

# Robot Mapping

## EKF SLAM

Cyrill Stachniss



# Simultaneous Localization and Mapping (SLAM)

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



# Definition of the SLAM Problem

## Given

- The robot's controls

$$u_{1:T} = \{u_1, u_2, u_3 \dots, u_T\}$$

- Observations

$$z_{1:T} = \{z_1, z_2, z_3 \dots, z_T\}$$

## Wanted

- Map of the environment

$$m$$

- Path of the robot

$$x_{0:T} = \{x_0, x_1, x_2 \dots, x_T\}$$

# Three Main Paradigms

Kalman  
filter

Particle  
filter

Graph-  
based

# Bayes Filter

- Recursive filter with prediction and correction step

- **Prediction**

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

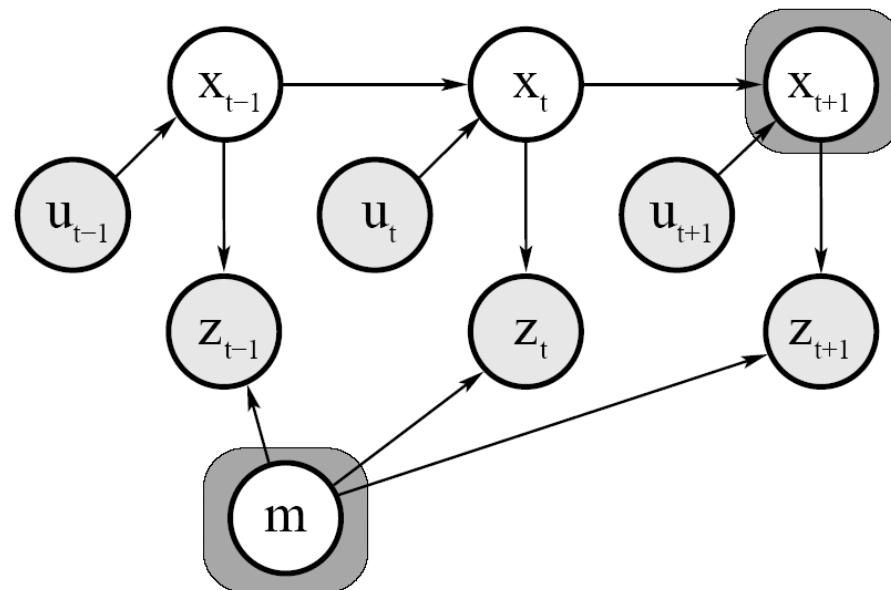
- **Correction**

$$bel(x_t) = \eta \, p(z_t \mid x_t) \, \overline{bel}(x_{t-1})$$

# EKF for Online SLAM

- The Kalman filter provides a solution to the online SLAM problem, i.e.

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



# 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  $\mu_t, \Sigma_t$

# EKF SLAM

- Application of the EKF to SLAM
- Estimate robot's pose and location of features in the environment
- Assumption: known correspondence
- State space is

$$x_t = \left( \underbrace{x, y, \theta}_{\text{robot's pose}} , \underbrace{m_{1,x}, m_{1,y}, \dots, m_{n,x}, m_{n,y}}_{\begin{array}{l} \text{landmark 1} \\ \vdots \\ \text{landmark n} \end{array}} \right)^T$$

# EKF SLAM: State Representation

- Map with  $n$  landmarks:  $(3+2n)$ -dimensional Gaussian
- Belief is represented by

$$\left( \begin{array}{c} x \\ y \\ \theta \\ m_{1,x} \\ m_{1,y} \\ \vdots \\ m_{n,x} \\ m_{n,y} \end{array} \right) \underbrace{\left( \begin{array}{ccc} \sigma_{xx} & \sigma_{xy} & \sigma_{x\theta} \\ \sigma_{yx} & \sigma_{yy} & \sigma_{y\theta} \\ \sigma_{\theta x} & \sigma_{\theta y} & \sigma_{\theta\theta} \\ \hline \sigma_{m_{1,x}x} & \sigma_{m_{1,x}y} & \sigma_{\theta} \\ \sigma_{m_{1,y}x} & \sigma_{m_{1,y}y} & \sigma_{\theta} \\ \vdots & \vdots & \vdots \\ \sigma_{m_{n,x}x} & \sigma_{m_{n,x}y} & \sigma_{\theta} \\ \sigma_{m_{n,y}x} & \sigma_{m_{n,y}y} & \sigma_{\theta} \end{array} \right)}_{\mu} \underbrace{\left( \begin{array}{ccccc} \sigma_{xm_{1,x}} & \sigma_{xm_{1,y}} & \dots & \sigma_{xm_{n,x}} & \sigma_{xm_{n,y}} \\ \sigma_{ym_{1,x}} & \sigma_{ym_{1,y}} & \dots & \sigma_{m_{n,x}} & \sigma_{m_{n,y}} \\ \sigma_{\theta m_{1,x}} & \sigma_{\theta m_{1,y}} & \dots & \sigma_{\theta m_{n,x}} & \sigma_{\theta m_{n,y}} \\ \hline \sigma_{m_{1,x}m_{1,x}} & \sigma_{m_{1,x}m_{1,y}} & \dots & \sigma_{m_{1,x}m_{n,x}} & \sigma_{m_{1,x}m_{n,y}} \\ \sigma_{m_{1,y}m_{1,x}} & \sigma_{m_{1,y}m_{1,y}} & \dots & \sigma_{m_{1,y}m_{n,x}} & \sigma_{m_{1,y}m_{n,y}} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \sigma_{m_{n,x}m_{1,x}} & \sigma_{m_{n,x}m_{1,y}} & \dots & \sigma_{m_{n,x}m_{n,x}} & \sigma_{m_{n,x}m_{n,y}} \\ \sigma_{m_{n,y}m_{1,x}} & \sigma_{m_{n,y}m_{1,y}} & \dots & \sigma_{m_{n,y}m_{n,x}} & \sigma_{m_{n,y}m_{n,y}} \end{array} \right)}_{\Sigma}$$

# EKF SLAM: State Representation

- More compactly

$$\left( \begin{array}{c} x_R \\ m_1 \\ \vdots \\ m_n \end{array} \right) \left( \begin{array}{cccc} \Sigma_{x_R x_R} & & & \\ \Sigma_{m_1 x_R} & \ddots & & \\ \vdots & & \ddots & \\ \Sigma_{m_n x_R} & & & \Sigma_{m_n m_n} \end{array} \right)$$

# EKF SLAM: State Representation

- Even more compactly (note:  $x_R \rightarrow x$ )

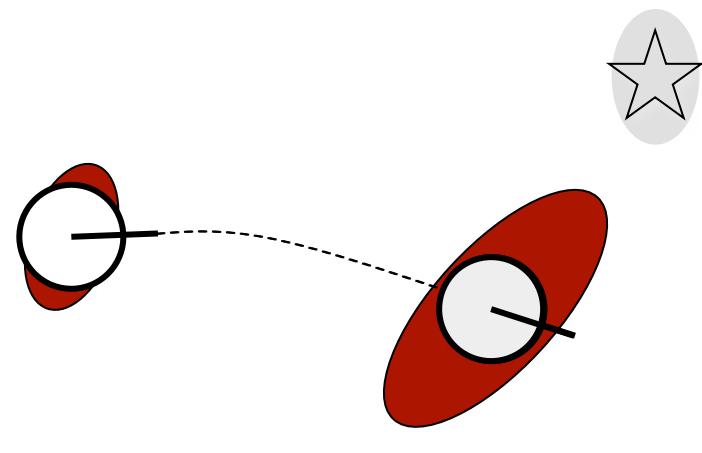
$$\left( \begin{array}{c} x \\ m \end{array} \right) \quad \left( \begin{array}{cc} \Sigma_{xx} & \Sigma_{xm} \\ \Sigma_{mx} & \Sigma_{mm} \end{array} \right)$$

$\underbrace{\hspace{10em}}_{\mu}$        $\underbrace{\hspace{10em}}_{\Sigma}$

# EKF SLAM: Filter Cycle

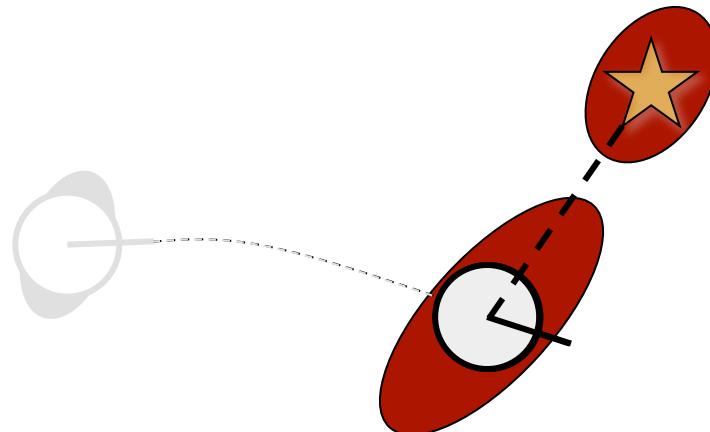
1. State prediction
2. Measurement prediction
3. Measurement
4. Data association
5. Update

# EKF SLAM: State Prediction



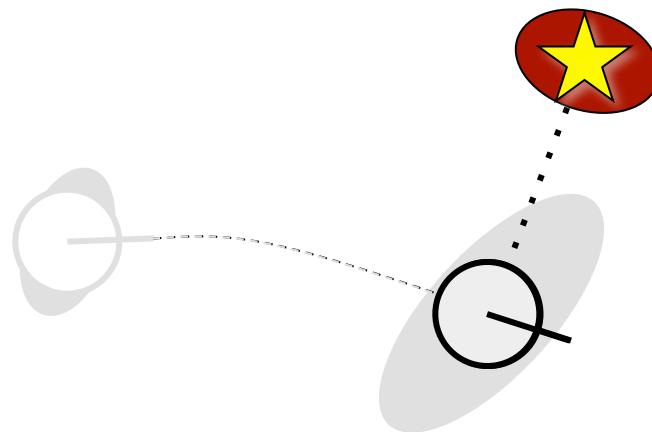
$$\begin{pmatrix} x_R \\ m_1 \\ \vdots \\ m_n \end{pmatrix} \underbrace{\quad}_{\mu} \quad \begin{pmatrix} \Sigma_{x_R x_R} & \Sigma_{x_R m_1} & \dots & \Sigma_{x_R m_n} \\ \Sigma_{m_1 x_R} & \Sigma_{m_1 m_1} & \dots & \Sigma_{m_1 m_n} \\ \vdots & \vdots & \ddots & \vdots \\ \Sigma_{m_n x_R} & \Sigma_{m_n m_1} & \dots & \Sigma_{m_n m_n} \end{pmatrix} \underbrace{\quad}_{\Sigma}$$

# EKF SLAM: Measurement Prediction



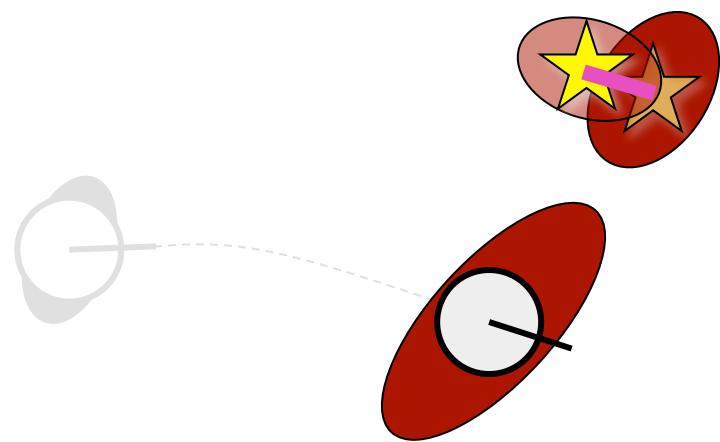
$$\begin{pmatrix} x_R \\ m_1 \\ \vdots \\ m_n \end{pmatrix} \underbrace{\begin{pmatrix} \Sigma_{x_R x_R} & \Sigma_{x_R m_1} & \cdots & \Sigma_{x_R m_n} \\ \Sigma_{m_1 x_R} & \Sigma_{m_1 m_1} & \cdots & \Sigma_{m_1 m_n} \\ \vdots & \vdots & \ddots & \vdots \\ \Sigma_{m_n x_R} & \Sigma_{m_n m_1} & \cdots & \Sigma_{m_n m_n} \end{pmatrix}}_{\Sigma}$$

# EKF SLAM: Obtained Measurement



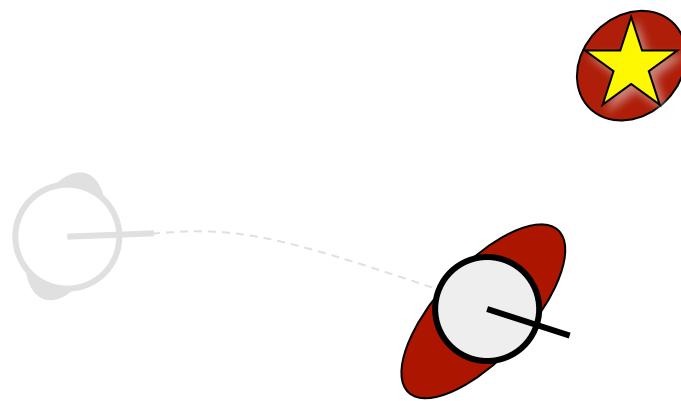
$$\underbrace{\begin{pmatrix} x_R \\ m_1 \\ \vdots \\ m_n \end{pmatrix}}_{\mu} \quad \underbrace{\begin{pmatrix} \Sigma_{x_R x_R} & \Sigma_{x_R m_1} & \cdots & \Sigma_{x_R m_n} \\ \Sigma_{m_1 x_R} & \Sigma_{m_1 m_1} & \cdots & \Sigma_{m_1 m_n} \\ \vdots & \vdots & \ddots & \vdots \\ \Sigma_{m_n x_R} & \Sigma_{m_n m_1} & \cdots & \Sigma_{m_n m_n} \end{pmatrix}}_{\Sigma}$$

# EKF SLAM: Data Association



$$\underbrace{\begin{pmatrix} x_R \\ m_1 \\ \vdots \\ m_n \end{pmatrix}}_{\mu} \quad \underbrace{\begin{pmatrix} \Sigma_{x_R x_R} & \Sigma_{x_R m_1} & \cdots & \Sigma_{x_R m_n} \\ \Sigma_{m_1 x_R} & \Sigma_{m_1 m_1} & \cdots & \Sigma_{m_1 m_n} \\ \vdots & \vdots & \ddots & \vdots \\ \Sigma_{m_n x_R} & \Sigma_{m_n m_1} & \cdots & \Sigma_{m_n m_n} \end{pmatrix}}_{\Sigma}$$

# EKF SLAM: Update Step



$$\left( \begin{array}{c} x_R \\ m_1 \\ \vdots \\ m_n \end{array} \right) \quad \left( \begin{array}{cccc} \Sigma_{x_R x_R} & \Sigma_{x_R m_1} & \cdots & \Sigma_{x_R m_n} \\ \Sigma_{m_1 x_R} & \Sigma_{m_1 m_1} & \cdots & \Sigma_{m_1 m_n} \\ \vdots & \vdots & \ddots & \vdots \\ \Sigma_{m_n x_R} & \Sigma_{m_n m_1} & \cdots & \Sigma_{m_n m_n} \end{array} \right)$$

$\underbrace{\hspace{10cm}}_{\mu}$        $\underbrace{\hspace{10cm}}_{\Sigma}$

# EKF-SLAM: Concrete Example

## Setup

- Robot moves in the plane
- Velocity-based motion model
- Robot observes point landmarks
- Range-bearing sensor
- Known data association
- Known number of landmarks

# Initialization

- Robot starts in its own reference frame (all landmarks unknown)
- $2N+3$  dimensions

$$\mu_0 = (0 \ 0 \ 0 \ \dots \ 0)^T$$

$$\Sigma_0 = \begin{pmatrix} 0 & 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & \infty & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & \infty \end{pmatrix}$$

# 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  $\mu_t, \Sigma_t$

# Prediction Step (Motion)

- Goal: Update state space based on the robot's motion
- Robot motion in the plane

$$\begin{pmatrix} x' \\ y' \\ \theta' \end{pmatrix} = \underbrace{\begin{pmatrix} x \\ y \\ \theta \end{pmatrix} + \begin{pmatrix} -\frac{v_t}{\omega_t} \sin \theta + \frac{v_t}{\omega_t} \sin(\theta + \omega_t \Delta t) \\ \frac{v_t}{\omega_t} \cos \theta - \frac{v_t}{\omega_t} \cos(\theta + \omega_t \Delta t) \\ \omega_t \Delta t \end{pmatrix}}_{g_{x,y,\theta}(u_t, (x,y,\theta)^T)}$$

- How to map that to the  $2N+3$  dim space?

# Update the State Space

- From the motion in the plane

$$\begin{pmatrix} x' \\ y' \\ \theta' \end{pmatrix} = \begin{pmatrix} x \\ y \\ \theta \end{pmatrix} + \begin{pmatrix} -\frac{v_t}{\omega_t} \sin \theta + \frac{v_t}{\omega_t} \sin(\theta + \omega_t \Delta t) \\ \frac{v_t}{\omega_t} \cos \theta - \frac{v_t}{\omega_t} \cos(\theta + \omega_t \Delta t) \\ \omega_t \Delta t \end{pmatrix}$$

- to the  $2N+3$  dimensional space

$$\begin{pmatrix} x' \\ y' \\ \theta' \\ \vdots \end{pmatrix} = \underbrace{\begin{pmatrix} x \\ y \\ \theta \\ \vdots \end{pmatrix} + \underbrace{\begin{pmatrix} 1 & 0 & 0 & 0 \dots 0 \\ 0 & 1 & 0 & 0 \dots 0 \\ 0 & 0 & 1 & \underbrace{0 \dots 0}_{2Ncols} \end{pmatrix}^T}_{F_x^T} \begin{pmatrix} -\frac{v_t}{\omega_t} \sin \theta + \frac{v_t}{\omega_t} \sin(\theta + \omega_t \Delta t) \\ \frac{v_t}{\omega_t} \cos \theta - \frac{v_t}{\omega_t} \cos(\theta + \omega_t \Delta t) \\ \omega_t \Delta t \end{pmatrix}}_{g(u_t, x_t)}$$

# Extended Kalman Filter Algorithm

- 1:  $\text{Extended\_Kalman\_filter}(\mu_{t-1}, \Sigma_{t-1}, u_t, z_t)$ :
- 2:  $\bar{\mu}_t = g(u_t, \mu_{t-1})$  **DONE**
- 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*  $\mu_t, \Sigma_t$

# Update Covariance

- The function  $g$  only affects the robot's motion and not the landmarks

Jacobian of the motion (3x3)

$$G_t = \begin{pmatrix} G_t^x & 0 \\ 0 & I \end{pmatrix}$$

↓  
Identity (2N x 2N)  
↑

# Jacobian of the Motion

$$G_t^x = \frac{\partial}{\partial(x, y, \theta)^T} \left[ \begin{pmatrix} x \\ y \\ \theta \end{pmatrix} + \begin{pmatrix} -\frac{v_t}{\omega_t} \sin \theta + \frac{v_t}{\omega_t} \sin(\theta + \omega_t \Delta t) \\ \frac{v_t}{\omega_t} \cos \theta - \frac{v_t}{\omega_t} \cos(\theta + \omega_t \Delta t) \\ \omega_t \Delta t \end{pmatrix} \right]$$

# This Leads to the Update

1: Extended\_Kalman\_filter( $\mu_{t-1}, \Sigma_{t-1}, u_t, z_t$ ):

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

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

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

$$= \begin{pmatrix} G_t^x & 0 \\ 0 & I \end{pmatrix} \begin{pmatrix} \Sigma_{xx} & \Sigma_{xm} \\ \Sigma_{mx} & \Sigma_{mm} \end{pmatrix} \begin{pmatrix} (G_t^x)^T & 0 \\ 0 & I \end{pmatrix} + R_t$$

$$= \begin{pmatrix} G_t^x \Sigma_{xx} (G_t^x)^T & G_t^x \Sigma_{xm} \\ (G_t^x \Sigma_{xm})^T & \Sigma_{mm} \end{pmatrix} + R_t$$

# Extended Kalman Filter Algorithm

- 1: **Extended\_Kalman\_filter**( $\mu_{t-1}, \Sigma_{t-1}, u_t, z_t$ ):
- 2:  ~~$\bar{\mu}_t = g(u_t, \mu_{t-1})$~~  **DONE**
- 3:  ~~$\bar{\Sigma}_t = G_t \Sigma_{t-1} G_t^T + R_t$~~  **DONE**
- 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*  $\mu_t, \Sigma_t$

# EKF SLAM – Prediction

**EKF-SLAM-Prediction**( $\mu_{t-1}, \Sigma_{t-1}, u_t, z_t, c_t, R_t$ ):

$$2: \quad F_x = \begin{pmatrix} 1 & 0 & 0 & 0 \cdots 0 \\ 0 & 1 & 0 & 0 \cdots 0 \\ 0 & 0 & 1 & 0 \cdots 0 \end{pmatrix}$$

$$3: \quad \bar{\mu}_t = \mu_{t-1} + F_x^T \begin{pmatrix} -\frac{v_t}{\omega_t} \sin \mu_{t-1, \theta} + \frac{v_t}{\omega_t} \sin(\mu_{t-1, \theta} + \omega_t \Delta t) \\ \frac{v_t}{\omega_t} \cos \mu_{t-1, \theta} - \frac{v_t}{\omega_t} \cos(\mu_{t-1, \theta} + \omega_t \Delta t) \\ \omega_t \Delta t \end{pmatrix}$$

$$4: \quad G_t = I + F_x^T \begin{pmatrix} 0 & 0 & -\frac{v_t}{\omega_t} \cos \mu_{t-1, \theta} + \frac{v_t}{\omega_t} \cos(\mu_{t-1, \theta} + \omega_t \Delta t) \\ 0 & 0 & -\frac{v_t}{\omega_t} \sin \mu_{t-1, \theta} + \frac{v_t}{\omega_t} \sin(\mu_{t-1, \theta} + \omega_t \Delta t) \\ 0 & 0 & 0 \end{pmatrix} F_x$$

$$5: \quad \bar{\Sigma}_t = G_t \Sigma_{t-1} G_t^T + \underbrace{F_x^T R_t^x F_x}_{R_t}$$

# Extended Kalman Filter Algorithm

- 1:  $\text{Extended\_Kalman\_filter}(\mu_{t-1}, \Sigma_{t-1}, u_t, z_t)$ :
- 2:  $\bar{\mu}_t = g(u_t, \mu_{t-1})$  **DONE**
- 3:  $\bar{\Sigma}_t = G_t \Sigma_{t-1} G_t^T + R_t$  **Apply & DONE**  

- 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:  $\text{return } \mu_t, \Sigma_t$

# EKF SLAM – Correction

- Known data association
- $c_t^i = j$ :  $i$ -th measurement observes the landmark with index  $j$
- Initialize landmark if unobserved
- Compute the expected observation
- Compute the Jacobian of  $h$
- Then, proceed with computing the Kalman gain

# Range-Bearing Observation

- Range-bearing observation  $z_t^i = (r_t^i, \phi_t^i)^T$
- If landmark has not been observed

$$\begin{pmatrix} \bar{\mu}_{j,x} \\ \bar{\mu}_{j,y} \end{pmatrix} = \begin{pmatrix} \bar{\mu}_{t,x} \\ \bar{\mu}_{t,y} \end{pmatrix} + \begin{pmatrix} r_t^i \cos(\phi_t^i + \bar{\mu}_{t,\theta}) \\ r_t^i \sin(\phi_t^i + \bar{\mu}_{t,\theta}) \end{pmatrix}$$

observed  
location of  
landmark j

estimated  
robot's  
location

relative  
measurement

# Expected Observation

- Compute expected observation according to the current estimate

$$\delta = \begin{pmatrix} \delta_x \\ \delta_y \end{pmatrix} = \begin{pmatrix} \bar{\mu}_{j,x} - \bar{\mu}_{t,x} \\ \bar{\mu}_{j,y} - \bar{\mu}_{t,y} \end{pmatrix}$$

$$q = \delta^T \delta$$

$$\hat{z}_t^i = \begin{pmatrix} \sqrt{q} \\ \text{atan2}(\delta_y, \delta_x) - \bar{\mu}_{t,\theta} \end{pmatrix}$$

# Jacobian for the Observation

- Based on  $\delta = \begin{pmatrix} \delta_x \\ \delta_y \end{pmatrix} = \begin{pmatrix} \bar{\mu}_{j,x} - \bar{\mu}_{t,x} \\ \bar{\mu}_{j,y} - \bar{\mu}_{t,y} \end{pmatrix}$   
 $q = \delta^T \delta$   
 $\hat{z}_t^i = \begin{pmatrix} \sqrt{q} \\ \text{atan2}(\delta_y, \delta_x) - \bar{\mu}_{t,\theta} \end{pmatrix}$

- Compute the Jacobian

$$\begin{aligned}\text{low } H_t^i &= \frac{\partial h(\bar{\mu}_t)}{\partial \bar{\mu}_t} \\ &= \frac{1}{q} \begin{pmatrix} -\sqrt{q}\delta_x & -\sqrt{q}\delta_y & 0 & +\sqrt{q}\delta_x & \sqrt{q}\delta_y \\ \delta_y & -\delta_x & -q & -\delta_y & \delta_x \end{pmatrix}\end{aligned}$$

# Jacobian for the Observation

- Use the computed Jacobian

$${}^{\text{low}} H_t^i = \frac{1}{q} \begin{pmatrix} -\sqrt{q}\delta_x & -\sqrt{q}\delta_y & 0 & +\sqrt{q}\delta_x & \sqrt{q}\delta_y \\ \delta_y & -\delta_x & -q & -\delta_y & \delta_x \end{pmatrix}$$

- map it to the high dimensional space

$$H_t^i = {}^{\text{low}} H_t^i F_{x,j}$$

$\downarrow$

$$F_{x,j} = \begin{pmatrix} 1 & 0 & 0 & 0 \cdots 0 & 0 & 0 & 0 \cdots 0 \\ 0 & 1 & 0 & 0 \cdots 0 & 0 & 0 & 0 \cdots 0 \\ 0 & 0 & 1 & 0 \cdots 0 & 0 & 0 & 0 \cdots 0 \\ 0 & 0 & 0 & 0 \cdots 0 & 1 & 0 & 0 \cdots 0 \\ 0 & 0 & 0 & \underbrace{0 \cdots 0}_{2j-2} & 0 & 1 & \underbrace{0 \cdots 0}_{2N-2j} \end{pmatrix}$$

# Next Steps as Specified...

- 1: Extended\_Kalman\_filter( $\mu_{t-1}, \Sigma_{t-1}, u_t, z_t$ ):
- 2:  ~~$\bar{\mu}_t = g(u_t, \mu_{t-1})$~~  **DONE**
- 3:  ~~$\bar{\Sigma}_t = G_t \Sigma_{t-1} G_t^T + R_t$~~  **DONE**
- 4:  $\rightarrow K_t = \bar{\Sigma}_t H_t^T (H_t \bar{\Sigma}_t H_t^T + Q_t)^{-1}$
- 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  $\mu_t, \Sigma_t$

# Extended Kalman Filter Algorithm

- 1:  $\text{Extended\_Kalman\_filter}(\mu_{t-1}, \Sigma_{t-1}, u_t, z_t)$ :
- 2:  $\bar{\mu}_t = g(u_t, \mu_{t-1})$  **DONE**
- 3:  $\bar{\Sigma}_t = G_t \Sigma_{t-1} G_t^T + R_t$  **DONE**
- 4:  $K_t = \bar{\Sigma}_t H_t^T (H_t \bar{\Sigma}_t H_t^T + Q_t)^{-1}$  **Apply & DONE**
- 5:  $\mu_t = \bar{\mu}_t + K_t(z_t - h(\bar{\mu}_t))$  **Apply & DONE**
- 6:  $\Sigma_t = (I - K_t H_t) \bar{\Sigma}_t$  **Apply & DONE**
- 7:   $return \mu_t, \Sigma_t$

# EKF SLAM – Correction (1/2)

## EKF\_SLAM\_Correction

6:     $Q_t = \begin{pmatrix} {\sigma_r}^2 & 0 \\ 0 & {\sigma_\phi}^2 \end{pmatrix}$

7:    for all observed features  $z_t^i = (r_t^i, \phi_t^i)^T$  do

8:       $j = c_t^i$

9:      if landmark  $j$  never seen before

10:        $\begin{pmatrix} \bar{\mu}_{j,x} \\ \bar{\mu}_{j,y} \end{pmatrix} = \begin{pmatrix} \bar{\mu}_{t,x} \\ \bar{\mu}_{t,y} \end{pmatrix} + \begin{pmatrix} r_t^i \cos(\phi_t^i + \bar{\mu}_{t,\theta}) \\ r_t^i \sin(\phi_t^i + \bar{\mu}_{t,\theta}) \end{pmatrix}$

11:     endif

12:      $\delta = \begin{pmatrix} \delta_x \\ \delta_y \end{pmatrix} = \begin{pmatrix} \bar{\mu}_{j,x} - \bar{\mu}_{t,x} \\ \bar{\mu}_{j,y} - \bar{\mu}_{t,y} \end{pmatrix}$

13:      $q = \delta^T \delta$

14:      $\hat{z}_t^i = \begin{pmatrix} \sqrt{q} \\ \text{atan2}(\delta_y, \delta_x) - \bar{\mu}_{t,\theta} \end{pmatrix}$

# EKF SLAM – Correction (2/2)

$$15: \quad F_{x,j} = \begin{pmatrix} 1 & 0 & 0 & 0 \cdots 0 & 0 & 0 & 0 \cdots 0 \\ 0 & 1 & 0 & 0 \cdots 0 & 0 & 0 & 0 \cdots 0 \\ 0 & 0 & 1 & 0 \cdots 0 & 0 & 0 & 0 \cdots 0 \\ 0 & 0 & 0 & 0 \cdots 0 & 1 & 0 & 0 \cdots 0 \\ 0 & 0 & 0 & \underbrace{0 \cdots 0}_{2j-2} & 0 & 1 & \underbrace{0 \cdots 0}_{2N-2j} \end{pmatrix}$$

$$16: \quad H_t^i = \frac{1}{q} \begin{pmatrix} -\sqrt{q}\delta_x & -\sqrt{q}\delta_y & 0 & +\sqrt{q}\delta_x & \sqrt{q}\delta_y \\ \delta_y & -\delta_x & -q & -\delta_y & +\delta_x \end{pmatrix} F_{x,j}$$

$$17: \quad K_t^i = \bar{\Sigma}_t H_t^{iT} (H_t^i \bar{\Sigma}_t H_t^{iT} + Q_t)^{-1}$$

$$18: \quad \bar{\mu}_t = \bar{\mu}_t + K_t^i (z_t^i - \hat{z}_t^i)$$

$$19: \quad \bar{\Sigma}_t = (I - K_t^i H_t^i) \bar{\Sigma}_t$$

20: endfor

$$21: \quad \mu_t = \bar{\mu}_t$$

$$22: \quad \Sigma_t = \bar{\Sigma}_t$$

23: return  $\mu_t, \Sigma_t$

# Implementation Notes

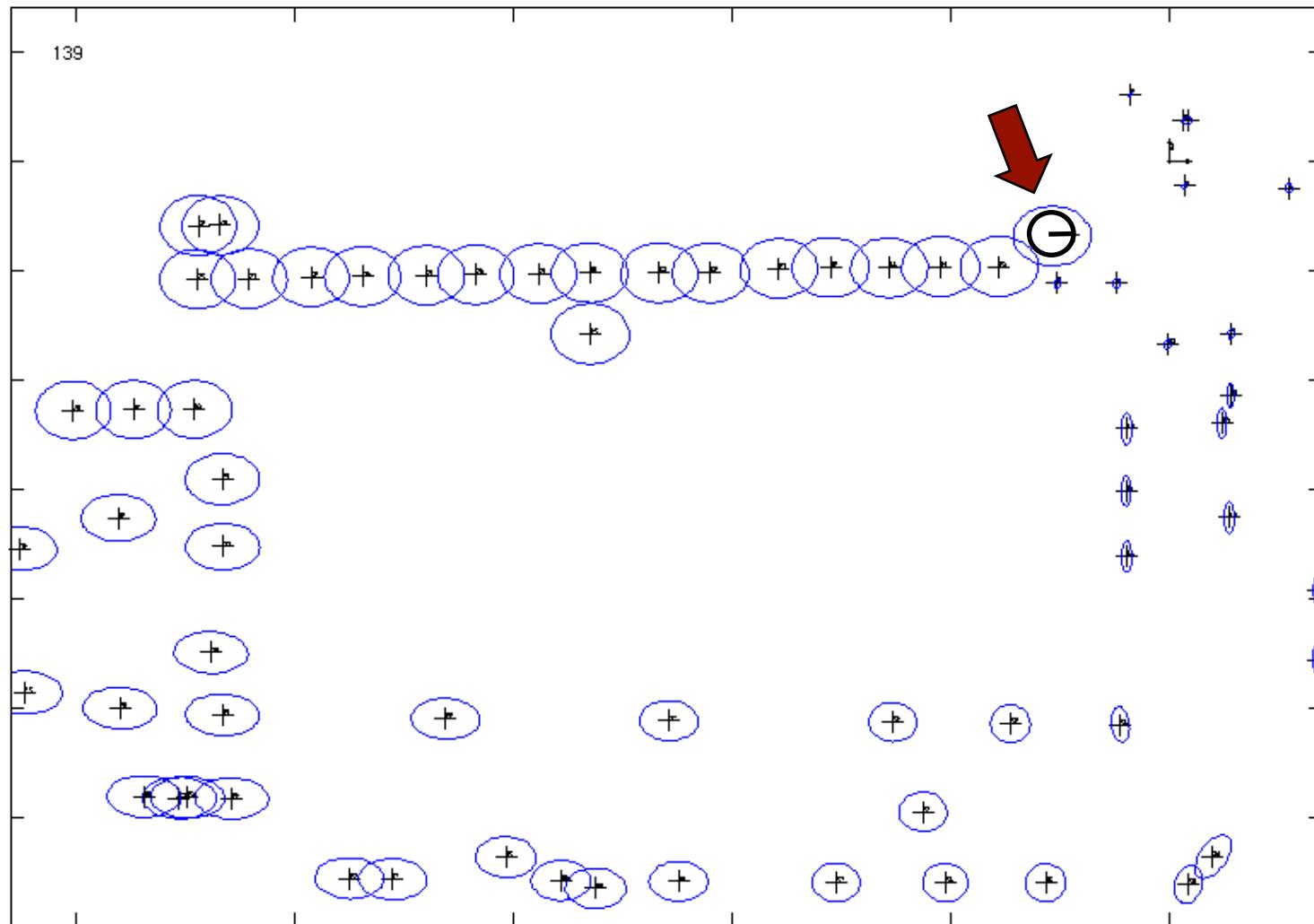
- Measurement update in a single step requires only one full belief update
- Always normalize the angular components
- You may not need to create the  $F$  matrices explicitly (e.g. in Octave)

# Done!

# Loop Closing

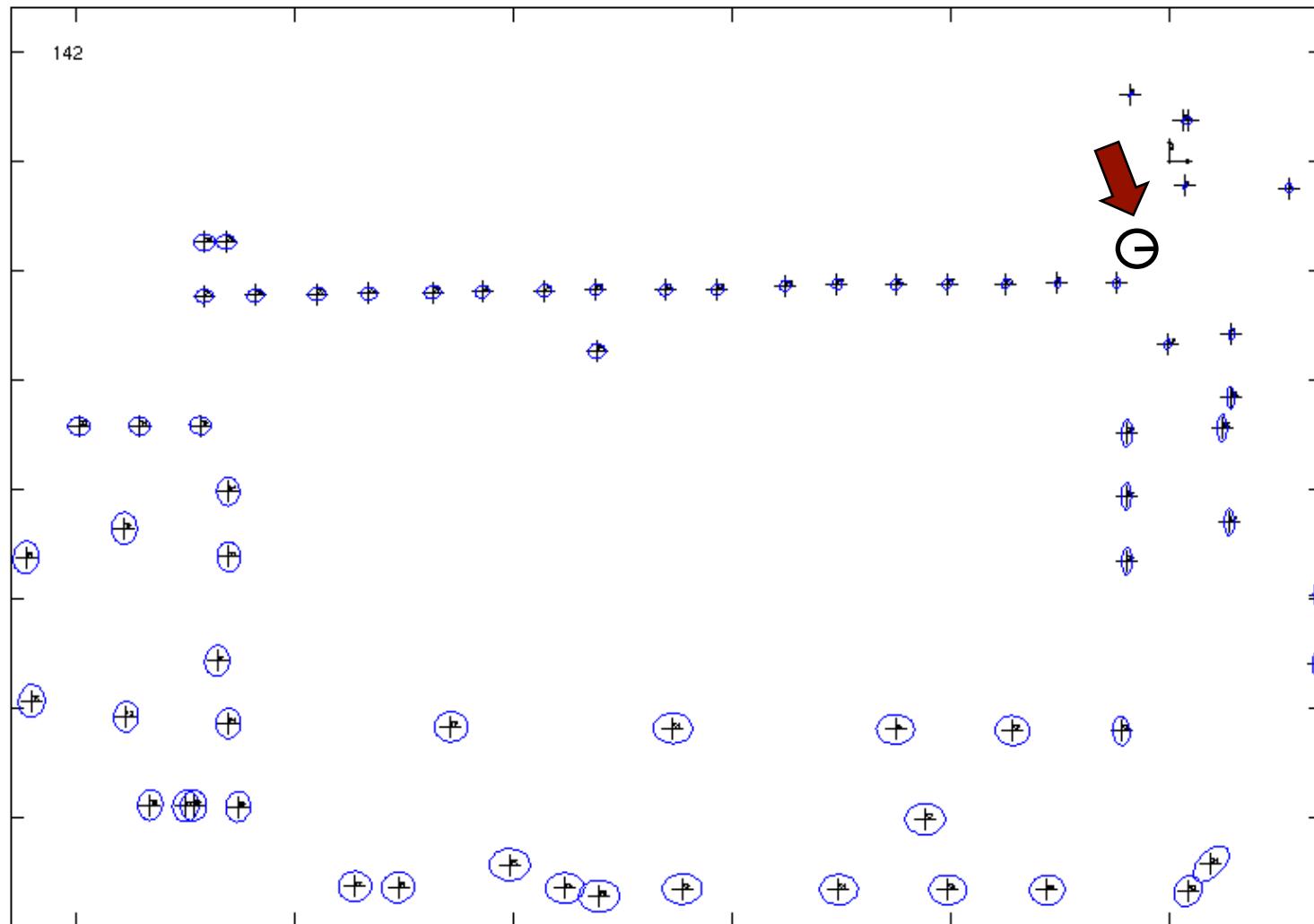
- **Recognizing an already mapped area**
- Data association with
  - high ambiguity
  - possible environment symmetries
- Uncertainties **collapse** after a loop closure  
(whether the closure was correct or not)

# Before the Loop Closure



Courtesy of K. Arras

# After the Loop Closure



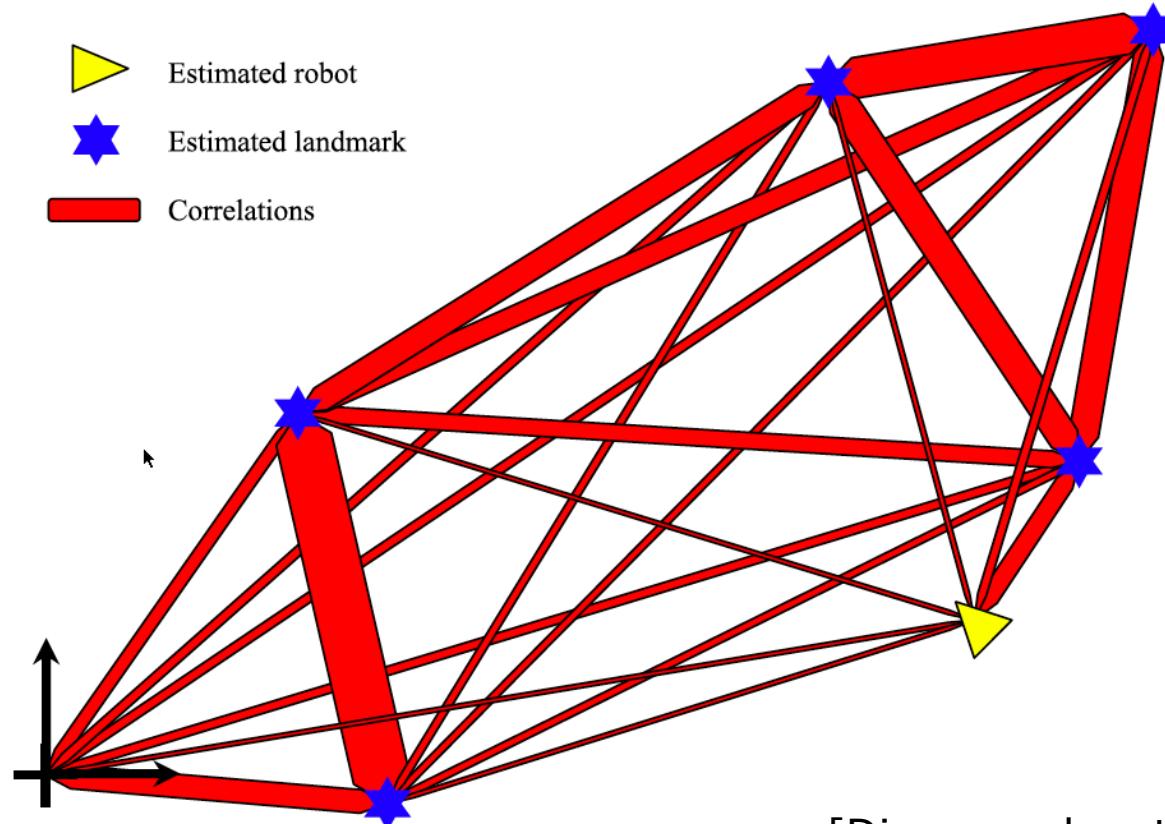
Courtesy of K. Arras

# SLAM: Loop Closure

- Loop closing **reduces the uncertainty** in robot and landmark estimates
- This can be **exploited when exploring** an environment for the sake of better (e.g. more accurate) maps
- **Wrong loop closures** lead to **filter divergence**

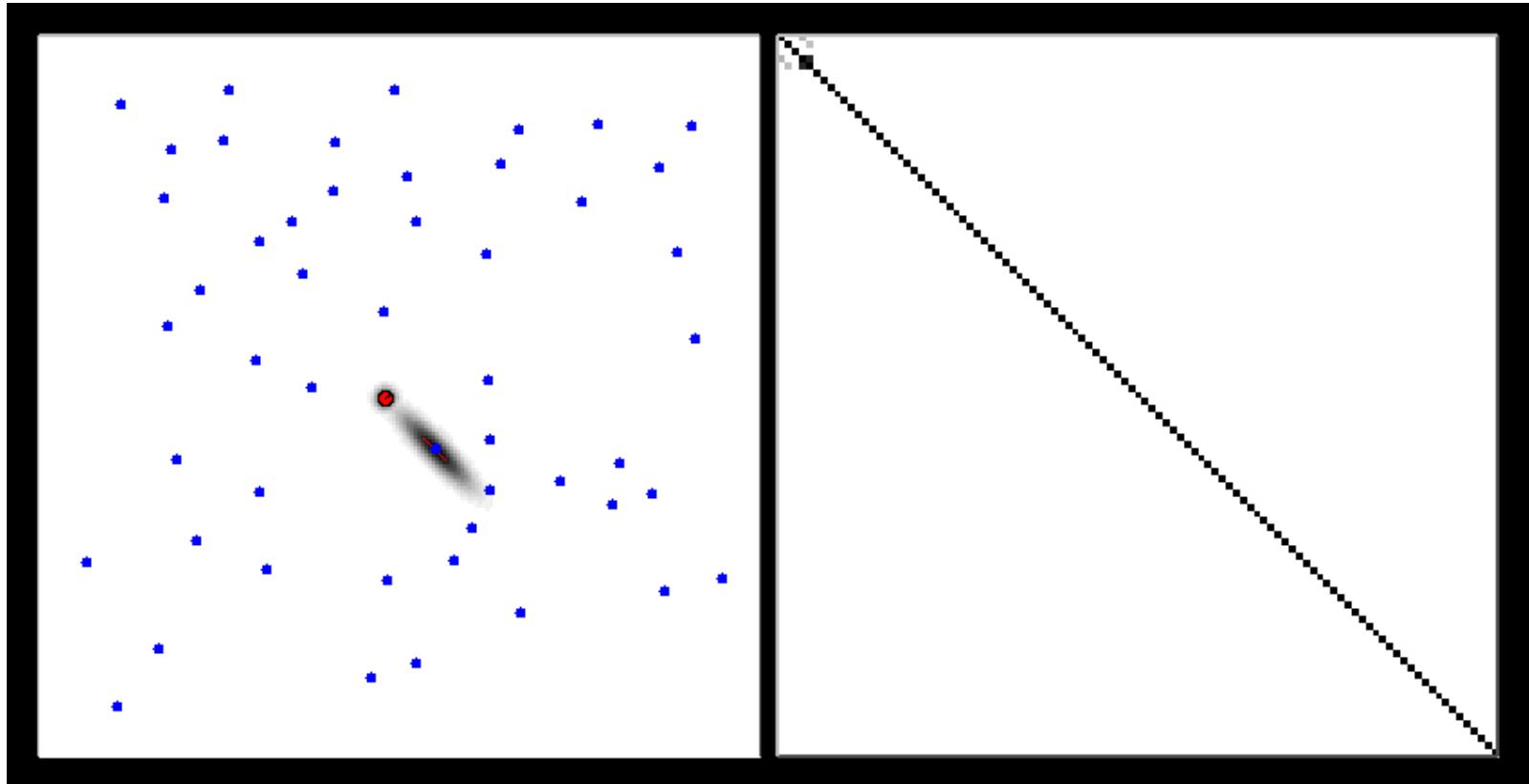
# EKF-SLAM Properties

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



[Dissanayake et al., 2001]

# EKF SLAM

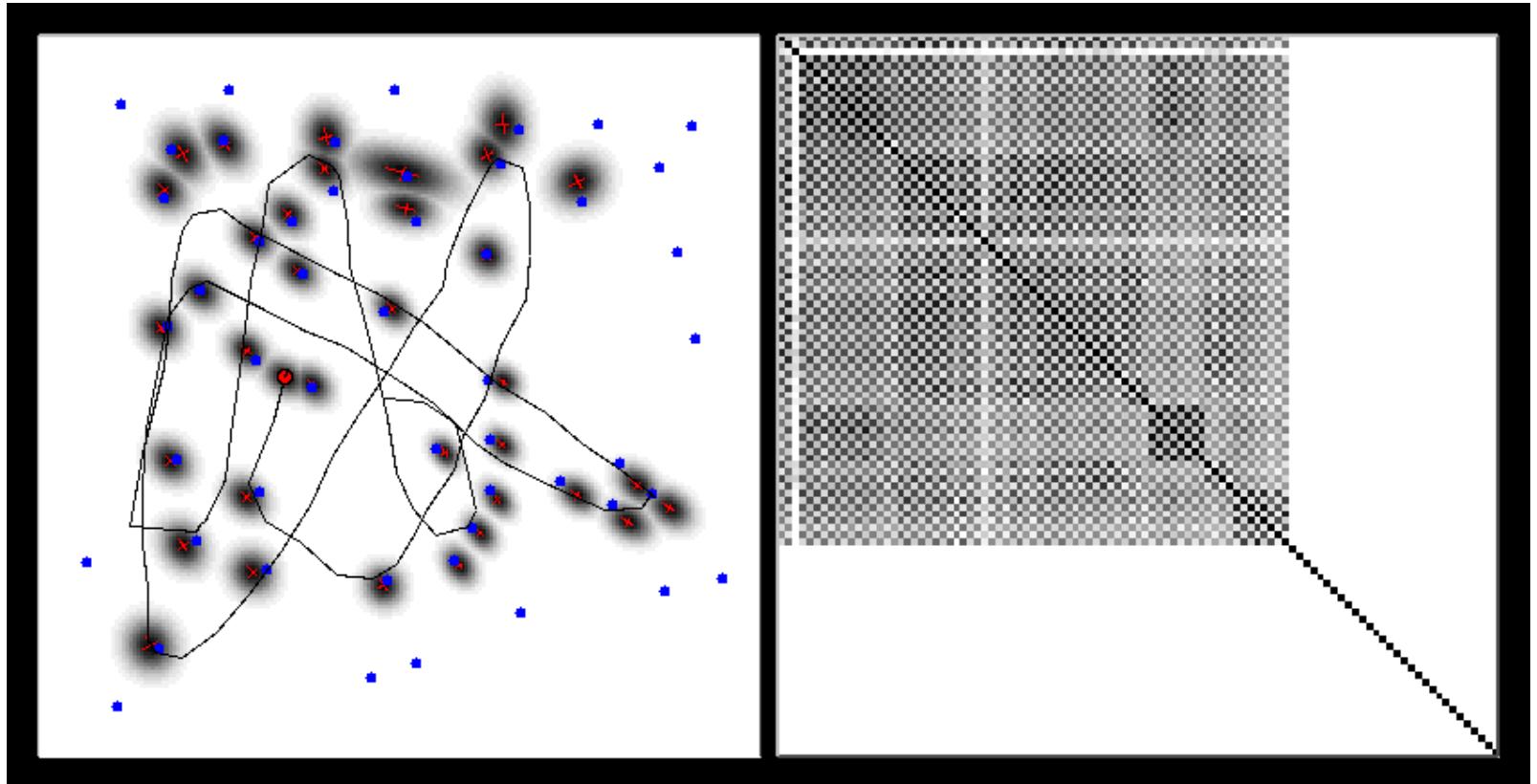


Map

Correlation matrix

Courtesy of M. Montemerlo

# EKF SLAM

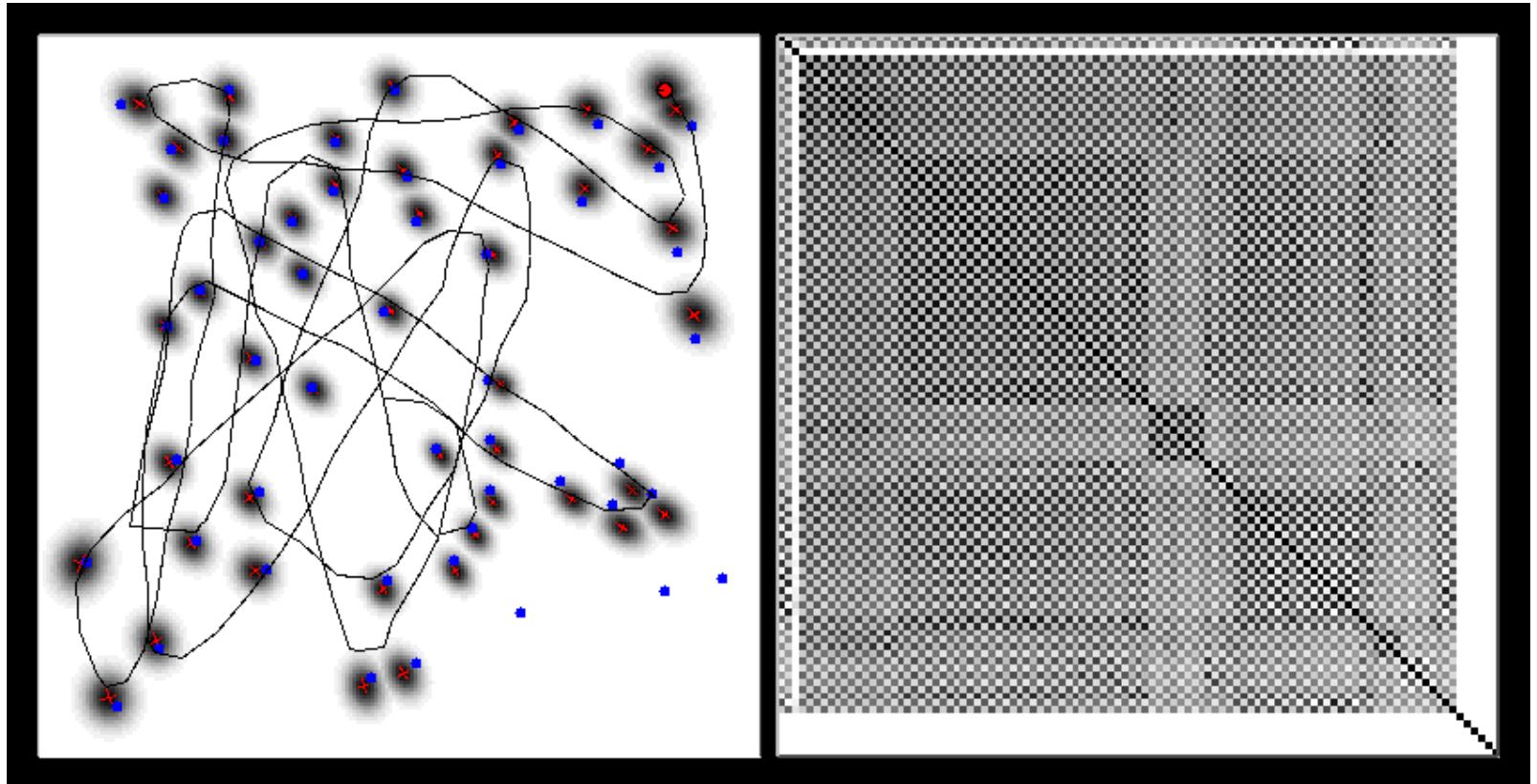


Map

Correlation matrix

Courtesy of M. Montemerlo

# EKF SLAM



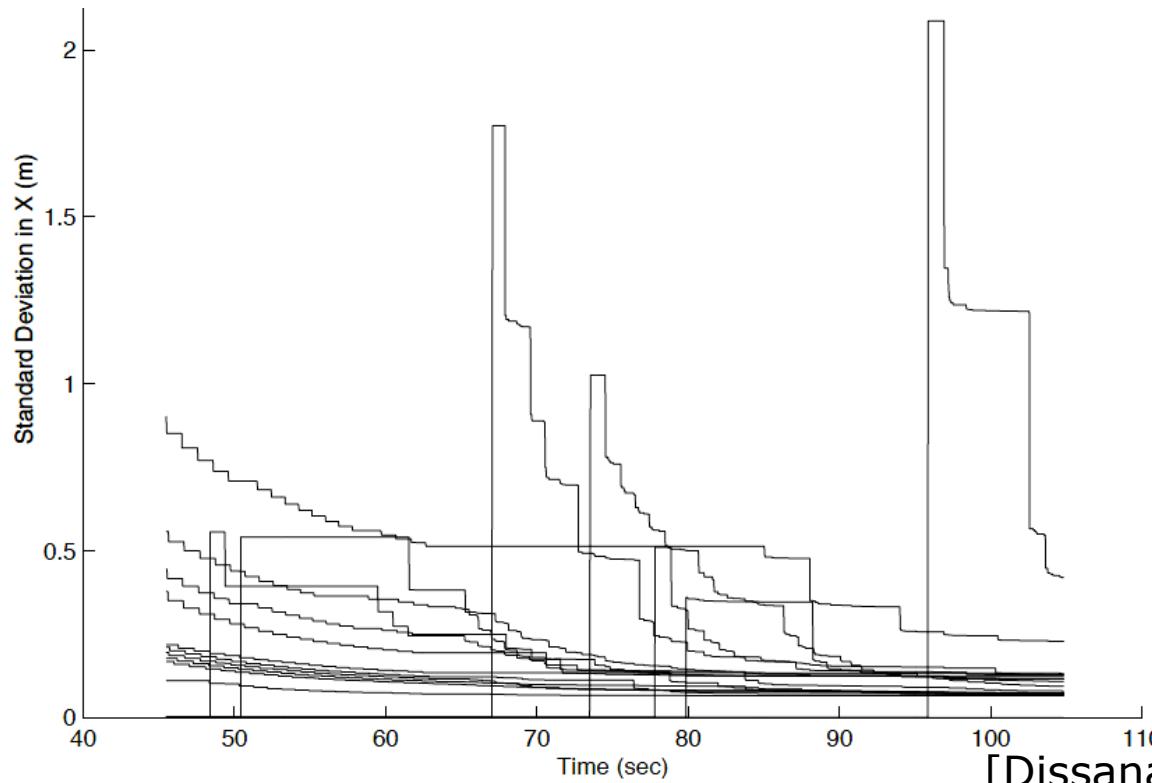
Map

Correlation matrix

Courtesy of M. Montemerlo

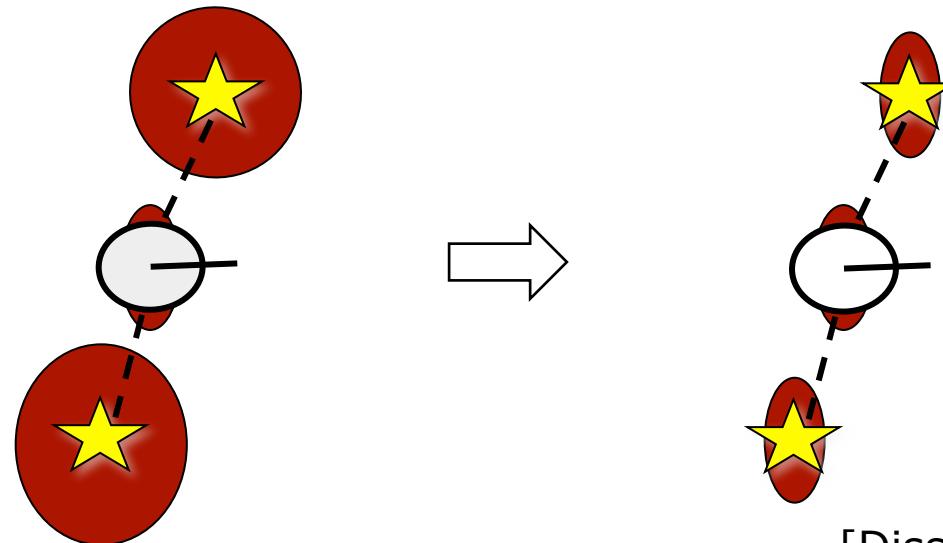
# EKF-SLAM Properties

- The **determinant** of any sub-matrix of the map covariance matrix **decreases monotonically**
- New landmarks are initialized with **max uncertainty**



# EKF-SLAM Properties

- 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]

# Example: Victoria Park Dataset



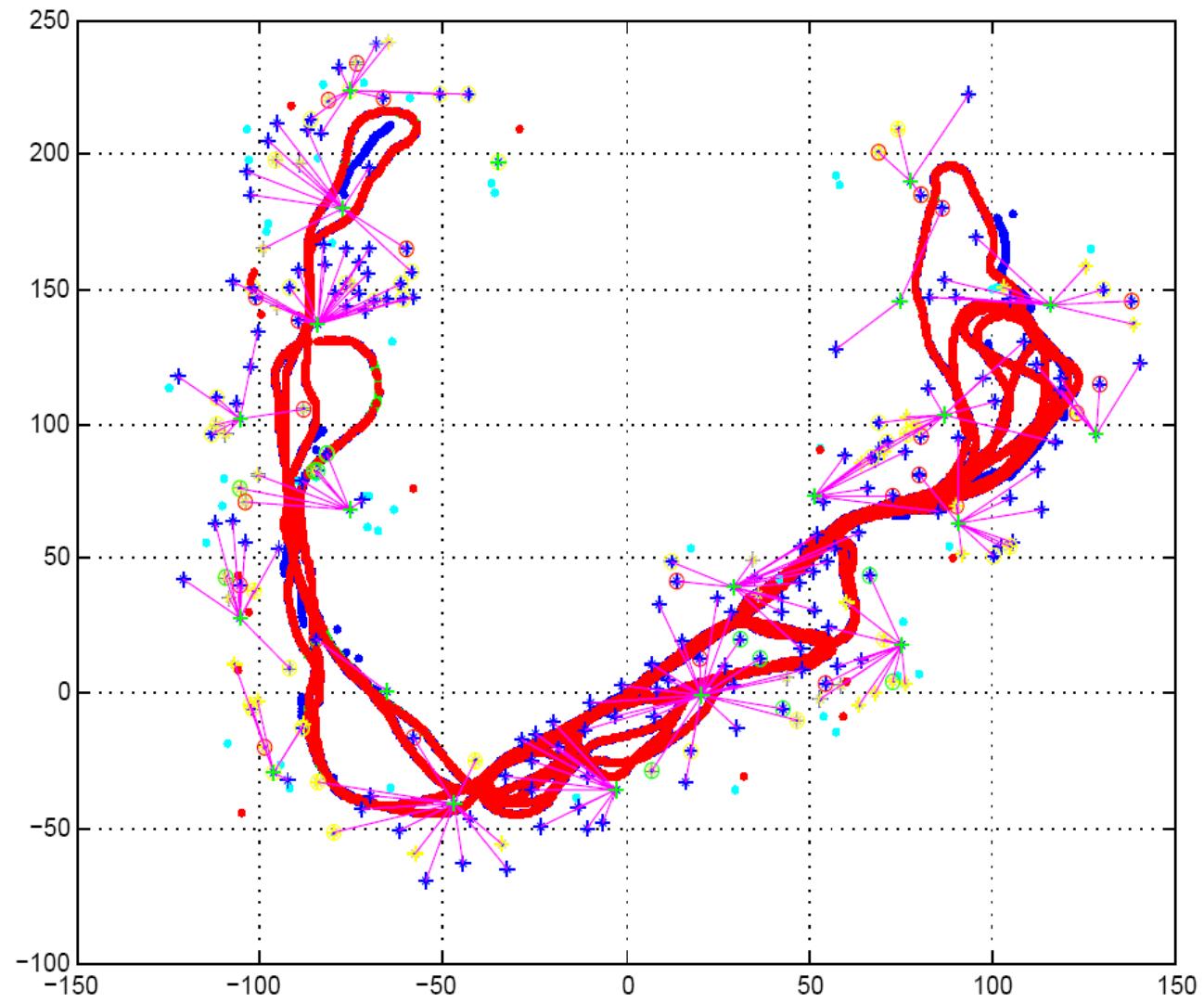
Courtesy of E. Nebot

# Victoria Park: Data Acquisition



Courtesy of E. Nebot

# Victoria Park: EKF Estimate



Courtesy of E. Nebot

# Victoria Park: Landmarks



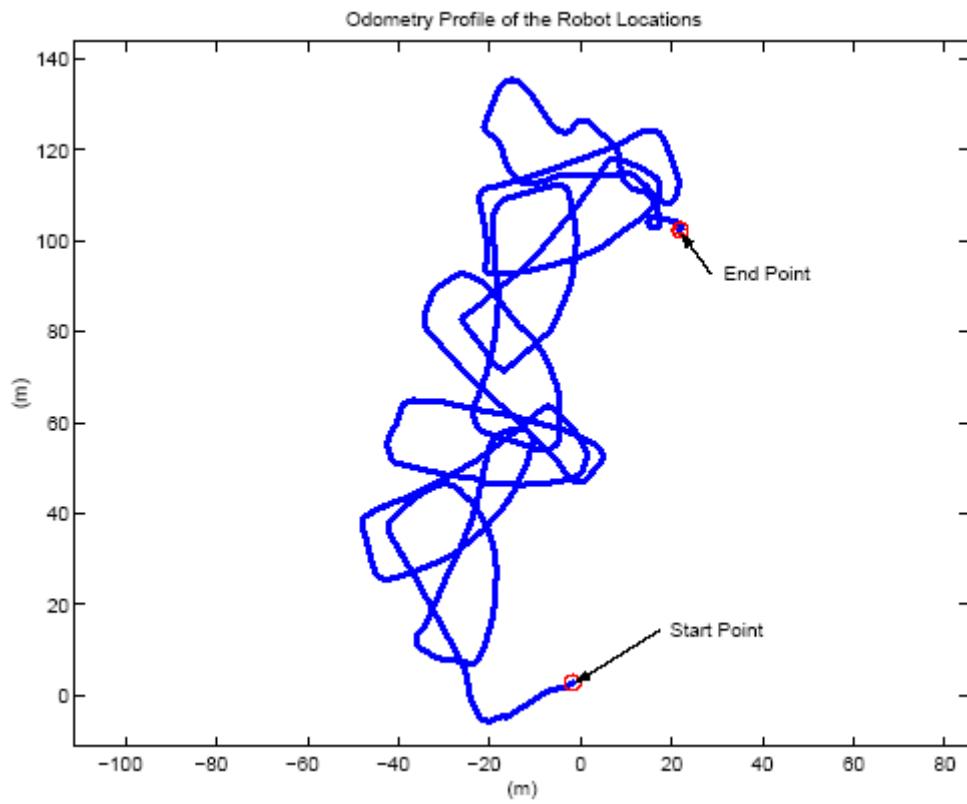
Courtesy of E. Nebot

# Example: Tennis Court Dataset

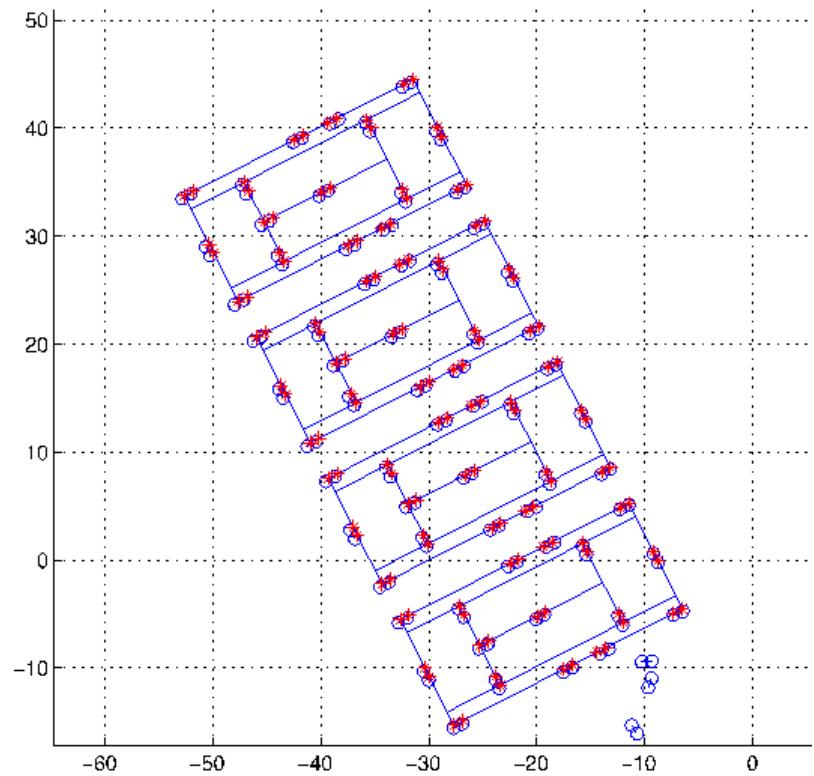


Courtesy of J. Leonard and M. Walter

# EKF SLAM: Tennis Court



odometry



estimated trajectory

# EKF-SLAM Complexity

- Cubic complexity depends only on the measurement dimensionality
- Cost per step: dominated by the number of landmarks:  $O(n^2)$
- Memory consumption:  $O(n^2)$
- The EKF becomes computationally intractable for large maps!

# EKF-SLAM Summary

- The first SLAM solution
- Convergence proof for the linear Gaussian case
- Can diverge if non-linearities are large (and the reality is non-linear...)
- Can deal only with a single mode
- Successful in medium-scale scenes
- Approximations exists to reduce the computational complexity

# Literature

## EKF SLAM

- Thrun et al.: “Probabilistic Robotics”, Chapter 10