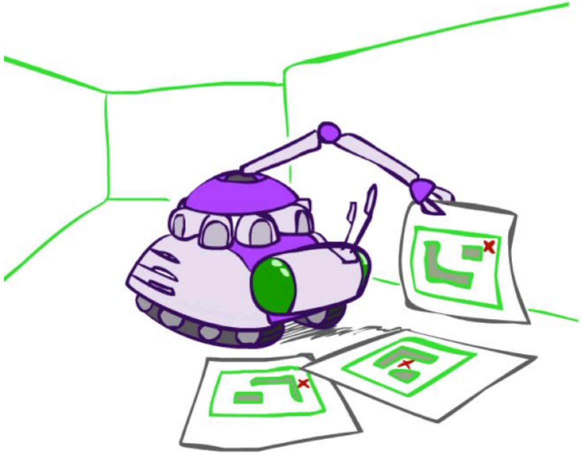
COMS W4733: Computational Aspects of Robotics

Lecture 23: EKF-SLAM and FastSLAM



Instructor: Tony Dear

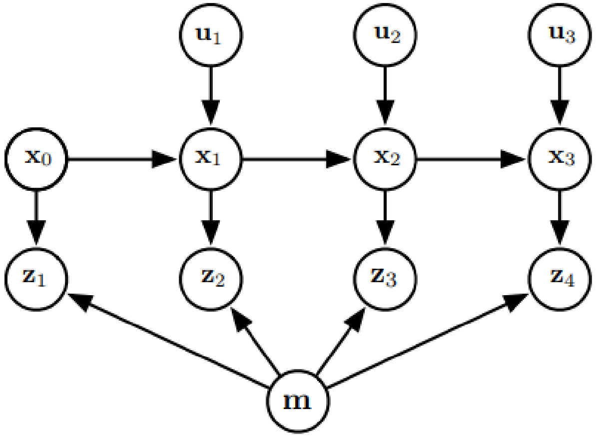
Slide materials from K. Arras, W. Burgard, J. Castellanos, G. Grisetti, J. Leonard, M. Montemerlo, C. Stachniss, et al.

SLAM

- **Simultaneous localization and mapping (SLAM)** entails learning a map and locating the robot at the same time!
- Clearly harder than either problem separately

- Main approach: Add the map (e.g., in the form of feature locations) to the state and update using Bayesian filtering

- Map is another hidden state!



Bayes Filter with Landmarks

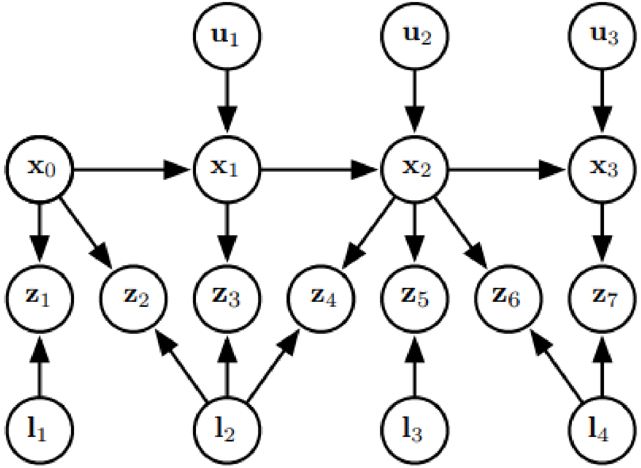
- Belief distribution: $$B(\boldsymbol{x}_t, \boldsymbol{m}) = p(\boldsymbol{x}_t, \boldsymbol{m} | \, \boldsymbol{u}_{1:t}, \boldsymbol{z}_{1:t})$$

- Prediction:

$$B'(\boldsymbol{x}_t, \boldsymbol{m}) = \int \underbrace{p(\boldsymbol{x}_t | \boldsymbol{x}_{t-1}, \boldsymbol{u}_t)}_{\text{Transition model}} B(\boldsymbol{x}_{t-1}, \boldsymbol{m}) \, d\boldsymbol{x}_{t-1}$$

- Observation:

$$B(\boldsymbol{x}_t, \boldsymbol{m}) = \eta^{-1} \underbrace{p(\boldsymbol{z}_t | \boldsymbol{x}_t, \boldsymbol{m})}_{\text{Observation model}} B'(\boldsymbol{x}_t, \boldsymbol{m})$$



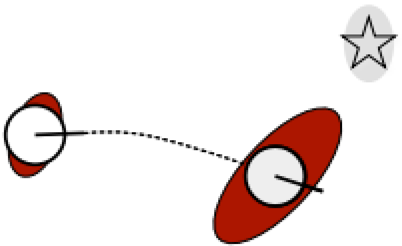
EKF for SLAM

- As with vanilla state estimation, we assume that our belief over the augmented state with landmark locations is Gaussian
- Run through standard Kalman operations to update robot state and map

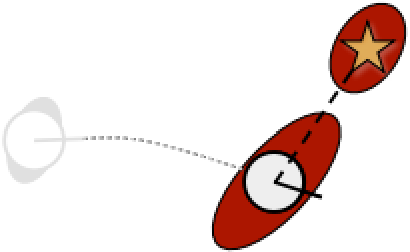
$$\boldsymbol{\mu} = \begin{pmatrix} x \\ y \\ \theta \\ m_{1x} \\ m_{1y} \\ \vdots \\ m_{nx} \\ m_{ny} \end{pmatrix} = \begin{pmatrix} \boldsymbol{X} \\ \boldsymbol{m}_1 \\ \vdots \\ \boldsymbol{m}_n \end{pmatrix} \qquad \boldsymbol{\Sigma} = \begin{pmatrix} \boldsymbol{\Sigma}_{\boldsymbol{XX}} & \Sigma_{\boldsymbol{X}\boldsymbol{m}_1} & \cdots & \Sigma_{\boldsymbol{X}\boldsymbol{m}_n} \\ \Sigma_{\boldsymbol{m}_1\boldsymbol{X}} & \boldsymbol{\Sigma}_{\boldsymbol{m}_1\boldsymbol{m}_1} & \cdots & \boldsymbol{\Sigma}_{\boldsymbol{m}_1\boldsymbol{m}_n} \\ \vdots & \vdots & \ddots & \vdots \\ \Sigma_{\boldsymbol{m}_n\boldsymbol{X}} & \boldsymbol{\Sigma}_{\boldsymbol{m}_n\boldsymbol{m}_1} & \cdots & \boldsymbol{\Sigma}_{\boldsymbol{m}_n\boldsymbol{m}_n} \end{pmatrix}$$

EKF SLAM Steps

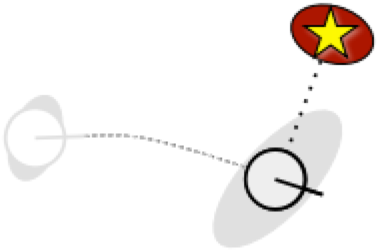
1. State transition



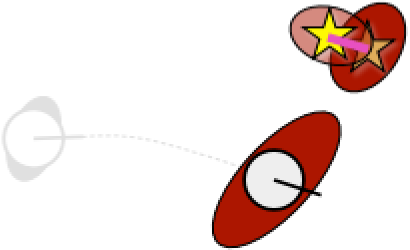
2. Predict measurement



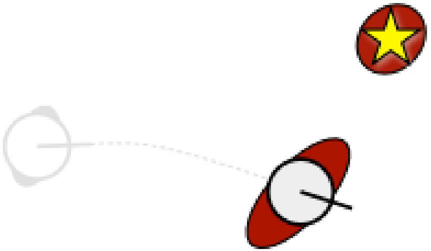
3. Make measurement



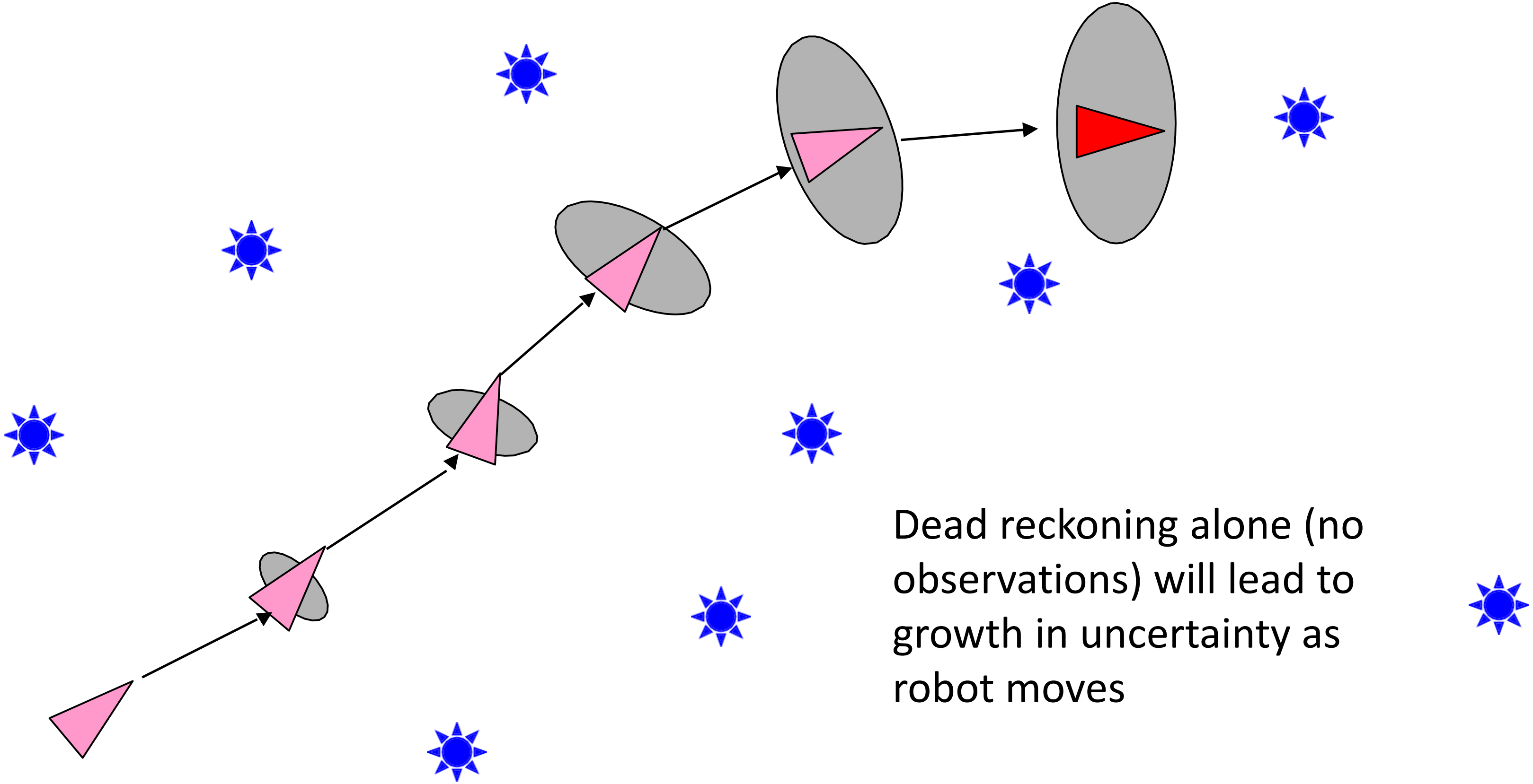
4. Associate data



5. Update belief

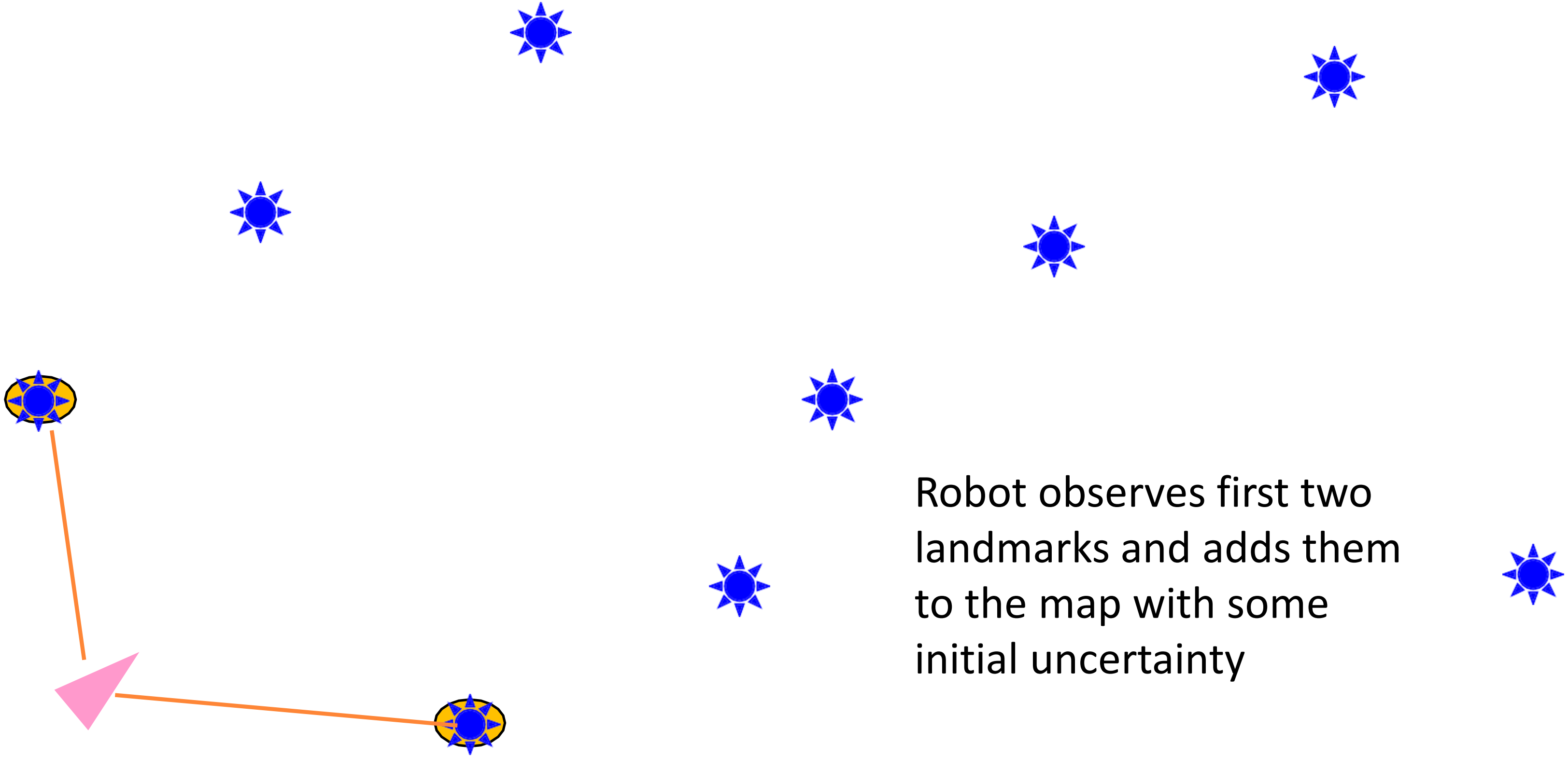


Example: SLAM Without Landmarks



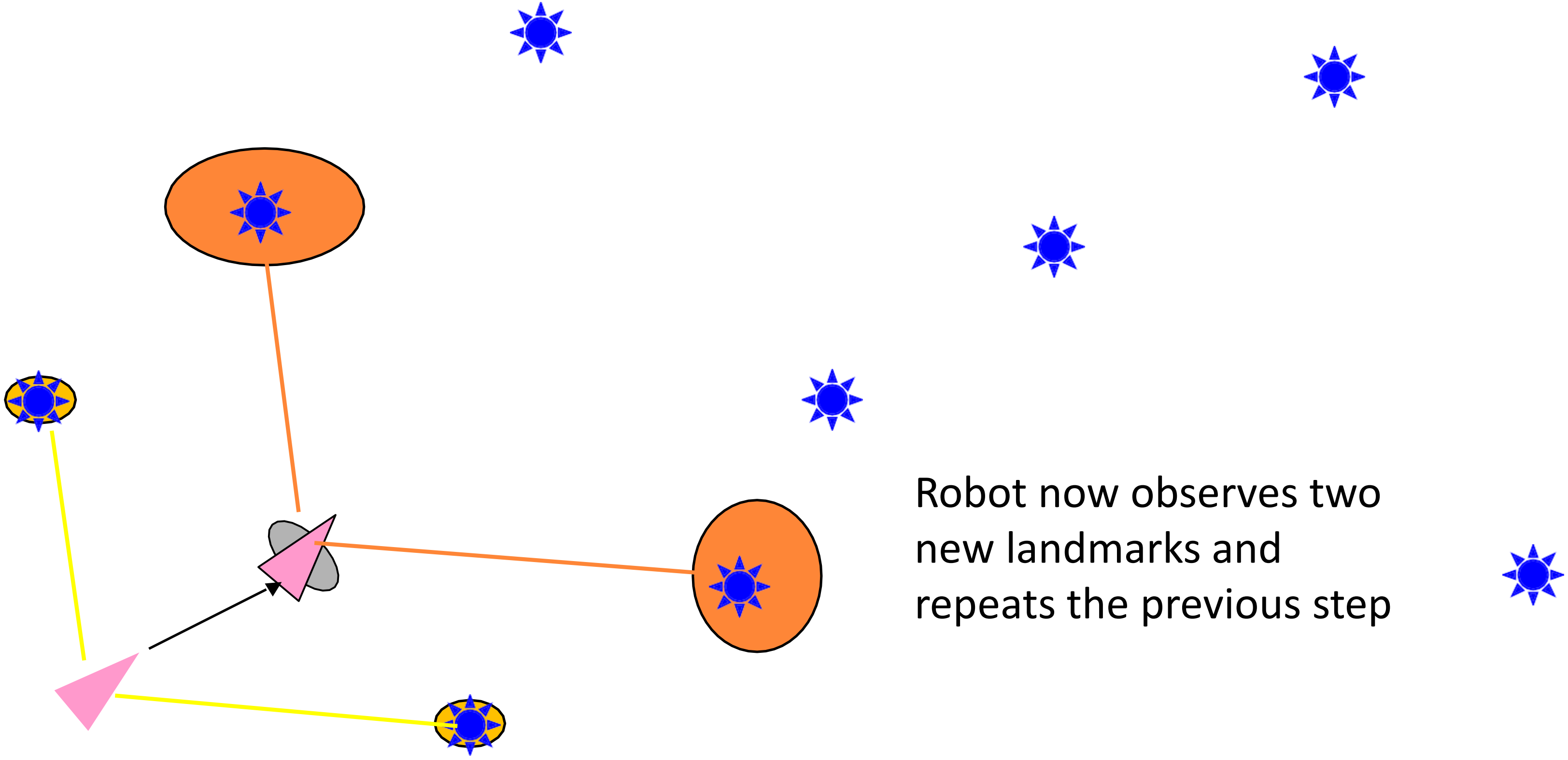
Dead reckoning alone (no observations) will lead to growth in uncertainty as robot moves

Example: SLAM With Landmarks



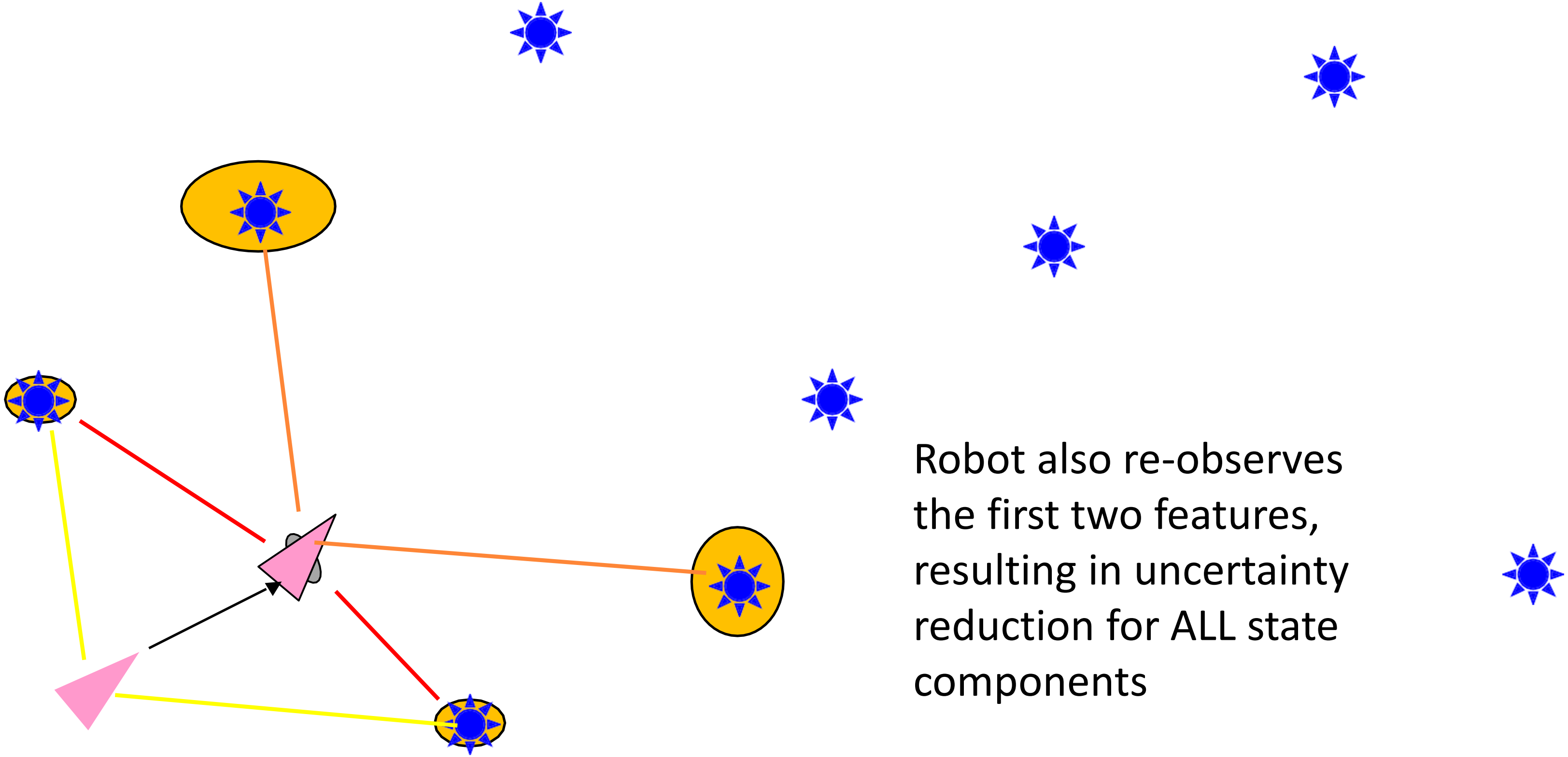
Robot observes first two landmarks and adds them to the map with some initial uncertainty

Example: SLAM With Landmarks



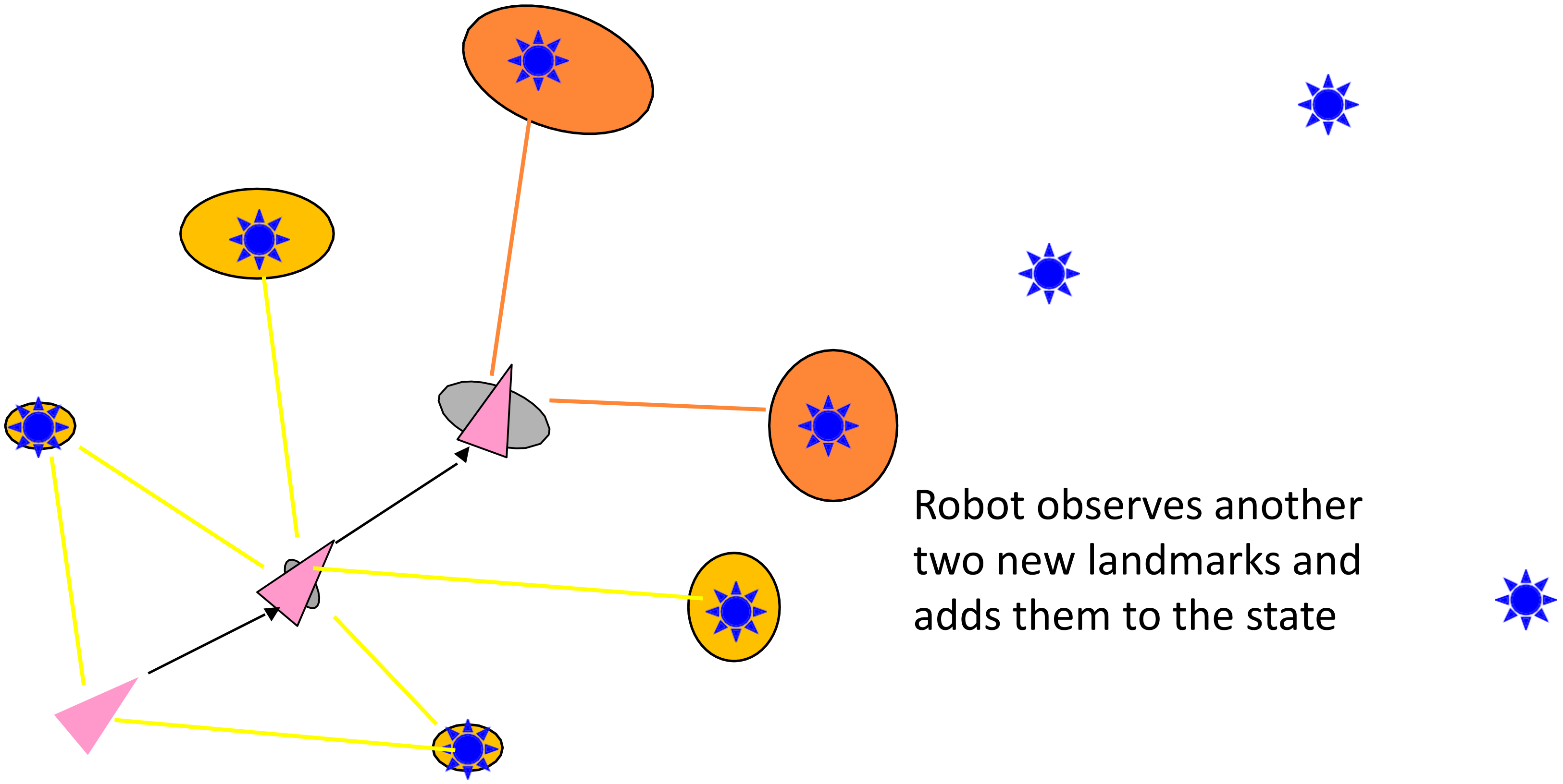
Robot now observes two new landmarks and repeats the previous step

Example: SLAM With Landmarks



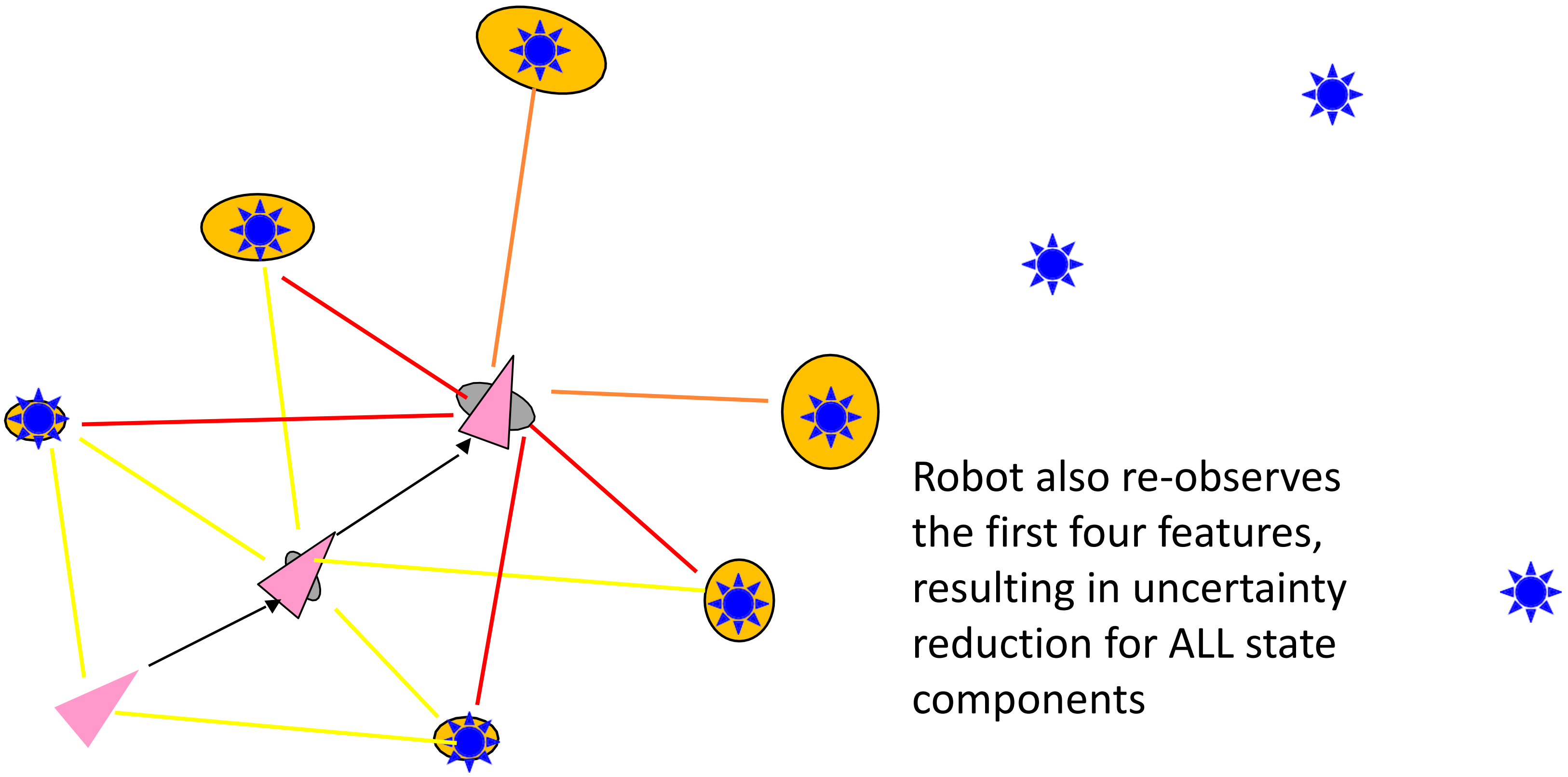
Robot also re-observes the first two features, resulting in uncertainty reduction for ALL state components

Example: SLAM With Landmarks



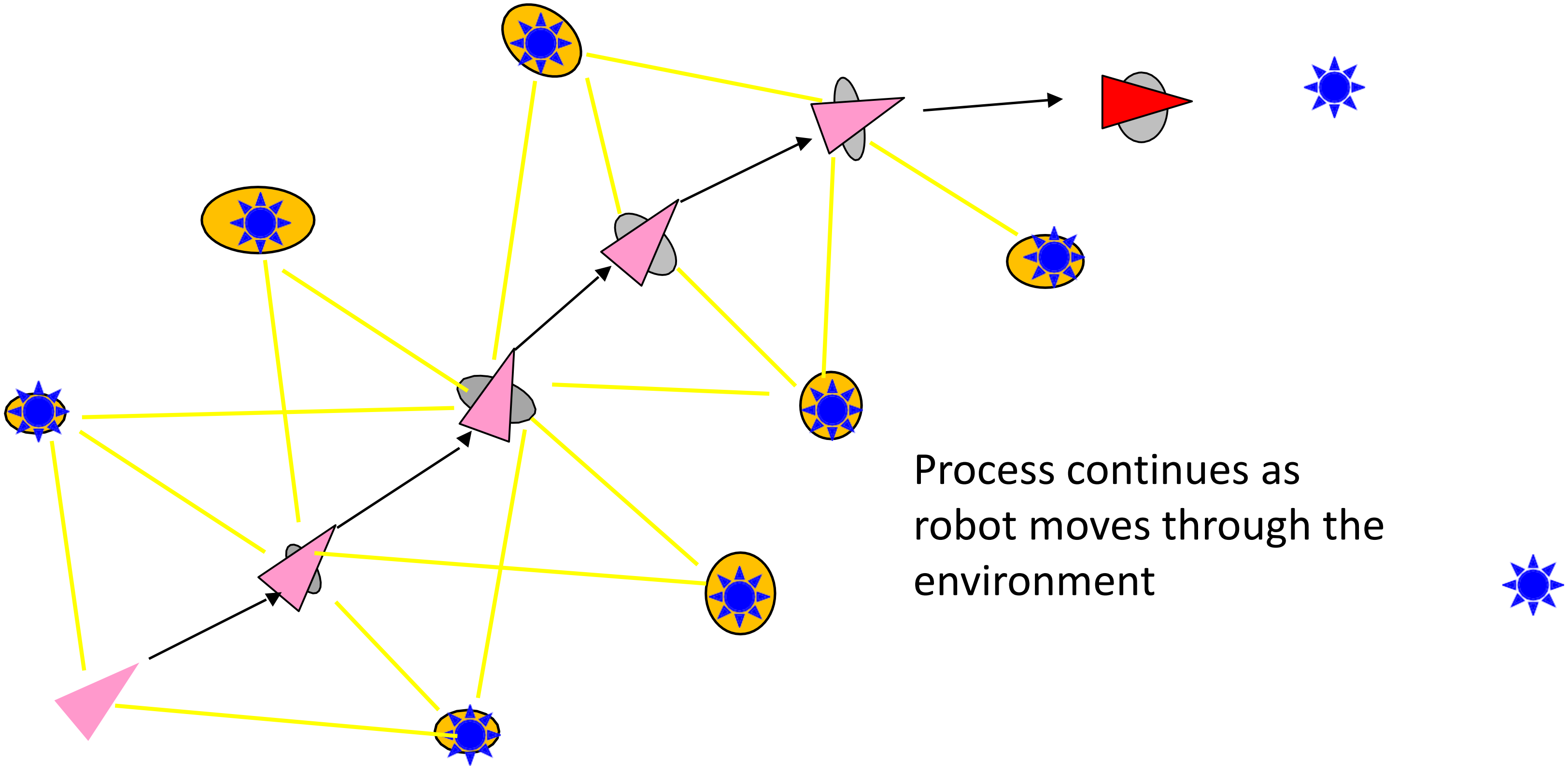
Robot observes another two new landmarks and adds them to the state

Example: SLAM With Landmarks



Robot also re-observes the first four features, resulting in uncertainty reduction for ALL state components

Example: SLAM With Landmarks



Process continues as robot moves through the environment

Transition Update: Mean

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

- State transition utilizes the full, possibly nonlinear, system model $\boldsymbol{f}$
- Map landmark predictions should remain unchanged if stationary
- Ex:

$$\boldsymbol{f}\left(\widehat{\boldsymbol{x}}_{k-1|k-1}, \boldsymbol{u}_k\right) = \begin{pmatrix} \hat{x} - \frac{v_k}{\omega_k}\sin\hat{\theta} + \frac{v_k}{\omega_k}\sin(\hat{\theta} + \omega_k \Delta t) \\ \hat{y} + \frac{v_k}{\omega_k}\cos\hat{\theta} - \frac{v_k}{\omega_k}\cos(\hat{\theta} + \omega_k \Delta t) \\ \hat{\theta} + \omega_k \Delta t \\ \boldsymbol{m}_{k-1} \end{pmatrix}$$

Transition Update: Covariance

$$\boldsymbol{P}_{xx,k|k-1} = \nabla \boldsymbol{f} \boldsymbol{P}_{xx,k-1|k-1} \nabla \boldsymbol{f}^T + \boldsymbol{Q}_k$$

- As with the mean, covariance (uncertainty estimates) of the landmarks should not change when the robot moves

$$\nabla \boldsymbol{f} = \begin{pmatrix} \boldsymbol{F}_k & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{I} \end{pmatrix}$$

- Ex:

$$\boldsymbol{F}_k = \frac{\partial}{\partial(\hat{x}, \hat{y}, \hat{\theta})} \begin{pmatrix} \hat{x} - \frac{v_k}{\omega_k} \sin \hat{\theta} + \frac{v_k}{\omega_k} \sin(\hat{\theta} + \omega_k \Delta t) \\ \hat{y} + \frac{v_k}{\omega_k} \cos \hat{\theta} - \frac{v_k}{\omega_k} \cos(\hat{\theta} + \omega_k \Delta t) \\ \hat{\theta} + \omega_k \Delta t \end{pmatrix} = \begin{pmatrix} 1 & 0 & -\frac{v_k}{\omega_k} \cos \hat{\theta} + \frac{v_k}{\omega_k} cos(\hat{\theta} + \omega_k \Delta t) \\ 0 & 1 & -\frac{v_k}{\omega_k} \sin \hat{\theta} + \frac{v_k}{\omega_k} \sin(\hat{\theta} + \omega_k \Delta t) \\ 0 & 0 & 1 \end{pmatrix}$$

Measurement Update

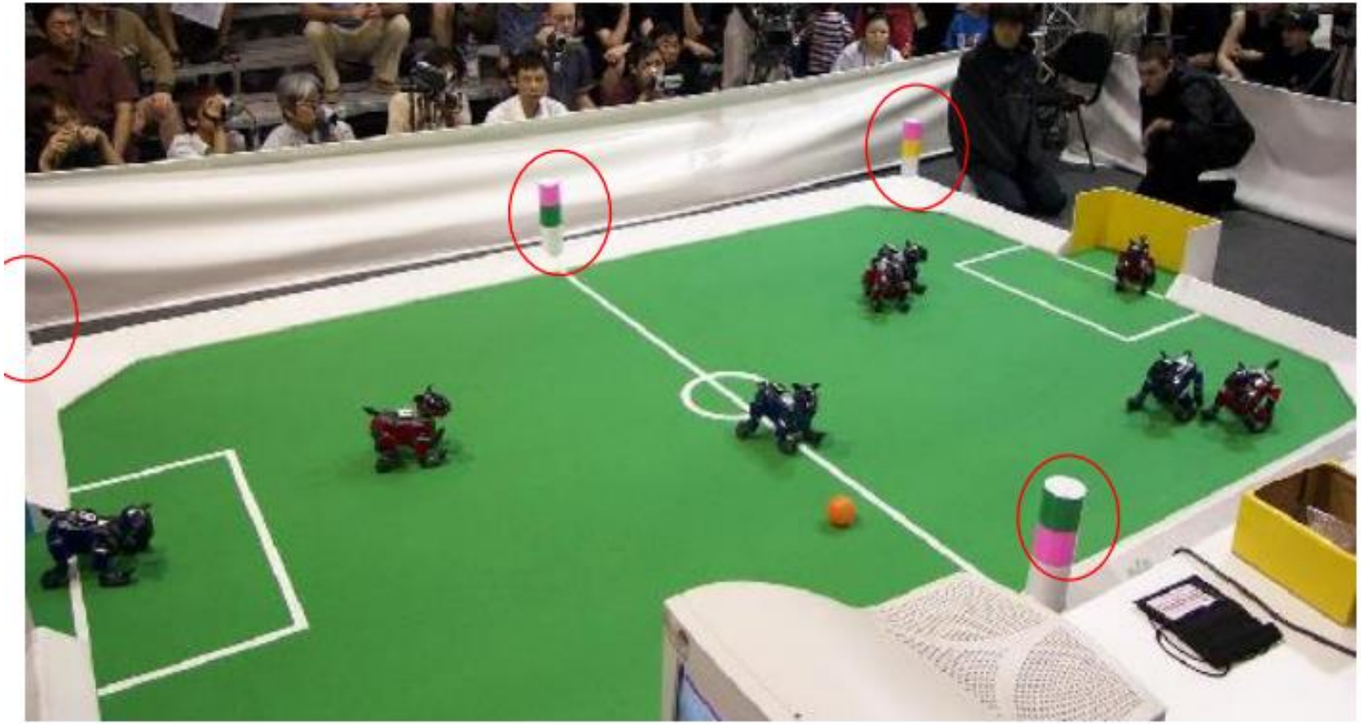
$$\boldsymbol{K}_k = \boldsymbol{P}_{k|k-1}\nabla\boldsymbol{h}^T\left(\nabla\boldsymbol{h}\boldsymbol{P}_{k|k-1}\nabla\boldsymbol{h}^T + \boldsymbol{R}_k\right)^{-1}$$

- Next step is to compute the Kalman to determine strength of updates
- We'll require the observation model here (or rather, its Jacobian)
- Example: Range-bearing measurement of landmark *j*

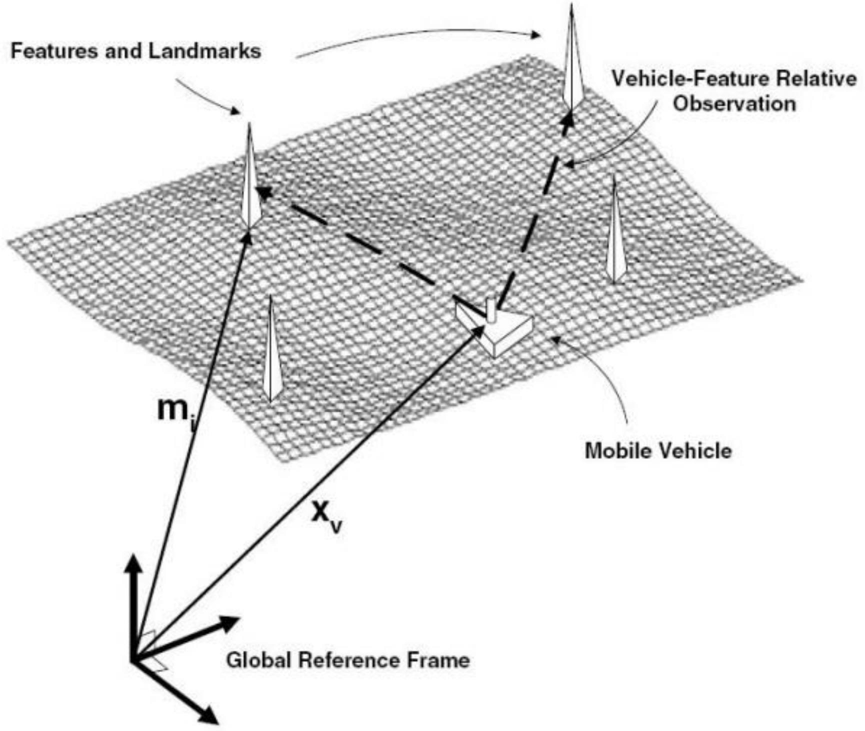
$$\boldsymbol{h}\left(\widehat{\boldsymbol{x}}_{k|k-1}, \widehat{\boldsymbol{m}}_{k-1}\right) = \begin{pmatrix} r_k \\ \phi_k \end{pmatrix} = \begin{pmatrix} \sqrt{\left(\widehat{m}_{jx} - \hat{x}\right)^2 + \left(\widehat{m}_{jy} - \hat{y}\right)^2} \\ \mathrm{Atan2}\left(\widehat{m}_{jy} - \hat{y}, \widehat{m}_{jx} - \hat{x}\right) - \hat{\theta} \end{pmatrix}$$

- Jacobian components of all other landmarks should be 0!

Landmark Observations



Landmark Observations



Observation Jacobian

$$\boldsymbol{K}_k = \boldsymbol{P}_{k|k-1}\nabla\boldsymbol{h}^T\left(\nabla\boldsymbol{h}\boldsymbol{P}_{k|k-1}\nabla\boldsymbol{h}^T + \boldsymbol{R}_k\right)^{-1}$$

$$\boldsymbol{h}\left(\widehat{\boldsymbol{x}}_{k|k-1}, \widehat{\boldsymbol{m}}_{k-1}\right) = \begin{pmatrix} r_k \\ \phi_k \end{pmatrix} = \begin{pmatrix} \sqrt{\left(\widehat{m}_{jx} - \hat{x}\right)^2 + \left(\widehat{m}_{jy} - \hat{y}\right)^2} \\ \mathrm{Atan2}\left(\widehat{m}_{jy} - \hat{y}, \widehat{m}_{jx} - \hat{x}\right) - \hat{\theta} \end{pmatrix}$$

$$\nabla\boldsymbol{h} = \frac{1}{r_k^2}\left(\underbrace{\begin{matrix} -\left(\widehat{m}_{jx} - \hat{x}\right)r_k & -\left(\widehat{m}_{jy} - \hat{y}\right)r_k & 0 \\ \left(\widehat{m}_{jy} - \hat{y}\right) & -\left(\widehat{m}_{jx} - \hat{x}\right) & -r_k^2 \end{matrix}}_{\text{Jacobian of } \boldsymbol{h} \text{ wrt robot state } \widehat{\boldsymbol{x}}} \quad \underbrace{\begin{matrix} 0 & \cdots & 0 \\ 0 & \cdots & 0 \end{matrix}}_{2(j-1)} \quad \underbrace{\begin{matrix} \left(\widehat{m}_{jx} - \hat{x}\right)r_k & \left(\widehat{m}_{jy} - \hat{y}\right)r_k \\ -\left(\widehat{m}_{jy} - \hat{y}\right) & \left(\widehat{m}_{jx} - \hat{x}\right) \end{matrix}}_{\text{Jacobian of } \boldsymbol{h} \text{ wrt } j \text{ landmark, } \widehat{m}_{jx}, \widehat{m}_{jy}} \quad \underbrace{\begin{matrix} 0 & \cdots & 0 \\ 0 & \cdots & 0 \end{matrix}}_{2(n-j)}\right)$$

Adding New Landmarks

- What if we observe a new landmark instead of one that is already known?
- Invert the observation function and append to the state:

$$\boldsymbol{h}^{-1}(\hat{\boldsymbol{x}}_{k|k-1}, \boldsymbol{z}_k) = \begin{pmatrix} \hat{m}_{jx} \\ \hat{m}_{jy} \end{pmatrix} = \begin{pmatrix} \hat{x} + r_k \cos(\hat{\theta} + \phi_k) \\ \hat{y} + r_k \sin(\hat{\theta} + \phi_k) \end{pmatrix} \qquad \hat{\boldsymbol{x}} \leftarrow \begin{pmatrix} \hat{\boldsymbol{x}} \\ \hat{m}_{jx} \\ \hat{m}_{jy} \end{pmatrix}$$

- What about the covariance? In theory we can compute Jacobians of the observation function to obtain correlations between new landmark and current state covariances (robot state + landmarks)
- Simpler approach: Assign an estimate (e.g. observation covariance $\boldsymbol{R}_k$) with 0 cross-correlations, and covariances will hopefully converge to true values later on

EKF SLAM Algorithm

- Transition model: $\boldsymbol{x}_k = \boldsymbol{f}(\boldsymbol{x}_{k-1}, \boldsymbol{u}_k) + \boldsymbol{w}_k \qquad \boldsymbol{w}_k \sim N(0, \boldsymbol{Q}_k)$
- Observation model: $\boldsymbol{z}_k = \boldsymbol{h}(\boldsymbol{x}_k, \boldsymbol{m}) + \boldsymbol{v}_k \qquad \boldsymbol{v}_k \sim N(0, \boldsymbol{R}_k)$
- Given: $\widehat{\boldsymbol{x}}_{k-1|k-1}$, $\boldsymbol{P}_{k-1|k-1}$, $\boldsymbol{u}_k$, $\boldsymbol{z}_k$
- Transition update: $\widehat{\boldsymbol{x}}_{k|k-1} = \boldsymbol{f}\left(\widehat{\boldsymbol{x}}_{k-1|k-1}, \boldsymbol{u}_k\right)$

$$\boldsymbol{P}_{xx,k|k-1} = \nabla \boldsymbol{f} \boldsymbol{P}_{xx,k-1|k-1} \nabla \boldsymbol{f}^T + \boldsymbol{Q}_k$$

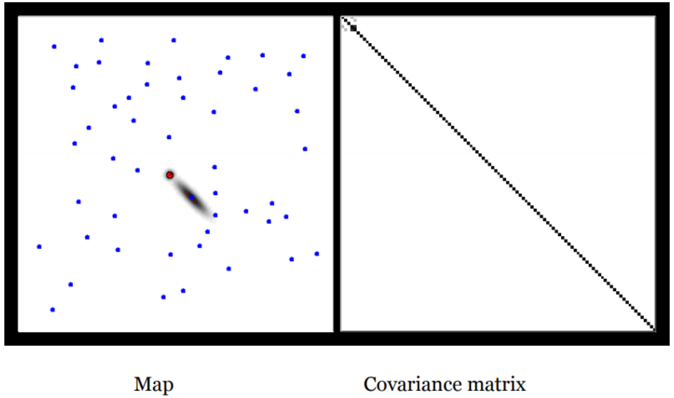
- Observation update: If landmark is new, add to state; otherwise:

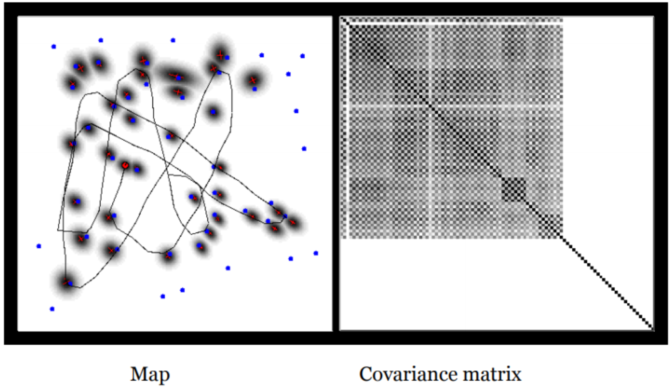
$$\boldsymbol{K}_k = \boldsymbol{P}_{k|k-1} \nabla \boldsymbol{h}^T \left(\nabla \boldsymbol{h} \boldsymbol{P}_{k|k-1} \nabla \boldsymbol{h}^T + \boldsymbol{R}_k\right)^{-1}$$

$$\begin{pmatrix} \widehat{\boldsymbol{x}}_{k|k} \\ \widehat{\boldsymbol{m}}_k \end{pmatrix} = \begin{pmatrix} \widehat{\boldsymbol{x}}_{k|k-1} \\ \widehat{\boldsymbol{m}}_{k-1} \end{pmatrix} + \boldsymbol{K}_k(\boldsymbol{z}_k - \boldsymbol{h}(\widehat{\boldsymbol{x}}_{k|k-1}, \widehat{\boldsymbol{m}}_{k-1})) \qquad \boldsymbol{P}_{k|k} = (\boldsymbol{I} - \boldsymbol{K}_k \nabla \boldsymbol{h}) \boldsymbol{P}_{k|k-1}$$

EKF SLAM Considerations

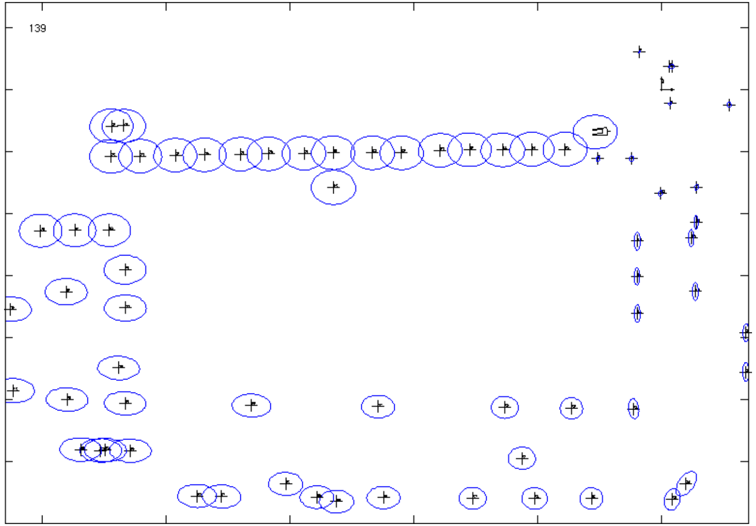
- Computational complexity increases quadratically in size of the map
- Highly nonlinear models may diverge or have suboptimal performance
- Convergence of EKF SLAM (if linear approximation is good): landmark estimates become fully correlated in the limit of infinitely many observations

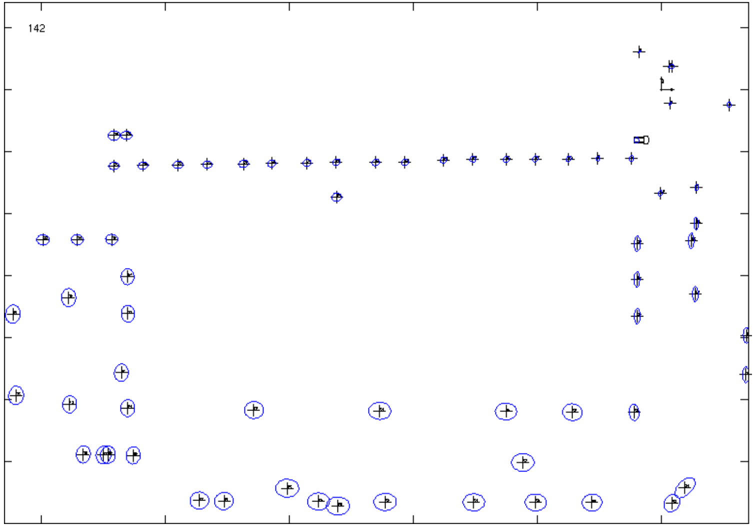




Loop Closure

- **Loop closure**: Observing a landmark that has been observed already
- If the robot *believes* it has closed a loop, then uncertainties of landmarks it observed before will tend to collapse very quickly





Data Association Problem

- Success of EKF SLAM depends on robustness of data association, or ability to recognize an observed landmark (or a new one)
- Loop closure helps reduce uncertainties but can be disastrous if incorrect

- For best performance, landmarks should be distinct and spaced out
- Gives greater resilience to both motion and measurement noise

- If multiple possible candidates, maximum likelihood (ML) methods can help pick the best landmark, but most methods only rely one such choice
- An incorrect choice due to ambiguity can lead to filter divergence!

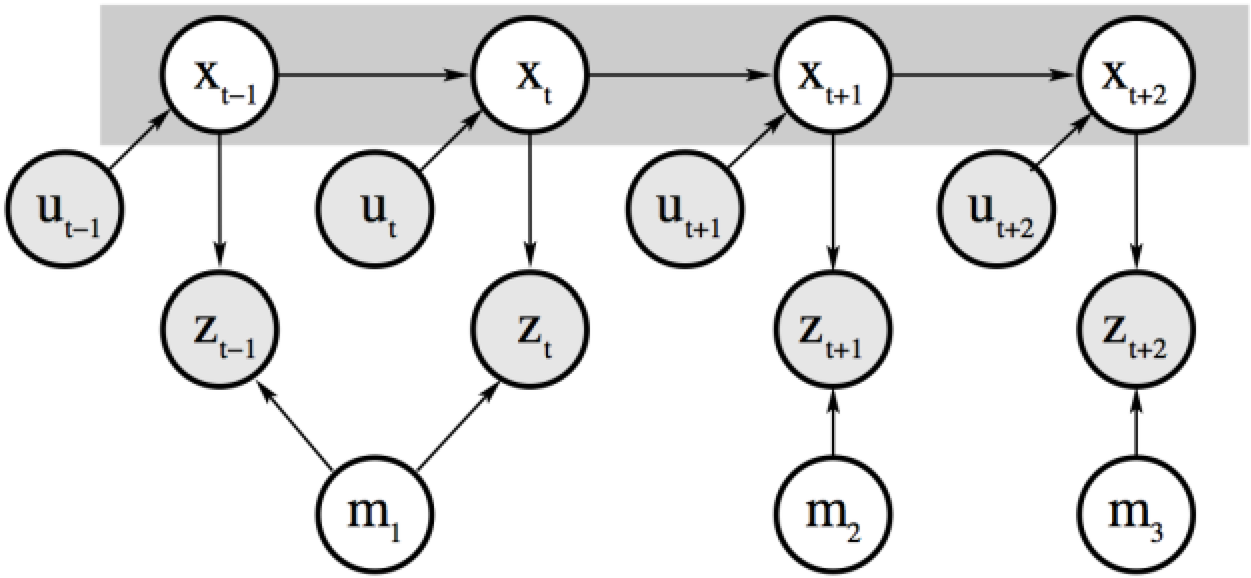
Particle Filters for SLAM

- It would be better if we can maintain and update beliefs for all relevant landmarks simultaneously, maybe weighted by observation likelihoods
- A task well-suited for particle filters!

- Problem: How to represent distributions without combinatorial explosion?
- Naively, each particle would have to represent joint distribution between robot state and all landmarks

- Observation: Although all landmarks are correlated, we can consider them independent *conditioned on* robot's state!

Conditional Independence of Landmarks



Rao-Blackwellized Particle Filter

- Let's factor our joint belief distribution:

$$B(\boldsymbol{x}_{1:t}, \boldsymbol{m}_{1:M}) = p(\boldsymbol{x}_{1:t}, \boldsymbol{m}_{1:M} | \boldsymbol{u}_{1:t}, \boldsymbol{z}_{1:t})$$

$$= p(\boldsymbol{x}_{1:t} | \boldsymbol{u}_{1:t}, \boldsymbol{z}_{1:t}) p(\boldsymbol{m}_{1:M} | \boldsymbol{x}_{1:t}, \boldsymbol{z}_{1:t})$$

$$= p(\boldsymbol{x}_{1:t} | \boldsymbol{u}_{1:t}, \boldsymbol{z}_{1:t}) \prod_{i=1}^{M} p(\boldsymbol{m}_i | \boldsymbol{x}_{1:t}, \boldsymbol{z}_{1:t})$$

State belief distribution can be maintained and updated using regular particle filter as before

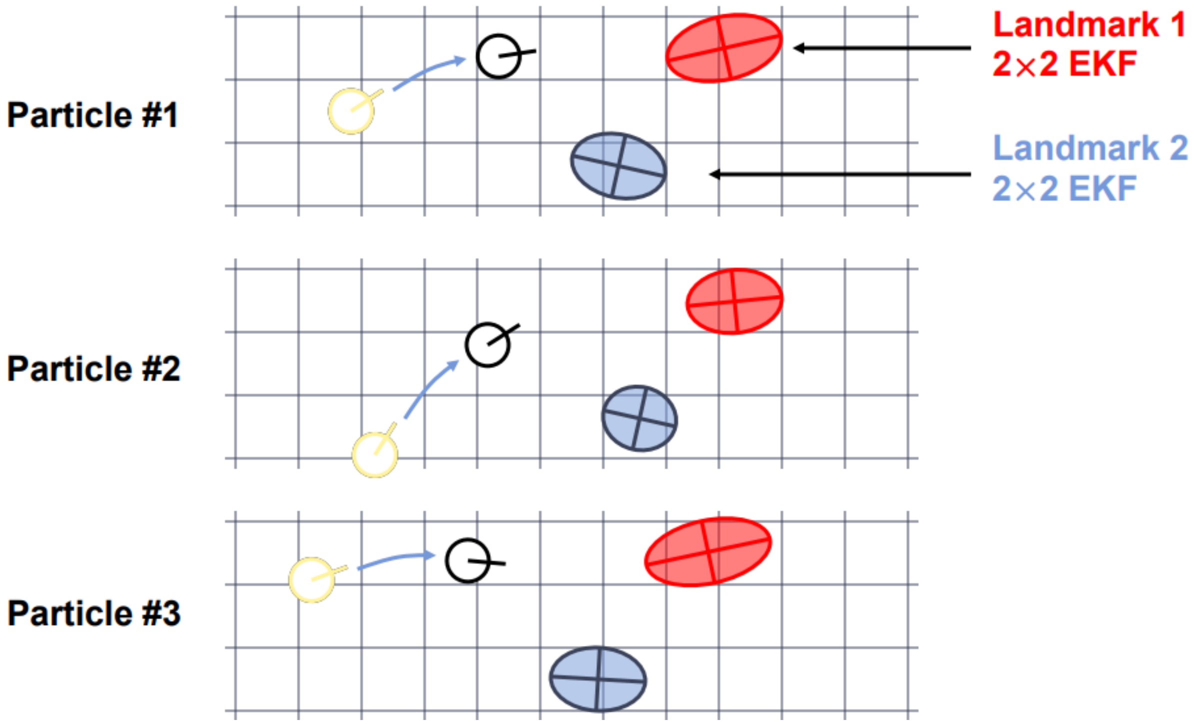
Each particle maintains M EKFS, one for each landmark, and updates based on EKF process (we don't need to sample these!)

FastSLAM

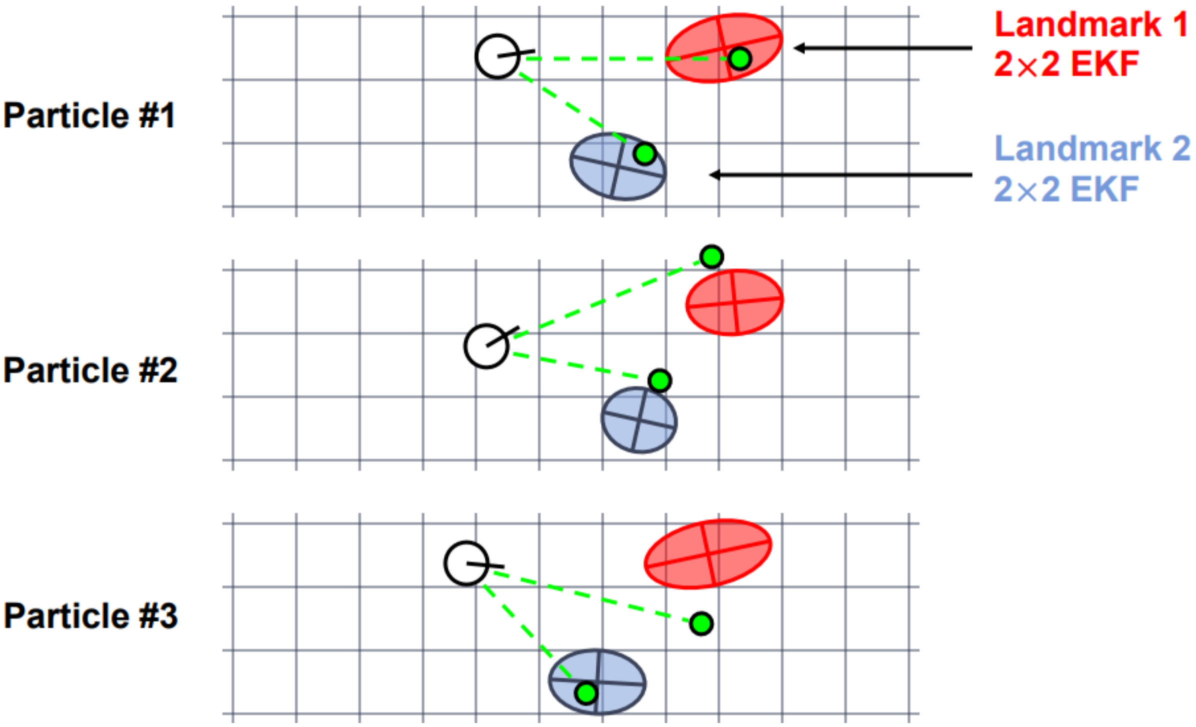
- Each particle contains the following information
  - Robot state (assumed to be *correct* from perspective of landmarks)
  - Small (2 by 2), independent EKFs for each landmark conditioned on robot state
  - Particle weight

Particle 1	x, y, θ	**Landmark 1**	**Landmark 2**	**...**	**Landmark M**
Particle 2	x, y, θ	**Landmark 1**	**Landmark 2**	**...**	**Landmark M**
Particle N	x, y, θ	**Landmark 1**	**Landmark 2**	**...**	**Landmark M**

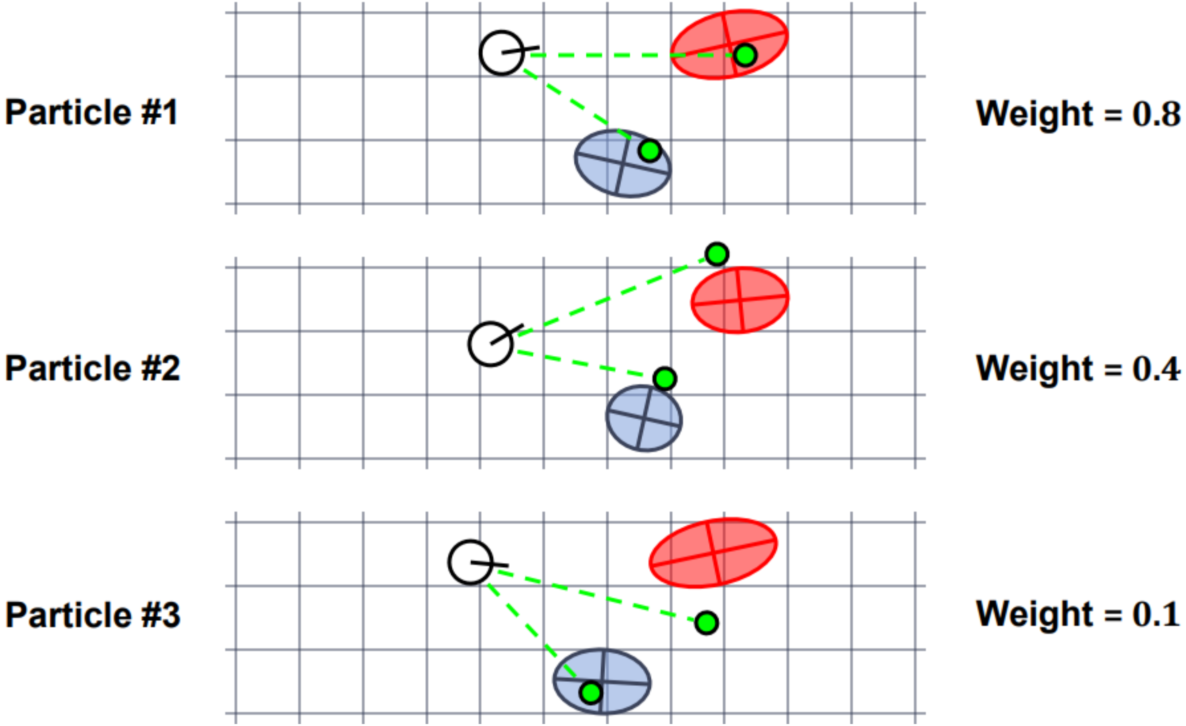
FastSLAM Example: Transition Update



FastSLAM Example: Measurement Update



FastSLAM Example: Weights and Resampling



FastSLAM Algorithm

- At any given step, we start with a set of particles as described previously

- For each particle:
  - Sample from transition model (move it): $p(\boldsymbol{x}_k|\boldsymbol{x}_{k-1},\boldsymbol{u}_k)$
  - Perform EKF observation update for observed landmark j:

$$\boldsymbol{K} = \boldsymbol{P}_{j,k-1}\nabla\boldsymbol{h}^T\left(\nabla\boldsymbol{h}\boldsymbol{P}_{j,k-1}\nabla\boldsymbol{h}^T + \boldsymbol{R}_j\right)^{-1}$$

$$\widehat{\boldsymbol{m}}_{j,k} = \widehat{\boldsymbol{m}}_{j,k-1} + \boldsymbol{K}(\boldsymbol{z}_k - \boldsymbol{h}(\widehat{\boldsymbol{x}}_k, \widehat{\boldsymbol{m}}_{j,k-1}))$$
$$\boldsymbol{P}_{j|k} = (\boldsymbol{I} - \boldsymbol{K}\nabla\boldsymbol{h})\boldsymbol{P}_{j,k-1}$$

  - Compute a corresponding weight (more on this below)

- Resample each particle according to current weights

Particle Weights

- With regular particle filters, particle weights were just observation likelihoods

$$p(\boldsymbol{z_k}|\boldsymbol{x_k}, \widehat{\boldsymbol{m}})$$

- Since we have EKFs for landmarks, this likelihood can be computed analytically

- Covariance of **innovation**, or difference between true and predicted measurements, from the Kalman gain: $\boldsymbol{\Sigma}_k = \nabla \boldsymbol{h} \boldsymbol{P}_{j,k-1} \nabla \boldsymbol{h}^T + \boldsymbol{R}_j$

- Particle weight (from Gaussian properties of EKF):

$$w_k = \frac{1}{\sqrt{2\pi|\boldsymbol{\Sigma}_k|}} \exp\left(-\frac{1}{2}(\boldsymbol{z}_k - \hat{\boldsymbol{z}}_k)^T \boldsymbol{\Sigma}_k^{-1} (\boldsymbol{z}_k - \hat{\boldsymbol{z}}_k)\right)$$

FastSLAM Considerations

- Conditional independence assumption allows us to sample only the robot's evolving state (nonlinear) while exploiting Gaussian nature of map beliefs

- As with regular particle filter, performance scales with number of particles
- Robust to data association ambiguities, can tolerate more nonlinearities

- Difficulties and problems of vanilla particle filters are still present
- Particle deprivation, over-dependence on measurement history, etc.