Probabilistic Robotics

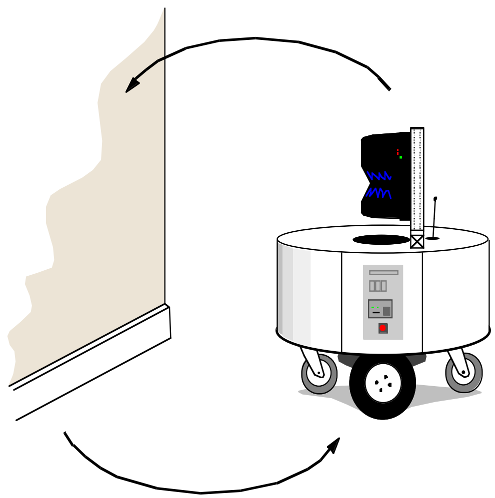
SLAM

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:

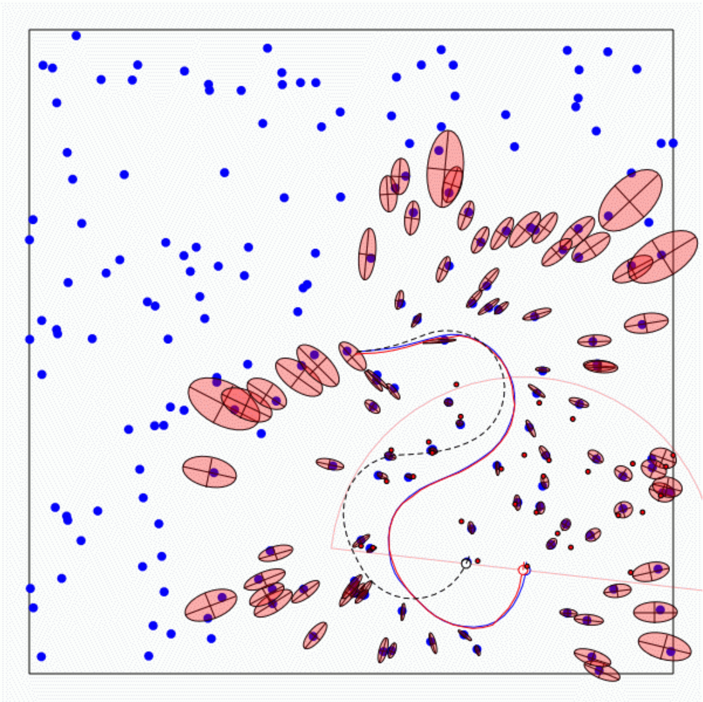
- The robot's controls

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

- Relative observations

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

Wanted:



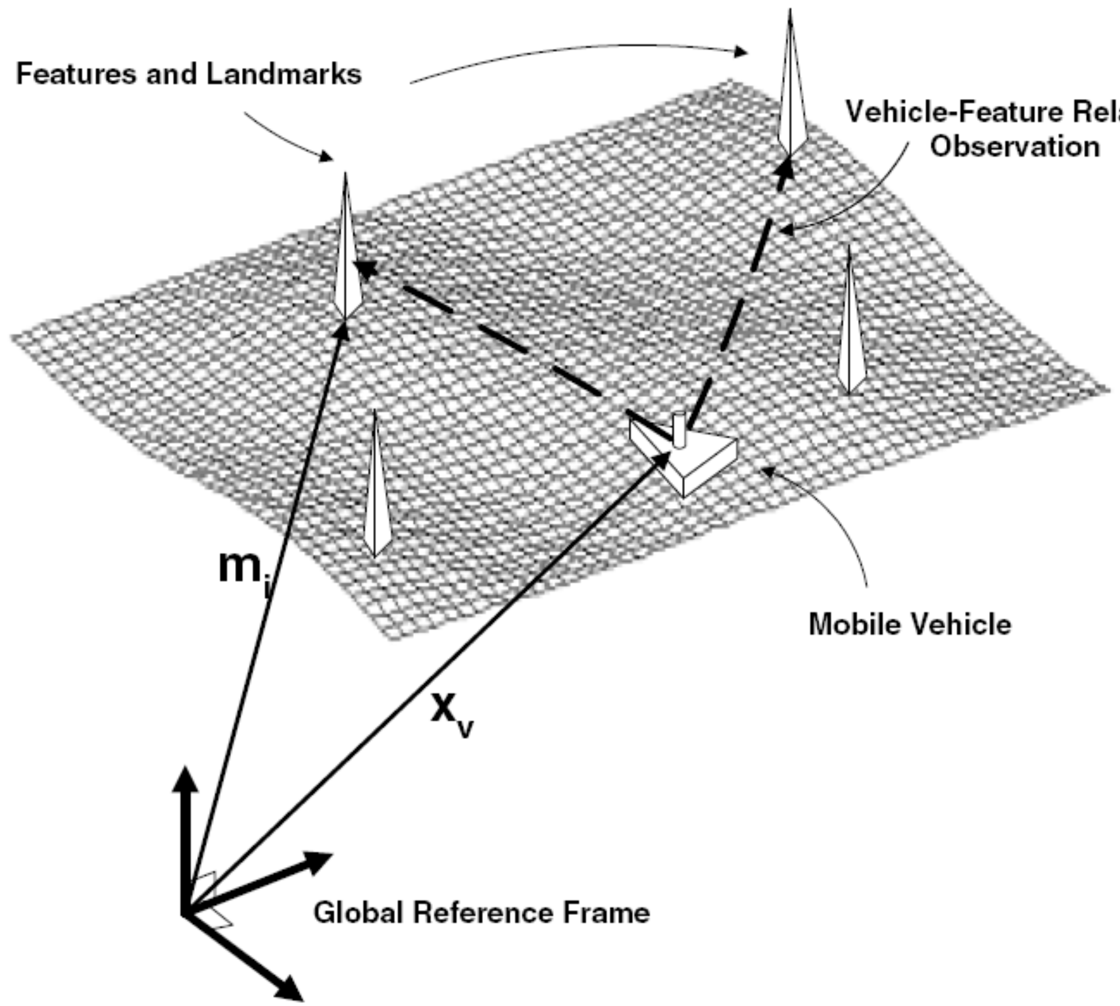
- Map of features

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

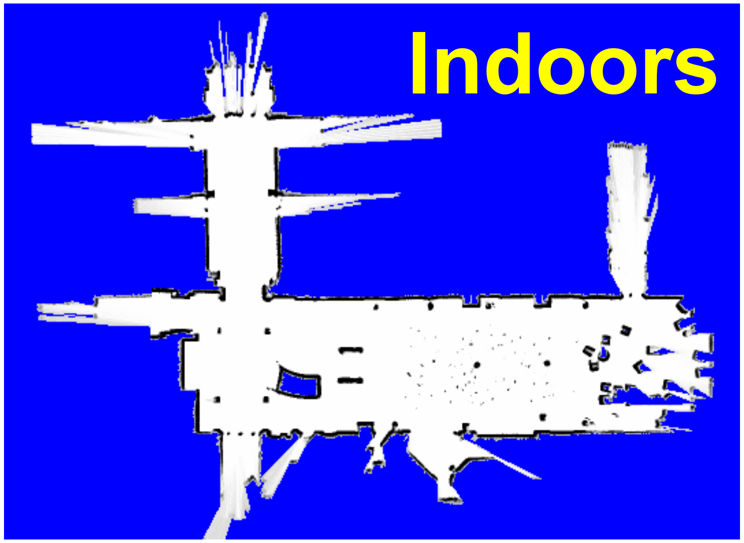
- Path of the robot

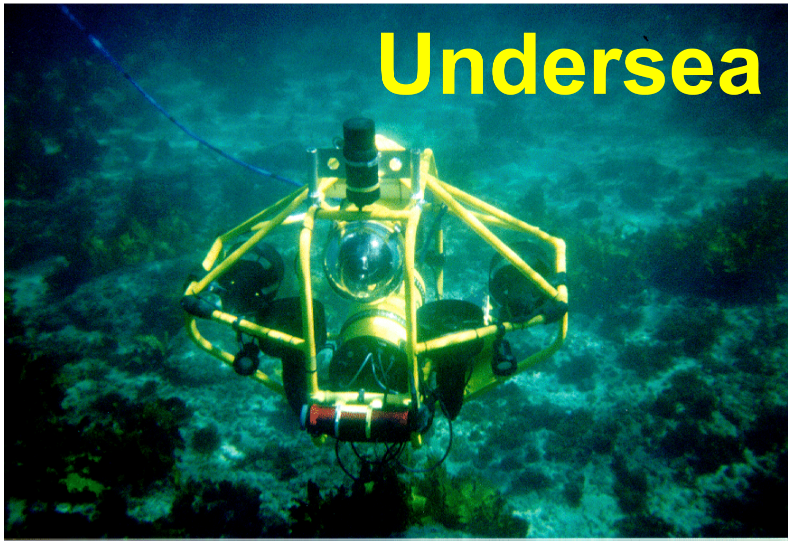
  $$\mathbf{X}_{0:k} = \{\mathbf{x}_0, \mathbf{x}_1, \cdots, \mathbf{x}_k\}$$

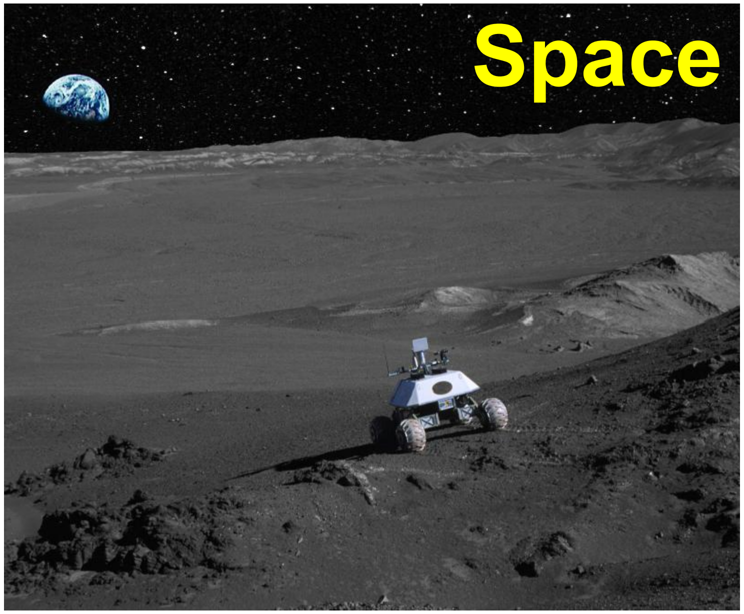
Structure of the Landmark-based SLAM-Problem



SLAM Applications

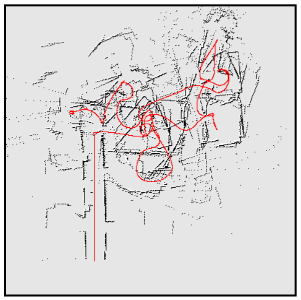
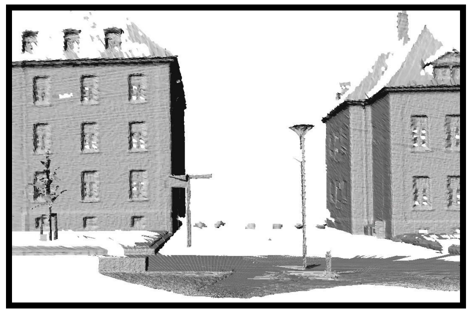
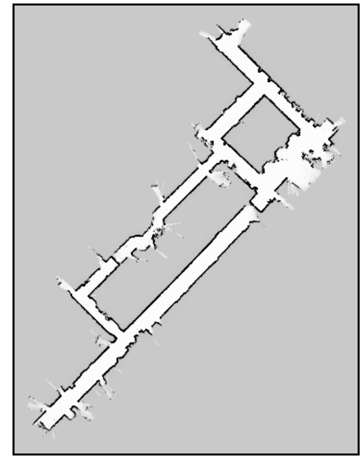






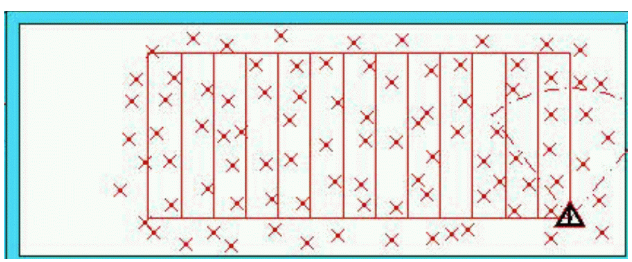
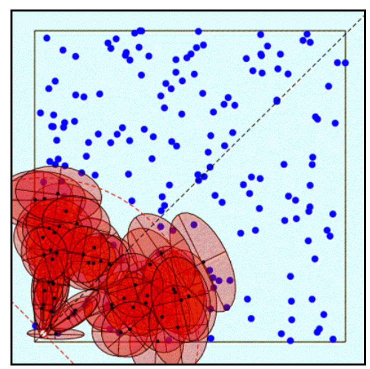
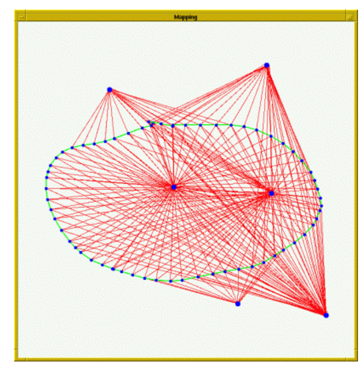
Representations

- Grid maps or scans

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

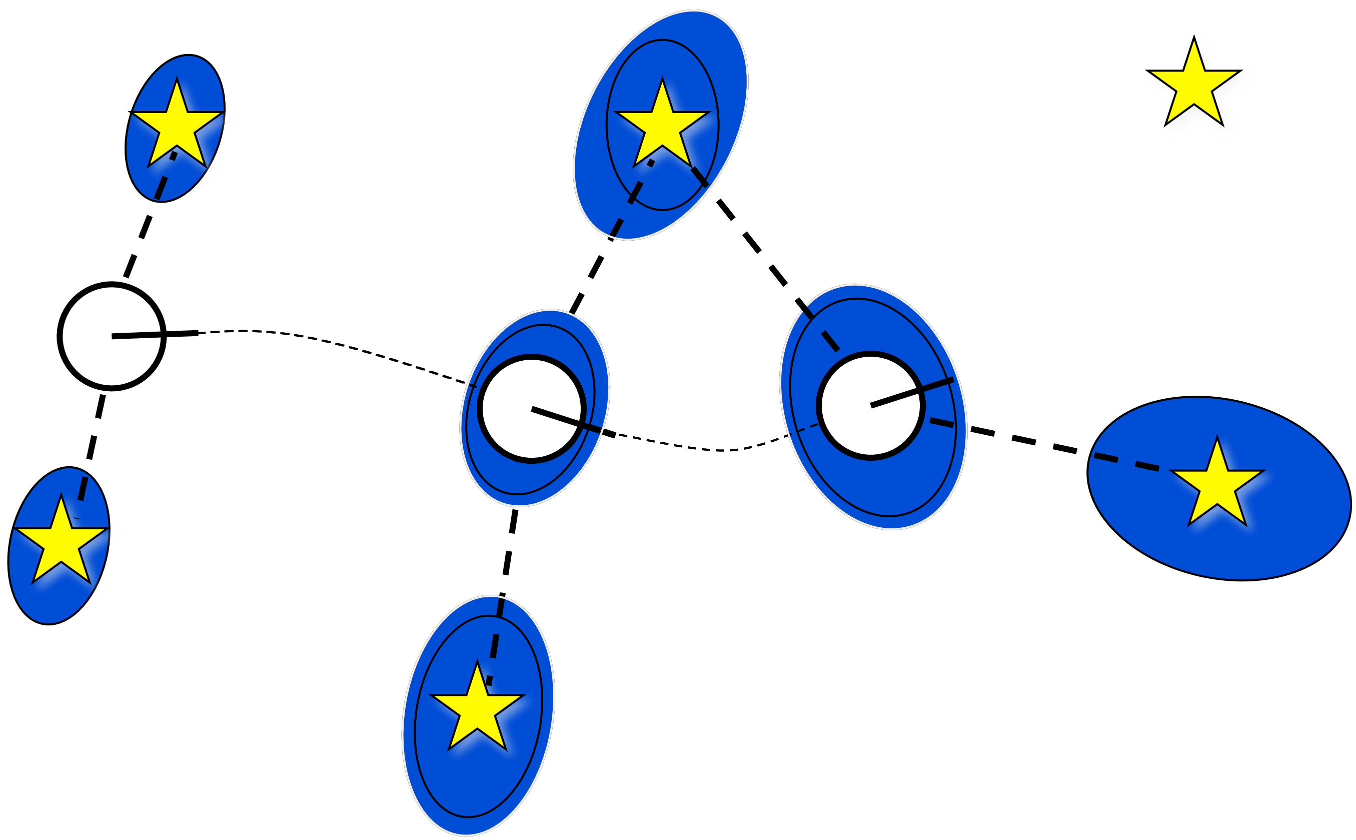
- Landmark-based

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

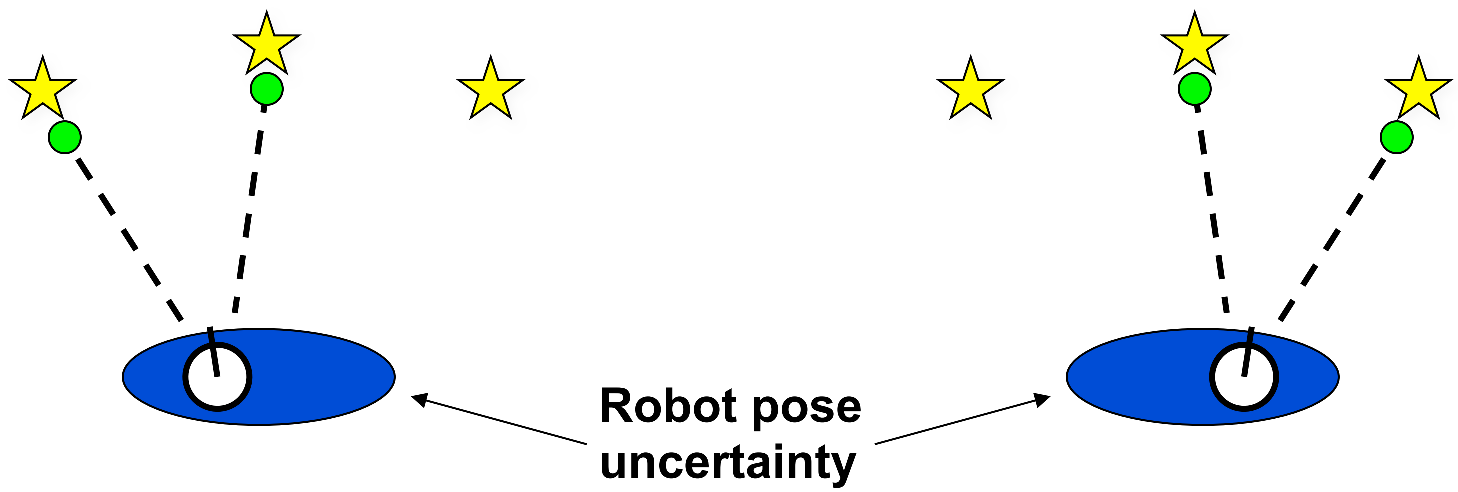
Why is SLAM a hard problem?

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!

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

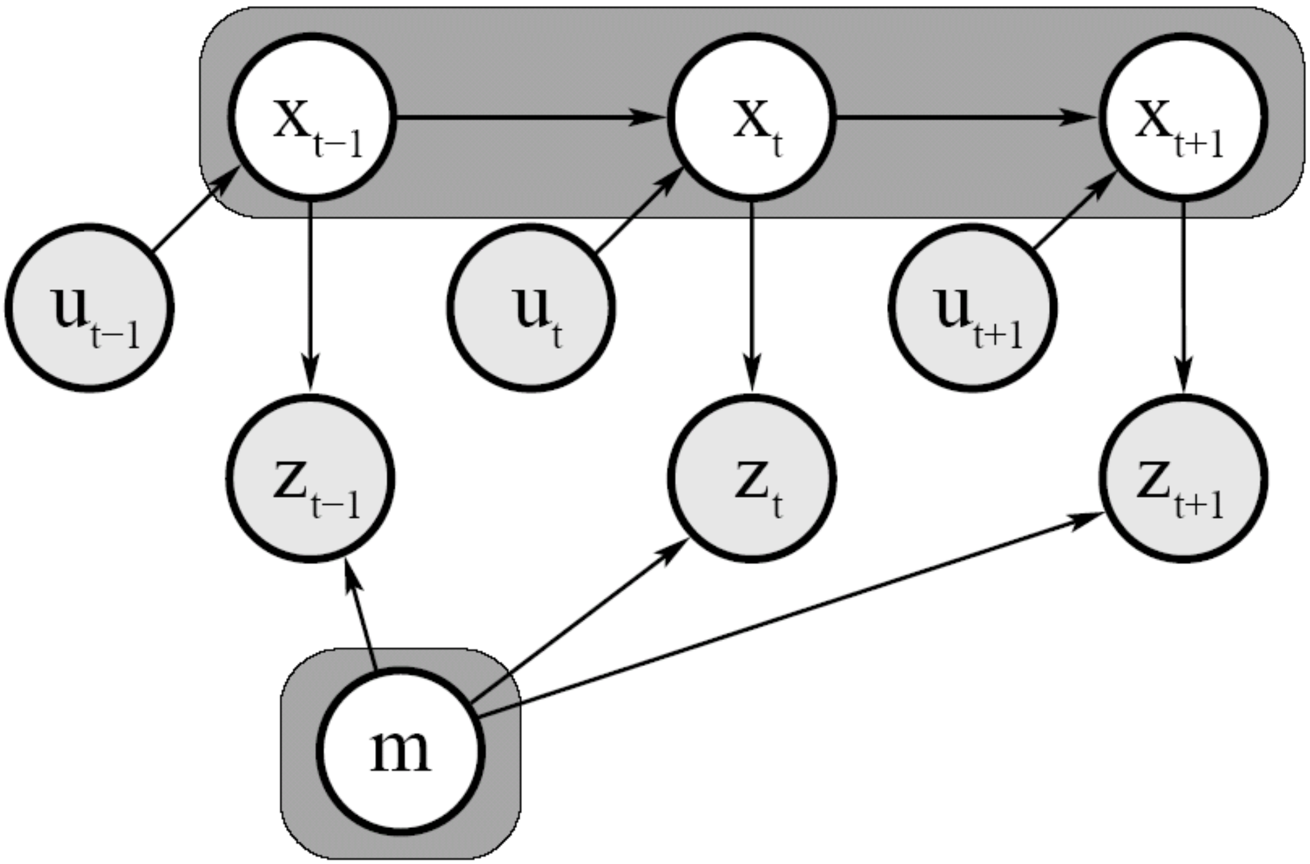
- Online SLAM:

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

Integrations (marginalization) typically done one at a time

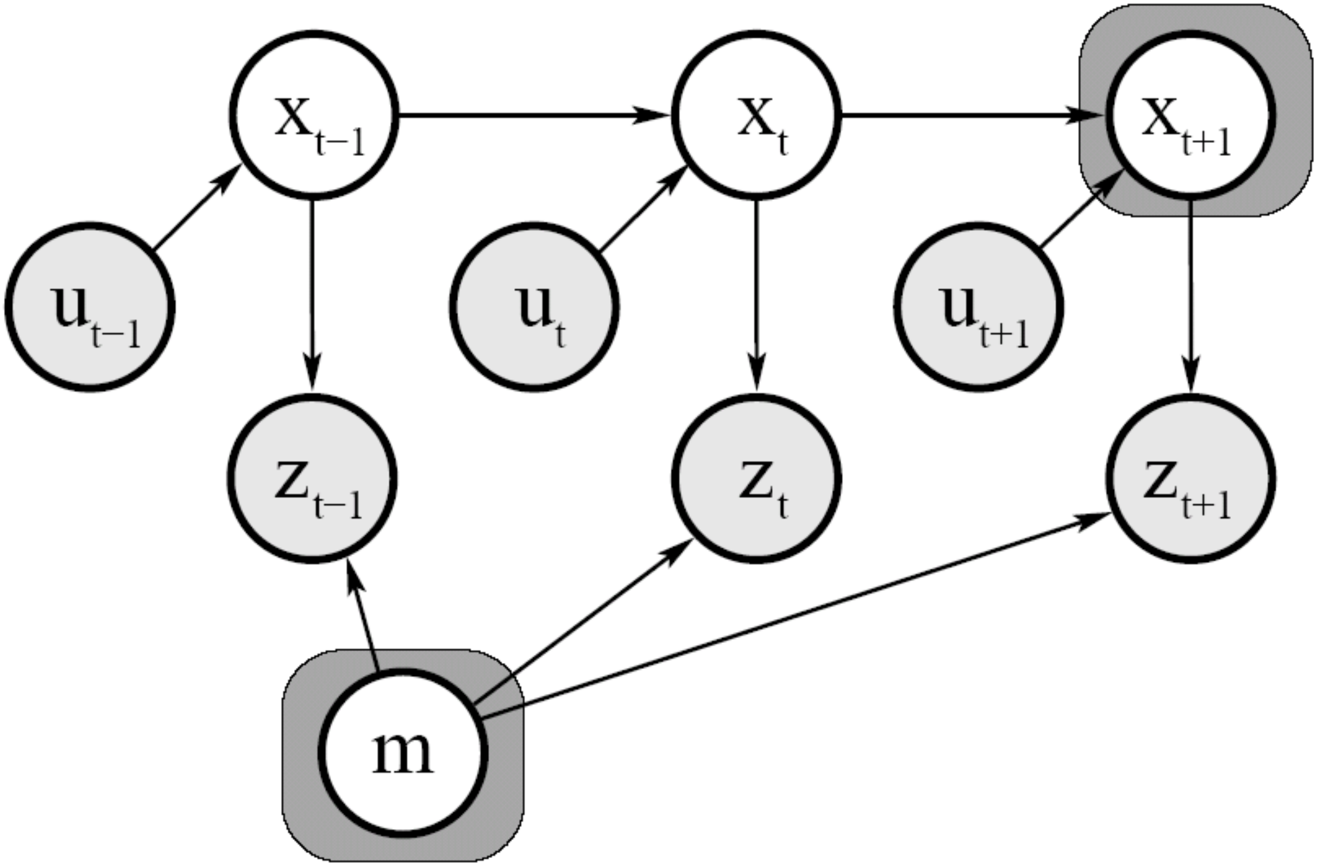
Estimates most recent pose and map!

Graphical Model of Full SLAM:



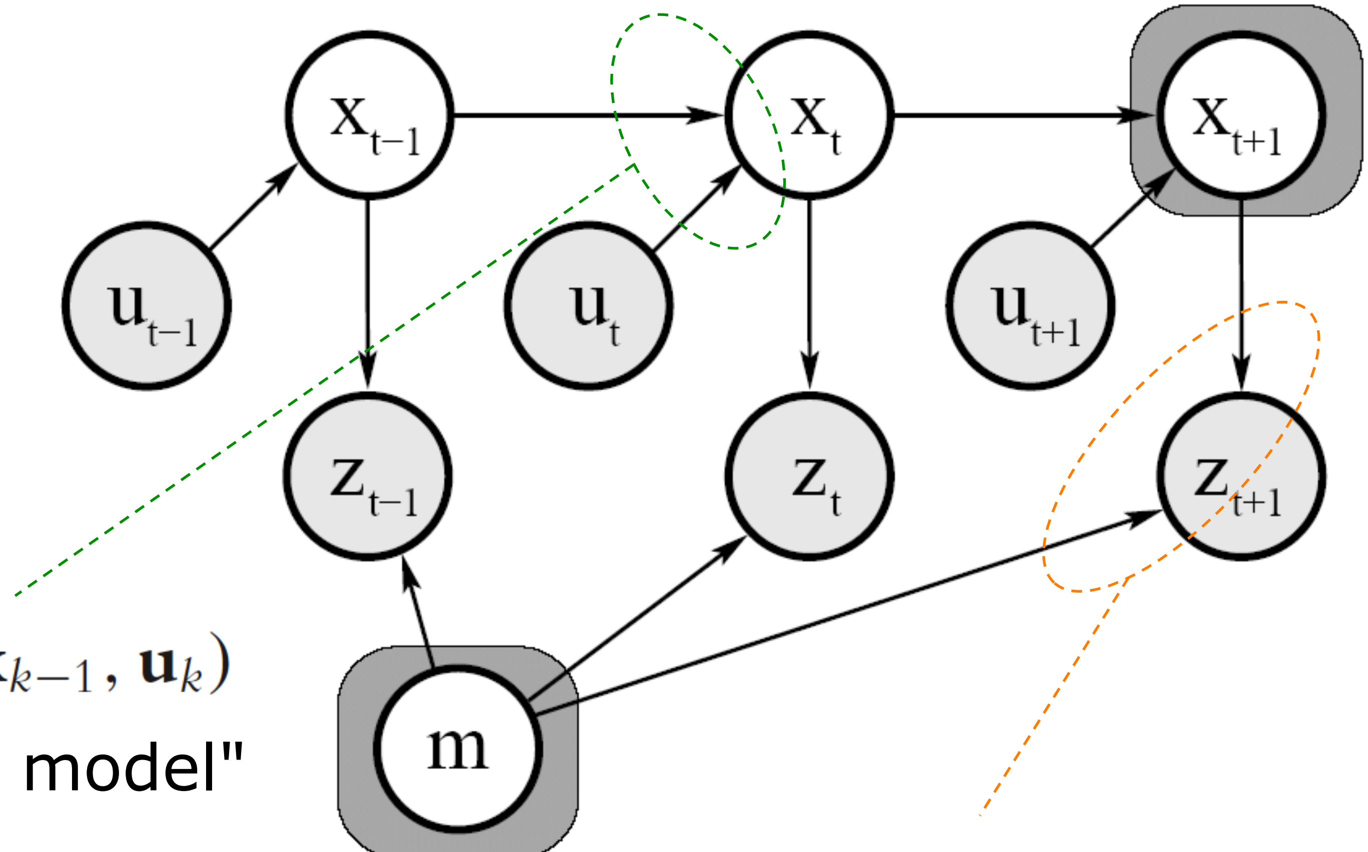
$$p(x_{1:t}, m \mid z_{1:t}, u_{1:t})$$

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"

Techniques for Generating Consistent Maps

- Scan matching
- EKF SLAM
- Fast-SLAM
- Probabilistic mapping with a single map and a posterior about poses
  Mapping + Localization
- Graph-SLAM, SEIFs

Scan Matching

Maximize the likelihood of the i-th pose and map relative to the (i-1)-th pose and map.

$$\hat{x}_t = \underset{x_t}{\arg\max} \left\{ p(z_t \mid x_t, \hat{m}^{[t-1]}) \cdot p(x_t \mid u_{t-1}, \hat{x}_{t-1}) \right\}$$

current measurement

map constructed so far

robot motion

Calculate the map $\hat{m}^{[t]}$ according to “mapping with known poses” based on the poses and observations.

Kalman Filter 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

Extended Kalman Filter

- Previously Extended Kalman Filter
line features detected from range data

- Now review extended Kalman Filter for landmark model

- Digression – (with slightly different notation)

Kalman Filter Components
(also known as: Way Too Many Variables…)

Linear discrete time dynamic system (motion model)

State Control input Process noise

$$x_{t+1} = F_t x_t + B_t u_t + G_t w_t$$

State transition function

Control input function

Noise input function with covariance Q

Measurement equation (sensor model)

Sensor reading State Sensor noise with covariance R

$$z_{t+1} = H_{t+1} x_{t+1} + n_{t+1}$$

Sensor function

Note: Write these down!!!

At last! The Kalman Filter...

Propagation (motion model):

$$\hat{x}_{t+1/t} = F_t \hat{x}_{t/t} + B_t u_t$$

$$P_{t+1/t} = F_t P_{t/t} F_t^T + G_t Q_t G_t^T$$

Update (sensor model):

$$\hat{z}_{t+1} = H_{t+1} \hat{x}_{t+1/t}$$

$$r_{t+1} = z_{t+1} - \hat{z}_{t+1}$$

$$S_{t+1} = H_{t+1} P_{t+1/t} H_{t+1}{}^T + R_{t+1}$$

$$K_{t+1} = P_{t+1/t} H_{t+1}{}^T S_{t+1}{}^{-1}$$

$$\hat{x}_{t+1/t+1} = \hat{x}_{t+1/t} + K_{t+1} r_{t+1}$$

$$P_{t+1/t+1} = P_{t+1/t} - P_{t+1/t} H_{t+1}{}^T S_{t+1}{}^{-1} H_{t+1} P_{t+1/t}$$

In words …

Propagation (motion model):

$$\hat{x}_{t+1/t} = F_t \hat{x}_{t/t} + B_t u_t$$

- State estimate is updated from system dynamics

$$P_{t+1/t} = F_t P_{t/t} F_t^T + G_t Q_t G_t^T$$

- Uncertainty estimate *GROWS*

Update (sensor model):

$$\hat{z}_{t+1} = H_{t+1} \hat{x}_{t+1/t}$$

- Compute expected value of sensor reading

$$r_{t+1} = z_{t+1} - \hat{z}_{t+1}$$

- Compute the difference between expected and "true"

$$S_{t+1} = H_{t+1} P_{t+1/t} H_{t+1}{}^T + R_{t+1}$$

- Compute covariance of sensor reading

$$K_{t+1} = P_{t+1/t} H_{t+1}{}^T S_{t+1}{}^{-1}$$

- Compute the Kalman Gain (how much to correct est.)

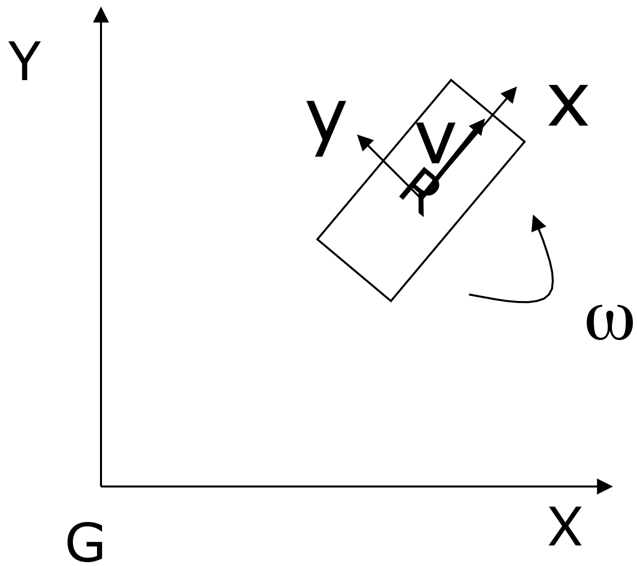
$$\hat{x}_{t+1/t+1} = \hat{x}_{t+1/t} + K_{t+1} r_{t+1}$$

- Multiply residual times gain to correct state estimate

$$P_{t+1/t+1} = P_{t+1/t} - P_{t+1/t} H_{t+1}{}^T S_{t+1}{}^{-1} H_{t+1} P_{t+1/t}$$

- Uncertainty estimate SHRINKS

Linearized Motion Model for a Robot



From a robot-centric perspective, the velocities look like this:

$$\dot{x}_t = V_t$$
$$\dot{y}_t = 0$$
$$\dot{\phi}_t = \omega_t$$

From the global perspective, the velocities look like this:

$$\dot{x}_t = V_t \cos \phi_t$$
$$\dot{y}_t = V_t \sin \phi_t$$
$$\dot{\phi}_t = \omega_t$$

The discrete time state estimate (including noise) looks like this:

$$\hat{x}_{t+1} = \hat{x}_t + (V_t + w_{V_t})\delta t \cos \hat{\phi}_t$$
$$\hat{y}_{t+1} = \hat{y}_t + (V_t + w_{V_t})\delta t \sin \hat{\phi}_t$$
$$\hat{\phi}_{t+1} = \hat{\phi}_t + (\omega_t + w_{\omega_t})\delta t$$

Problem! We don't know linear and rotational velocity errors. The state estimate will rapidly diverge if this is the only source of information!

Linearized Motion Model for a Robot

Now, we have to compute the covariance matrix Propagation equations.

The indirect Kalman filter derives the pose equations from the estimated error:

$$x_{t+1} - \hat{x}_{t+1} = \tilde{x}_{t+1}$$

$$y_{t+1} - \hat{y}_{t+1} = \tilde{y}_{t+1}$$

$$\phi_{t+1} - \hat{\phi}_{t+1} = \tilde{\phi}_{t+1}$$

In order to linearize the system, the following small-angle assumptions are made:

$$\cos\tilde{\phi} \cong 1$$

$$\sin\tilde{\phi} \cong \tilde{\phi}$$

Linearized Motion Model for a Robot

From the error-state propagation equation, we can obtain the State propagation and noise input functions *F* and *G* :

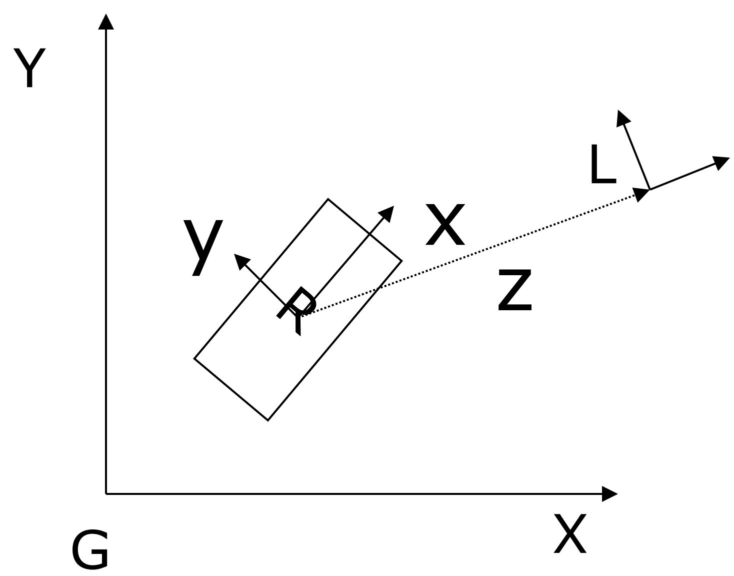
$$\begin{bmatrix} \tilde{x}_{t+1} \\ \tilde{y}_{t+1} \\ \tilde{\phi}_{t+1} \end{bmatrix} = \begin{bmatrix} 1 & 0 & -V_m \delta t \sin\hat{\phi} \\ 0 & 1 & V_m \delta t \cos\hat{\phi} \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \tilde{x}_t \\ \tilde{y}_t \\ \tilde{\phi}_t \end{bmatrix} + \begin{bmatrix} -\delta t \cos\phi_R & 0 \\ -\delta t \sin\phi_R & 0 \\ 0 & -\delta t \end{bmatrix} \begin{bmatrix} w_{V_t} \\ w_{\omega_t} \end{bmatrix}$$

$$\tilde{X}_{t+1} = F_t \tilde{X}_t + G_t W_t$$

From these values, we can easily compute the standard covariance propagation equation:

$$P_{t+1/t} = F_t P_{t/t} F_t^T + G_t Q_t G_t^T$$

Sensor Model for a Robot with a Perfect Map



From the robot, the measurement looks like this:

$$z_{t+1} = \begin{bmatrix} x_{L_{t+1}} \\ y_{L_{t+1}} \\ \phi_{L_{t+1}} \end{bmatrix} + \begin{bmatrix} n_x \\ n_y \\ n_\phi \end{bmatrix}$$

From a global perspective, the measurement looks like:

$$z_{t+1} = \begin{bmatrix} \cos\phi_{t+1} & -\sin\phi_{t+1} & 0 \\ \sin\phi_{t+1} & \cos\phi_{t+1} & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_{L_{t+1}} - x_{t+1} \\ y_{L_{t+1}} - y_{t+1} \\ \phi_{L_{t+1}} - \phi_{t+1} \end{bmatrix} + \begin{bmatrix} n_x \\ n_y \\ n_\phi \end{bmatrix}$$

The measurement equation is nonlinear and must also be linearized!

Sensor Model for a Robot with a Perfect Map

Now, we have to compute the linearized sensor function.

Once again, we make use of the indirect Kalman filter where the error in the reading must be estimated.

In order to linearize the system, the following small-angle assumptions are made:

$$\cos\tilde{\phi} \cong 1$$
$$\sin\tilde{\phi} \cong \tilde{\phi}$$

The final expression for the error in the sensor reading is:

$$\begin{bmatrix} \tilde{x}_{L_{t+1}} \\ \tilde{y}_{L_{t+1}} \\ \tilde{\phi}_{L_{t+1}} \end{bmatrix} = \begin{bmatrix} -\cos\hat{\phi}_{t+1} & -\sin\hat{\phi}_{t+1} & -\sin\hat{\phi}_{t+1}(x_L - \hat{x}_{t+1}) + \cos\hat{\phi}_t(y_L - \hat{y}_{t+1}) \\ \sin\hat{\phi}_{t+1} & -\cos\hat{\phi}_{t+1} & -\cos\hat{\phi}_{t+1}(x_L - \hat{x}_{t+1}) - \sin\hat{\phi}_t(y_L - \hat{y}_{t+1}) \\ 0 & 0 & -1 \end{bmatrix} \begin{bmatrix} \tilde{x}_{t+1} \\ \tilde{y}_{t+1} \\ \tilde{\phi}_{t+1} \end{bmatrix} + \begin{bmatrix} n_x \\ n_y \\ n_\phi \end{bmatrix}$$

- end of digression

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!

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

(E)KF-SLAM

- Map with N landmarks:(3+2N)-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

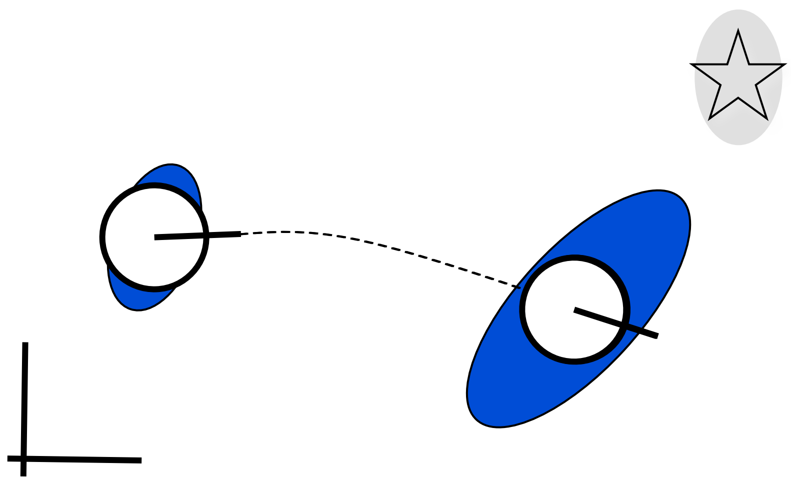
EKF SLAM: Building the Map

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:

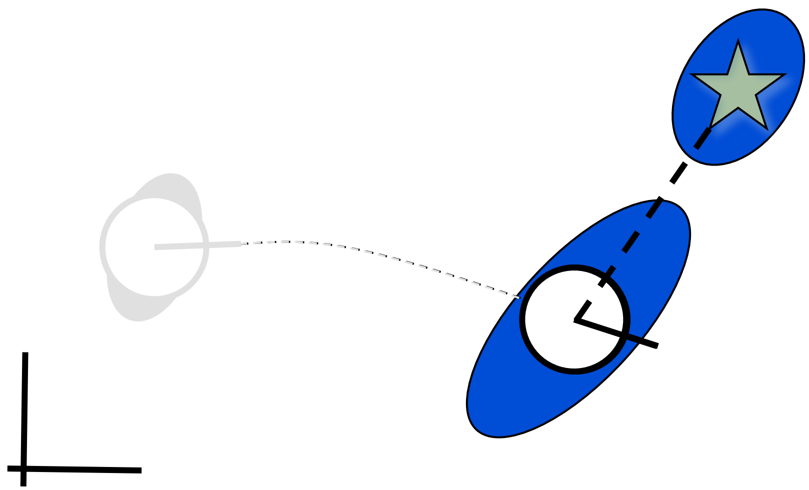
$$\hat{C}_{RM_i} = F_x\, C_{RM_i}$$

(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



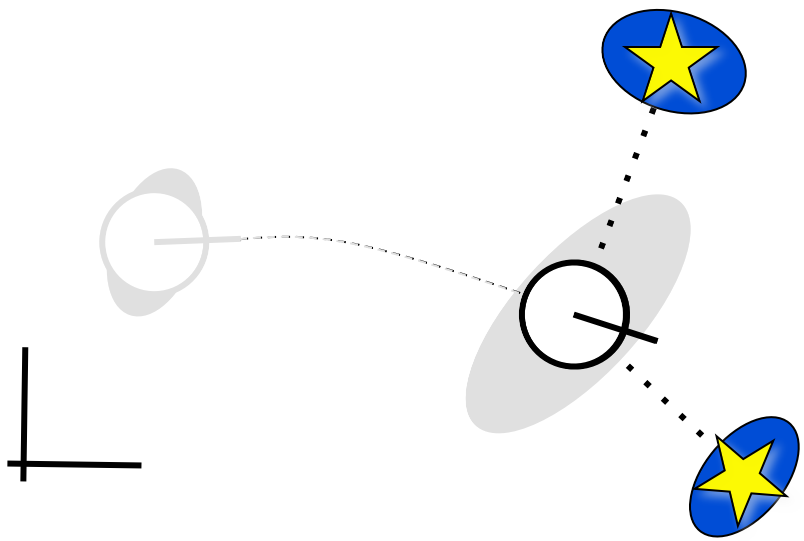
Global-to-local
frame transform h

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

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

EKF SLAM: Building the Map

- Observation



(x,y)-point landmarks

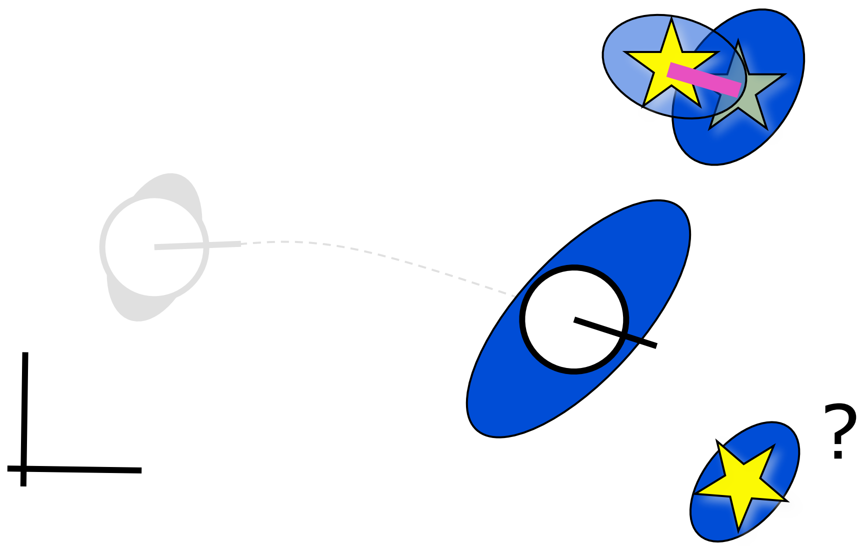
$$\mathbf{z}_k = \begin{bmatrix} x_1 \\ y_1 \\ x_2 \\ y_2 \end{bmatrix} = \begin{bmatrix} \mathbf{z}_1 \\ \mathbf{z}_2 \end{bmatrix}$$

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

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

EKF SLAM: Building the Map

- Data Association



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

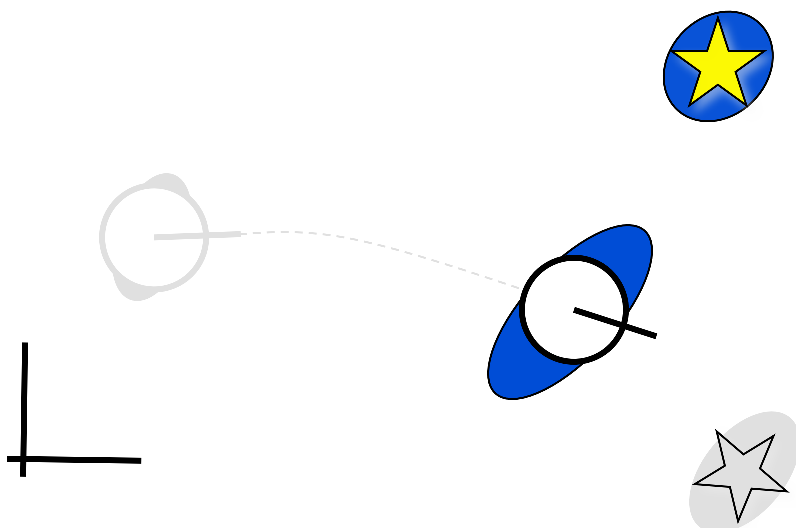
$$\begin{aligned} \nu_k^{ij} &= \mathbf{z}_k^j - \hat{\mathbf{z}}_k^i \\ S_k^{ij} &= R_k^j + H^i \, \hat{C}_k \, H^{i\,T} \end{aligned}$$

(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

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

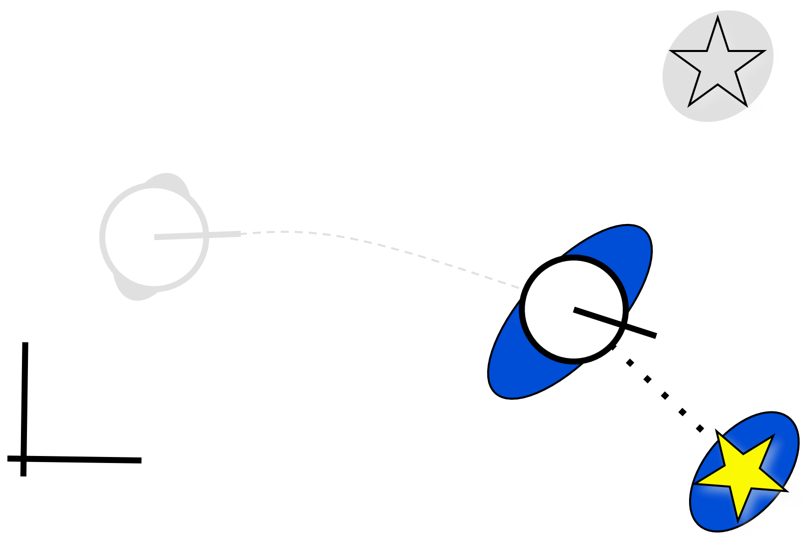
$$\mathbf{x}_k = \hat{\mathbf{x}}_k + K_k \, \nu_k$$

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

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

EKF SLAM: Building the Map

- Integrating New Landmarks



State augmented by

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

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

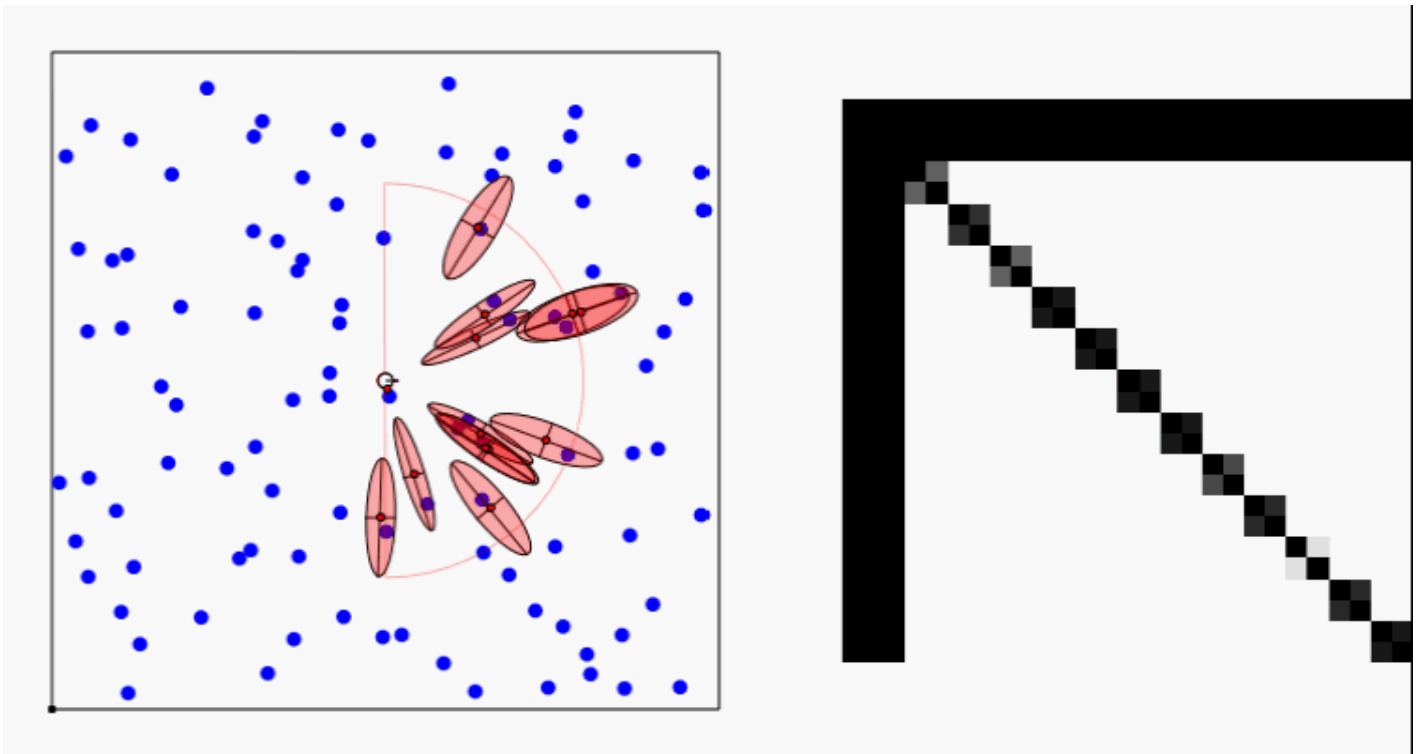
Cross-covariances:

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

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

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

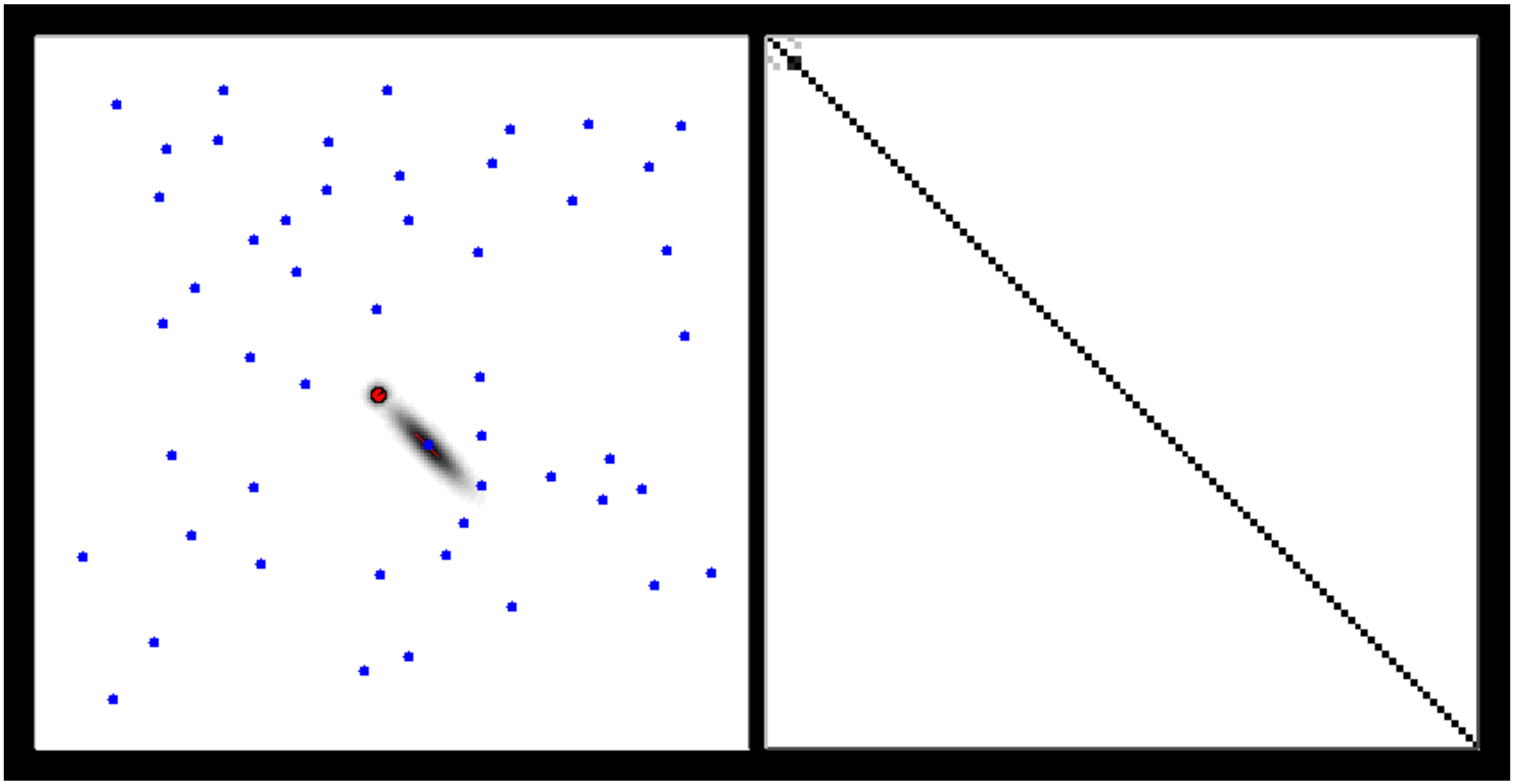
Classical Solution – The EKF



Blue path = true path   **Red path** = estimated path   **Black path** = odometry

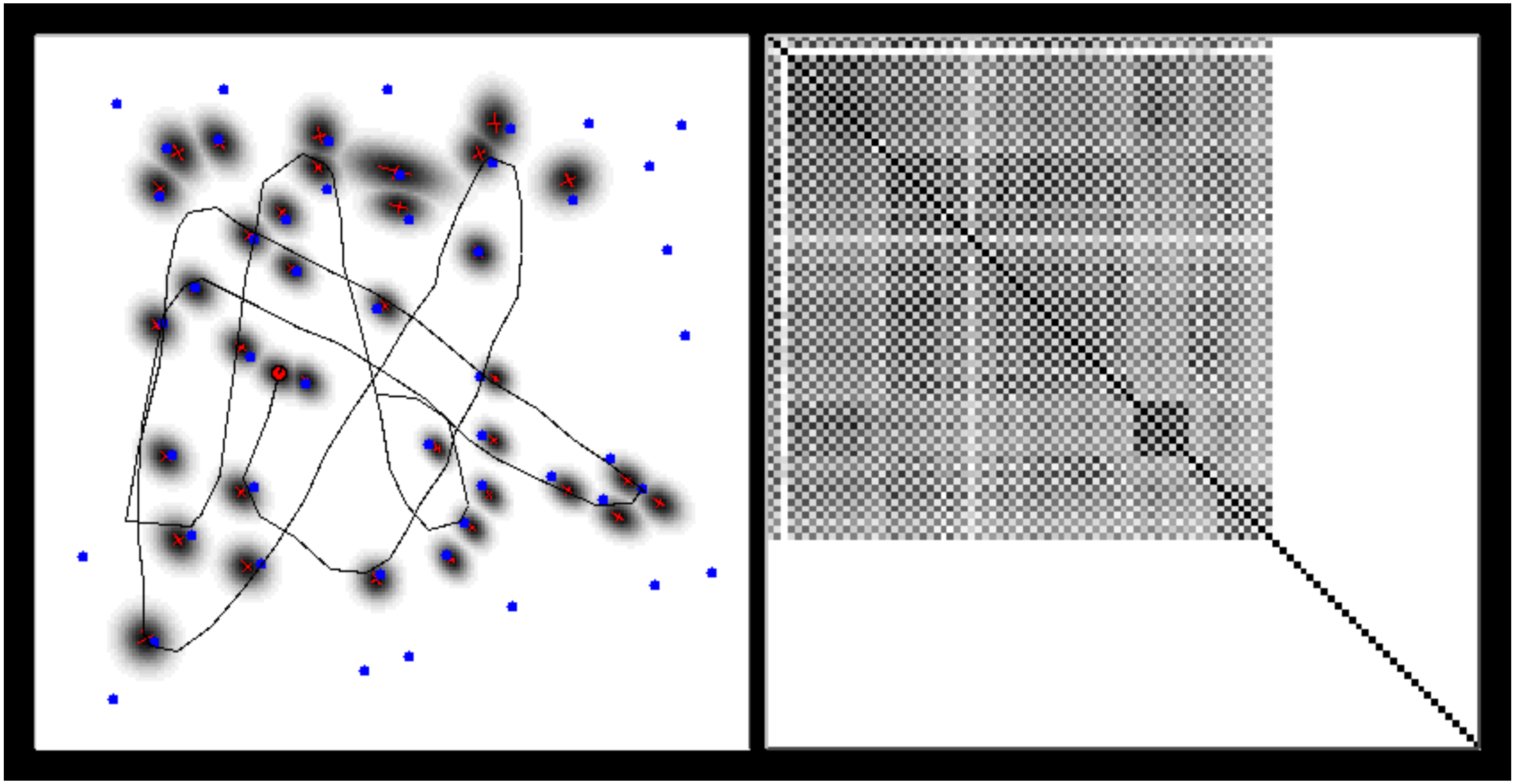
- Approximate the SLAM posterior with a high-dimensional Gaussian [Smith & Cheesman, 1986] …
- Single hypothesis data association

EKF-SLAM



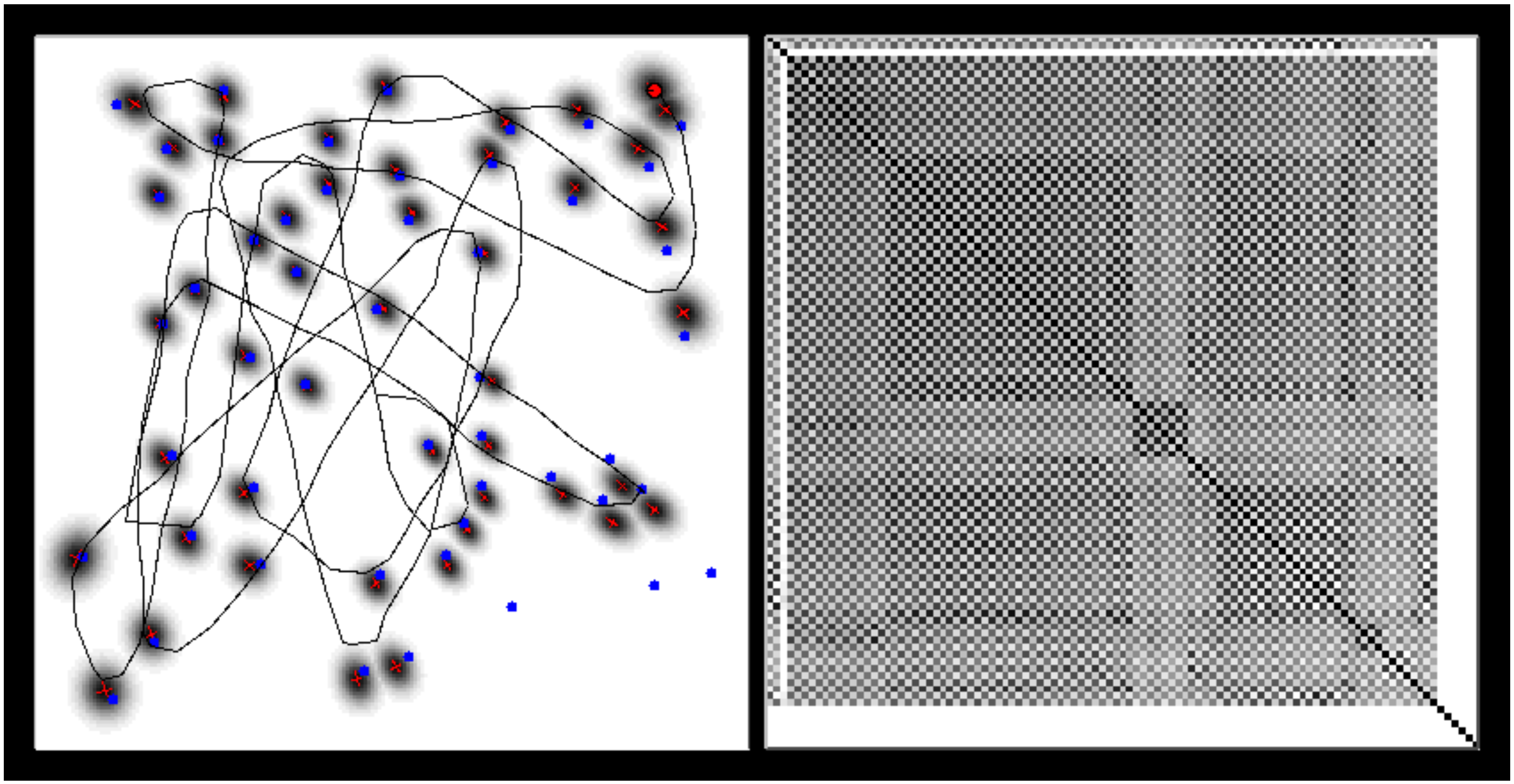
Map Correlation matrix

EKF-SLAM



Map Correlation matrix

EKF-SLAM



Map Correlation matrix

Properties of KF-SLAM (Linear Case) [Dissanayake et al., 2001]

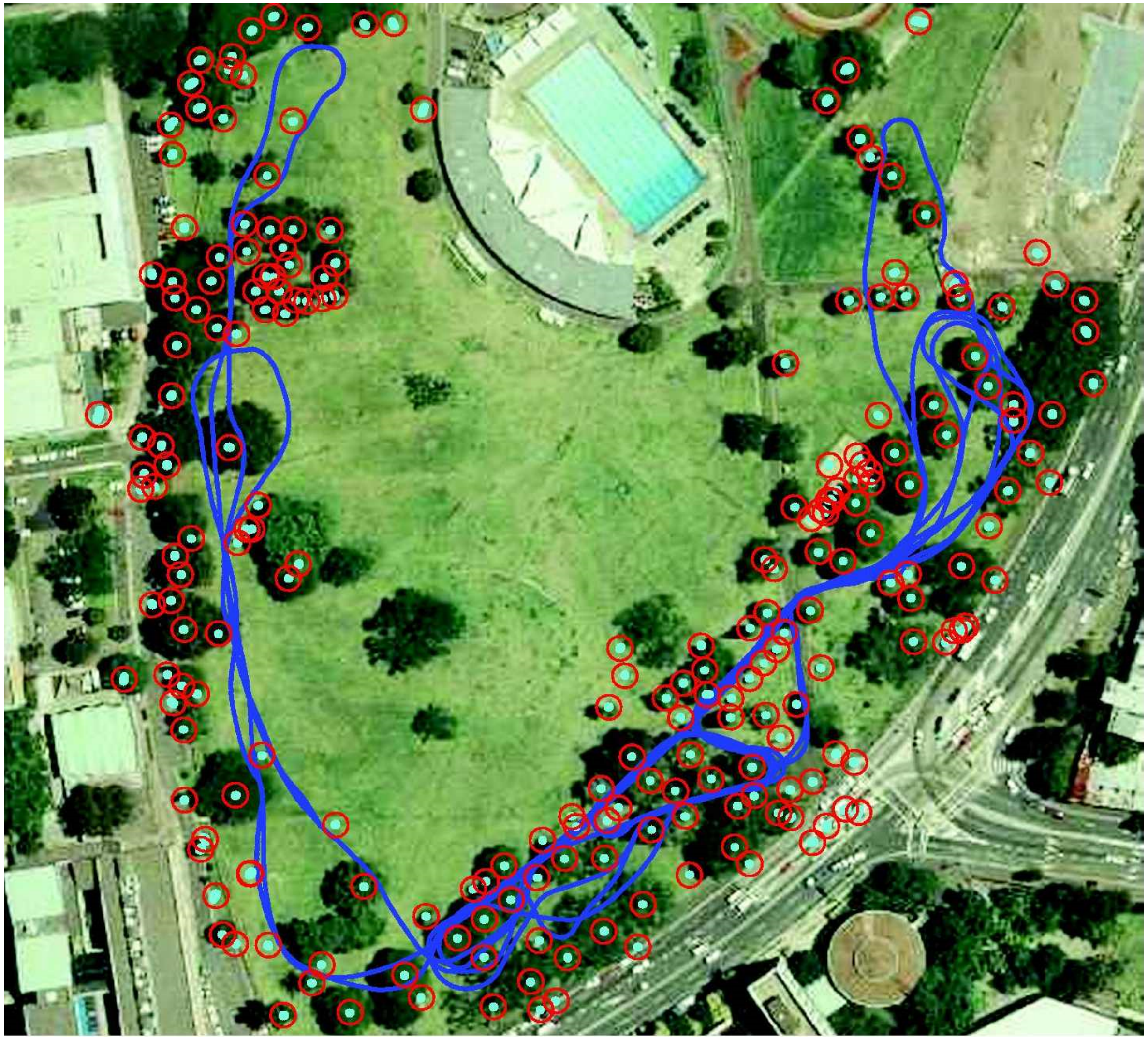
Theorem:

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

Theorem:

In the limit the landmark estimates become fully correlated

Victoria Park Data Set



[courtesy by E. Nebot]

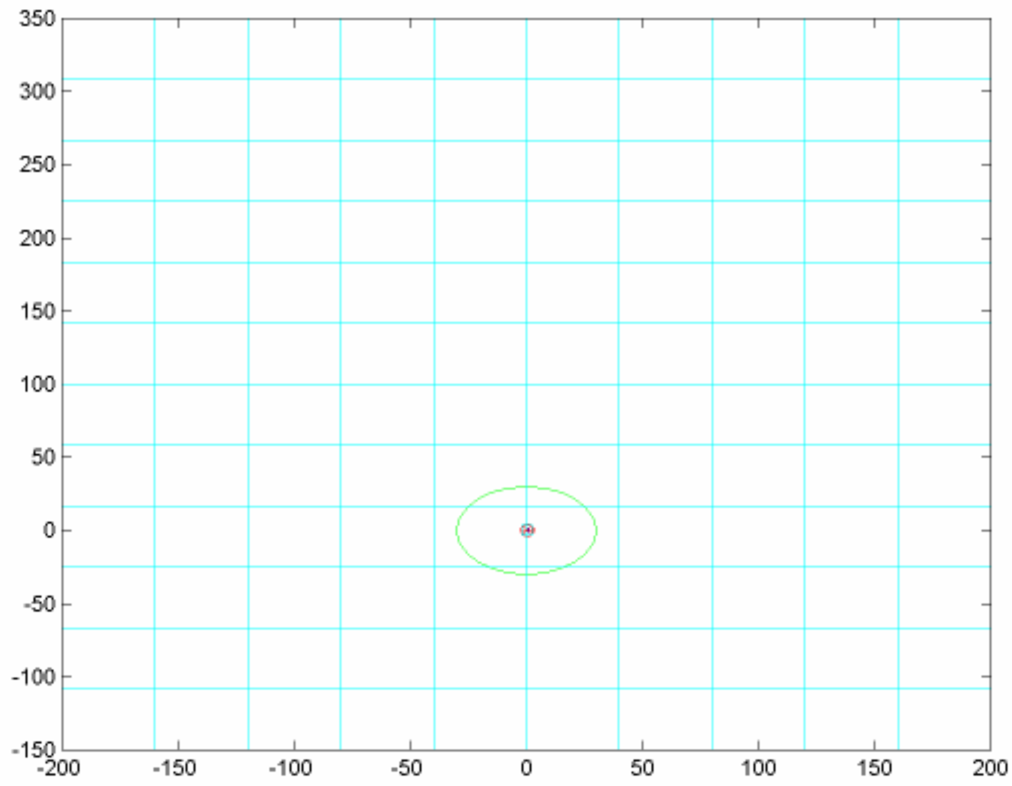
Victoria Park Data Set Vehicle

[courtesy by E. Nebot]

Data Acquisition

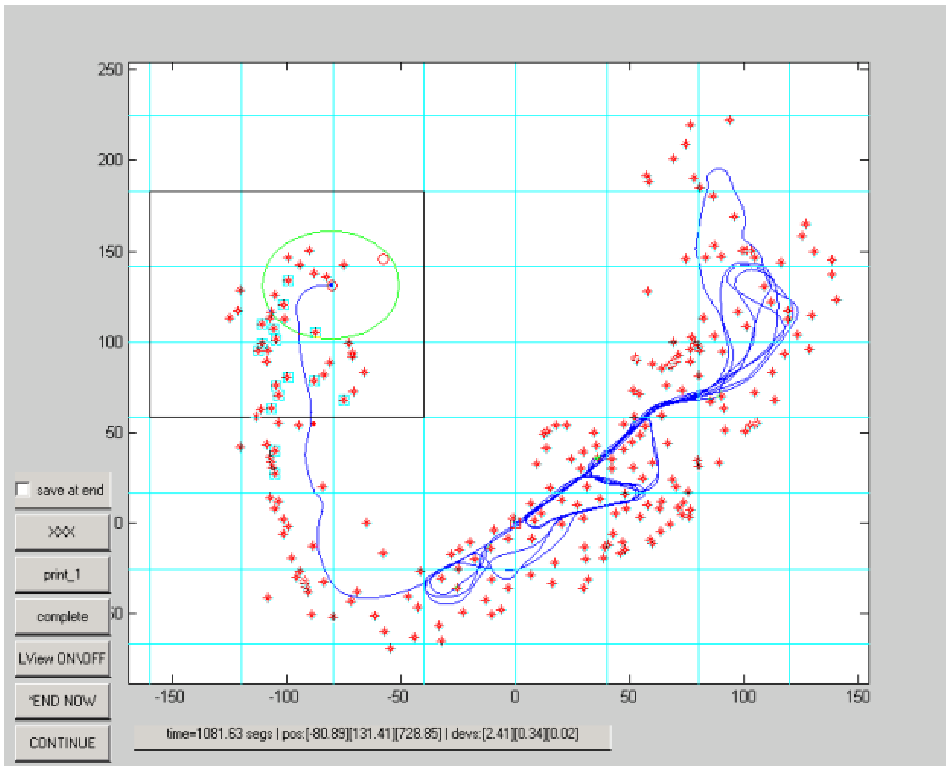
[courtesy by E. Nebot]

SLAM



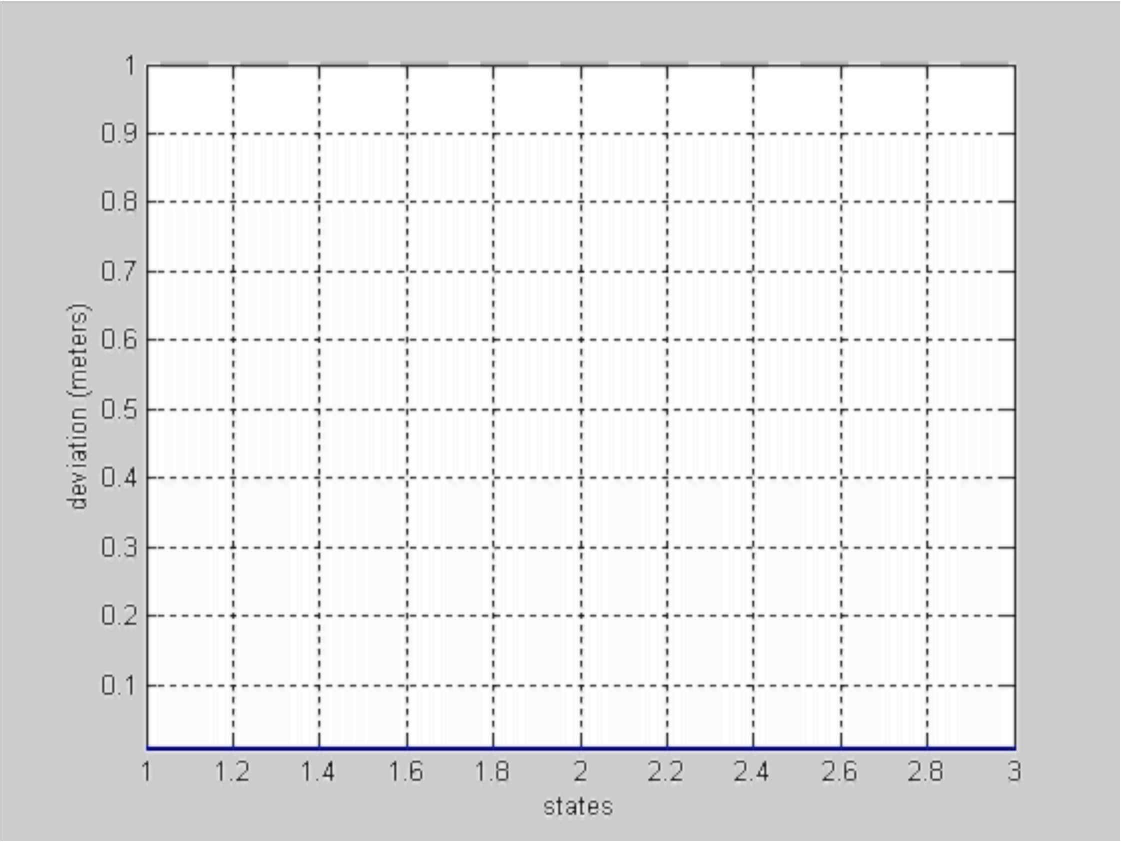
[courtesy by E. Nebot]

Map and Trajectory



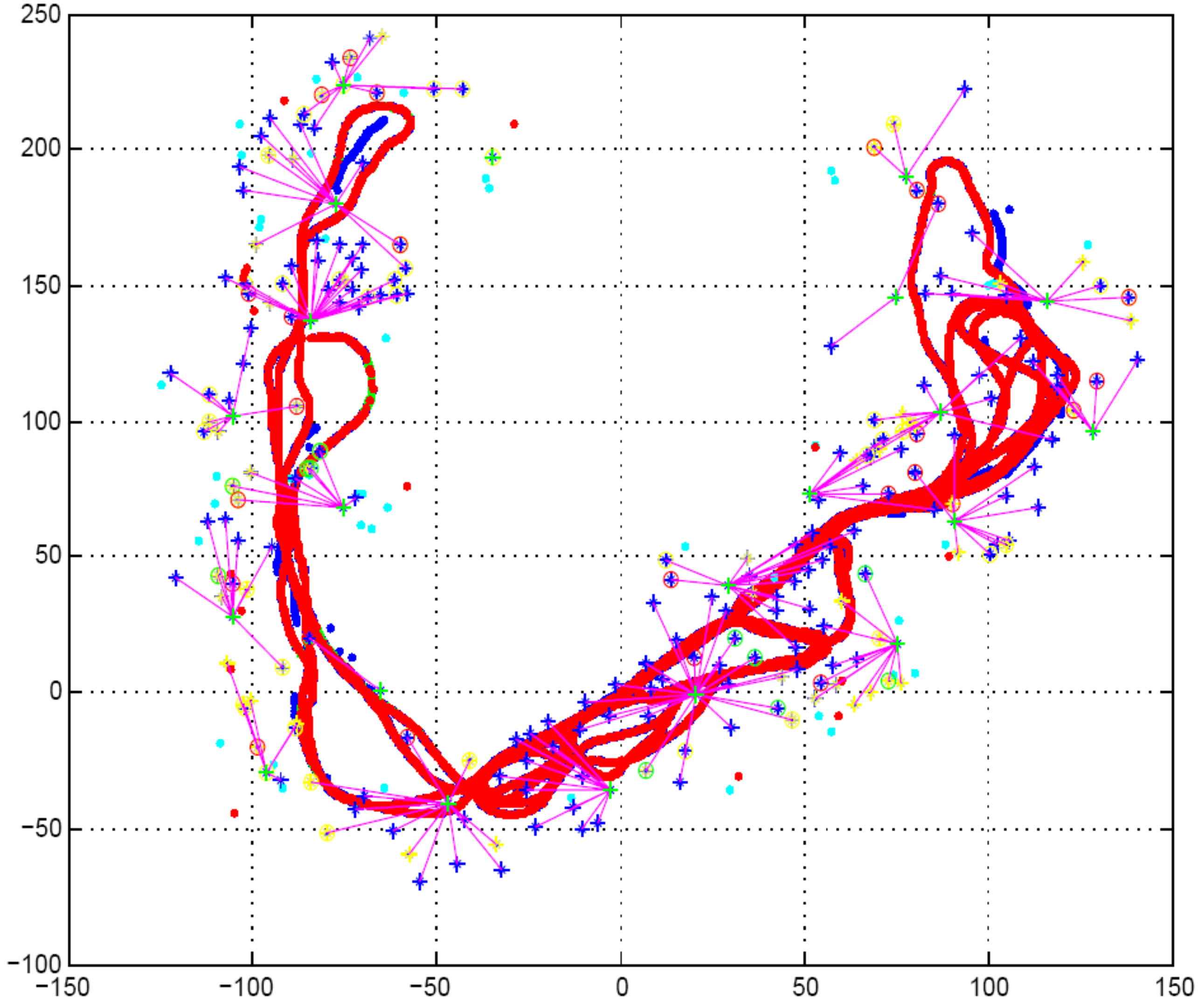
[courtesy by E. Nebot]

Landmark Covariance



[courtesy by E. Nebot]

Estimated Trajectory

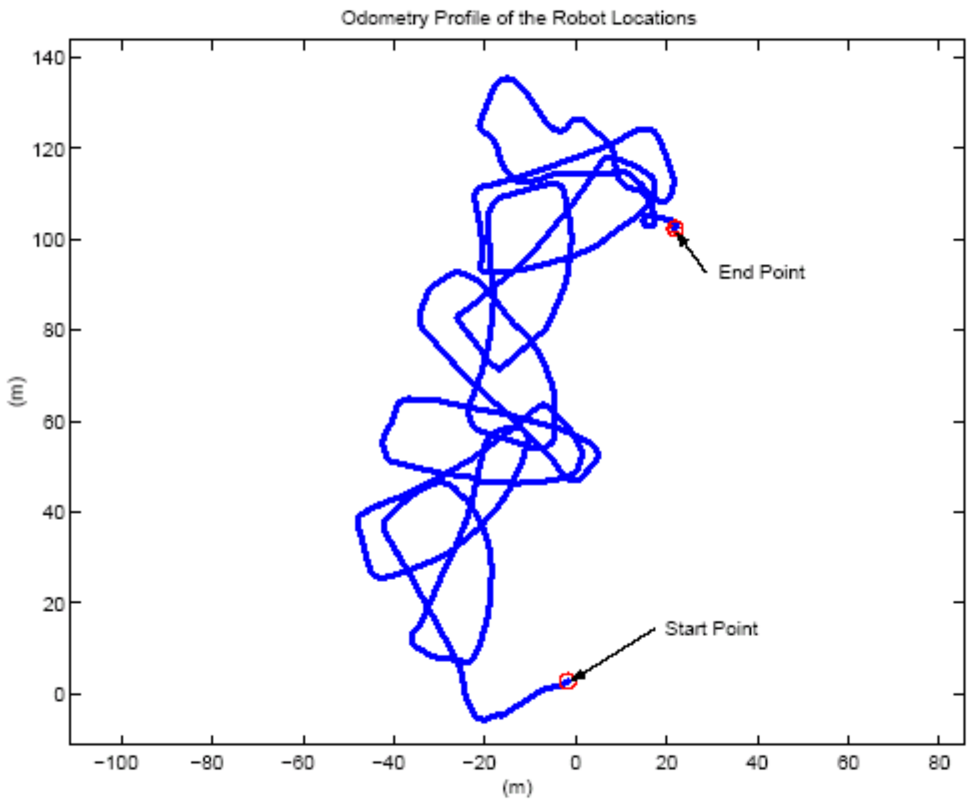


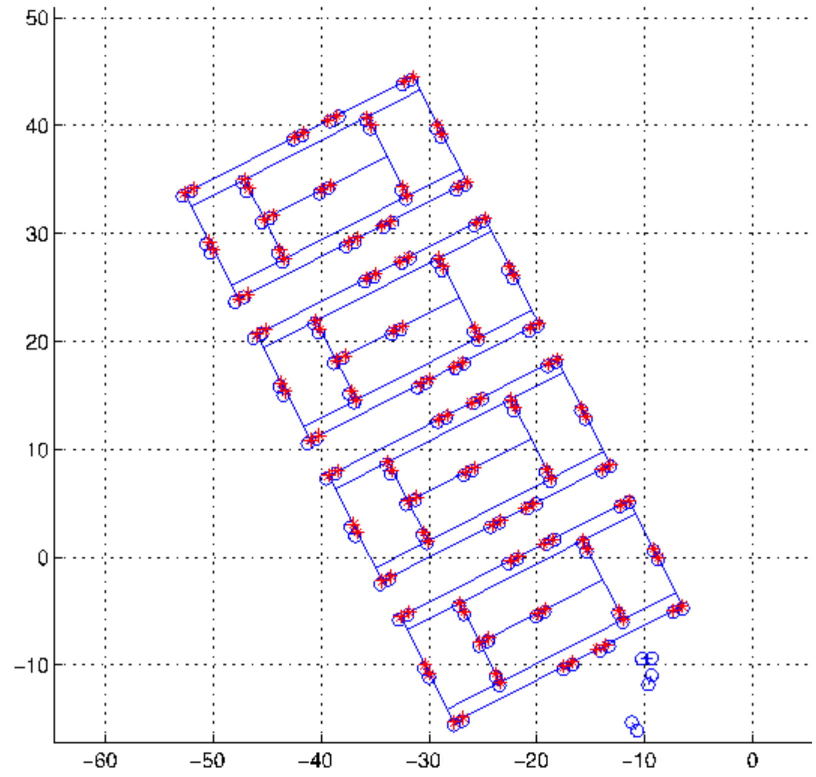
[courtesy by E. Nebot]

EKF SLAM Application

[courtesy by John Leonard]

EKF SLAM Application





odometry

estimated trajectory

[courtesy by John Leonard]

Approximations for SLAM

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

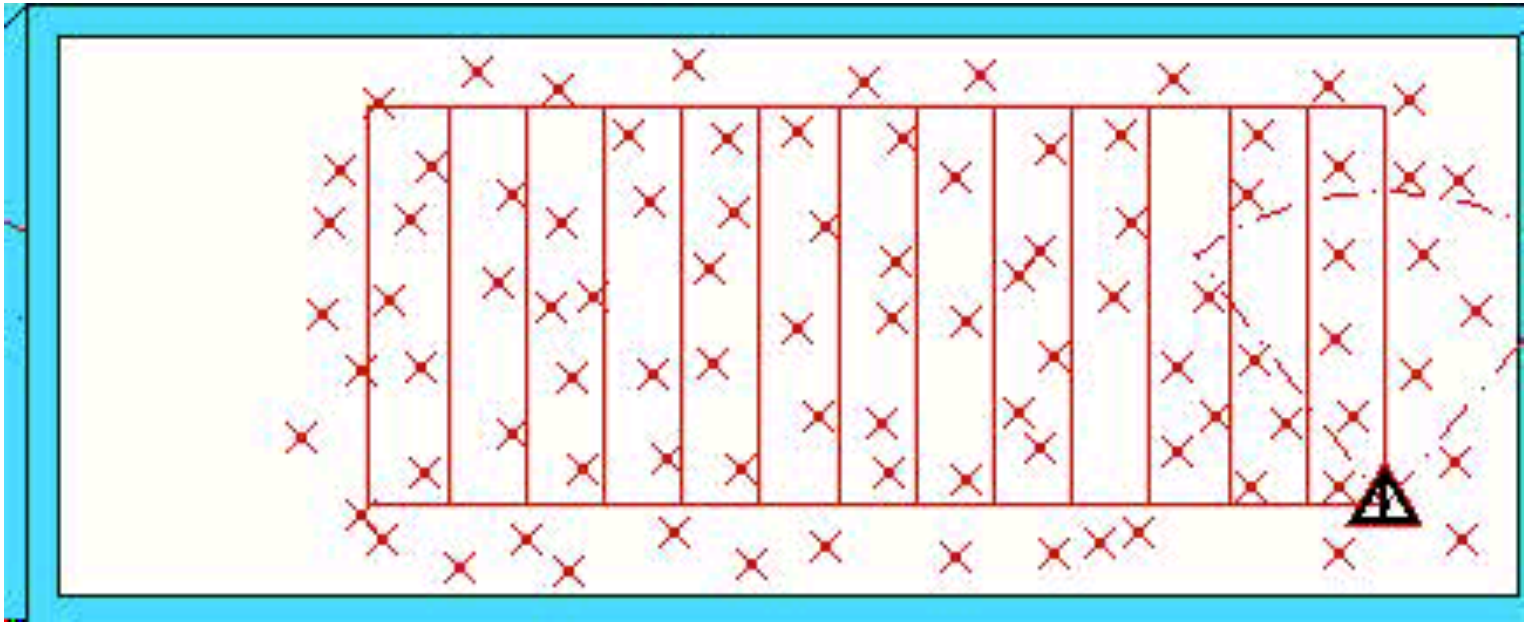
- Sparse links (correlations)
  [Lu & Milios 97, Guivant & Nebot 01]

- 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]

Sub-maps for EKF SLAM



[Leonard et al, 1998]

EKF-SLAM Summary

- Quadratic in the number of landmarks: $O(n^2)$
- Convergence results for the linear case.
- Can diverge if nonlinearities are large!
- Have been applied successfully in large-scale environments.
- Approximations reduce the computational complexity.