CSE-490R
Robotics

SLAM + Fast-SLAM

(Partial slides borrowed from Dieter Fox, Wolfram Burgard & Cyril Stachniss)

What is SLAM?

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

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

- Mapping: Build map given poses and observations

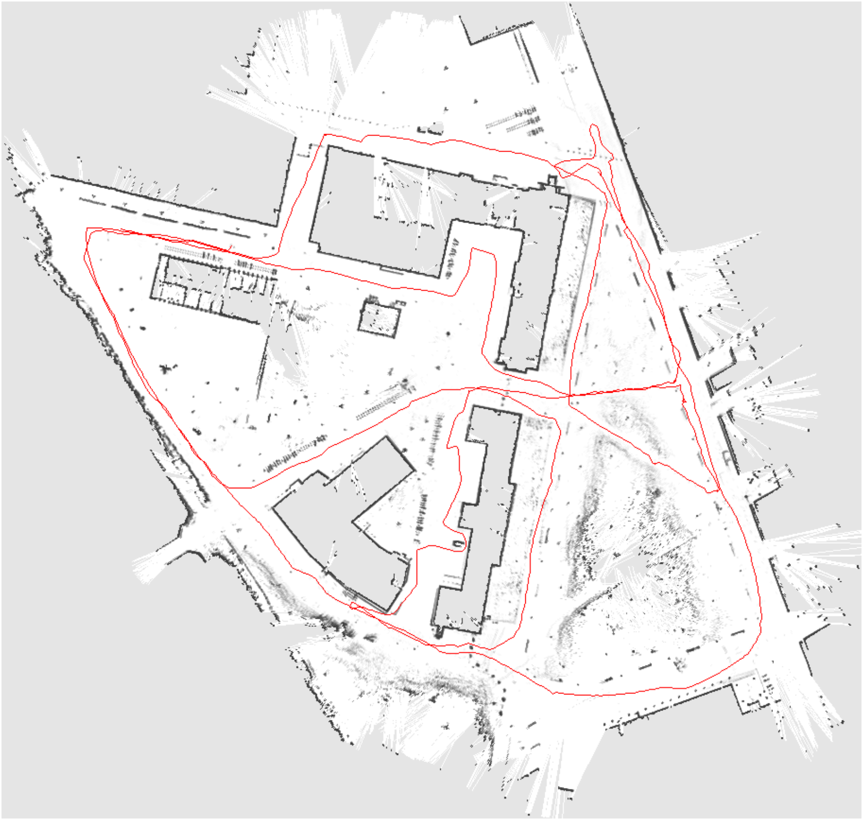
$$p(m \mid x_{1:t}, z_{1:t})$$

- Simultaneous Localization And Mapping (SLAM): Find poses and map given controls and observations

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

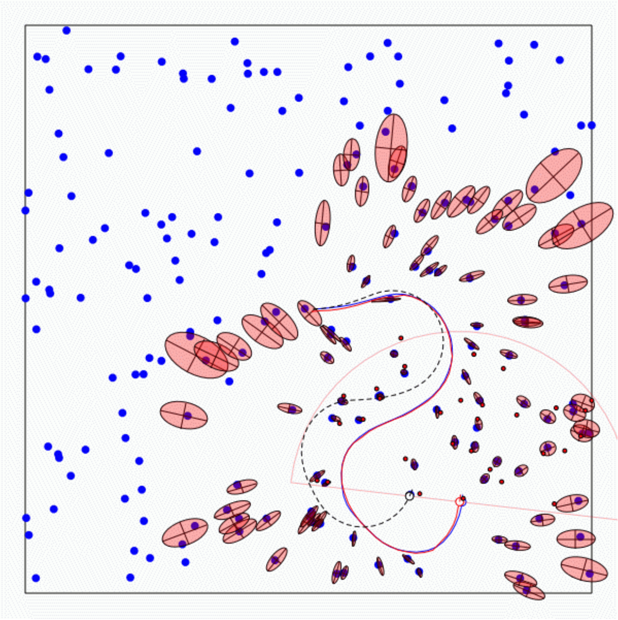
SLAM

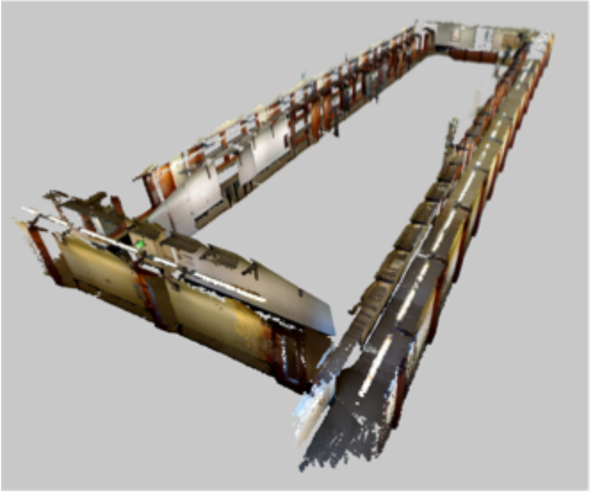
- A robot moving though an unknown, static environment
- **Given:**
  - The robot's controls
  - Observations of nearby features
- **Estimate:**
  - Map of environment
  - Path of the robot



Map representations

- Typical representations are:
  - Feature-based
  - Grid maps (occupancy maps)
  - 3D representations (voxels, surfels, octrees etc)





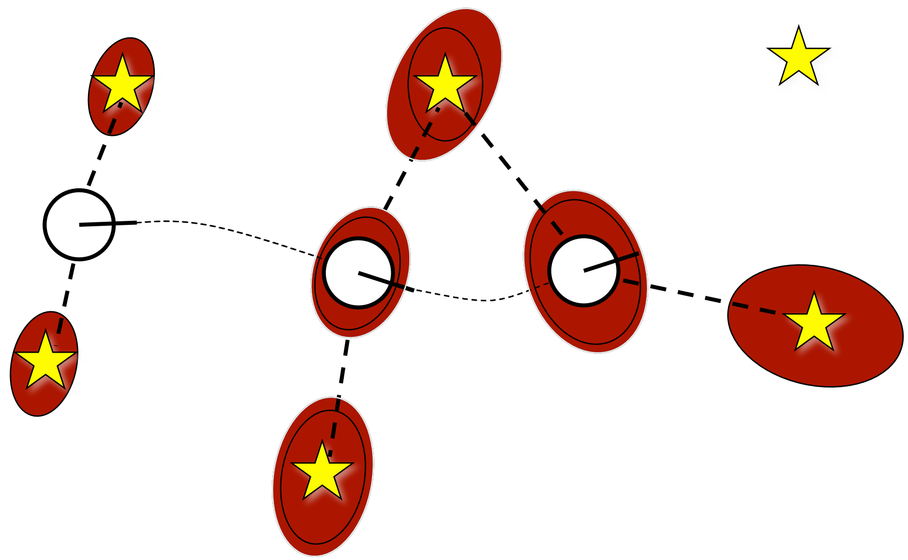
Why is SLAM hard?

- SLAM is the task of building a map while estimating the pose of the robot relative to this map

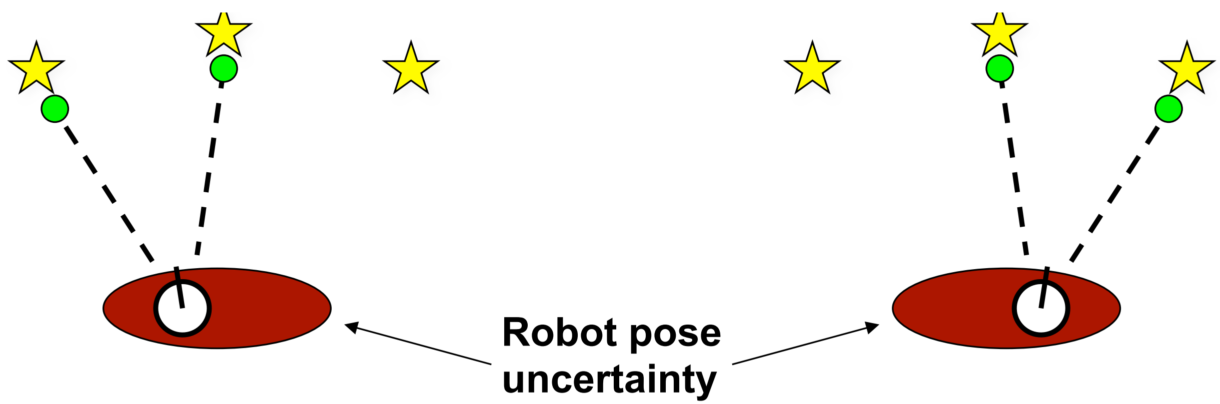
- Chicken-or-egg problem:
  - A map is needed to localize the robot
  - A pose estimate is needed to build a map

Why is SLAM hard?

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



Robot path error correlates errors in the map



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

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

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

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

starting location, typically (0,0,0)

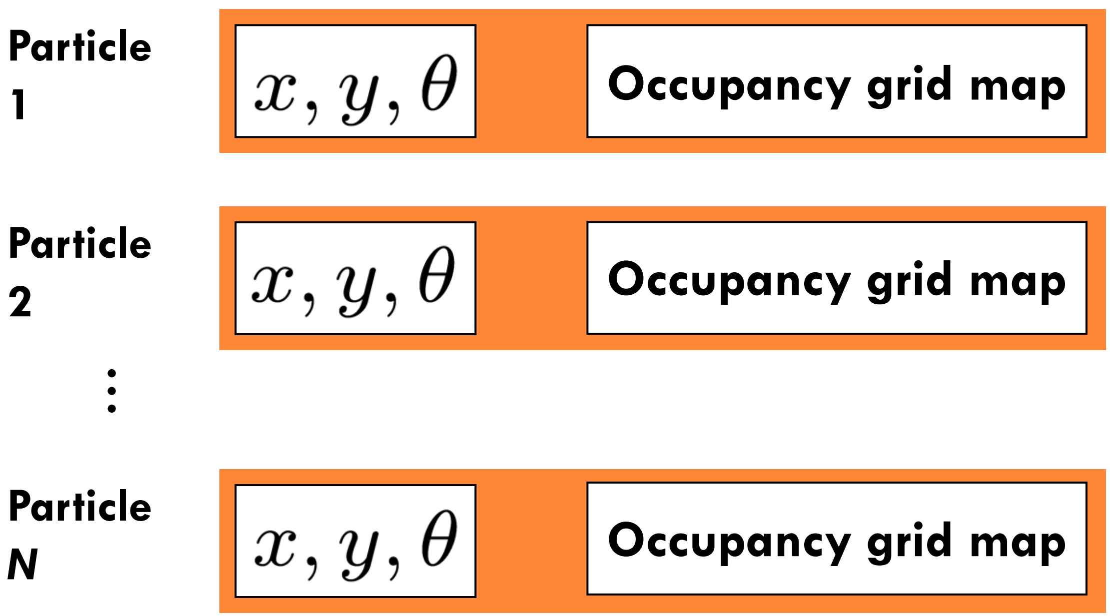
pose hypothesis at time $t=2$

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

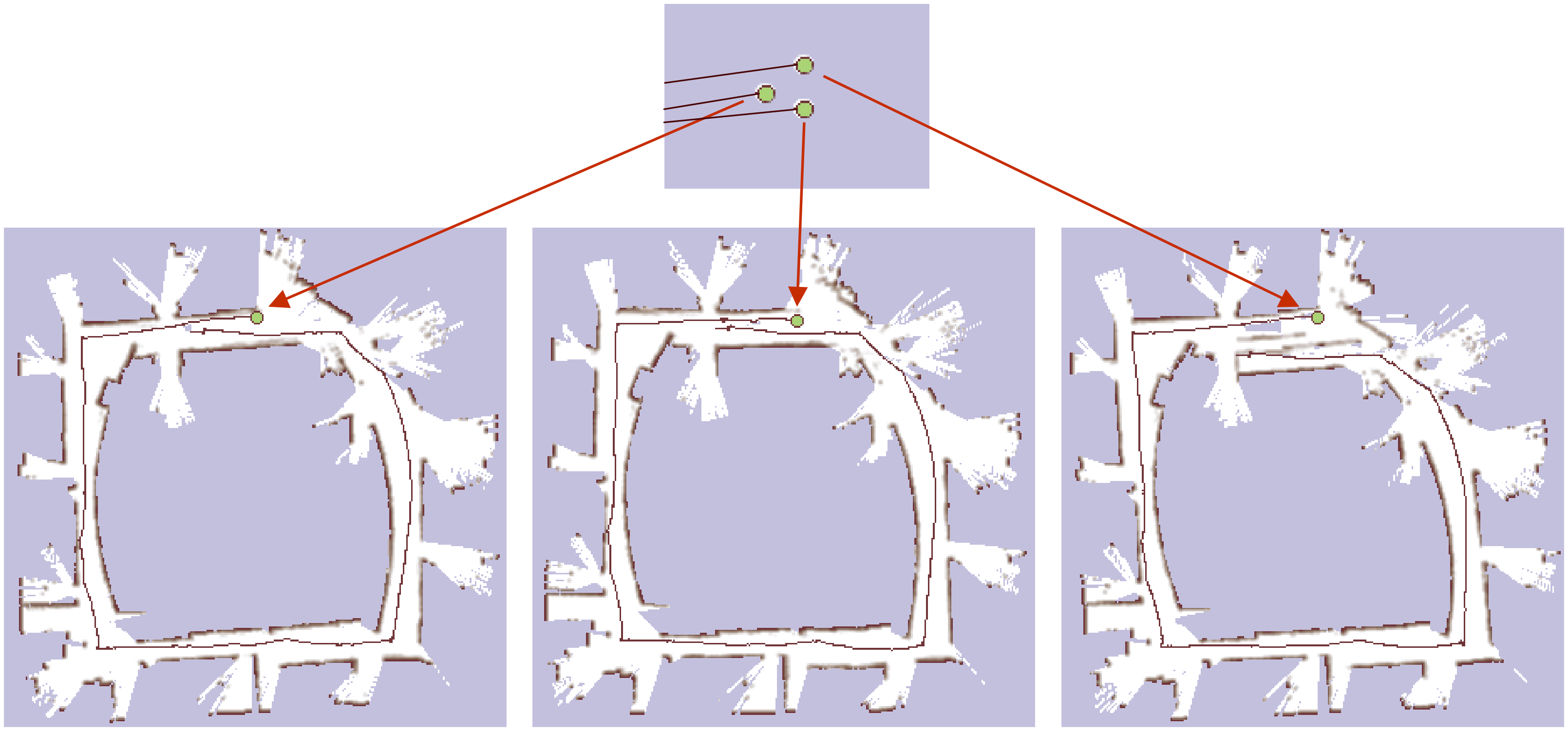
Courtesy: C. Stachniss

FastSLAM

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



FastSLAM – Particle representation



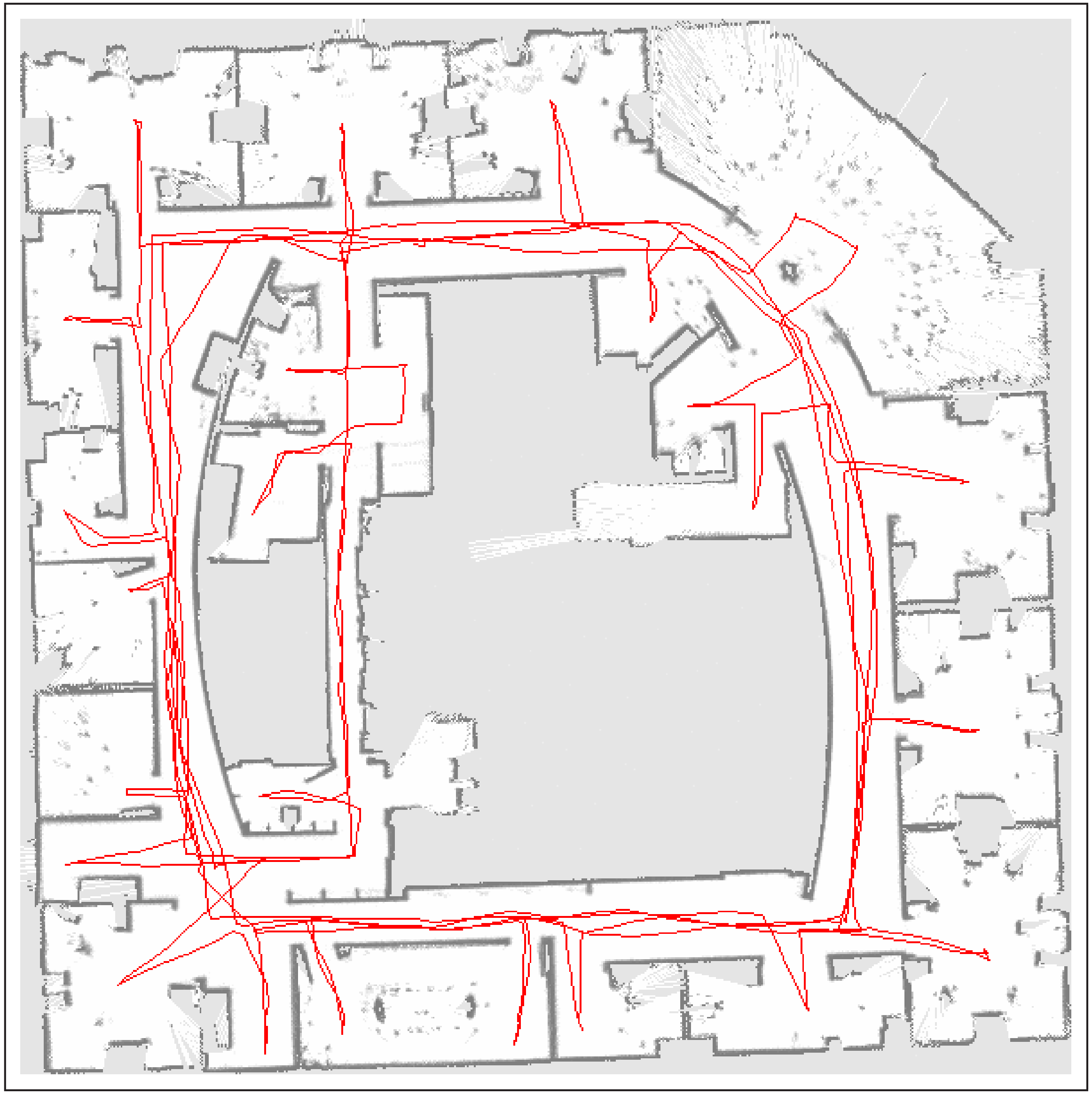
FastSLAM Algorithm

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

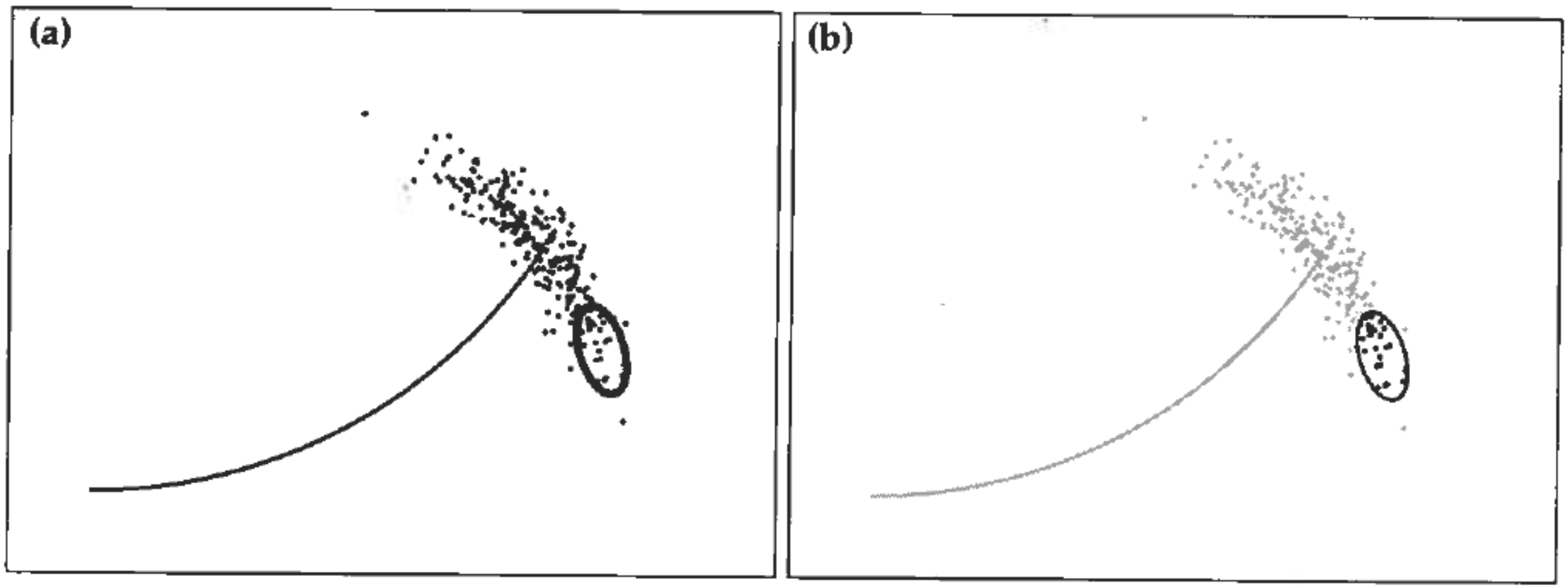
Pure odometry

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…

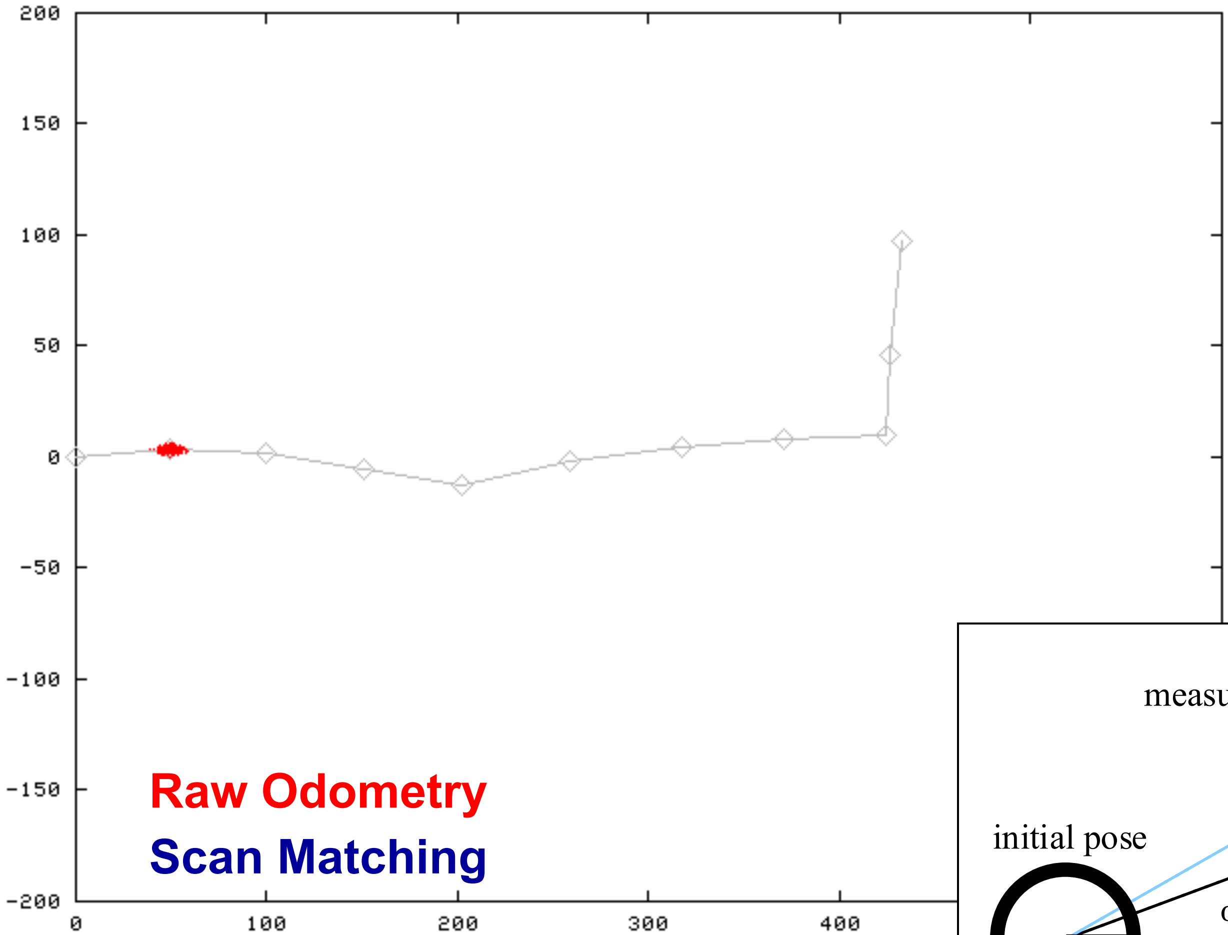
[Montemerlo et al., 2003]

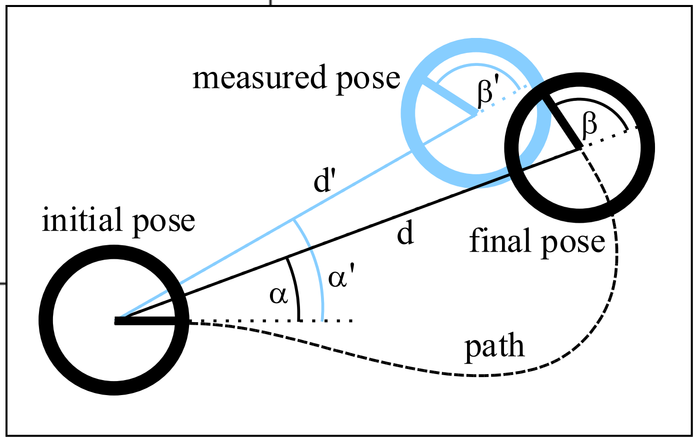
Courtesy: C. Stachniss

Generating better proposals

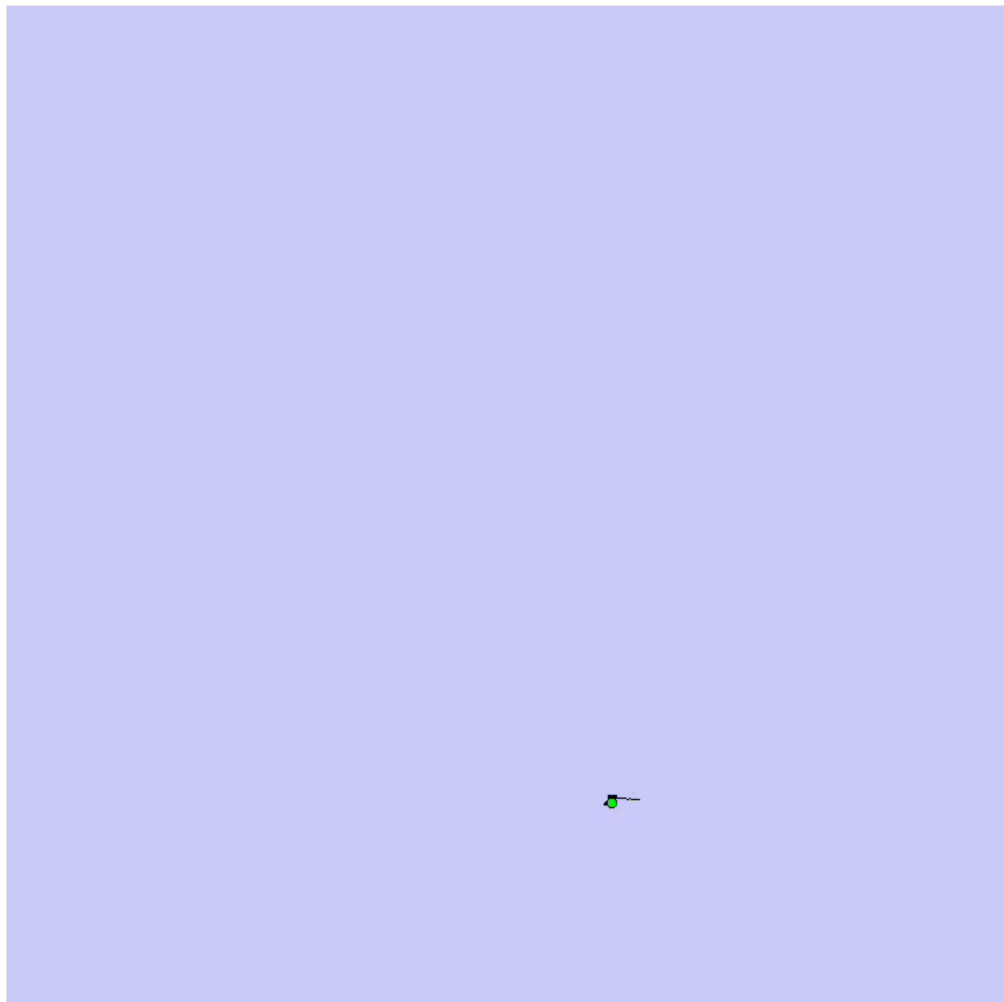
- Use scan-matching to compute highly accurate odometry measurements from consecutive range scans.
- Use the improved odometry in the prediction step to get highly accurate proposal distributions.

Motion Model for Scan Matching





Rao-Blackwellized Mapping with Scan-Matching



Map: Intel Research Lab Seattle

Loop Closure

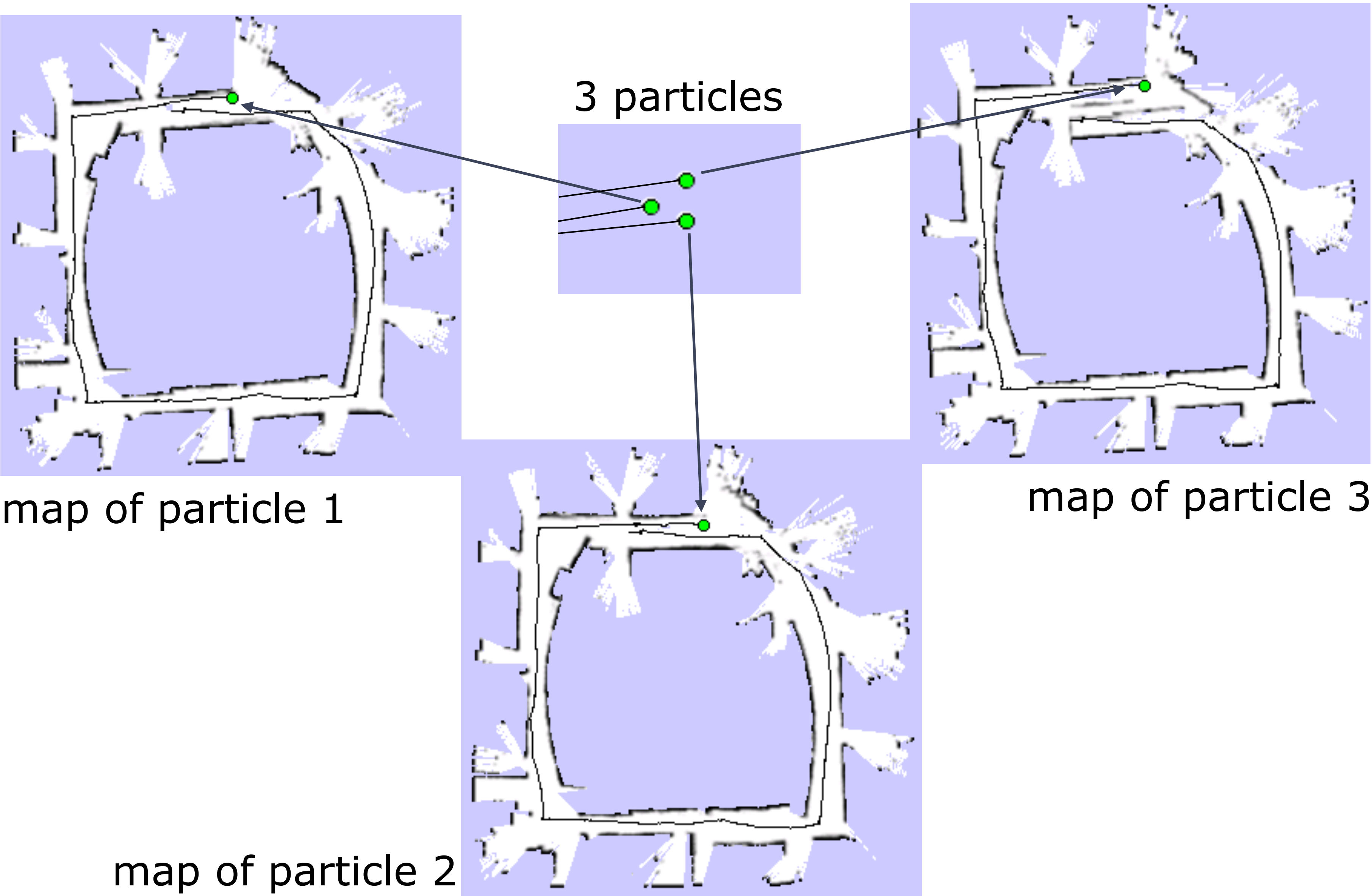
- Loop closure involves
  - Recognizing when the robot has returned to a previously mapped region
  - Using this information to reduce the uncertainty in the map estimate

- Without loop closure, the uncertainty can grow without bounds

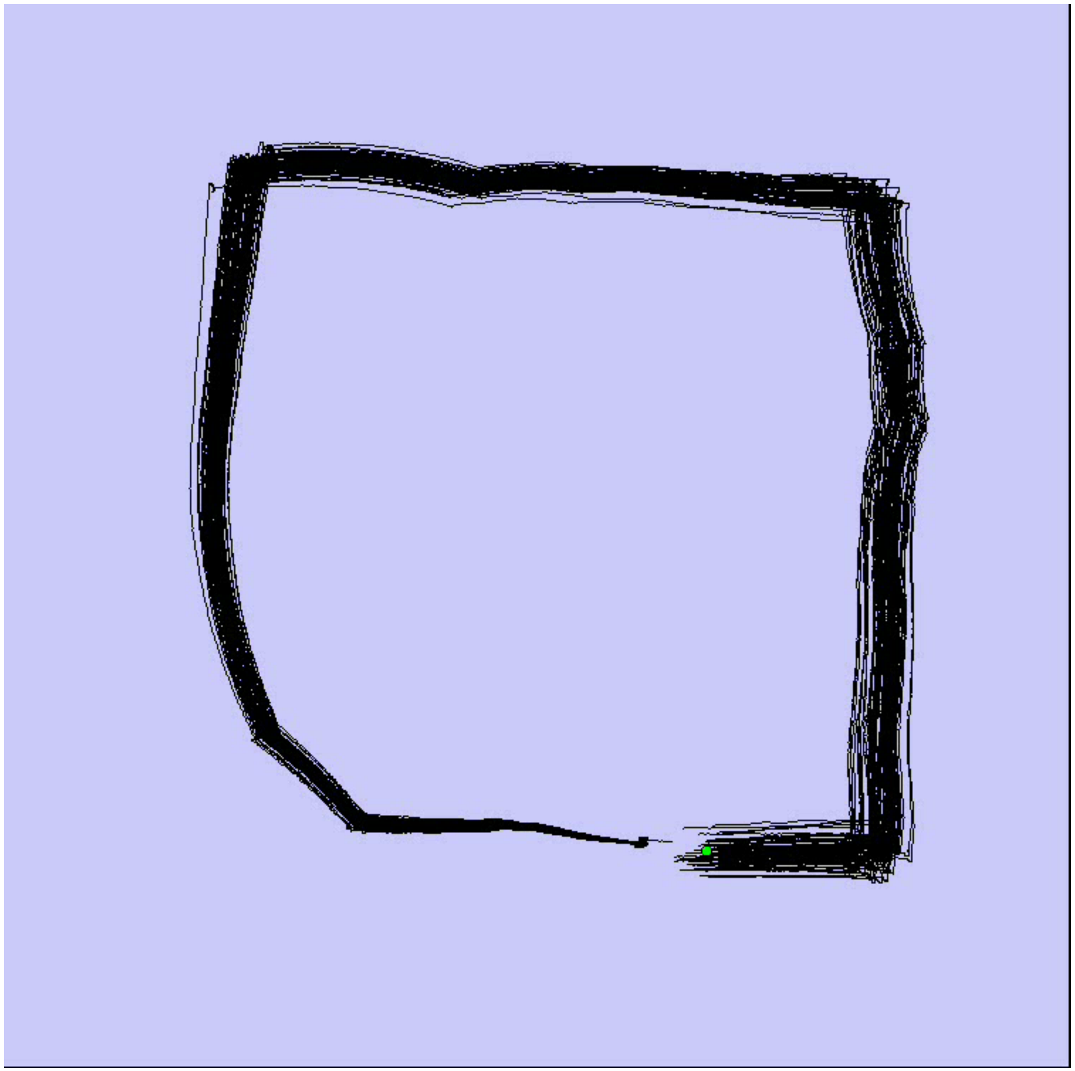
Loop Closure in FastSLAM

- 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

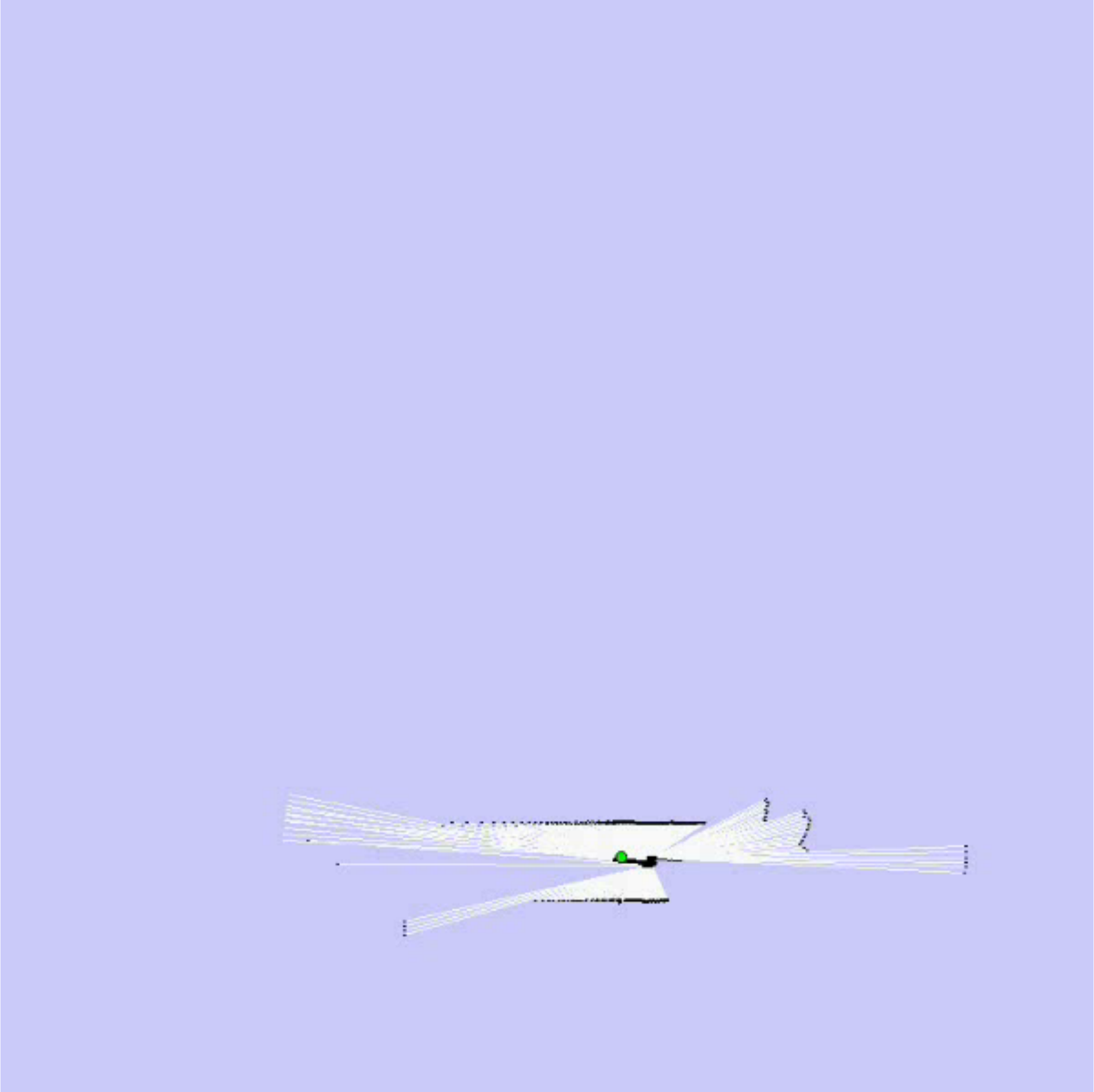


Rao-Blackwellized Mapping with Scan-Matching



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.

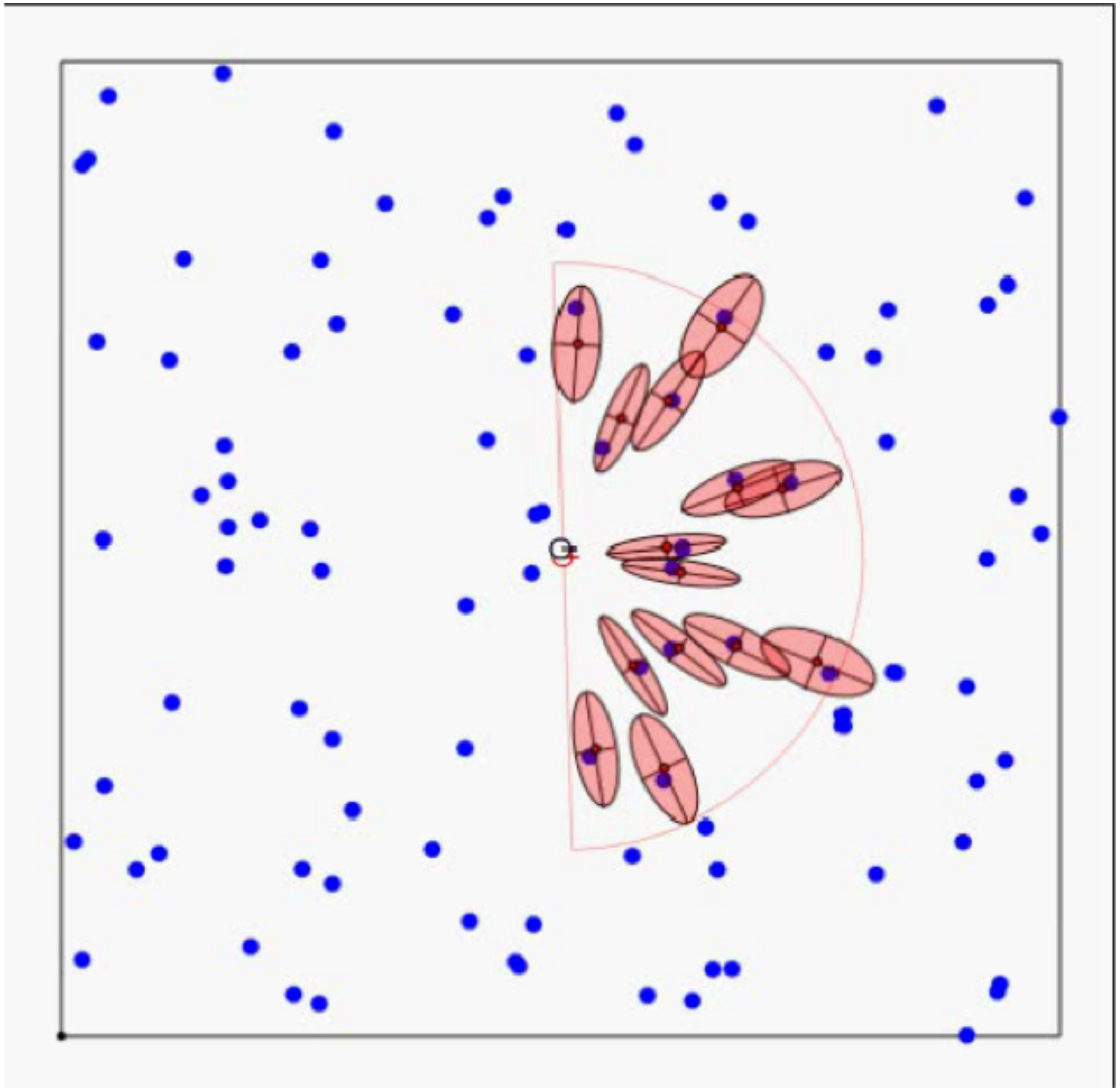
Outdoor Campus Map

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

Work by Grisetti et al.

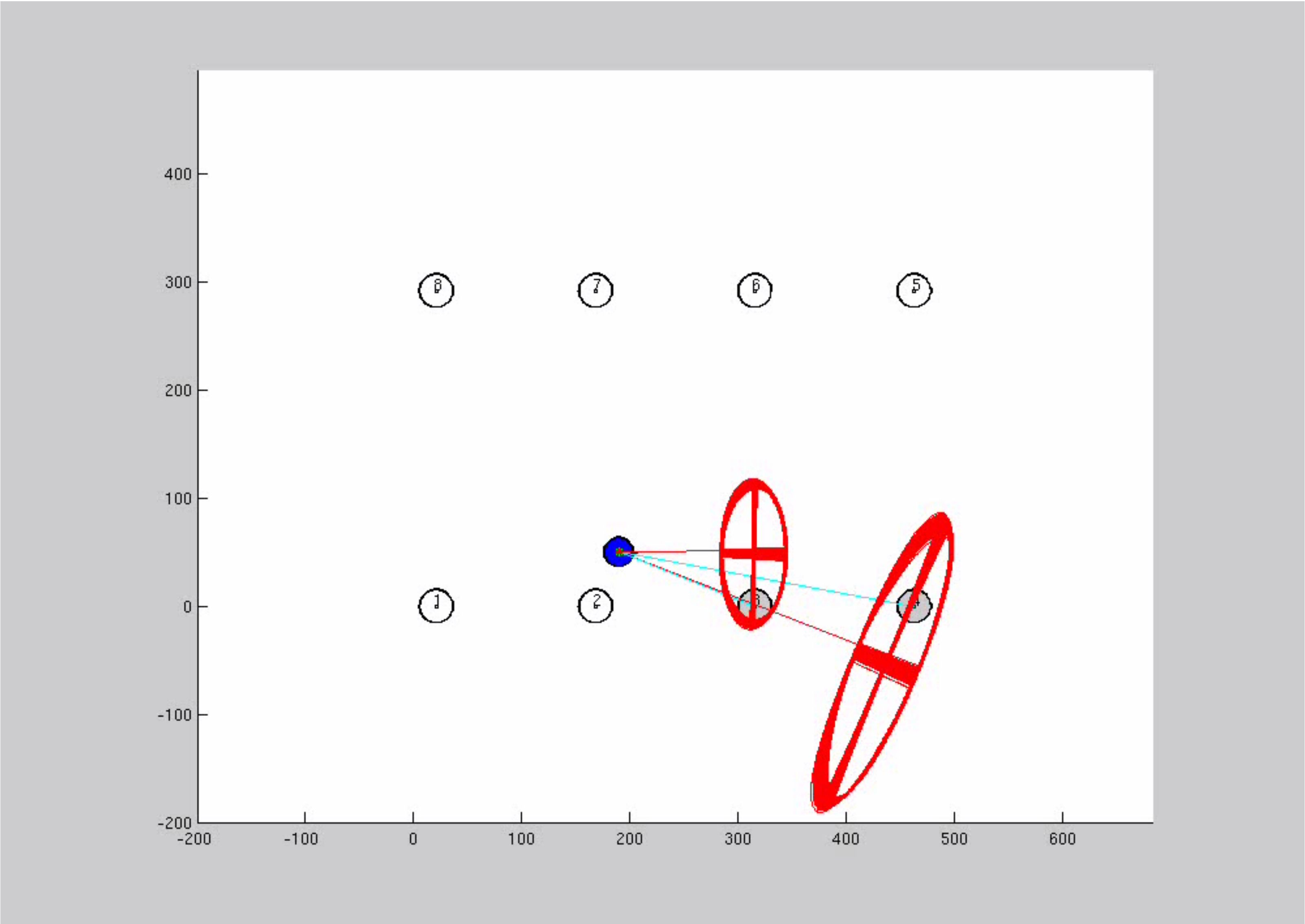
FastSLAM for feature-based maps

FastSLAM in Action



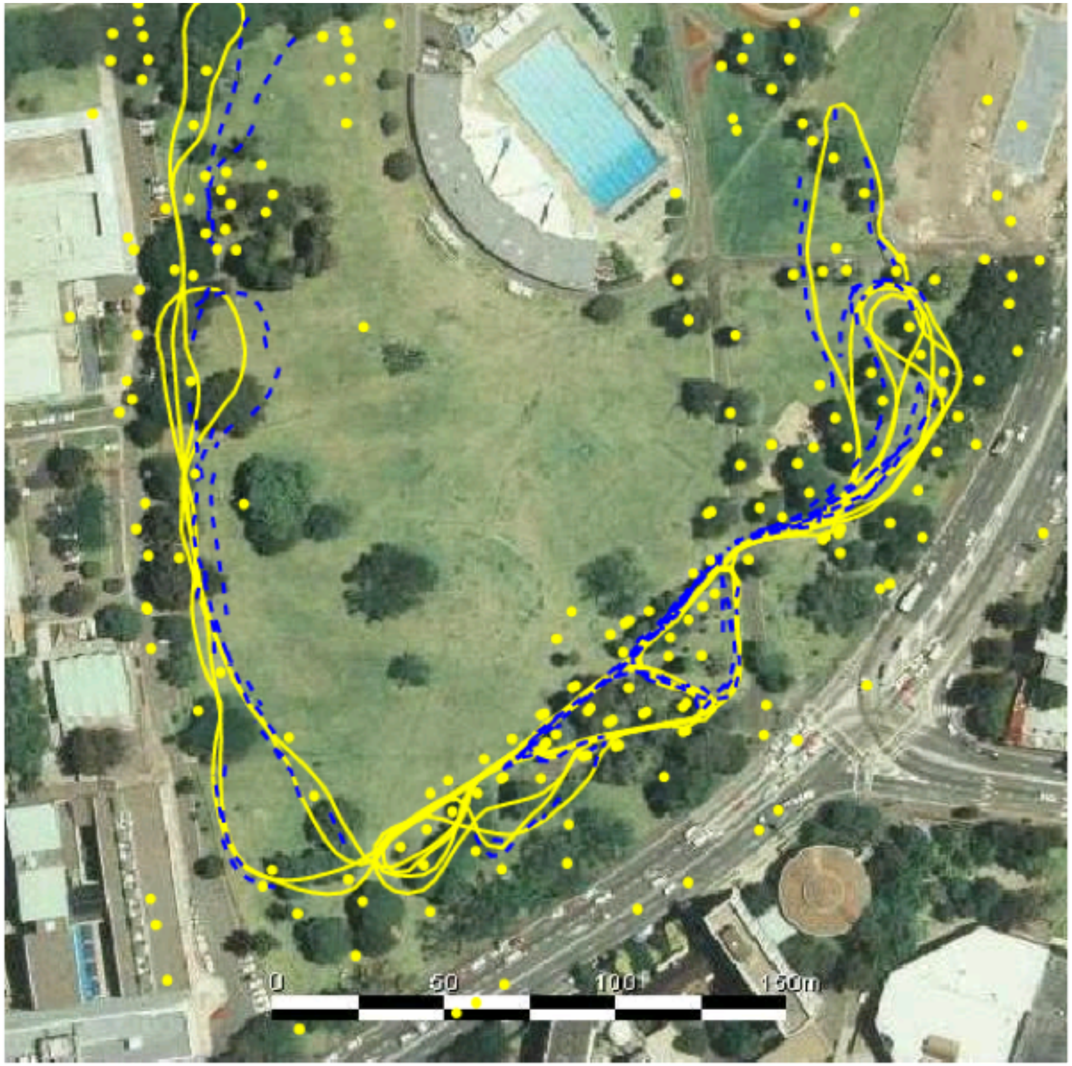
Courtesy: M. Montemerlo

FastSLAM – Video – All Maps



Results – Victoria Park

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



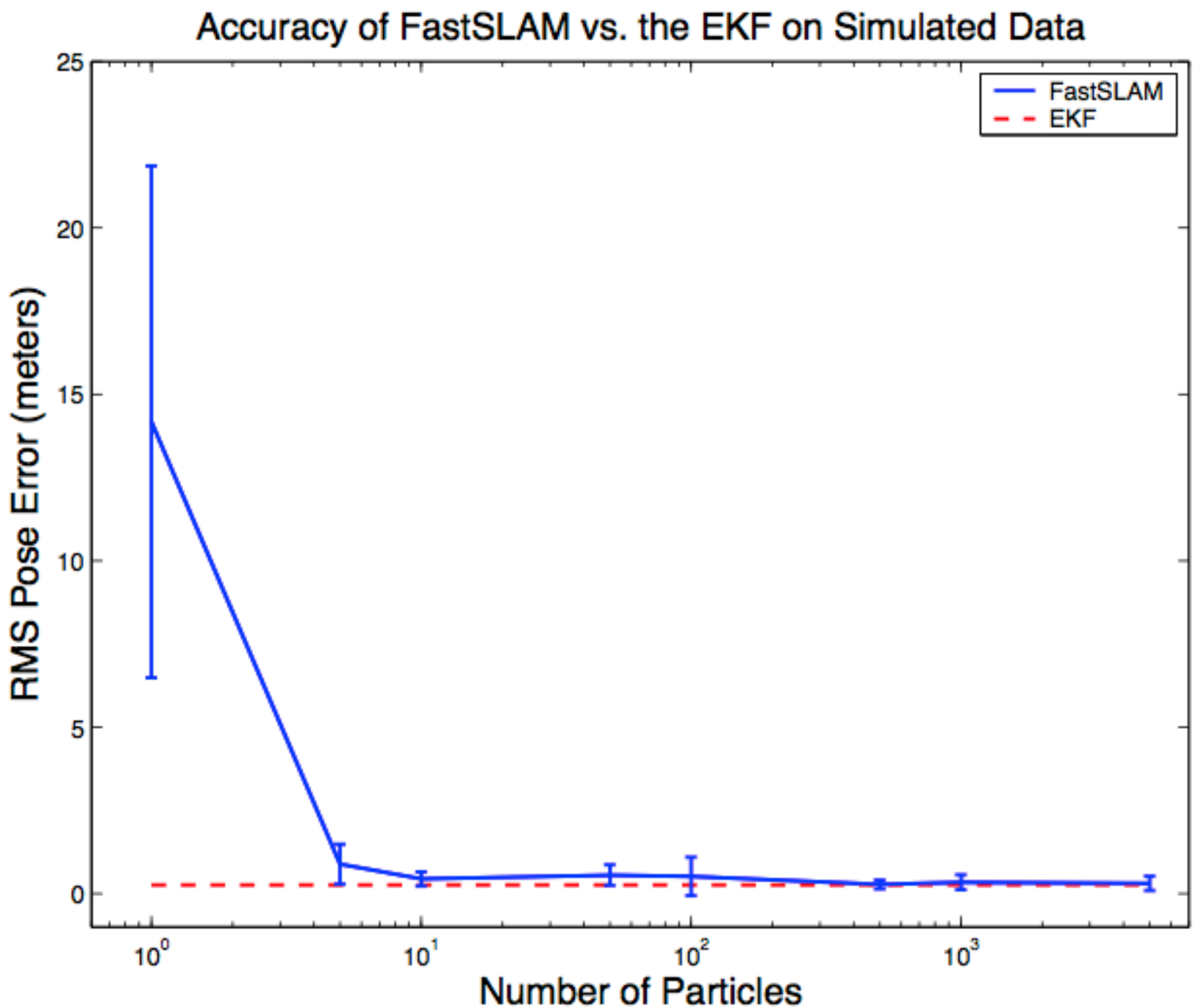
Blue = GPS
Yellow = FastSLAM

Courtesy: M. Montemerlo

Results – Victoria Park (Video)

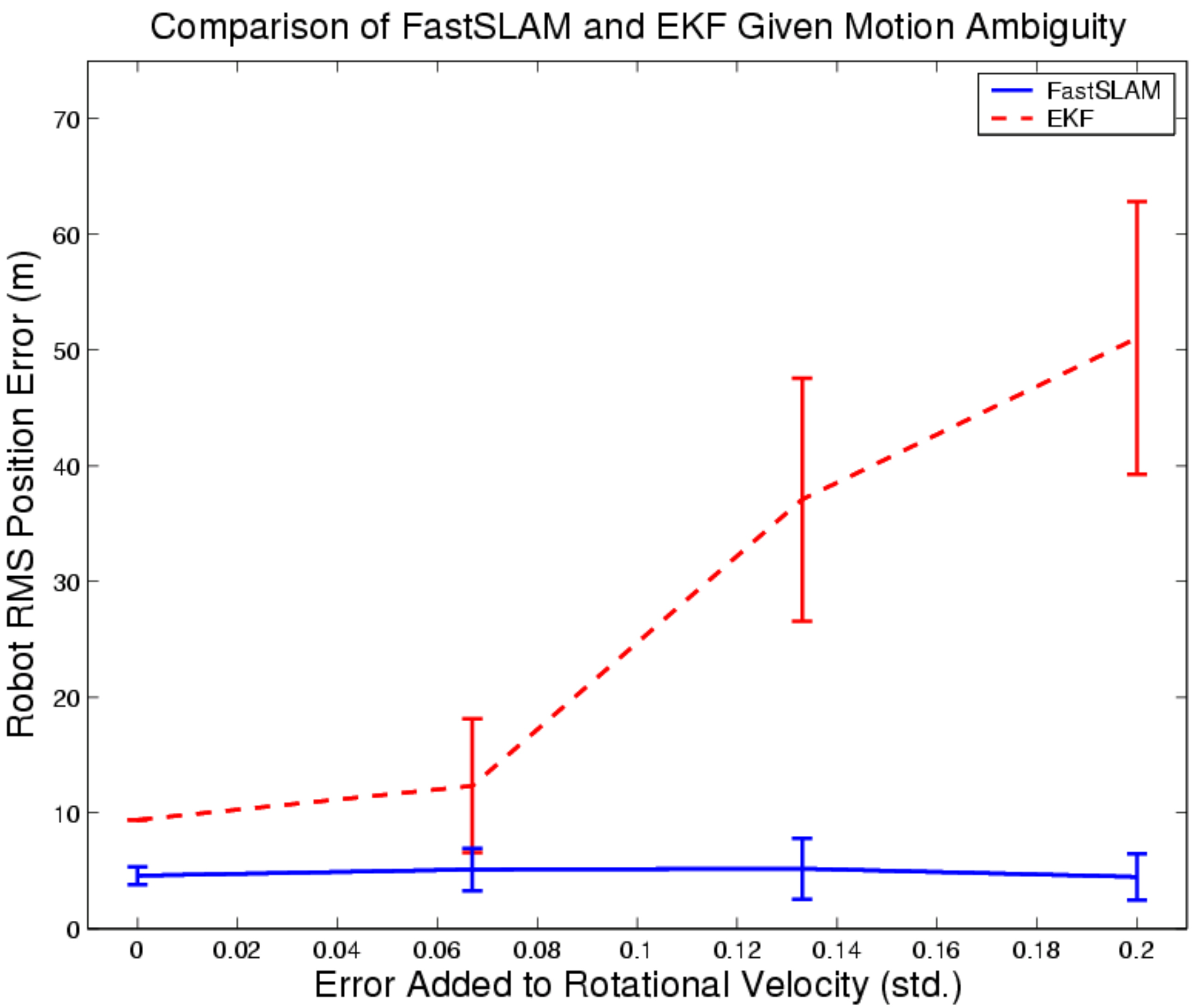
Courtesy: M. Montemerlo

Results (Sample Size)



Courtesy: M. Montemerlo

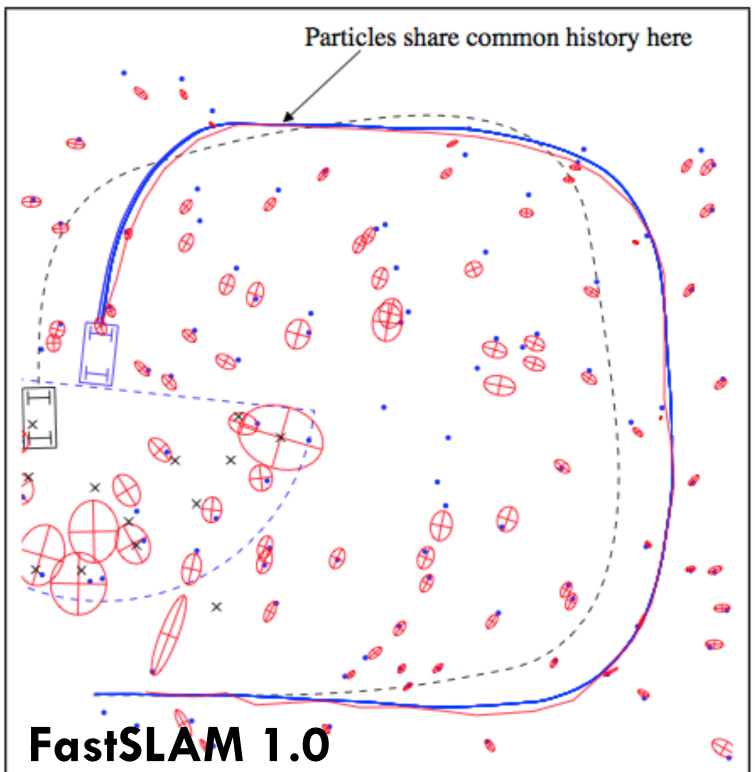
Results (Motion Uncertainty)

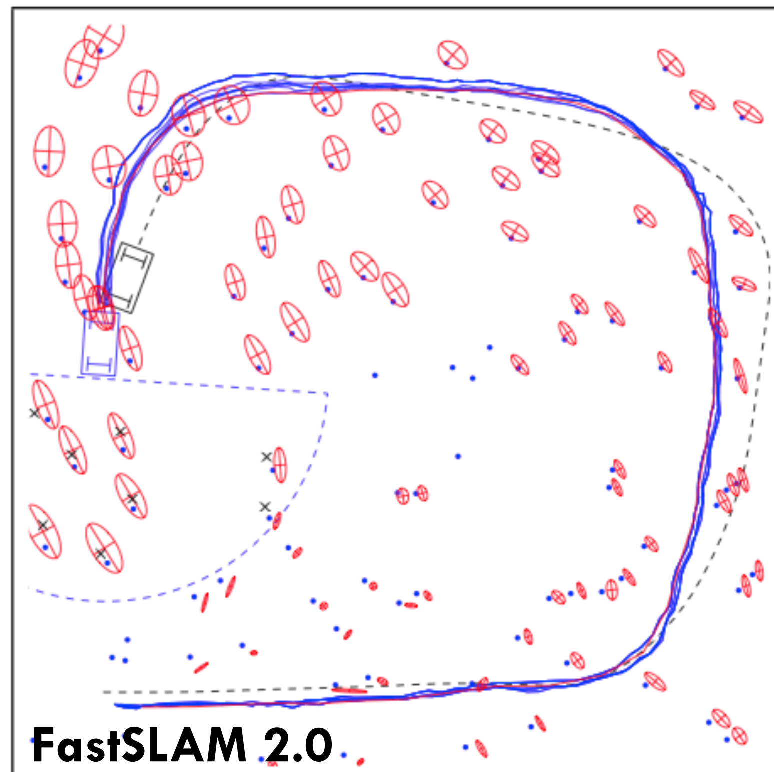


Courtesy: M. Montemerlo

FastSLAM Problems

- How to determine the sample size?
- Particle deprivation, especially when closing (multiple) loops

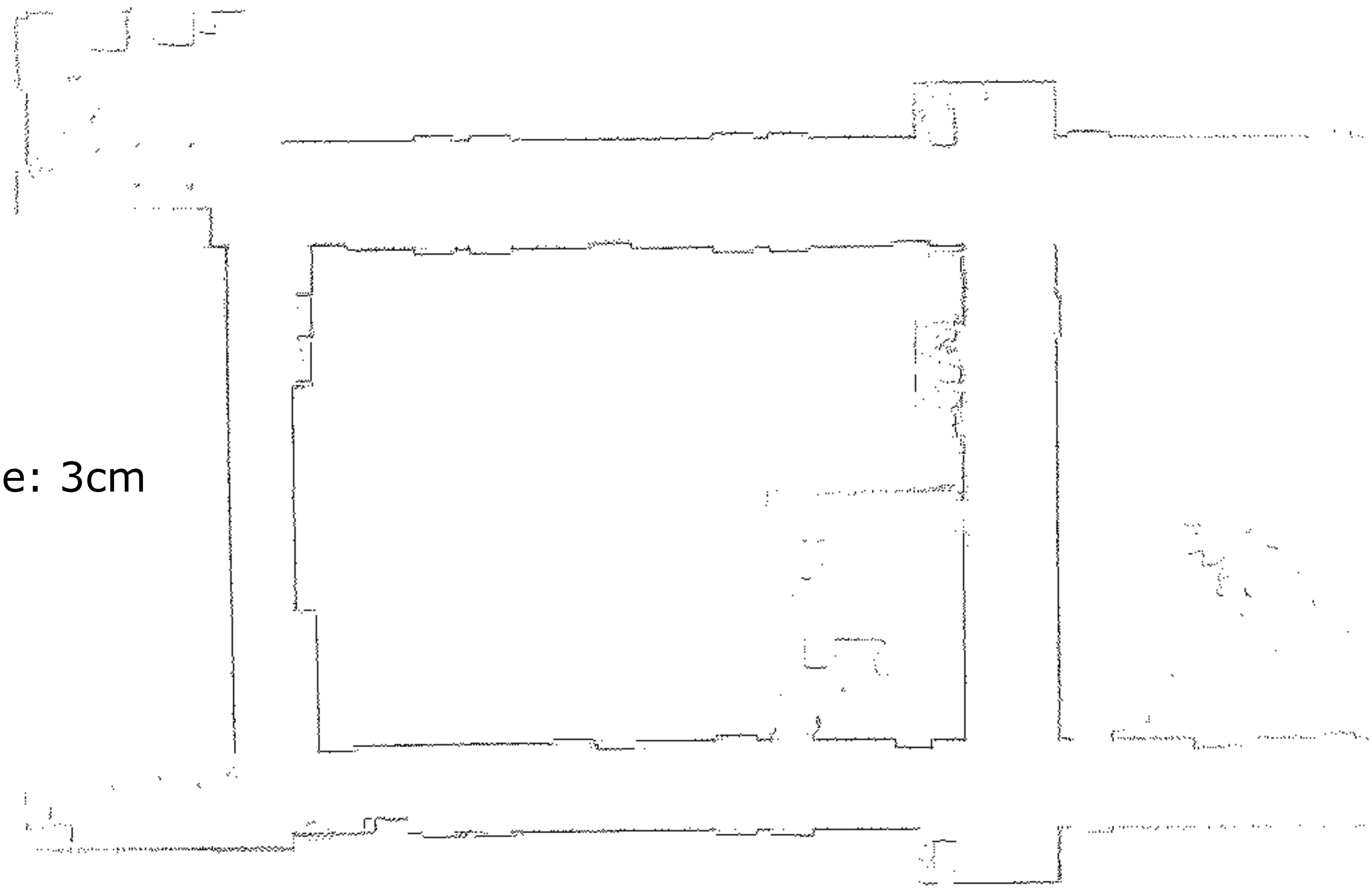




Courtesy: M. Montemerlo

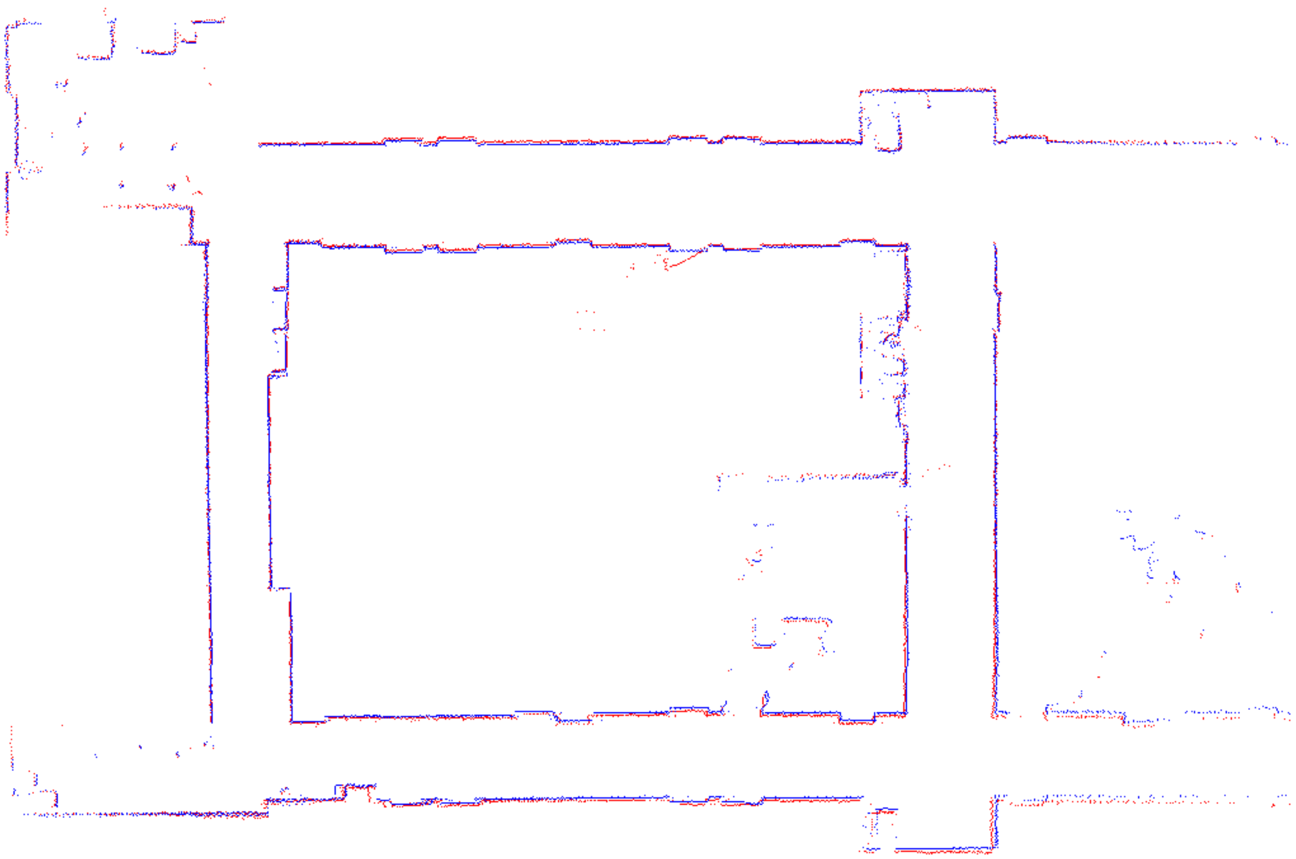
DP-SLAM: High-Res Fast-SLAM via History Sharing

scale: 3cm

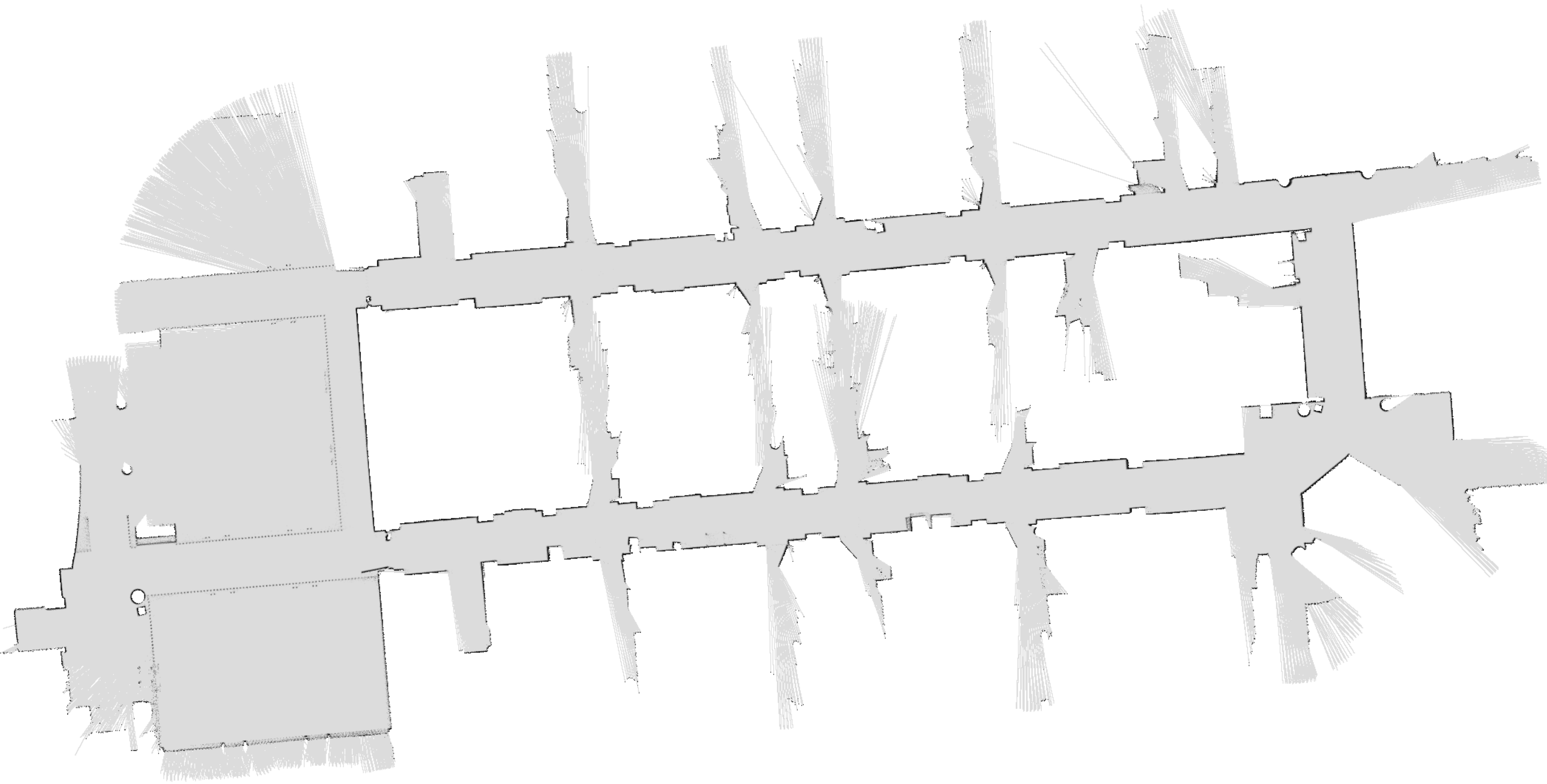


Run at real-time speed on 2.4GHz Pentium 4 at 10cm/s

Consistency

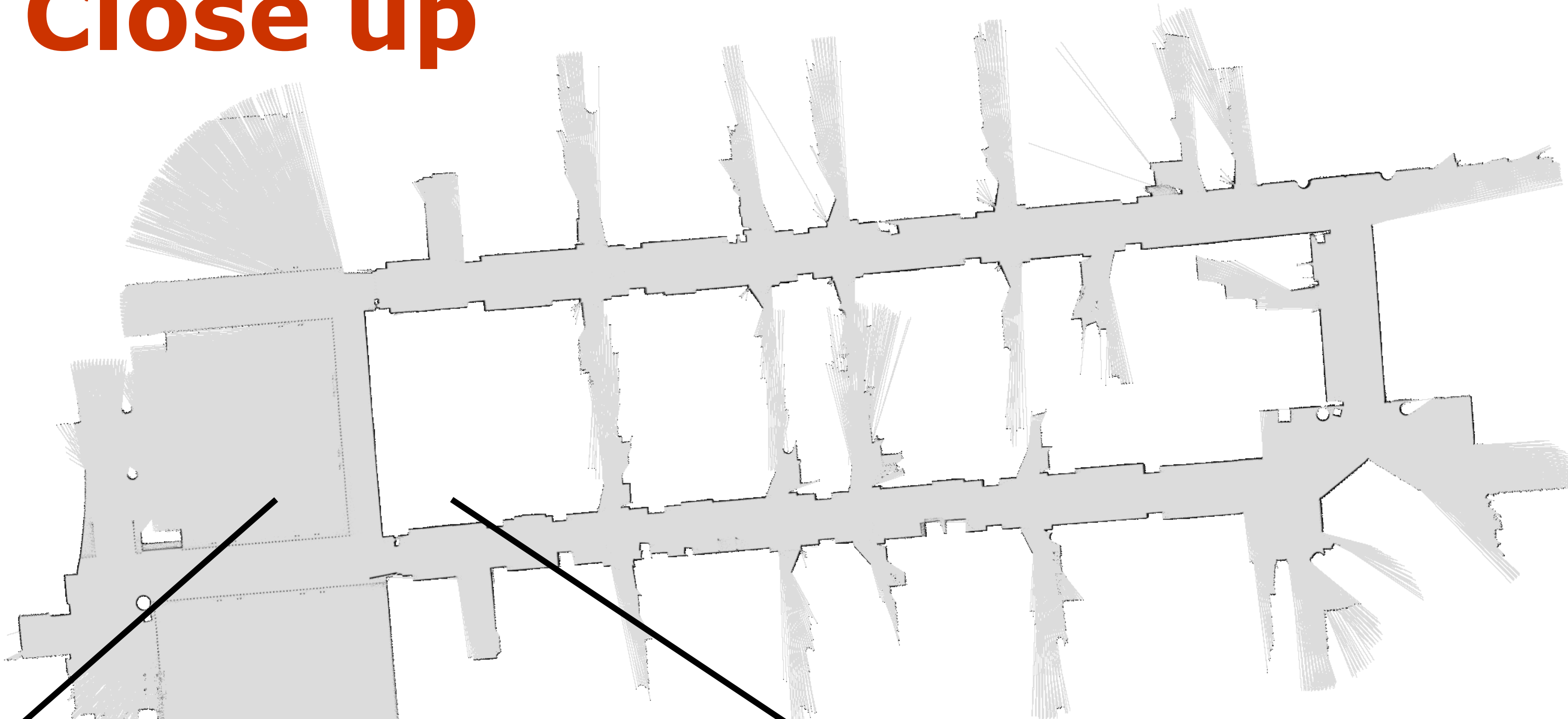


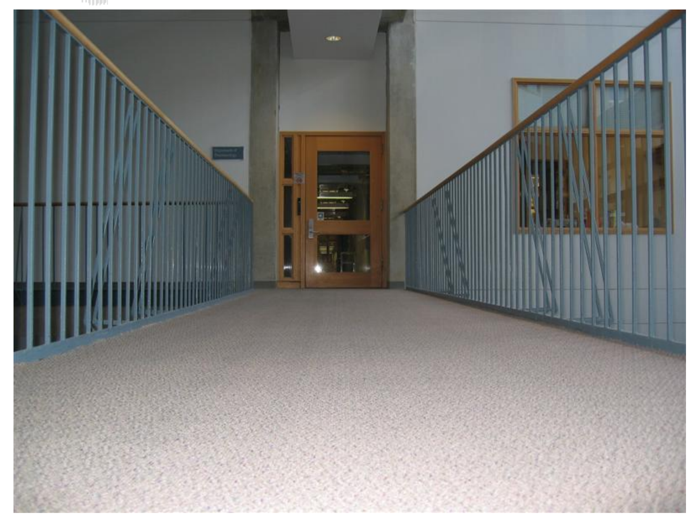
Results obtained with DP-SLAM 2.0 (offline)



Eliazar & Parr, 04

Close up





End courtesy of Eliazar & Parr

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

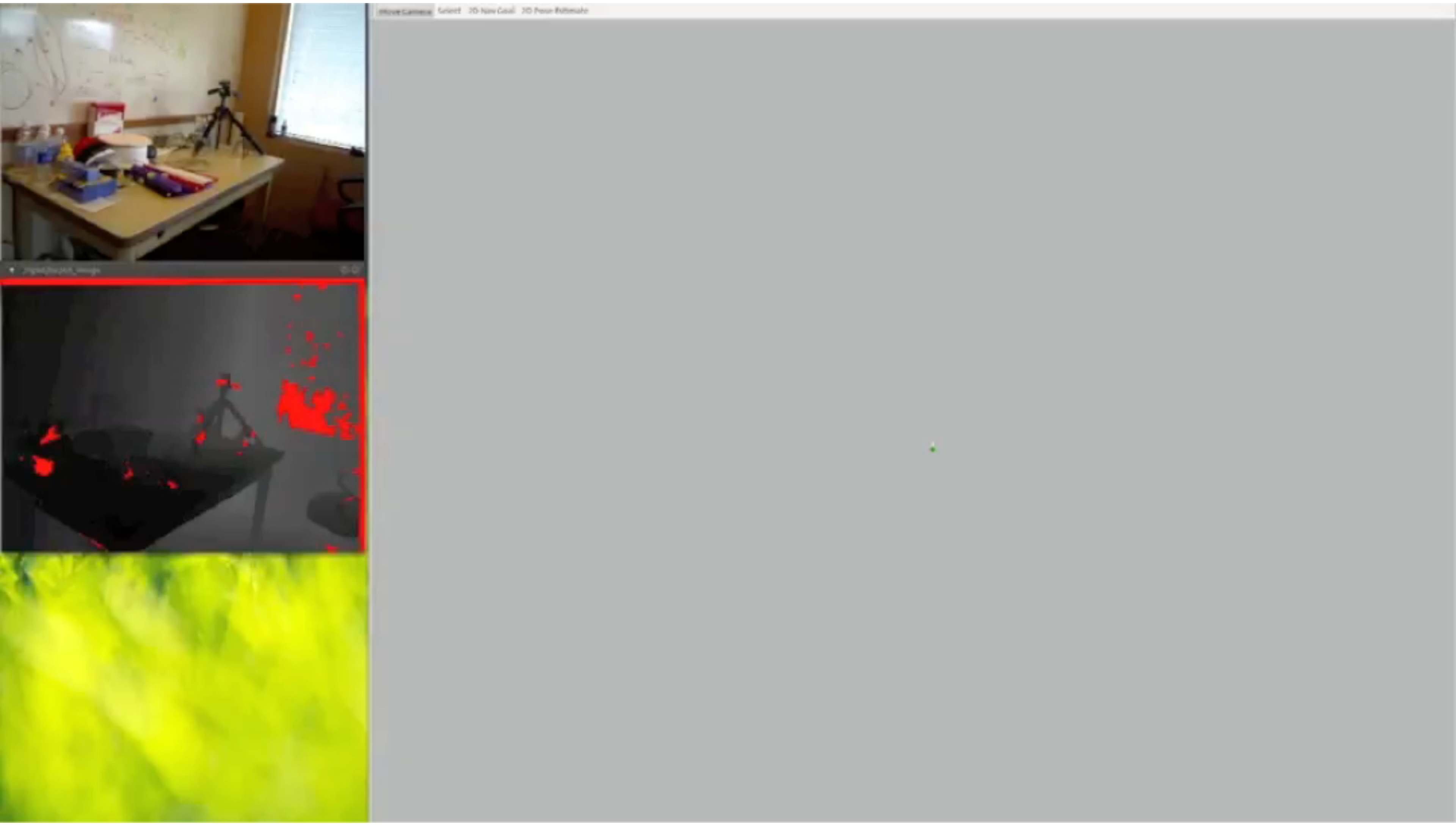
Literature

FastSLAM

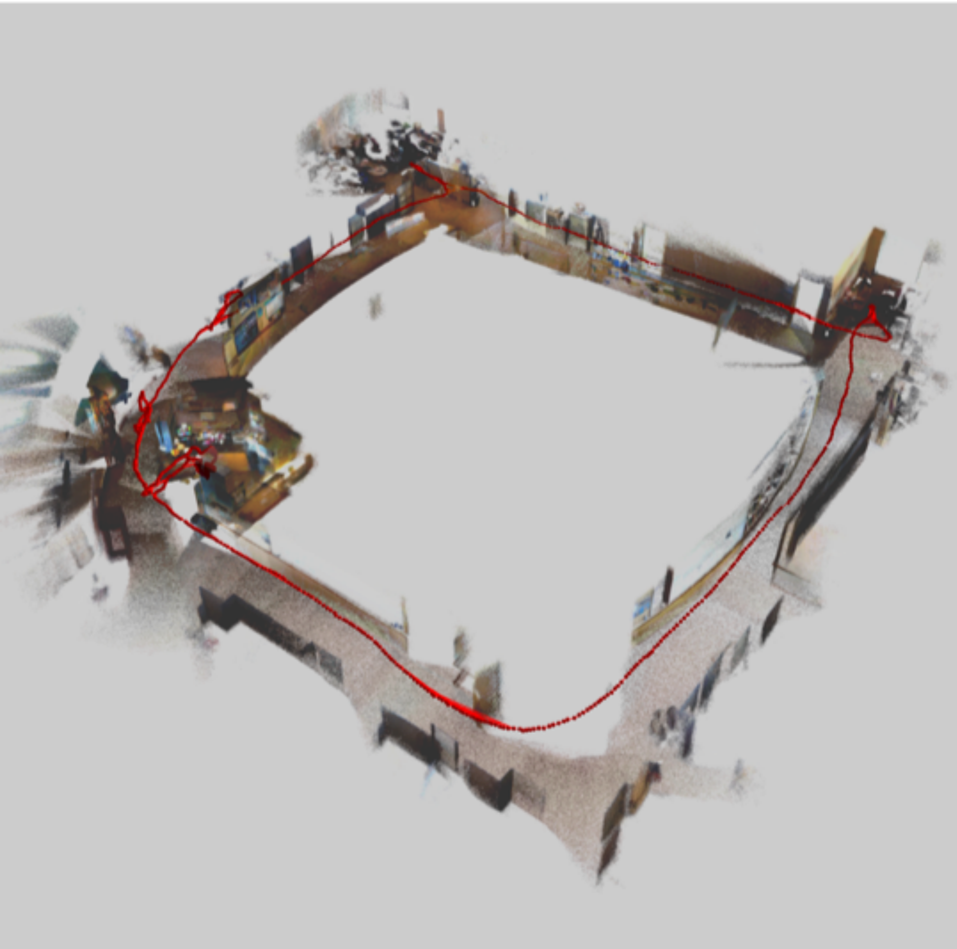
- Thrun et al.: “Probabilistic Robotics”, Chapter 13.1-13.3 + 13.8 (see errata!)
- Montemerlo, Thrun, Kollar, Wegbreit: FastSLAM: A Factored Solution to the Simultaneous Localization and Mapping Problem, 2002
- Montemerlo and Thrun: Simultaneous Localization and Mapping with Unknown Data Association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

Courtesy: C. Stachniss

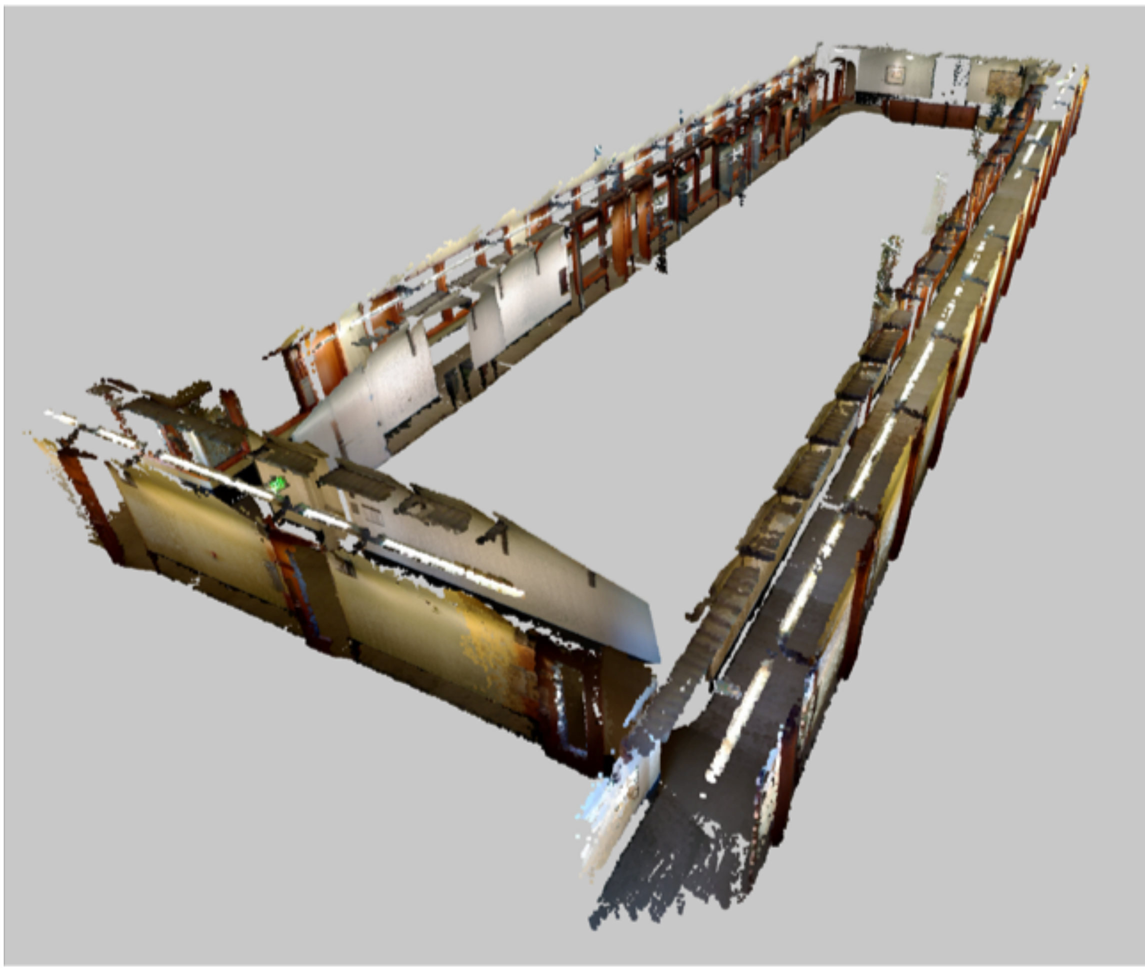
RGBD SLAM

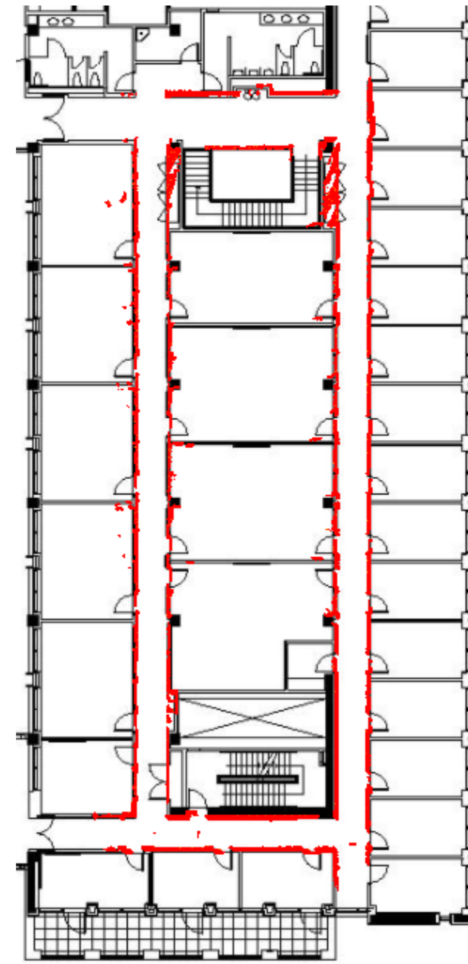


Resulting Map

Experiments: Overlay 1





Experiments: Overlay 2