Introduction to Mobile Robotics

SLAM: Simultaneous Localization and Mapping

Wolfram Burgard, Cyrill Stachniss,

Maren Bennewitz, Kai Arras

Slides by Kai Arras and Wolfram Burgard

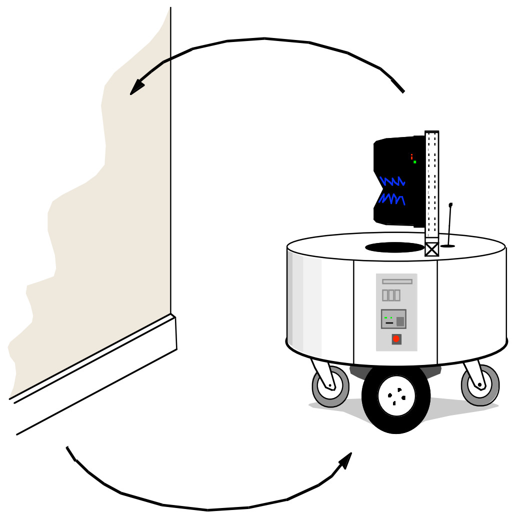
Last update: June 2010

The SLAM Problem

SLAM is the process by which a robot **builds a map** of the environment and, at the same time, uses this map to **compute its location**

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

The SLAM Problem



- SLAM is a **chicken-or-egg problem**:
  → A map is needed for localizing a robot
  → A pose estimate is needed to build a map
- Thus, SLAM is (regarded as) a **hard problem** in robotics

The SLAM Problem

- SLAM is considered **one of the most fundamental problems** for robots to become truly autonomous
- A variety of different approaches to address the SLAM problem have been presented
- **Probabilistic methods** rule
- History of SLAM dates back to the **mid-eighties** (stone-age of mobile robotics)

The SLAM Problem

Given:

- The robot's controls

$$\mathbf{U}_{0:k} = \{\mathbf{u}_1, \mathbf{u}_2, \cdots, \mathbf{u}_k\}$$

- Relative observations

$$\mathbf{Z}_{0:k} = \{\mathbf{z}_1, \mathbf{z}_2, \cdots, \mathbf{z}_k\}$$

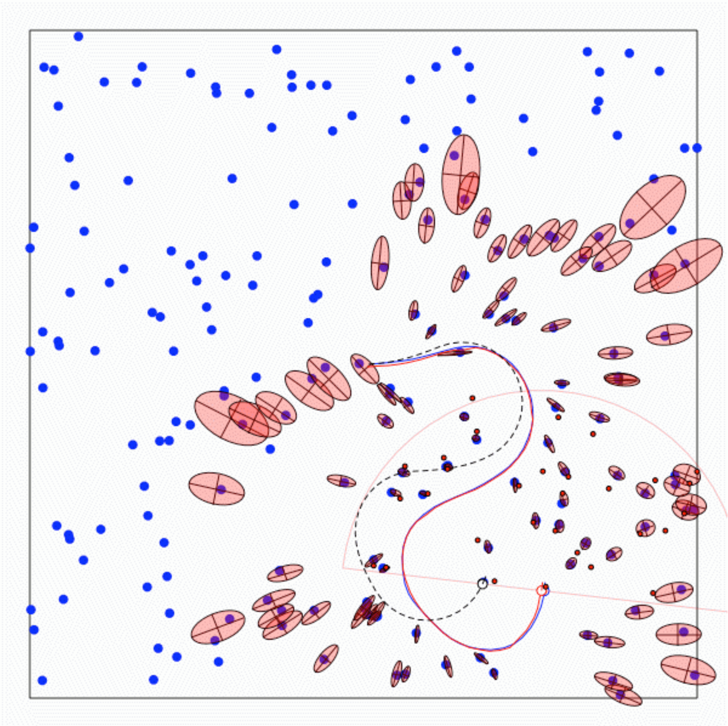
Wanted:

- Map of features

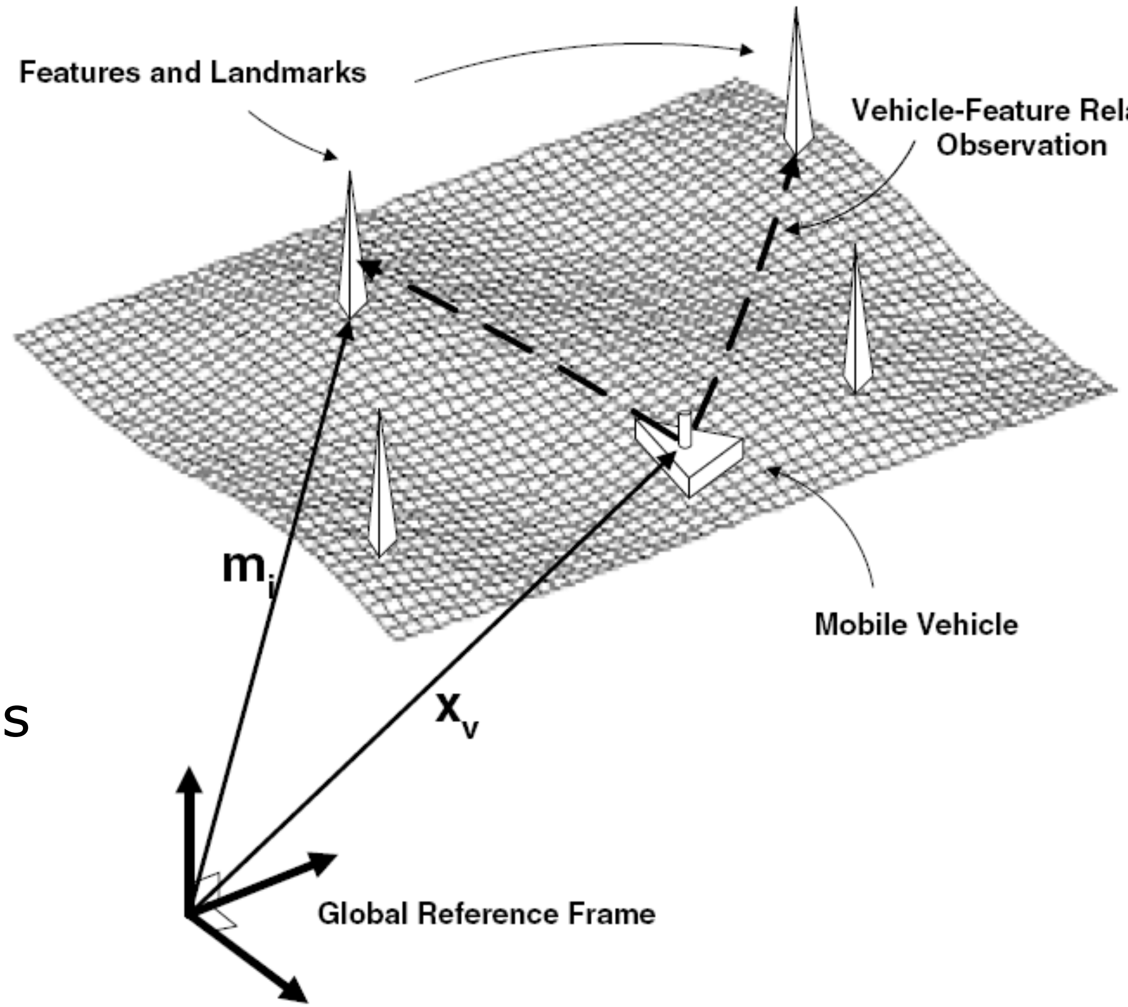
$$\mathbf{m} = \{\mathbf{m}_1, \mathbf{m}_2, \cdots, \mathbf{m}_n\}$$

- Path of the robot

$$\mathbf{X}_{0:k} = \{\mathbf{x}_0, \mathbf{x}_1, \cdots, \mathbf{x}_k\}$$



The SLAM Problem



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

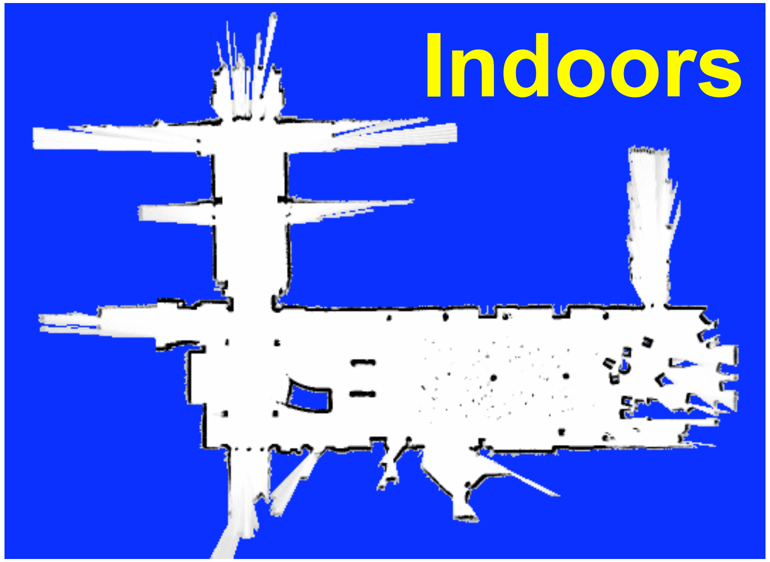
SLAM Applications

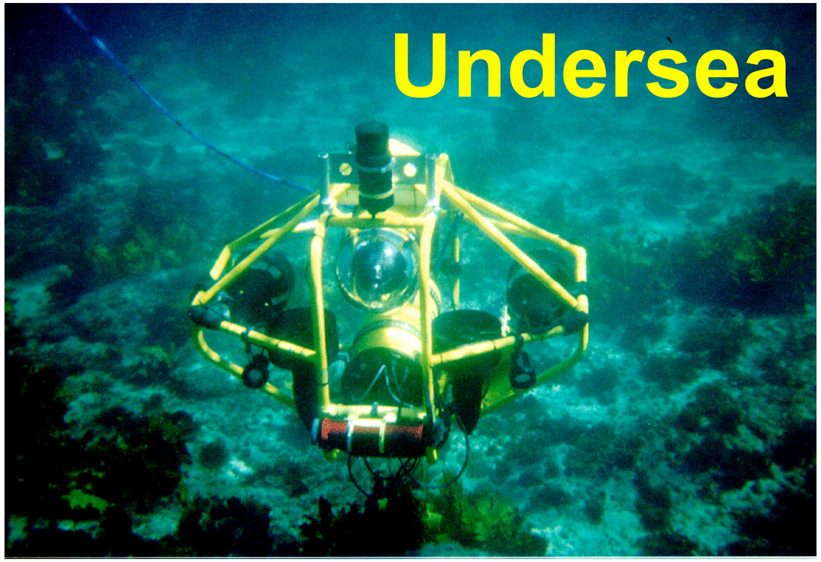
SLAM is central to a range of indoor, outdoor, in-air and underwater **applications** for both manned and autonomous vehicles.

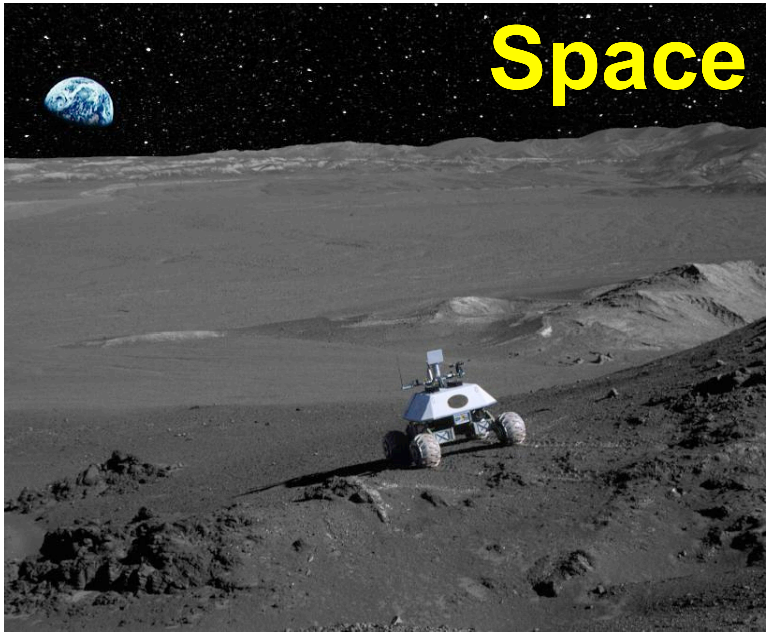
Examples:

- At home: vacuum cleaner, lawn mower
- Air: surveillance with unmanned air vehicles
- Underwater: reef monitoring
- Underground: exploration of abandoned mines
- Space: terrain mapping for localization

SLAM Applications



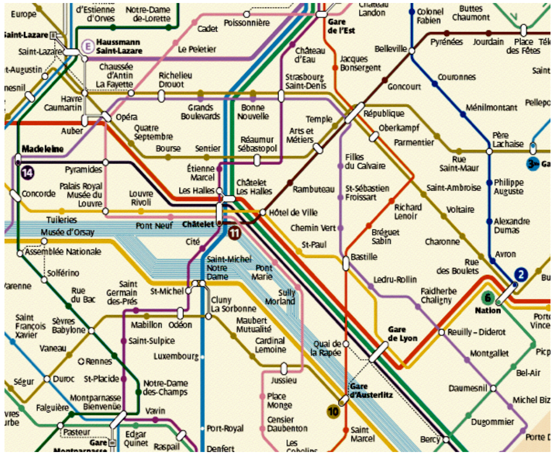


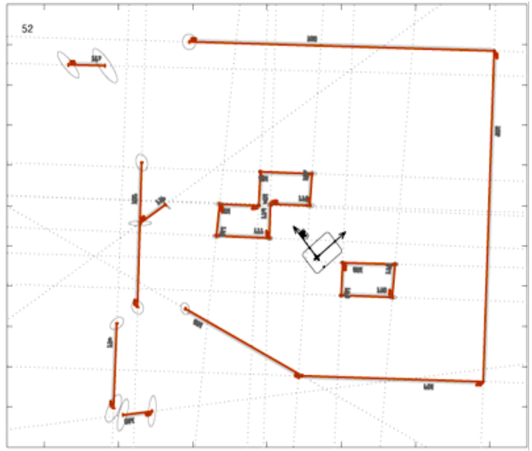


Map Representations

Examples:

Subway map, city map, landmark-based map

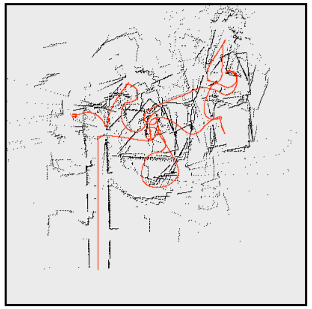


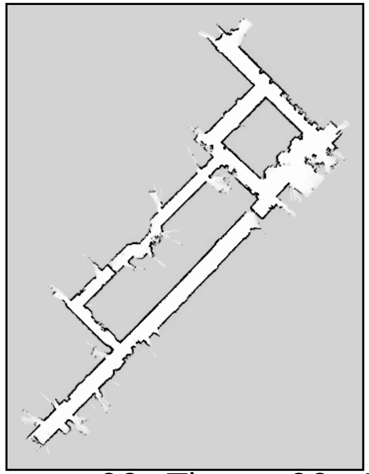


Maps are **topological** and/or **metric models** of the environment

Map Representations

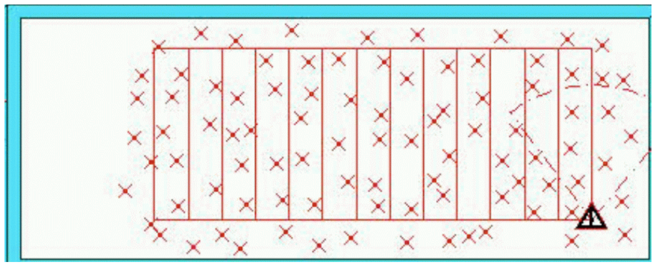
- Grid maps or scans, 2d, 3d

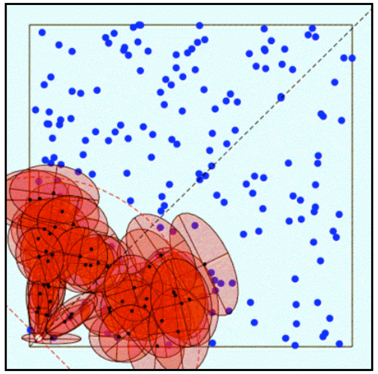




[Lu & Milios, 97; Gutmann, 98: Thrun 98; Burgard, 99; Konolige & Gutmann, 00; Thrun, 00; Arras, 99; Haehnel, 01;…]

- Landmark-based

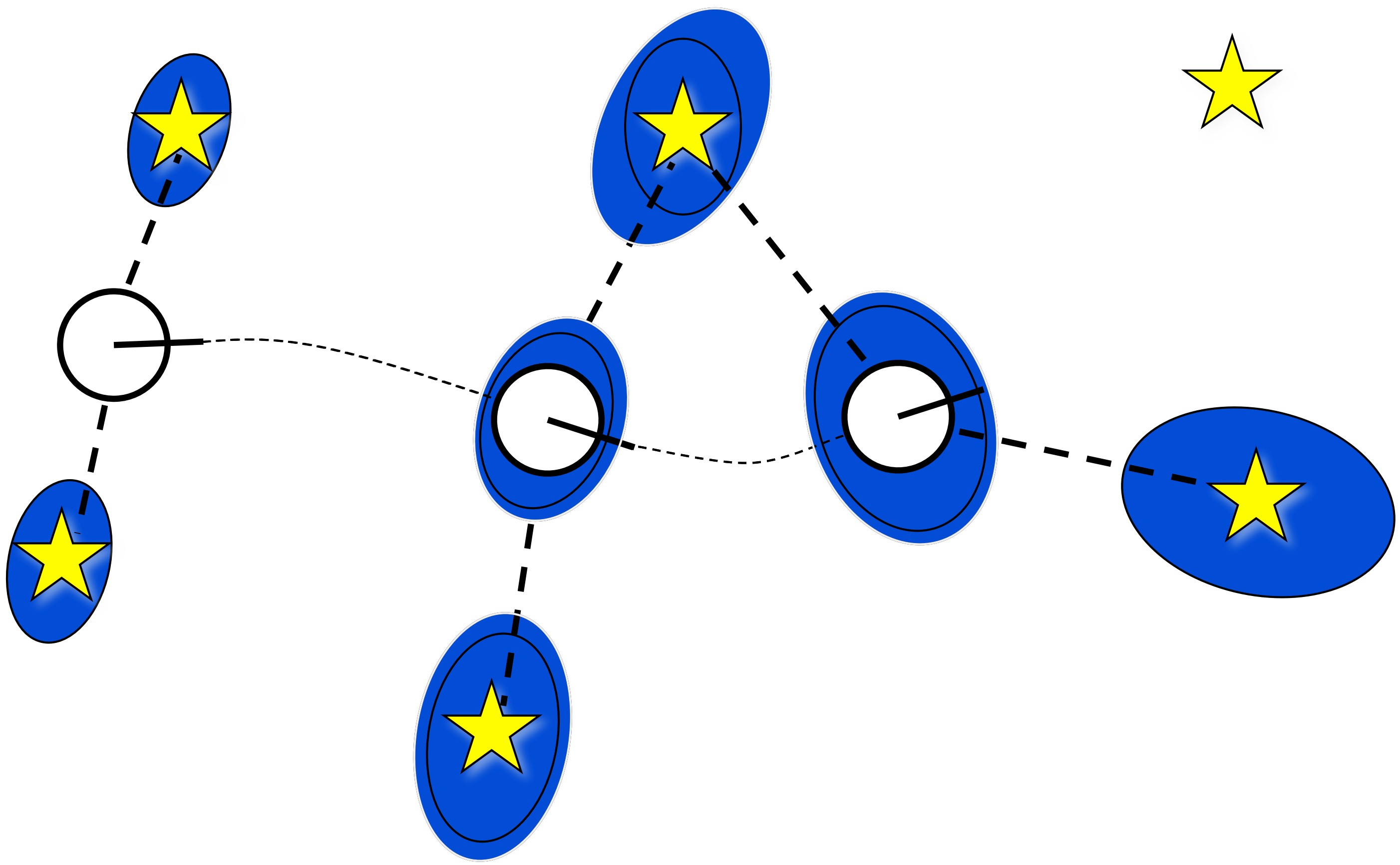




[Leonard et al., 98; Castelanos et al., 99: Dissanayake et al., 2001; Montemerlo et al., 2002;…

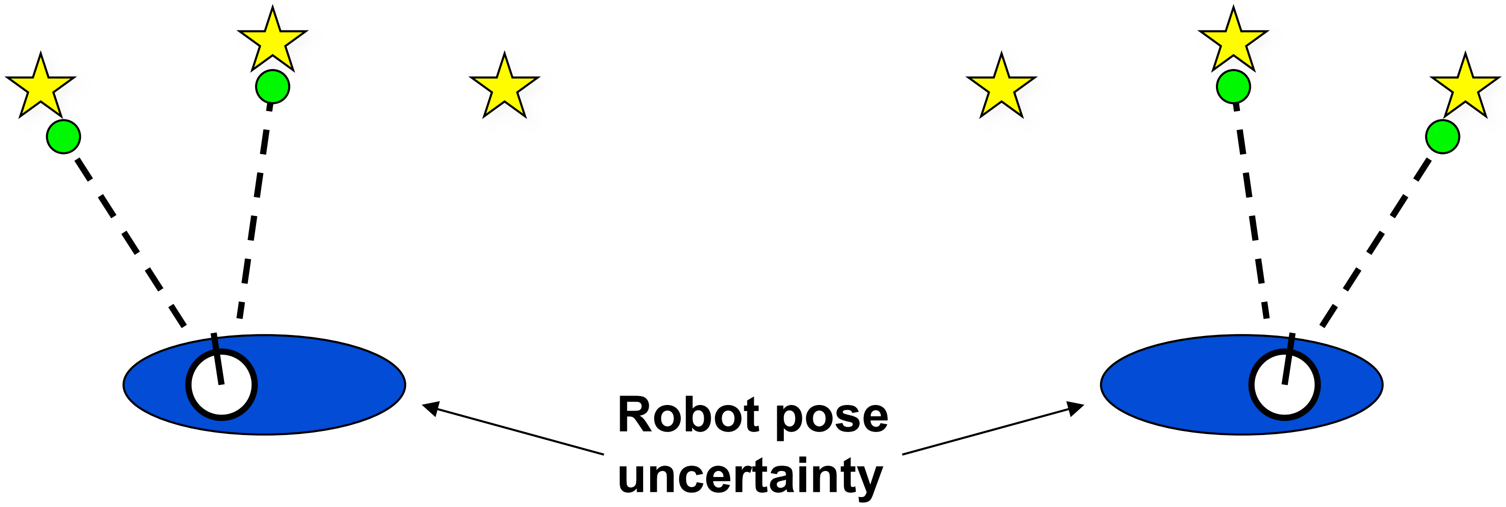
Why is SLAM a hard problem?

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



2. Errors in map and pose estimates correlated

Why is SLAM a hard problem?



- In the real world, the **mapping between observations and landmarks is unknown** (origin uncertainty of measurements)
- **Data Association**: 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!

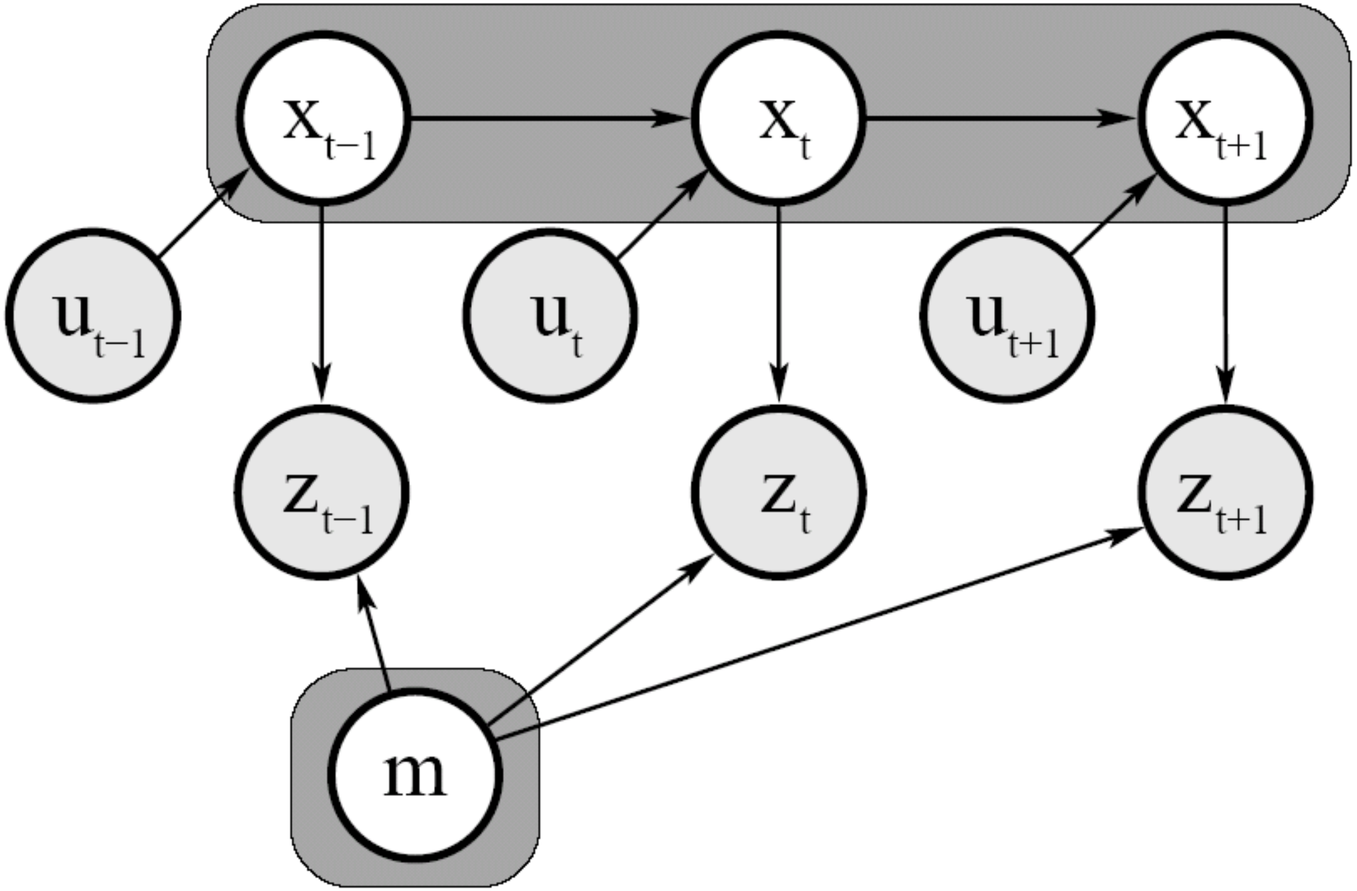
- Online SLAM:

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

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

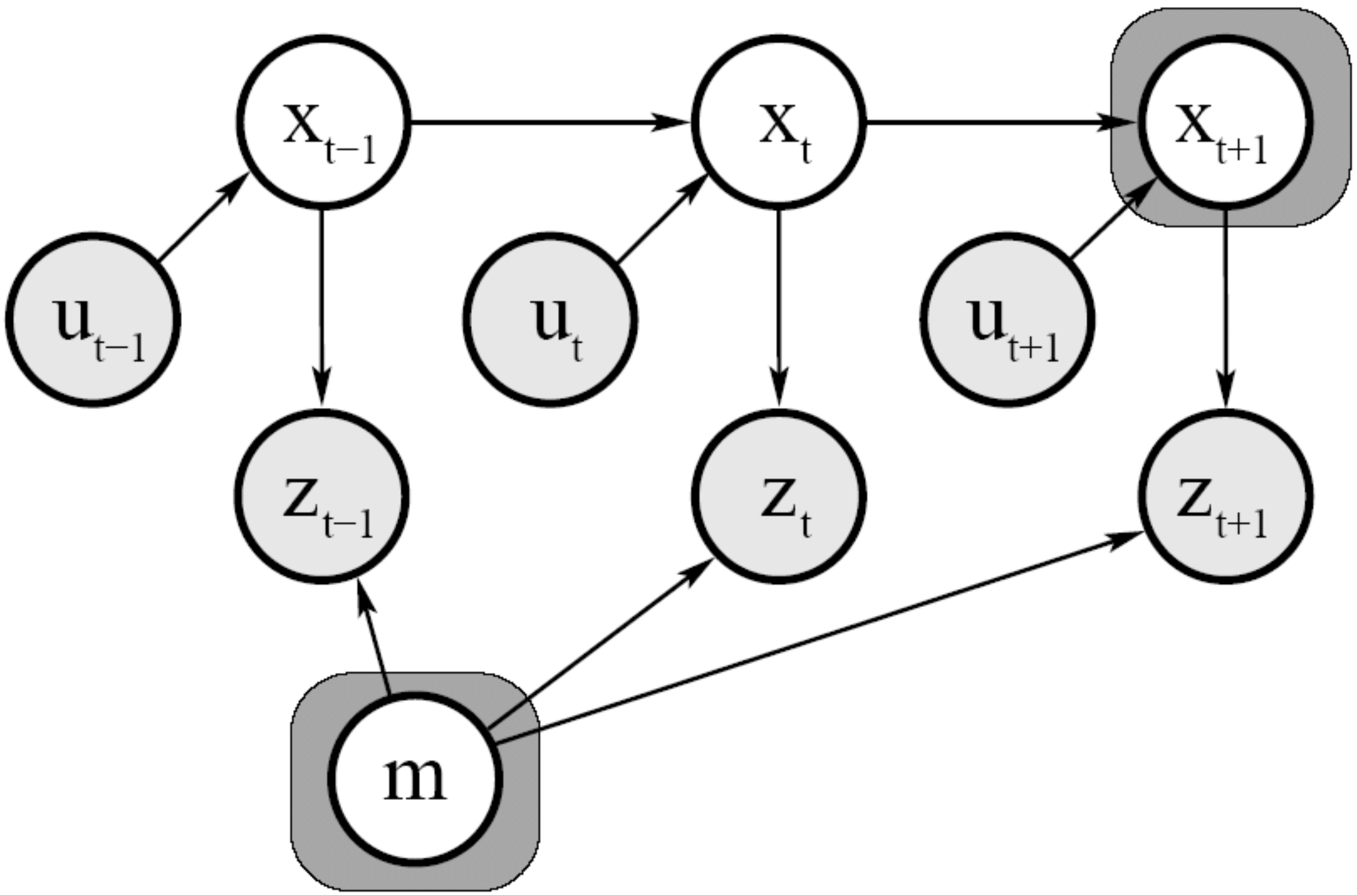
Estimates most recent pose and map!

Graphical Model of Full SLAM



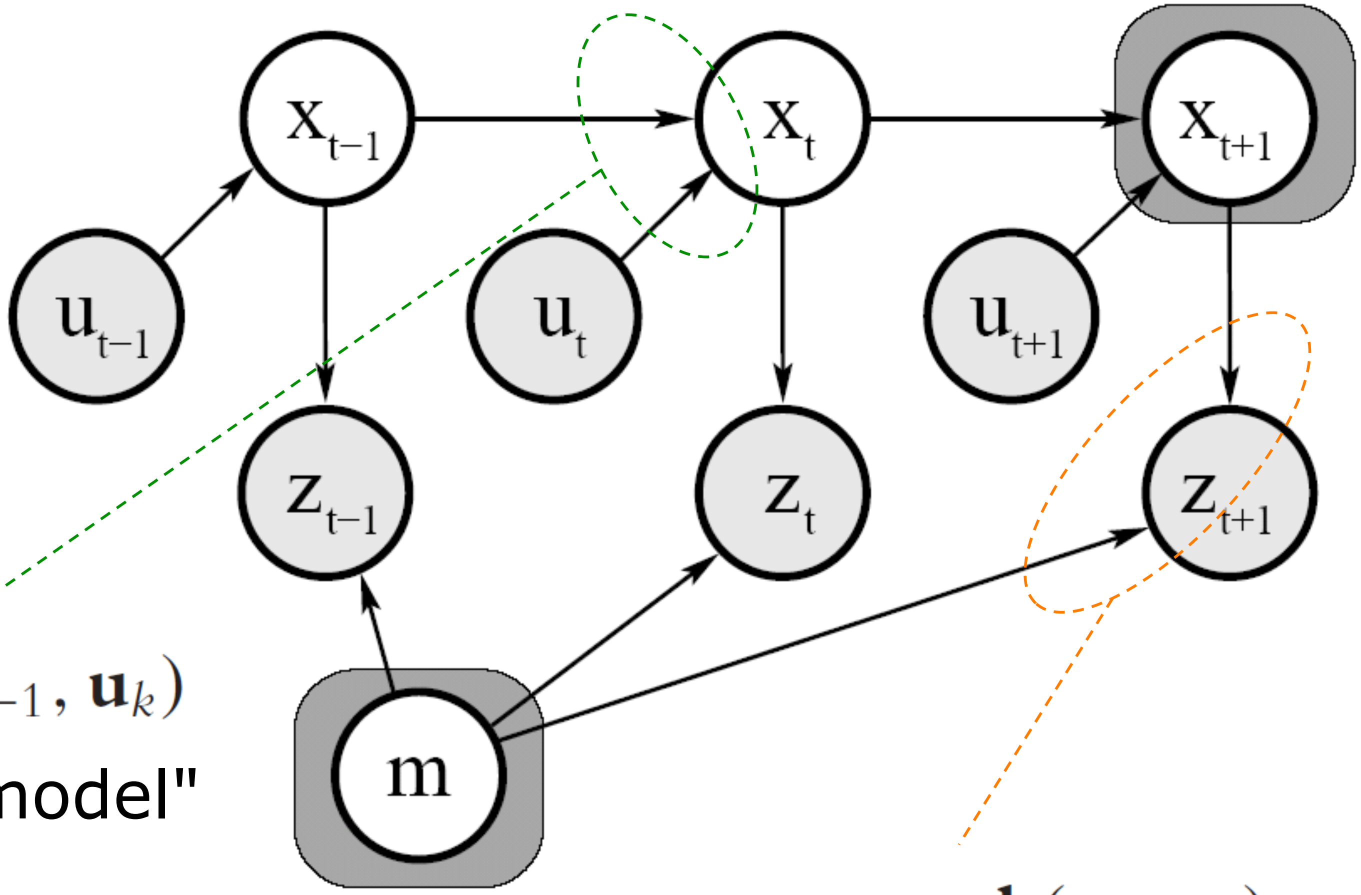
$$p(x_{1:t}, m \mid z_{1:t}, u_{1:t})$$

Graphical Model of Online SLAM



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

Graphical Model: Models



$$\mathbf{x}_k = \mathbf{f}(\mathbf{x}_{k-1}, \mathbf{u}_k)$$

"Motion model"

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

"Observation model"

Remember? KF Algorithm

1. Algorithm **Kalman_filter**(μ_{t-1}, Σ_{t-1}, u_t, z_t):
2. Prediction:
3. $\bar{\mu}_t = A_t \mu_{t-1} + B_t u_t$
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

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

- **SLAM**

  Landmarks are **simply added** to the state.

  Growing state vector and covariance matrix!

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

EKF SLAM: State representation

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

$$Bel(x_t, m_t) = \left\langle \begin{pmatrix} x \\ y \\ \theta \\ l_1 \\ l_2 \\ \vdots \\ l_N \end{pmatrix}, \begin{pmatrix} \sigma_x^2 & \sigma_{xy} & \sigma_{x\theta} & \sigma_{xl_1} & \sigma_{xl_2} & \cdots & \sigma_{xl_N} \\ \sigma_{xy} & \sigma_y^2 & \sigma_{y\theta} & \sigma_{yl_1} & \sigma_{yl_2} & \cdots & \sigma_{yl_N} \\ \sigma_{x\theta} & \sigma_{y\theta} & \sigma_\theta^2 & \sigma_{\theta l_1} & \sigma_{\theta l_2} & \cdots & \sigma_{\theta l_N} \\ \sigma_{xl_1} & \sigma_{yl_1} & \sigma_{\theta l_1} & \sigma_{l_1}^2 & \sigma_{l_1 l_2} & \cdots & \sigma_{l_1 l_N} \\ \sigma_{xl_2} & \sigma_{yl_2} & \sigma_{\theta l_2} & \sigma_{l_1 l_2} & \sigma_{l_2}^2 & \cdots & \sigma_{l_2 l_N} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ \sigma_{xl_N} & \sigma_{yl_N} & \sigma_{\theta l_N} & \sigma_{l_1 l_N} & \sigma_{l_2 l_N} & \cdots & \sigma_{l_N}^2 \end{pmatrix} \right\rangle$$

- Can handle hundreds of dimensions

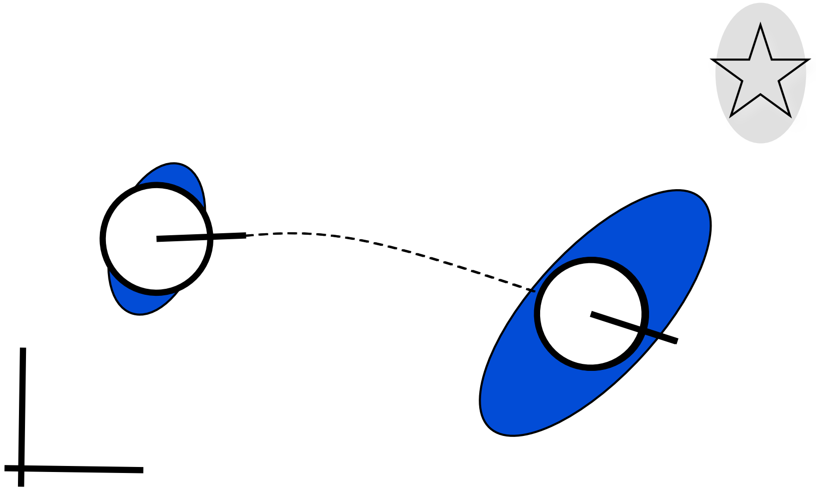
EKF SLAM: Building the Map

Filter Cycle, Overview:

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

EKF SLAM: Building the Map

- State Prediction



Odometry:

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

$$\hat{C}_R = F_x \, C_R \, F_x^T + F_u \, U \, F_u^T$$

Robot-landmark cross-covariance prediction:

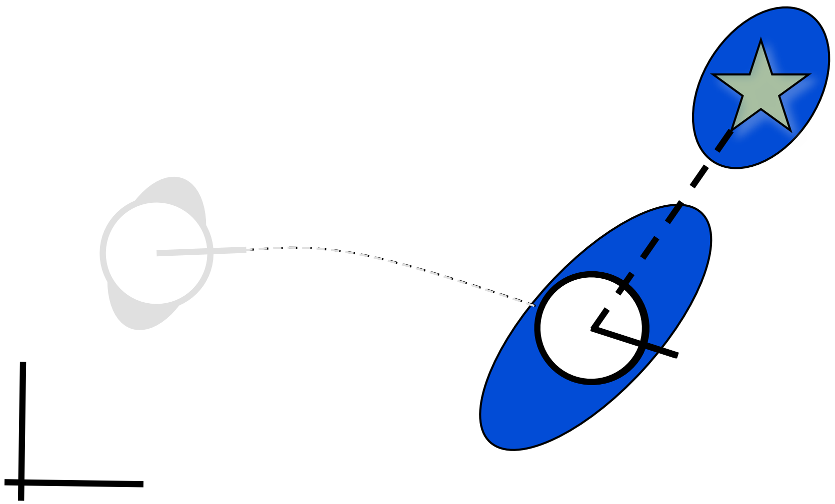
$$\hat{C}_{RM_i} = F_x \, C_{RM_i}$$

(skipping time index k)

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

EKF SLAM: Building the Map

- Measurement Prediction



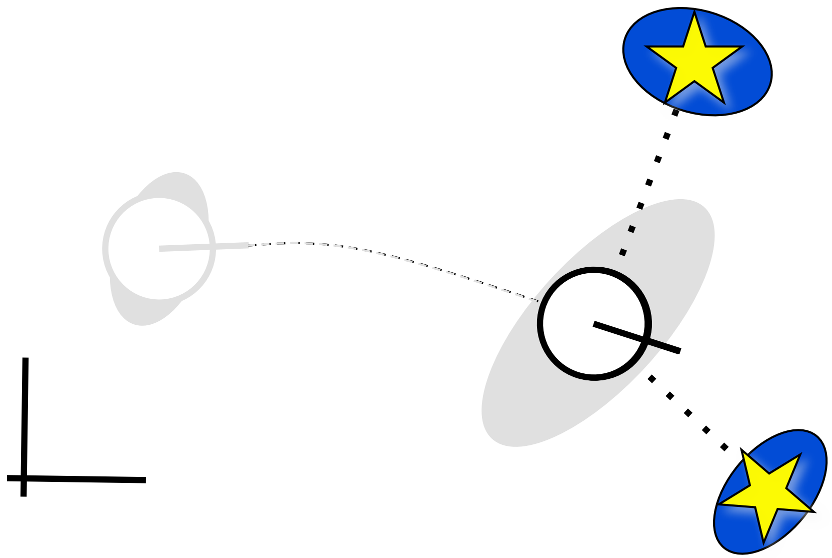
Global-to-local frame transform h

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

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

EKF SLAM: Building the Map

- Observation



(*x,y*)-point landmarks

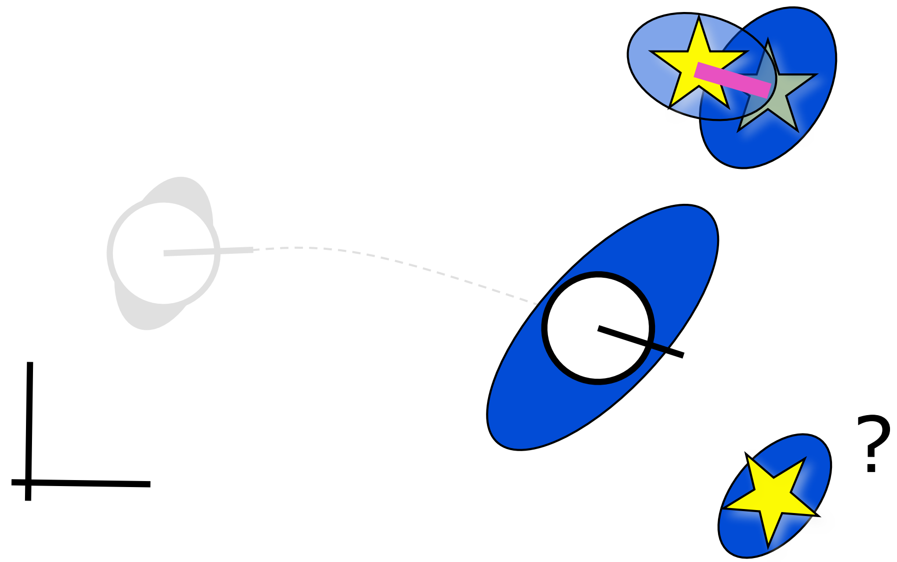
$$\mathbf{z}_k = \begin{bmatrix} x_1 \\ y_1 \\ x_2 \\ y_2 \end{bmatrix} = \begin{bmatrix} \mathbf{z}_1 \\ \mathbf{z}_2 \end{bmatrix}$$

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

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

EKF SLAM: Building the Map

- Data Association



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

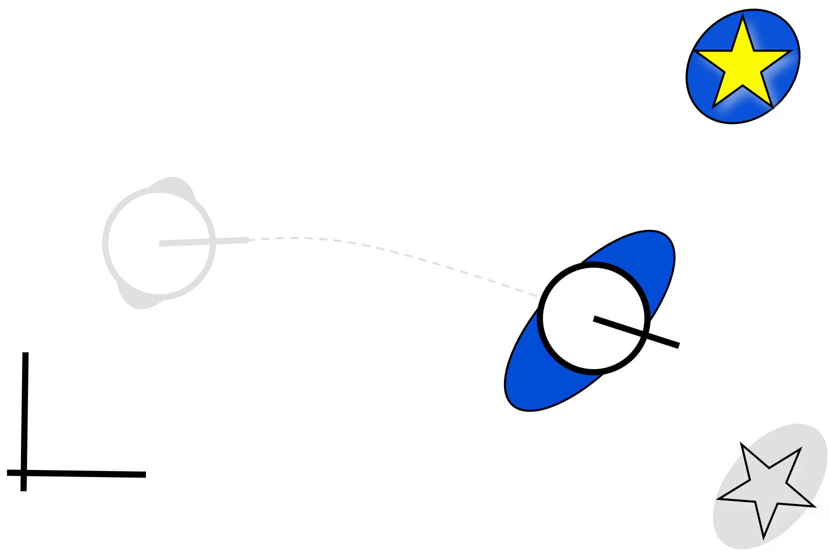
$$\begin{aligned} \nu_k^{ij} &= \mathbf{z}_k^j - \hat{\mathbf{z}}_k^i \\ S_k^{ij} &= R_k^j + H^i \, \hat{C}_k \, H^{i\,T} \end{aligned}$$

(Gating)

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

EKF SLAM: Building the Map

- Filter Update



The usual Kalman filter expressions

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

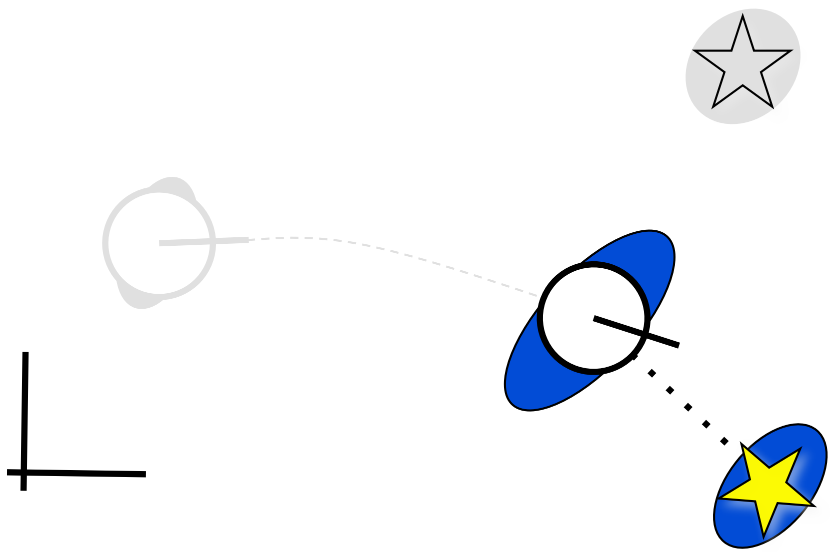
$$\mathbf{x}_k = \hat{\mathbf{x}}_k + K_k \, \nu_k$$

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

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

EKF SLAM: Building the Map

- Integrating New Landmarks



State augmented by

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

$$C_{M_{n+1}} = G_R\, C_R\, G_R^T + G_z\, R_j\, G_z^T$$

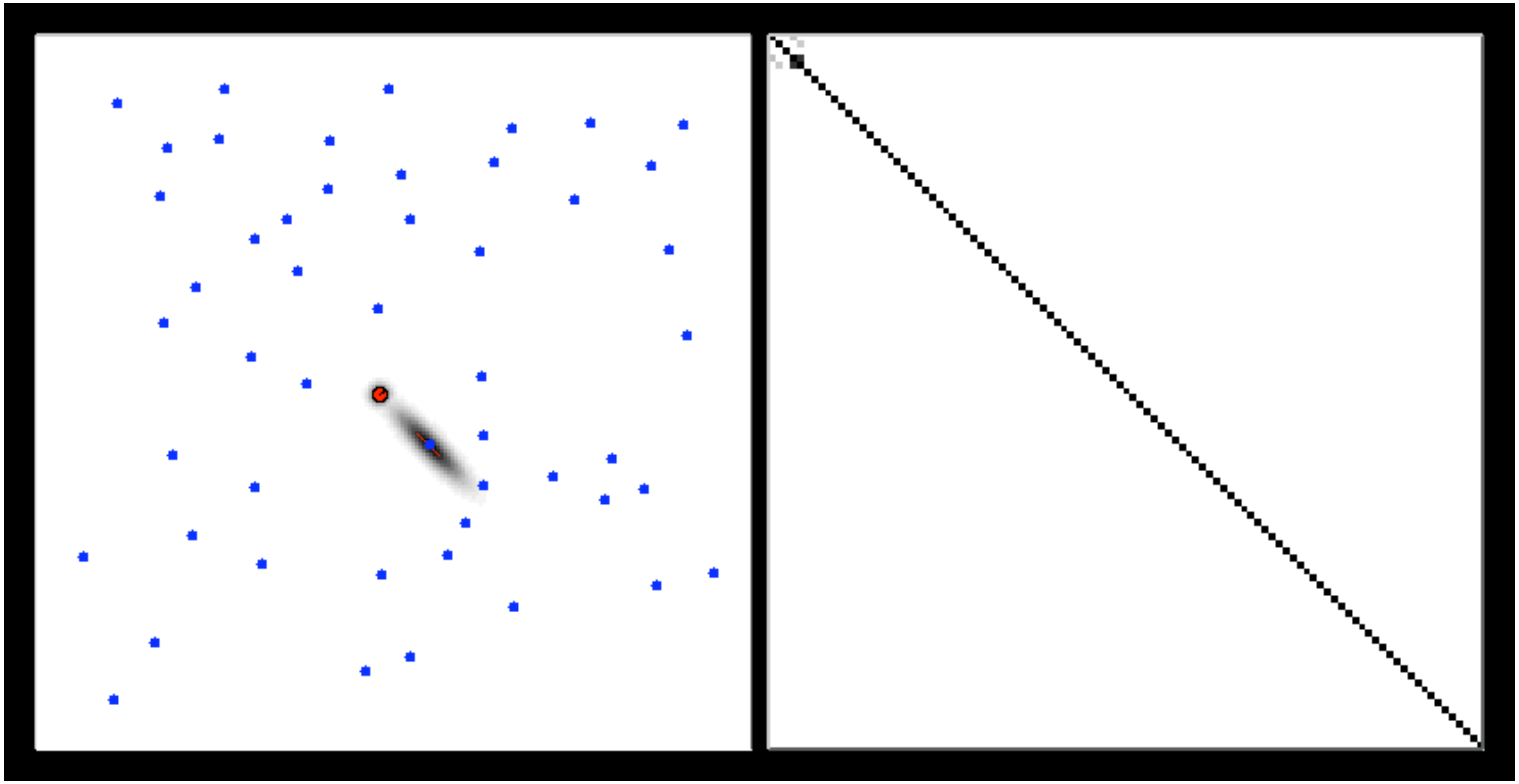
Cross-covariances:

$$C_{M_{n+1}M_i} = G_R\, C_{RM_i}$$

$$C_{M_{n+1}R} = G_R\, C_R$$

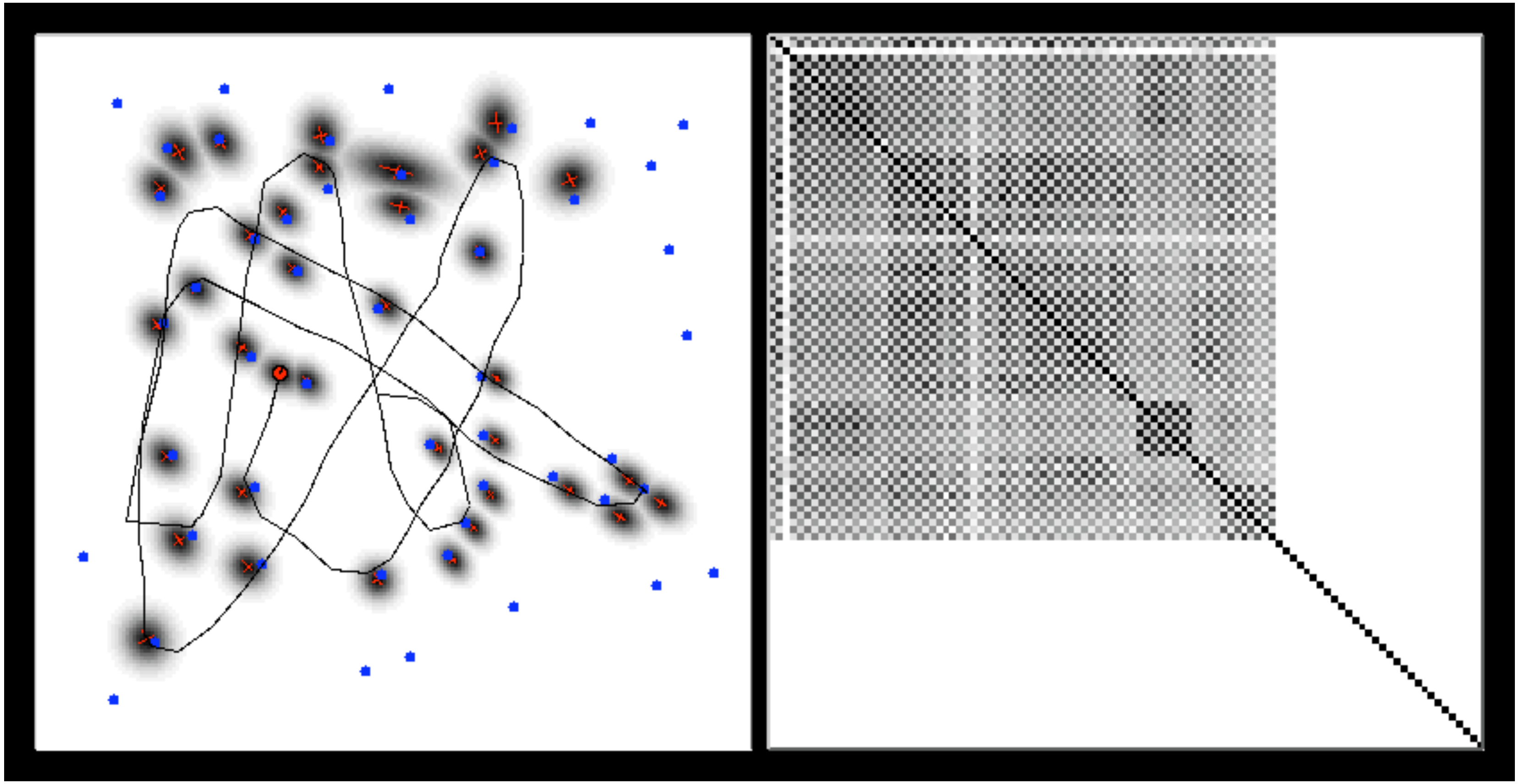
$$\mathbf{x}_k = \begin{bmatrix} \mathbf{x}_R \\ \mathbf{m}_1 \\ \mathbf{m}_2 \\ \vdots \\ \mathbf{m}_n \\ \mathbf{m}_{n+1} \end{bmatrix}_k \qquad C_k = \begin{bmatrix} C_R & C_{RM_1} & C_{RM_2} & \cdots & C_{RM_n} & C_{RM_{n+1}} \\ C_{M_1R} & C_{M_1} & C_{M_1M_2} & \cdots & C_{M_1M_n} & C_{M_1M_{n+1}} \\ C_{M_2R} & C_{M_2M_1} & C_{M_2} & \cdots & C_{M_2M_n} & C_{M_2M_{n+1}} \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ C_{M_nR} & C_{M_nM_1} & C_{M_nM_2} & \cdots & C_{M_n} & C_{M_nM_{n+1}} \\ C_{M_{n+1}R} & C_{M_{n+1}M_1} & C_{M_{n+1}M_2} & \cdots & C_{M_{n+1}M_n} & C_{M_{n+1}} \end{bmatrix}_k$$

EKF SLAM



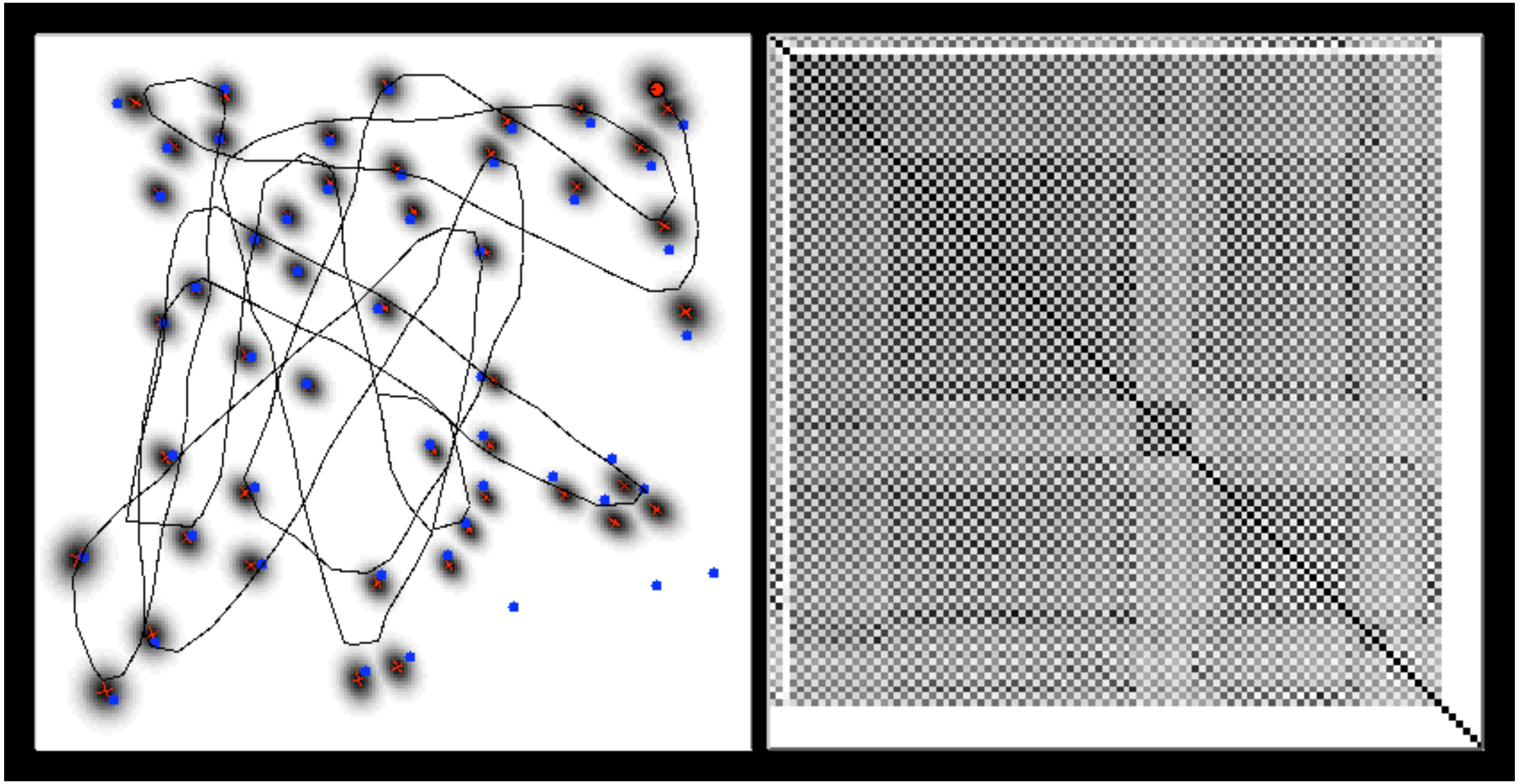
Map Correlation matrix

EKF SLAM



Map Correlation matrix

EKF SLAM



Map Correlation matrix

EKF SLAM: Correlations Matter

- What if we **neglected** cross-correlations?

$$C_k = \begin{bmatrix} C_R & 0 & \cdots & 0 \\ 0 & C_{M_1} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & C_{M_n} \end{bmatrix}_k$$

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

$$C_{M_iM_{i+1}} = \mathbf{0}_{2\times 2}$$

→ Landmark and robot uncertainties would become overly optimistic

→ Validation gates for matching too small

→ Data association would fail

→ Multiple map entries of the same landmark

→ Inconsistent map

Video

Error Propagation (cont.)

Want to derive:

$$C_{YZ} = A\, C_{XZ}$$

In words: how is the **cross-correlation** C_{XZ} between two normally distributed RVs X and Z with moments x, C_X and z, C_Z **affected** by a **linear transform** of X of the form

$$\mathbf{y} = A\,\mathbf{x} + B \quad ?$$

We recall that the following holds:

$$C_Y = A\, C_X\, A^T$$

Error Propagation (cont.)

We **augment** the linear mapping by the **variable of interest**

$$\begin{bmatrix} \mathbf{y} \\ \mathbf{z} \end{bmatrix} = \begin{bmatrix} A & 0 \\ 0 & I \end{bmatrix} \begin{bmatrix} \mathbf{x} \\ \mathbf{z} \end{bmatrix} + \begin{bmatrix} B \\ 0 \end{bmatrix}$$

Note that this implements

$$\mathbf{y} = A\,\mathbf{x} + B$$

$$\mathbf{z} = \mathbf{z}$$

Error Propagation (cont.)

Renaming the variables of the augmented system

$$\mathbf{x}' = [\mathbf{x} \quad \mathbf{z}]^T \qquad\qquad \mathbf{y}' = [\mathbf{y} \quad \mathbf{z}]^T$$

gives $\mathbf{y}' = A' \, \mathbf{x}' + B'$ with the augmented covariance matrices

$$C_{Y'} = \begin{bmatrix} C_Y & C_{YZ} \\ C_{ZY} & C_Z \end{bmatrix} \quad C_{X'} = \begin{bmatrix} C_X & C_{XZ} \\ C_{ZX} & C_Z \end{bmatrix}$$

The augmented covariance matrix is again given by

$$C_{Y'} = A' \, C_{X'} \, A'^T$$

Error Propagation (cont.)

Resubstitution yields

$$\begin{aligned} C_{Y'} &= \begin{bmatrix} A & 0 \\ 0 & I \end{bmatrix} \begin{bmatrix} C_X & C_{XZ} \\ C_{ZX} & C_Z \end{bmatrix} \begin{bmatrix} A^T & 0 \\ 0 & I \end{bmatrix} \\ &= \begin{bmatrix} A\,C_X & A\,C_{XZ} \\ C_{ZX} & C_Z \end{bmatrix} \begin{bmatrix} A^T & 0 \\ 0 & I \end{bmatrix} \\ &= \begin{bmatrix} A\,C_X\,A^T & A\,C_{XZ} \\ C_{ZX}\,A^T & C_Z \end{bmatrix} \end{aligned}$$

Thus:

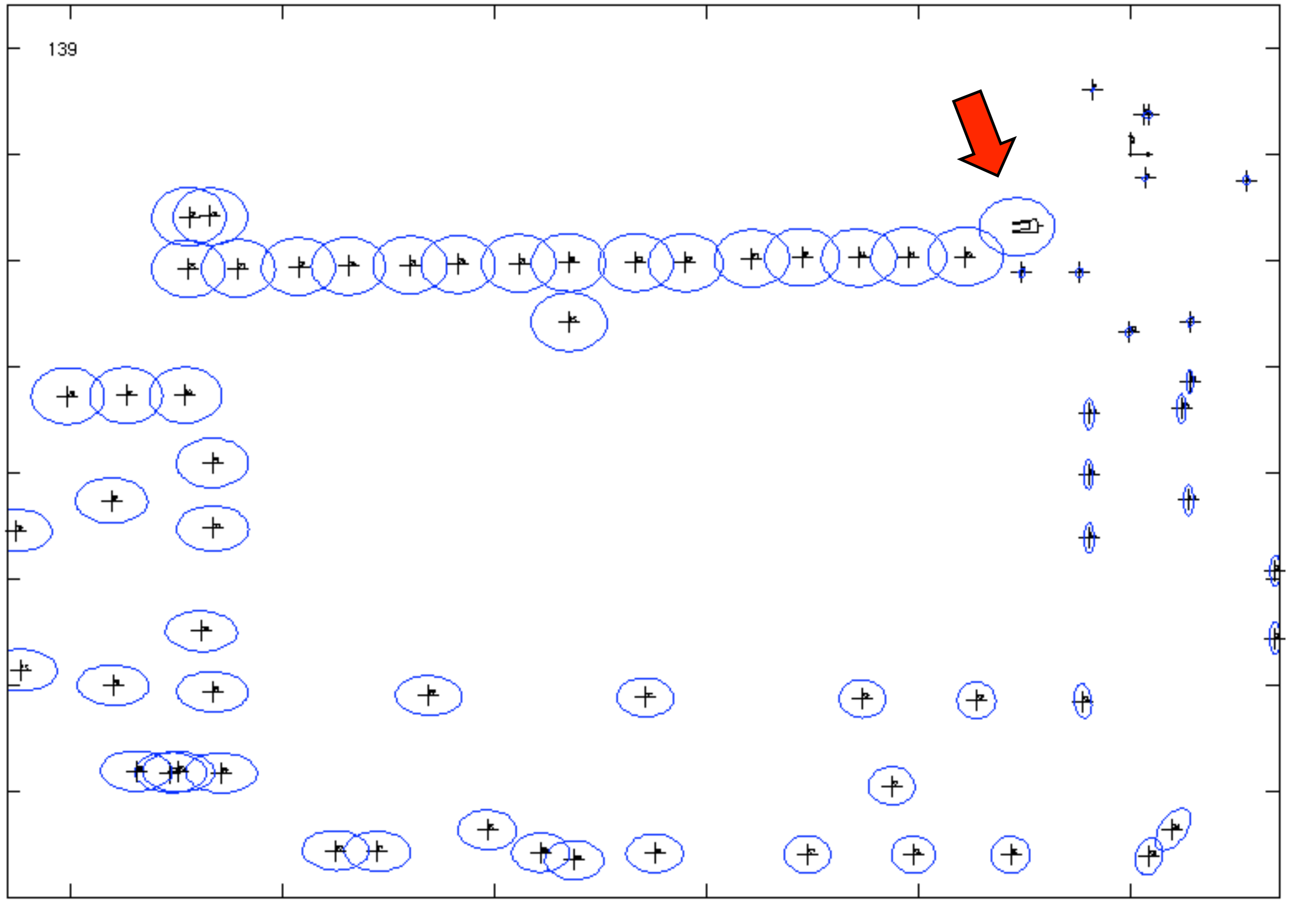
$$C_{YZ} = A\,C_{XZ}$$

SLAM: Loop Closure

- Loop closure is the problem of **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
- Uncertainties **collapse** after a loop closure (whether the closure was correct or not)

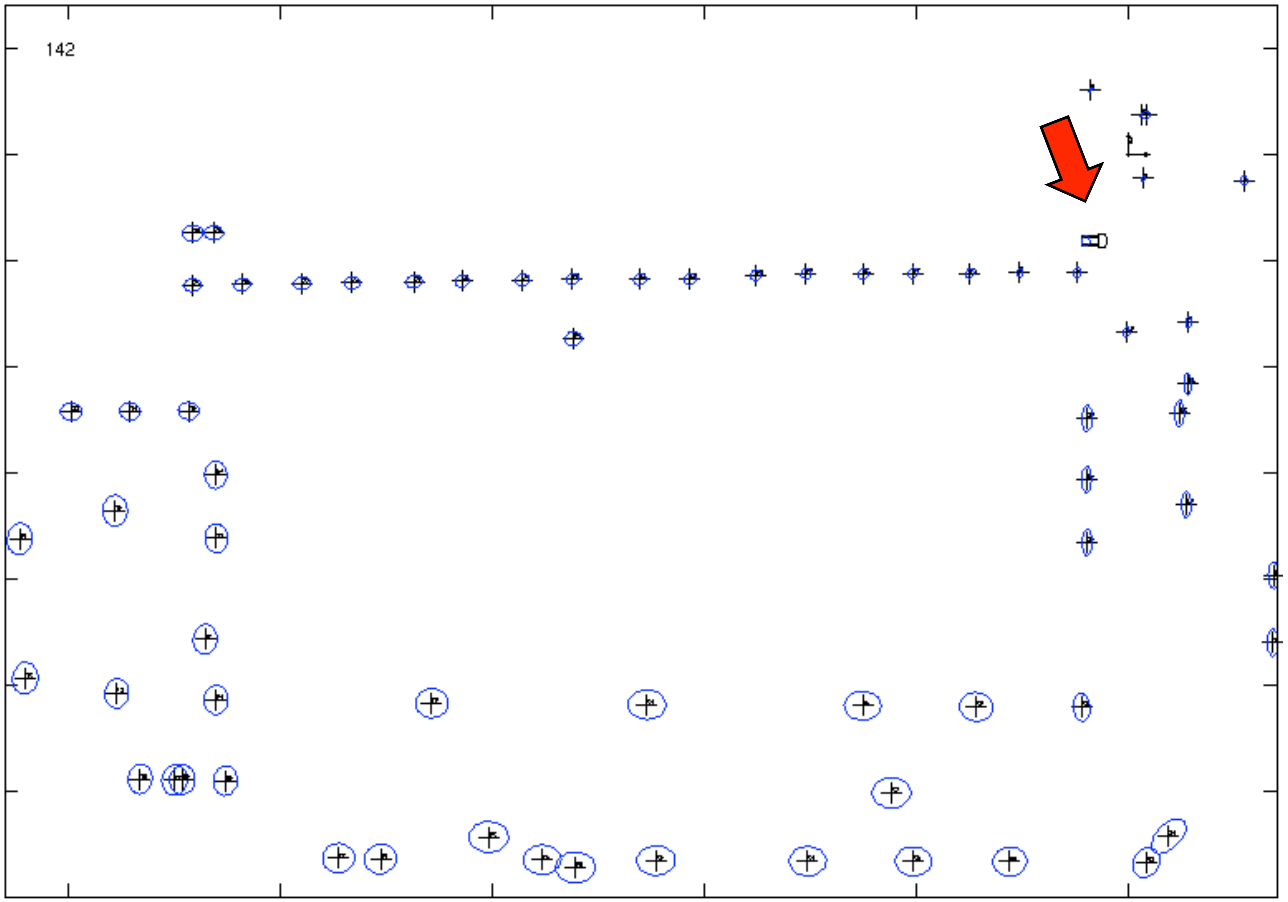
SLAM: Loop Closure

- Before loop closure



SLAM: Loop Closure

- After loop closure



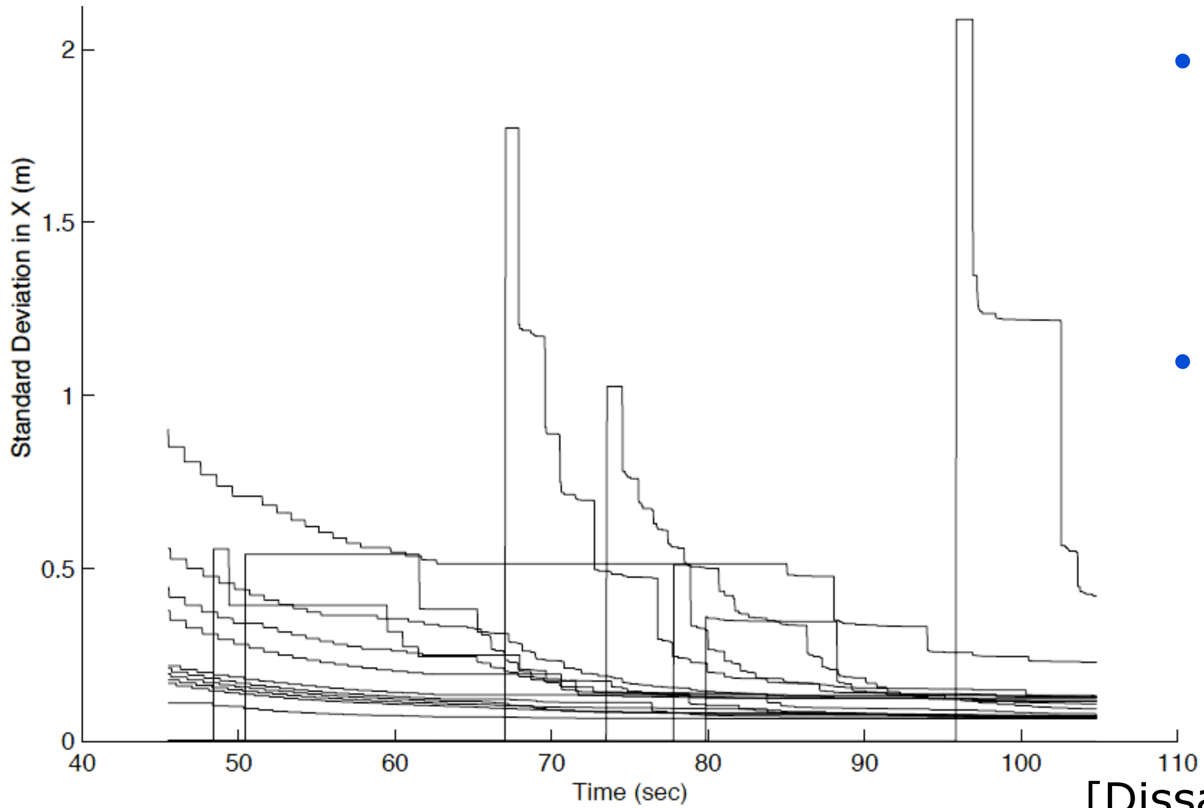
SLAM: Loop Closure

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

→ 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

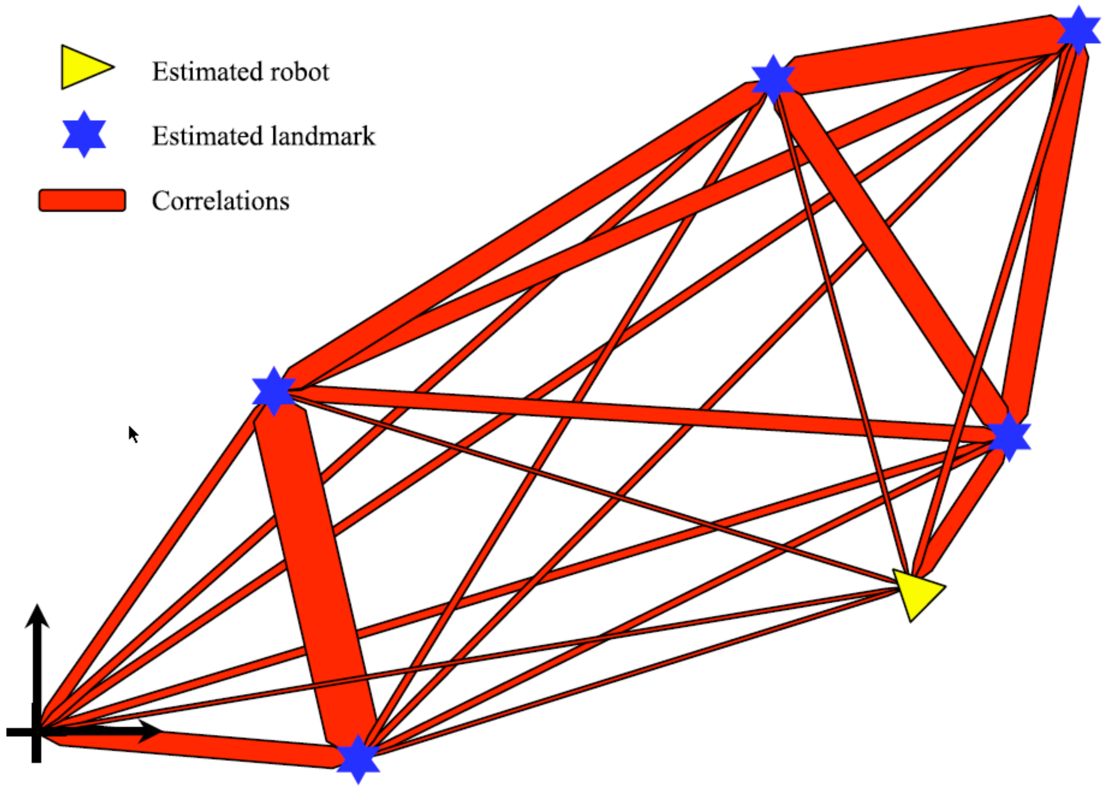


- When a new land-mark is initialized, its **uncertainty** is **maximal**
- Landmark uncer-tainty **decreases monotonically** with each new observation

[Dissanayake et al., 2001]

KF-SLAM Properties (Linear Case)

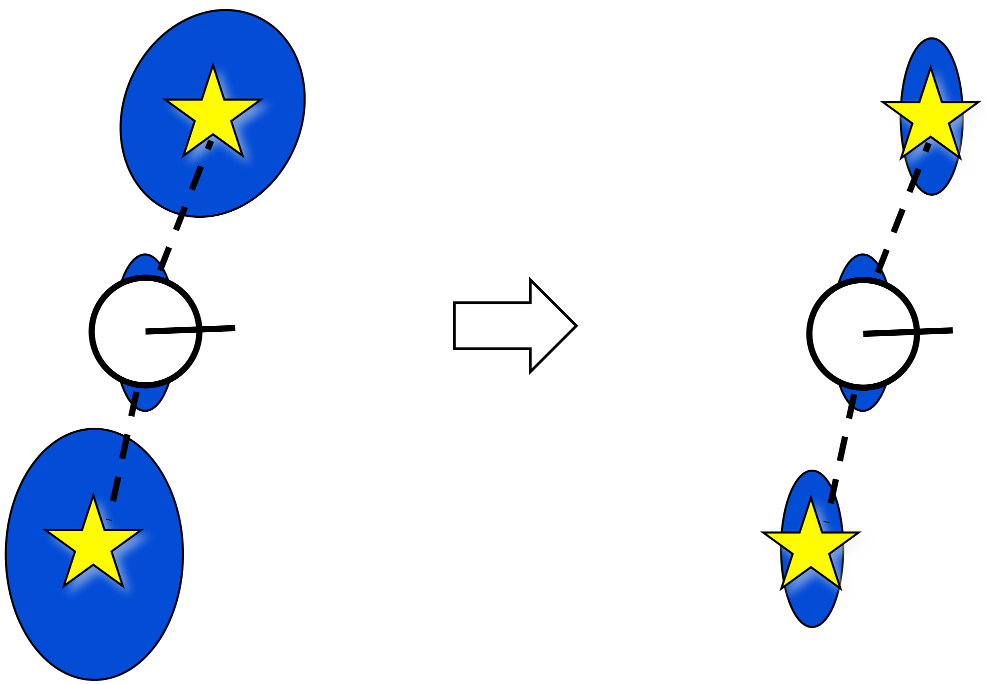
- In the limit, the landmark estimates become **fully correlated**



[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]

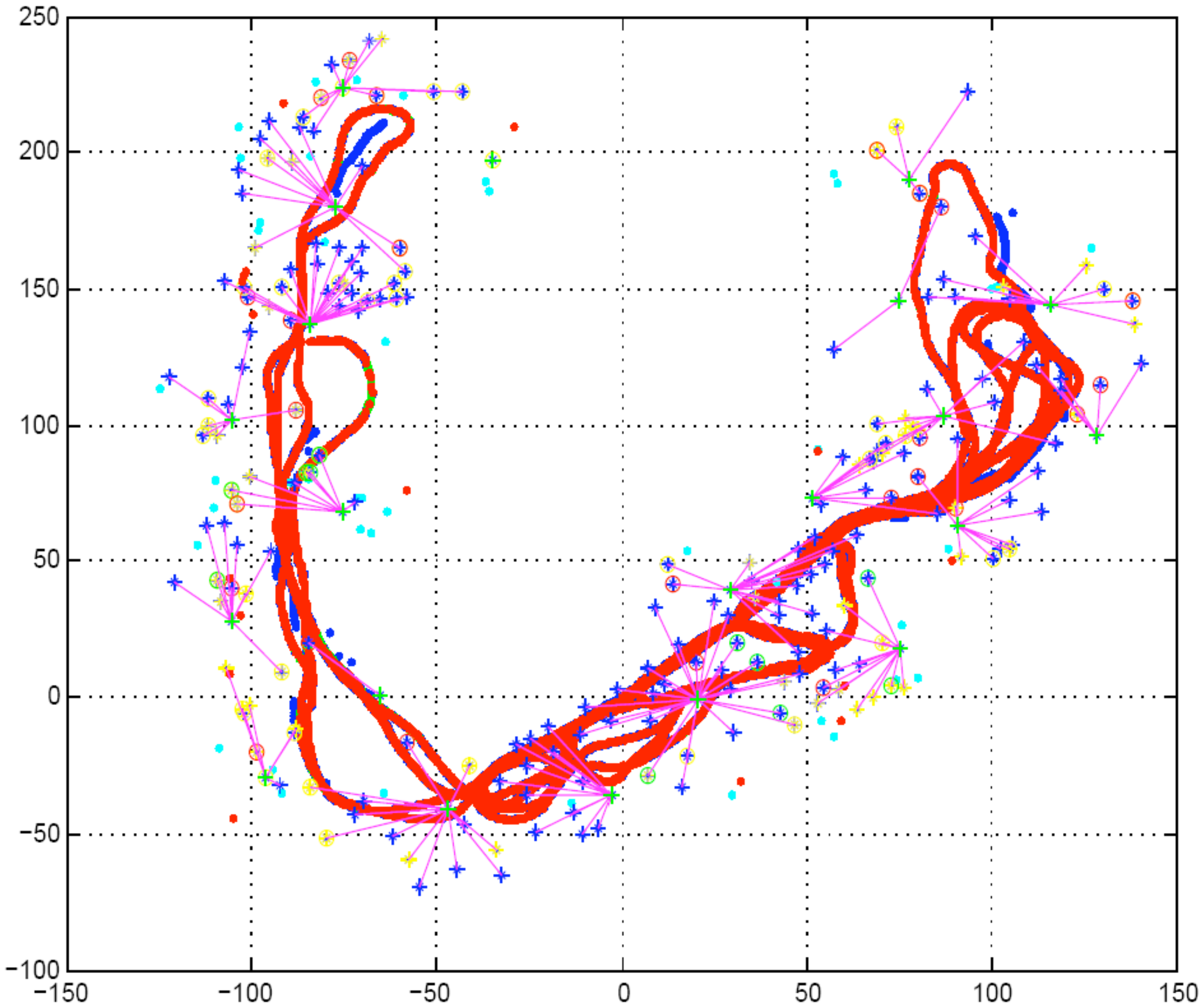
EKF SLAM Example: Victoria Park

Syndey, Australia

Victoria Park: Data Acquisition

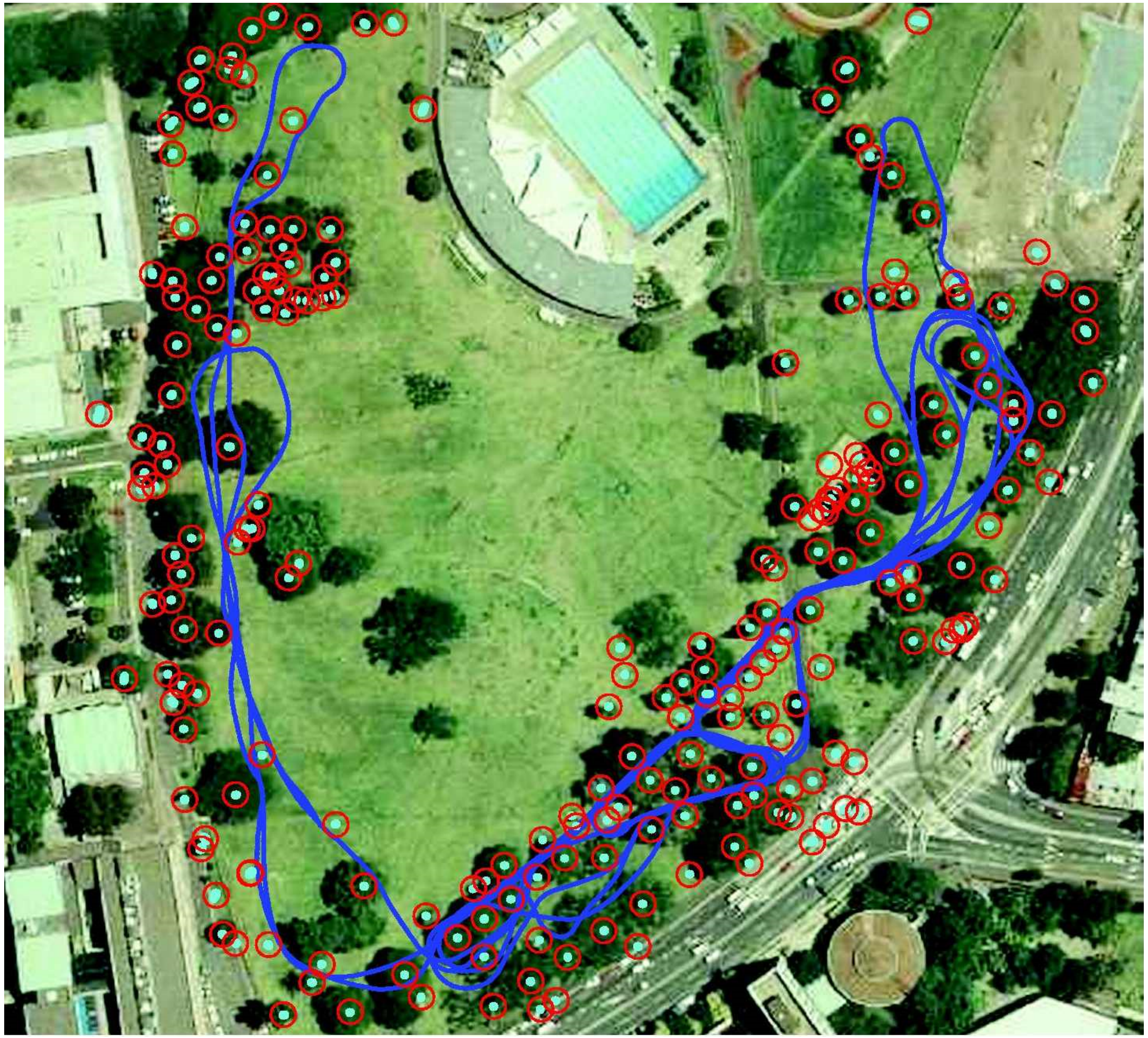
[courtesy by E. Nebot]

Victoria Park: Estimated Trajectory



[courtesy by E. Nebot]

Victoria Park: Landmarks

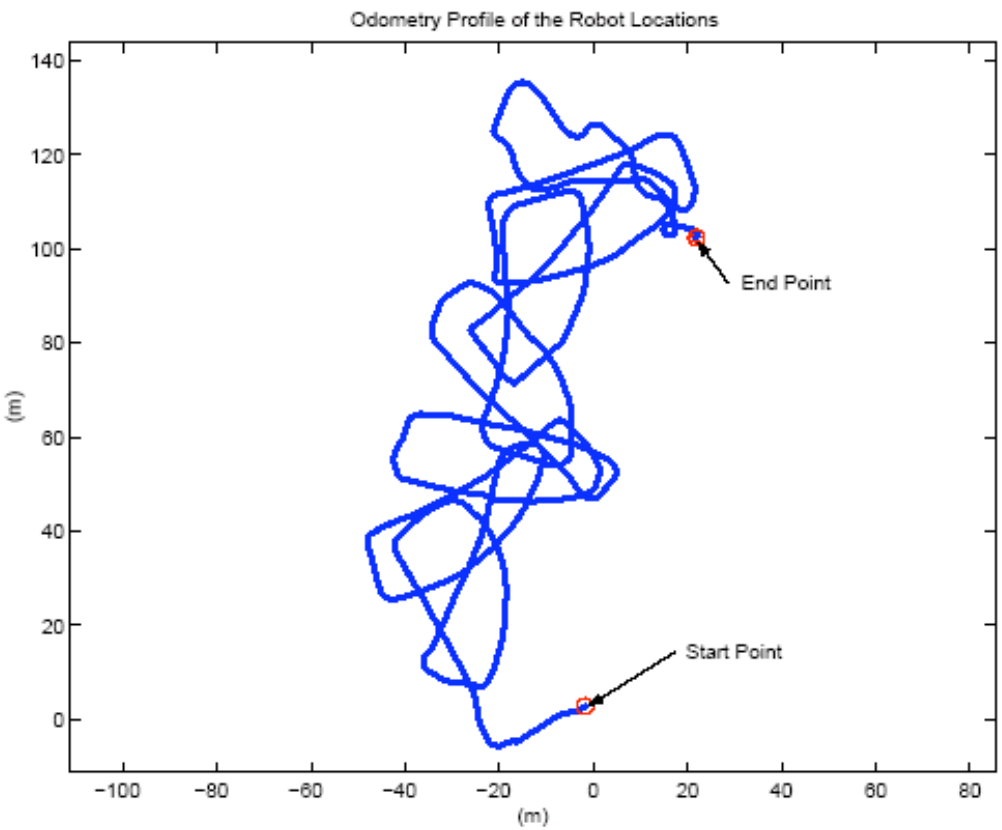


[courtesy by E. Nebot]

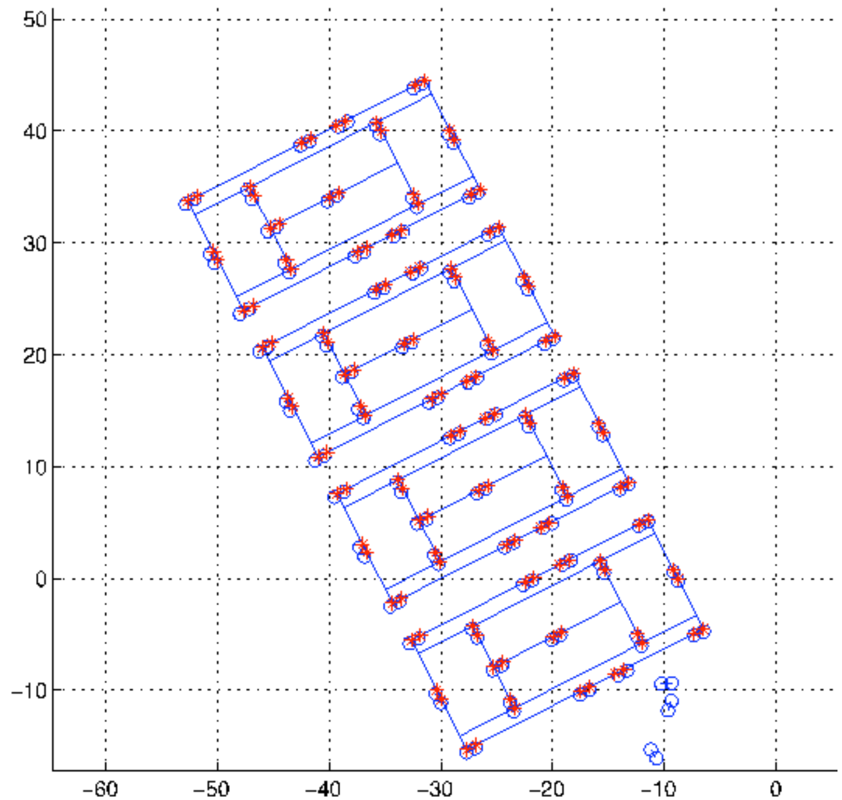
EKF SLAM Example: Tennis Court

[courtesy by J. Leonard]

EKF SLAM Example: Tennis Court



odometry

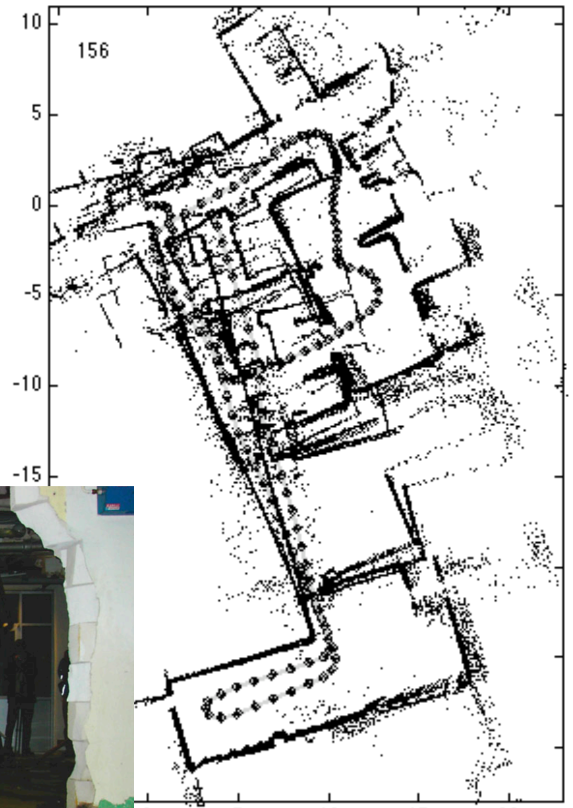


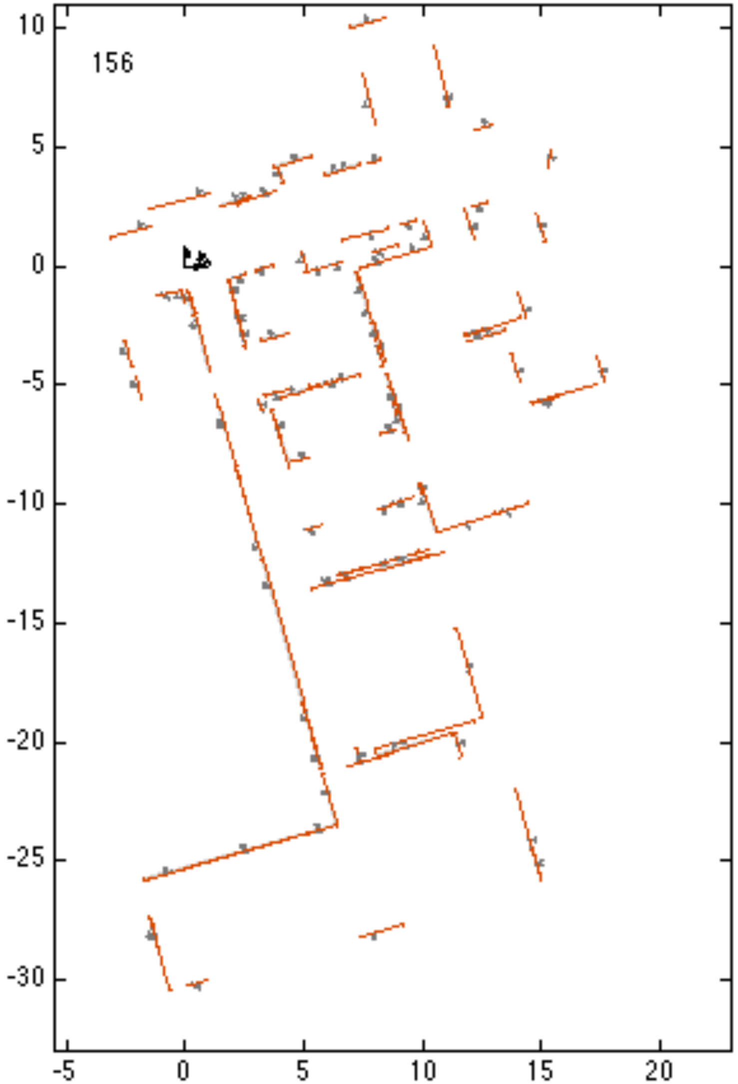
estimated trajectory

[courtesy by John Leonard]

EKF SLAM Example: Line Features

- KTH Bakery Data Set

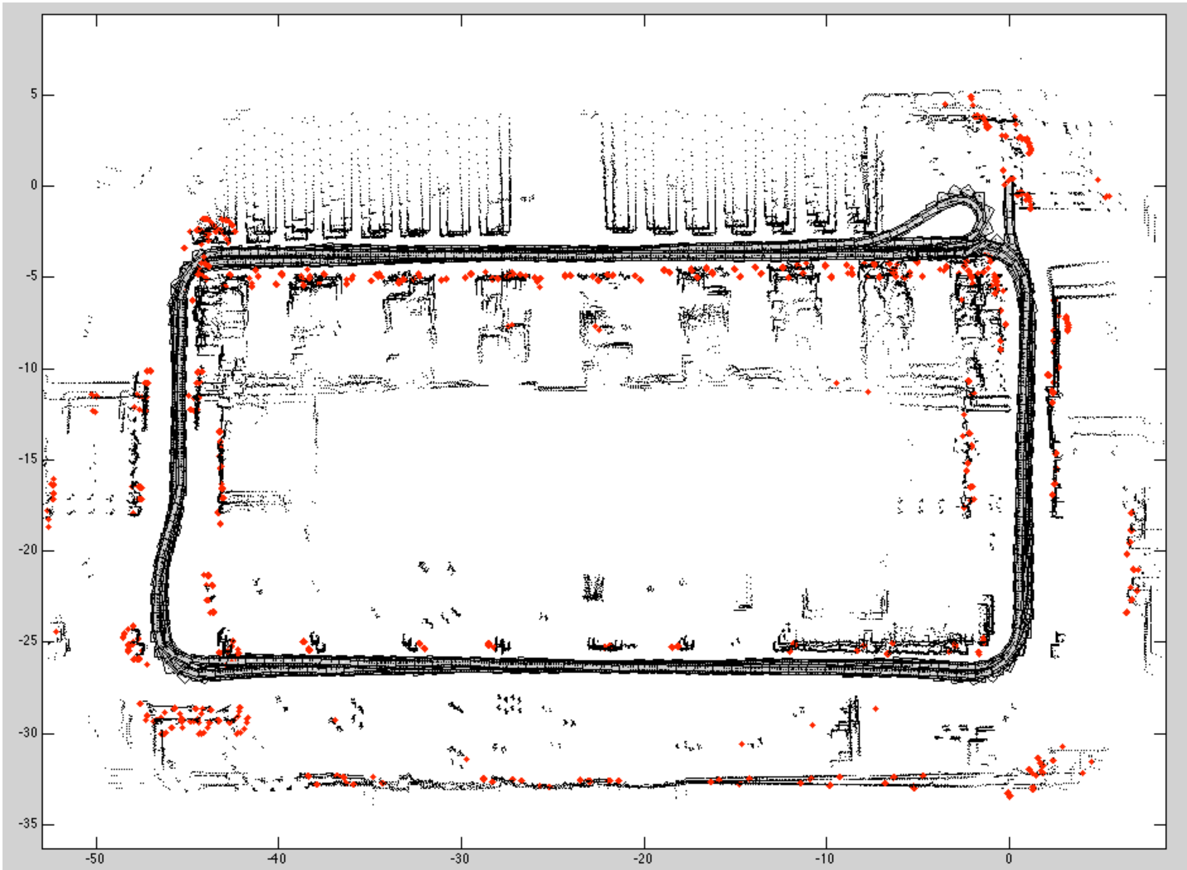




[Wulf et al., ICRA 04]

EKF SLAM Example: AGV

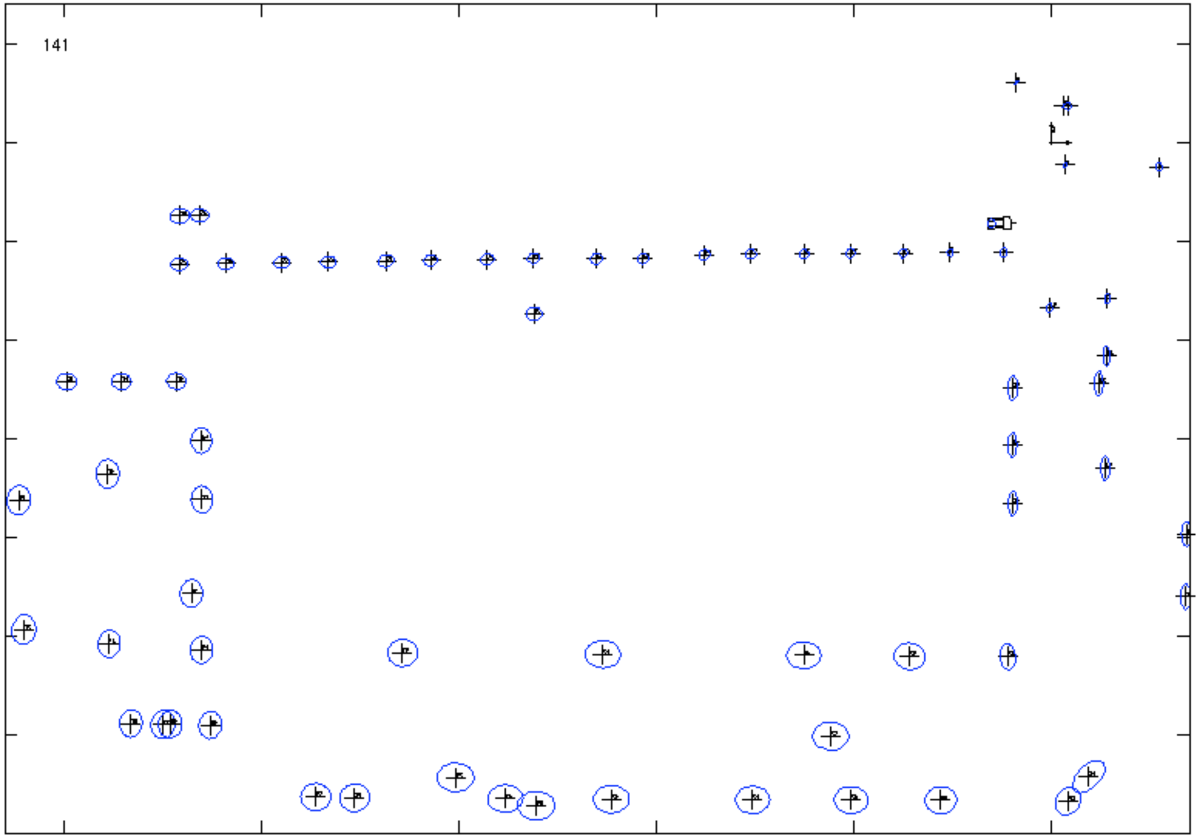
- Pick-and-Place AGV at Geiger AG, Ludwigsburg (Project by LogObject/Nurobot)



[courtesy by LogObject/Nurobot]

EKF SLAM Example: AGV

- Pick-and-Place AGV at Geiger AG, Ludwigsburg (Project by LogObject/Nurobot)



[courtesy by LogObject/Nurobot]

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**: $O(n^2)$

Problem: becomes computationally intractable for large maps!

➔ Approaches exist that make EKF-SLAM amortized $O(n)$ / $O(n^2)$ / $O(n^2)$
D&C SLAM *[Paz et al., 2006]*

SLAM Techniques

- EKF SLAM
- FastSLAM
- Graphical SLAM
- Topological SLAM
  (mainly place recognition)
- Scan Matching / Visual Odometry
  (only locally consistent maps)
- Approximations for SLAM: Local submaps, Sparse extended information filters, Sparse links, Thin junction tree filters, etc.

EKF-SLAM: Summary

- **Convergence proof** for linear case!
- **Can diverge** if nonlinearities are large (and the reality **is** nonlinear...)
- First-order error propagation becomes a **problem**. Uncertainties large with respect to the degree of non-linearity
- Has been **successfully applied** in **medium-scale environments**
- Approximations **reduce** the **computational complexity**

Approximations for SLAM

- Local submaps
  [Leonard et al.99, Bosse et al. 02, Newman et al. 03]

- Sparse links (correlations)
  [Lu & Milios 97, Guivant & Nebot 01]

- Sparse extended information filters
  [Frese et al. 01, Thrun et al. 02]

- Thin junction tree filters
  [Paskin 03]

- Rao-Blackwellisation (FastSLAM)
  [Murphy 99, Montemerlo et al. 02, Eliazar et al. 03, Haehnel et al. 03]