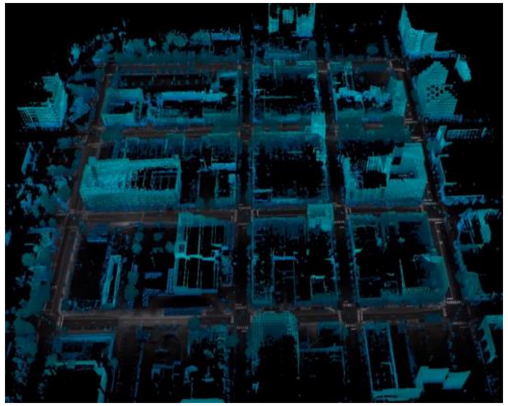
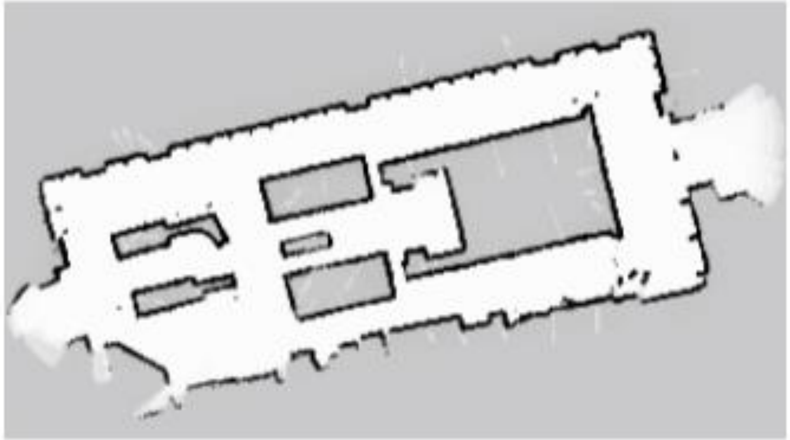
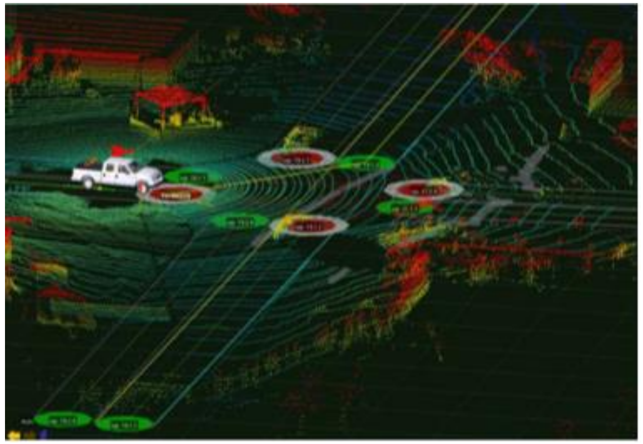
BYU Electrical & Computer Engineering





EKF SLAM AND DATA ASSOCIATION

ECEN 633: Robotic Localization and Maping

Slides courtesy of Ryan Eustice.

The SLAM Problem

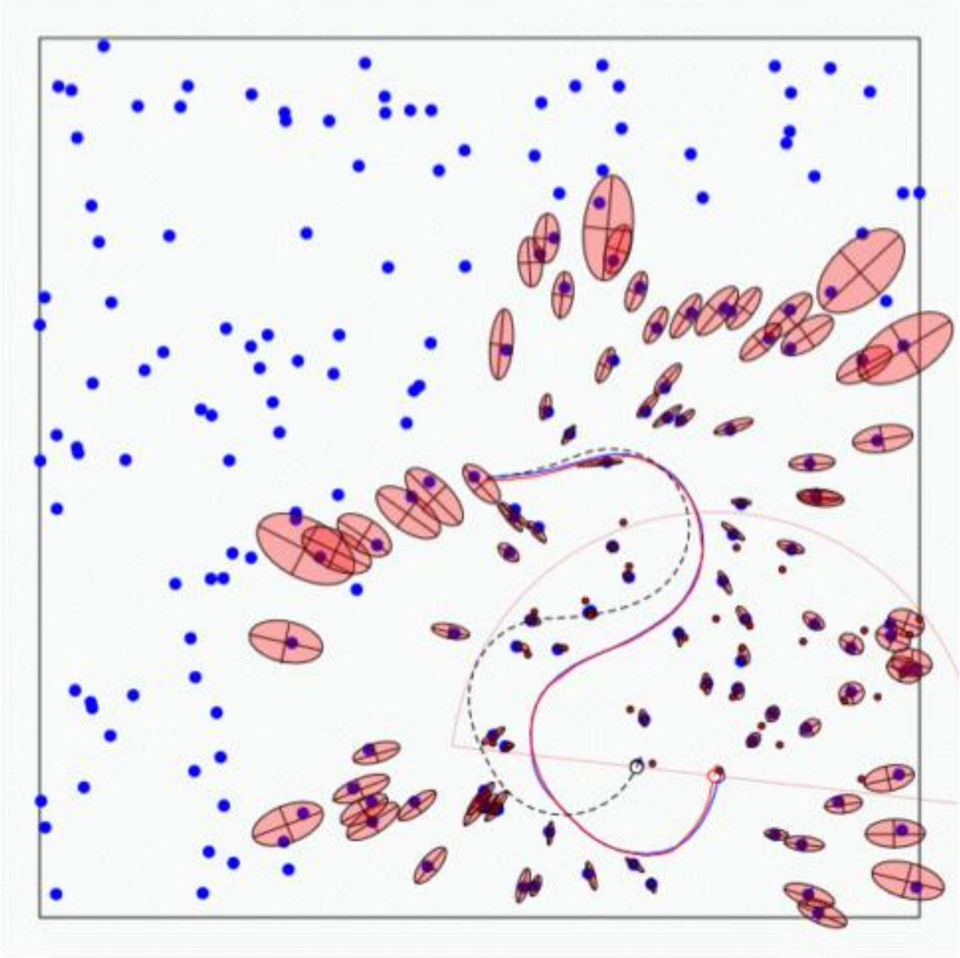
- A robot is exploring an unknown, static environment

Given:

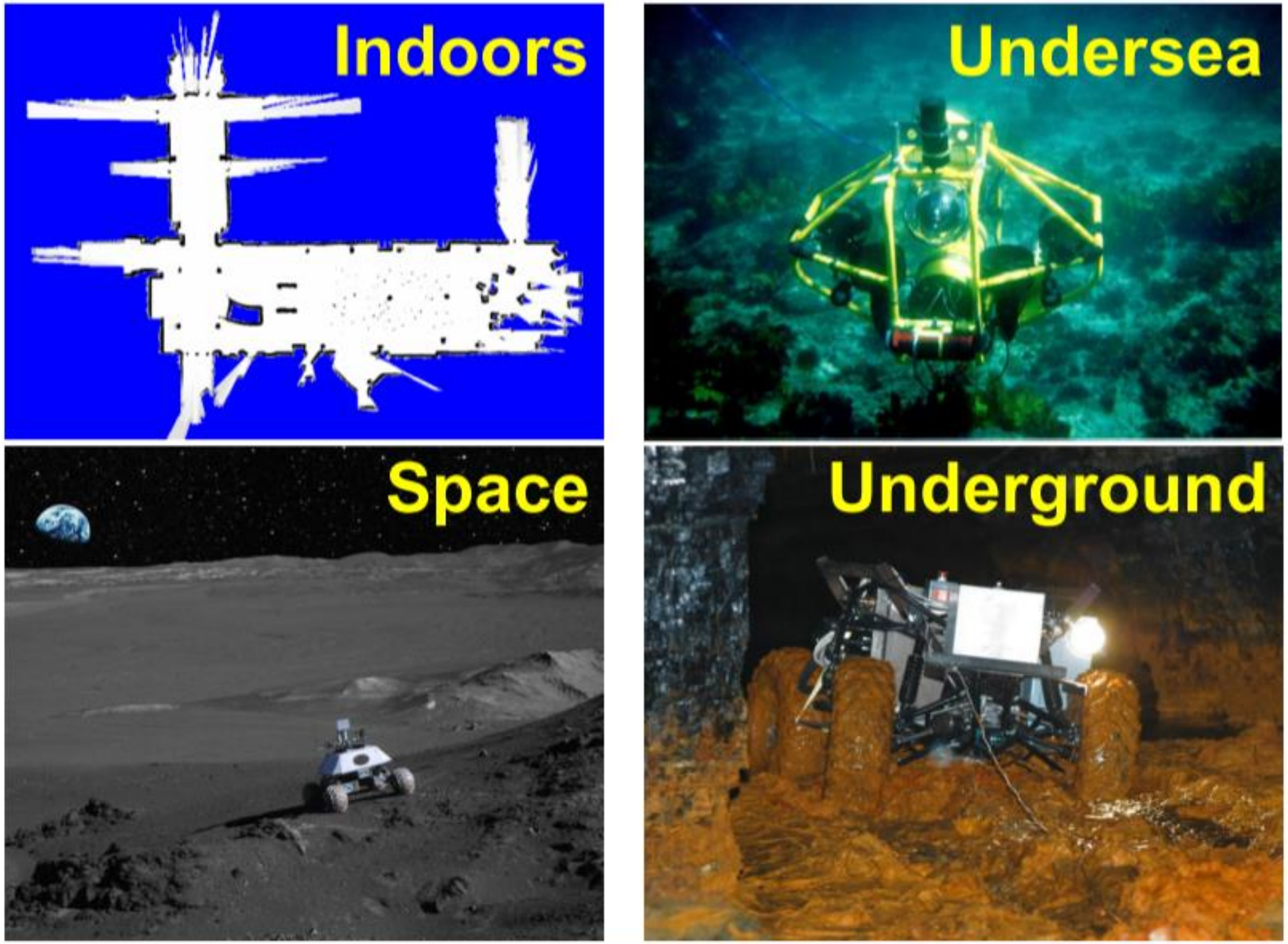
- The robot's controls
- Observations of nearby features

Estimate:

- Map of features
- Path of the robot



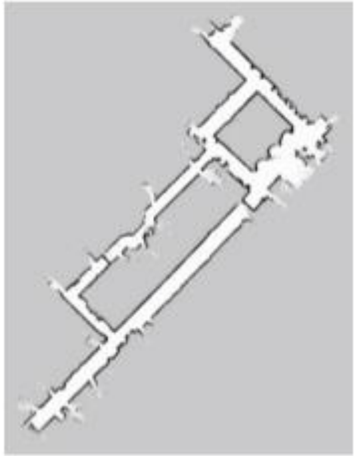
SLAM Applications



Representations

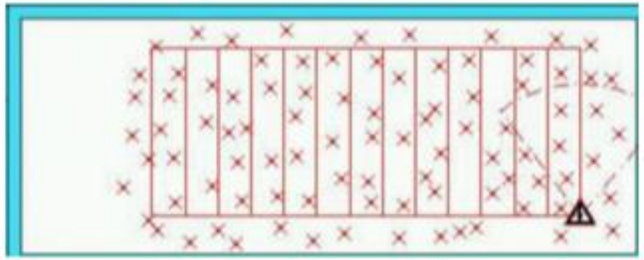
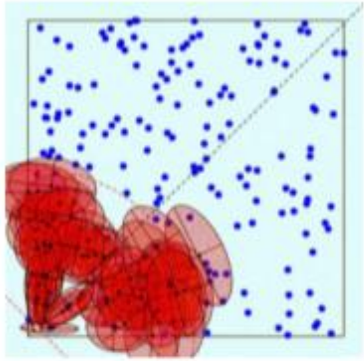
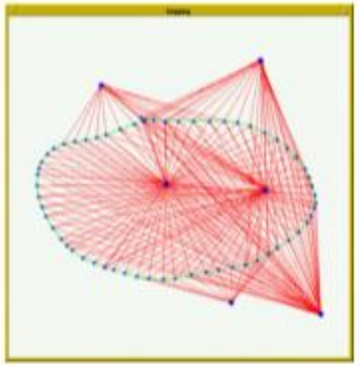
- Grid maps or scans





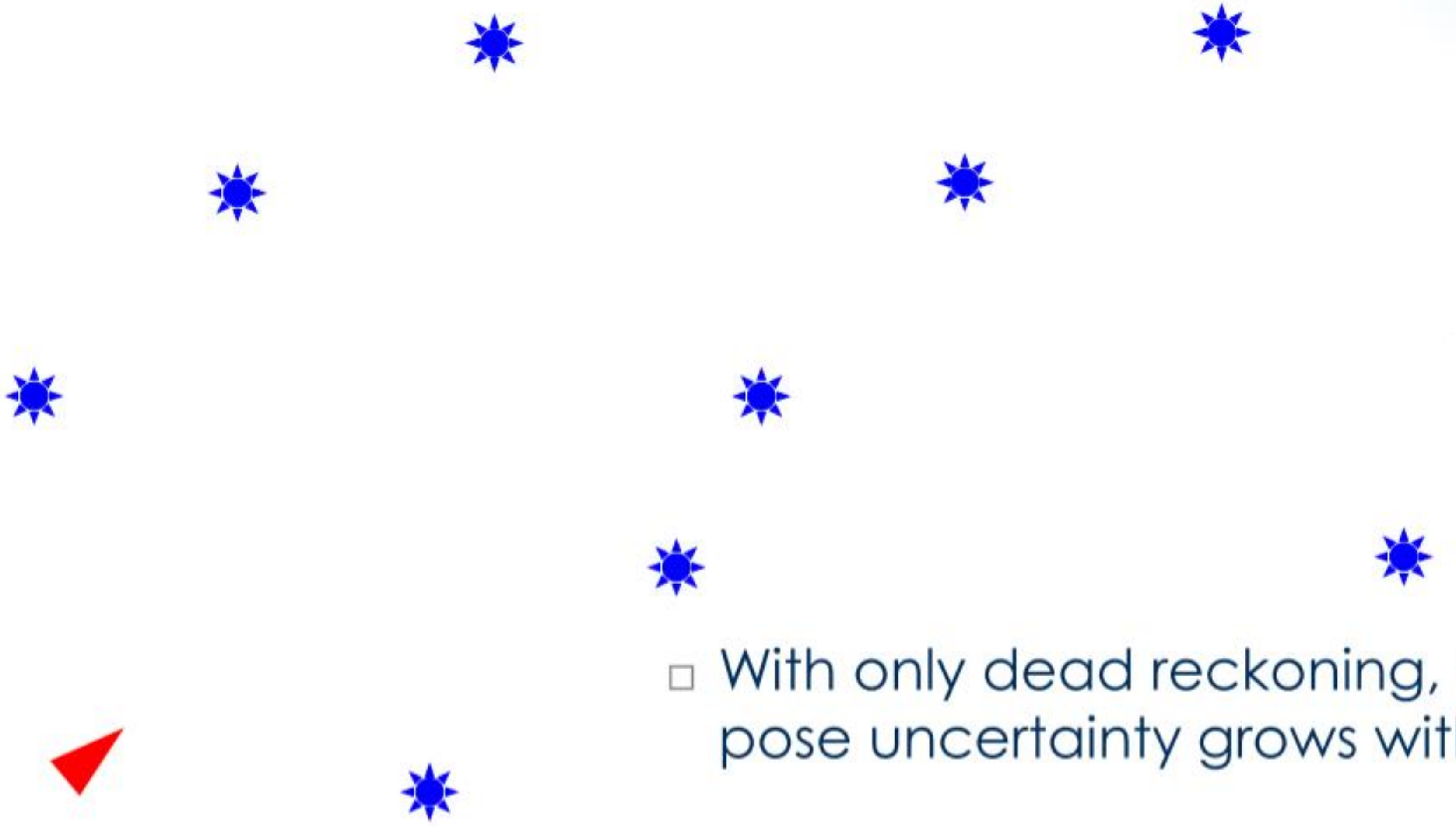
[Lu & Milios, 97; Gutmann, 98: Thrun 98; Burgard, 99; Konolige & Gutmann, 00; Thrun, 00; Arras, 99; Haehnel, 01;...]

- Landmark-based

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

Illustration of SLAM Without Landmarks



- With only dead reckoning, vehicle pose uncertainty grows without bound

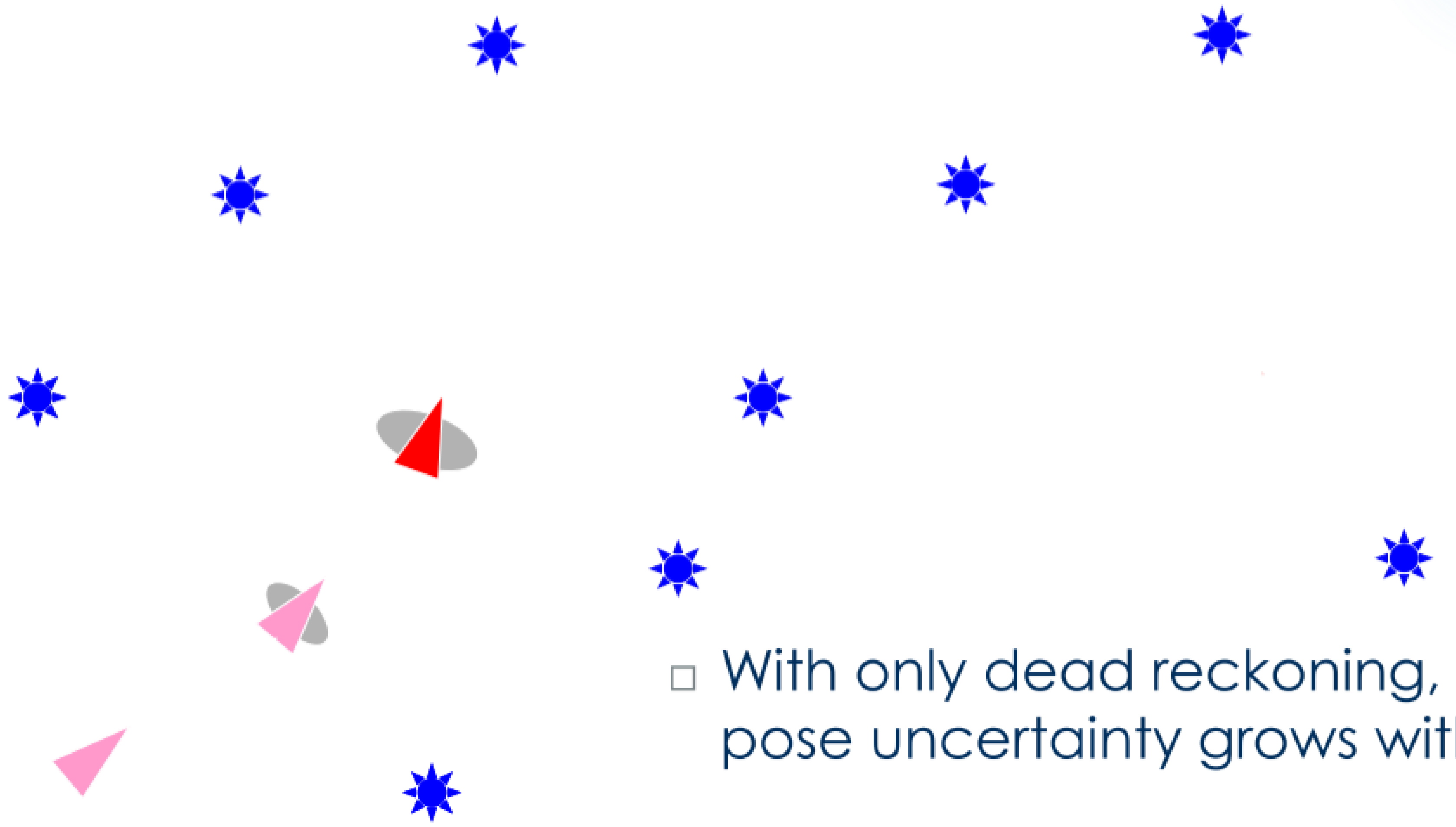
Courtesy J. Leonard

Illustration of SLAM Without Landmarks

- With only dead reckoning, vehicle pose uncertainty grows without bound

Courtesy J. Leonard

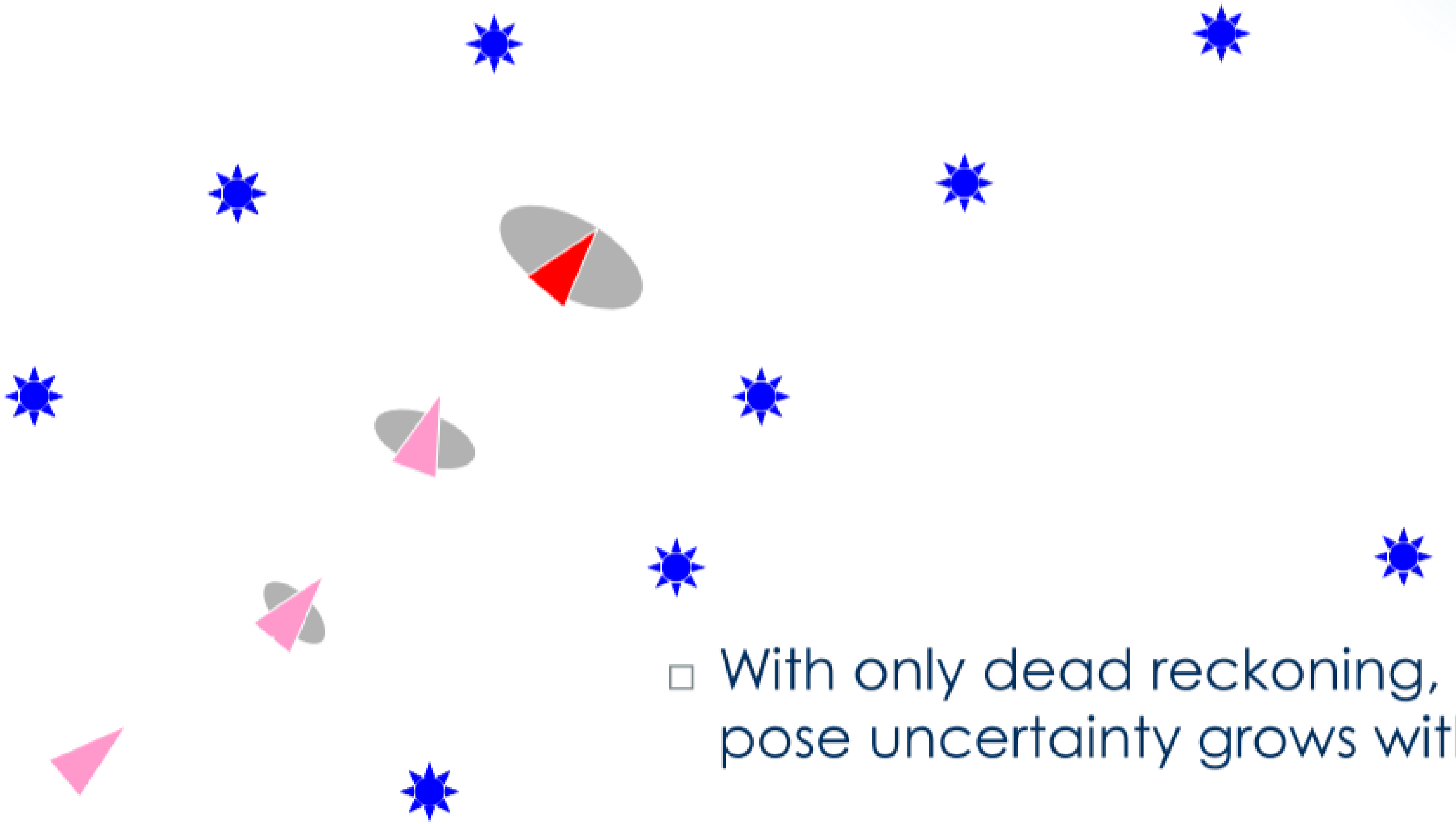
Illustration of SLAM Without Landmarks



- With only dead reckoning, vehicle pose uncertainty grows without bound

Courtesy J. Leonard

Illustration of SLAM Without Landmarks



- With only dead reckoning, vehicle pose uncertainty grows without bound

Courtesy J. Leonard

Illustration of SLAM Without Landmarks

- With only dead reckoning, vehicle pose uncertainty grows without bound

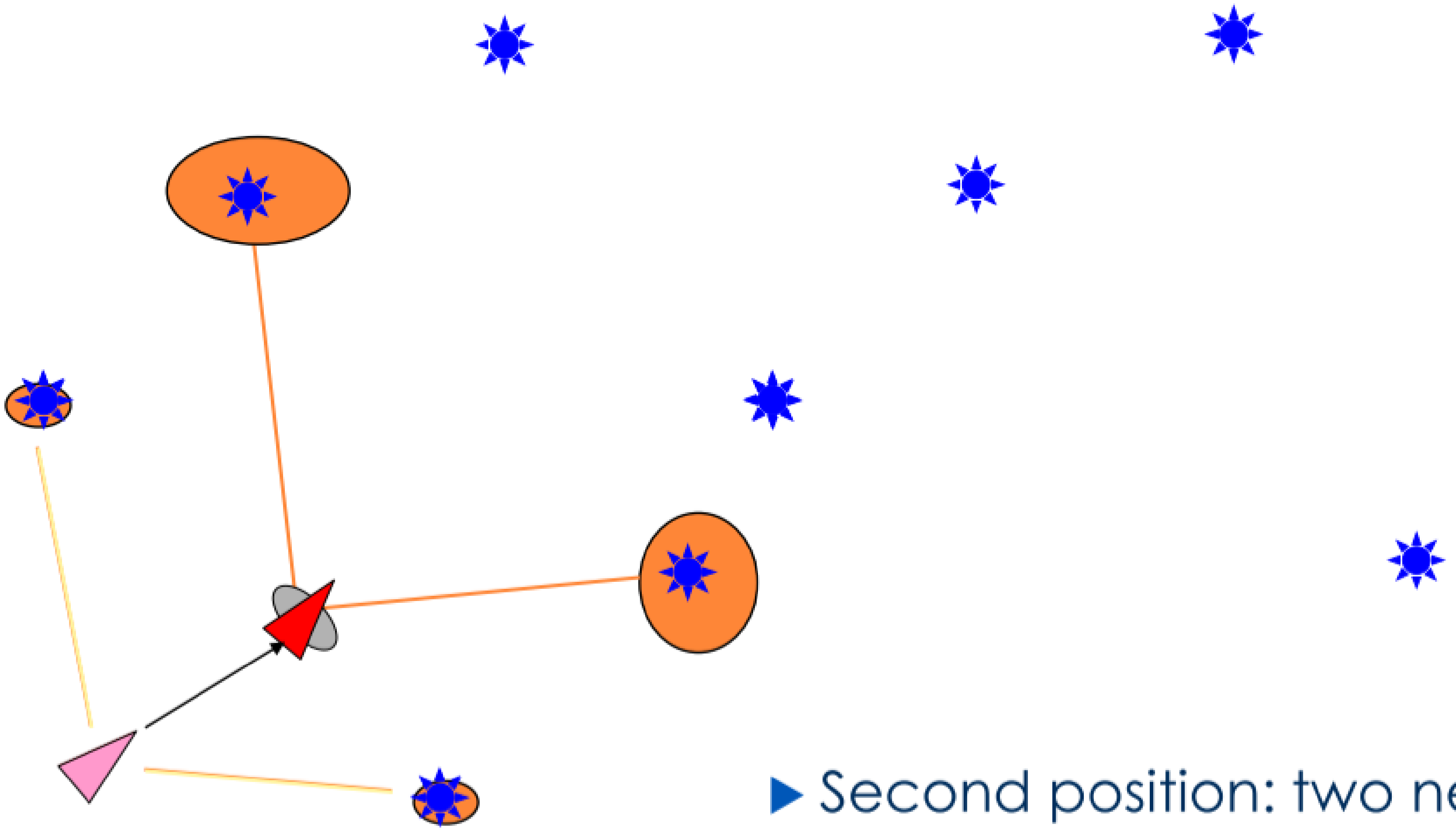
Courtesy J. Leonard

Illustration of SLAM Without Landmarks

- With only dead reckoning, vehicle pose uncertainty grows without bound

Courtesy J. Leonard

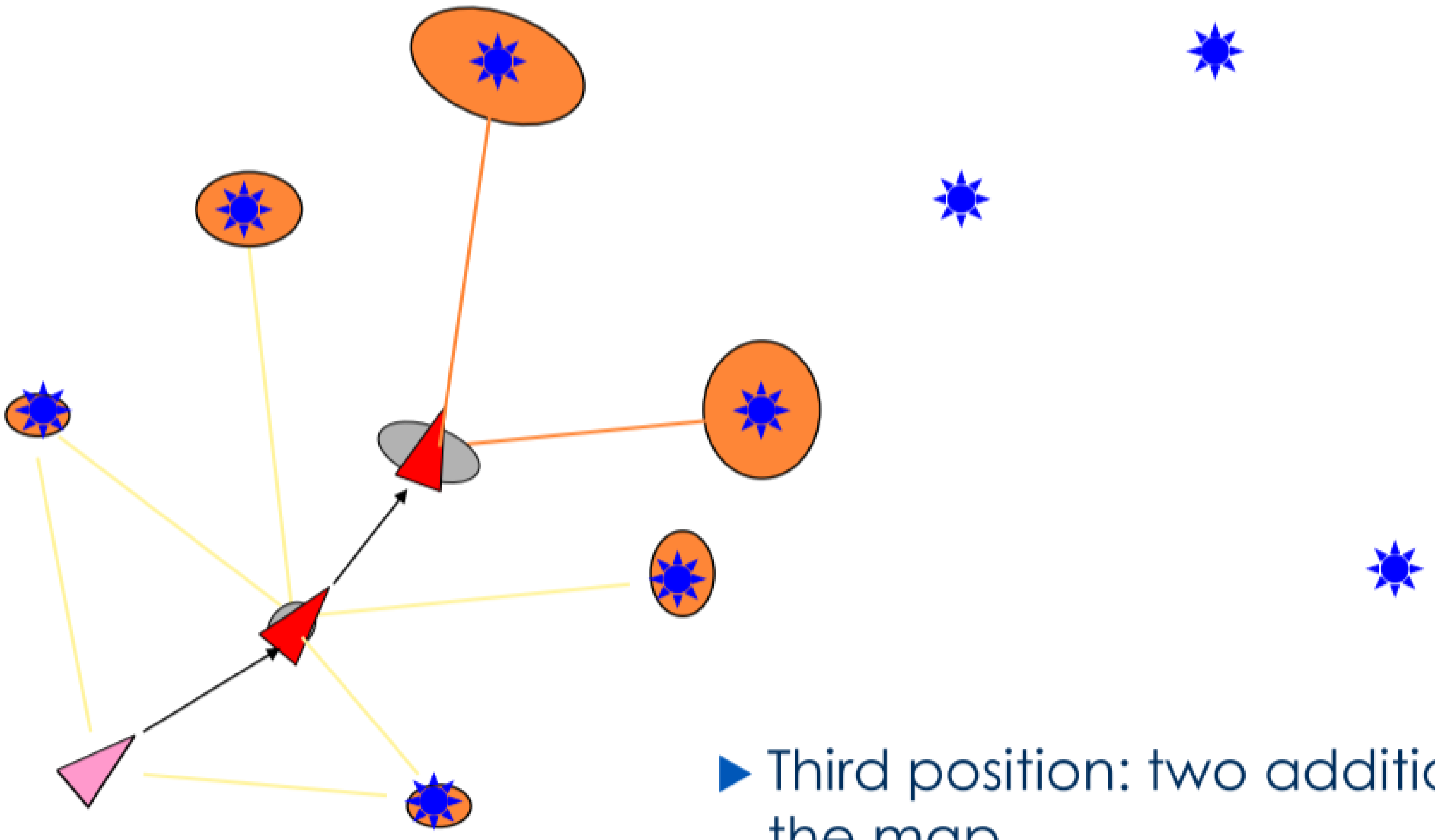
Repeat, with Measurements of Landmarks



- Second position: two new features observed

Courtesy J. Leonard

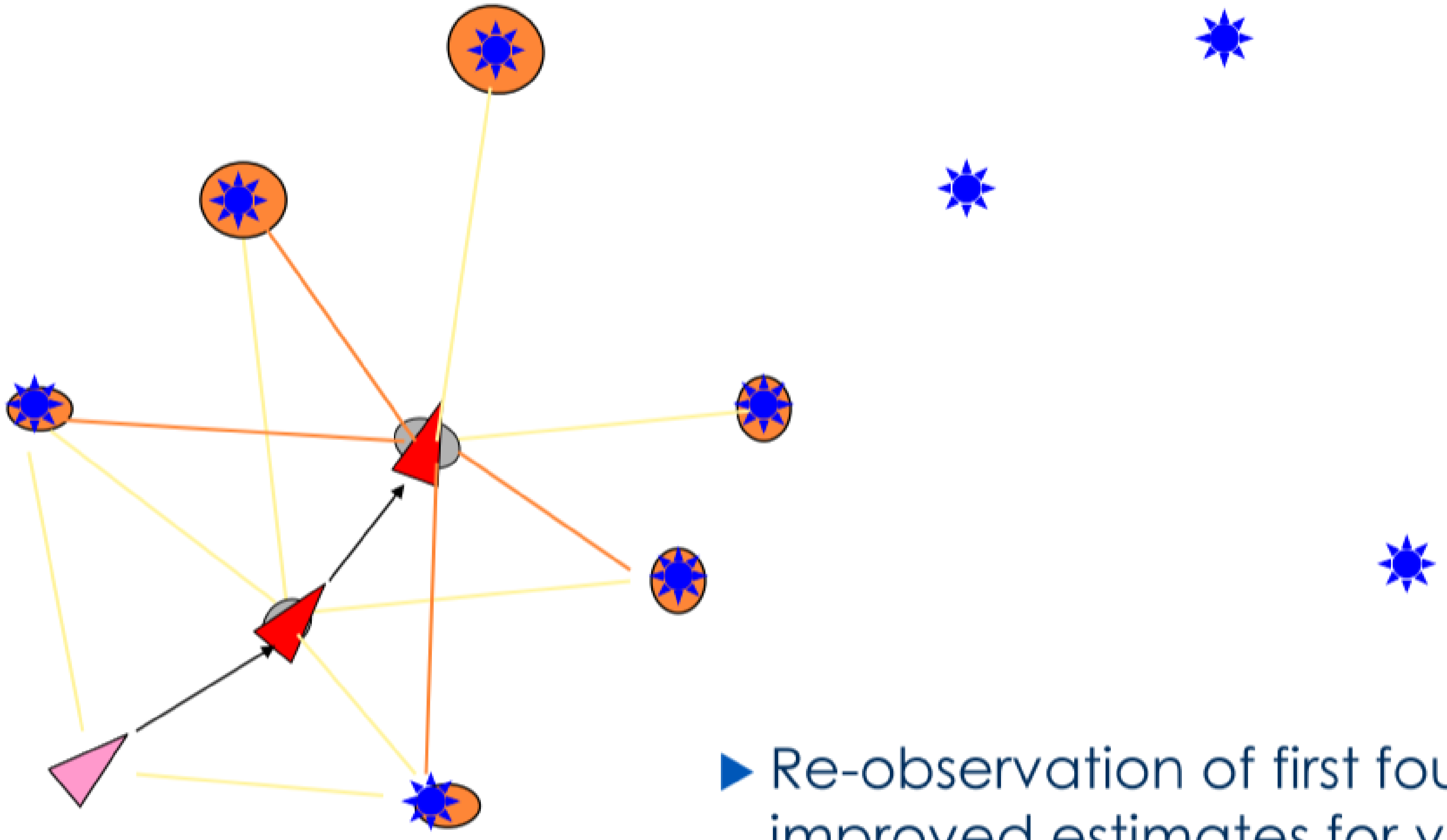
Repeat, with Measurements of Landmarks



- Third position: two additional features added to the map

Courtesy J. Leonard

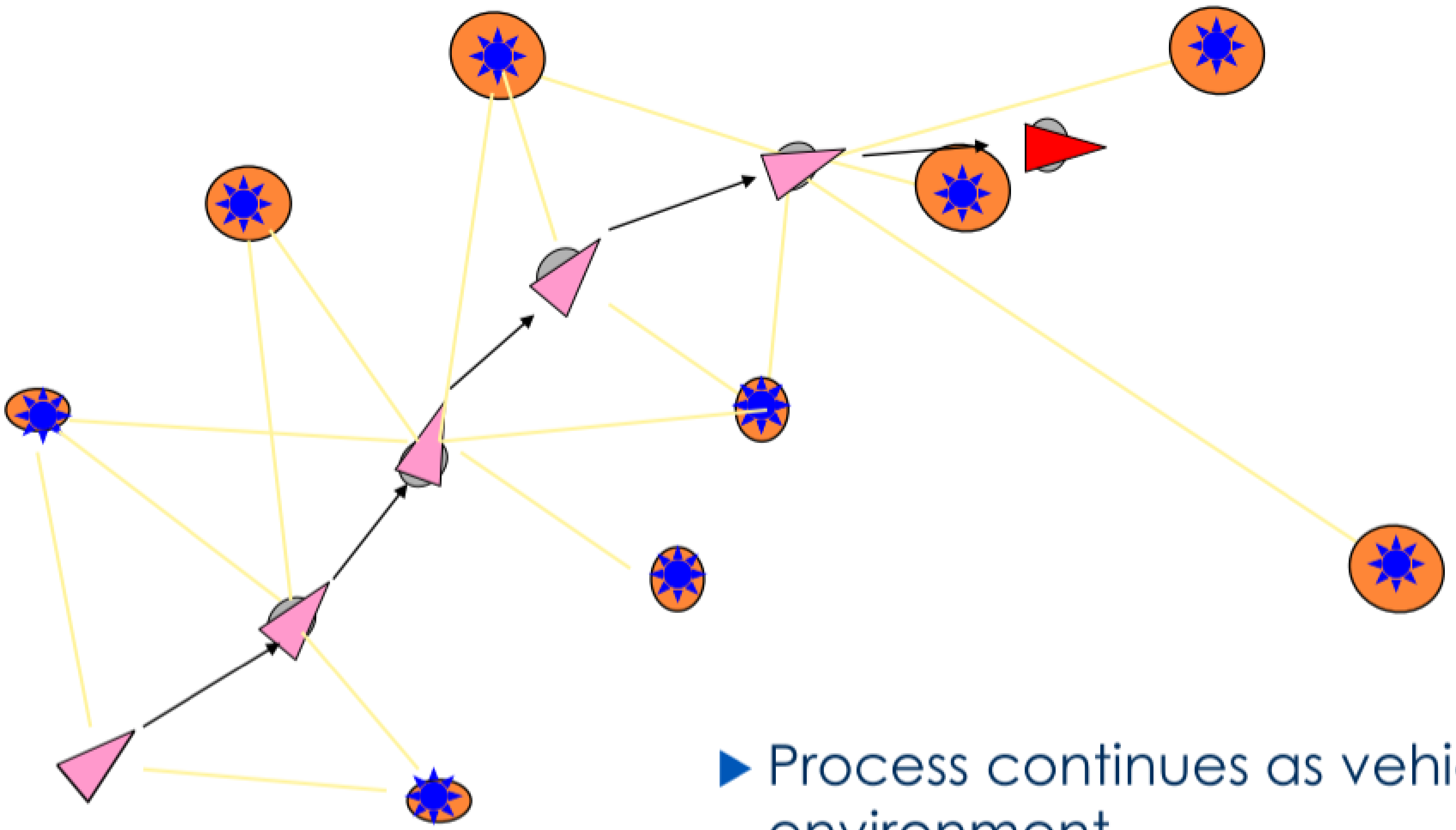
Repeat, with Measurements of Landmarks



- Re-observation of first four features results in improved estimates for vehicle and <u>all</u> features

Courtesy J. Leonard

Repeat, with Measurements of Landmarks

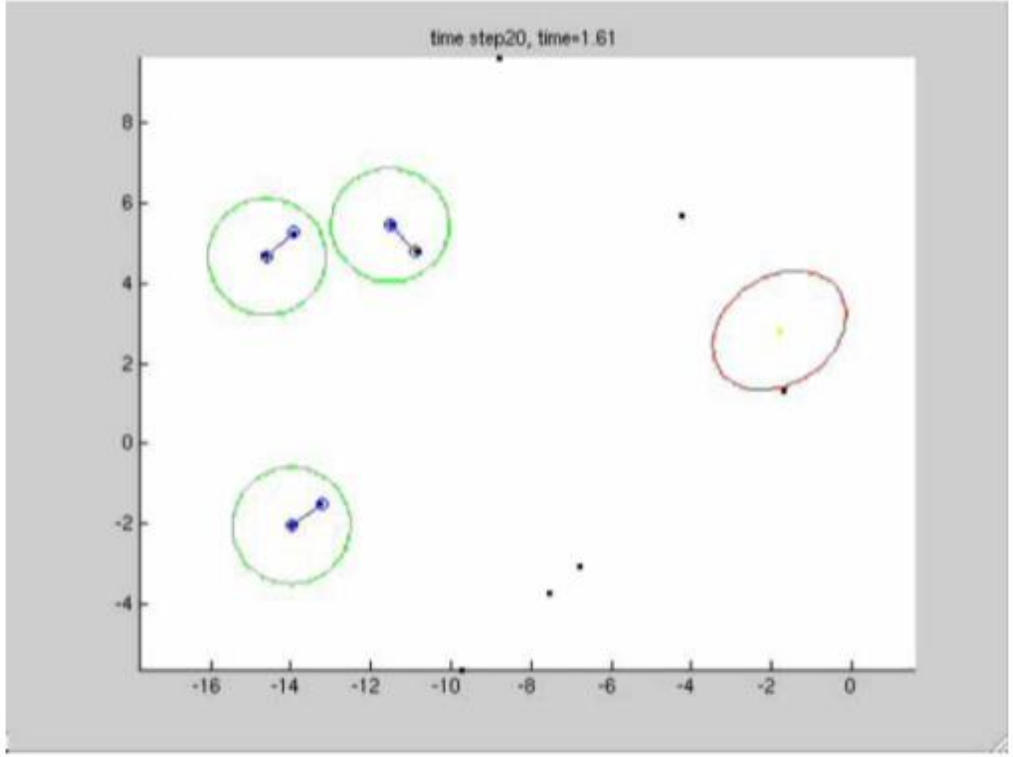


- Process continues as vehicle moves through environment

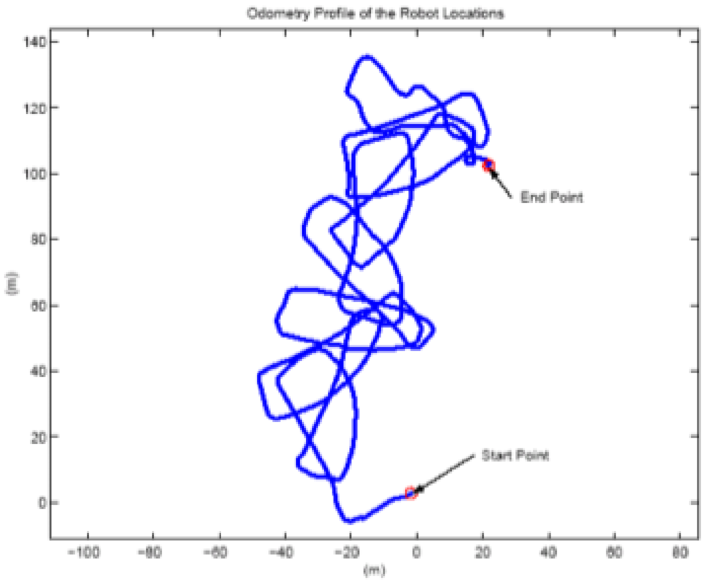
Courtesy J. Leonard

SLAM Using Landmarks

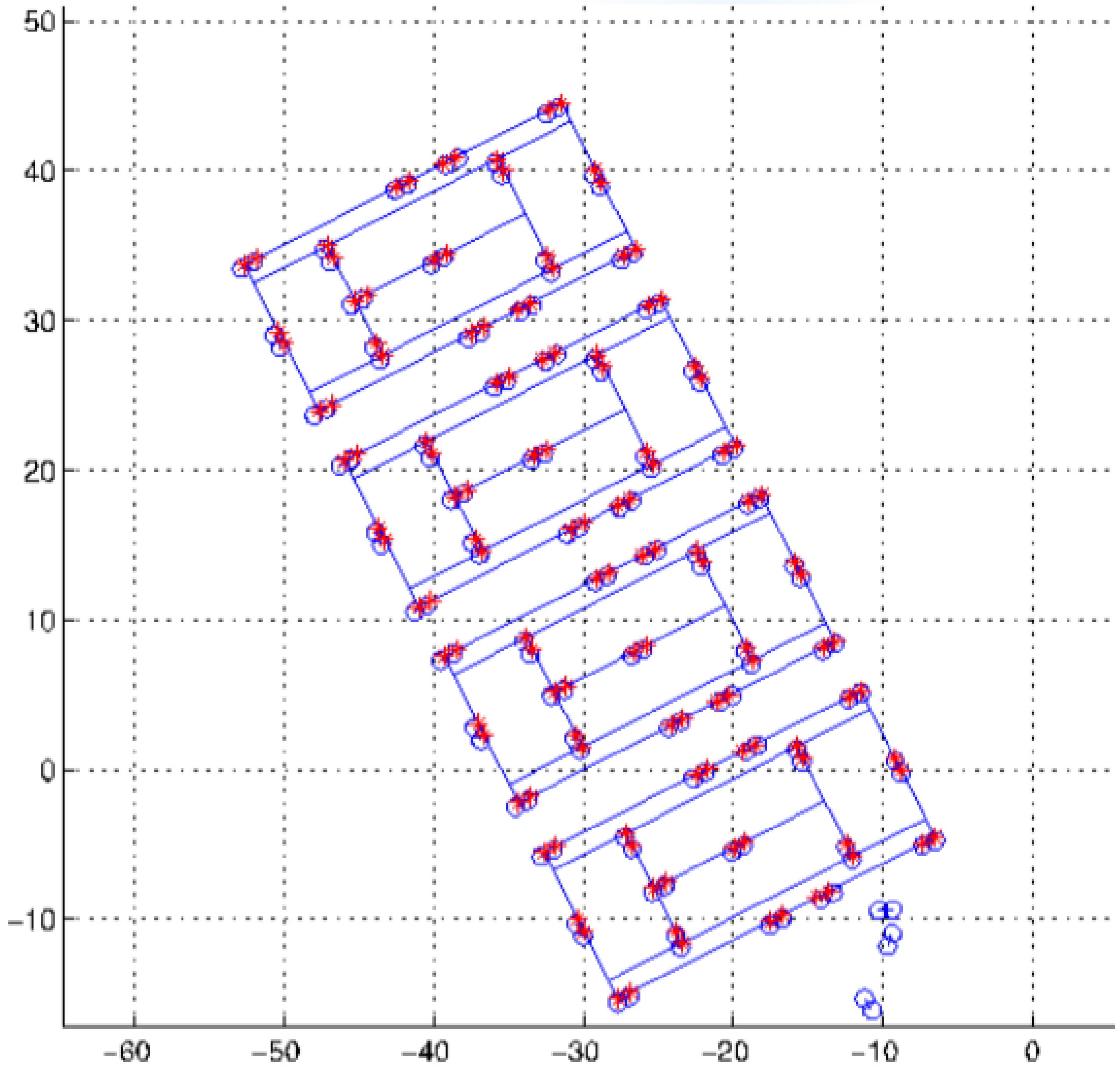
1. Move
2. Sense
3. Associate measurements with known features
4. Update state estimates for robot and previously mapped features
5. Find new features from unassociated measurements
6. Initialize new features
7. Repeat



Comparison with Ground Truth



odometry

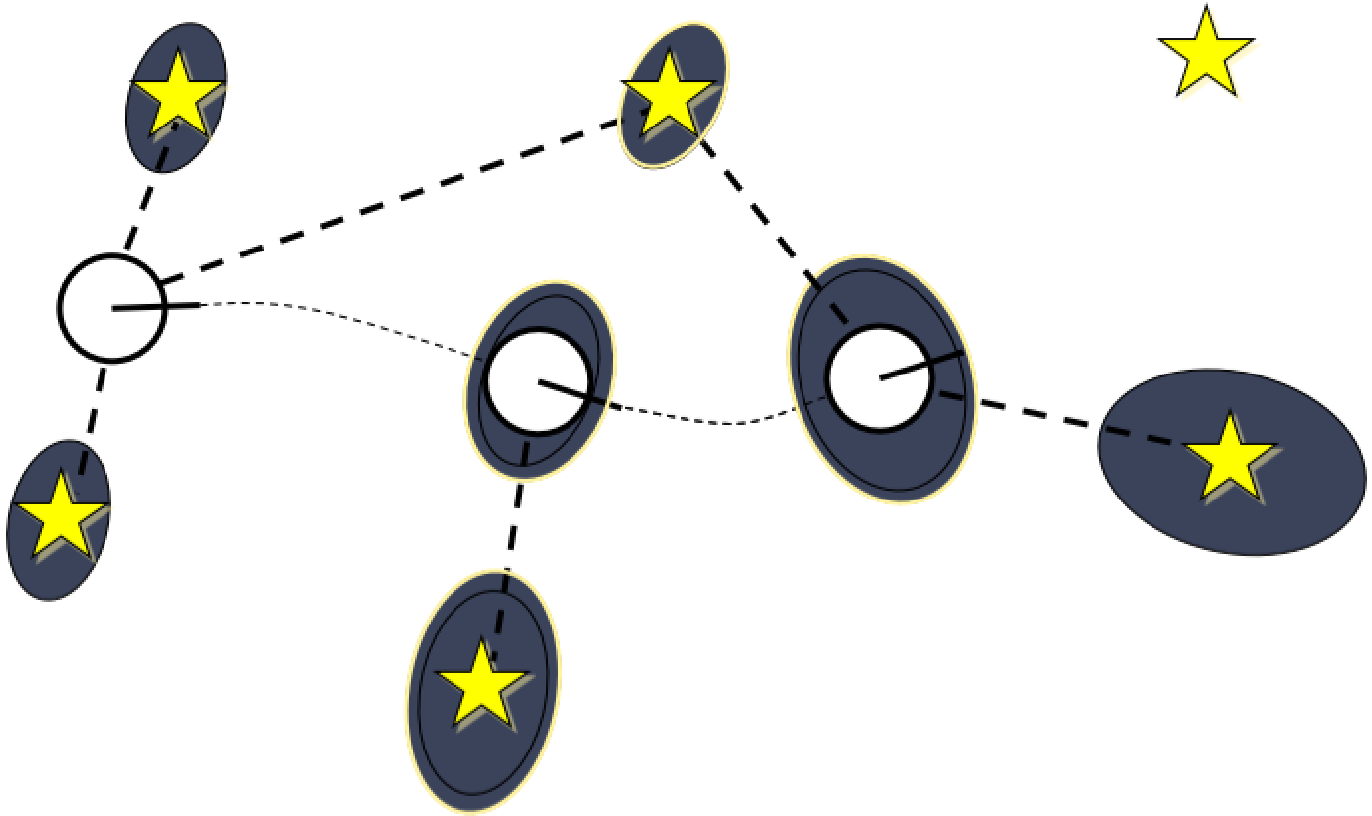


SLAM result

Courtesy J. Leonard

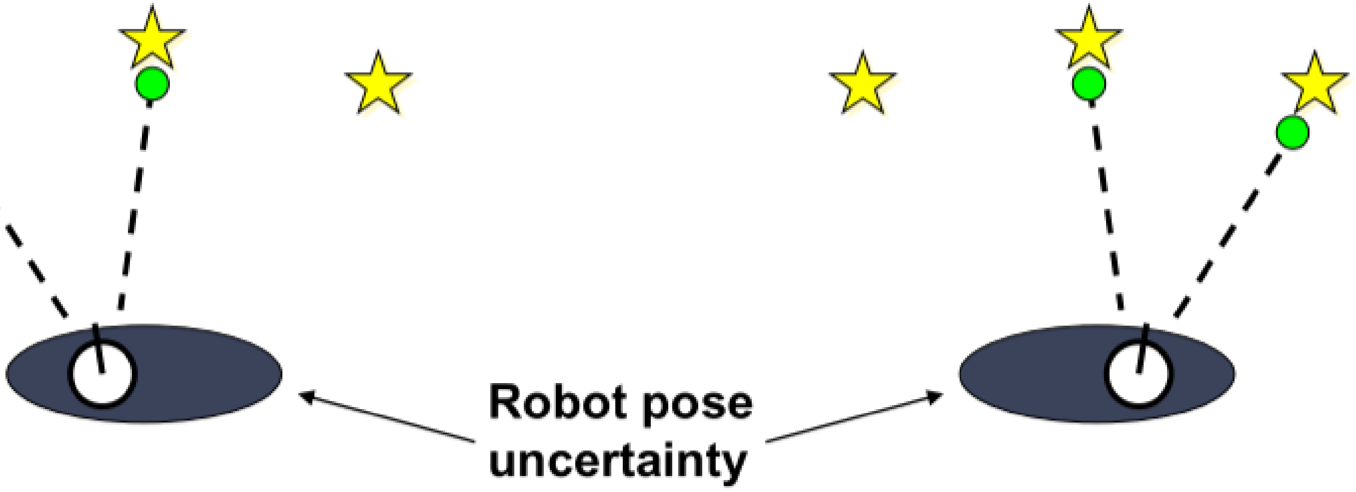
Why is SLAM a Hard Problem?

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



Robot path error correlates errors in the map

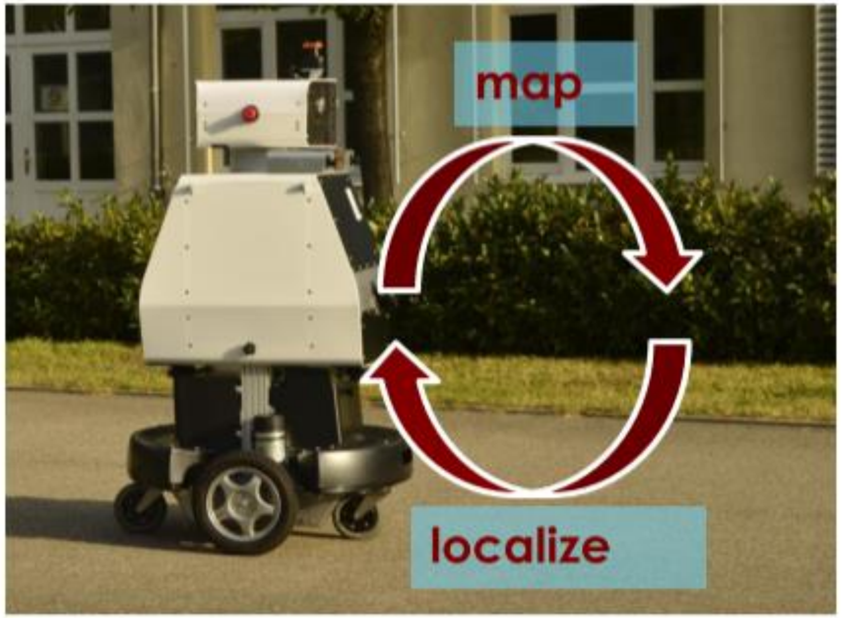
Why is SLAM a Hard Problem?



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

Simultaneous Localization and Mapping (SLAM)

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



Courtesy: Cyrill Stachniss

Three Main Paradigms

Kalman filter	Particle filter	Graph-based

Courtesy: Cyrill Stachniss

Bayes Filter

- Recursive filter with prediction and correction step

- Prediction

$$\overline{bel}(\mathbf{x}_t) = \int p(\mathbf{x}_t \mid \mathbf{u}_t, \mathbf{x}_{t-1})\ bel(\mathbf{x}_{t-1})\, d\mathbf{x}_{t-1}$$

- Correction

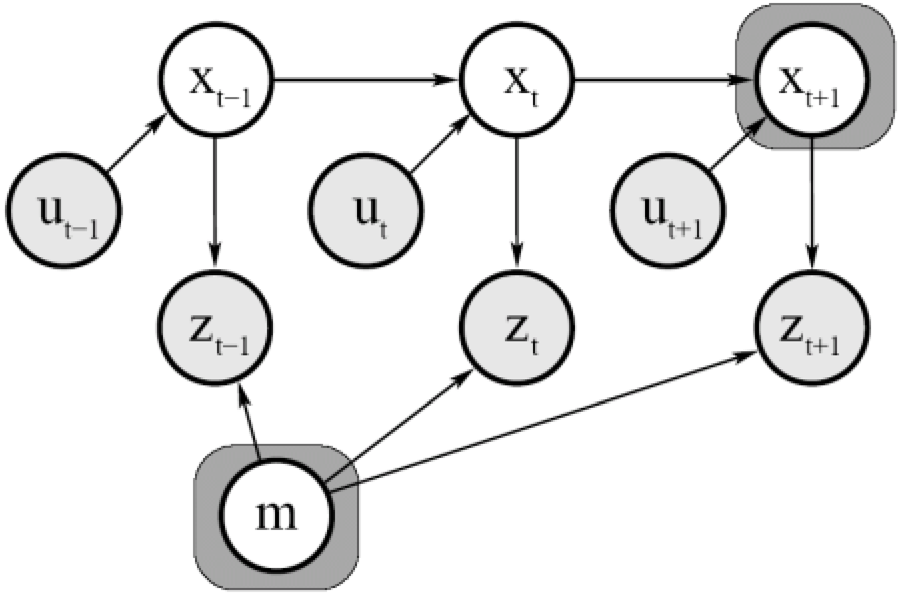
$$bel(\mathbf{x}_t) = \eta\ p(\mathbf{z}_t \mid \mathbf{x}_t)\ \overline{bel}(\mathbf{x}_t)$$

Courtesy: Cyrill Stachniss

EKF for Online SLAM

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

$$p(\mathbf{x}_t, \mathbf{m} \mid \mathbf{z}_{1:t}, \mathbf{u}_{1:t})$$



Courtesy: Thrun, Burgard, Fox

Extended Kalman Filter Algorithm

1: **Extended_Kalman_filter**$(\boldsymbol{\mu}_{t-1}, \Sigma_{t-1}, \mathbf{u}_t, \mathbf{z}_t)$**:**

2: $\bar{\boldsymbol{\mu}}_t = g(\mathbf{u}_t, \boldsymbol{\mu}_{t-1})$

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

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

5: $\boldsymbol{\mu}_t = \bar{\boldsymbol{\mu}}_t + K_t(\mathbf{z}_t - h(\bar{\boldsymbol{\mu}}_t))$

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

7: *return* $\boldsymbol{\mu}_t, \Sigma_t$

Courtesy: Cyrill Stachniss

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

$$\mathbf{y}_t = [\ \underbrace{x, y, \theta}_{\mathbf{x}_t\text{, robot's pose}}\ ,\ \underbrace{m_{1,x}, m_{1,y}}_{\mathbf{m}_1\text{, landmark 1}}\ , \ldots,\ \underbrace{m_{n,x}, m_{n,y}}_{\mathbf{m}_n\text{, landmark n}}\ ]^\top$$

Courtesy: Cyrill Stachniss

EKF SLAM: State Representation

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

$$\underbrace{\begin{pmatrix} x \\ y \\ \theta \\ m_{1,x} \\ m_{1,y} \\ \vdots \\ m_{n,x} \\ m_{n,y} \end{pmatrix}}_{\mu}
\quad
\underbrace{\begin{pmatrix}
\sigma_{xx} & \sigma_{xy} & \sigma_{x\theta} & \sigma_{xm_{1,x}} & \sigma_{xm_{1,y}} & \cdots & \sigma_{xm_{n,x}} & \sigma_{xm_{n,y}} \\
\sigma_{yx} & \sigma_{yy} & \sigma_{y\theta} & \sigma_{ym_{1,x}} & \sigma_{ym_{1,y}} & \cdots & \sigma_{m_{n,x}} & \sigma_{m_{n,y}} \\
\sigma_{\theta x} & \sigma_{\theta y} & \sigma_{\theta\theta} & \sigma_{\theta m_{1,x}} & \sigma_{\theta m_{1,y}} & \cdots & \sigma_{\theta m_{n,x}} & \sigma_{\theta m_{n,y}} \\
\sigma_{m_{1,x}x} & \sigma_{m_{1,x}y} & \sigma_{\theta} & \sigma_{m_{1,x}m_{1,x}} & \sigma_{m_{1,x}m_{1,y}} & \cdots & \sigma_{m_{1,x}m_{n,x}} & \sigma_{m_{1,x}m_{n,y}} \\
\sigma_{m_{1,y}x} & \sigma_{m_{1,y}y} & \sigma_{\theta} & \sigma_{m_{1,y}m_{1,x}} & \sigma_{m_{1,y}m_{1,y}} & \cdots & \sigma_{m_{1,y}m_{n,x}} & \sigma_{m_{1,y}m_{n,y}} \\
\vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\
\sigma_{m_{n,x}x} & \sigma_{m_{n,x}y} & \sigma_{\theta} & \sigma_{m_{n,x}m_{1,x}} & \sigma_{m_{n,x}m_{1,y}} & \cdots & \sigma_{m_{n,x}m_{n,x}} & \sigma_{m_{n,x}m_{n,y}} \\
\sigma_{m_{n,y}x} & \sigma_{m_{n,y}y} & \sigma_{\theta} & \sigma_{m_{n,y}m_{1,x}} & \sigma_{m_{n,y}m_{1,y}} & \cdots & \sigma_{m_{n,y}m_{n,x}} & \sigma_{m_{n,y}m_{n,y}}
\end{pmatrix}}_{\Sigma}$$

Courtesy: Cyrill Stachniss

EKF SLAM: State Representation

- More compactly

$$\underbrace{\begin{pmatrix} \mathbf{x}_R \\ \mathbf{m}_1 \\ \vdots \\ \mathbf{m}_n \end{pmatrix}}_{\boldsymbol{\mu}} \quad \underbrace{\begin{pmatrix} \Sigma_{\mathbf{x}_R\mathbf{x}_R} & \Sigma_{\mathbf{x}_R\mathbf{m}_1} & \cdots & \Sigma_{\mathbf{x}_R\mathbf{m}_n} \\ \Sigma_{\mathbf{m}_1\mathbf{x}_R} & \Sigma_{\mathbf{m}_1\mathbf{m}_1} & \cdots & \Sigma_{\mathbf{m}_1\mathbf{m}_n} \\ \vdots & \vdots & \ddots & \vdots \\ \Sigma_{\mathbf{m}_n\mathbf{x}_R} & \Sigma_{\mathbf{m}_n\mathbf{m}_1} & \cdots & \Sigma_{\mathbf{m}_n\mathbf{m}_n} \end{pmatrix}}_{\Sigma}$$

Courtesy: Cyrill Stachniss

EKF SLAM: State Representation

- Even more compactly (note: $\mathbf{x}_R \to \mathbf{x}$)

$$\underbrace{\begin{pmatrix} \mathbf{x} \\ \mathbf{m} \end{pmatrix}}_{\boldsymbol{\mu}} \quad \underbrace{\begin{pmatrix} \Sigma_{\mathbf{xx}} & \Sigma_{\mathbf{xm}} \\ \Sigma_{\mathbf{mx}} & \Sigma_{\mathbf{mm}} \end{pmatrix}}_{\Sigma}$$

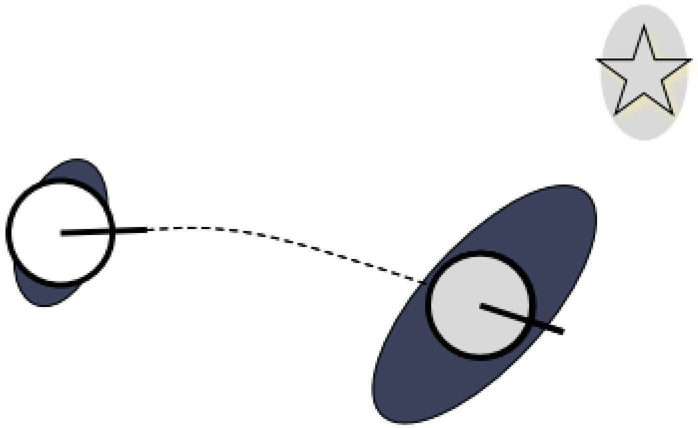
Courtesy: Cyrill Stachniss

EKF SLAM: Filter Cycle

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

Courtesy: Cyrill Stachniss

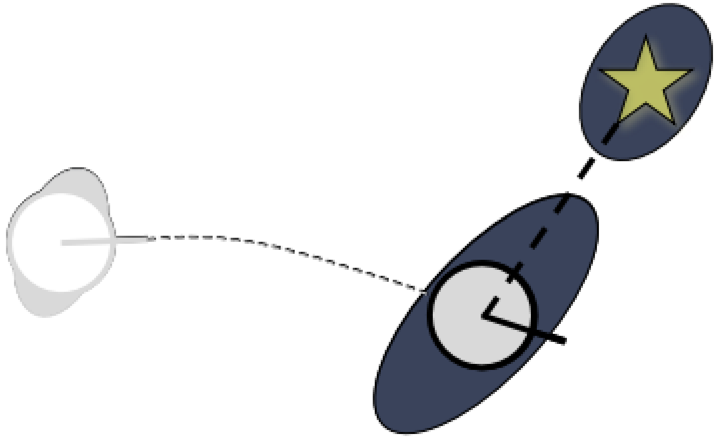
EKF SLAM: State Prediction



$$
\underbrace{\begin{pmatrix} \mathbf{x}_R \\ \mathbf{m}_1 \\ \vdots \\ \mathbf{m}_n \end{pmatrix}}_{\boldsymbol{\mu}}
\quad
\underbrace{\begin{pmatrix} \Sigma_{\mathbf{x}_R\mathbf{x}_R} & \Sigma_{\mathbf{x}_R\mathbf{m}_1} & \cdots & \Sigma_{\mathbf{x}_R\mathbf{m}_n} \\ \Sigma_{\mathbf{m}_1\mathbf{x}_R} & \Sigma_{\mathbf{m}_1\mathbf{m}_1} & \cdots & \Sigma_{\mathbf{m}_1\mathbf{m}_n} \\ \vdots & \vdots & \ddots & \vdots \\ \Sigma_{\mathbf{m}_n\mathbf{x}_R} & \Sigma_{\mathbf{m}_n\mathbf{m}_1} & \cdots & \Sigma_{\mathbf{m}_n\mathbf{m}_n} \end{pmatrix}}_{\Sigma}
$$

Courtesy: Cyrill Stachniss

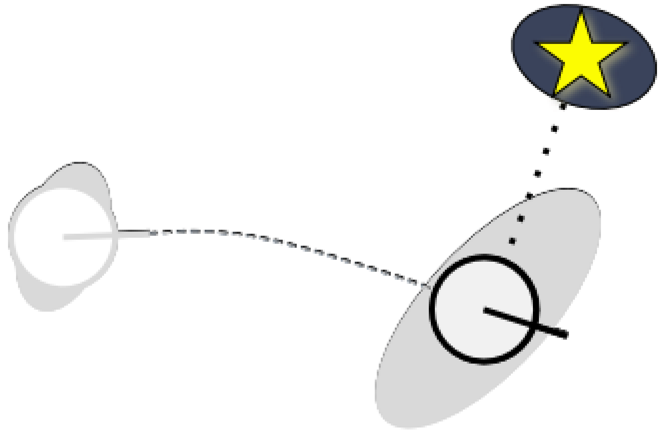
EKF SLAM: Measurement Prediction



$$\underbrace{\begin{pmatrix} \mathbf{x}_R \\ \mathbf{m}_1 \\ \vdots \\ \mathbf{m}_n \end{pmatrix}}_{\boldsymbol{\mu}} \underbrace{\begin{pmatrix} \Sigma_{\mathbf{x}_R\mathbf{x}_R} & \Sigma_{\mathbf{x}_R\mathbf{m}_1} & \cdots & \Sigma_{\mathbf{x}_R\mathbf{m}_n} \\ \Sigma_{\mathbf{m}_1\mathbf{x}_R} & \Sigma_{\mathbf{m}_1\mathbf{m}_1} & \cdots & \Sigma_{\mathbf{m}_1\mathbf{m}_n} \\ \vdots & \vdots & \ddots & \vdots \\ \Sigma_{\mathbf{m}_n\mathbf{x}_R} & \Sigma_{\mathbf{m}_n\mathbf{m}_1} & \cdots & \Sigma_{\mathbf{m}_n\mathbf{m}_n} \end{pmatrix}}_{\Sigma}$$

Courtesy: Cyrill Stachniss

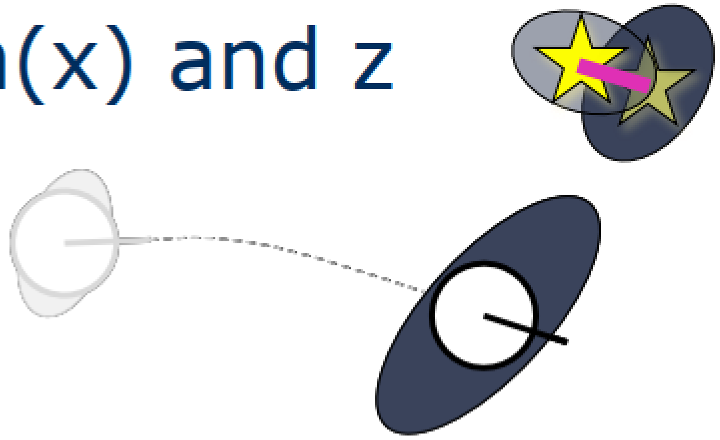
EKF SLAM: Obtained Measurement



$$\underbrace{\begin{pmatrix} \mathbf{x}_R \\ \mathbf{m}_1 \\ \vdots \\ \mathbf{m}_n \end{pmatrix}}_{\boldsymbol{\mu}} \quad \underbrace{\begin{pmatrix} \Sigma_{\mathbf{x}_R\mathbf{x}_R} & \Sigma_{\mathbf{x}_R\mathbf{m}_1} & \cdots & \Sigma_{\mathbf{x}_R\mathbf{m}_n} \\ \Sigma_{\mathbf{m}_1\mathbf{x}_R} & \Sigma_{\mathbf{m}_1\mathbf{m}_1} & \cdots & \Sigma_{\mathbf{m}_1\mathbf{m}_n} \\ \vdots & \vdots & \ddots & \vdots \\ \Sigma_{\mathbf{m}_n\mathbf{x}_R} & \Sigma_{\mathbf{m}_n\mathbf{m}_1} & \cdots & \Sigma_{\mathbf{m}_n\mathbf{m}_n} \end{pmatrix}}_{\Sigma}$$

Courtesy: Cyrill Stachniss

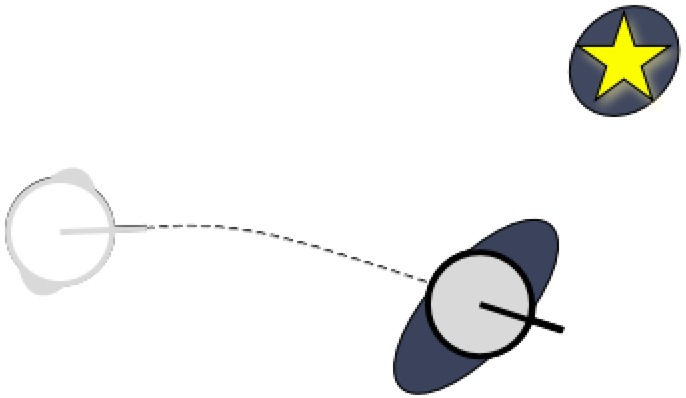
EKF SLAM: Data Association and Difference Between h(x) and z



$$\underbrace{\begin{pmatrix} \mathbf{x}_R \\ \mathbf{m}_1 \\ \vdots \\ \mathbf{m}_n \end{pmatrix}}_{\boldsymbol{\mu}} \quad \underbrace{\begin{pmatrix} \Sigma_{\mathbf{x}_R\mathbf{x}_R} & \Sigma_{\mathbf{x}_R\mathbf{m}_1} & \cdots & \Sigma_{\mathbf{x}_R\mathbf{m}_n} \\ \Sigma_{\mathbf{m}_1\mathbf{x}_R} & \Sigma_{\mathbf{m}_1\mathbf{m}_1} & \cdots & \Sigma_{\mathbf{m}_1\mathbf{m}_n} \\ \vdots & \vdots & \ddots & \vdots \\ \Sigma_{\mathbf{m}_n\mathbf{x}_R} & \Sigma_{\mathbf{m}_n\mathbf{m}_1} & \cdots & \Sigma_{\mathbf{m}_n\mathbf{m}_n} \end{pmatrix}}_{\Sigma}$$

Courtesy: Cyrill Stachniss

EKF SLAM: Update Step



$$\underbrace{\begin{pmatrix} \mathbf{x}_R \\ \mathbf{m}_1 \\ \vdots \\ \mathbf{m}_n \end{pmatrix}}_{\boldsymbol{\mu}} \quad \underbrace{\begin{pmatrix} \Sigma_{\mathbf{x}_R\mathbf{x}_R} & \Sigma_{\mathbf{x}_R\mathbf{m}_1} & \cdots & \Sigma_{\mathbf{x}_R\mathbf{m}_n} \\ \Sigma_{\mathbf{m}_1\mathbf{x}_R} & \Sigma_{\mathbf{m}_1\mathbf{m}_1} & \cdots & \Sigma_{\mathbf{m}_1\mathbf{m}_n} \\ \vdots & \vdots & \ddots & \vdots \\ \Sigma_{\mathbf{m}_n\mathbf{x}_R} & \Sigma_{\mathbf{m}_n\mathbf{m}_1} & \cdots & \Sigma_{\mathbf{m}_n\mathbf{m}_n} \end{pmatrix}}_{\Sigma}$$

Courtesy: Cyrill Stachniss

EKF SLAM: Concrete Example

Setup

- Robot moves in the 2D plane
- Velocity-based motion model
- Robot observes point landmarks
- Range-bearing sensor
- Known data association
- Known number of landmarks

Courtesy: Cyrill Stachniss

Initialization

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

$$\boldsymbol{\mu}_0 = \begin{pmatrix} 0 & 0 & 0 & \dots & 0 \end{pmatrix}^\top$$

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

Courtesy: Cyrill Stachniss

Extended Kalman Filter Algorithm

1: **Extended_Kalman_filter**$(\boldsymbol{\mu}_{t-1}, \Sigma_{t-1}, \mathbf{u}_t, \mathbf{z}_t)$**:**

2: $\bar{\boldsymbol{\mu}}_t = g(\mathbf{u}_t, \boldsymbol{\mu}_{t-1})$

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

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

5: $\boldsymbol{\mu}_t = \bar{\boldsymbol{\mu}}_t + K_t(\mathbf{z}_t - h(\bar{\boldsymbol{\mu}}_t))$

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

7: $return\ \boldsymbol{\mu}_t, \Sigma_t$

Courtesy: Cyrill Stachniss

Prediction Step (Motion)

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

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

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

Courtesy: Cyrill Stachniss

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

$$\underbrace{\begin{pmatrix} x' \\ y' \\ \theta' \\ \vdots \end{pmatrix}}_{\mathbf{y}_{t+1}} = \underbrace{\begin{pmatrix} x \\ y \\ \theta \\ \vdots \end{pmatrix} + \underbrace{\begin{pmatrix} 1 & 0 & 0 & 0\ldots0 \\ 0 & 1 & 0 & 0\ldots0 \\ 0 & 0 & 1 & \underbrace{0\ldots0}_{2n \text{ cols}} \end{pmatrix}^{\top}}_{F_{\mathbf{x}}^{\top}} \begin{pmatrix} -\frac{v_t}{\omega_t}\sin\theta + \frac{v_t}{\omega_t}\sin(\theta + \omega_t \Delta t) \\ \frac{v_t}{\omega_t}\cos\theta - \frac{v_t}{\omega_t}\cos(\theta + \omega_t \Delta t) \\ \omega_t \Delta t \end{pmatrix}}_{g(\mathbf{u}_t, \mathbf{y}_t)}$$

Courtesy: Cyrill Stachniss

Extended Kalman Filter Algorithm

1: $\textbf{Extended\_Kalman\_filter}(\boldsymbol{\mu}_{t-1}, \Sigma_{t-1}, \mathbf{u}_t, \mathbf{z}_t)$**:**

2: $\cancel{\bar{\boldsymbol{\mu}}_t = g(\mathbf{u}_t, \boldsymbol{\mu}_{t-1})}$ **DONE**

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

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

5: $\boldsymbol{\mu}_t = \bar{\boldsymbol{\mu}}_t + K_t(\mathbf{z}_t - h(\bar{\boldsymbol{\mu}}_t))$

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

7: $return\ \boldsymbol{\mu}_t, \Sigma_t$

Courtesy: Cyrill Stachniss

Update Covariance

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

Jacobian of the motion (3x3)

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

Identity ($2n$x$2n$)

Courtesy: Cyrill Stachniss

Jacobian of the Motion

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

Courtesy: Cyrill Stachniss

Jacobian of the Motion

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

Courtesy: Cyrill Stachniss

Jacobian of the Motion

$$
\begin{aligned}
G_t^{\mathbf{x}} &= \frac{\partial}{\partial(x,y,\theta)^\top}\left[\begin{pmatrix} x \\ y \\ \theta \end{pmatrix} + \begin{pmatrix} -\frac{v_t}{\omega_t}\sin\theta + \frac{v_t}{\omega_t}\sin(\theta+\omega_t\Delta t) \\ \frac{v_t}{\omega_t}\cos\theta - \frac{v_t}{\omega_t}\cos(\theta+\omega_t\Delta t) \\ \omega_t\,\Delta t \end{pmatrix}\right] \\
&= I + \frac{\partial}{\partial(x,y,\theta)^\top}\begin{pmatrix} -\frac{v_t}{\omega_t}\sin\theta + \frac{v_t}{\omega_t}\sin(\theta+\omega_t\Delta t) \\ \frac{v_t}{\omega_t}\cos\theta - \frac{v_t}{\omega_t}\cos(\theta+\omega_t\Delta t) \\ \omega_t\,\Delta t \end{pmatrix} \\
&= I + \begin{pmatrix} 0 & 0 & -\frac{v_t}{\omega_t}\cos\theta + \frac{v_t}{\omega_t}\cos(\theta+\omega_t\Delta t) \\ 0 & 0 & -\frac{v_t}{\omega_t}\sin\theta + \frac{v_t}{\omega_t}\sin(\theta+\omega_t\Delta t) \\ 0 & 0 & 0 \end{pmatrix}
\end{aligned}
$$

Courtesy: Cyrill Stachniss

Jacobian of the Motion

$$
\begin{aligned}
G_t^{\mathbf{x}} &= \frac{\partial}{\partial(x,y,\theta)^\top}\left[\begin{pmatrix} x \\ y \\ \theta \end{pmatrix} + \begin{pmatrix} -\frac{v_t}{\omega_t}\sin\theta + \frac{v_t}{\omega_t}\sin(\theta+\omega_t\Delta t) \\ \frac{v_t}{\omega_t}\cos\theta - \frac{v_t}{\omega_t}\cos(\theta+\omega_t\Delta t) \\ \omega_t\Delta t \end{pmatrix}\right] \\
&= I + \frac{\partial}{\partial(x,y,\theta)^\top}\begin{pmatrix} -\frac{v_t}{\omega_t}\sin\theta + \frac{v_t}{\omega_t}\sin(\theta+\omega_t\Delta t) \\ \frac{v_t}{\omega_t}\cos\theta - \frac{v_t}{\omega_t}\cos(\theta+\omega_t\Delta t) \\ \omega_t\Delta t \end{pmatrix} \\
&= I + \begin{pmatrix} 0 & 0 & -\frac{v_t}{\omega_t}\cos\theta + \frac{v_t}{\omega_t}\cos(\theta+\omega_t\Delta t) \\ 0 & 0 & -\frac{v_t}{\omega_t}\sin\theta + \frac{v_t}{\omega_t}\sin(\theta+\omega_t\Delta t) \\ 0 & 0 & 0 \end{pmatrix} \\
&= \begin{pmatrix} 1 & 0 & -\frac{v_t}{\omega_t}\cos\theta + \frac{v_t}{\omega_t}\cos(\theta+\omega_t\Delta t) \\ 0 & 1 & -\frac{v_t}{\omega_t}\sin\theta + \frac{v_t}{\omega_t}\sin(\theta+\omega_t\Delta t) \\ 0 & 0 & 1 \end{pmatrix}
\end{aligned}
$$

Courtesy: Cyrill Stachniss

This Leads to the Time Propagation

1: $\textbf{Extended\_Kalman\_filter}(\boldsymbol{\mu}_{t-1}, \Sigma_{t-1}, \mathbf{u}_t, \mathbf{z}_t)$:

2: ~~$\bar{\boldsymbol{\mu}}_t = g(\mathbf{u}_t, \boldsymbol{\mu}_{t-1})$~~ **Apply & DONE**

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

$$\begin{aligned}
\bar{\Sigma}_t &= G_t \Sigma_{t-1} G_t^\top + R_t \\
&= \begin{pmatrix} G_t^{\mathbf{x}} & 0 \\ 0 & I \end{pmatrix} \begin{pmatrix} \Sigma_{\mathbf{xx}} & \Sigma_{\mathbf{xm}} \\ \Sigma_{\mathbf{mx}} & \Sigma_{\mathbf{mm}} \end{pmatrix} \begin{pmatrix} {G_t^{\mathbf{x}}}^\top & 0 \\ 0 & I \end{pmatrix} + R_t \\
&= \begin{pmatrix} G_t^{\mathbf{x}} \Sigma_{\mathbf{xx}} {G_t^{\mathbf{x}}}^\top & G_t^{\mathbf{x}} \Sigma_{\mathbf{xm}} \\ (G_t^{\mathbf{x}} \Sigma_{\mathbf{xm}})^\top & \Sigma_{\mathbf{mm}} \end{pmatrix} + R_t
\end{aligned}$$

Courtesy: Cyrill Stachniss

EKF SLAM: Motion Prediction Step

$$\textbf{EKF\_SLAM}(\boldsymbol{\mu}_{t-1}, \Sigma_{t-1}, \mathbf{u}_t, \mathbf{z}_t, c_t, R_t^{\mathbf{x}}):$$

$$\textbf{EKF\_SLAM\_Prediction:}$$

$$1: \quad F_{\mathbf{x}} = \begin{pmatrix} 1 & 0 & 0 & 0 \cdots 0 \\ 0 & 1 & 0 & 0 \cdots 0 \\ 0 & 0 & 1 & 0 \cdots 0 \end{pmatrix}$$

$$2: \quad \bar{\boldsymbol{\mu}}_t = \boldsymbol{\mu}_{t-1} + F_{\mathbf{x}}^\top \begin{pmatrix} -\frac{v_t}{\omega_t} \sin \mu_{t-1,\theta} + \frac{v_t}{\omega_t} \sin(\mu_{t-1,\theta} + \omega_t \Delta t) \\ \frac{v_t}{\omega_t} \cos \mu_{t-1,\theta} - \frac{v_t}{\omega_t} \cos(\mu_{t-1,\theta} + \omega_t \Delta t) \\ \omega_t \Delta t \end{pmatrix}$$

$$3: \quad G_t = I + F_{\mathbf{x}}^\top \begin{pmatrix} 0 & 0 & -\frac{v_t}{\omega_t} \cos \mu_{t-1,\theta} + \frac{v_t}{\omega_t} \cos(\mu_{t-1,\theta} + \omega_t \Delta t) \\ 0 & 0 & -\frac{v_t}{\omega_t} \sin \mu_{t-1,\theta} + \frac{v_t}{\omega_t} \sin(\mu_{t-1,\theta} + \omega_t \Delta t) \\ 0 & 0 & 0 \end{pmatrix} F_{\mathbf{x}}$$

$$4: \quad \bar{\Sigma}_t = G_t \, \Sigma_{t-1} \, G_t^\top + \underbrace{F_{\mathbf{x}}^\top \, R_t^{\mathbf{x}} \, F_{\mathbf{x}}}_{R_t}$$

Courtesy: Cyrill Stachniss

Extended Kalman Filter Algorithm

$$\textbf{Extended\_Kalman\_filter}(\boldsymbol{\mu}_{t-1}, \Sigma_{t-1}, \mathbf{u}_t, \mathbf{z}_t)\textbf{:}$$

$$\cancel{\bar{\boldsymbol{\mu}}_t = g(\mathbf{u}_t, \boldsymbol{\mu}_{t-1})}$$ **DONE**

$$\cancel{\bar{\Sigma}_t = G_t\, \Sigma_{t-1}\, G_t^\top + R_t}$$ **DONE**

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

$$\boldsymbol{\mu}_t = \bar{\boldsymbol{\mu}}_t + K_t(\mathbf{z}_t - h(\bar{\boldsymbol{\mu}}_t))$$

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

$$\text{return } \boldsymbol{\mu}_t, \Sigma_t$$

Courtesy: Cyrill Stachniss

Extended Kalman Filter Algorithm

1: $\textbf{Extended\_Kalman\_filter}(\boldsymbol{\mu}_{t-1}, \Sigma_{t-1}, \mathbf{u}_t, \mathbf{z}_t)$:

2: $\cancel{\bar{\boldsymbol{\mu}}_t = g(\mathbf{u}_t, \boldsymbol{\mu}_{t-1})}$ DONE

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

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

5: $\boldsymbol{\mu}_t = \bar{\boldsymbol{\mu}}_t + K_t(\mathbf{z}_t - h(\bar{\boldsymbol{\mu}}_t))$

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

7: $return\ \boldsymbol{\mu}_t, \Sigma_t$

Courtesy: Cyrill Stachniss

EKF SLAM: Correction Step

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

Courtesy: Cyrill Stachniss

Range-Bearing Observation

- Range-Bearing observation $\mathbf{z}_t^i = (r_t^i, \phi_t^i)^\top$

- 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

Can initialize from a single measurement b/c the observation function is bijective for range/bearing. If using either range or bearing only, would need to accumulate observations over a time window and triangulate to initialize.

Courtesy: Cyrill Stachniss

Expected Observation

- Compute expected observation according to the current estimate

$$
\begin{aligned}
\delta &= \begin{pmatrix} \delta_x \\ \delta_y \end{pmatrix} = \begin{pmatrix} \bar{\mu}_{j,x} - \bar{\mu}_{t,x} \\ \bar{\mu}_{j,y} - \bar{\mu}_{t,y} \end{pmatrix} \\
q &= \delta^\top \delta \\
\hat{\mathbf{z}}_t^i &= \begin{pmatrix} \sqrt{q} \\ \operatorname{atan2}(\delta_y, \delta_x) - \bar{\mu}_{t,\theta} \end{pmatrix} \\
&= h(\bar{\boldsymbol{\mu}}_t)
\end{aligned}
$$

Courtesy: Cyrill Stachniss

Jacobian for the Observation

- Based on

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

- Compute the Jacobian

$$^{\text{low}}H_t^i = \frac{\partial h(\bar{\boldsymbol{\mu}}_t)}{\partial \bar{\boldsymbol{\mu}}_t}$$

low-dim space $(x, y, \theta, m_{j,x}, m_{j,y})$

Courtesy: Cyrill Stachniss

Jacobian for the Observation

- Based on

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

- Compute the Jacobian

$$\begin{aligned}
{}^{\text{low}}H_t^i &= \frac{\partial h(\bar{\boldsymbol{\mu}}_t)}{\partial \bar{\boldsymbol{\mu}}_t} \\
&= \begin{pmatrix} \frac{\partial \sqrt{q}}{\partial x} & \frac{\partial \sqrt{q}}{\partial y} & \dots \\ \frac{\partial \operatorname{atan2}(\dots)}{\partial x} & \frac{\partial \operatorname{atan2}(\dots)}{\partial y} & \dots \end{pmatrix}
\end{aligned}$$

Courtesy: Cyrill Stachniss

The First Component

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

- We obtain (by applying the chain rule)

$$\begin{aligned}\frac{\partial\sqrt{q}}{\partial x} &= \frac{1}{2}\frac{1}{\sqrt{q}}2\,\delta_x\,(-1)\\ &= \frac{1}{q}(-\sqrt{q}\delta_x)\end{aligned}$$

Courtesy: Cyrill Stachniss

Jacobian for the Observation

- Based on

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

- Compute the Jacobian

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

Courtesy: Cyrill Stachniss

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

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

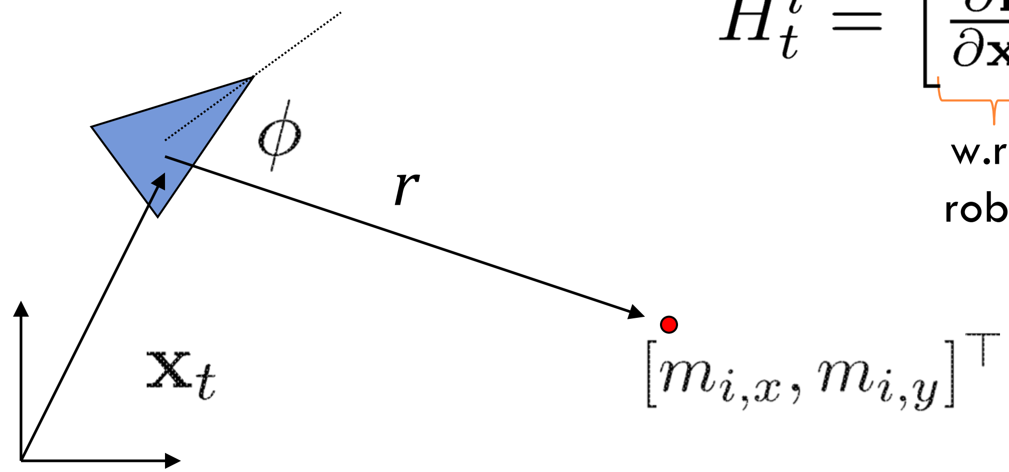
$$ F_{\mathbf{x},\mathbf{j}} = \begin{pmatrix} 1 & 0 & 0 & 0\cdots0 & 0 & 0 & 0\cdots0 \\ 0 & 1 & 0 & 0\cdots0 & 0 & 0 & 0\cdots0 \\ 0 & 0 & 1 & 0\cdots0 & 0 & 0 & 0\cdots0 \\ 0 & 0 & 0 & 0\cdots0 & 1 & 0 & 0\cdots0 \\ 0 & 0 & 0 & \underbrace{0\cdots0}_{2j-2} & 0 & 1 & \underbrace{0\cdots0}_{2n-2j} \end{pmatrix} $$

Courtesy: Cyrill Stachniss

In other words, what that projection matrix $F_{\mathbf{x},\mathbf{j}}$ does …

$$
\begin{aligned}
\mathbf{z}_t &\triangleq \begin{bmatrix} r \\ \phi \end{bmatrix} + \mathbf{w}_t \\
&= h(\mathbf{y}_t) + \mathbf{w}_t \\
&= \begin{bmatrix} \sqrt{(m_{i,x} - x_R(k))^2 + (m_{i,y} - y_R(k))^2} \\ \mathrm{atan2}(\frac{m_{i,y} - y_R(k)}{m_{i,x} - x_R(k)}) - \theta_R \end{bmatrix} + \mathbf{w}_t
\end{aligned}
$$

$$
H_t^i = \begin{bmatrix} \underbrace{\frac{\partial \mathbf{h}}{\partial \mathbf{x}_t}}_{\text{w.r.t. robot}} & \underbrace{\ldots 0 \ldots}_{\text{w.r.t. other landmarks}} & \underbrace{\frac{\partial \mathbf{h}}{\partial \mathbf{m}_i}}_{\text{w.r.t. } \boldsymbol{m}_i} & \underbrace{\ldots 0 \ldots}_{\text{w.r.t. other landmarks}} \end{bmatrix}
$$



Next Steps as Specified…

1: $\textbf{Extended\_Kalman\_filter}(\boldsymbol{\mu}_{t-1}, \Sigma_{t-1}, \mathbf{u}_t, \mathbf{z}_t)$:

2: $\cancel{\bar{\boldsymbol{\mu}}_t = g(\mathbf{u}_t, \boldsymbol{\mu}_{t-1})}$ DONE

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

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

5: $\boldsymbol{\mu}_t = \bar{\boldsymbol{\mu}}_t + K_t(\mathbf{z}_t - h(\bar{\boldsymbol{\mu}}_t))$

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

7: $\quad return\ \boldsymbol{\mu}_t, \Sigma_t$

Courtesy: Cyrill Stachniss

Extended Kalman Filter Algorithm

$$\textbf{Extended\_Kalman\_filter}(\boldsymbol{\mu}_{t-1}, \Sigma_{t-1}, \mathbf{u}_t, \mathbf{z}_t):$$

$\cancel{\bar{\boldsymbol{\mu}}_t = g(\mathbf{u}_t, \boldsymbol{\mu}_{t-1})}$ **DONE**

$\cancel{\bar{\Sigma}_t = G_t\, \Sigma_{t-1}\, G_t^\top + R_t}$ **DONE**

$\cancel{K_t = \bar{\Sigma}_t\, H_t^\top (H_t\, \bar{\Sigma}_t\, H_t^\top + Q_t)^{-1}}$ **Apply & DONE**

$\cancel{\boldsymbol{\mu}_t = \bar{\boldsymbol{\mu}}_t + K_t(\mathbf{z}_t - h(\bar{\boldsymbol{\mu}}_t))}$ **Apply & DONE**

$\cancel{\Sigma_t = (I - K_t\, H_t)\, \bar{\Sigma}_t}$ **Apply & DONE**

$\text{return } \boldsymbol{\mu}_t, \Sigma_t$

Courtesy: Cyrill Stachniss

EKF SLAM – Correction (1/2)

EKF_SLAM_Correction

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

for all observed features $\mathbf{z}_t^i = (r_t^i, \phi_t^i)^\top$ *do*

$\quad j = c_t^i$

$\quad$ *if landmark* j *never seen before*

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

$\quad$ *endif*

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

$$q = \delta^\top \delta$$

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

EKF SLAM – Correction (2/2)

$$F_{\mathbf{x},\mathbf{j}} = \begin{pmatrix} 1 & 0 & 0 & 0\cdots0 & 0 & 0 & 0\cdots0 \\ 0 & 1 & 0 & 0\cdots0 & 0 & 0 & 0\cdots0 \\ 0 & 0 & 1 & 0\cdots0 & 0 & 0 & 0\cdots0 \\ 0 & 0 & 0 & 0\cdots0 & 1 & 0 & 0\cdots0 \\ 0 & 0 & 0 & \underbrace{0\cdots0}_{2j-2} & 0 & 1 & \underbrace{0\cdots0}_{2n-2j} \end{pmatrix}$$

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

$$K_t^i = \bar{\Sigma}_t \, H_t^{i\top} (H_t^i \, \bar{\Sigma}_t \, H_t^{i\top} + Q_t)^{-1}$$

$$\bar{\boldsymbol{\mu}}_t = \bar{\boldsymbol{\mu}}_t + K_t^i (\mathbf{z}_t^i - \hat{\mathbf{z}}_t^i)$$

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

endfor

$$\boldsymbol{\mu}_t = \bar{\boldsymbol{\mu}}_t$$

$$\Sigma_t = \bar{\Sigma}_t$$

return $\boldsymbol{\mu}_t, \Sigma_t$

Courtesy: Cyrill Stachniss

EKF-SLAM Stacked (Batch) Update

- Update

$$\bar{\boldsymbol{\mu}}_{z_t} = h_s(\bar{\boldsymbol{\mu}}_t, \mathbf{c}_t)$$
$$S_t = H_t \bar{\Sigma}_t H_t^\top + Q_t$$
$$K_t = \bar{\Sigma}_t H_t^\top S_t^{-1}$$
$$\boldsymbol{\mu}_t = \bar{\boldsymbol{\mu}}_t + K_t(\mathbf{z}_t - \bar{\boldsymbol{\mu}}_{z_t})$$
$$\Sigma_t = (I - K_t H_t)\bar{\Sigma}_t$$

$$z_t = \begin{bmatrix} r_1^t \\ \theta_1^t \\ \vdots \\ r_n^t \\ \theta_n^t \end{bmatrix}$$

What would H_t look like?

K_t is a (3+2n) £ (2k) gain matrix composed of a weighted linear combination of the columns of $\sum_t$ associated with R and m_i indices

Hence, in general, because K_t is full, it non-trivially changes <u>all</u> of the elements in $\sum_t$ during the covariance update

EKF-SLAM Sequential Update

- for i=1:k

$$\bar{\boldsymbol{\mu}}_{z_t^i} = h(\bar{\boldsymbol{\mu}}_r, \mathbf{m}_i)$$

$$S_t^i = H_t^i \bar{\Sigma}_t {H_t^i}^\top + Q_t^i$$

$$K_t^i = \bar{\Sigma}_t {H_t^i}^\top {S_t^i}^{-1}$$

$$\bar{\boldsymbol{\mu}}_t = \bar{\boldsymbol{\mu}}_t + K_t^i(\mathbf{z}_t^i - \bar{\boldsymbol{\mu}}_{z_t^i})$$

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

Think of this loop as a zero motion prediction + update step

endfor

$$\boldsymbol{\mu}_t = \bar{\boldsymbol{\mu}}_t$$

$$\Sigma_t = \bar{\Sigma}_t$$

Batch vs Sequential Updates

- When can we do this?
  - When sensor measurements are independent, i.e.

$$Q_t = \text{diag}(Q_t^1, \ldots, Q_t^k)$$

- Why?
  - Because of conditional independence

$$p(\mathbf{z}_t \mid \mathbf{x}_t, \mathbf{z}_{1:t-1}, \mathbf{u}_{1:t}) = \prod_{i=1}^{k} p(\mathbf{z}_t^i \mid \mathbf{x}_t)$$

- Are Batch vs. Sequential Updates Equivalent?
  - Yes – when observation models are linear (KF)
  - Not quite – when observation models are nonlinear (EKF)
    - Why? because our linearization point for each H_t^i changes as we perform the updates sequentially

Implementation Notes

- Measurement update in a single step requires only one full belief update
- Always normalize the angular components
- You do not need to create the F projection matrices explicitly (e.g., in Matlab), they are shown for illustration of dimensionality.
  - Prediction and update expressions can be more efficiently implemented accounting for the sparsity that exists in the Jacobians.

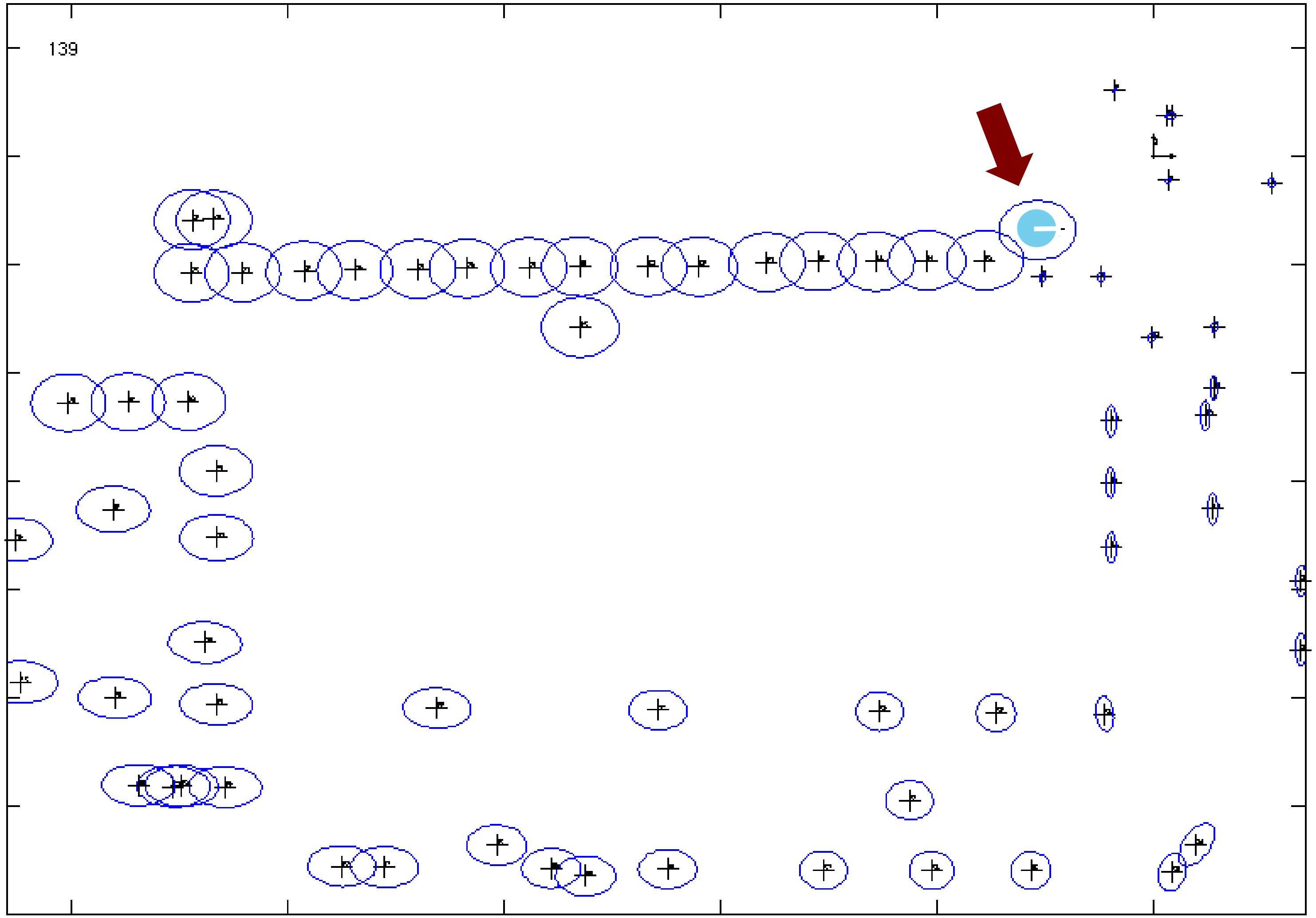
Courtesy: Cyrill Stachniss

Loop-Closing

- Loop-closing means recognizing an already mapped area
- Data association under
  - high ambiguity
  - possible environment symmetries
- Uncertainties **collapse** after a loop-closure (whether the closure was correct or not)

Courtesy: Cyrill Stachniss

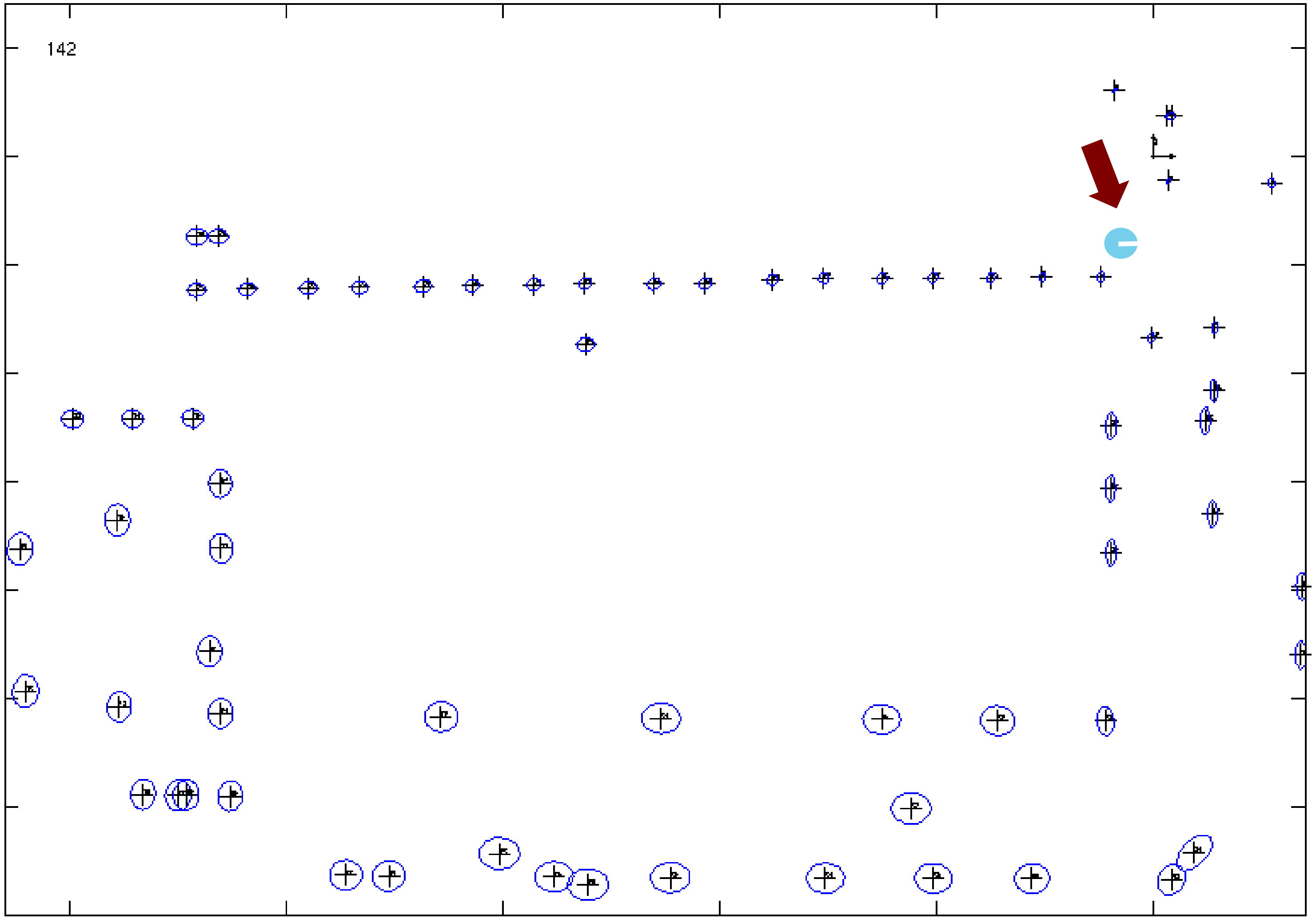
Before the Loop-Closure



Courtesy: K. Arras

Courtesy: Cyrill Stachniss

After the Loop-Closure



Courtesy: K. Arras

Courtesy: Cyrill Stachniss

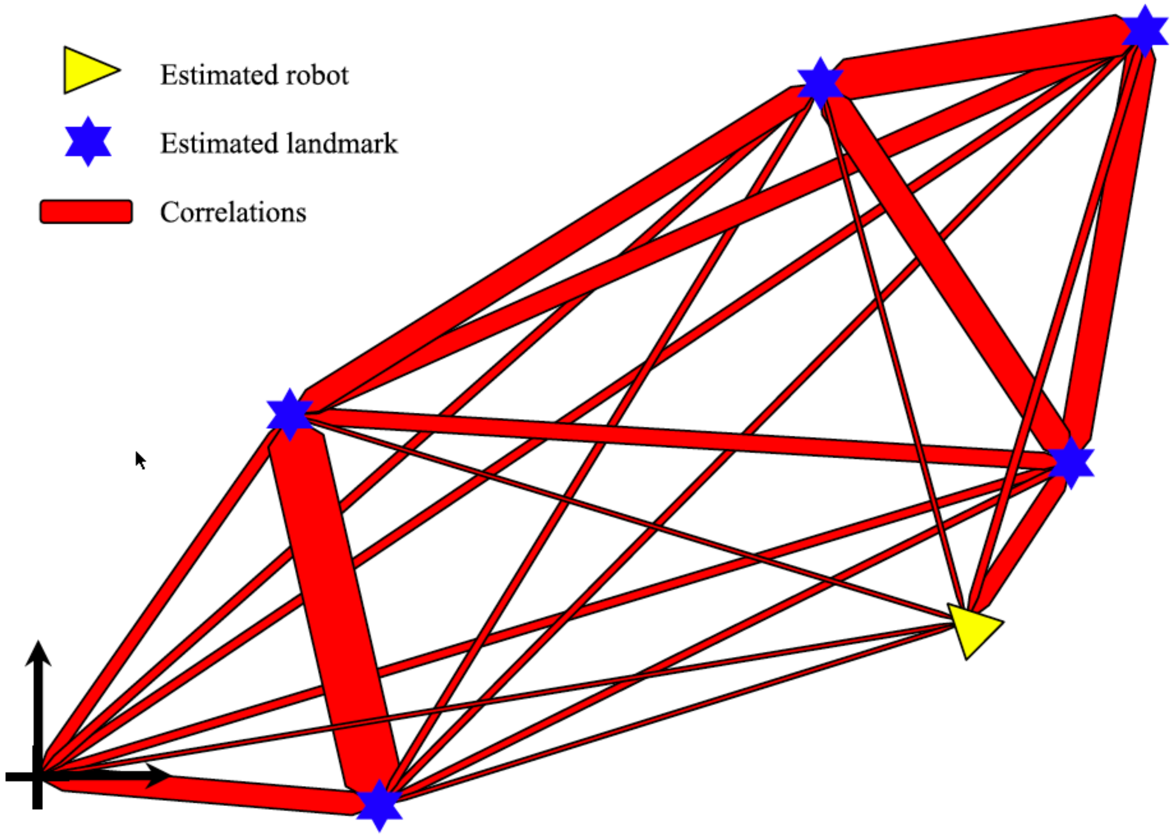
Loop-Closures in SLAM

- 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**

Courtesy: Cyrill Stachniss

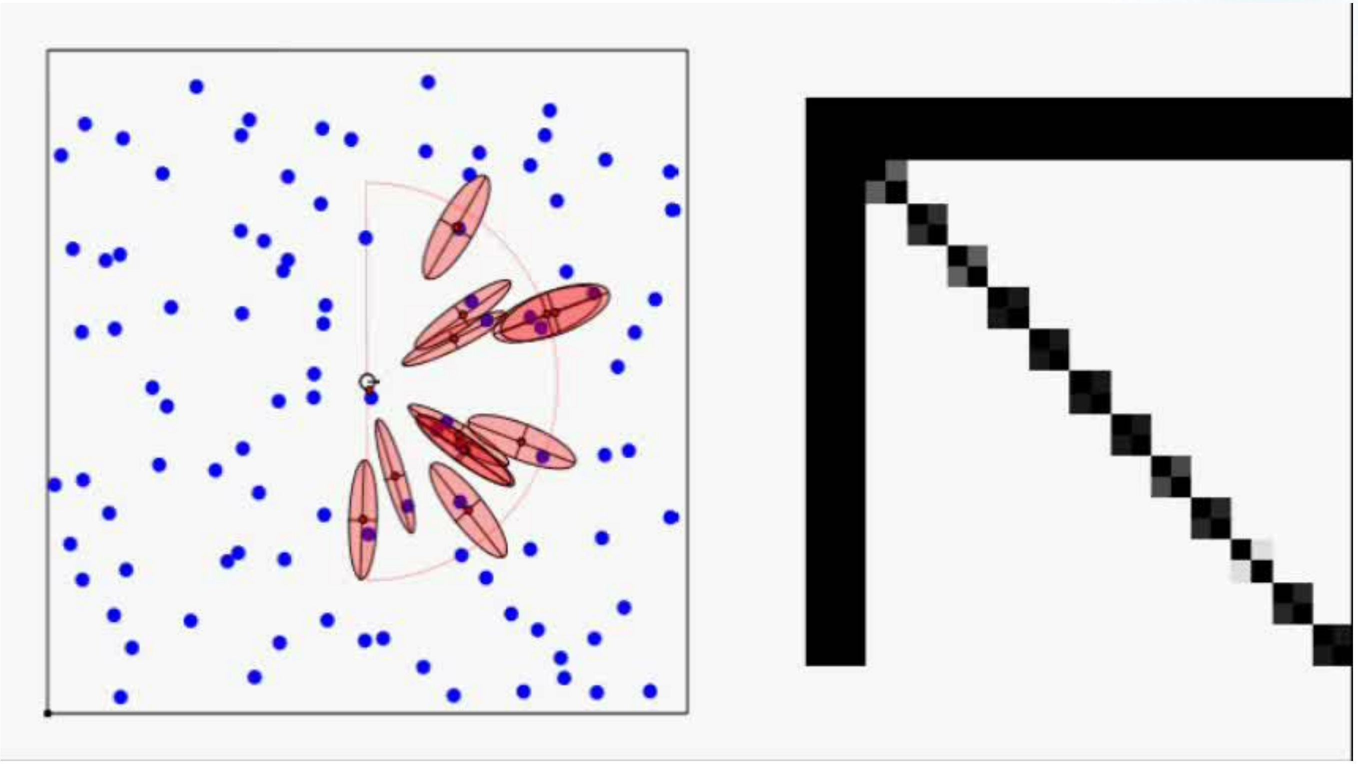
EKF SLAM Correlations

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



Courtesy: Dissanayake

EKF SLAM Correlations

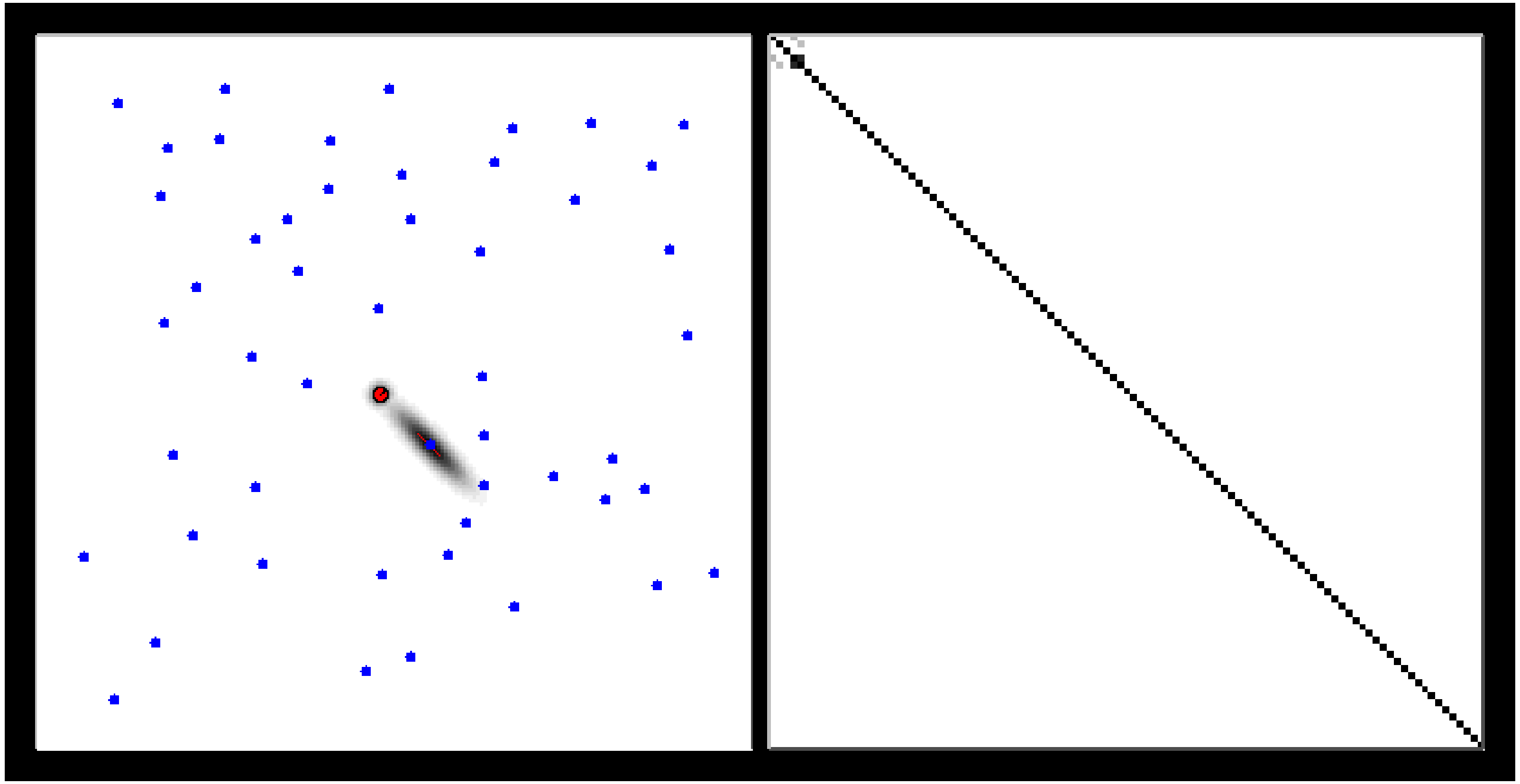


Blue path = true path **Red path** = estimated path **Black path** = odometry

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

Courtesy: M. Montemerlo

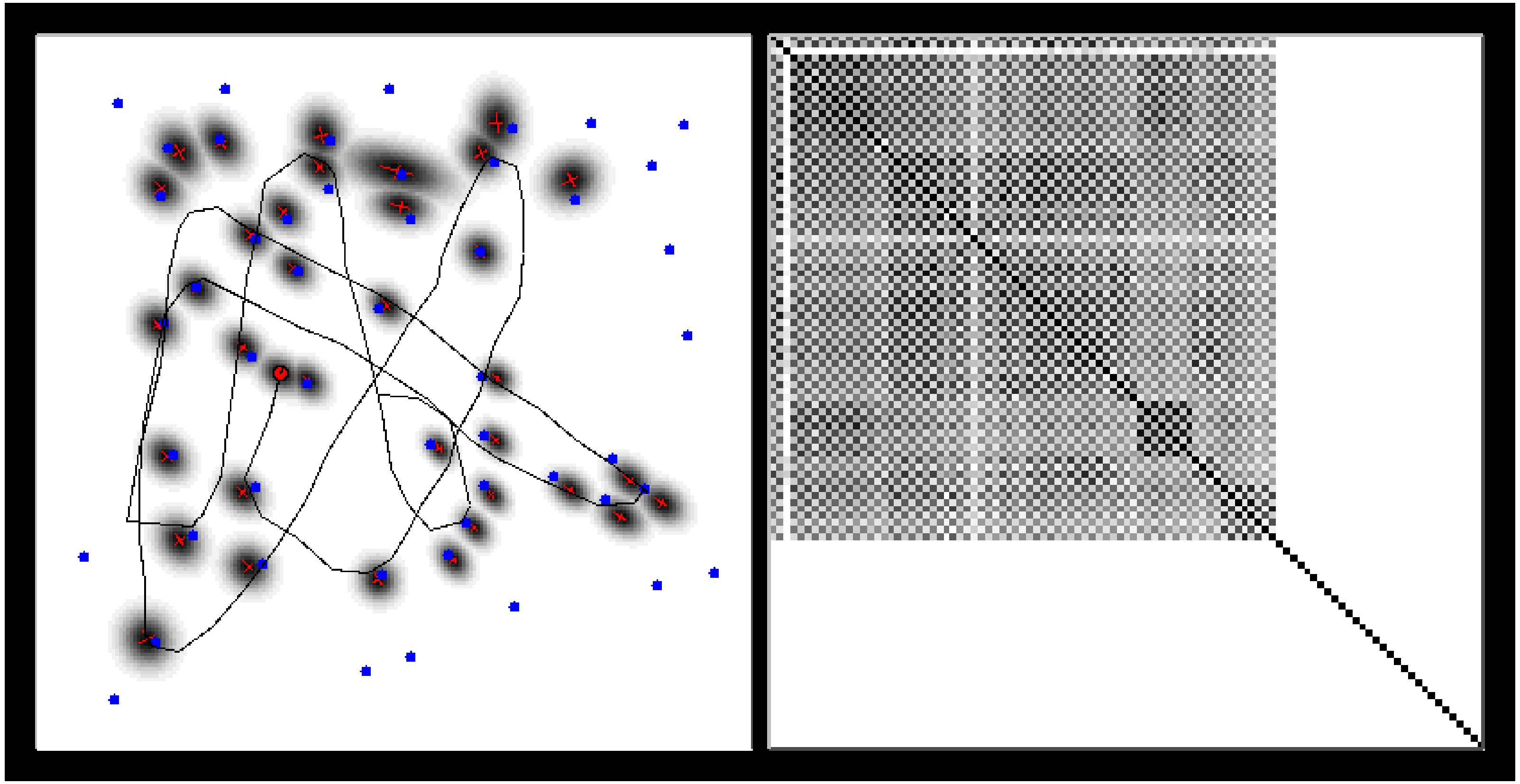
EKF SLAM Correlations



Map Correlation matrix

Courtesy: M. Montemerlo

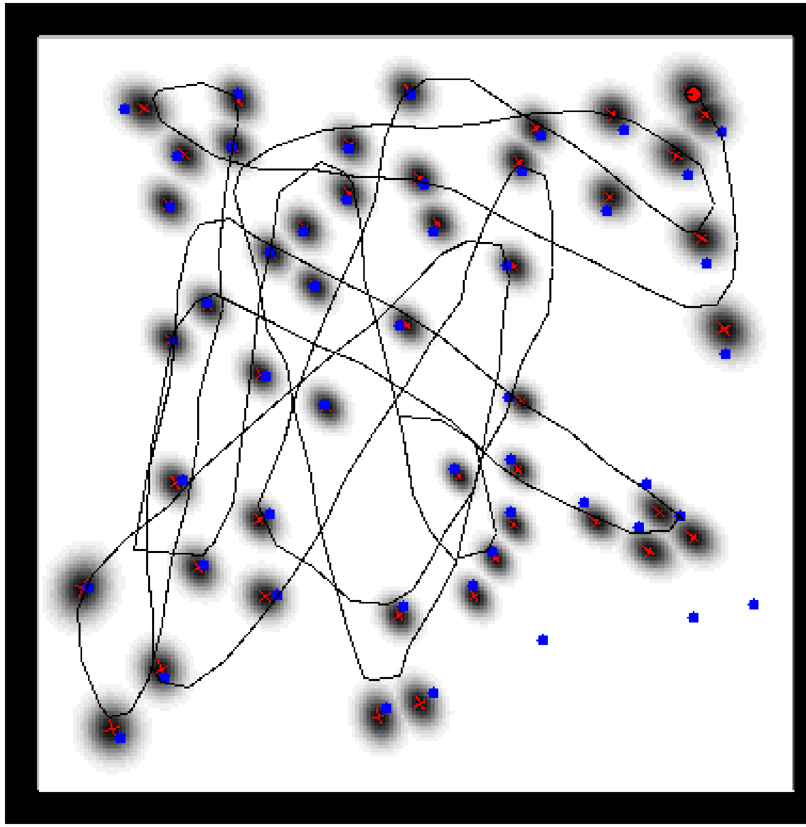
EKF SLAM Correlations



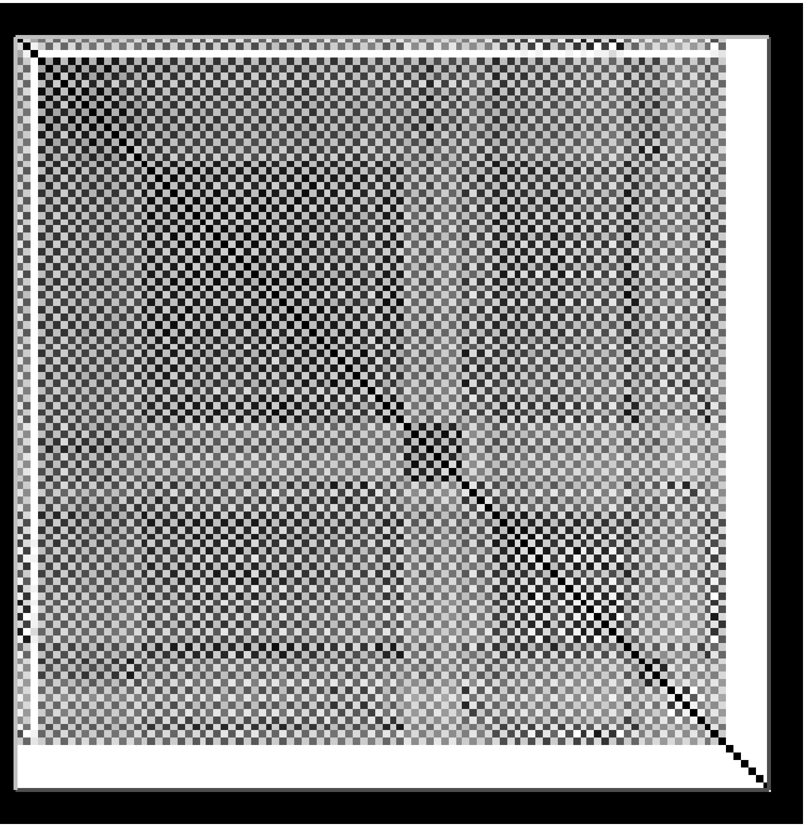
Map Correlation matrix

Courtesy: M. Montemerlo

EKF SLAM Correlations



Map



Correlation matrix

Courtesy: M. Montemerlo

EKF SLAM Correlations

- Covariance Matrix in the Linear Gaussian SLAM solution
  - Pros :
    - Enables us to transfer information from one part of the environment to the other parts
    - Prevents overconfidence and divergence
  - Cons :
    - Increases computational complexity

Cross correlations

Maintain ⟷ **Ignore**

Slow
Consistent

Fast
Inconsistent

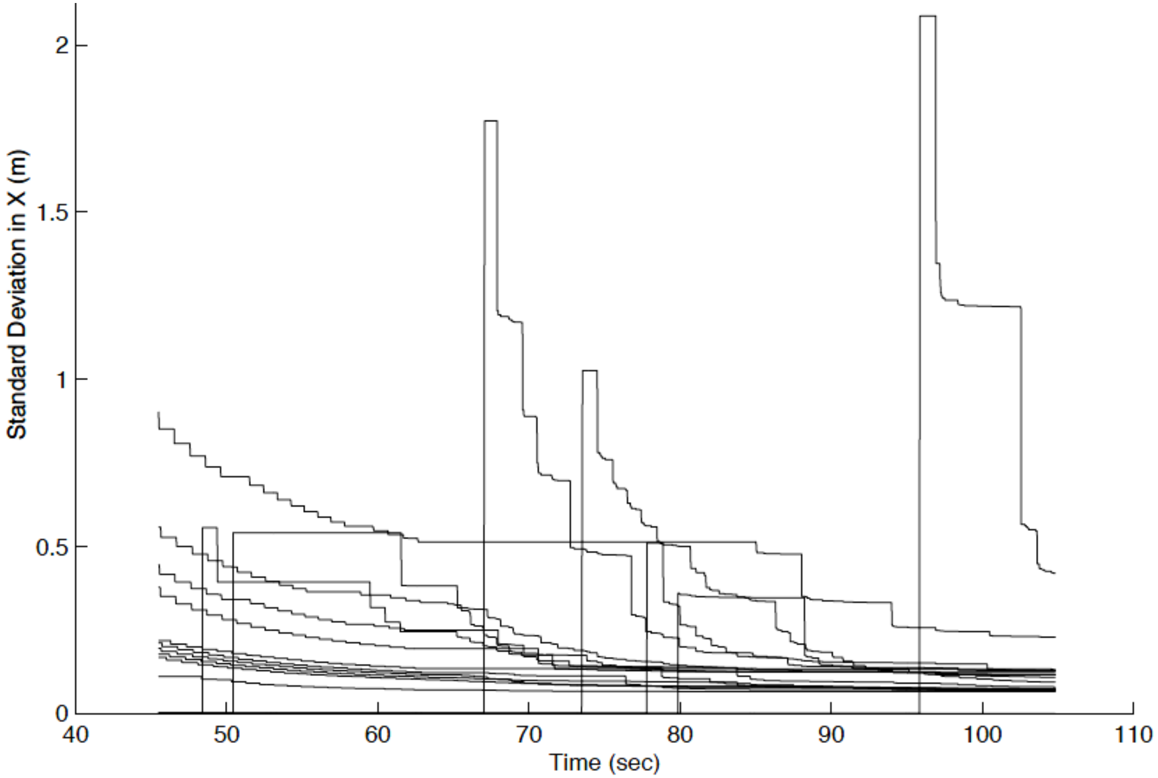
EKF SLAM Correlations

- The correlation between the robot's pose and the landmarks **cannot** be ignored
- Assuming independence generates too optimistic estimates of the uncertainty

Courtesy: J.M. Castellanos

EKF SLAM Uncertainties

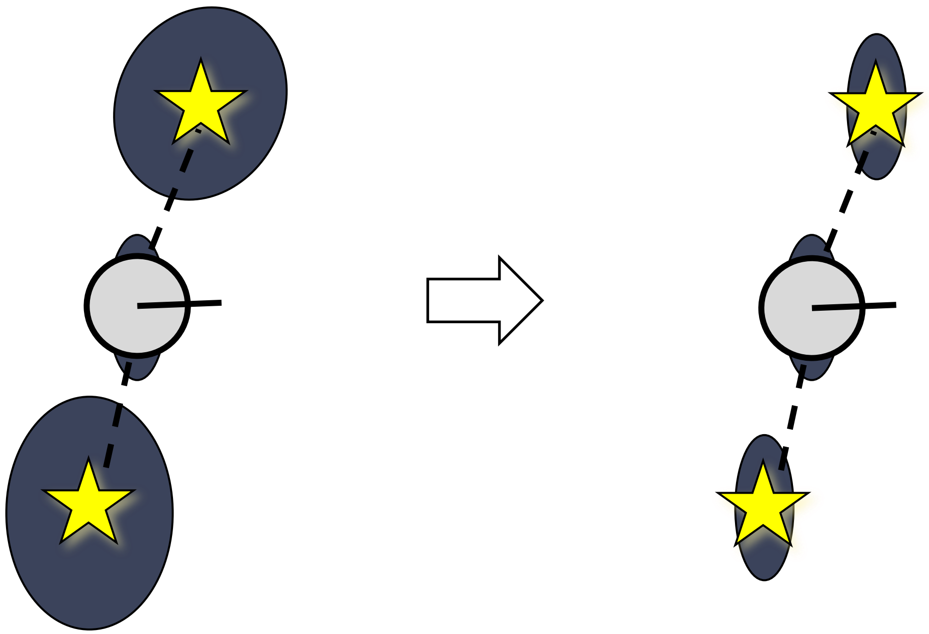
- The **determinant** of any sub-matrix of the map covariance matrix **decreases monotonically**
- New landmarks are initialized with **maximum uncertainty**



Courtesy: Dissanayake

EKF SLAM in the Limit

- In the limit, the covariance associated with any single landmark location estimate is determined only by the initial covariance in the vehicle location estimate.



Courtesy: Dissanayake

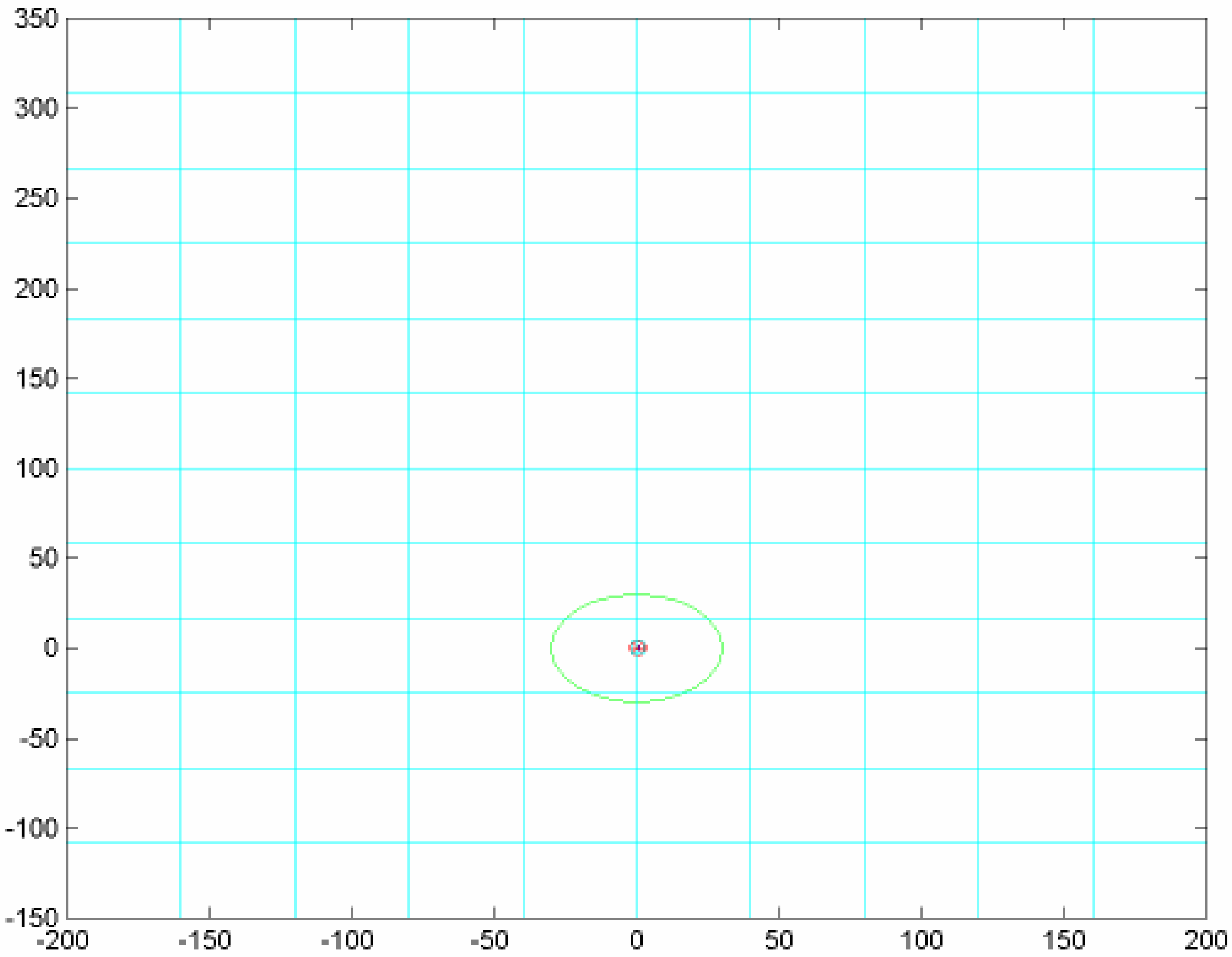
Example: Victoria Park Dataset

Courtesy: E. Nebot

Victoria Park: Data Acquisition

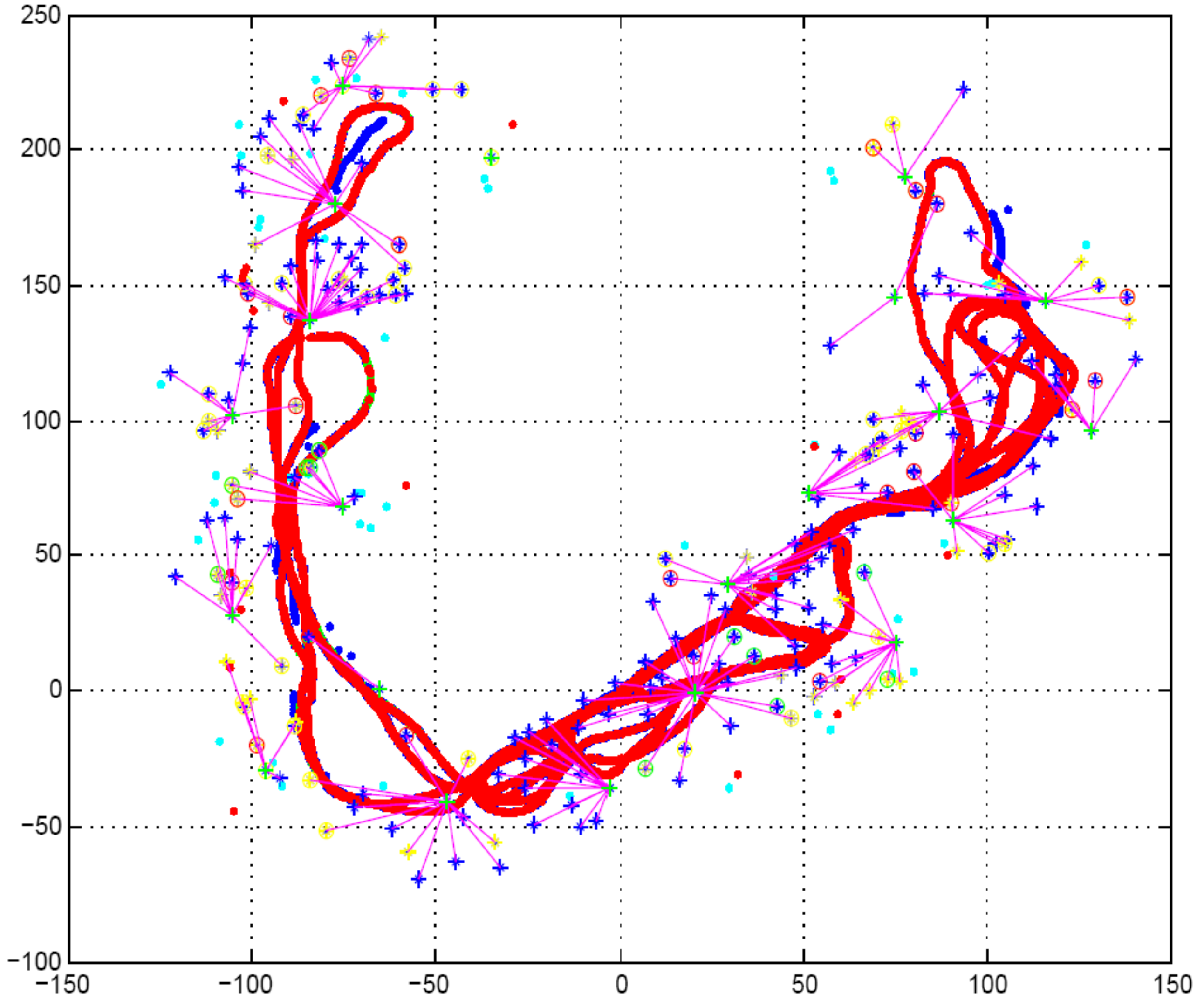
Courtesy: E. Nebot

Victoria Park: EKF Estimate



Courtesy: E. Nebot

Victoria Park: EKF Estimate

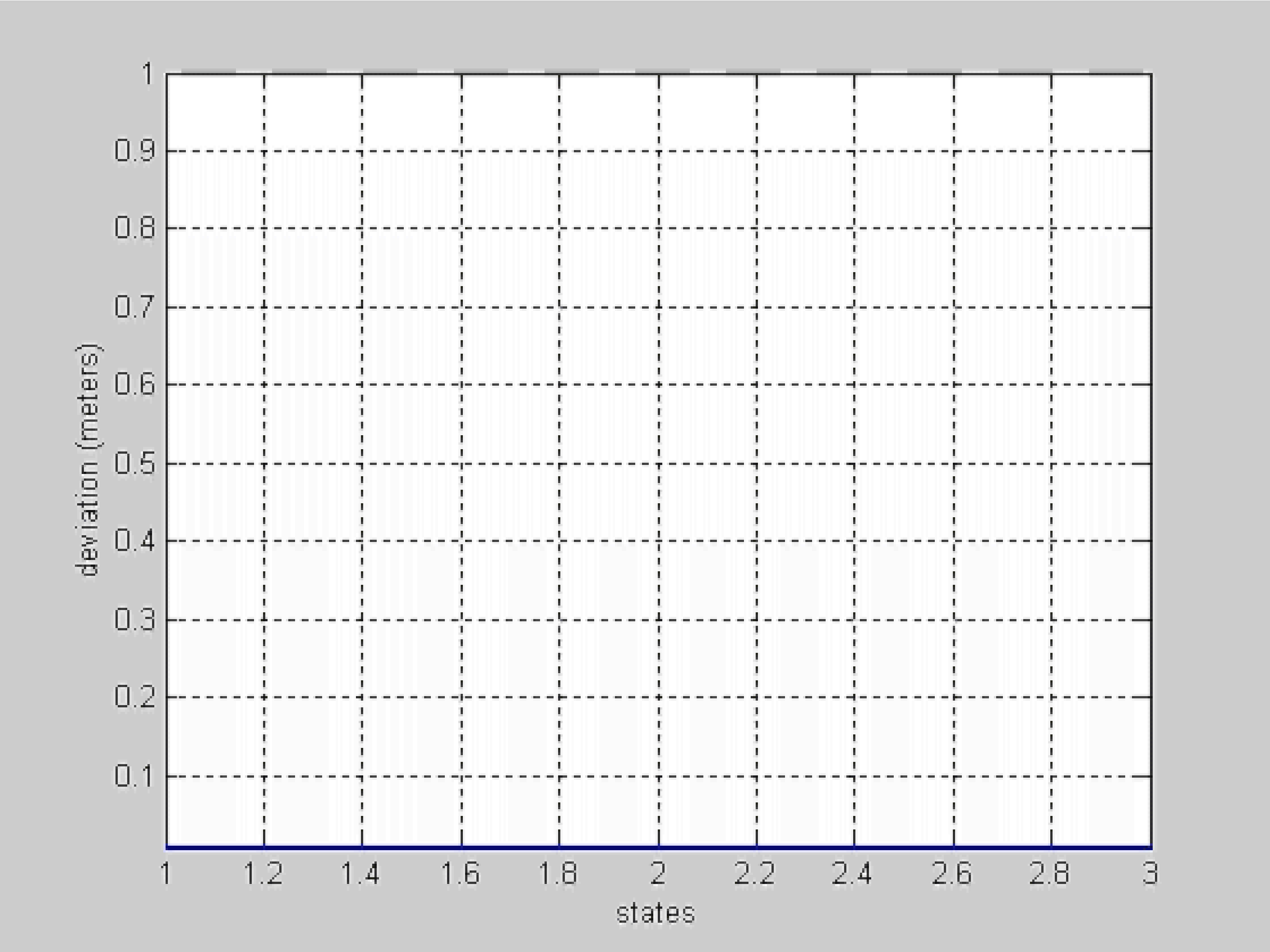


Courtesy: E. Nebot

Victoria Park: Landmarks

Courtesy: E. Nebot

Victoria Park: Landmark Covariance



Courtesy: E. Nebot

Andrew Davison: MonoSLAM

Courtesy: E. Nebot

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!

Courtesy: Cyrill Stachniss

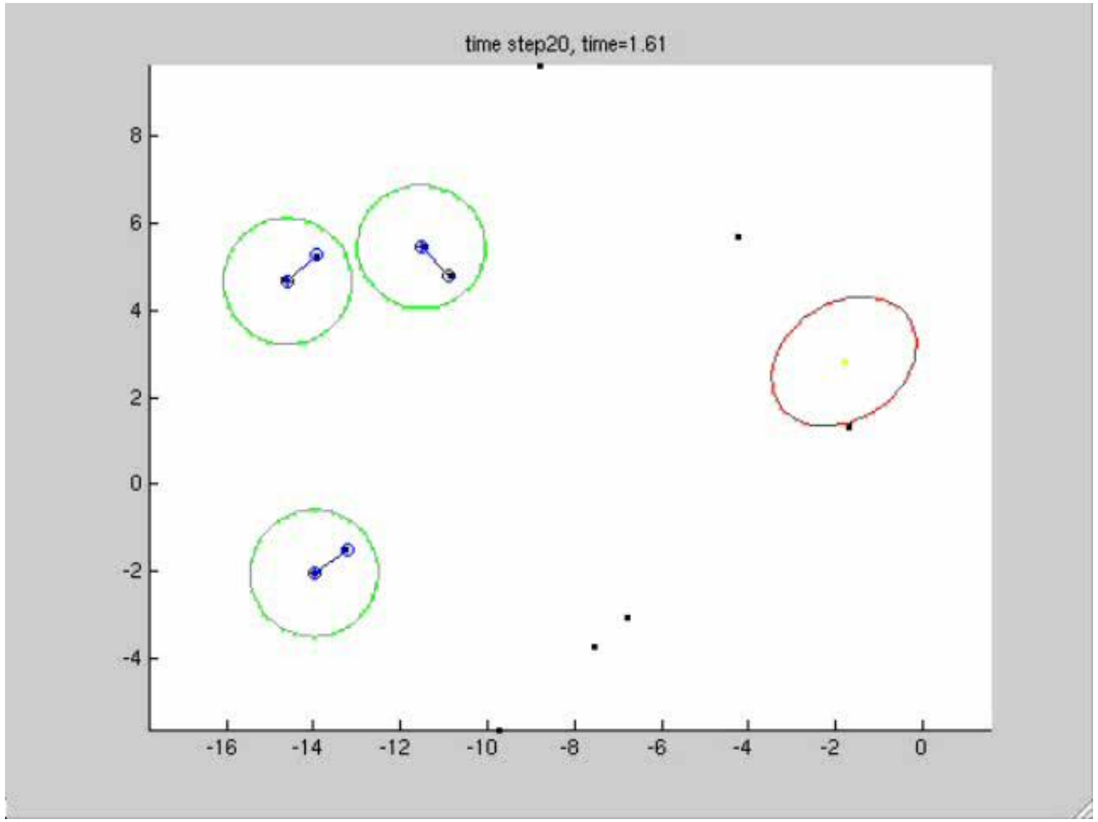
EKF SLAM Summary

- 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.

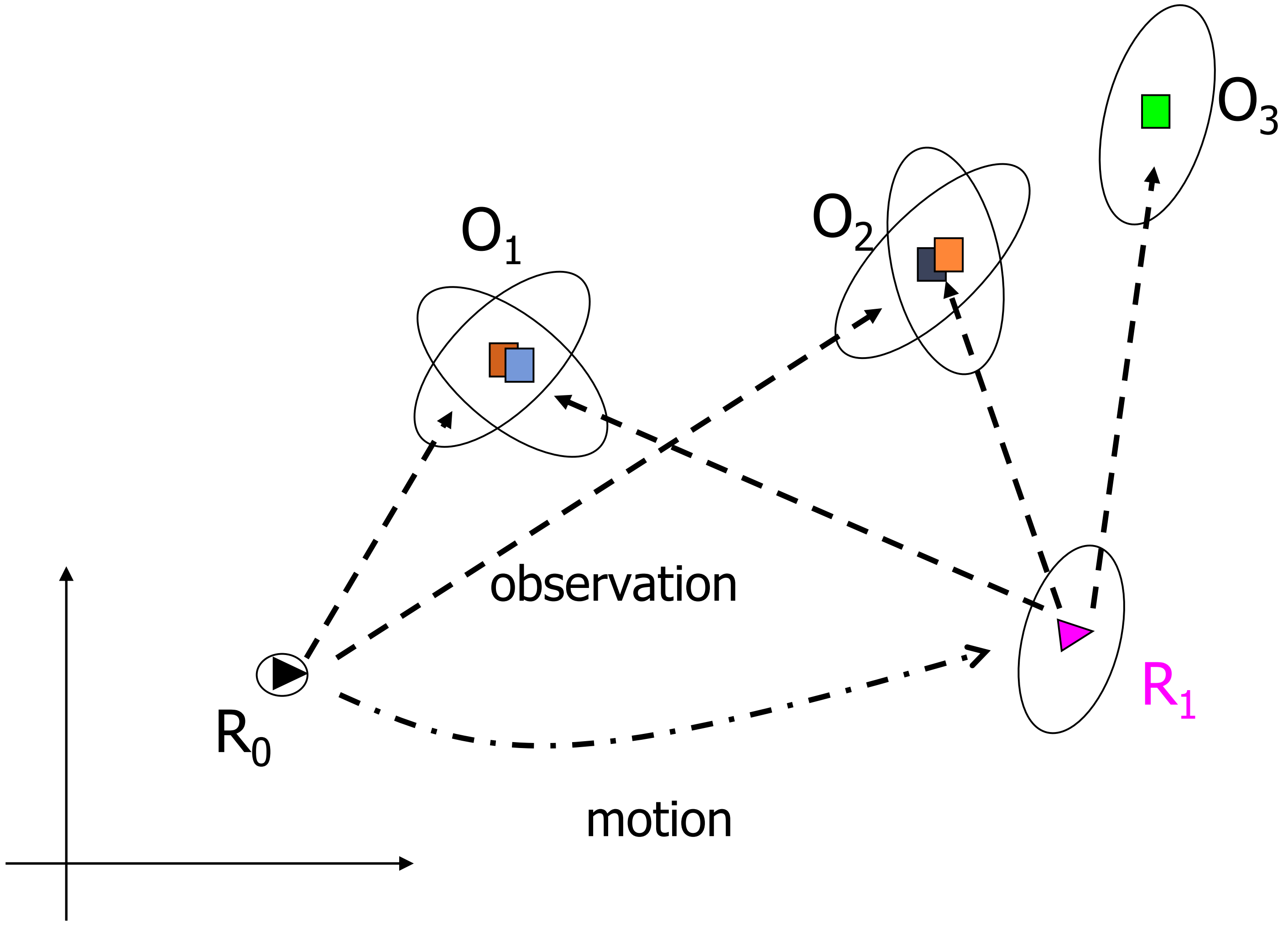
SLAM Using Landmarks

1. Move
2. Sense
3. Associate measurements with known features
4. Update state estimates for robot and previously mapped features
5. Find new features from unassociated measurements
6. Initialize new features
7. Repeat

MIT Indoor Track



Incremental Map Making



Courtesy J. Leonard

EKF-SLAM State Augmentation

- Augmentation model

$$\mathbf{x}_t = \begin{bmatrix} \mathbf{x}_r \\ \mathbf{x}_L \\ \mathbf{x}_{\ell_{n+1}} \end{bmatrix} = \begin{bmatrix} \mathbf{x}_r \\ \mathbf{x}_L \\ \mathbf{g}(\mathbf{x}_r, \hat{\mathbf{z}}_t^i) \end{bmatrix} = \begin{bmatrix} \mathbf{x}_r \\ \mathbf{x}_L \\ \mathbf{g}(\mathbf{x}_r, \mathbf{z}_t^i - \delta_t^i) \end{bmatrix}$$

True measurement

Actual measurement corrupted by noise where $\mathbf{z}_t^i = \hat{\mathbf{z}}_t^i + \delta_t$

$$\approx \begin{bmatrix} \mathbf{x}_r \\ \mathbf{x}_L \\ \mathbf{g}(\mu_r, \mathbf{z}_t^i) + G_r(\mathbf{x}_r - \mu_r) + G_\delta(\delta_t - 0) \end{bmatrix}$$

State Augmentation of Belief

- Before

$$\mu_t = \begin{bmatrix} \mu_r \\ \mu_L \end{bmatrix} \qquad \Sigma_t = \begin{bmatrix} \Sigma_{rr} & \Sigma_{rL} \\ \Sigma_{Lr} & \Sigma_{LL} \end{bmatrix}$$

- After

$$\mu_{t_{\text{aug}}} = \begin{bmatrix} \mu_r \\ \mu_L \\ \mu_{\ell_i} \end{bmatrix} = \begin{bmatrix} \mu_r \\ \mu_L \\ \mathbf{g}(\mu_r, \mathbf{z}_t^i) \end{bmatrix}$$

$$\Sigma_{t_{\text{aug}}} = \begin{bmatrix} \Sigma_{rr} & \Sigma_{rL} & \Sigma_{r\ell_i} \\ \Sigma_{Lr} & \Sigma_{LL} & \Sigma_{L\ell_i} \\ \Sigma_{\ell_i r} & \Sigma_{\ell_i L} & \Sigma_{\ell_i \ell_i} \end{bmatrix} \quad \begin{aligned} \Sigma_{\ell_i r} &= \Sigma_{r\ell_i}^\top = G_r \Sigma_{rr} \\ \Sigma_{\ell_i L} &= \Sigma_{L\ell_i}^\top = G_r \Sigma_{rL} \\ \Sigma_{\ell_i \ell_i} &= G_r \Sigma_{rr} G_r^\top + G_\delta Q_t^i G_\delta^\top \end{aligned}$$

EKF-SLAM Algorithm

- Pseudocode

```
while 1
  Do Prediction
  if no z_t are available
     return
  else
    Determine correspondence variables c_t
      if any c_t are new
          Do State Augmentation
      endif
      Do Update (either batch or sequential)
  endwhile
```

Data Association

- We've assumed known data association
  - "Observation i is of landmark j"
- What happens if we don't have data association?
- What are the consequences if we get data association wrong?
  - Wormholes

Importance of Data Association

- Measurement z is used to improve estimate of x:

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

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

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

- If the association of measurement $\boldsymbol{E_i}$ with feature $\boldsymbol{F_j}$ is.....

correct: spurious:

error: $\mathbf{x} - \hat{\mathbf{x}}$

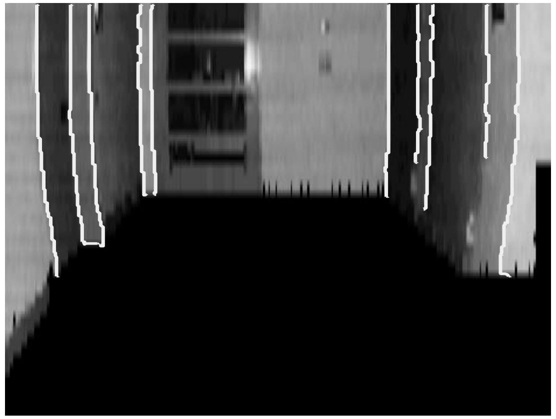
covariance: P

Consistency Divergence!

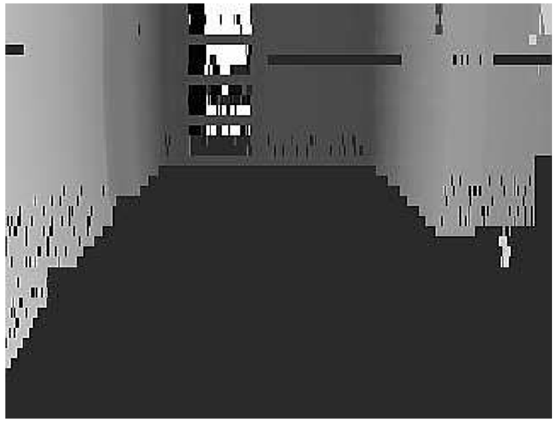
Courtesy Jose Neira

Data Association

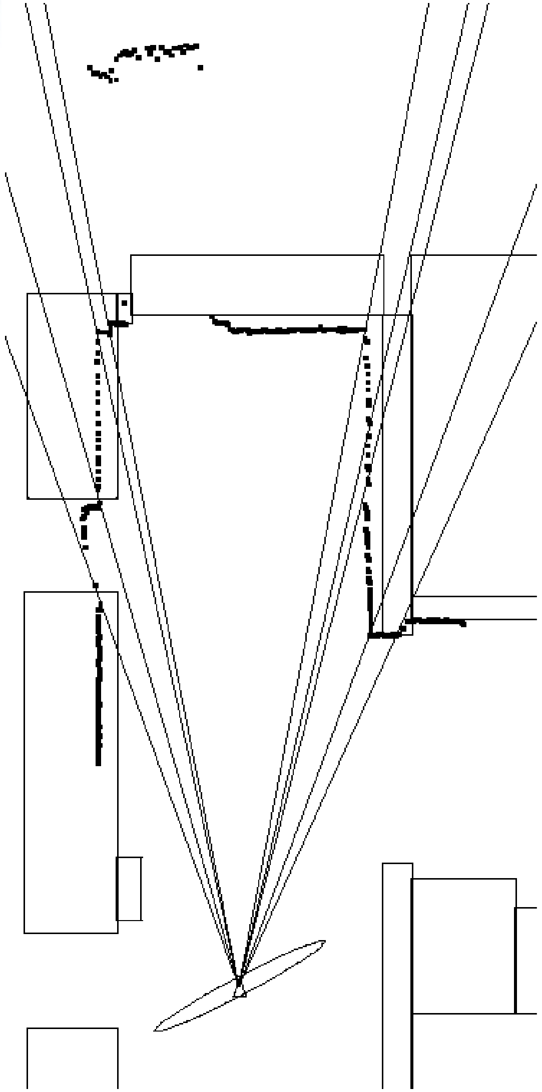
- Given an environment map
- And a set of sensor observations
- Associate observations with map elements



Vision



Laser



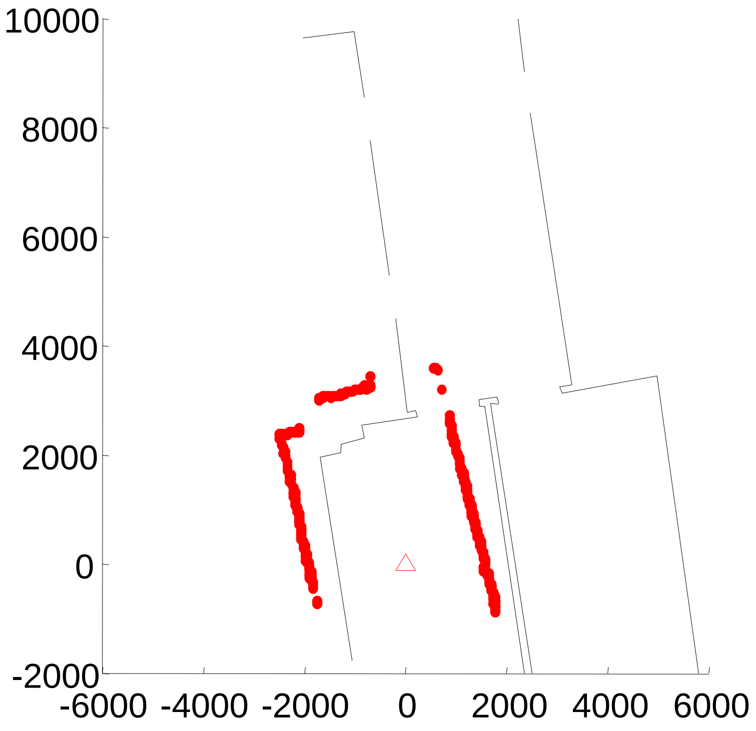
Courtesy Jose Neira

Difficulties: clutter

- Influence of the type, density, precision and robustness of features considered:

Laser scanner:

- *Small amount of features* (n)
- *Small amount of measurements* (m)
- Low spuriousness

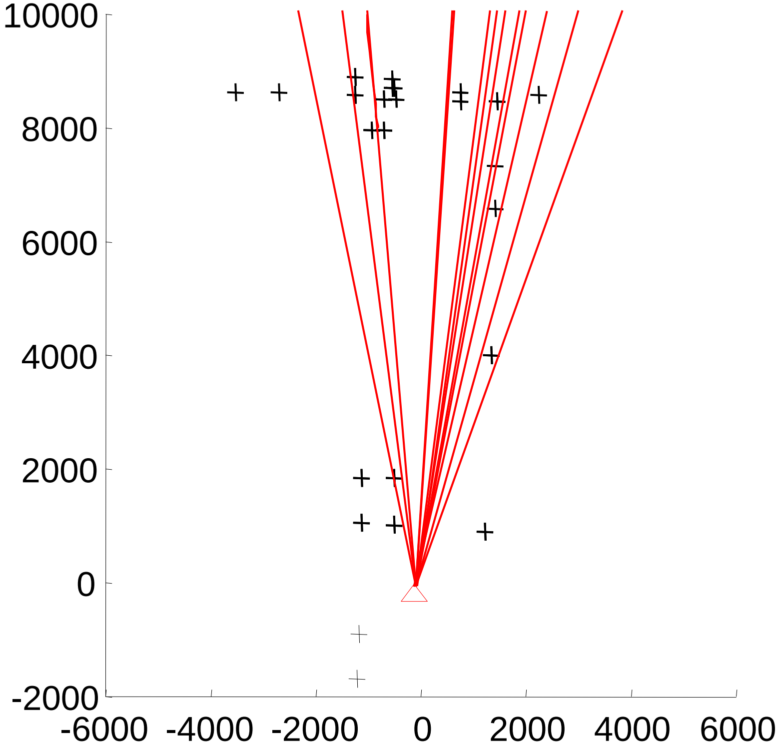


Low clutter

Courtesy Jose Neira

Difficulties: clutter

- Vertical Edge Monocular vision:



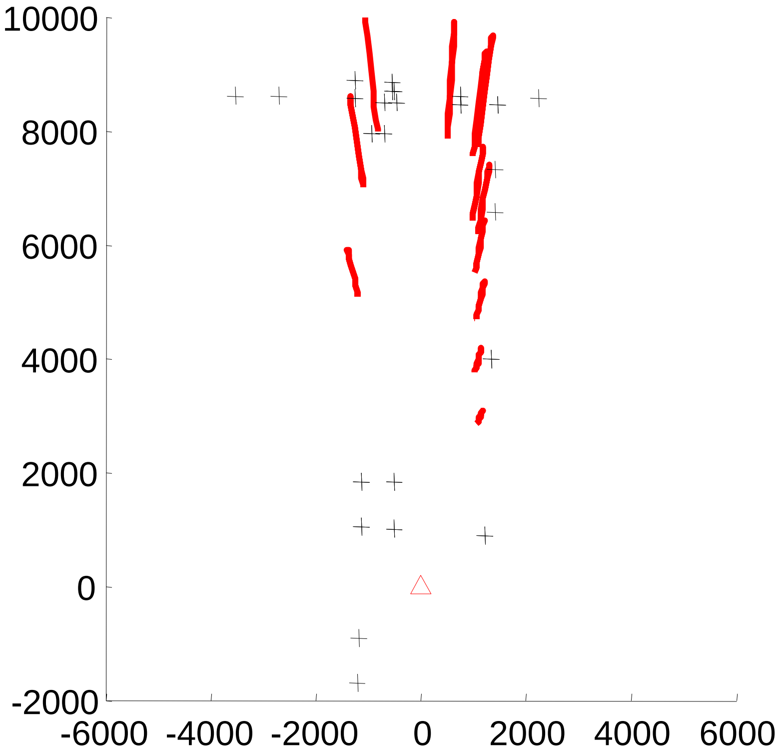
High clutter

- Many features (n large)
- Many measurements (m large)
- no depth information
- higher spuriousness

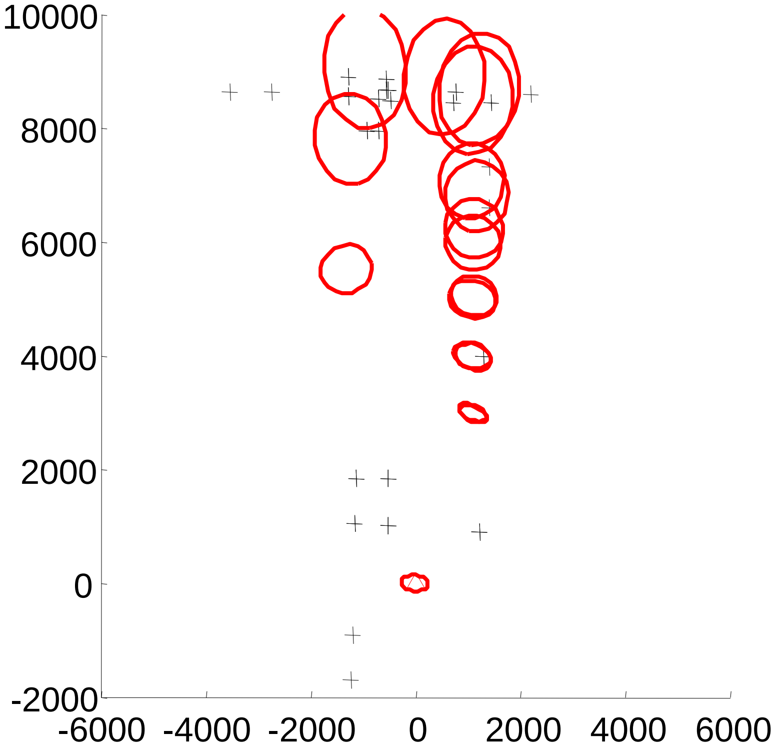
Courtesy Jose Neira

Difficulties: imprecision

- Both the sensor and the vehicle introduce imprecision



Vertical Edge Trinocular vision:
variable depth precision
good angular precision



Robot imprecision:
introduces CORRELATED error

Courtesy Jose Neira

Approaches to Data Association

- Search in configuration space: find robot location with maximal data to map overlapping
  - Can be done with raw data
  - Or with features
    - Speed of convergence?

- Search in correspondence space: find a consistent correspondence hypothesis and compute robot location
  1. Extract features from data
  2. Feature-based map (points, lines, trees, …)
  3. Search for measurement to map feature correspondences
     - Exponential number of solutions?

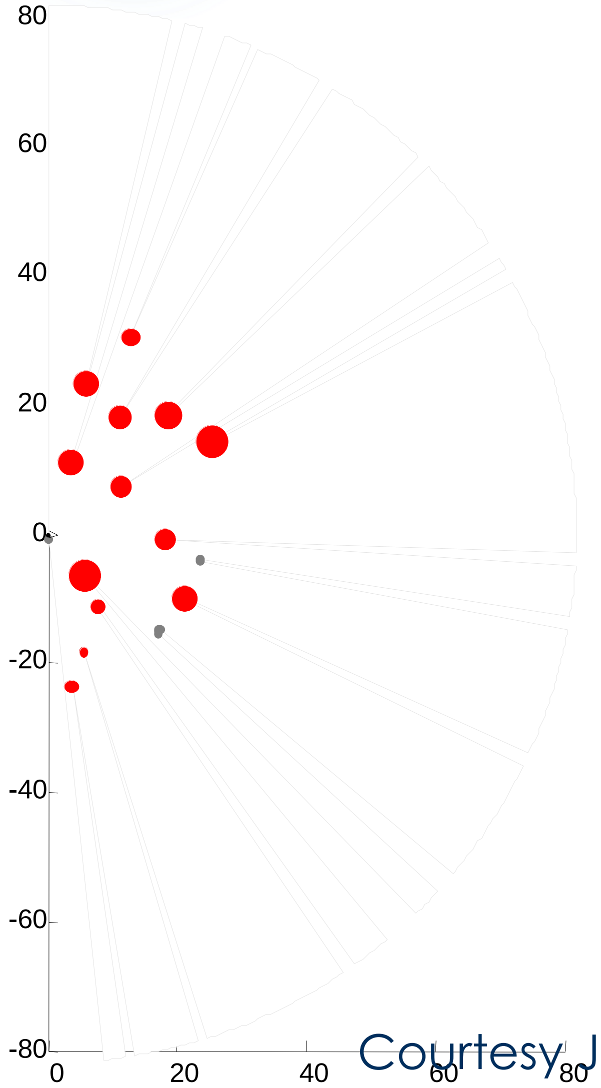
Courtesy Jose Neira

Example

- Vehicle with SICK laser

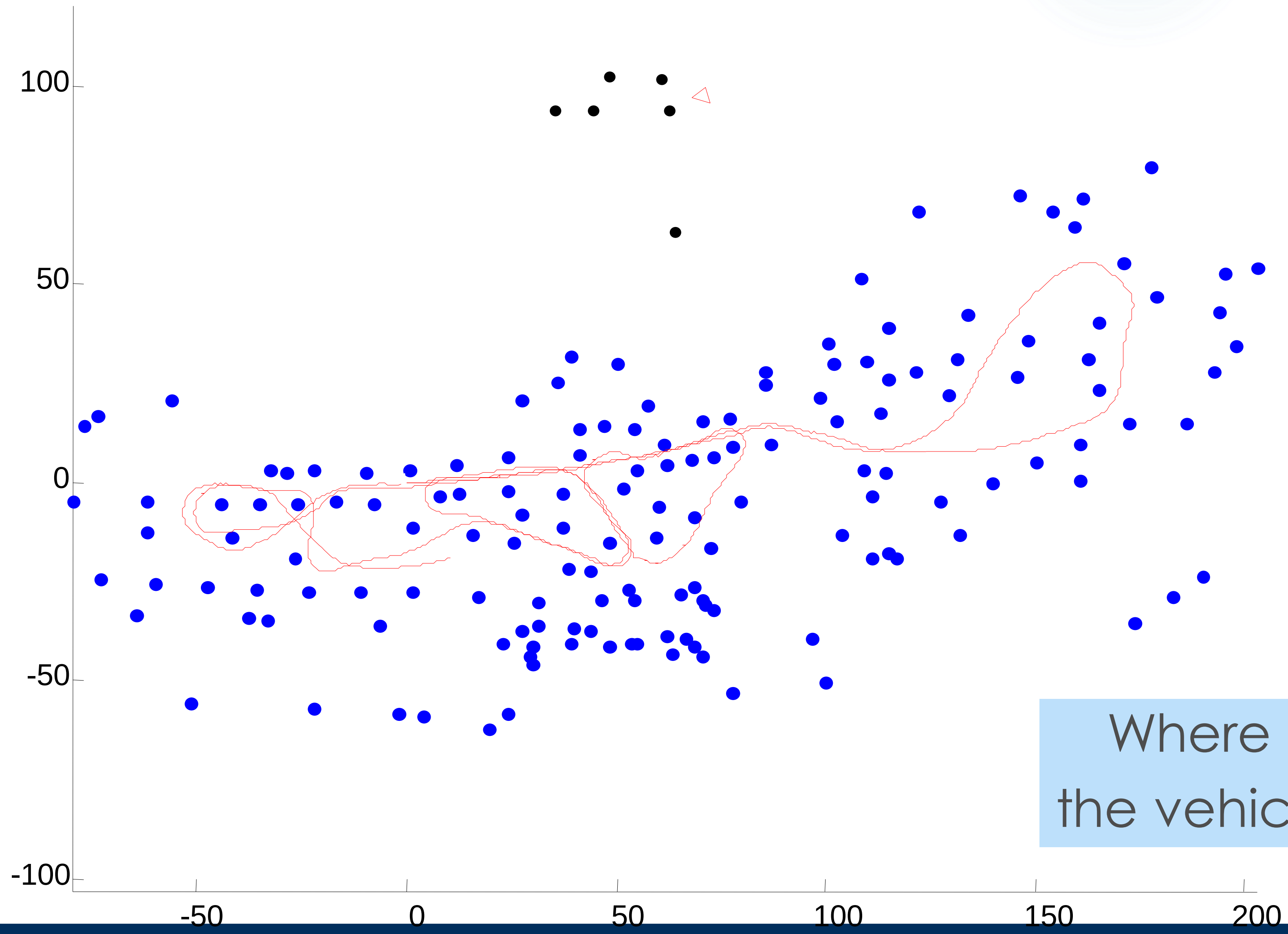
- Victoria Park, Sydney

- Detect trees:



Courtesy Jose Neira

Search in configuration space



Where is the vehicle?

Courtesy Jose Neira

Search in configuration space

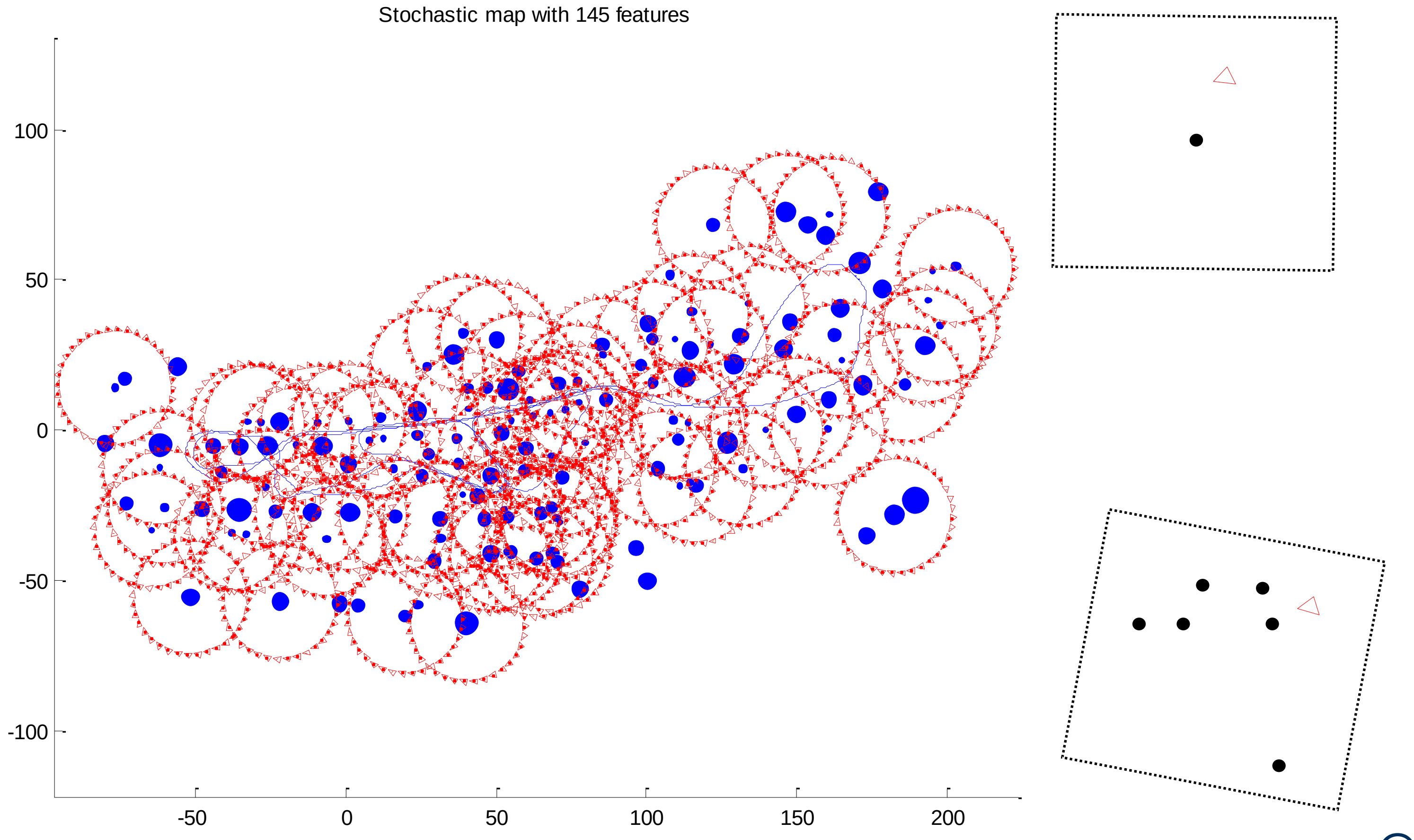
- Evaluate all vehicle locations looking for tree matchings

```
; (ui, vi): observation i relative to robot
; (xj, yj): absolute location of map feature j
;
mv = 0;
for x = xmin to xmax step xstep,
    for y = ymin to ymax step ystep,
        for t = tmin to tmax step tstep,
            v = 0;
            for i = 1 to m,
                (xi, yi) = compose(x,y,t,ui,vi);
                for j = 1 to n,
                    d = dist (xi, yi, xj, vj);
                    if d < dmax
                       v = v + 1;
                    fi
                rof
            rof
            if v > mv
               mv = v; best = (x,y,t);
            fi
        rof
    rof
rof
```

Courtesy Jose Neira

Data-driven search in configuration space (let the trees vote)

- Each pairing constrains the possible location of the vehicle (to a circle in this case).



Courtesy Jose Neira

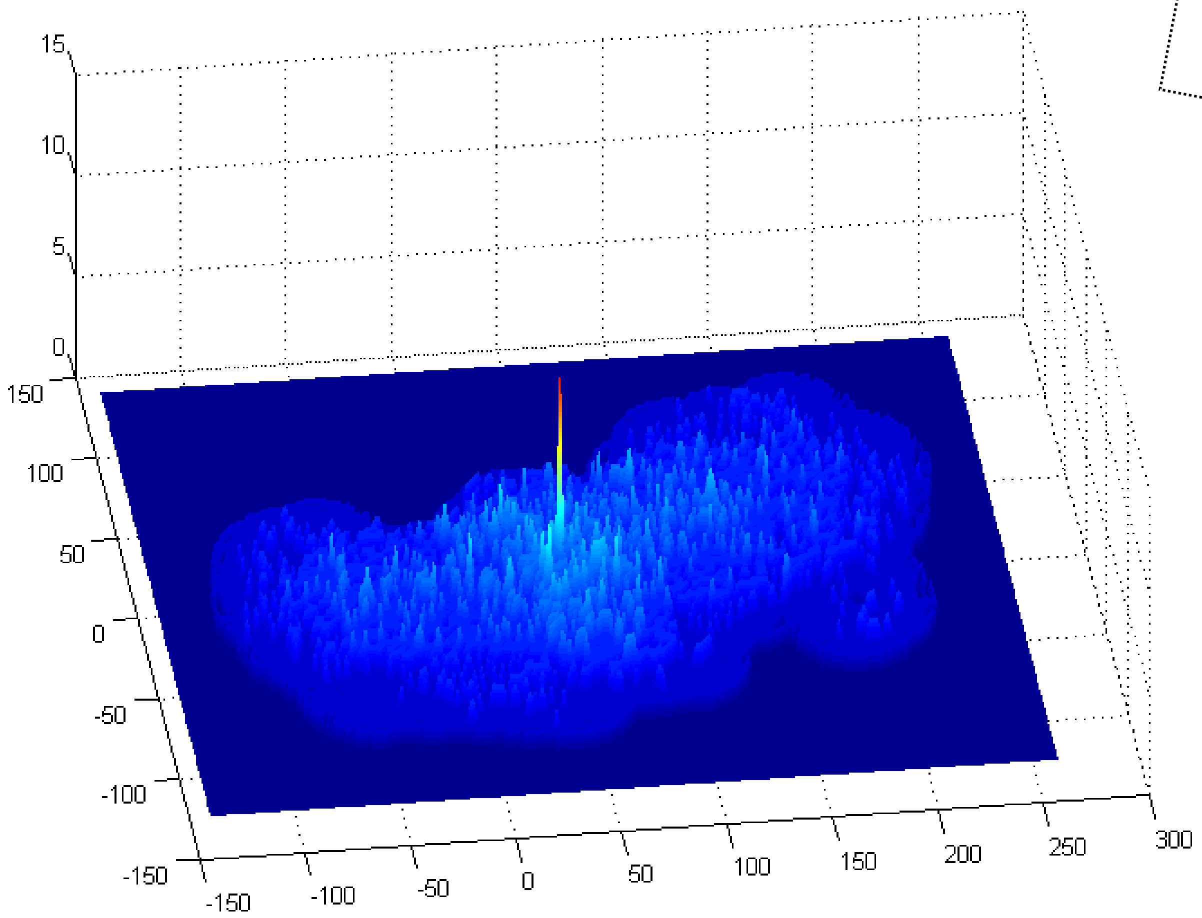
Data-driven search in configuration space (let the trees vote)

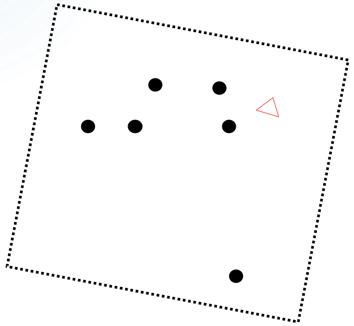
```
; (ui, vi): observation i relative to robot
; (xj, yj): absolute location of map feature j
;
for t = tmin to tmax step tstep,
   R = rotation(t);
   for i = 1 to m,
      (xi, yi) = R * (ui, vi);
      for j = 1 to n,
         (x, y) = (xj, yj) - (xi, vi);
         addvote(x, y, t);
      rof
   rof
rof
```

Courtesy Jose Neira

Results

- The most voted solution(s) stands out





Courtesy Jose Neira

Data Association
(in correspondence space)

- n map features:

$$\mathcal{F} = \{F_1 \dots F_n\}$$

- m sensor measurements:

$$\mathcal{E} = \{E_1 \dots E_m\}$$

- **Goal:** obtain a hypothesis that associates each observation E_i with a feature F_{j_i}

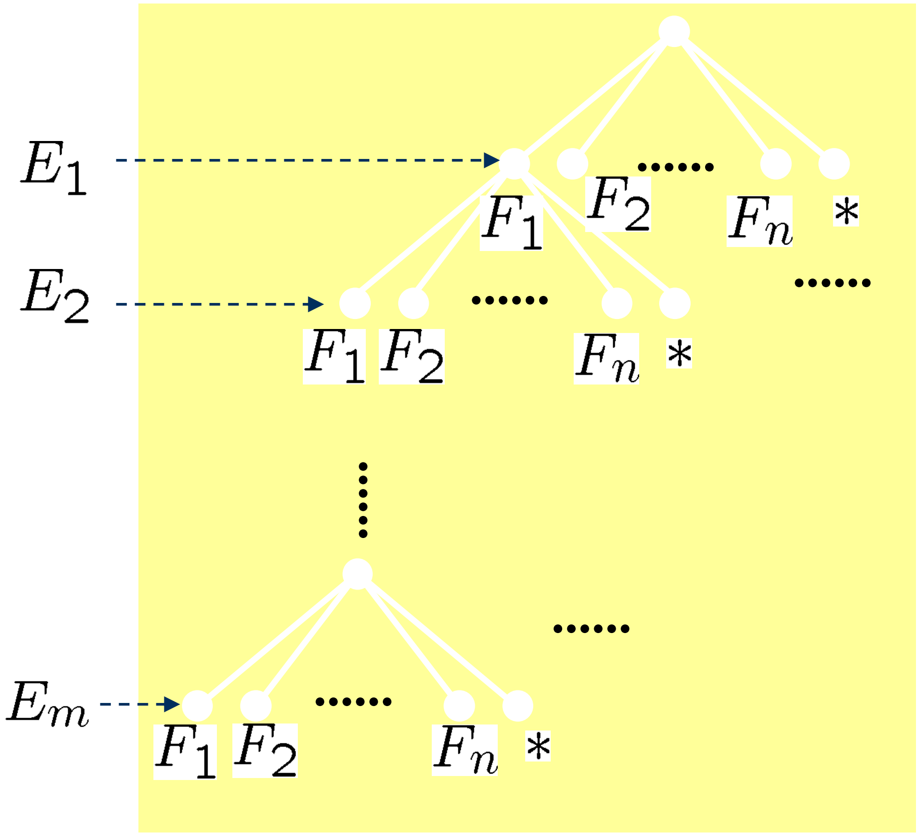
$$\mathcal{H}_m = [j_1 \dots j_i \dots j_m]$$

$$E_i \leftarrow j_i \rightarrow F_{j_i}$$

- Non matched observations:

$$j_i = 0$$

Interpretation tree (Grimson et al. 87):

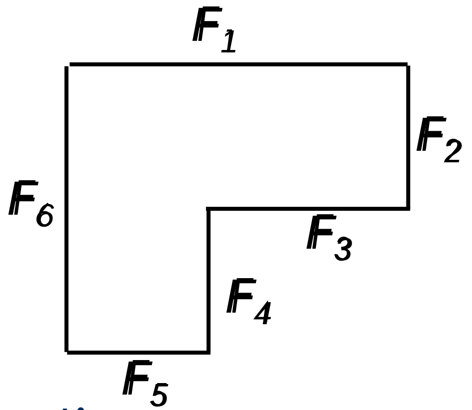


$(n+1)^m$ possible hypotheses

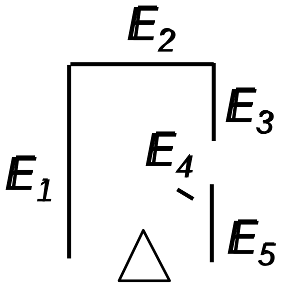
Courtesy Jose Neira

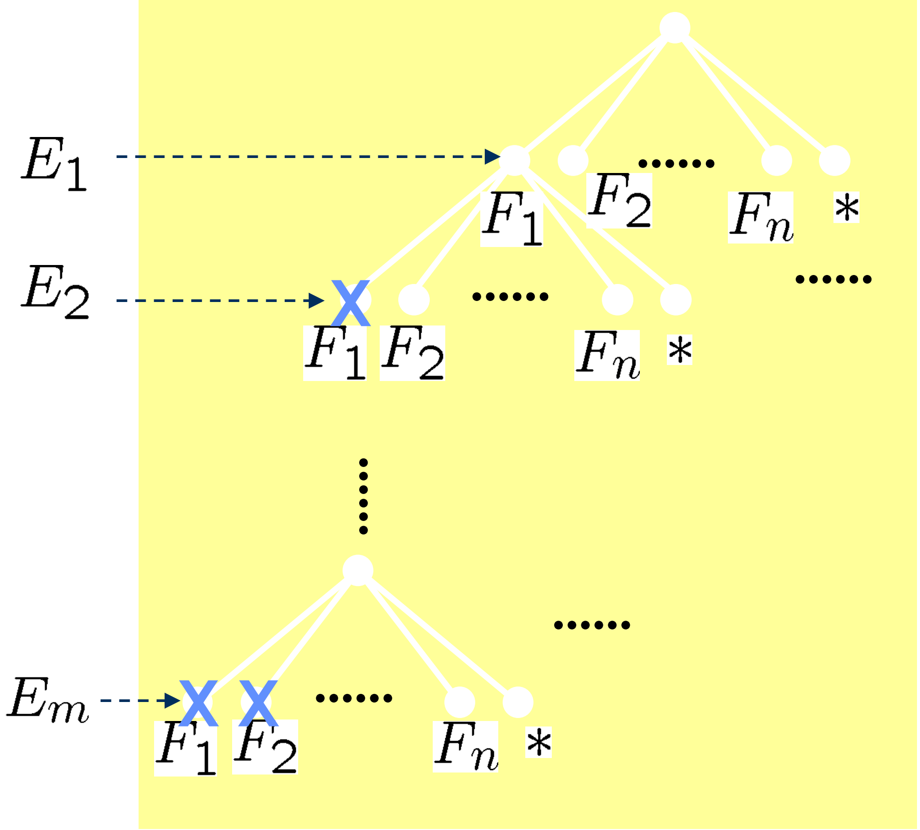
Use constraints to prune the tree

- Map:



- Observations:





- Constraints:
  - Feature location (needs an estimation of robot location)
  - Geometric relations: angles, distances,… (location independent)

Courtesy Jose Neira

Individual Compatibility

- Measurement equation for observation E_i and feature F_j

$$\mathbf{z}_i = \mathbf{h}_{ij}(\mathbf{x}_{\mathcal{F}}^B) + \mathbf{w}_i \qquad E[\mathbf{w}_i \mathbf{w}_i^T] = \mathbf{R}_i$$

$$\mathbf{z}_i \simeq \mathbf{h}_{ij}(\hat{\mathbf{x}}_{\mathcal{F}}^B) + \mathbf{H}_{ij}(\mathbf{x}_{\mathcal{F}}^B - \hat{\mathbf{x}}_{\mathcal{F}}^B) \qquad \mathbf{H}_{ij} = \left.\frac{\partial \mathbf{h}_{ij}}{\partial \mathbf{x}_{\mathcal{F}}^B}\right|_{(\hat{\mathbf{x}}_{\mathcal{F}}^B)}$$

- $\boldsymbol{E_i}$ and $\boldsymbol{F_j}$ are compatible if:

$$D_{ij}^2 = (\mathbf{z}_i - \mathbf{h}_{ij}(\hat{\mathbf{x}}_{\mathcal{F}}^B))^T \mathbf{P}_{ij}^{-1} (\mathbf{z}_i - \mathbf{h}_{ij}(\hat{\mathbf{x}}_{\mathcal{F}}^B)) < \chi_{d,\alpha}^2$$

$$\mathbf{P}_{ij} = \mathbf{H}_{ij} \mathbf{P}_{\mathcal{F}}^B \mathbf{H}_{ij}^T + \mathbf{R}_i$$

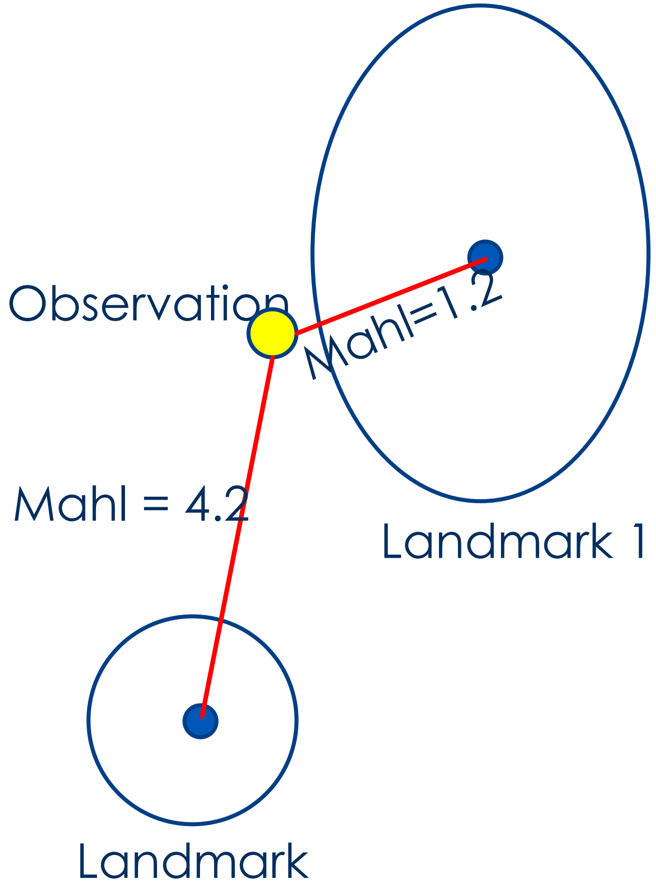
$d = length(\mathbf{z}_i)$

- Nearest Neighbour (NN) rule:
  - associate $\boldsymbol{E_i}$ with the feature $\boldsymbol{F_j}$ having smallest Mahalanobis distance $\boldsymbol{D_{ij}}$

Courtesy Jose Neira

Data Association: NN

- Simplest data association scheme:
  - Nearest neighbor.
- We have probabilistic estimates of where our landmarks are
- When we get an observation, which landmark does it match "best"?
  - Mahalanobis distance

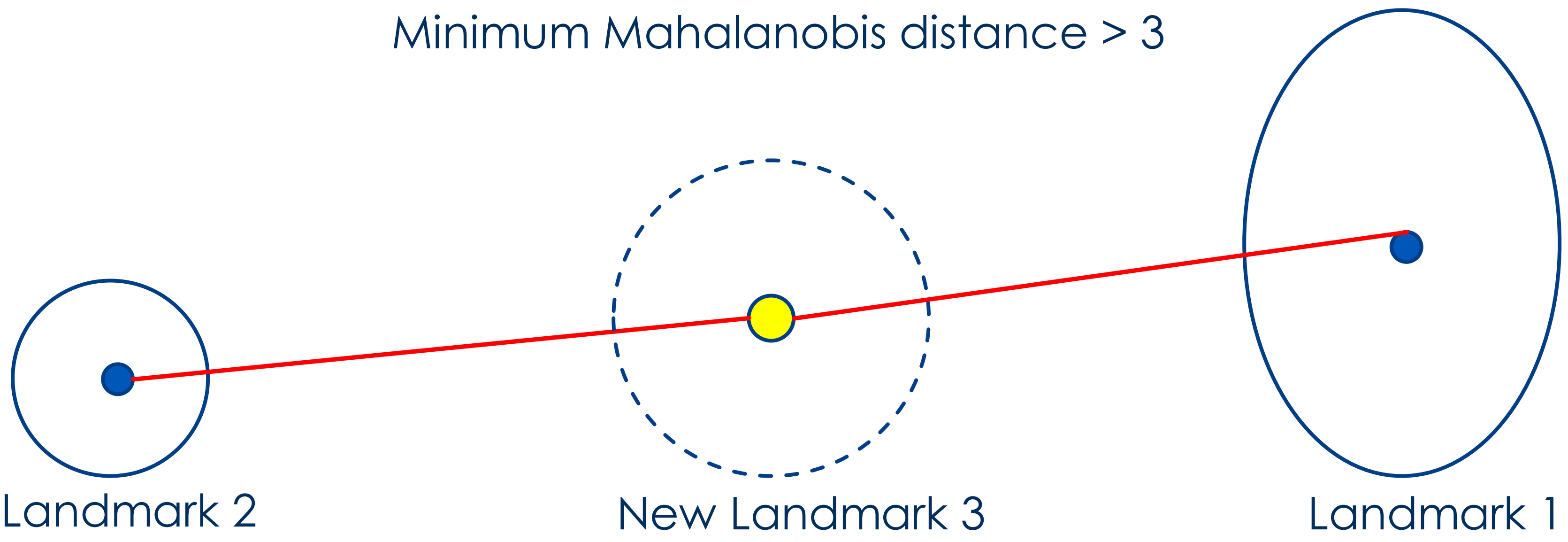


Nearest Neighbor

- Mahalanobis distance

$$K^2 = (x-\mu)^T \Sigma^{-1} (x-\mu)$$

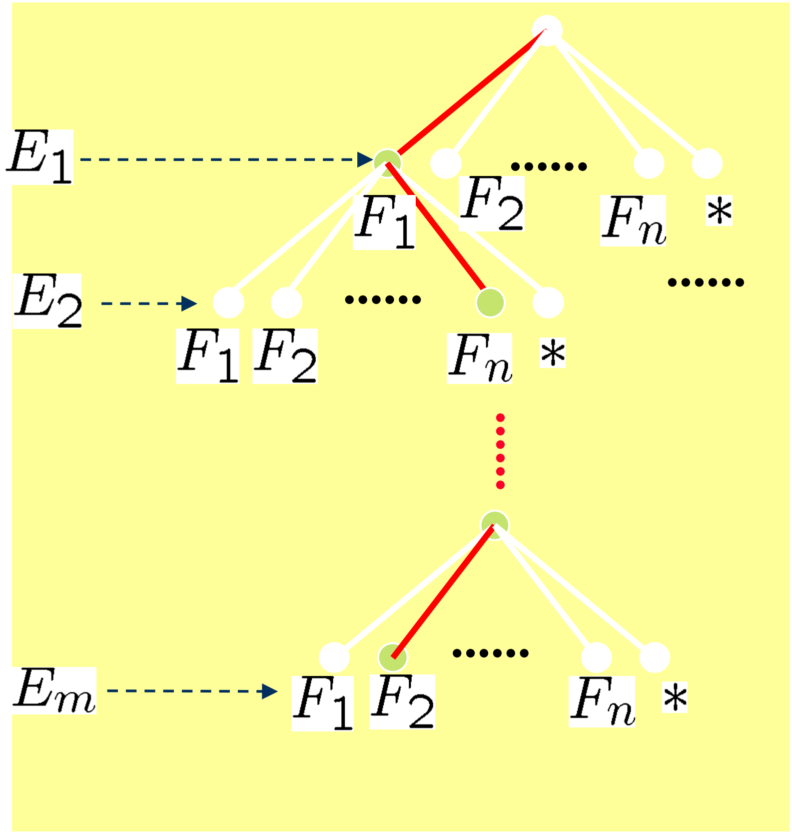
- What if Mahalanobis distance is really big?



Nearest Neighbor

```
NN:
for i = 1 to m -- measurement E_i
    D2min = Mahalanobis2(E_i, F_1)
    nearest = 1
    for j = 2 to n -- feature F_j
        Dij2 = Mahalanobis2(E_i, F_j)
        if Dij2 < D2min then
            nearest = j
            D2min = Dij2
        fi
    rof
    if D2min <= Chi2(d_i,alpha) then
        H(i) = nearest
    else
        H(i) = 0
    fi
rof
```

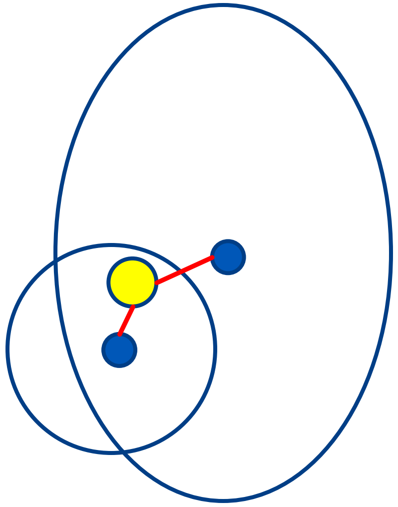
Greedy algorithm: $O(mn)$

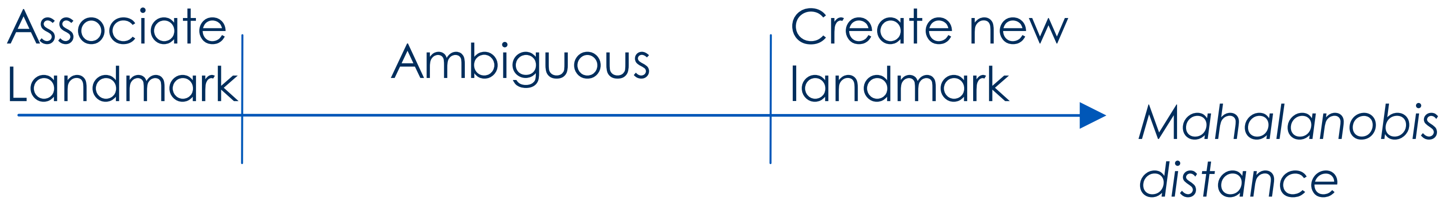


Courtesy Jose Neira

Nearest Neighbor

- Computational Cost?
  - Greedy algorithm: $O(mn)$
- Shortcomings?
- How do we make NN robust?
  - Reject matches if two “close” landmarks.
  - Avoid making decisions if unsure.





Why NN is brittle

- What we want to do is choose the joint correspondences that maximizes the likelihood of the observations given the data, i.e.

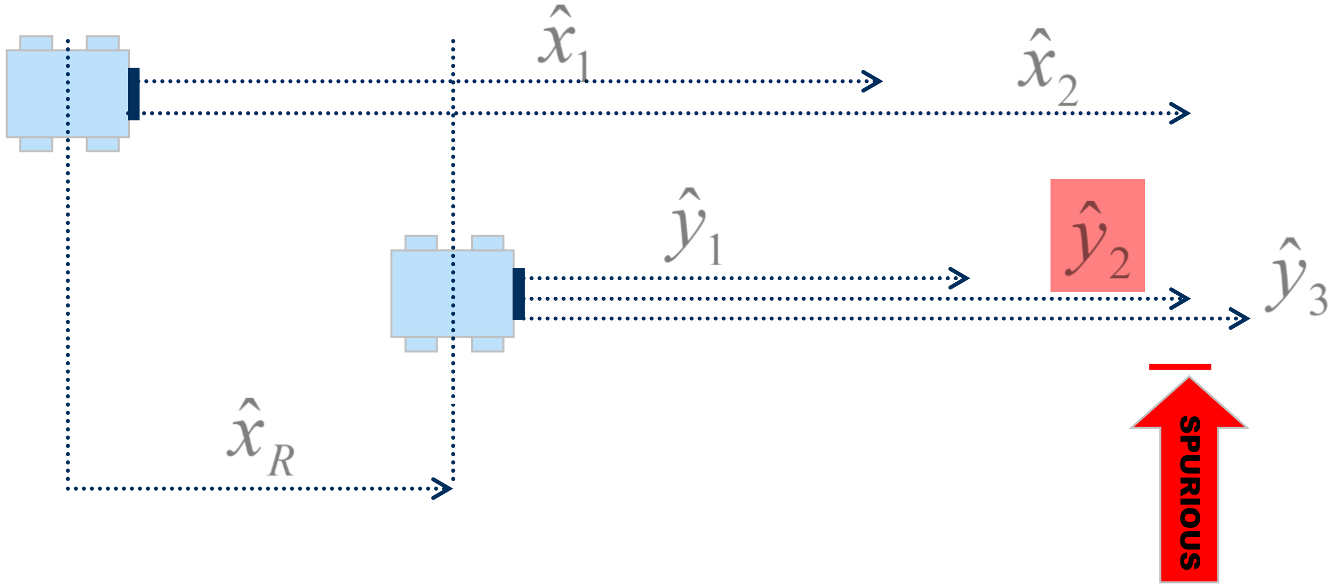
$$\hat{\mathbf{c}}_t = \underset{\mathbf{c}_t}{\operatorname{argmax}}\, p(\mathbf{z}_t \mid \mathbf{c}_{1:t}, \mathbf{z}_{1:t-1}, \mathbf{u}_{1:t})$$

  - Note this has exponentially many assignments

- Incremental ML, assumes conditional independence to simplify the problem so that the joint maximization becomes the maximization over individual assignments

$$\hat{\mathbf{c}}_t = \underset{\mathbf{c}_t}{\operatorname{argmax}} \prod_{i=1}^{m} p(\mathbf{z}_t^i \mid c_t^i, \mathbf{c}_{1:t-1}, \mathbf{z}_{1:t-1}, \mathbf{u}_{1:t})$$

Example: MonoRob



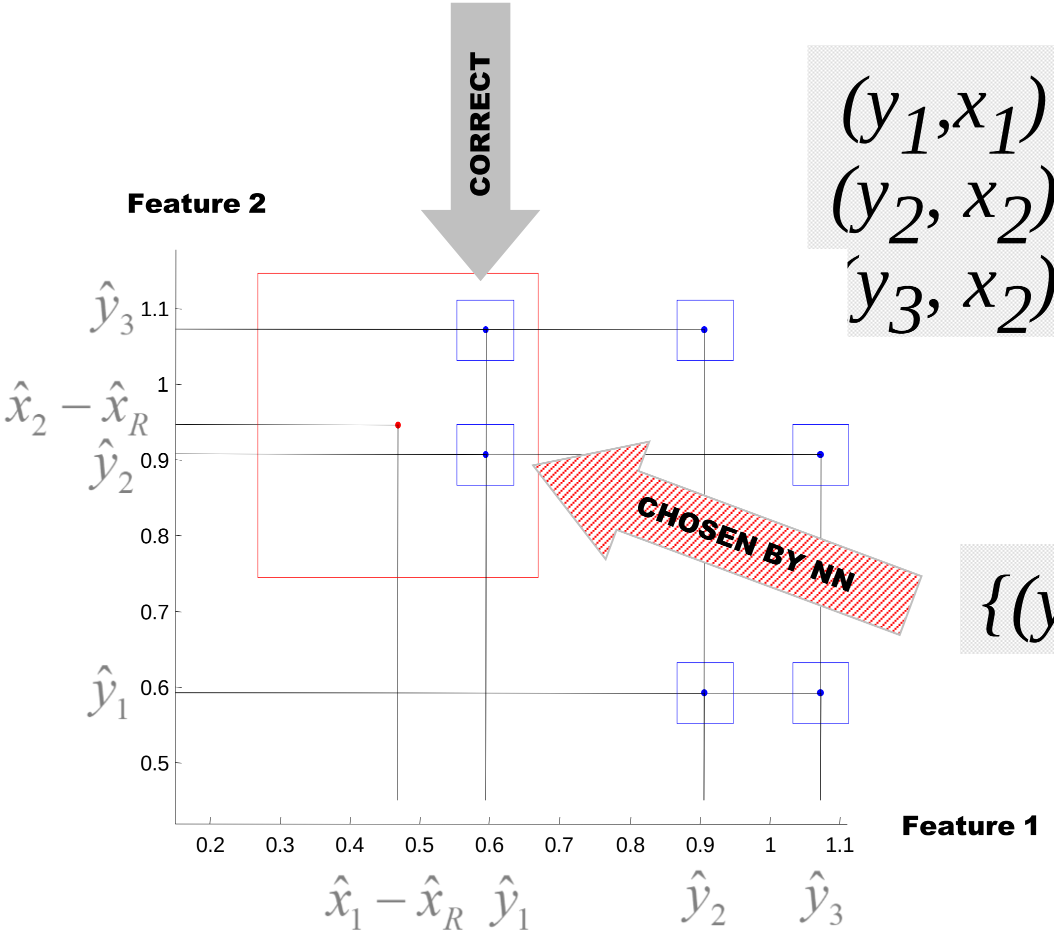
Nearest Neighbor: $\mathcal{H} = [1,\ 2,\ 0]$

True Association: $\mathcal{H} = [1,\ 0,\ 2]$

Courtesy Jose Neira

The fallacy of the Nearest Neighbour

- Parings may be individually compatible, but not jointly compatible:



(y_1,x_1) is IC compatible
(y_2, x_2) is IC compatible
(y_3, x_2) is IC compatible

$\{(y_1,x_1), (y_2, x_2)\}$

Courtesy Jose Neira

Joint Compatitibily

- Given a hypothesis

$$\mathcal{H} = [j_1, j_2, \cdots, j_s]$$

- Joint measurement equation

$$\begin{aligned} \mathbf{z}_{\mathcal{H}} &= \mathbf{h}_{\mathcal{H}}(\mathbf{x}_{\mathcal{F}}^{B}) + \mathbf{w}_{\mathcal{H}} \\ \mathbf{h}_{\mathcal{H}} &= \begin{bmatrix} \mathbf{h}_{1j_1} \\ \mathbf{h}_{2j_2} \\ \vdots \\ \mathbf{h}_{sj_s} \end{bmatrix} \end{aligned}$$

- The joint hypothesis is compatible if:

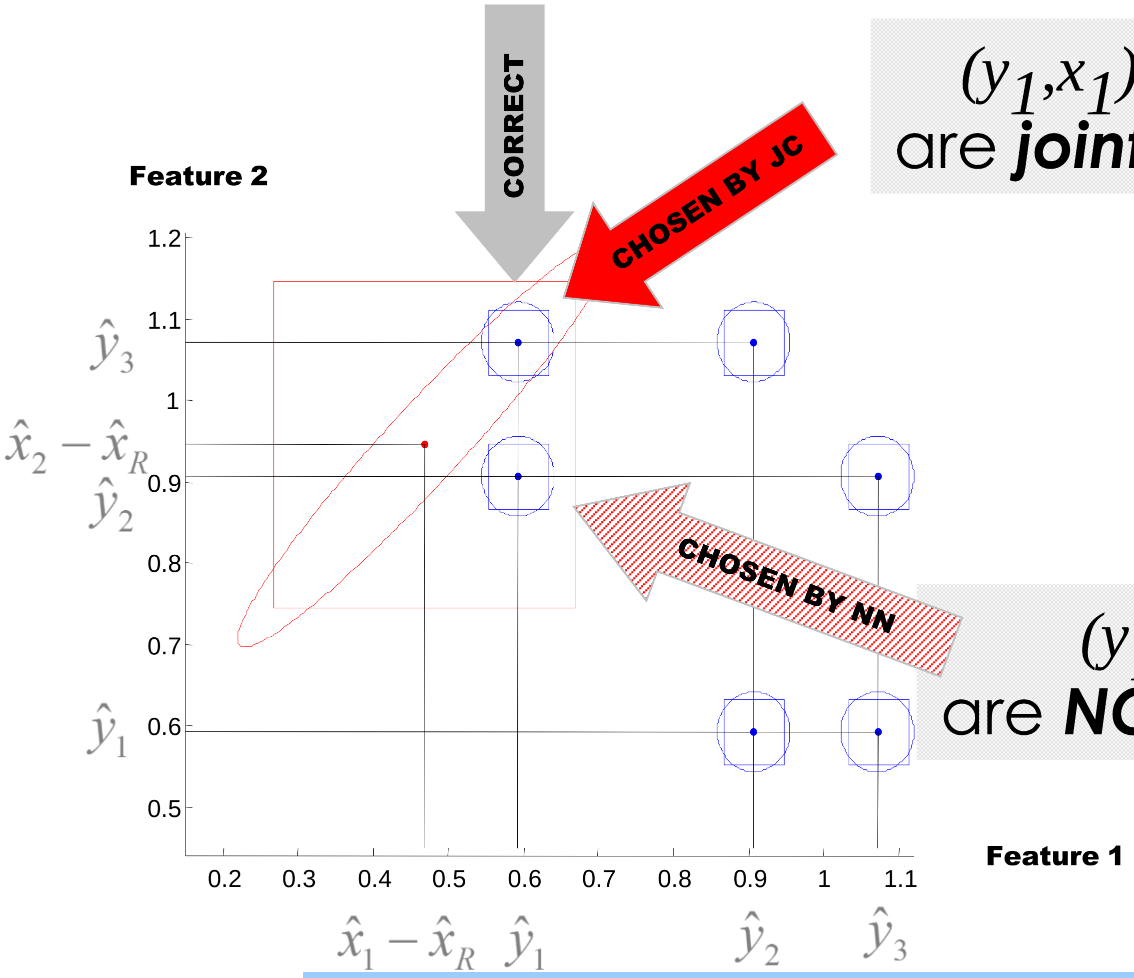
$$\begin{aligned} D_{\mathcal{H}}^2 &= (\mathbf{z}_{\mathcal{H}} - \mathbf{h}_{\mathcal{H}}(\hat{\mathbf{x}}_{\mathcal{F}}^{B}))^T C_{\mathcal{H}}^{-1} (\mathbf{z}_{\mathcal{H}} - \mathbf{h}_{\mathcal{H}}(\hat{\mathbf{x}}_{\mathcal{F}}^{B})) < \chi_{d,\alpha}^2 \\ C_{\mathcal{H}} &= \mathbf{H}_{\mathcal{H}} \mathbf{P}_{\mathcal{F}}^{B} \mathbf{H}_{\mathcal{H}}^T + \mathbf{R}_{\mathcal{H}} \end{aligned}$$

d = length(z)

Courtesy Jose Neira

Individual .vs. Joint Compatibility

- Joint Compatibility assures consistency:



(y_1, x_1) and (y_3, x_2) are ***jointly*** compatible

(y_1, x_1) and (y_2, x_2) are ***NOT jointly*** compatible

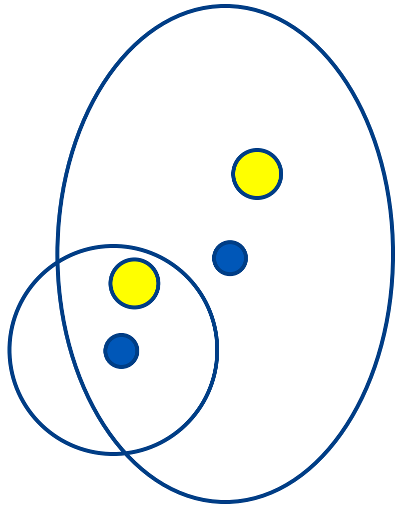
J. Neira, J.D. Tardós. Data Association in Stochastic Mapping using the Joint Compatibility Test, IEEE Trans. Robotics and Automation, Vol. 17, No. 6, Dec 2001, pp 890 –897

Courtesy Jose Neira

Joint Data Association

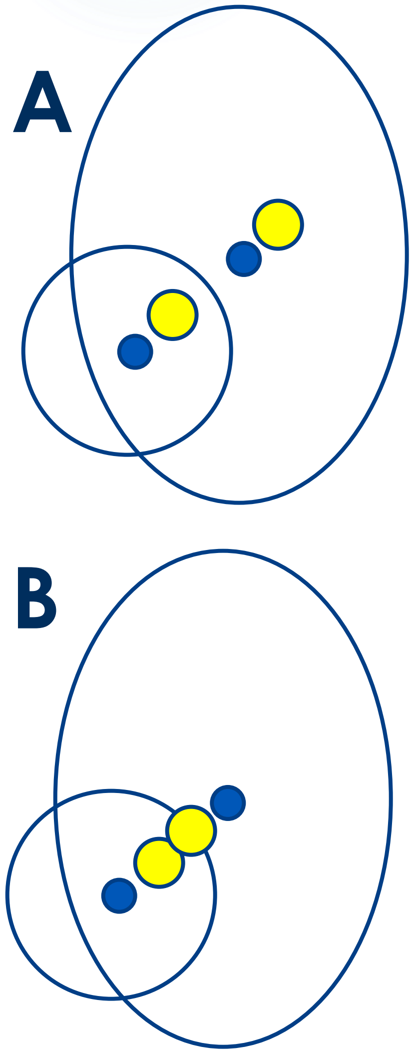
- Data association on individual objects is highly error prone
  - Wait until we have more data
  - Process several observations *jointly*
  - Pick best joint assignment

- Enforce mutual exclusion
  - Each observation can match to only one landmark
  - Each landmark can match to only one observation



Joint Assignment

- In EKF, landmark positions tend to be correlated
  - They have trajectory error in common
  - i.e., landmark estimates move together.
- Which of these assignments is more jointly compatible?



Example of how robot pose introduces correlation in predicted observations

- Suppose we have two measurements, i.e., m=2

$$\mathbf{z}_t = \{\mathbf{z}_t^1, \mathbf{z}_t^2\}$$

- Our observation Jacobian looks like:

$$H_{\mathcal{H}} = \begin{bmatrix} H_r, & 0..., & H_i, & 0, & ...0 \\ H_r, & 0..., & 0, & H_j, & ...0 \end{bmatrix}$$

- The observation covariance is then:

$$C_{\mathcal{H}} = H_{\mathcal{H}} \Sigma_t H_{\mathcal{H}}^\top + R_t$$

$$= \begin{bmatrix} H_r\Sigma_{rr}H_r^\top + H_r\Sigma_{ri}H_i^\top + H_i\Sigma_{ir}H_r^\top + H_i\Sigma_{ii}H_i^\top & H_r\Sigma_{rr}H_r^\top + H_r\Sigma_{rj}H_j^\top + H_i\Sigma_{ir}H_r^\top + H_i\Sigma_{ij}H_j^\top \\ H_r\Sigma_{rr}H_r^\top + H_r\Sigma_{ri}H_i^\top + H_j\Sigma_{jr}H_r^\top + H_j\Sigma_{ji}H_i^\top & H_r\Sigma_{rr}H_r^\top + H_r\Sigma_{rj}H_j^\top + H_j\Sigma_{jr}H_r^\top + H_j\Sigma_{jj}H_j^\top \end{bmatrix}$$

$$+ \begin{bmatrix} R_t^1 & 0 \\ 0 & R_t^2 \end{bmatrix}$$

Computing Joint Compatibility

- Hypothetical data association
  - observed three landmarks:
  - (Nice linear observation model this time)

$$\begin{bmatrix} z_1 \\ z_2 \\ z_3 \\ z_4 \\ z_5 \\ z_6 \end{bmatrix} = f(x, w) = \begin{bmatrix} x_3 + w_1 \\ x_4 + w_2 \\ x_{11} + w_3 \\ x_{12} + w_4 \\ x_7 + w_5 \\ x_8 + w_6 \end{bmatrix}$$

- What is the Mahalanobis distance of the observation with respect to the noisy observation of the RHS?
  - What is the mean and covariance of the RHS?

$$\mu_f = E[f(x, w)]$$
$$\Sigma_f = J_x \Sigma_x J_x^T + J_w \Sigma_w J_w^T$$

$$K^2 = (x - \mu)^T \Sigma^{-1} (x - \mu)$$

Joint Compatibility: Problems

- We need to search over possible assignments.
  - Still an exponential number!
  - For each, we have to compute Mahalanobis distance
    - Involves inverting a matrix of size $2N$ x $2N$ (where N = # of jointly estimated landmarks)

- Joint Compatibility Branch & Bound is (partial) solution.
  - Reduce search space via heuristics
  - Use incremental matrix inversion “trick”

Joint Compatibility Branch and Bound (JCBB)

Purpose: Find the largest hypothesis with jointly consistent pairings

JCBB(H, i): -- *find pairings for feature E_i, $i <= m$*

if i > m -- *leaf node?*

if pairings(H) > pairings(Best) -- *did better?*

Best = H

else

for j = 1 to n

if individual_compatibility(E_i, F_j) and then

joint_compatibility(H, E_i, F_j)

JCBB([H j], i + 1) -- *pairing (E_i, F_j) accepted*

if pairings(H) + m – i >= pairings(Best) -- *can do better?*

JCBB([H 0], i + 1) -- *star node: E_i not paired*

JCBB Discussion

- Upsides:
  - Nice probabilistic foundation
  - Quite effective in practice

- Downsides:
  - Still exponential time in the worst case
  - Like all joint assignment schemes, adds latency to data association.