FastSLAM: A Factored Solution To the Simultaneous Localization And Mapping Problem

Michael Montemerlo, Sebastian Thrun, Daphne Koller and Ben Wegbreit

Sebastián Gálvez Ortiz

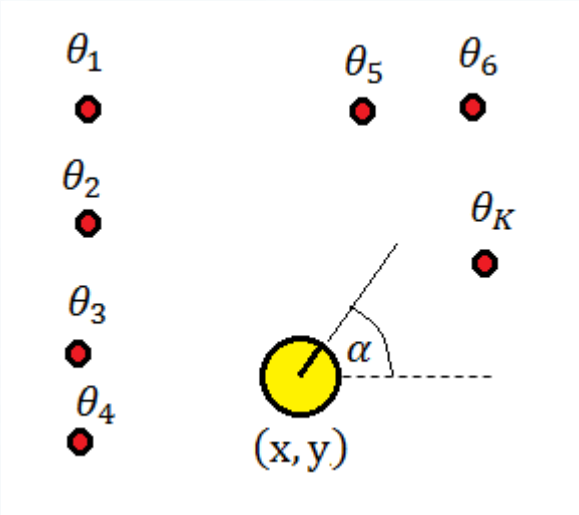
23/06/2015

Content

- Introduction:
  - Definition of the SLAM Problem
  - Problems with EKF SLAM solution
  - Factorization of SLAM
- FastSLAM Algorithm
  - Path Estimation
  - Landmark Location Estimation
  - Efficient Implementation
- Results
- Conclusion

Introduction

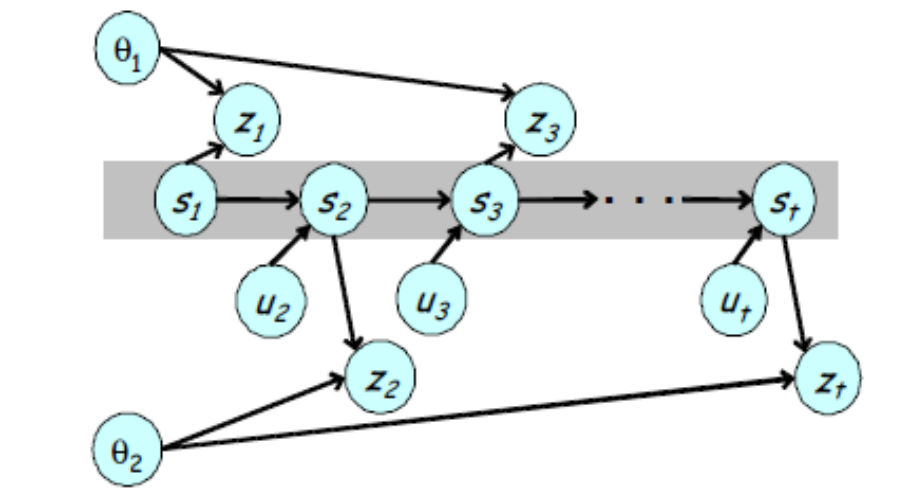
- Robot in a plane, with a pose $s_t = (x, y, \alpha)$ at time *t*
- There are *K* immobile landmarks with locations: $\{\theta_k\}_{k=1,\ldots,K}$



- Controls or odometry inputs u_t at time *t*.
- Measurement z_t at time *t* of a single landmark (range & bearing).

SLAM Problem

- Given the observations $z^t = z_1, z_2, z_3, \ldots, z_t$ and the controls $u^t = u_2, u_3, u_4, \ldots, u_t$ we want to estimate **simultaneously** the path $s^t = s_1, s_2, s_3, \ldots, s_t$ and the map of landmarks $\theta = \{\theta_1, \theta_2\}$



- This problem is simplified if we know the **correspondences** $n^t = n_1, n_2, n_3, \ldots, n_t$

¿Can anyone explain what are the *correspondences*?

SLAM Problem

- Given that we have:
  - The ***motion model:*** $p(s_t|u_t, s_{t-1})$
  - The ***measurement model:*** $p(z_t|\theta, s_t, n_t)$
- The SLAM problem can be described as using the observations z and controls u to find the **posterior belief:**

$$p(s^t, \theta|z^t, u^t, n^t)$$

EKF SLAM Disadvantages

- EKF SLAM considers the beliefs as *linearized Gaussian probability distributions.*

¿What do you think are the main problems of EKF SLAM?

EKF SLAM Disadvantages

- EKF SLAM considers the beliefs as *linearized Gaussian probability distributions.*
- ***Problems of EKF SLAM***

  - Linearization approximation error.
  - Monomodal belief distribution.
  - Kidnapped robot problem.
  - Computation of Jacobian matrixes.
  - $O(K^2)$ complexity.
  - Can only manage a few hundred landmarks.

Factorization of SLAM

- There is an implicit **conditional independence** on the path estimation and landmark location estimation:

  - *All the landmark observations depend on the robot path.*

- Then, we can obtain a factored representation of the posterior belief:

$$p(s^t, \theta | z^t, u^t, n^t) = p(s^t | z^t, u^t, n^t) \cdot p(\theta | z^t, u^t, n^t, s^t)$$

- Assuming conditional independence of landmark locations, the posterior is:

$$p(s^t, \theta | z^t, u^t, n^t) = p(s^t | z^t, u^t, n^t) \cdot \prod_{k}^{K} p(\theta_k | z^t, u^t, n^t, s^t)$$

¿Why does this help to solve the SLAM problem?

Factorization of SLAM

- There is an implicit **conditional independence** on the path estimation and landmark location estimation:

  - *All the landmark observations depend on the robot path.*

- Then, we can obtain a factored representation of the posterior belief:

$$p(s^t, \theta | z^t, u^t, n^t) = p(s^t | z^t, u^t, n^t) \cdot p(\theta | z^t, u^t, n^t, s^t)$$

- Assuming conditional independence of landmark locations, the posterior is:

$$p(s^t, \theta | z^t, u^t, n^t) = p(s^t | z^t, u^t, n^t) \cdot \prod_{k}^{K} p(\theta_k | z^t, u^t, n^t, s^t)$$

Landmark estimation

Localization problem

FastSLAM Algorithm

- FastSLAM uses the previous factorization (also known as Rao-Blackwellization).
- Then we can use a Particle Filter for path estimation, and Kalman filters to estimate the landmarks (given a estimated path).
- Each particle has an entire path estimate and K landmark estimates.
- The Particle Filter approximates non-linear probability distributions, allowing multiple hypothesis on the robot's pose.

Particle Filter Path Estimation

- Similar to MonteCarlo Localization, FastSLAM uses a set of M particles $S_t = \{s^{t,[m]}\}_m$, each representing a guess on the path.
- **First:** Each particle's pose at time t is sampled from the motion model:

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

- **Second:** The sampled pose is added to the path up to previous time of the particle: $s^{t-1,[m]}$, generating a temporary set $\widehat{S_t}$
- If we assume that the particle set S_{t-1} is distributed according to $p(s^{t-1}|z^{t-1}, u^{t-1}, n^{t-1})$, then the temporary set $\widehat{S_t}$ approximates the **proposal distribution:** $p(s^t|z^{t-1}, u^t, \mathrm{n}^{\mathrm{t}-1})$.

Particle Filter Path Estimation

- **Third:** As any particle filter, we assign an importance weight $w_t^{[m]}$ to each particle, calculated as:

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

- **Fourth (Resampling):** With a probability proportional to this weight, each particle is sampled from the temporary particle set, creating the new set S_t.
- This new set approximates the posterior belief for the path $p(s^t|z^t, u^t, n^t)$.

¿How would you make sure this is a good approximation of the belief?

Particle Filter Path Estimation

- **Third:** As any particle filter, we assign an importance weight $w_t^{[m]}$ to each particle, calculated as:

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

- **Fourth (Resampling):** With a probability proportional to this weight, each particle is sampled from the temporary particle set, creating the new set S_t.
- This new set approximates the posterior belief for the path $p(s^t|z^t, u^t, n^t)$.

One way is by using a large number of particles.

Landmark Location Estimation

- Each landmark estimation is represented using a Kalman Filter of size 2, conditional to the robot pose.
- Then, each particle has attached K Kalman Filters, so the particle set actually is: $S_t = \left\{ s^{t,[m]}, \mu_1^{[m]}, \Sigma_1^{[m]}, \dots, \mu_K^{[m]}, \Sigma_K^{[m]} \right\}_m$

  Where $\mu_k^{[m]}, \Sigma_k^{[m]}$ are mean and covariance of the Gaussian representing the k-th landmark θ_k, on the m-th particle.
- Depending on the correspondence value of the observation made at time t, the posterior belief for the landmark location is unchanged $(n_t \neq k)$ or updated $(n_t = k)$.

Landmark Location Estimation

- When the landmark θ_k was observed at time t, the belief on the location of that landmark must be updated:

$$p(\theta_k|s^t, z^t, u^t) = p(z_t|\theta_k, s_t, n_t) \cdot p(\theta_k|s^{t-1}, z^{t-1}, u^{t-1}, n^{t-1})$$

- To perform this update, a traditional Extended Kalman Filter is implemented, linearizing the measurement model $p(z_t|s_t, \theta, n_t)$.

- If the landmark θ_k was not observed at time t, the Gaussian does not change.

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

Landmark Location Estimation

¿What is the main difference between the Gaussians used on EKF SLAM and those used in FastSLAM?

Landmark Location Estimation

¿What is the main difference between the Gaussians used on EKF SLAM and those used in FastSLAM?

- EKF SLAM only calculates one Gaussian of dimension 2K+3 (in the known correspondences problem)

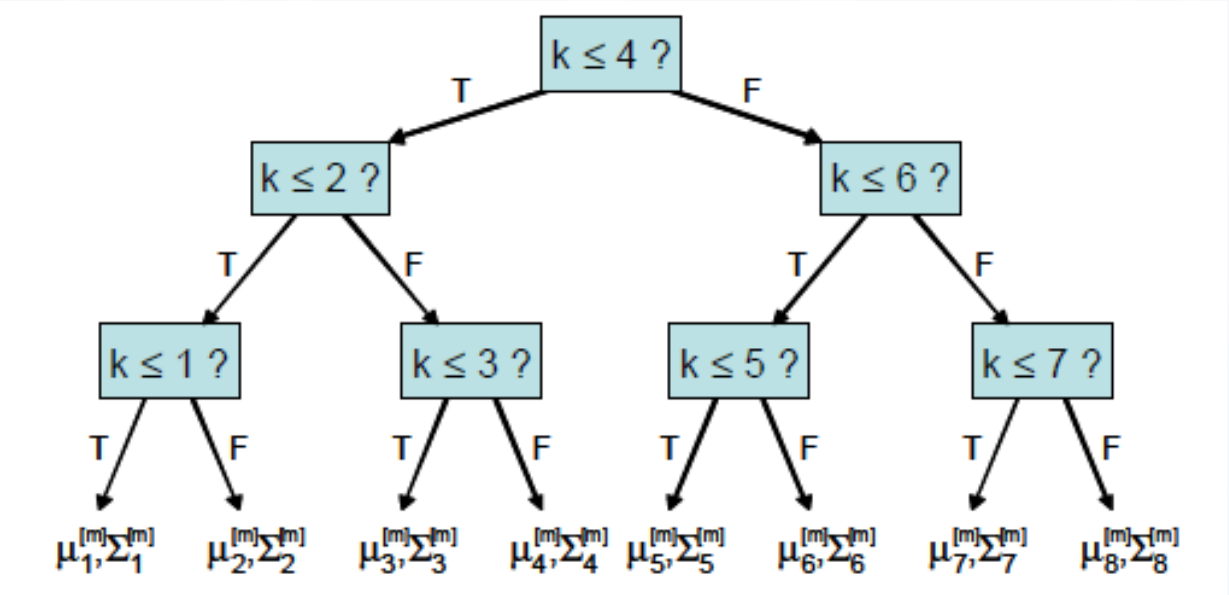
- FastSLAM calculates $K \cdot M$ Gaussians of dimension 2, which is faster. However, in a naive implementation requires $O(M \cdot K)$ time.

Efficient implementation

- As the particle filter requires the resampling step, in which the chosen particles must be copied to the new set S_t, from the temporary set $\widehat{S}_t$.
- When resampling, each new particle differs from the corresponding one in S_{t-1} in the path estimate, but at most only in one landmark estimate.
- Since the only Gaussian that changes after a new observation is the one with index n_t, then it's not necessary to copy all the landmark estimates on every step.
- A binary tree structure representing the landmark estimates can be useful to reduce the time required when resampling.

Efficient implementation

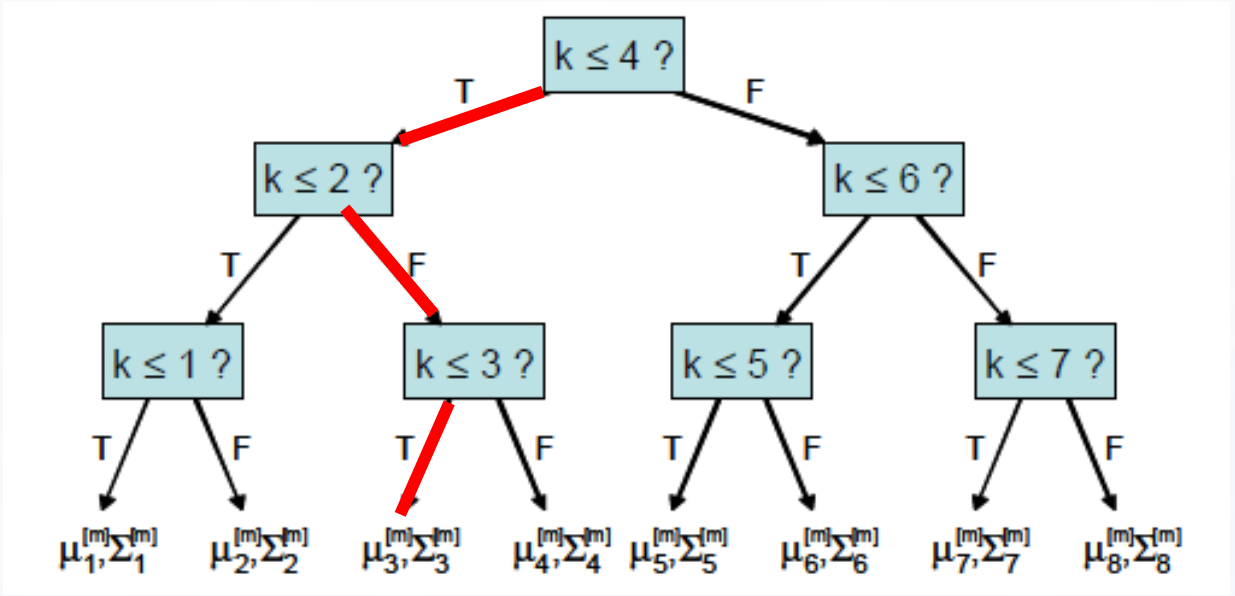
- Consider the following tree representing K=8 landmark estimates in a single particle:



¿How many steps are required to access the data of a specific Gaussian?

Efficient implementation

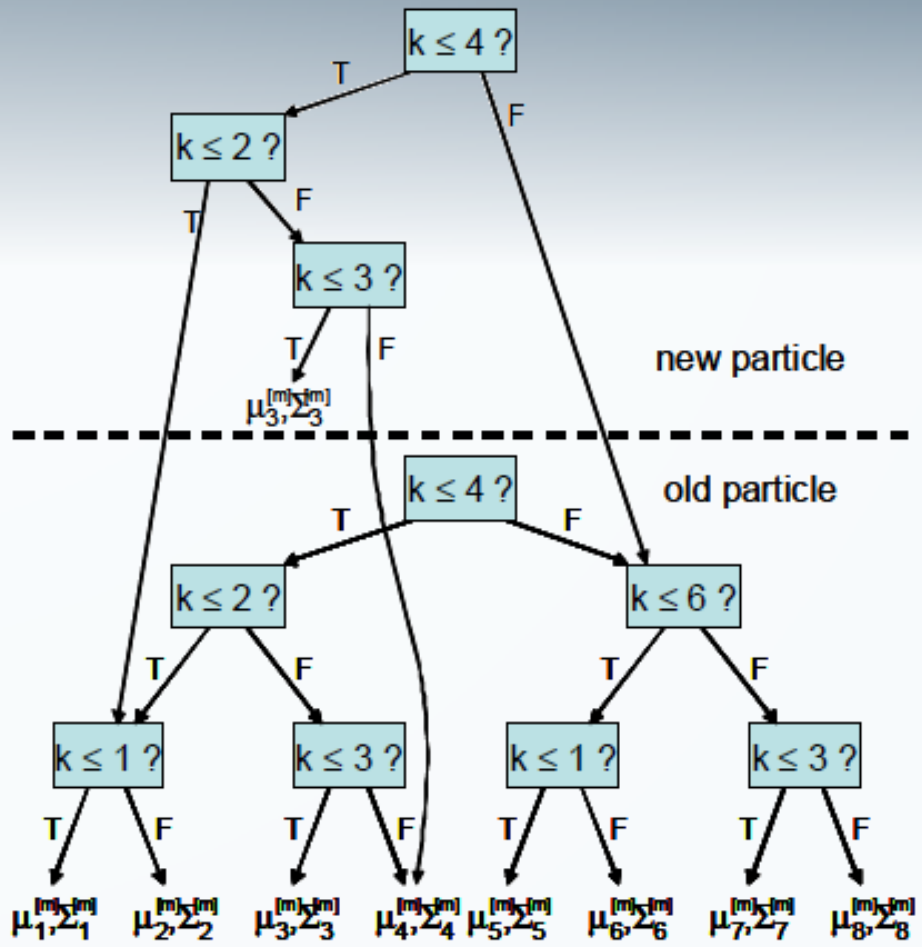
- Consider the following tree representing K=8 landmark estimates in a single particle:



Answer: For this example, log(8)=3. In general, O(log(K)) time.

Efficient implementation

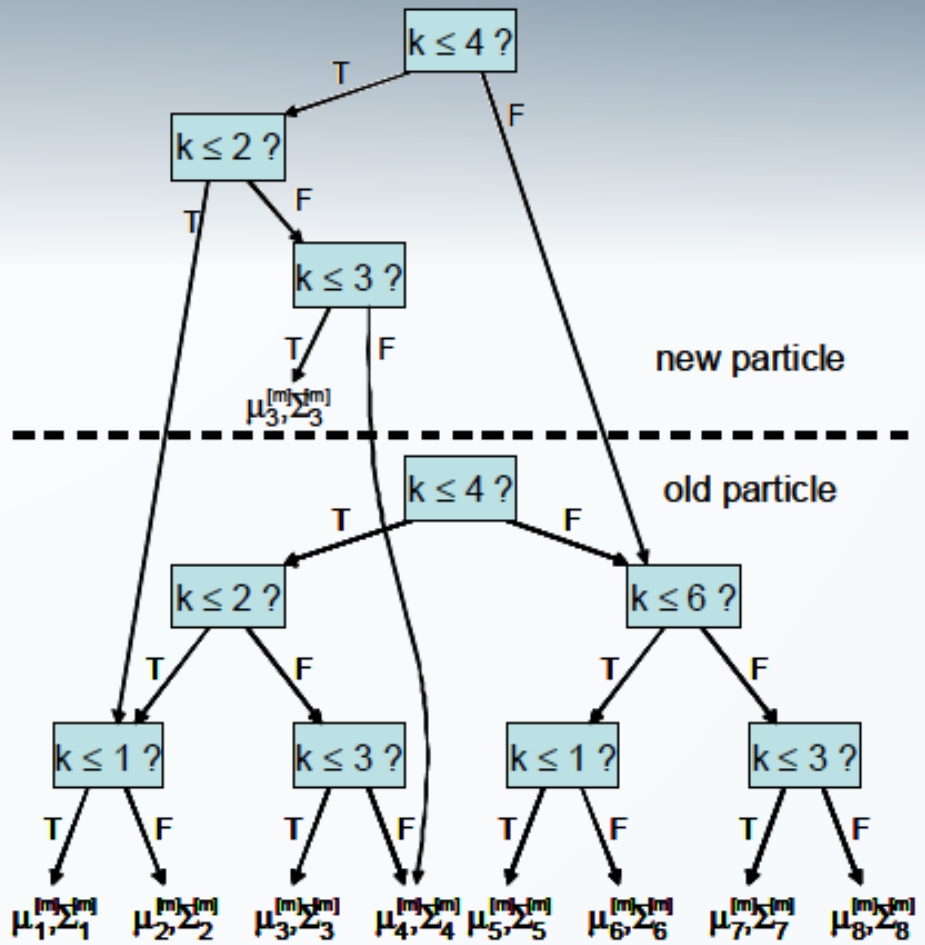
- A new control u_t and measurement z_t is incorporated, with n_t=3.
- The only Gaussian updated must be the one corresponding to θ_3.
- Instead of changing the whole tree, we create a path to the new values of $\mu_3^{[m]}, \Sigma_3^{[m]}$.
- Then, copy the corresponding pointers of the branches that leave the path.



Efficient implementation

- Both generating the new particle's tree and accessing a Gaussian data takes $O(\log(K))$ time.

- Since we need to do this for M particles, the updating process of the resampling step is made in time $O(M \cdot \log(K))$



Results

- This algorithm was tested both on real-world experiments and simulation experiments.

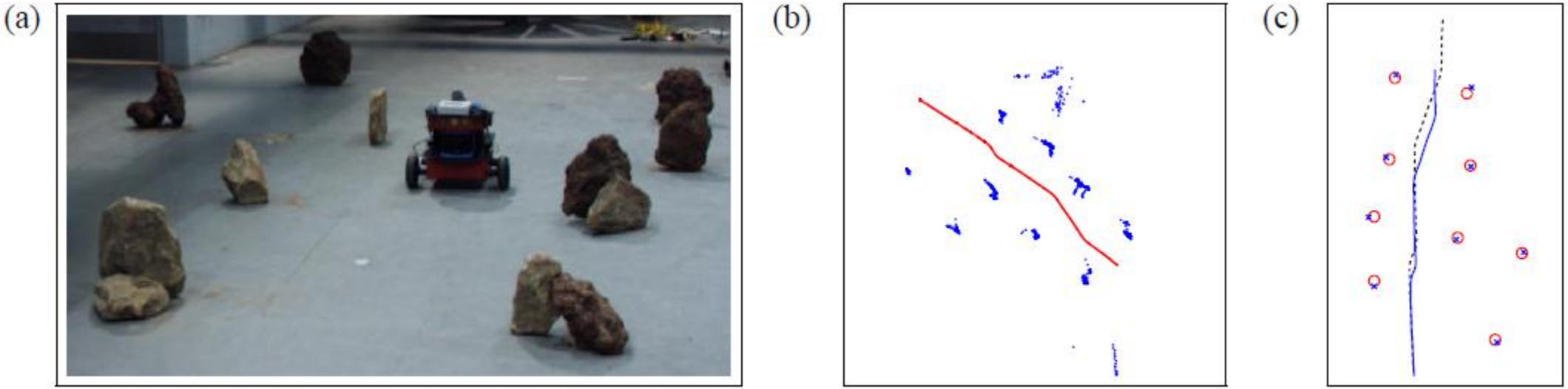


Figure 4: (a) Physical robot mapping rocks, in a testbed developed for Mars Rover research. (b) Raw range and path data. (c) Map generated using FastSLAM (dots), and locations of rocks determined manually (circles).

- Average residual map error of 8.3 [cm]

Results

- By using extensive simulations, using M=100 particles, FastSLAM was able to map up to 50000 landmarks.

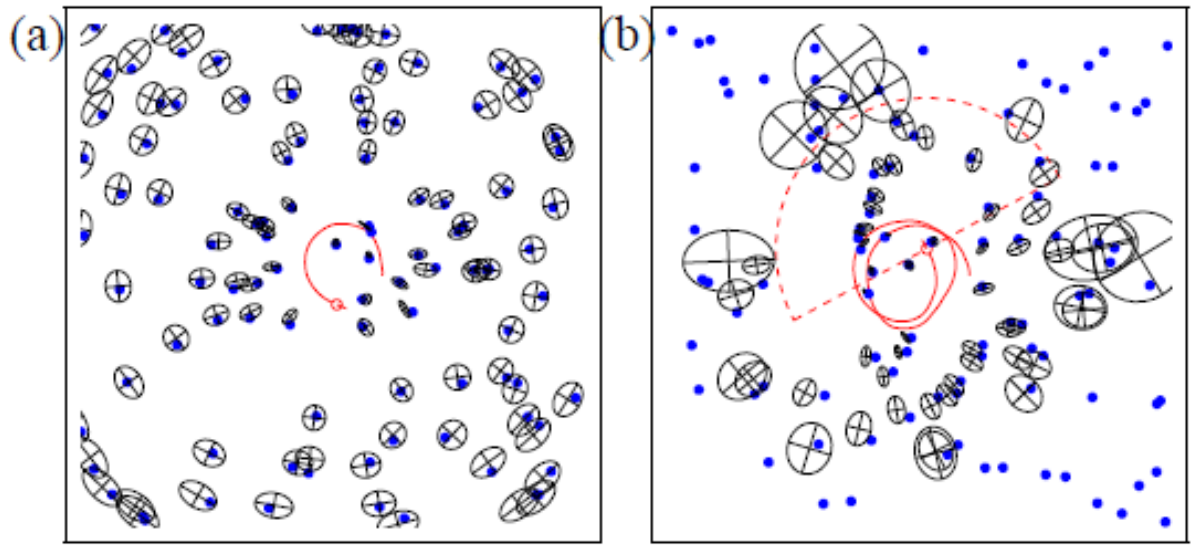


Figure 5: Maps and estimated robot path, generated using sensors with (a) large and (b) small perceptual fields. The correct landmark locations are shown as dots, and the estimates as ellipses, whose sizes correspond to the residual uncertainty.

Conclusion

- FastSLAM considerably increases the number of landmarks that can be computed in a SLAM algorithm (at that time).

- This algorithm successfully takes advantage on the Rao-Blackwellized representation of the posterior belief to solve the path estimation problem using all the particle filter capabilities in a non-linear motion model, while combining the simplicity of extended Kalman filters for landmark estimations.

- FastSLAM is significantly faster than EKF SLAM, as the efficient implementation requires $O(\mathrm{M} \cdot \log(K))$ time, in contrast to $O(K^2)$ from EKF SLAM.

References

- Montemerlo, M., Thrun, S., Koller, D., & Wegbreit, B. (2002, July). FastSLAM: A factored solution to the simultaneous localization and mapping problem. In*AAAI/IAAI* (pp. 593-598).

- "EL7031: Part IV -SLAM & the Extended Kalman Filter (EKF)" - Lecture from *Material Docente*.

- "EL7031: Part V - SLAM – Advanced Methods"- Lecture from *Material Docente*.