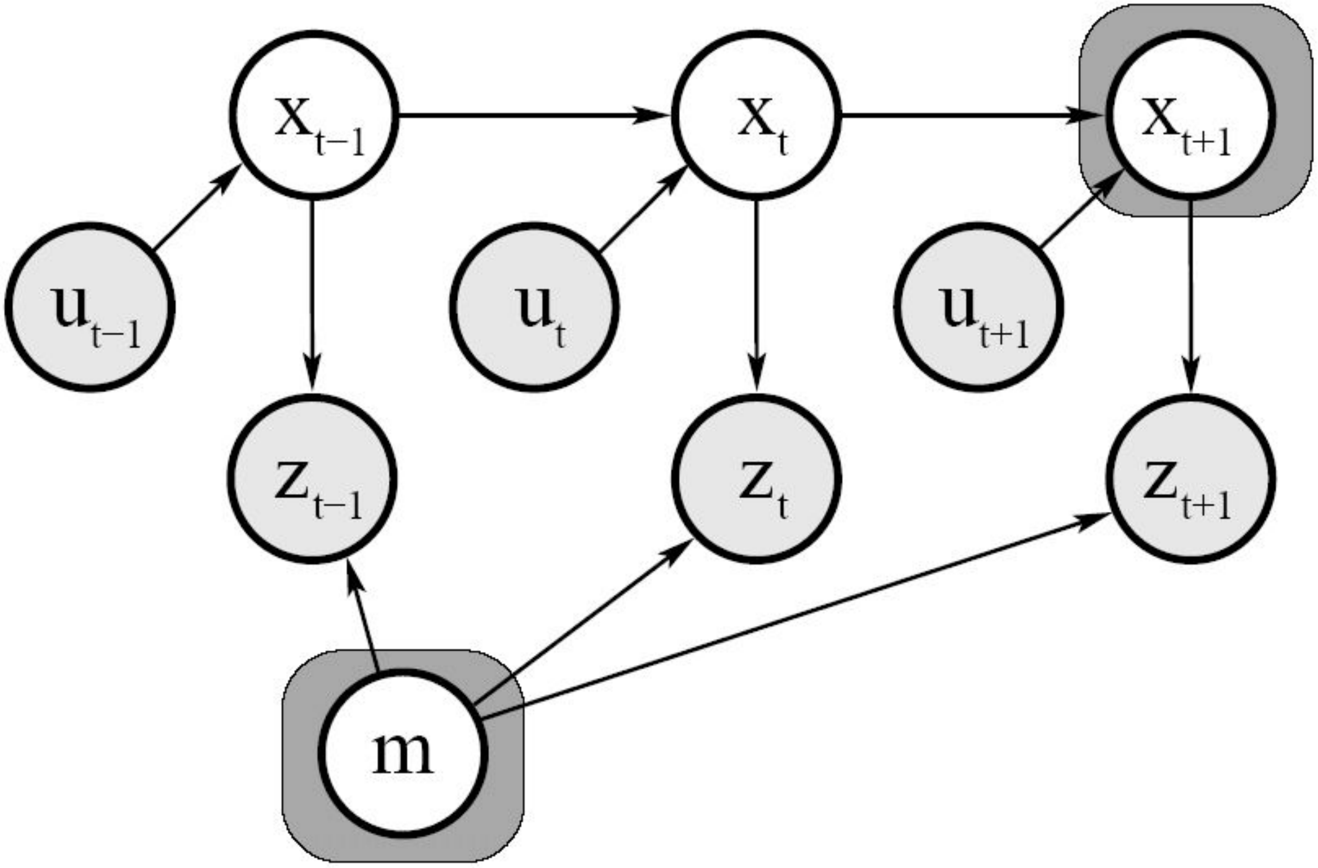
- Online SLAM:

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

Integrations typically done one at a time

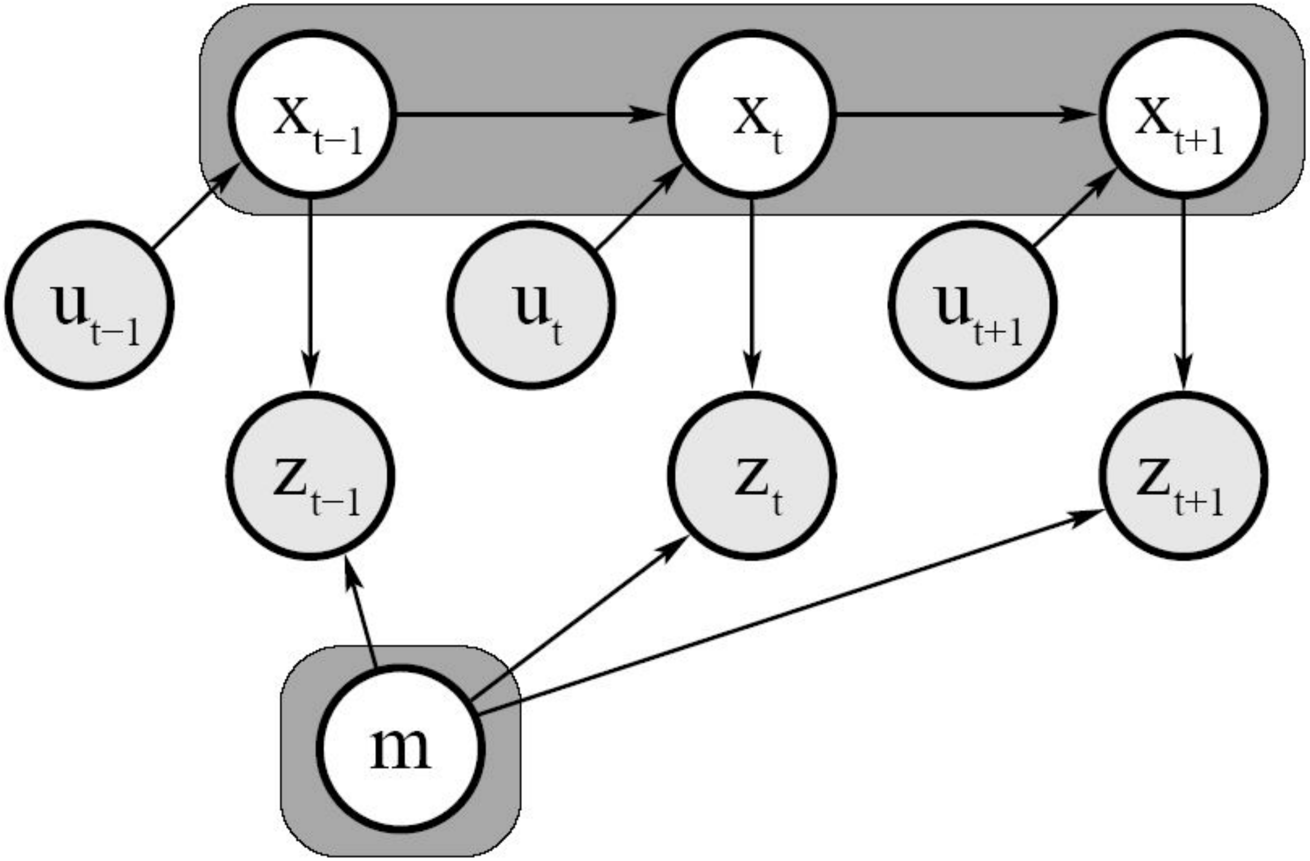
Estimates most recent pose and map!

Graphical Model of Online SLAM:



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

Graphical Model of Full SLAM:



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

Three Main Paradigms

Kalman filter

Particle filter

Graph-based

Bayes Filter

- Recursive filter with prediction and correction step

- Prediction

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

- Correction

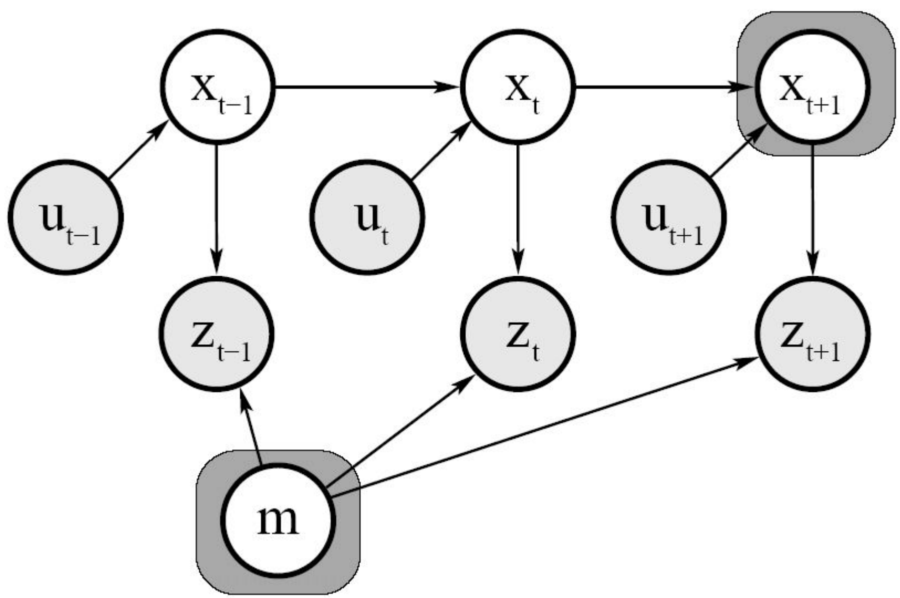
$$bel(x_t) = \eta \, p(z_t \mid x_t) \; \overline{bel}(x_t)$$

Courtesy: Cyrill

EKF for Online SLAM

- We consider here the Kalman filter as a solution to the online SLAM problem

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



Courtesy: Thrun, Burgard, Fox

Extended Kalman Filter Algorithm

1: **Extended_Kalman_filter**$(\mu_{t-1}, \Sigma_{t-1}, u_t, z_t)$**:**

2: $\bar{\mu}_t = g(u_t, \mu_{t-1})$

3: $\bar{\Sigma}_t = G_t \, \Sigma_{t-1} \, G_t^T + R_t$

4: $K_t = \bar{\Sigma}_t \, H_t^T (H_t \, \bar{\Sigma}_t \, H_t^T + Q_t)^{-1}$

5: $\mu_t = \bar{\mu}_t + K_t(z_t - h(\bar{\mu}_t))$

6: $\Sigma_t = (I - K_t \, H_t) \, \bar{\Sigma}_t$

7: $return \; \mu_t, \Sigma_t$

Courtesy: Cyrill

EKF SLAM

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

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

Courtesy: Cyrill

Literature

EKF SLAM

- Thrun et al.: “Probabilistic Robotics”, Chapter 10
- Smith, Self, & Cheeseman: “Estimating Uncertain Spatial Relationships in Robotics”
- Dissanayake et al.: “A Solution to the Simultaneous Localization and Map Building (SLAM) Problem”
- Durrant-Whyte & Bailey: “SLAM Part 1” and “SLAM Part 2” tutorials

Three Main Paradigms

Kalman filter

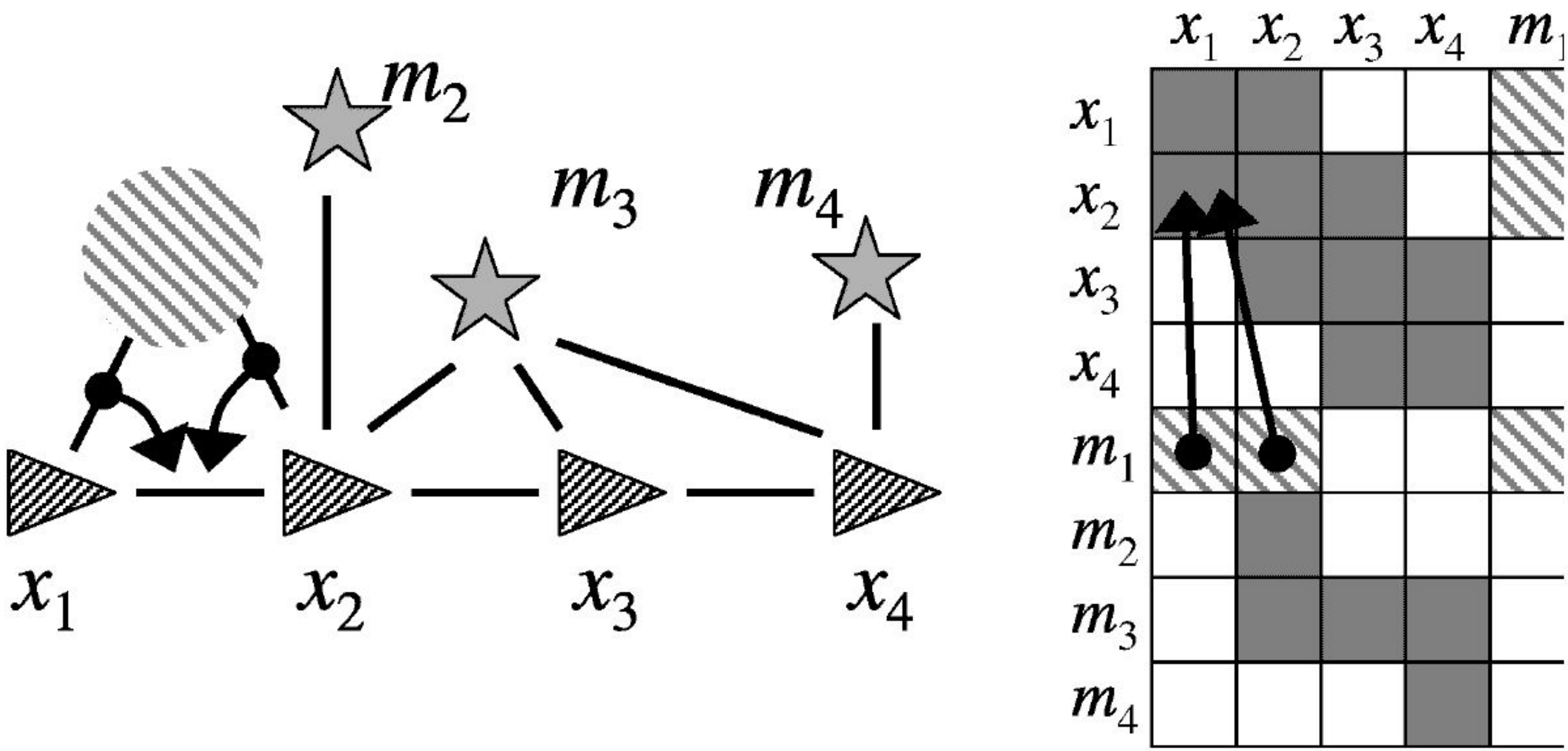
Particle filter

Graph-based

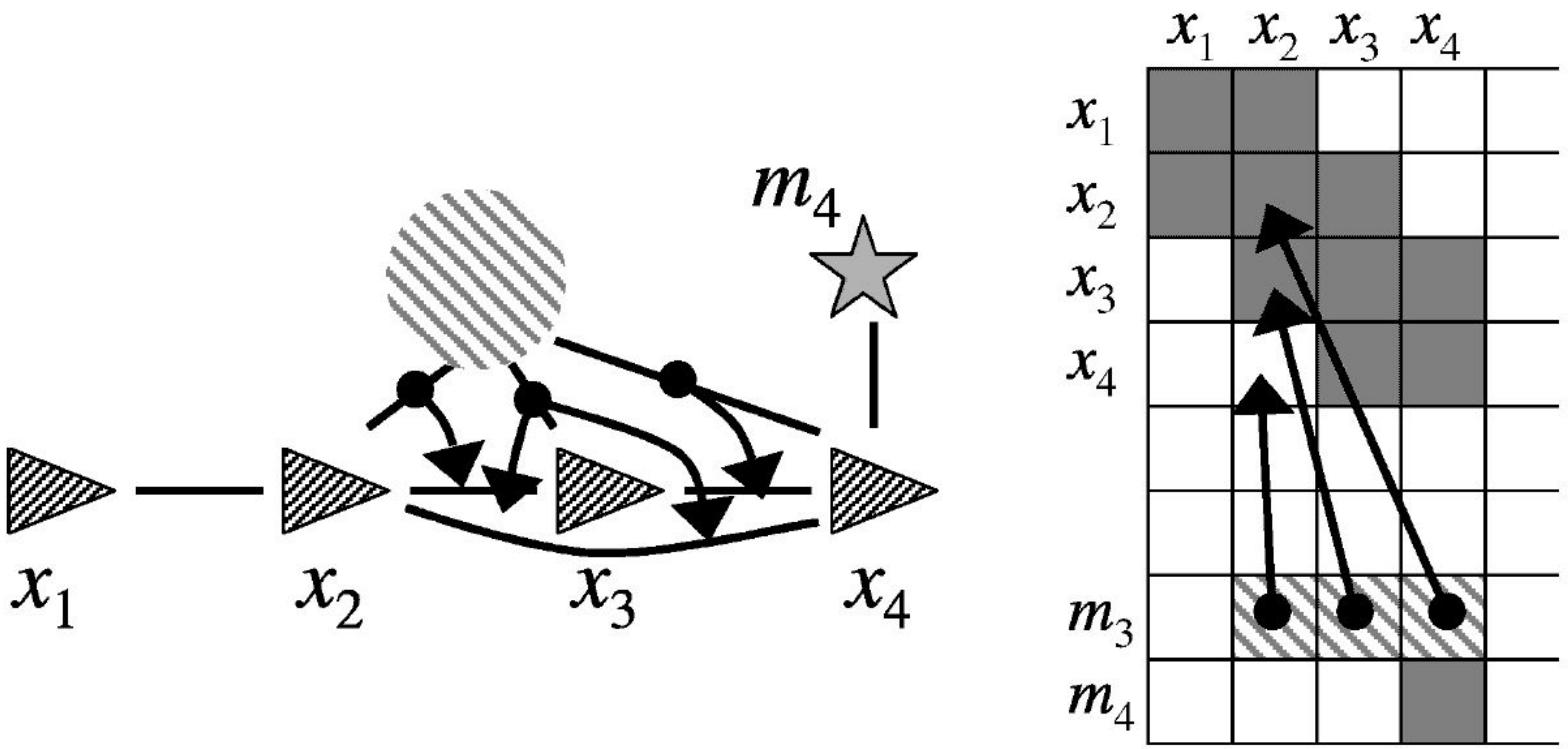
Graph-SLAM

- Full SLAM technique
- Generates probabilistic links
- Computes map only occasionally
- Based on Information Filter form

Graph-SLAM Inference (1)



Graph-SLAM Inference (2)



Graph-SLAM Summary

- Adresses full SLAM problem
- Constructs link graph between poses and poses/landmarks
- Graph is sparse: number of edges linear in number of nodes
- Inference performed by building information matrix and vector (linearized form)
- Map recovered by reduction to robot poses, followed by conversion to moment representation, followed by estimation of landmark positions
- ML estimate by minimization of $J_{GraphSLAM}$
- Data association by iterative greedy search

Three Main Paradigms

Kalman filter	Particle filter	Graph-based

Particle Filters

- Represent belief by random samples
- Sampling Importance Resampling (SIR) principle
  - Draw the new generation of particles
  - Assign an importance weight to each particle
  - Resampling
- Applications are localization, tracking, …

Particle Filter Algorithm

1. Sample the particles from the proposal distribution

$$x_t^{[j]} \sim \pi(x_t \mid \ldots)$$

2. Compute the importance weights

$$w_t^{[j]} = \frac{target(x_t^{[j]})}{proposal(x_t^{[j]})}$$

1. Resampling: Draw sample i with probability $w_t^{[i]}$ and repeat J times

Courtesy: C. Stachniss

Particle Filters for SLAM

- Localization: state space is $<x, y, \theta>$
- SLAM: state space is $<x, y, \theta, map>$
  - For grid maps: $<c_{11}, c_{12}, \ldots, c_{1n}, c_{21}, \ldots, c_{nm}>$
  - For feature maps: $<l_1, l_2, \ldots, l_m>$
- 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 the 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.

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

$$x_{1:t}\,, m$$

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

$$\underline{x_{1:t}}\,,\ \underline{m}$$

Key Idea

$$\underline{x_{1:t}}\ ,\ \underline{m}$$

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

Rao-Blackwellization

- 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

Courtesy: C. Stachniss

Rao-Blackwellization for SLAM

- Factorization of the SLAM posterior

poses map observations & controls

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

First introduced for SLAM by Murphy in 1999

K. Murphy, Bayesian map learning in dynamic environments, In Proc. Advances in Neural Information Processing Systems, 1999

Courtesy: C. Stachniss

Rao-Blackwellization for SLAM

- Factorization of the SLAM posterior

poses map observations & controls

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

path posterior map posterior

First introduced for SLAM by Murphy in 1999

K. Murphy, Bayesian map learning in dynamic environments, In Proc. Advances in Neural Information Processing Systems, 1999

Courtesy: C. Stachniss

Rao-Blackwellization for SLAM

- Factorization of the SLAM posterior

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

Grid cells are conditionally independent given the poses

First exploited in FastSLAM by Montemerlo et al., 2002

Courtesy: C. Stachniss

Rao-Blackwellization for SLAM

- Factorization of the SLAM posterior

$$p(x_{1:t}, m \mid z_{1:t}, u_{1:t}) =$$
$$p(x_{1:t} \mid z_{1:t}, u_{1:t})\, p(m \mid x_{1:t}, z_{1:t})$$
$$p(x_{1:t} \mid z_{1:t}, u_{1:t}) \underline{\prod_{i=1}^{N} p(m^i \mid x_{1:t}, z_{1:t})}$$

Grid cells are conditionally independent given the poses

First exploited in FastSLAM by Montemerlo et al., 2002

Courtesy: C. Stachniss

Rao-Blackwellization for SLAM

- Factorization of the SLAM posterior

$$
\begin{aligned}
p(x_{1:t}, m \mid z_{1:t}, u_{1:t}) = \\
p(x_{1:t} \mid z_{1:t}, u_{1:t})\, p(m \mid x_{1:t}, z_{1:t}) \\
\underbrace{p(x_{1:t} \mid z_{1:t}, u_{1:t})}_{\text{particle filter for localization}}\, \underbrace{\prod_{i=1}^{N} p\left(m^i \mid x_{1:t}, z_{1:t}\right)}_{\text{occupancy grid mapping}}
\end{aligned}
$$

First exploited in FastSLAM by Montemerlo et al., 2002

Courtesy: C. Stachniss

Modeling the Robot's Path

- Sample-based representation for
  $p(x_{1:t} \mid z_{1:t}, u_{1:t})$
- Each sample is a path hypothesis

$$x_1 \qquad x_2 \qquad x_3 \qquad \ldots$$

starting location,
typically (0,0,0)

pose hypothesis
at time *t*=2

- Past poses of a sample are not revised
- No need to maintain past poses in the sample set

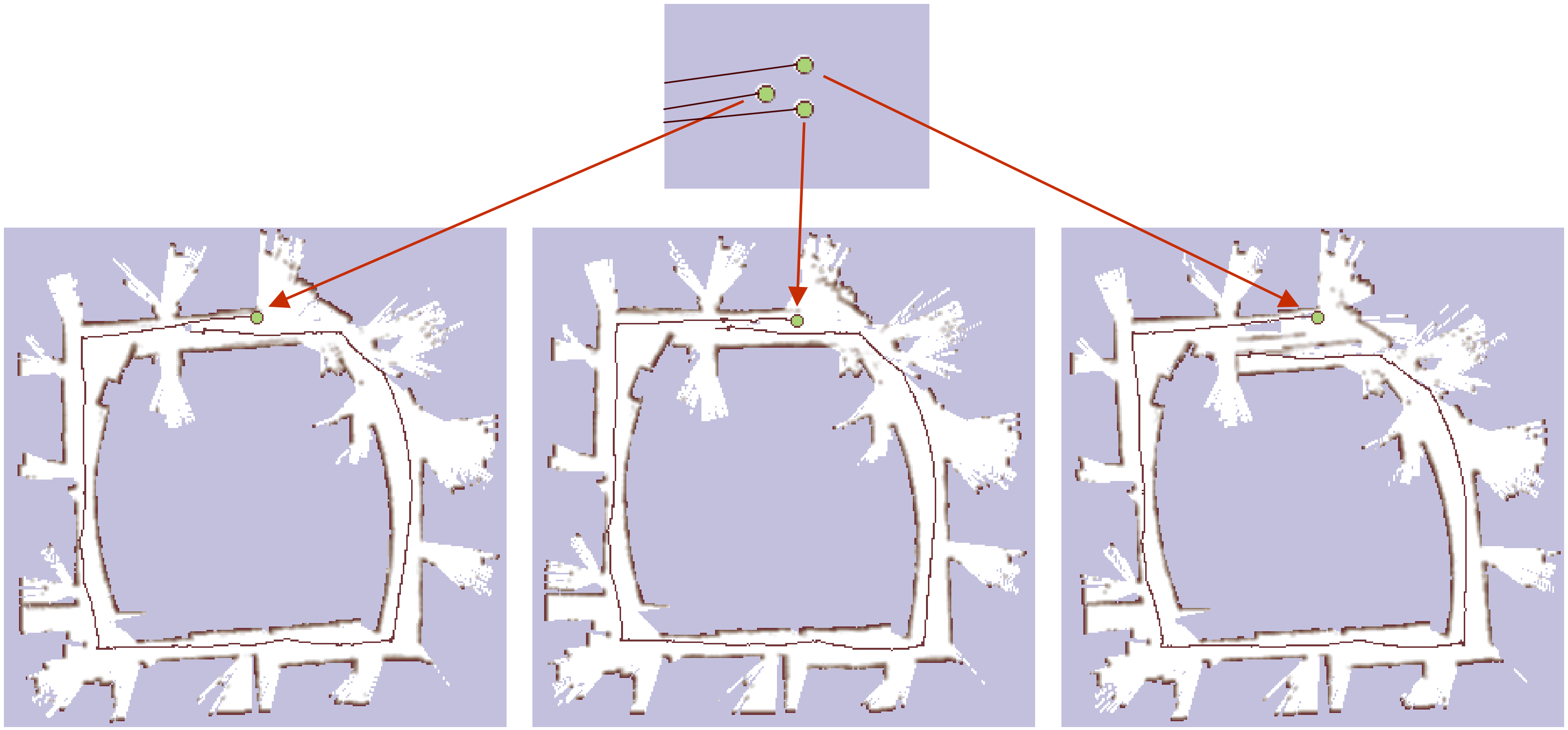
Courtesy: C. Stachniss

FastSLAM

- Proposed by Montemerlo et al. in 2002 (for landmark based SLAM)
- Each particle has a pose and a map

Particle 1	x, y, θ	**Occupancy grid map**
Particle 2	x, y, θ	**Occupancy grid map**
$\vdots$		
Particle *N*	x, y, θ	**Occupancy grid map**

FastSLAM – Particle representation



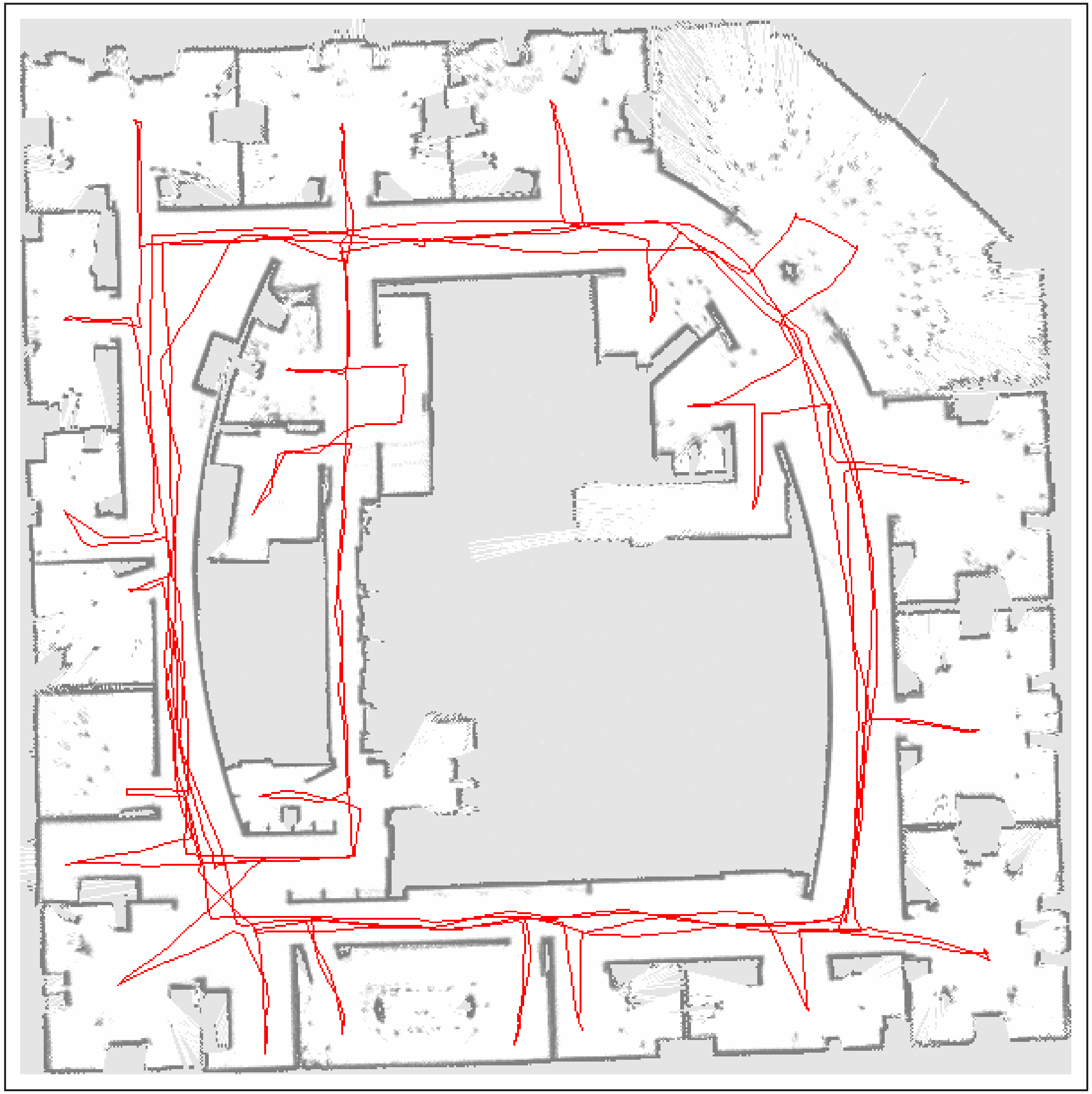
FastSLAM Algorithm

Algorithm FastSLAM_occupancy_grids($\mathcal{X}_{t-1}, u_t, z_t$**):**

$\bar{\mathcal{X}}_t = \mathcal{X}_t = \emptyset$
for $k = 1$ *to* M *do*
$x_t^{[k]} = \textbf{sample\_motion\_model}(u_t, x_{t-1}^{[k]})$
$w_t^{[k]} = \textbf{measurement\_model\_map}(z_t, x_t^{[k]}, m_{t-1}^{[k]})$
$m_t^{[k]} = \textbf{updated\_occupancy\_grid}(z_t, x_t^{[k]}, m_{t-1}^{[k]})$
$\bar{\mathcal{X}}_t = \bar{\mathcal{X}}_t + \langle x_t^{[k]}, m_t^{[k]}, w_t^{[k]} \rangle$
endfor
for $k = 1$ *to* M *do*
draw i *with probability* $\propto w_t^{[i]}$
add $\langle x_t^{[i]}, m_t^{[i]} \rangle$ *to* $\mathcal{X}_t$
endfor
return $\mathcal{X}_t$

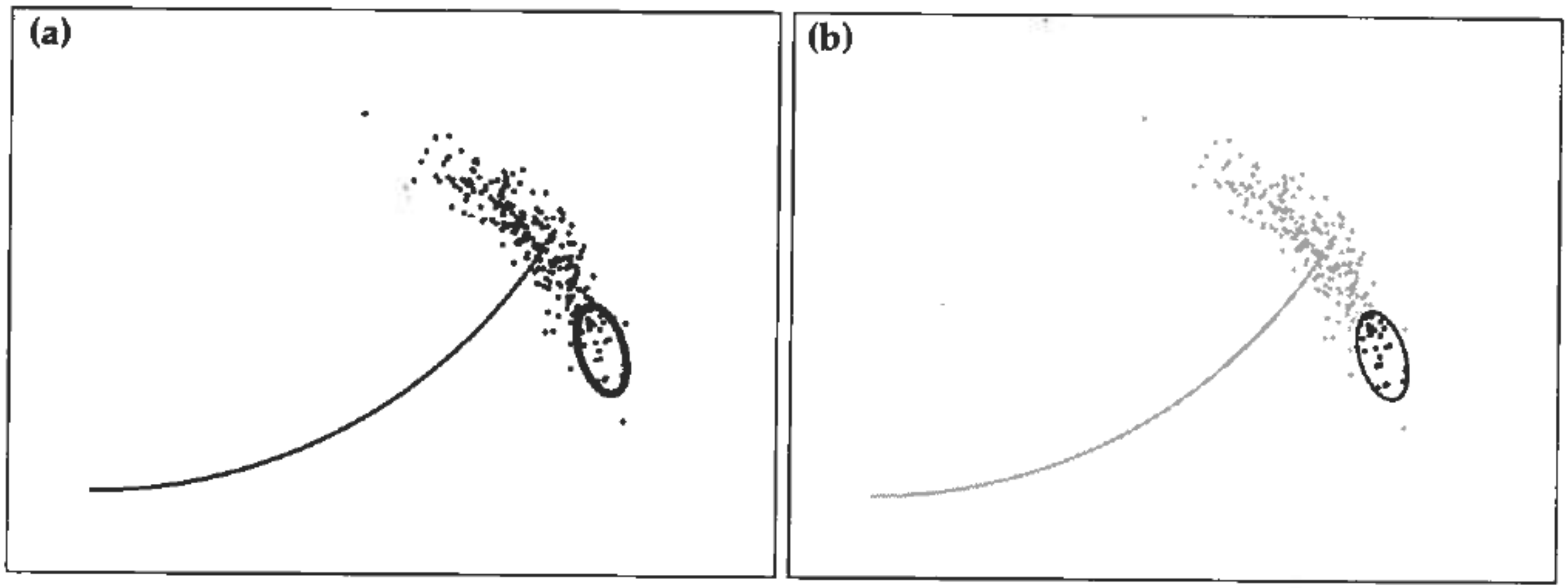
Pure odometry

FastSLAM – Best particle



Weakness of FastSLAM 1.0

- Proposal Distribution
- Importance weighting



FastSLAM 1.0 to FastSLAM 2.0

- FastSLAM 1.0 uses the motion model as the proposal distribution

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

- FastSLAM 2.0 **considers also the measurements during sampling**
- Especially useful if an accurate sensor is used (compared to the motion noise)

[Montemerlo et al., 2003]

Courtesy: C. Stachniss

FastSLAM 2.0 (Informally)

- FastSLAM 2.0 samples from

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

- Results in a more peaked proposal distribution
- Less particles are required
- More robust and accurate
- But more complex…

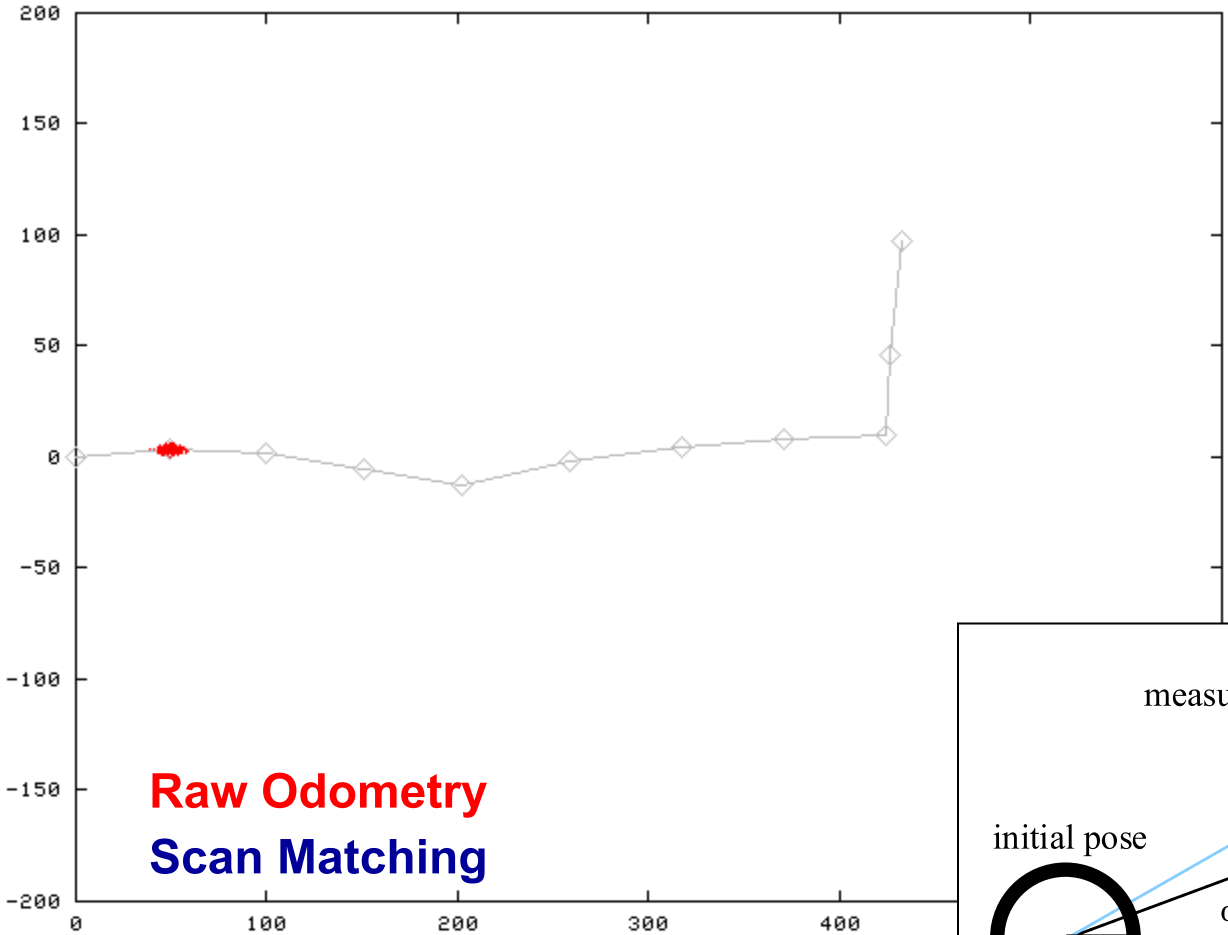
[Montemerlo et al., 2003]

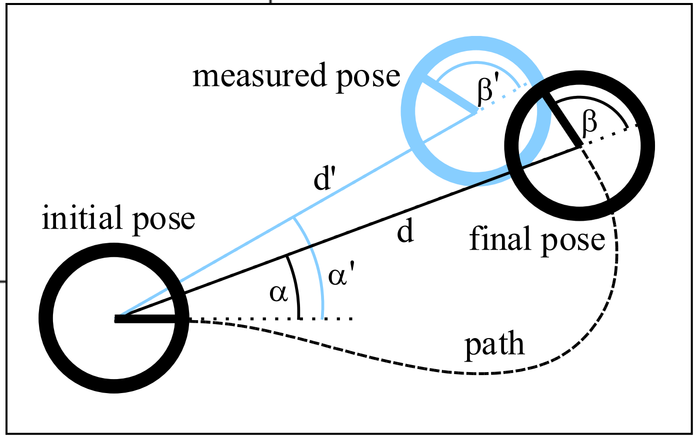
Courtesy: C. Stachniss

Generating better proposals

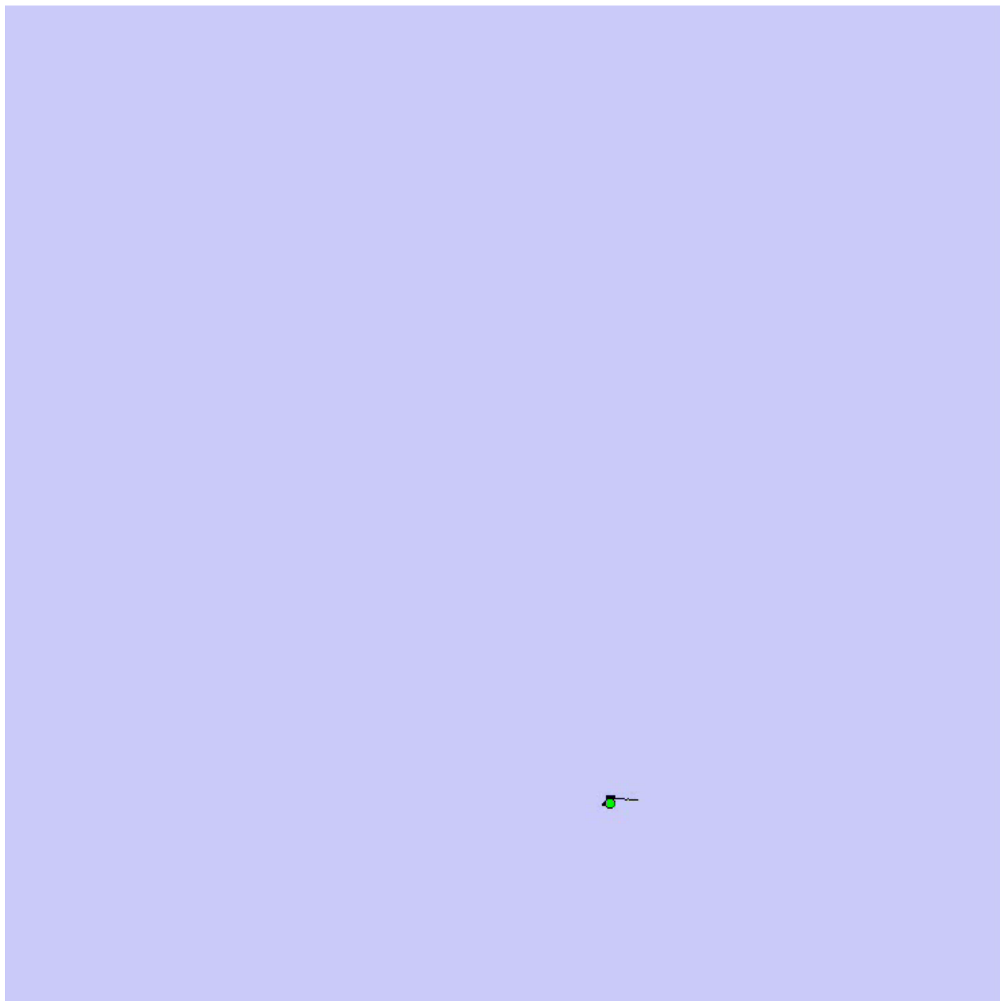
- Use scan-matching to compute highly accurate odometry measurements from consecutive range scans.
- Use the improved odometry in the prediction step to get highly accurate proposal distributions.

Motion Model for Scan Matching





Rao-Blackwellized Mapping with Scan-Matching



Map: Intel Research Lab Seattle

Loop Closure

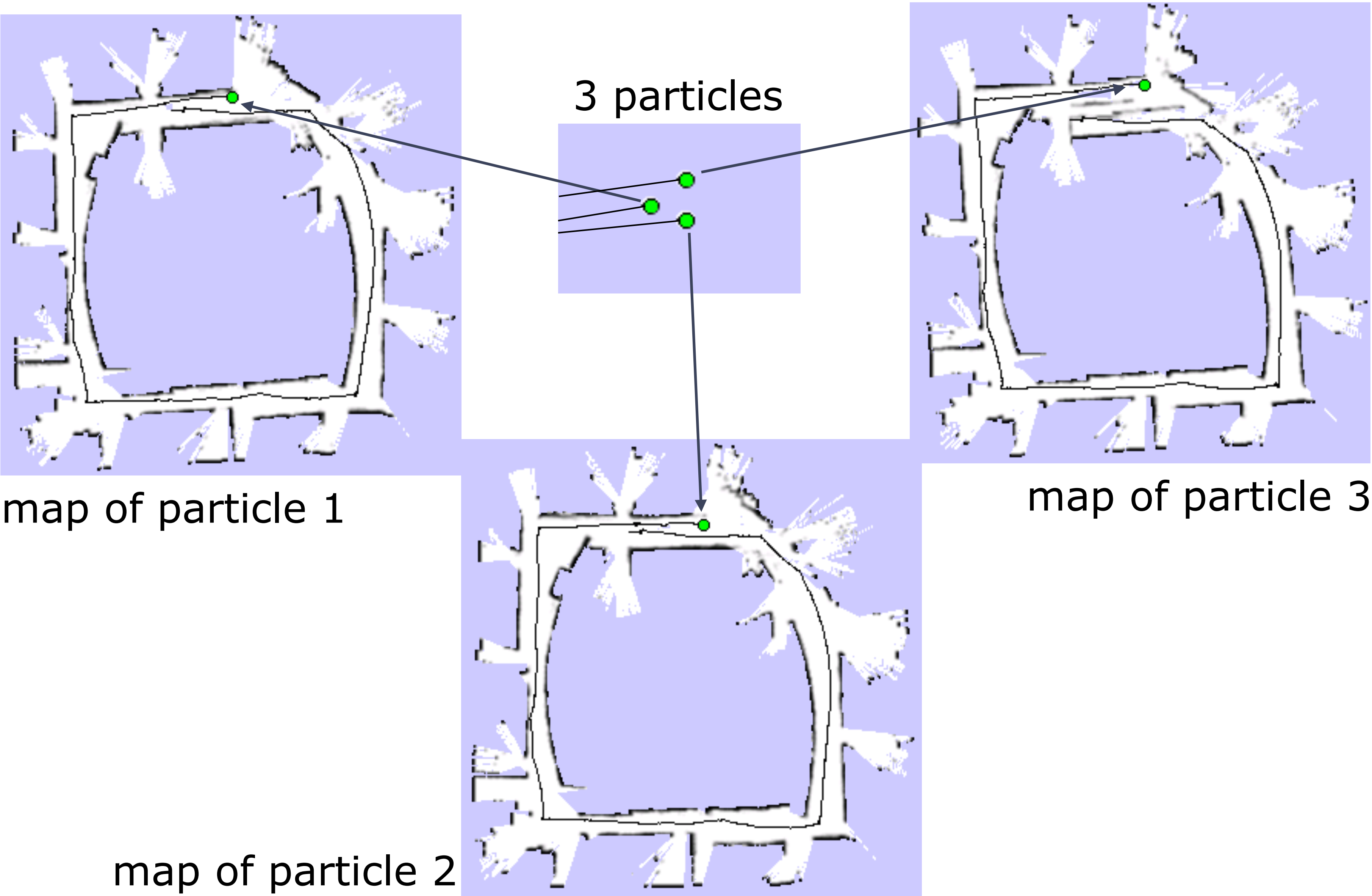
- Loop closure involves
  - Recognizing when the robot has returned to a previously mapped region
  - Using this information to reduce the uncertainty in the map estimate

- Without loop closure, the uncertainty can grow without bounds

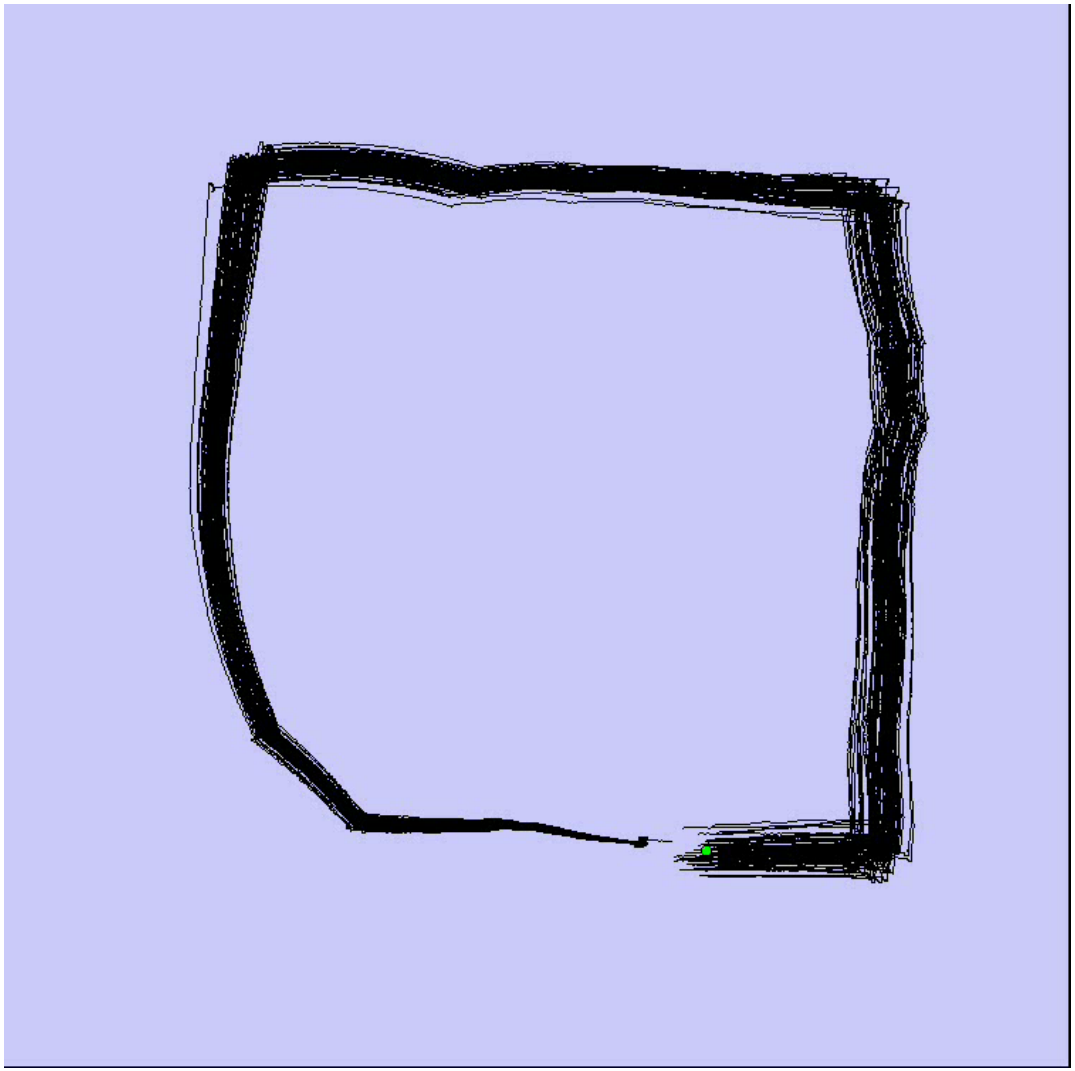
Loop Closure in FastSLAM

- Each particle has it’s own map
- Maps which agree to closing the loop are weighed higher than others
- These maps are more likely to be resampled
- **Key:** Need diversity of paths/particles/maps

Loop Closure Example

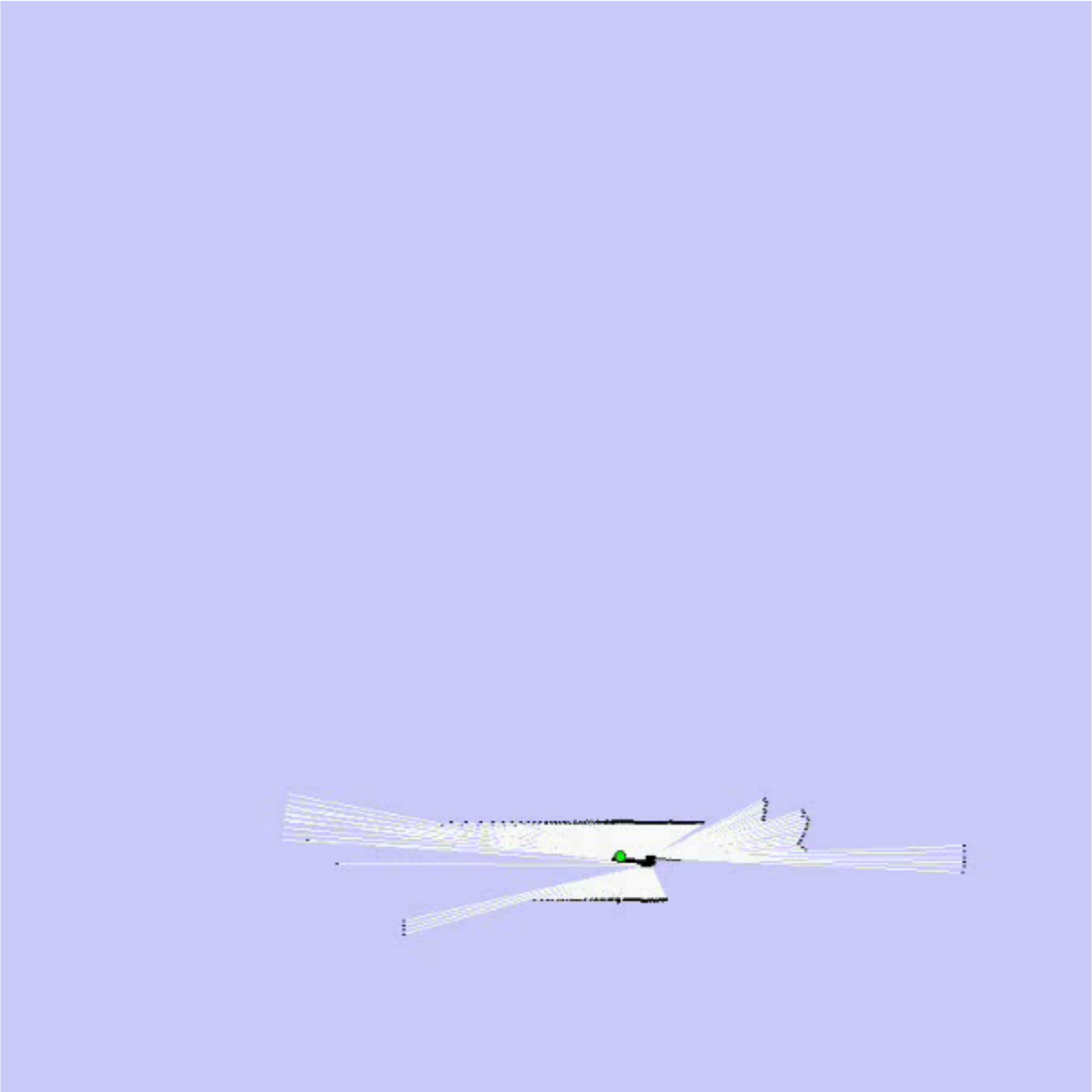


Rao-Blackwellized Mapping with Scan-Matching



Map: Intel Research Lab Seattle

Rao-Blackwellized Mapping with Scan-Matching



Map: Intel Research Lab Seattle

Example (Intel Lab)

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

Work by Grisetti et al.

Outdoor Campus Map

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

Work by Grisetti et al.

FastSLAM Summary

- Particle filter-based SLAM
- Rao-Blackwellization: model the robot's path by sampling and compute the landmarks given the poses
- Allow for per-particle data association
- FastSLAM 1.0 and 2.0 differ in the proposal distribution
- Complexity $\mathcal{O}(N \log M)$

Courtesy: C. Stachniss

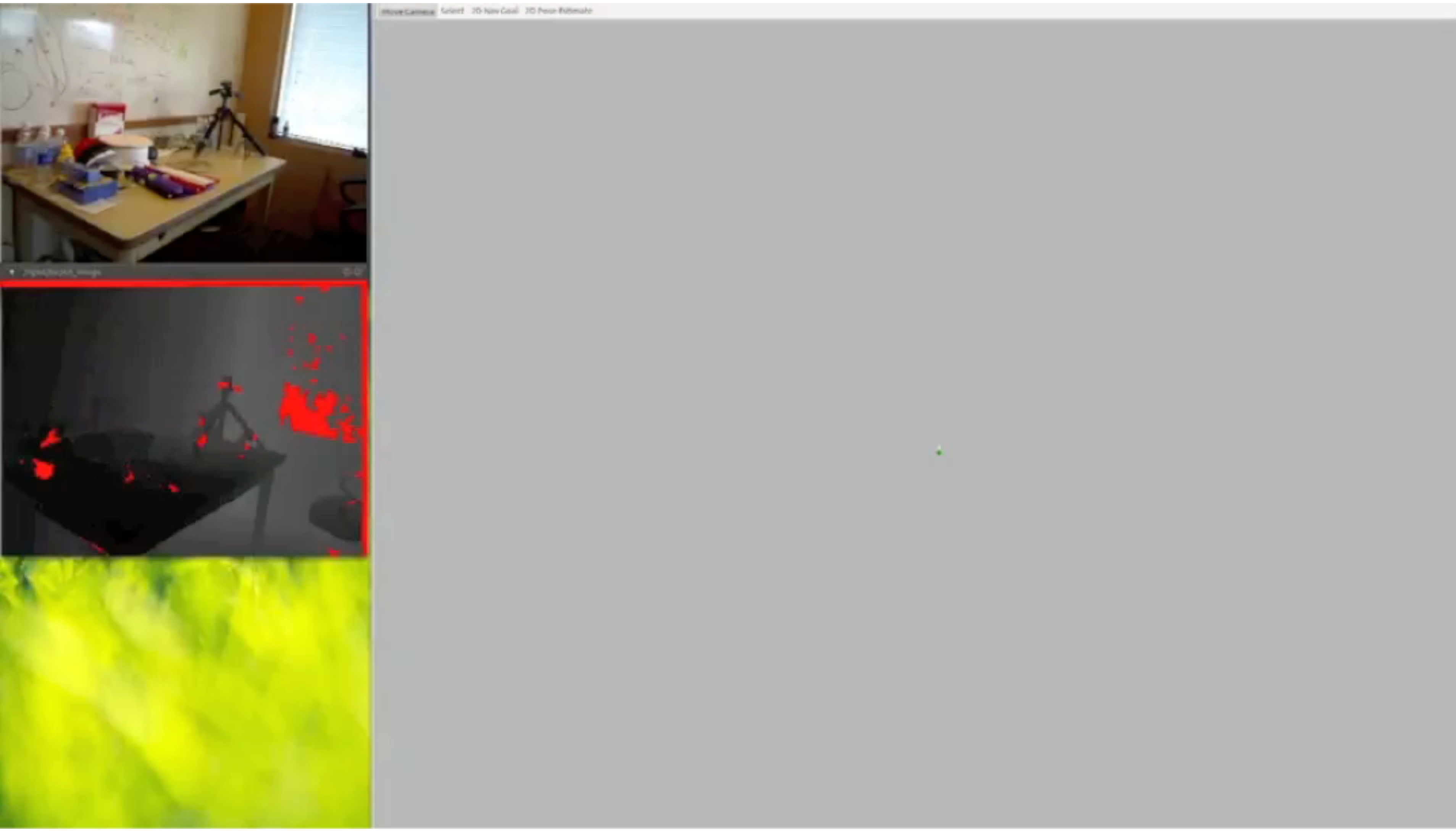
Literature

FastSLAM

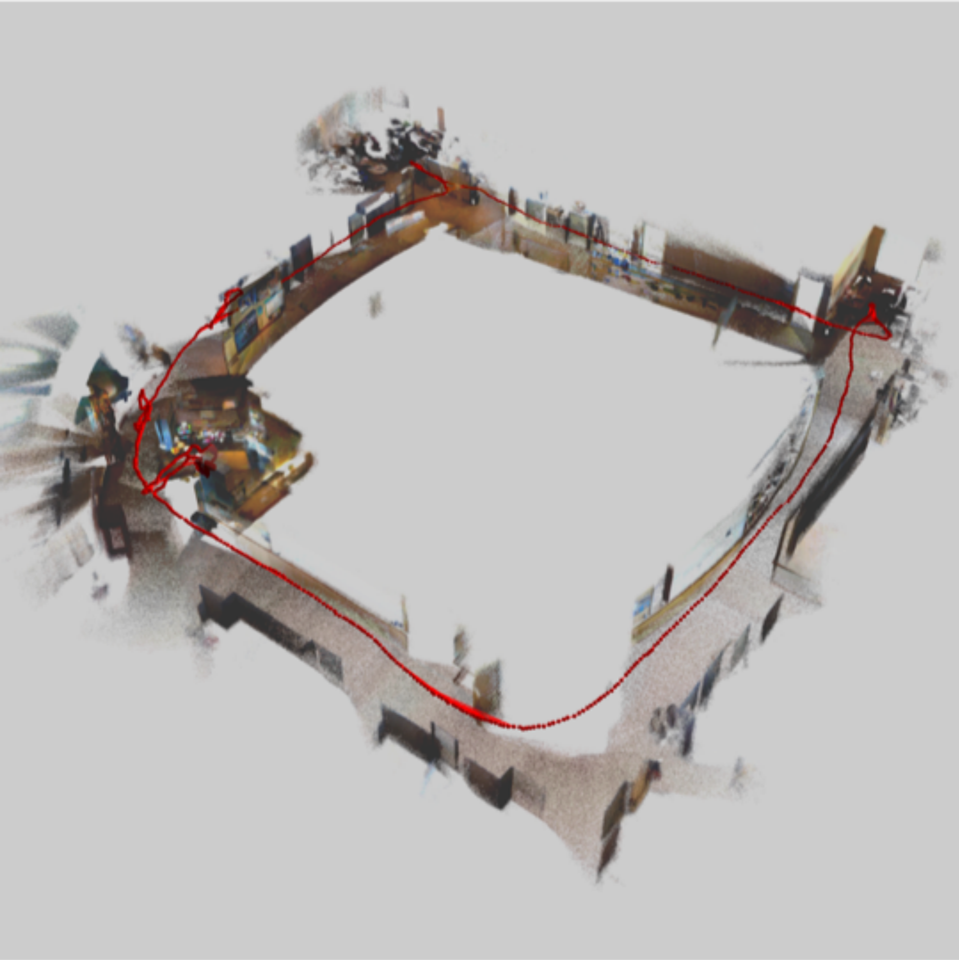
- Thrun et al.: “Probabilistic Robotics”, Chapter 13.1-13.3 + 13.8 (see errata!)
- Montemerlo, Thrun, Kollar, Wegbreit: FastSLAM: A Factored Solution to the Simultaneous Localization and Mapping Problem, 2002
- Montemerlo and Thrun: Simultaneous Localization and Mapping with Unknown Data Association Using FastSLAM, 2003

Courtesy: C. Stachniss

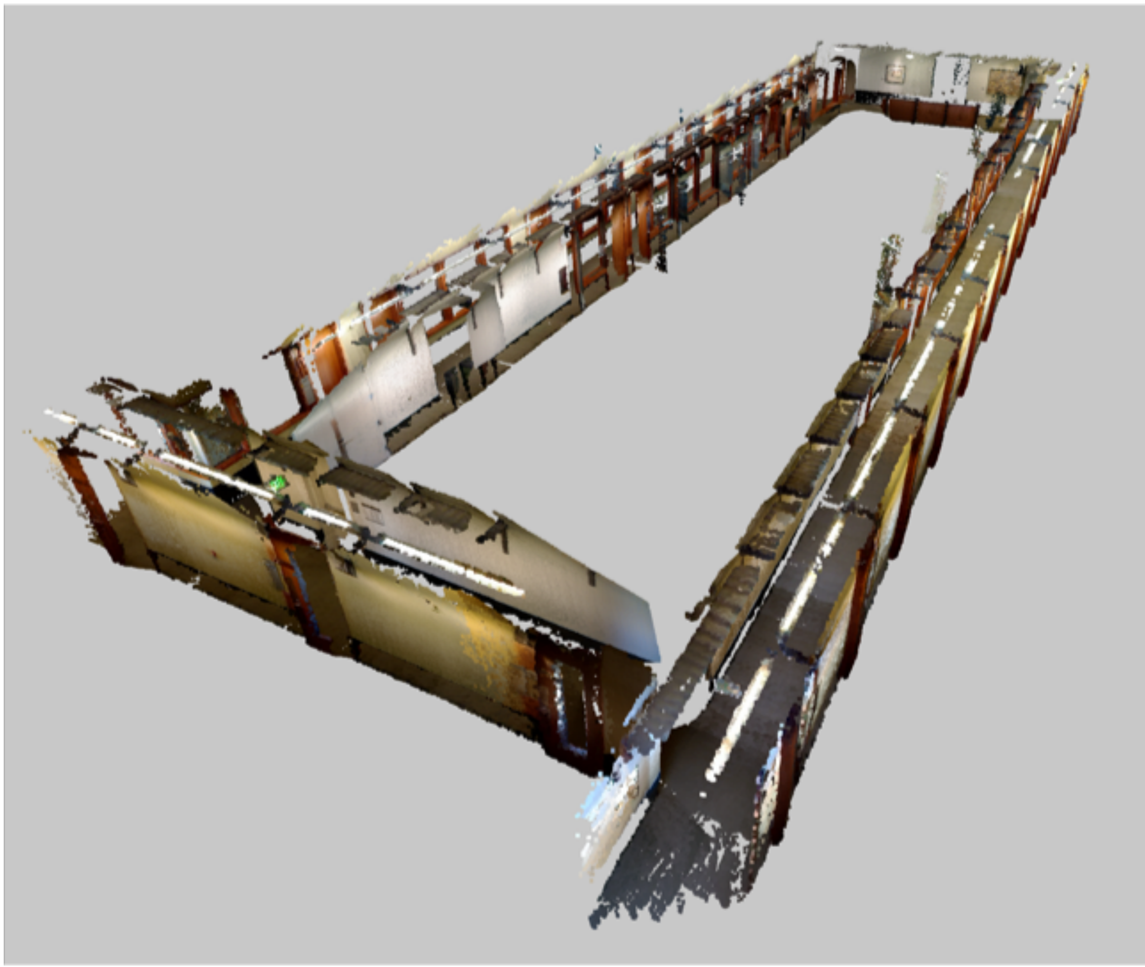
RGBD SLAM

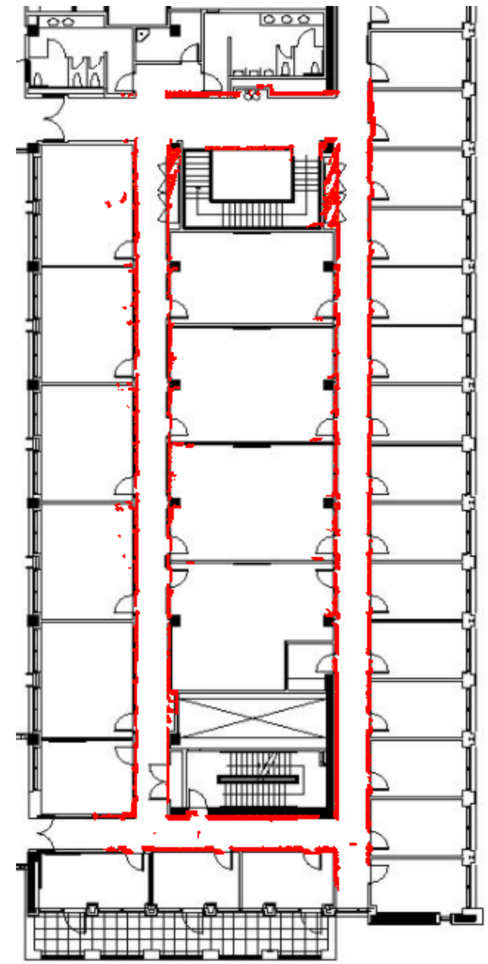


Resulting Map



Experiments: Overlay 1





Experiments: Overlay 2