

Robotica Probabilistica

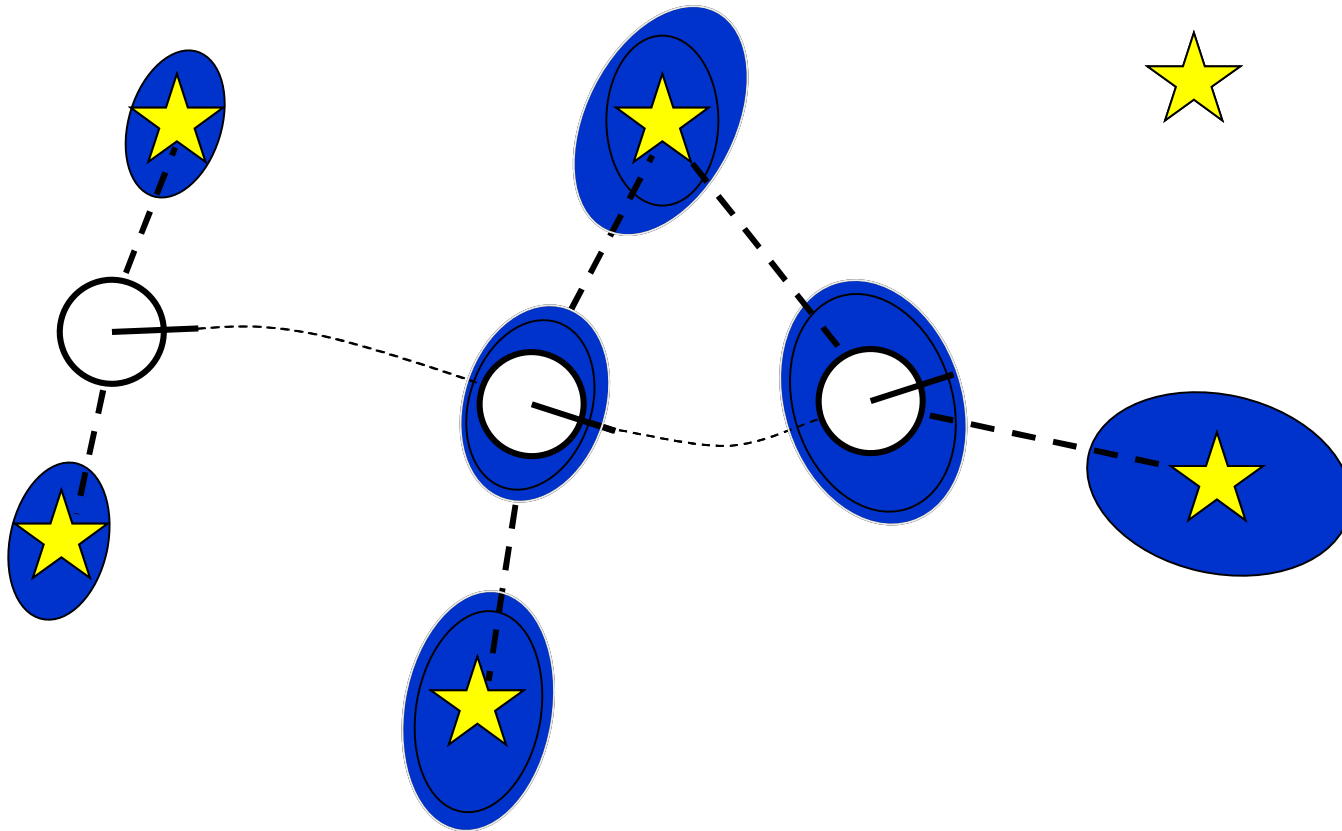
FastSLAM

SLAM con particle filters

- SLAM con particle filters
- Problema della dimensione: molte partices e l'algoritmo scala in modo esponenziale
- Si può utilizzare però una fattorizzazione: filtri Rao-Blackwellized: se un oracolo ci dice il path, la stima della locazione delle features è indipendente.
- FastSLAM: Particelle per la posa, Gaussiane per le features
- Mantiene più ipotesi sulle associazioni di dati (più robusto); non linearità; full SLAM & online SLAM trattati in modo uniforme.

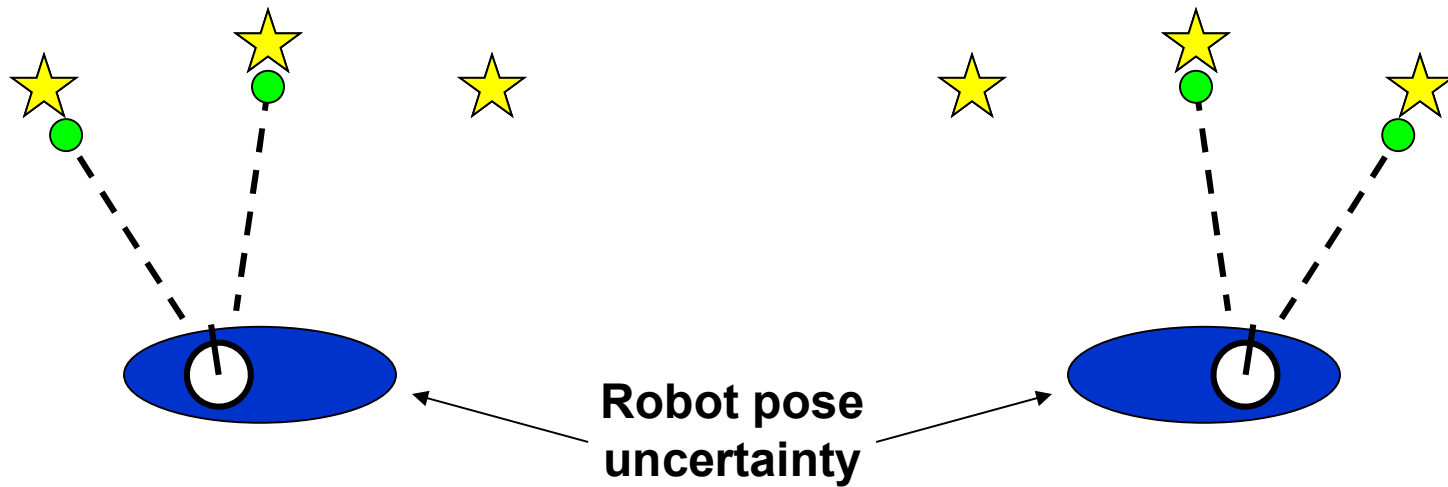
Why is SLAM a hard problem?

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



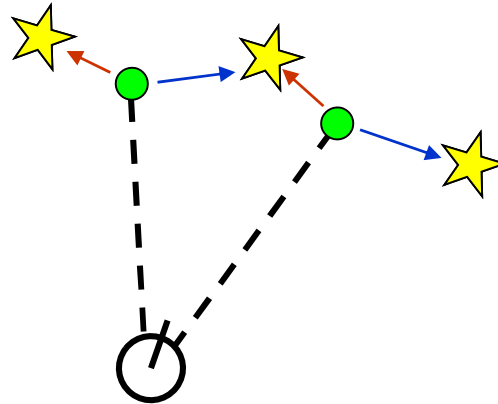
Robot path error correlates errors in the map

Why is SLAM a hard problem?



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

Data Association Problem



- A data association is an assignment of observations to landmarks
- In general there are more than $\binom{n}{m}$ (n observations, m landmarks) possible associations
- Also called “assignment problem”

Filtri Particellari

- Rappresentano belief con random **samples**
- Stima di processi **non-Gaussiani, nonlinear**
- Principio di Sampling Importance Resampling (SIR):
 - Estrai la nuova generazione di particelle
 - Assegna un peso di importanza ad ogni particle
 - Resampling
- Applicazioni tipiche sono: tracking, localization, etc.

Localizzazione vs. SLAM

- Un particle filter può essere usato per risolvere entrambi i problemi
- Localizzazione: spazio di stato $\langle x, y, \theta \rangle$
- SLAM: spazio di stato $\langle x, y, \theta, map \rangle$
 - per mappe di landmark = $\langle l_1, l_2, \dots, l_m \rangle$
 - Per mappe a grid = $\langle c_{11}, c_{12}, \dots, c_{1n}, c_{21}, \dots, c_{nm} \rangle$
- **Problema:** il numero di particelle necessario per rappresentare il posterior cresce esponenzialmente con la dimensione dello spazio di stato

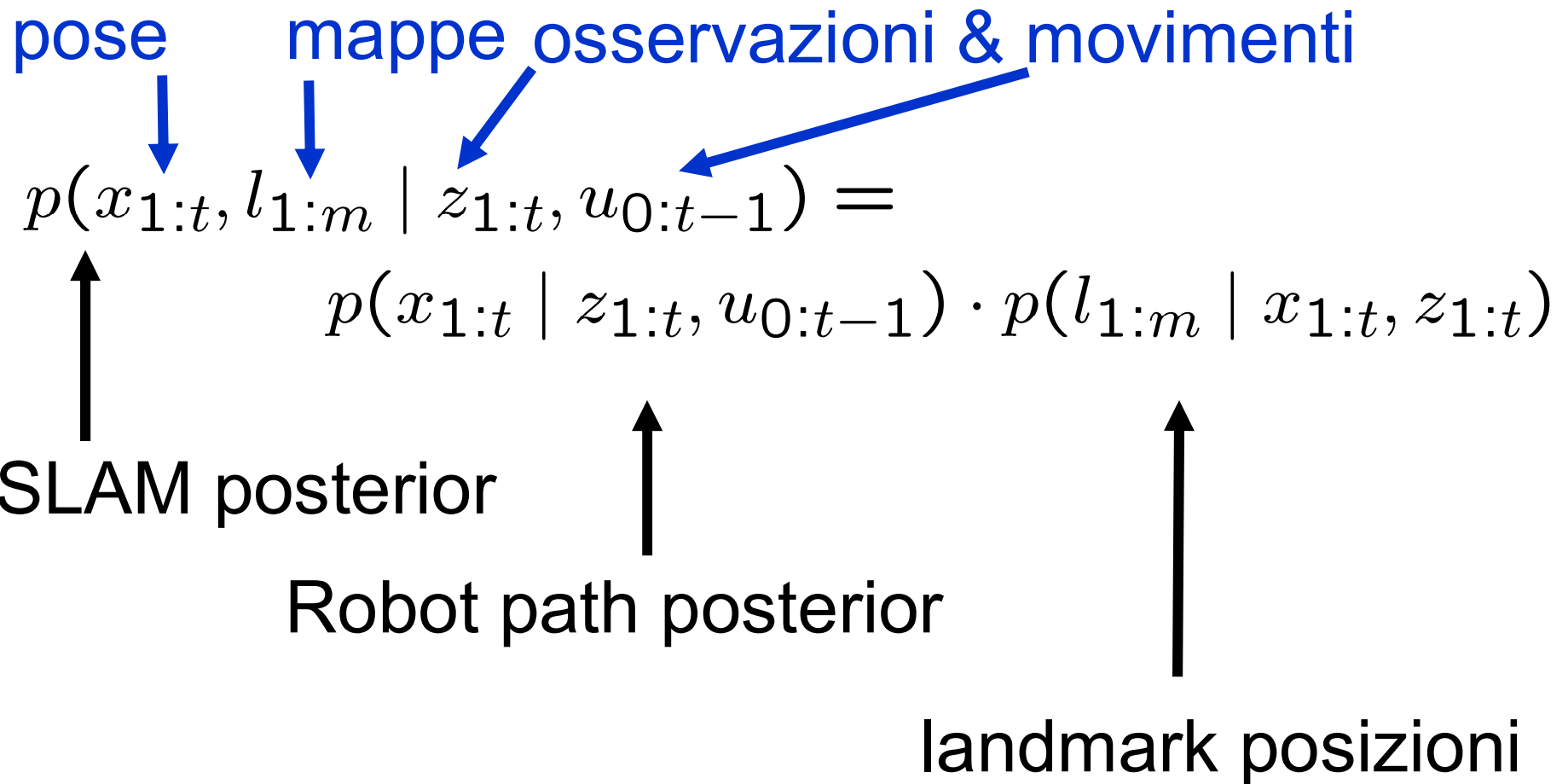
Dipendenze

- Esiste una dipendenza probabilistica tra le dimensioni dello spazio di stato (posa, mappa)?
- Se è così, possiamo usare le dipendenze per risolvere il problema in modo più efficiente?

Dipendenze

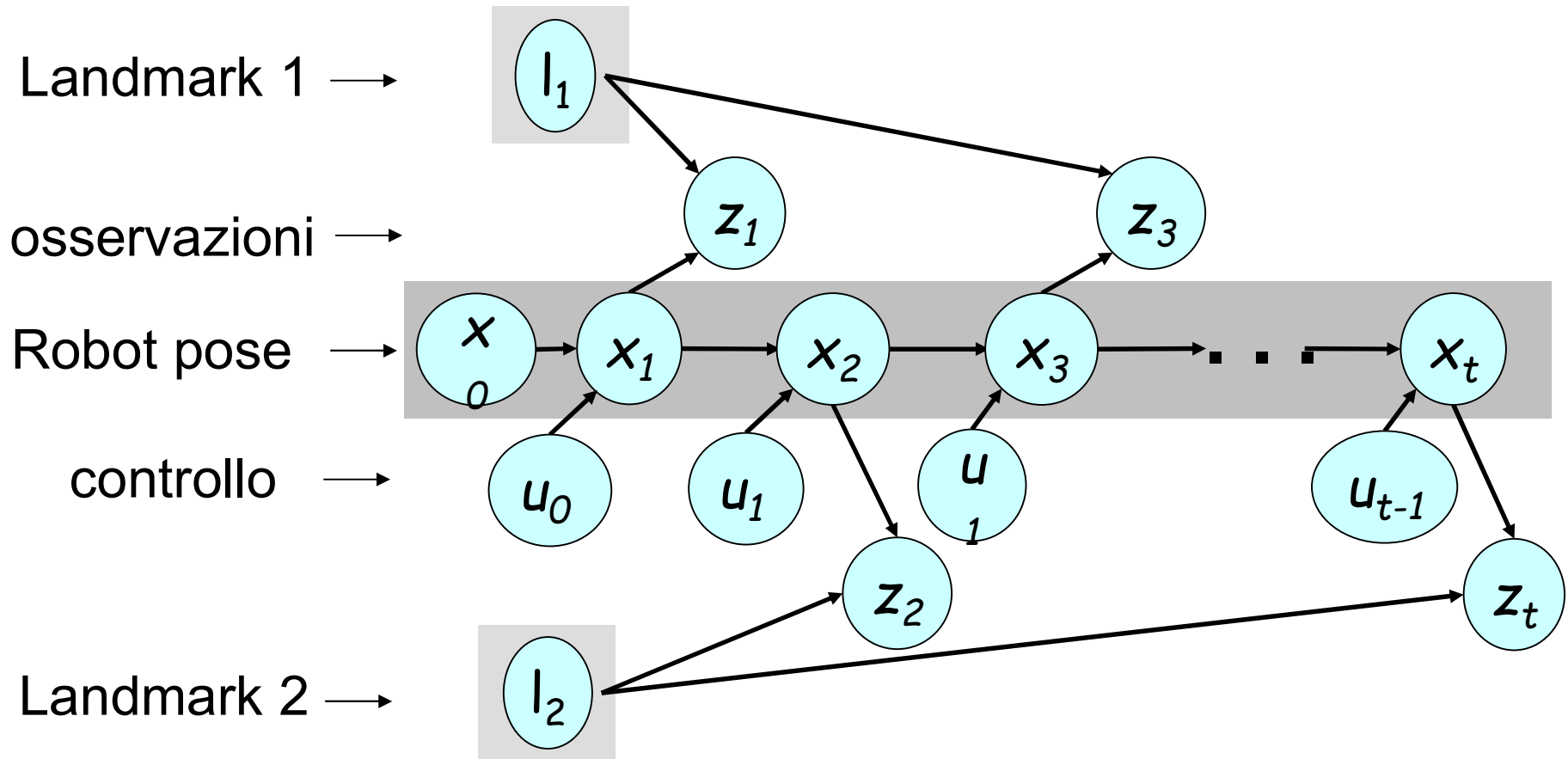
- Esiste una dipendenza tra le dimensioni dello spazio di stato?
- Se è così, possiamo usare le dipendenze per risolvere il problema più efficientemente?
- Nel contesto dello SLAM:
 - La mappa dipende dalle pose del robot;
 - Noi sappiamo come costruire una mappa data la posizione nota del sensore.

Posteriore Fattorizzato (Landmark)



Aiuta a risolvere il problema?

Mapping con Landmarks



La conoscenza del percorso reale rende le posizioni dei landmark cond. indipendenti

Posterior Fattorizzato

$$\begin{aligned} & p(x_{1:t}, l_{1:m} \mid z_{1:t}, u_{0:t-1}) \\ &= p(x_{1:t} \mid z_{1:t}, u_{0:t-1}) \cdot p(l_{1:m} \mid x_{1:t}, z_{1:t}) \\ &= p(x_{1:t} \mid z_{1:t}, u_{0:t-1}) \cdot \prod_{i=1}^M p(l_i \mid x_{1:t}, z_{1:t}) \end{aligned}$$

Robot path posterior
(problema localizzazione)

Cond. indipendente
posizione landmark

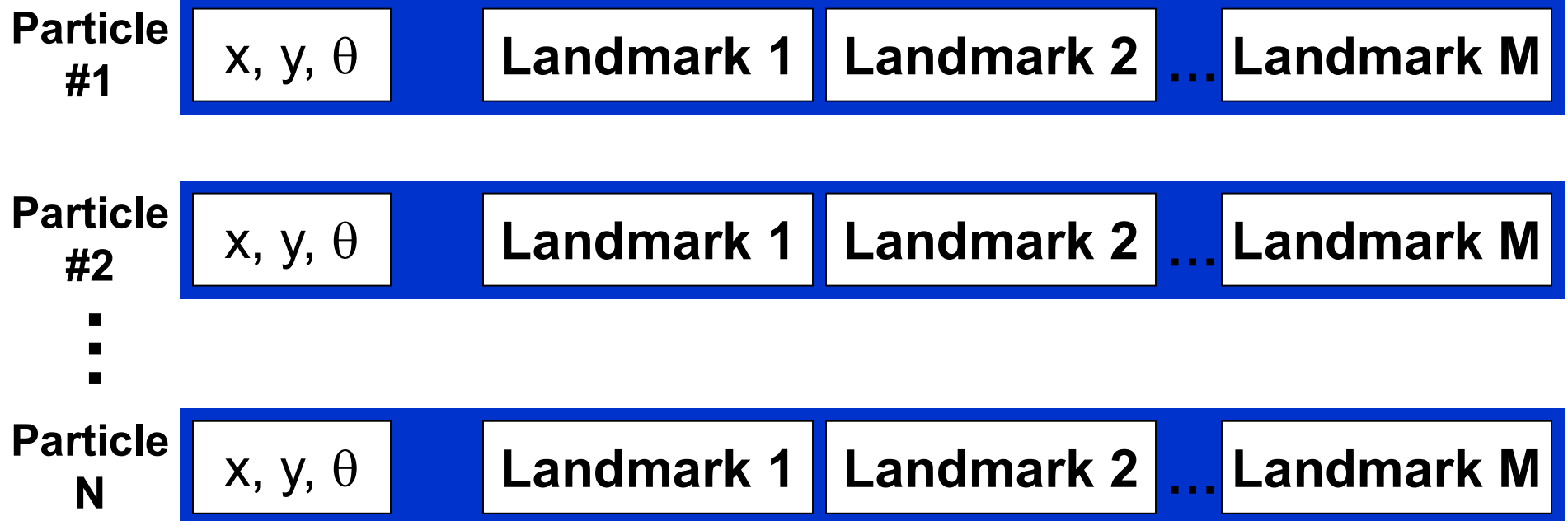
Rao-Blackwellization

$$p(x_{1:t}, l_{1:m} \mid z_{1:t}, u_{0:t-1}) = p(x_{1:t} \mid z_{1:t}, u_{0:t-1}) \cdot \prod_{i=1}^M p(l_i \mid x_{1:t}, z_{1:t})$$

- Questa fattorizzazione anche chiamata Rao-Blackwellization
- Dato che il secondo termine può essere calcolato efficientemente il particle filtering diventa possibile

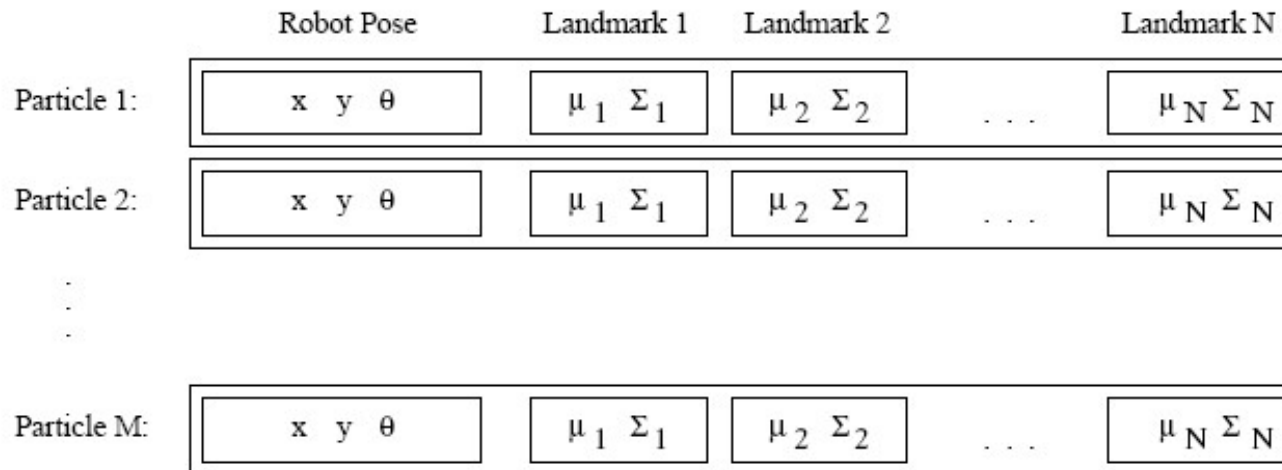
FastSLAM

- Rao-Blackwellized particle filtering basato su landmarks [Montemerlo et al., 2002]
- Ogni landmark rappresentato da un 2x2 Extended Kalman Filter (EKF)
- Ogni particella deve mantenere M EKFs



Fast SLAM

- Ogni particella mantiene N EKF



- Per M volte:
 - **Retrieval**: prendi una posa $x_{k,t-1}$
 - **Predizione**: campiona una posa da $p(x_{k,t} \mid x_{k,t-1}, u_t)$
 - **Update di misura**: per ogni misura i , trova la corrispondenza j , e aggiorna il corrispondente EKF
 - **Peso**: pesa la nuova particella w_k
- **Ricampionamento**: ricampiona le M particelle

Action Update

- Per ogni posa, campionamento dal modello probabilistico di moto:

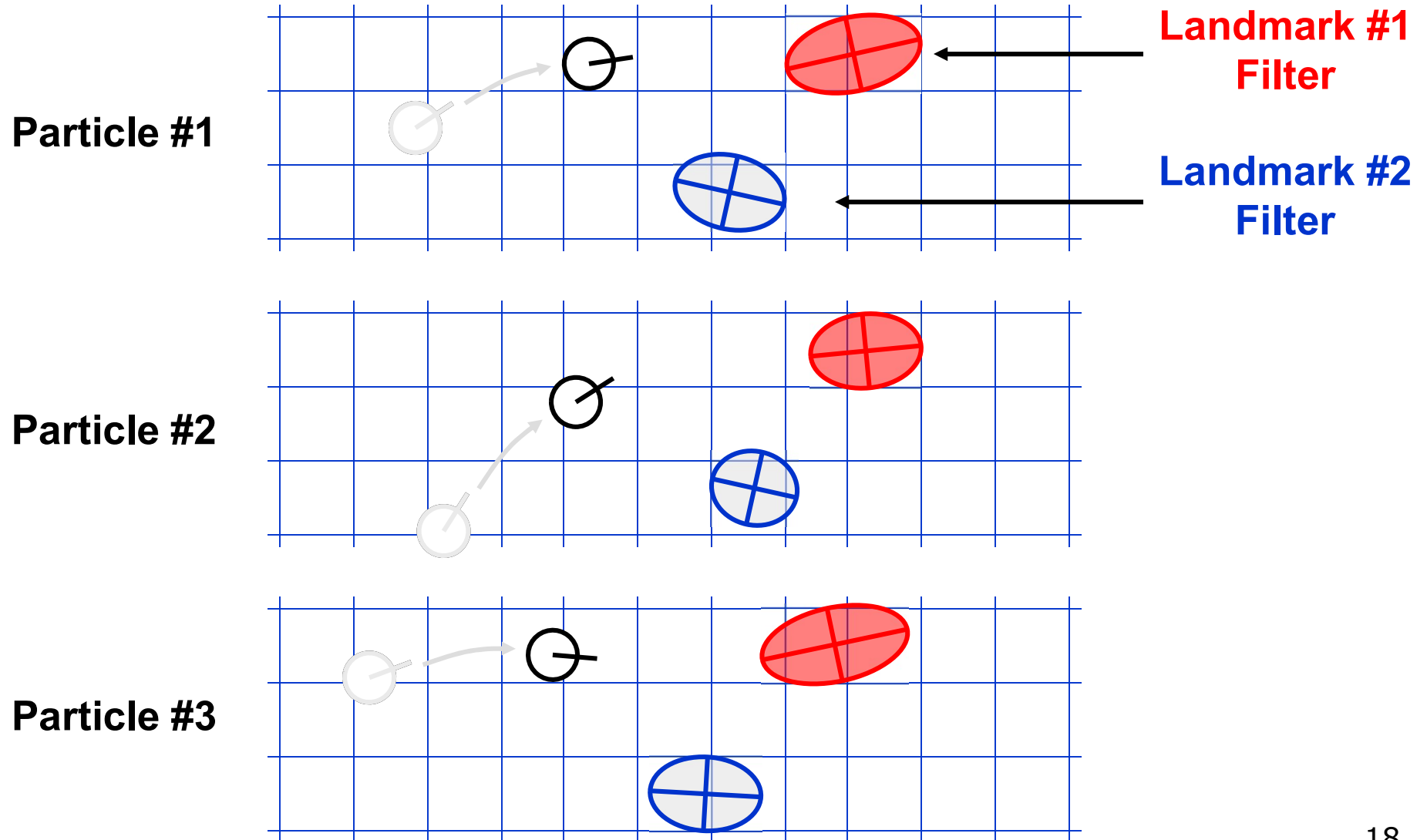
$$s_t^{[m]} \sim p(s_t | u_t, s_{t-1}^{[m]})$$

- Esempio, dal modello in velocità:

$$v_t' \sim N(v; v_t, \alpha_1 v_t + \alpha_2)$$
$$\omega_t' \sim N(\omega; \omega_t, \alpha_3 \omega_t + \alpha_4)$$

- Particella aggiunta in un insieme di particelle

FastSLAM – Action Update



FastSLAM – Sensor Update

Corrispondenze note n_1, \dots, n_t

Se feature n osservata, allora update

$$p(\theta_{n_t} | s^t, z^t, u^t, n^t) \stackrel{Markov}{=} \eta p(z_t | \theta_{n_t}, s_t, n_t) p(\theta_{n_t} | s^{t-1}, z^{t-1}, u^{t-1}, n^{t-1})$$

Se feature n non osservata, stessa probabilità

$$p(\theta_{n \neq n_t} | s^t, z^t, u^t, n^t) = p(\theta_{n \neq n_t} | s^{t-1}, z^{t-1}, u^{t-1}, n^{t-1})$$

Aggiornamento come EKF

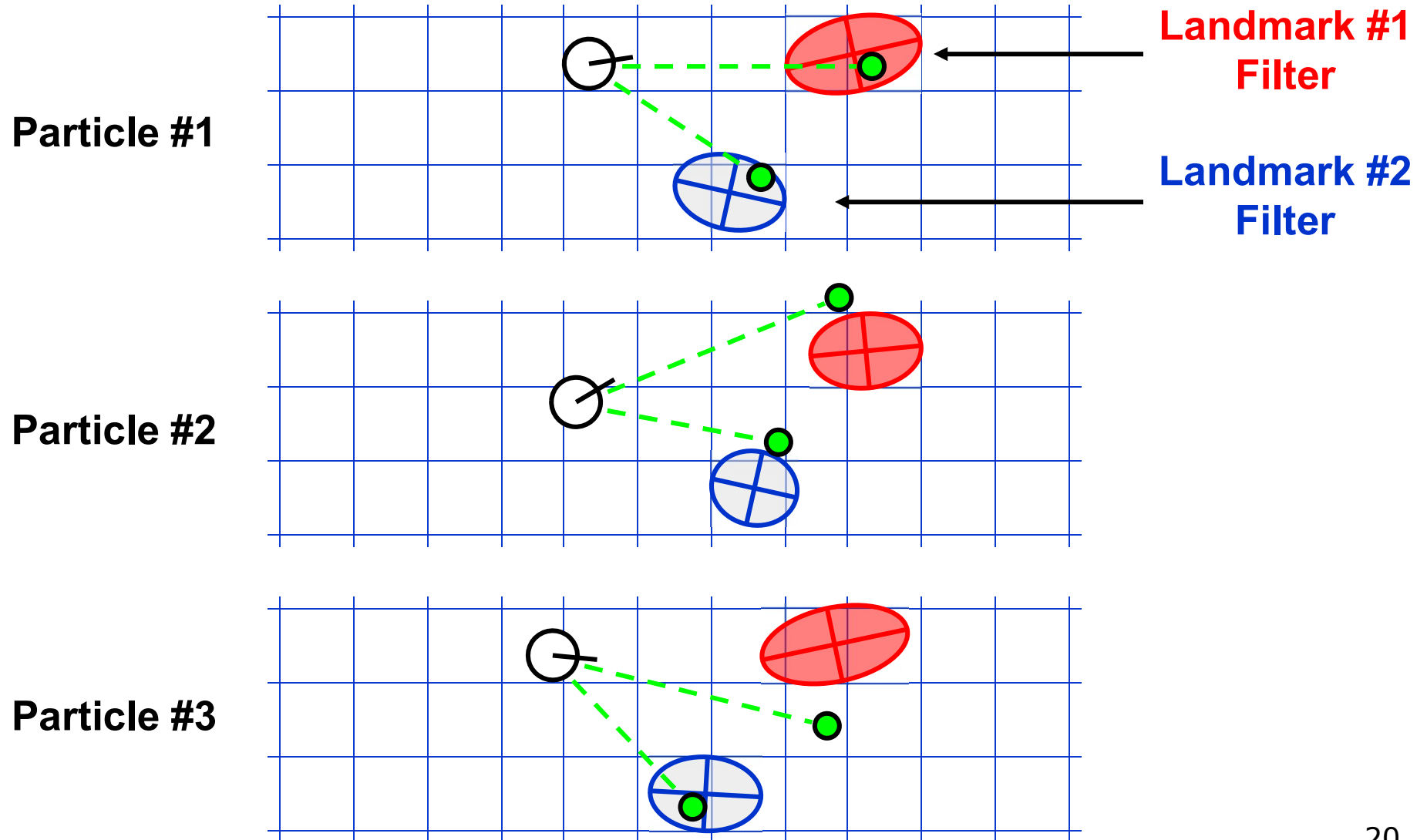
$$Z_{n,t} = G_{\theta_{n_t}} \Sigma_{n_t,t-1}^{[m]} G_{\theta_{n_t}}^T + R_t$$

$$K_t = \Sigma_{n_t,t-1}^{[m]} G_{\theta_{n_t}}^T Z_{n,t}^{-1}$$

$$\mu_{n_t,t}^{[m]} = \mu_{n_t,t-1}^{[m]} + K_t (z_t - \hat{z}_t)$$

$$\Sigma_{n_t,t}^{[m]} = (I - K_t G_{\theta_{n_t}}) \Sigma_{n_t,t-1}^{[m]}$$

FastSLAM – Sensor Update



FastSLAM - Resampling

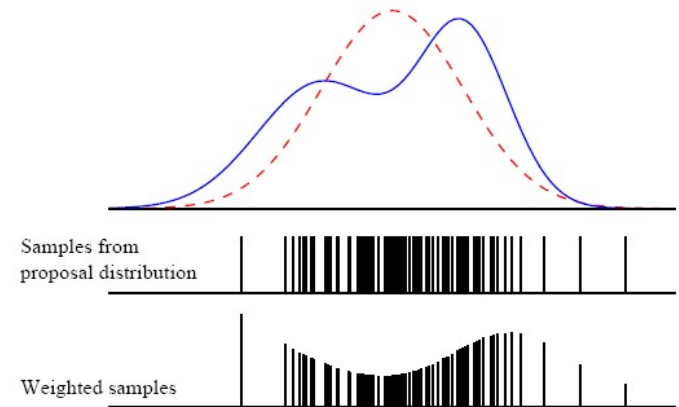
- Peso delle particelle e ricampionamento:
le particelle non sono distribuite secondo il posterior, non è inclusa la misura
- Occorre il peso (considerando le corrispondenze):

$$w_t^{[m]} = \frac{\text{target distribution}}{\text{proposal distribution}} = \frac{p(s^{t,[m]} | z^t, u^t, n^t)}{p(s^{t,[m]} | z^{t-1}, u^t, n^{t-1})}$$

$$\begin{aligned} w_t^{[m]} &\stackrel{\text{Markov}}{=} \frac{p(z_t | s^{t,[m]}, z^{t-1}, u^t, n^t) p(s^{t,[m]} | z^{t-1}, u^t, n^{t-1})}{p(s^{t,[m]} | z^{t-1}, u^t, n^{t-1})} \\ &= p(z_t | s^{t,[m]}, z^{t-1}, u^t, n^t) \end{aligned}$$

$$w_t^{[m]} = \frac{1}{\sqrt{|2\pi Z_{n_t,t}|}} \exp\left\{-\frac{1}{2}(z_t - \hat{z}_{n_t,t})^T [Z_{n_t,t}]^{-1} (z_t - \hat{z}_{n_t,t})\right\}$$

measurement covariance



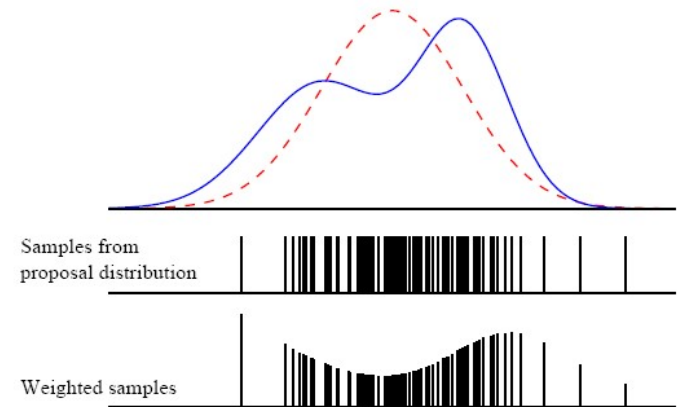
Innovazione: valore osservato - valore stimato

FastSLAM - Resampling

- Peso delle particelle e ricampionamento:
le particelle non sono distribuite secondo il posterior, non è inclusa la misura
- Occorre il peso (considerando le corrispondenze):

$$\begin{aligned}
 w^{[k]} &= \eta p(z_t | x_{1:t}^{[k]}, z_{1:t-1}) \\
 &= \eta \int p(z_t | x_{1:t}^{[k]}, z_{1:t-1}, m_j) p(m_j | x_{1:t}^{[k]}, z_{1:t-1}) dm_j \\
 &= \eta \int \underbrace{p(z_t | x_t^{[k]}, m_j)}_{\mathcal{N}(z_t; \hat{z}^{[k]}, Q_t)} \underbrace{p(m_j | x_{1:t-1}^{[k]}, z_{1:t-1})}_{\mathcal{N}(m_j; \mu_{j,t-1}^{[k]}, \Sigma_{j,t-1}^{[k]})} dm_j
 \end{aligned}$$

measurement covariance



Innovazione: valore osservato - valore stimato

FastSLAM - Resampling

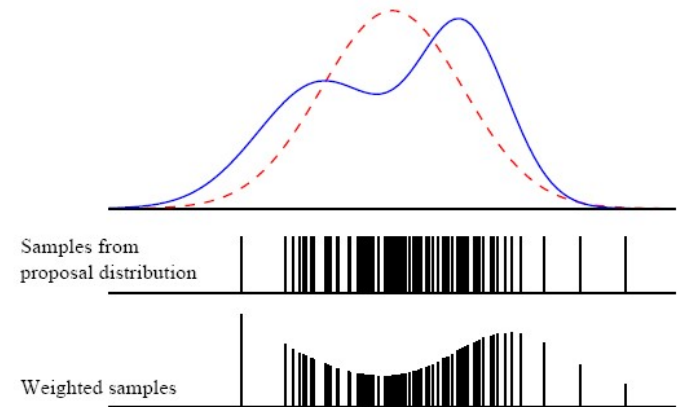
- Peso delle particelle e ricampionamento:
le particelle non sono distribuite secondo il posterior, non è inclusa la misura
- Occorre il peso (considerando le corrispondenze):

$$w^{[k]} = \eta \int \underbrace{p(m_j | x_{1:t-1}^{[k]}, z_{1:t-1})}_{\mathcal{N}(m_j; \mu_{j,t-1}^{[k]}, \Sigma_{j,t-1}^{[k]})} \underbrace{p(z_t | x_t^{[k]}, m_j)}_{\mathcal{N}(z_t; \hat{z}^{[k]}, Q_t)} dm_j$$

$$Q = H \Sigma_{j,t-1}^{[k]} H^T + Q_t$$

$$w^{[k]} \simeq |2\pi Q|^{-\frac{1}{2}} \exp \left\{ -\frac{1}{2} (z_t - \hat{z}^{[k]})^T Q^{-1} (z_t - \hat{z}^{[k]}) \right\}$$

measurement covariance

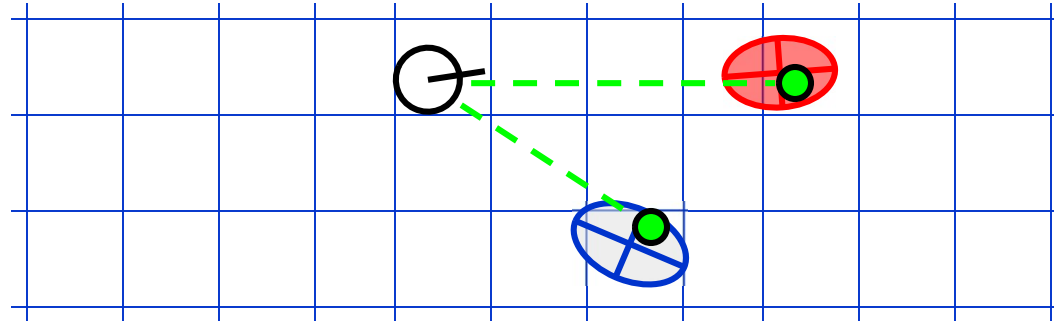


Innovazione: valore osservato - valore stimato

FastSLAM – Sensor Update

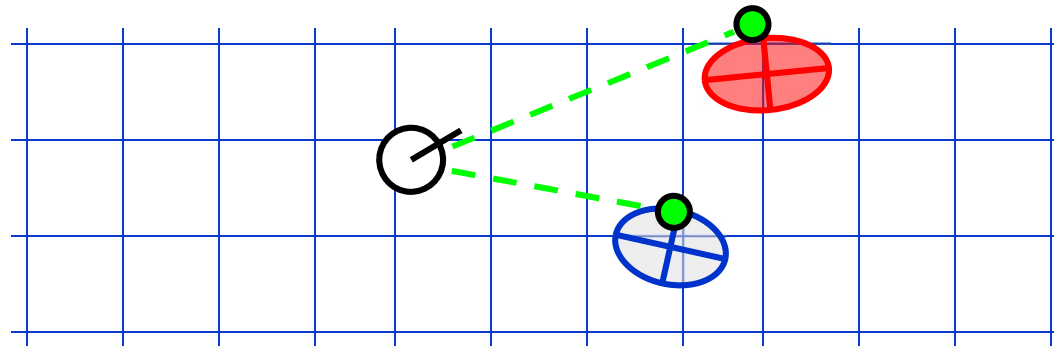
Peso per la verosomiglianza

Particle #1



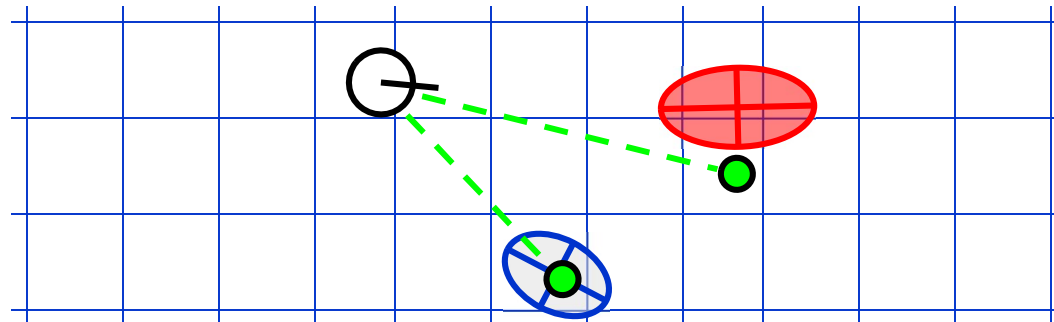
Weight = 0.8

Particle #2



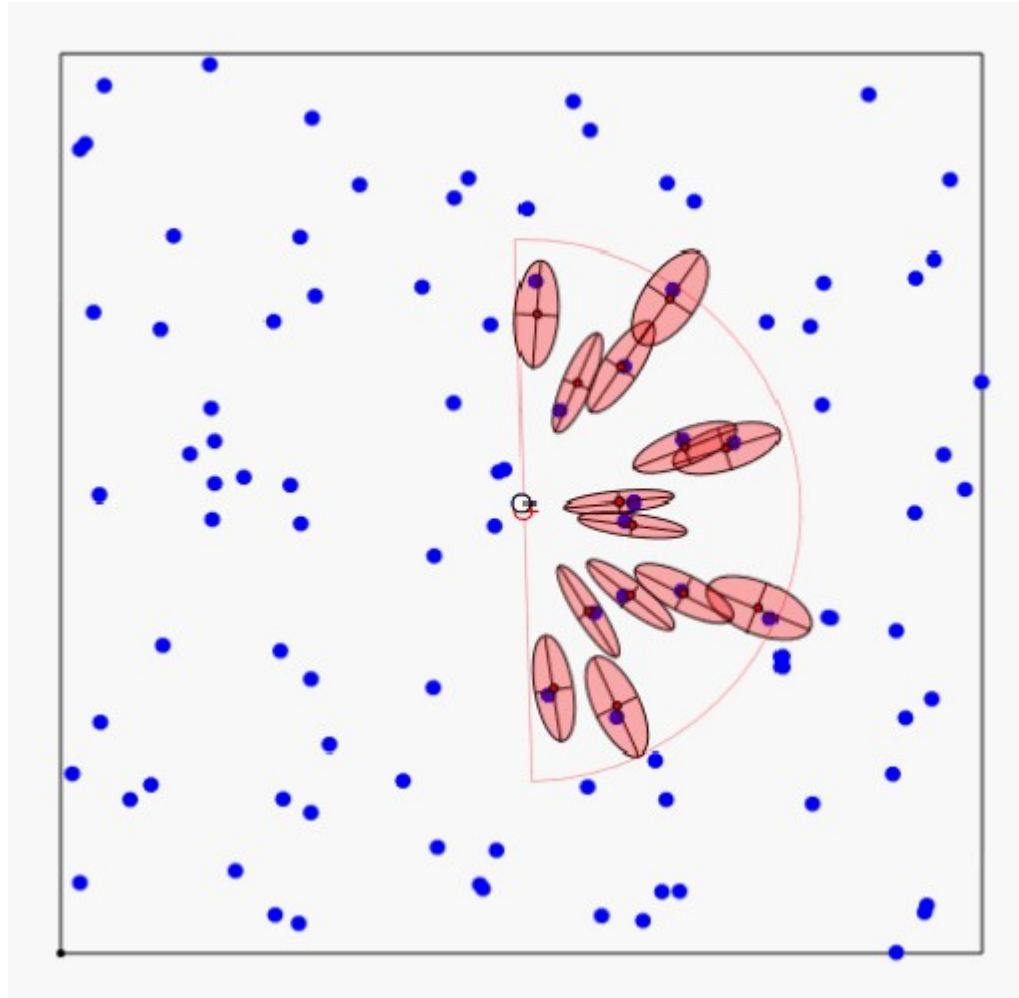
Weight = 0.4

Particle #3

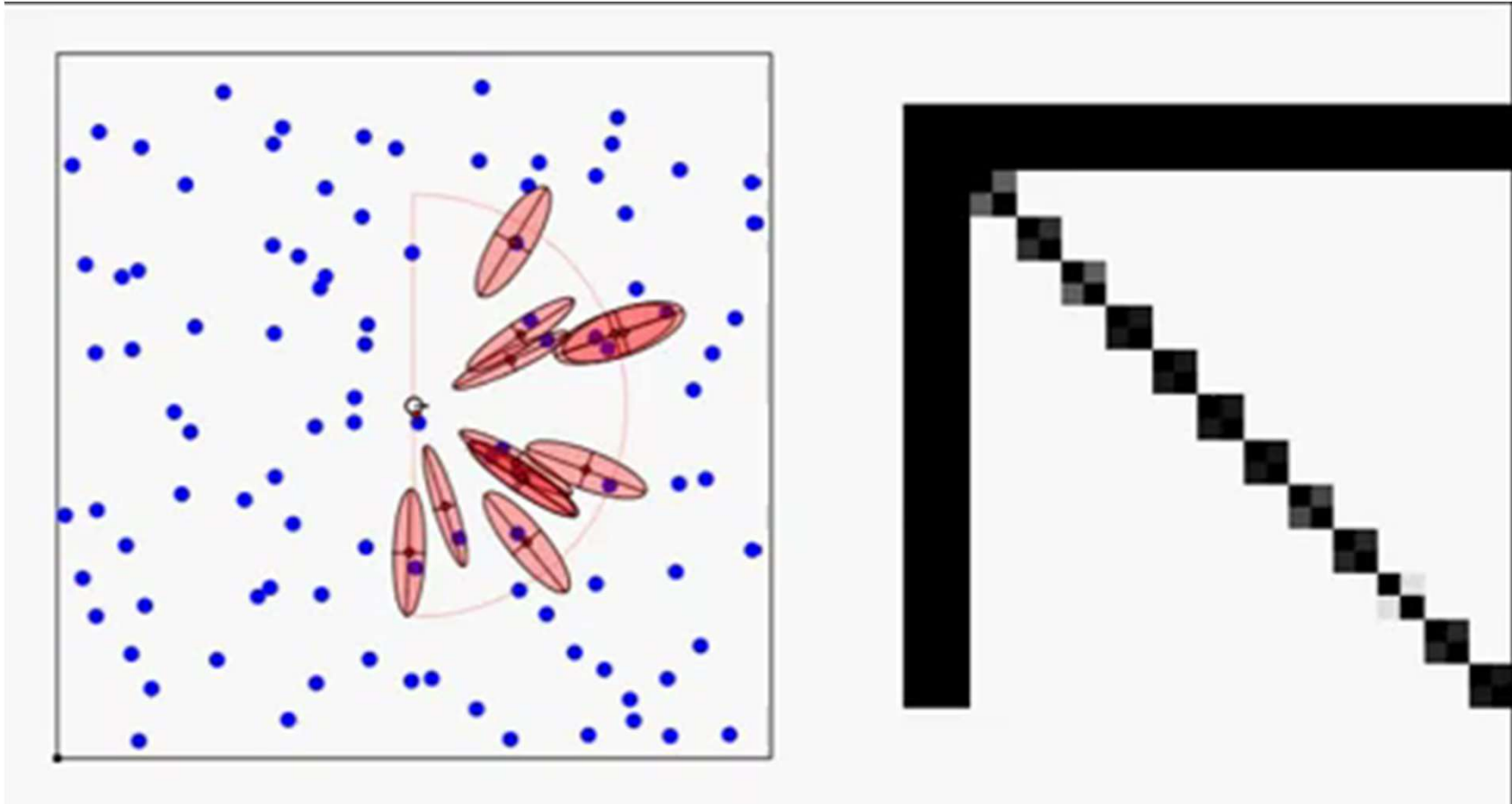


Weight = 0.1

FastSLAM



FastSLAM



Courtesy: Mike Montemerlo

FastSLAM Complessità (Naive)

- Update particelle del robot basate su controllo u_{t-1} $O(N)$
Constant time per particle

- Incorporare osservazioni z_t nei filtri di Kalman $O(N \cdot M)$
M per particle

- Resampling del particle set $O(N \cdot M)$
Log time per particle

N = Numero di particelle
M = Numero di features

$O(N \cdot M)$
M per particle

FastSLAM Complessità

- Update particelle del robot basate su controllo u_{t-1} $O(N)$
Constant time per particle

- Incorporare osservazioni z_t nei filtri di Kalman $O(N \cdot \log(M))$
Log time per particle

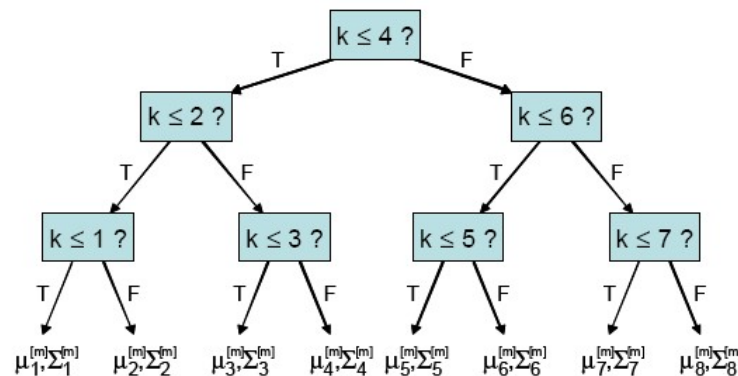
- Resampling del particle set $O(N \cdot \log(M))$
Log time per particle

N = Numero di particelle
M = Numero di features

$O(N \cdot \log(M))$
Log time per particle

FastSLAM Complessità

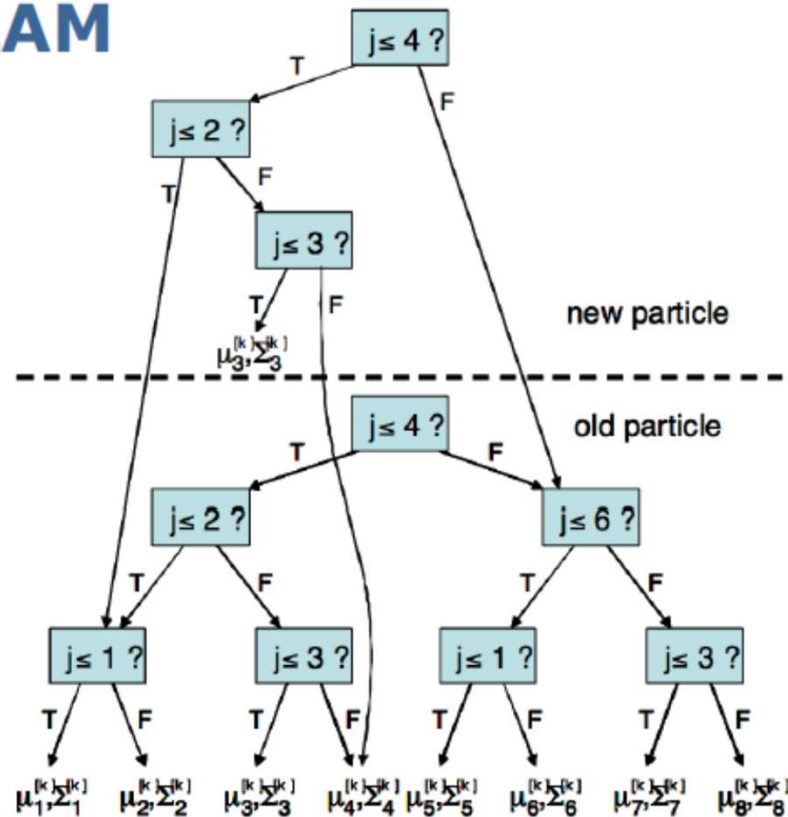
- I passi di update e di peso sono lineari nel numero delle particelle
- Il passo di resampling se naive allora lineare nel numero di feature
- Se ogni particella organizzata come albero binario di stimatori di feature allora $\log N$
- Fino a 1000000 landmark



FastSLAM Complessità

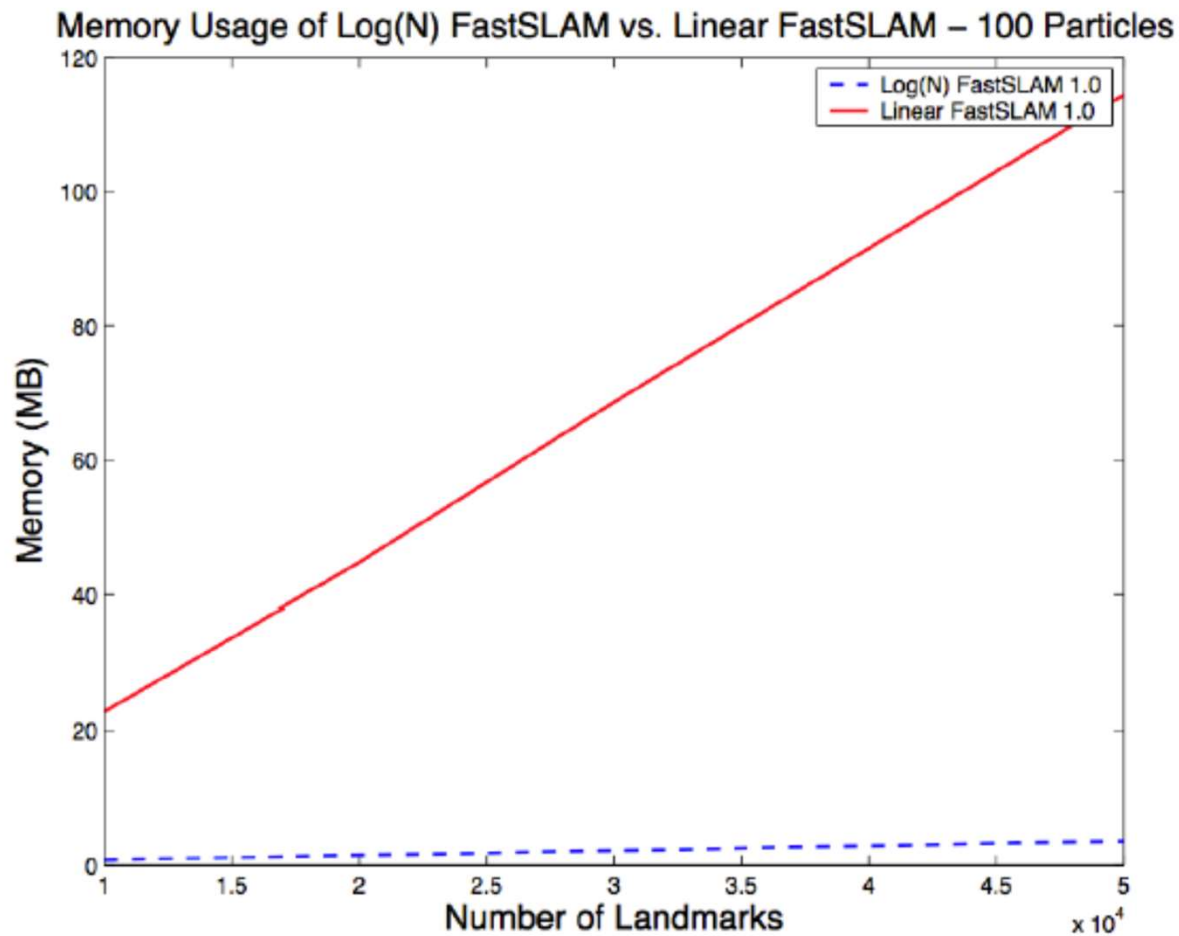
- I passi di update e di peso sono lineari nel numero delle particelle

A Better Data Structure for FastSLAM



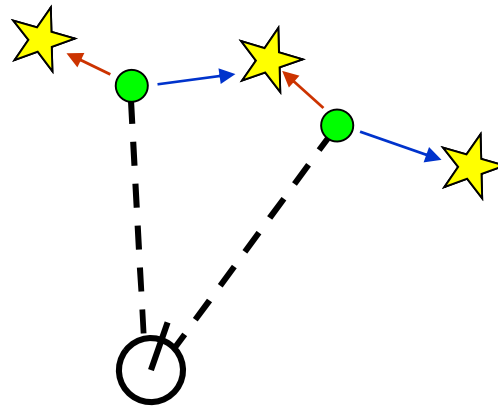
FastSLAM Complessità

- Memory Complexity



Problema di Data Association

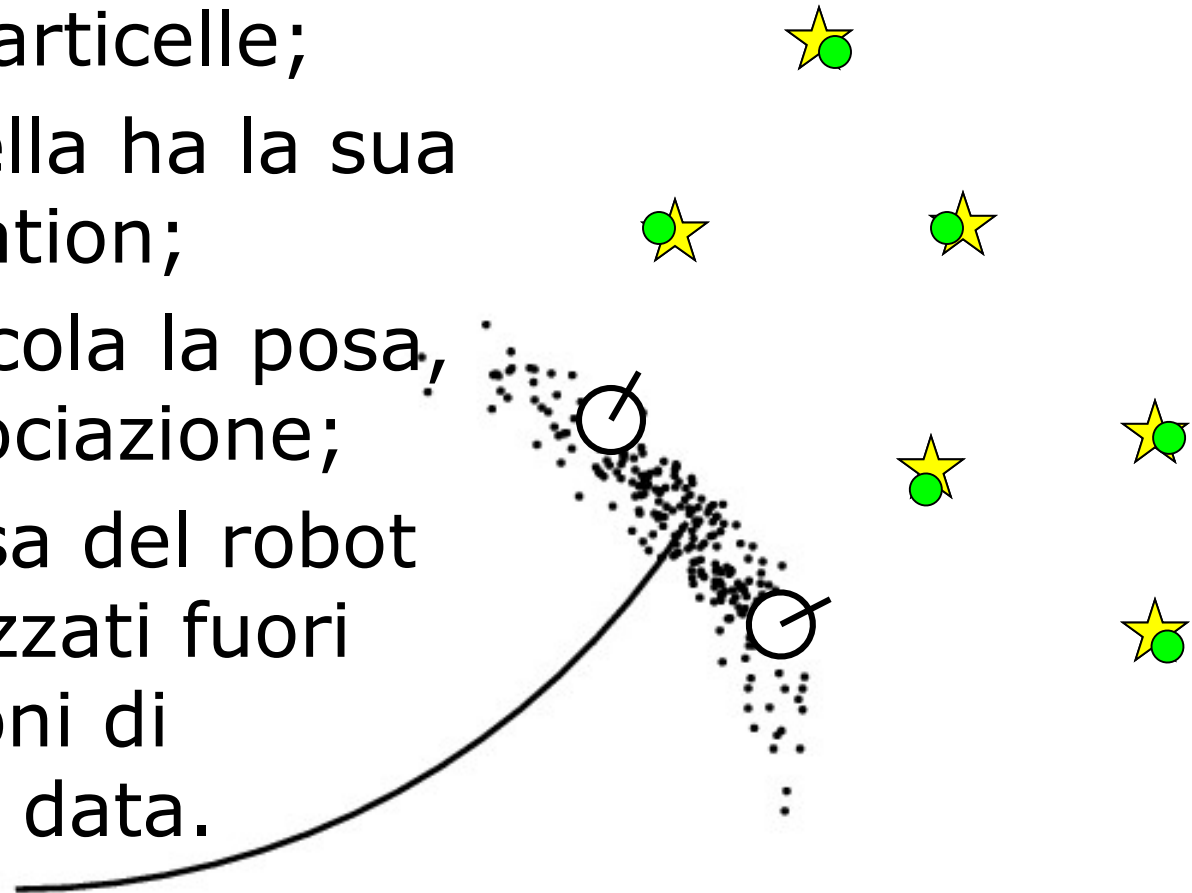
- Quali osservazioni appartengono a quale landmark?



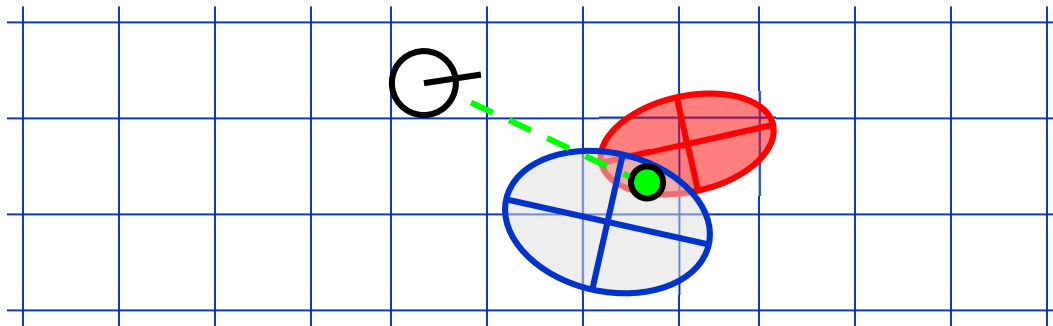
- Un SLAM robusto deve considerare possibili associazioni di dati
- Le possibili associazioni di dati dipendono anche dalla posa del robot

Data Association multi-ipotesi

- Se la data association è basata su particelle;
- Ogni particella ha la sua data association;
- Prima si calcola la posa, quindi l'associazione;
- Errori di posa del robot sono fattorizzati fuori dalle decisioni di associazioni data.



Associazione dati per particella



Osservazione
generata dal landmark
blu o rosso?

$$P(\text{observation}|\text{red}) = 0.3$$

$$P(\text{observation}|\text{blue}) = 0.7$$

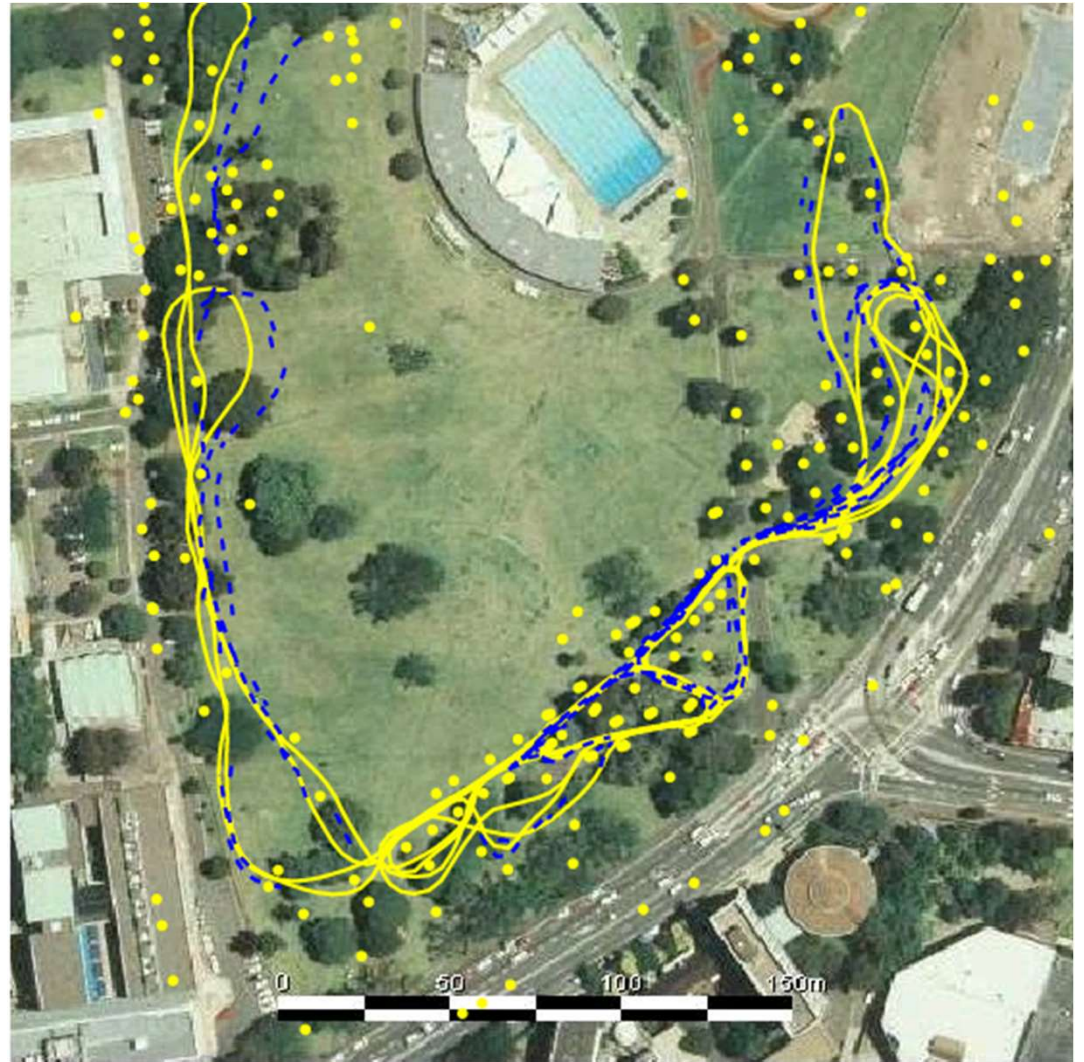
- Due opzioni per l'associazione per-particle:
 - Prendi la più probabile
 - Prendi un'associazione random pesata con il likelihood di osservazione
- Se la probabilità è bassa, genera un nuovo landmark

Risultati – Victoria Park

- Traversata di 4 km
- < 5 m RMS (Root Mean Square) errore di posizione
- 100 particelle

Blue = GPS

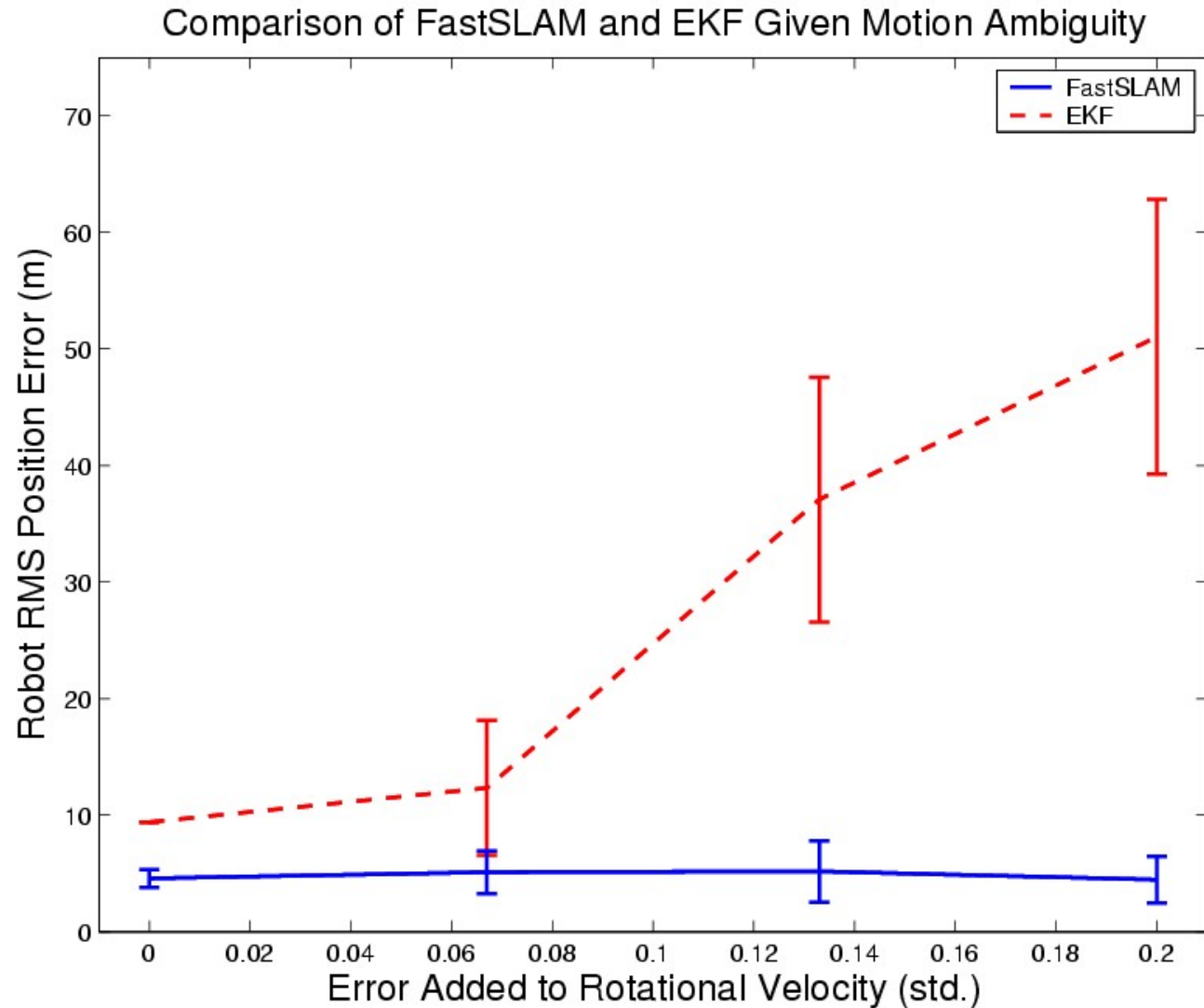
Yellow = FastSLAM



Risultati – Victoria Park



Risultati – Data Association



Risultati – Accuratezza

