Content:

1. Introduction
- Problem of range only
- SLAM and RO-SLAM
- Objectives

2. SLAM based on EKF
- Formulation of EKF

3. RO-SLAM
- Problem statement
- Proposing two questions
- Solution and answers
- Formulation of RO-SLAM

4. Results and discussion
- Simulation 2D Map
- Simulation 3D Map
- Experimental results
- Discussion

5. Conclusion
- Does objective met?
- Some findings

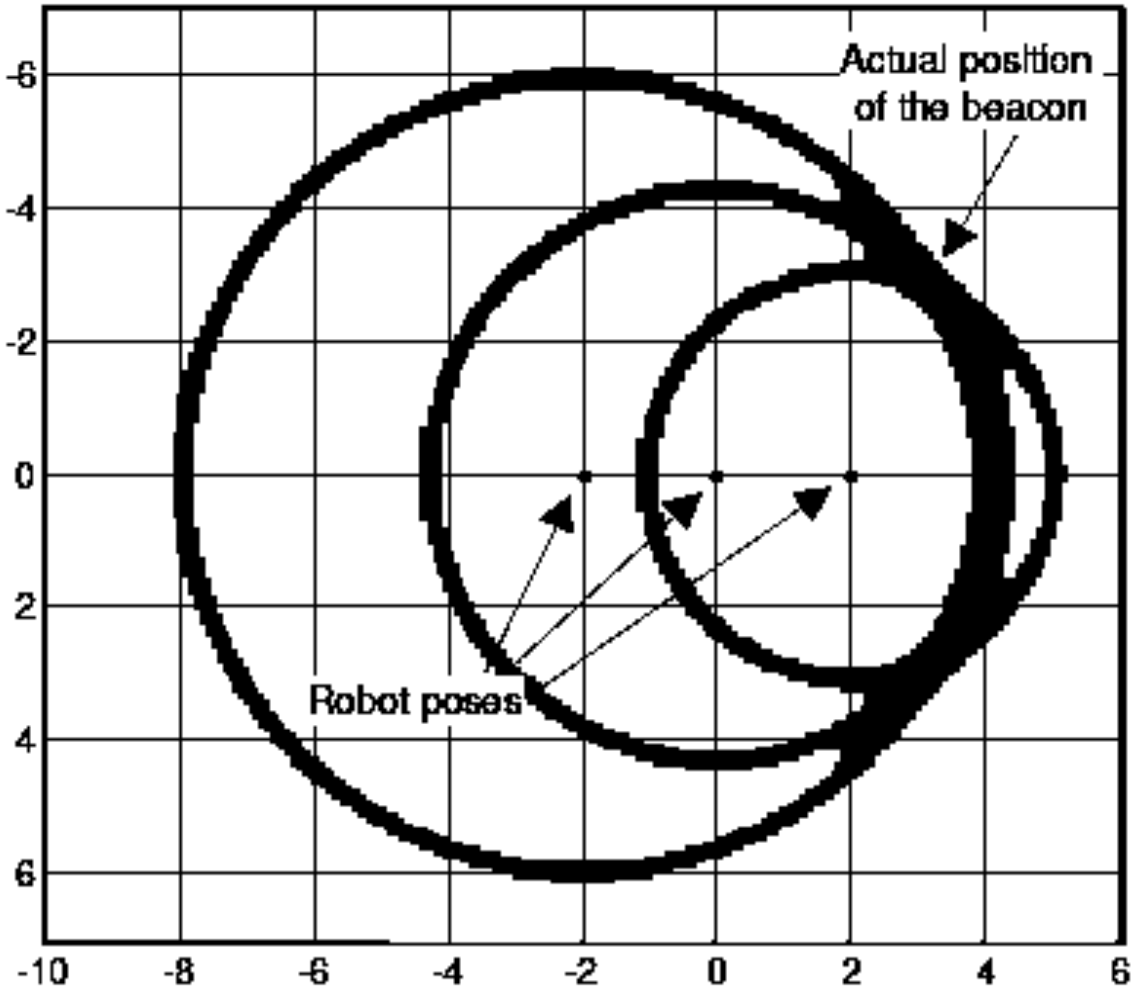
Bayesian Range Only SLAM with SOGs

Sepideh Hadadi, Muhammad Zain Bashir
Autonomous Robotics, MSCV program, Le Creusot Condorcet, 08 June 2017.

SLAM and RO-SLAM

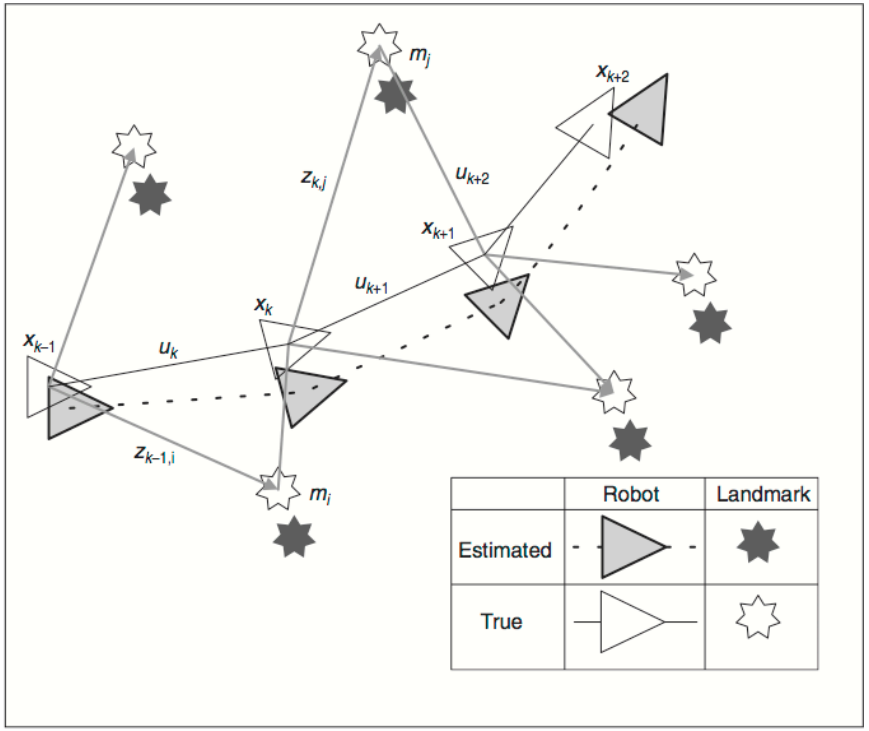
- SLAM: simultaneous Mapping and localization
- RO-SLAM: range only SLAM
  - Only range data is available
  - Information about bearing is not available
  - Instead of angle information, a sample of measurement is available between two consecutive points
  - The aim of RO-SLAM is use sample measurements to solve ambiguity arises due to lack of bearing data and to limit the search area.
  - RO-SLAM is an updated version of SLAM which explains how samples can be used to guide a robot in the absence of one information sensor data.

Problem in RO-SLAM



1. **Ambiguity:** the very likely possibility of multiple plausible hypotheses
2. **Large search region:** the large portion of the environment where a beacon could be, given just one observation

SLAM principle



EKM-SLAM measures only in the maneuvering point at time K. while RBPF-SLAM measures at n points between time k and k+1.

1. **Initialization of the robot**
2. **localization**
3. **Mapping**
4. **Sensing**

SLAM will always use several different types of sensors.

1. **Range (r)**
2. ~~**Bearing(θ)**~~
3. **Sample of measurement and estimation**

3. Kinematic modeling

Kinematic model deals only with the movement of the robot.

Formulation of Extended Kalman Filter

$$X_{k+1} = \begin{bmatrix} x_k + \Delta D_k \cos(\theta_k) \\ y_k + \Delta D_k \sin(\theta_k) \\ \theta_k + \Delta\theta_k \end{bmatrix} + w_k \triangleq f(X_k) + 0 \times u_k + w_k$$

$$Z_{k+1} = \begin{bmatrix} \sqrt{(x_b - x_k)^2 + (y_b - y_k)^2} \\ \sqrt{(x_k - x_{k-1})^2 + (y_k - y_{k-1})^2} \\ \theta_k - \theta_{k-1} \end{bmatrix} + v_{k+1} = \begin{bmatrix} r_k \\ \Delta D_k \\ \Delta\theta_k \end{bmatrix} + v_{k+1} \triangleq h(X_{k+1}) + v_{k+1}$$

EKF Algorithm:

$$F_k = \left.\frac{\partial f}{\partial X}\right|_{X = \hat{X}_{k+1}^-}$$

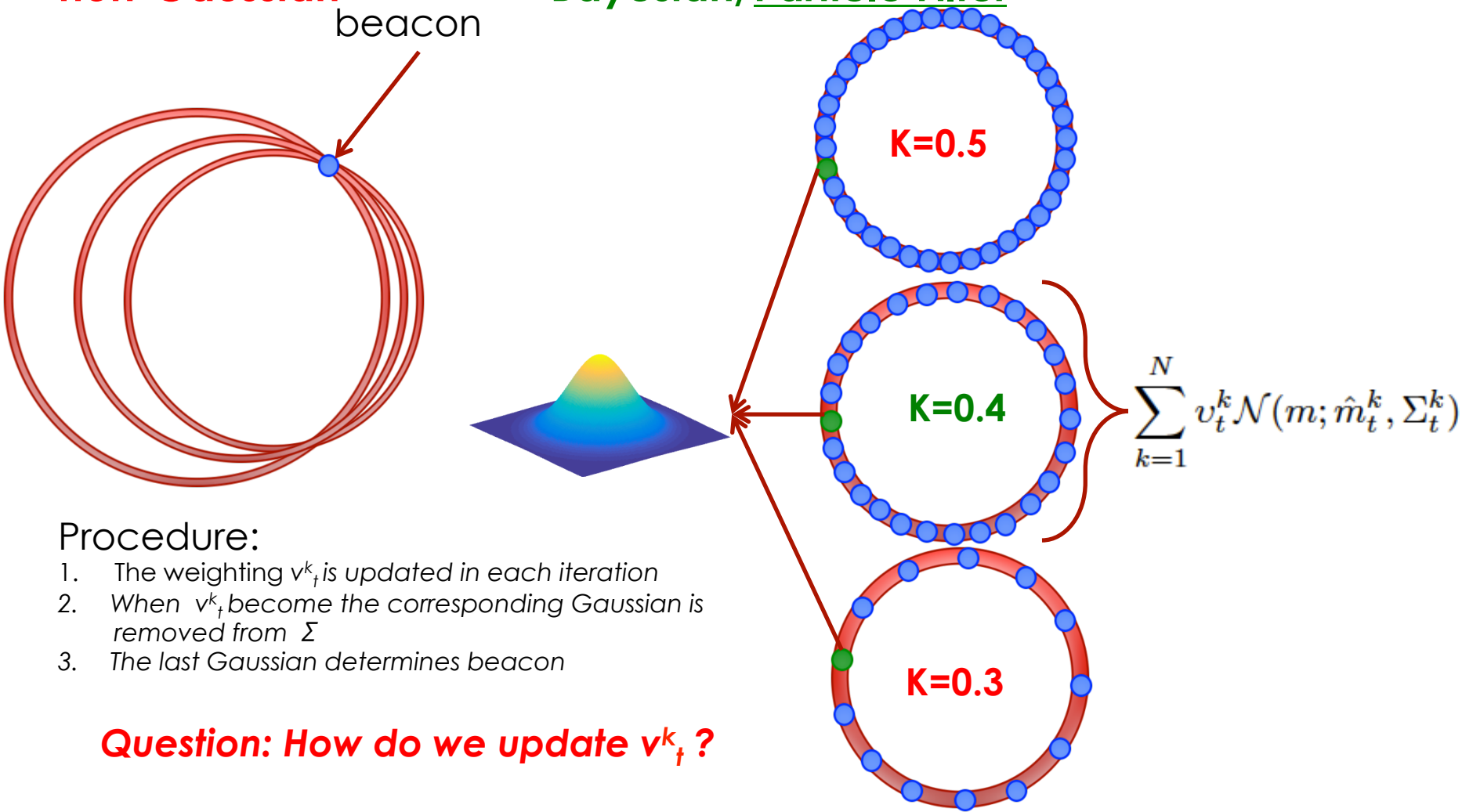
$$H_{k+1} = \left.\frac{\partial h}{\partial X}\right|_{X = \hat{X}_{k+1}^-}$$

KF Algorithm:

$$\begin{aligned}
x_k &= F_{k-1}x_{k-1} + w_{k-1} \\
y_k &= H_k x_k + v_k \\
P_{k+1}^- &= F_k P_k^+ F_k^T + Q_k \\
K_k &= P_k^- H_k^T (H_k P_k^- H_k^T + R_k)^{-1} \\
P_k^+ &= (I - K_k H_k) P_k^- (I - K_k H_k)^T + K_k R_k K_k^T \\
\hat{x}_k^- &= F_{k-1}\hat{x}_{k-1}^+ \\
\hat{x}_k^+ &= \hat{x}_k^- + K_k(y_k - H_k\hat{x}_k^-)
\end{aligned}$$

RO-SLAM base on RBPF

- Process equation is **non-linear** and noise distribution is **non-Gaussian** → **Bayesian/Particle Filter**



Procedure:

1. The weighting v^k_t *is updated in each iteration*
2. *When* v^k_t *become the corresponding Gaussian is removed from* Σ
3. *The last Gaussian determines beacon*

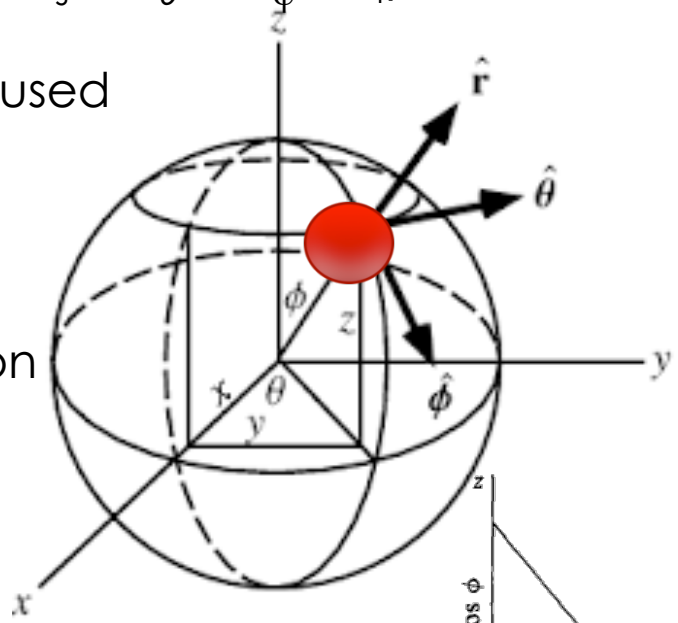
Question: How do we update v^k_t ***?***

Formulation of error distribution

The parameters of the Gaussian are explained in **spherical coordinate** because of simplicity

Each Gaussian has three variance ($\sigma_r = \sigma_s$, $\sigma_\theta = \sigma_\varphi = \sigma_t$)

Instead of r, θ , ϕ vector V_1, V_2, V_3 will be used to be uniform with other literature



Spherical to **Cartesian** coordination conversion:

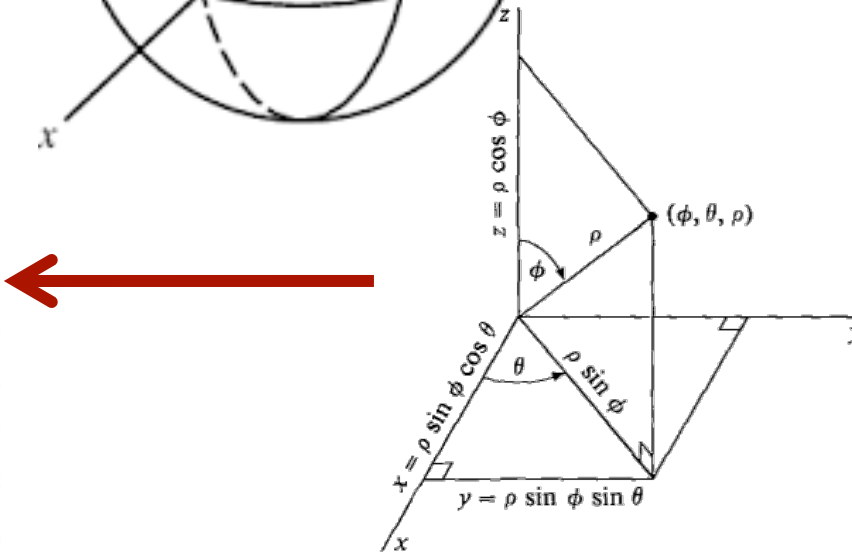
$$A_r = A_x \sin\theta\cos\phi + A_y \sin\theta\sin\phi + A_z \cos\theta$$

$$A_\theta = A_x \cos\theta\cos\phi + A_y \cos\theta\sin\phi - A_z \sin\theta$$

$$A_\phi = -A_x \sin\phi + A_y \cos\phi$$

In matrix form we can write

$$\begin{bmatrix} A_r \\ A_\theta \\ A_\phi \end{bmatrix} = \begin{bmatrix} \sin\theta\cos\phi & \sin\theta\sin\phi & \cos\theta \\ \cos\theta\cos\phi & \cos\theta\sin\phi & -\sin\theta \\ -\sin\phi & \cos\phi & 0 \end{bmatrix} \begin{bmatrix} A_x \\ A_y \\ A_z \end{bmatrix}$$



Formulation of RO-SLAM

$$p(m|x^t, z^t) \propto \underbrace{p(m|x^{t-1}, z^{t-1})}_{prior} \underbrace{p(z_t|m, x^t, z^{t-1})}_{invers\ sensor\ model} \quad (1)$$

$$p(z_t|m, x^t, z^{t-1}) \propto exp\left(-\frac{1}{2}\frac{|x_t - m|^2}{\sigma_s^2}\right) \quad (2)$$

$$p(z_t|m, x^t, z^{t-1}) \approx \sum_{k=1}^{N} v_t^k N(m, \hat{m}_t^k, \sum_t^k) \quad (3)$$

$$\hat{m}_t^{ij} = \begin{pmatrix} x_0 + r\cos(\alpha_i)\cos(\beta_j) \\ y_0 + r\sin(\alpha_i)\cos(\beta_j) \\ z_0 + r\sin(\alpha_i) \end{pmatrix}$$

$$\Sigma_t^{ij} = \begin{pmatrix} v_1 & v_2 & v_3 \end{pmatrix} \begin{pmatrix} \sigma_s^2 & 0 & 0 \\ 0 & \sigma_t^2 & 0 \\ 0 & 0 & \sigma_t^2 \end{pmatrix} \begin{pmatrix} v_1^T \\ v_2^T \\ v_3^T \end{pmatrix}$$

$$v_t^k = v_{t-1}^k N\left(z_t, h_t^k, {\sigma_t^k}^2\right)$$

$$\sigma_t^{k\,2} = H\Sigma_t^k H^T + \sigma_s^2$$

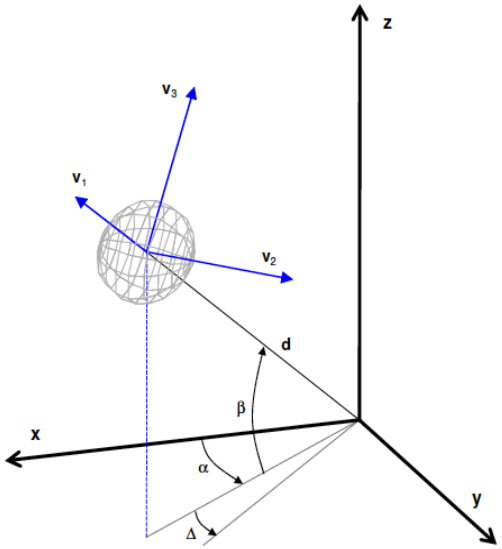
$$h_t^k = h(x_t^{[i]}, \hat{m}_t^k)$$

Spherical unique vector

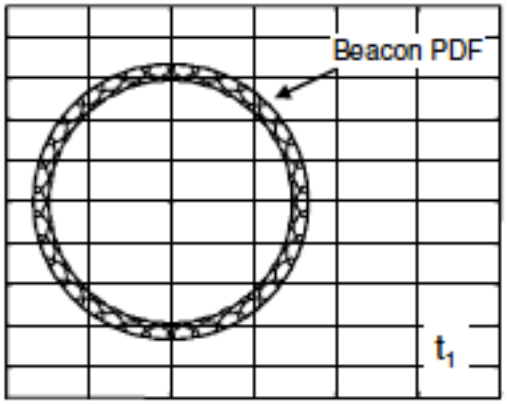
Sensor position

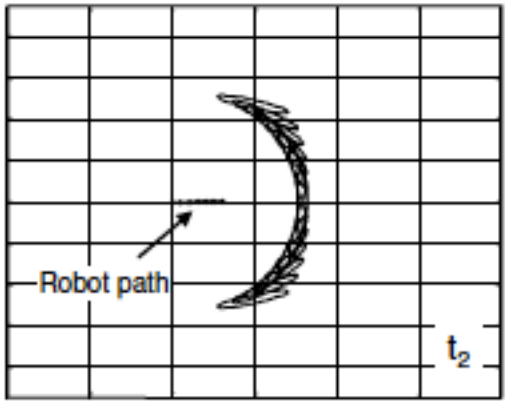
Sensor variance

Angular variance

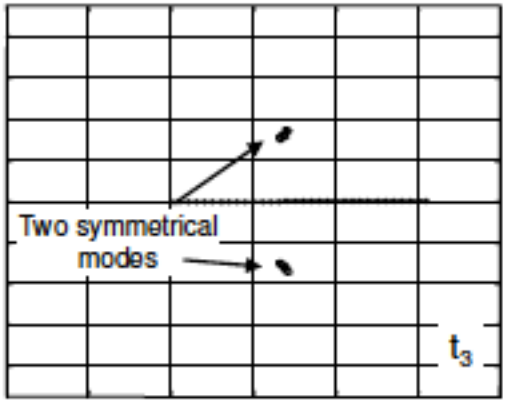


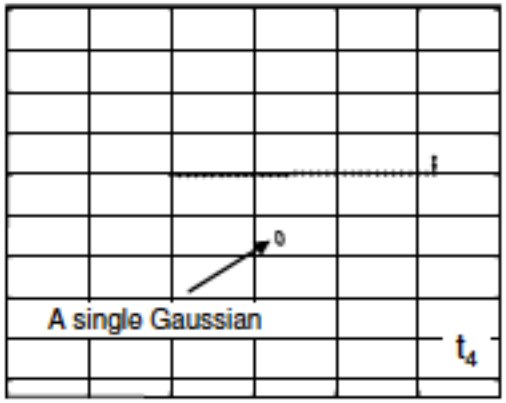
Removing ambiguity and reducing search area using RBPF





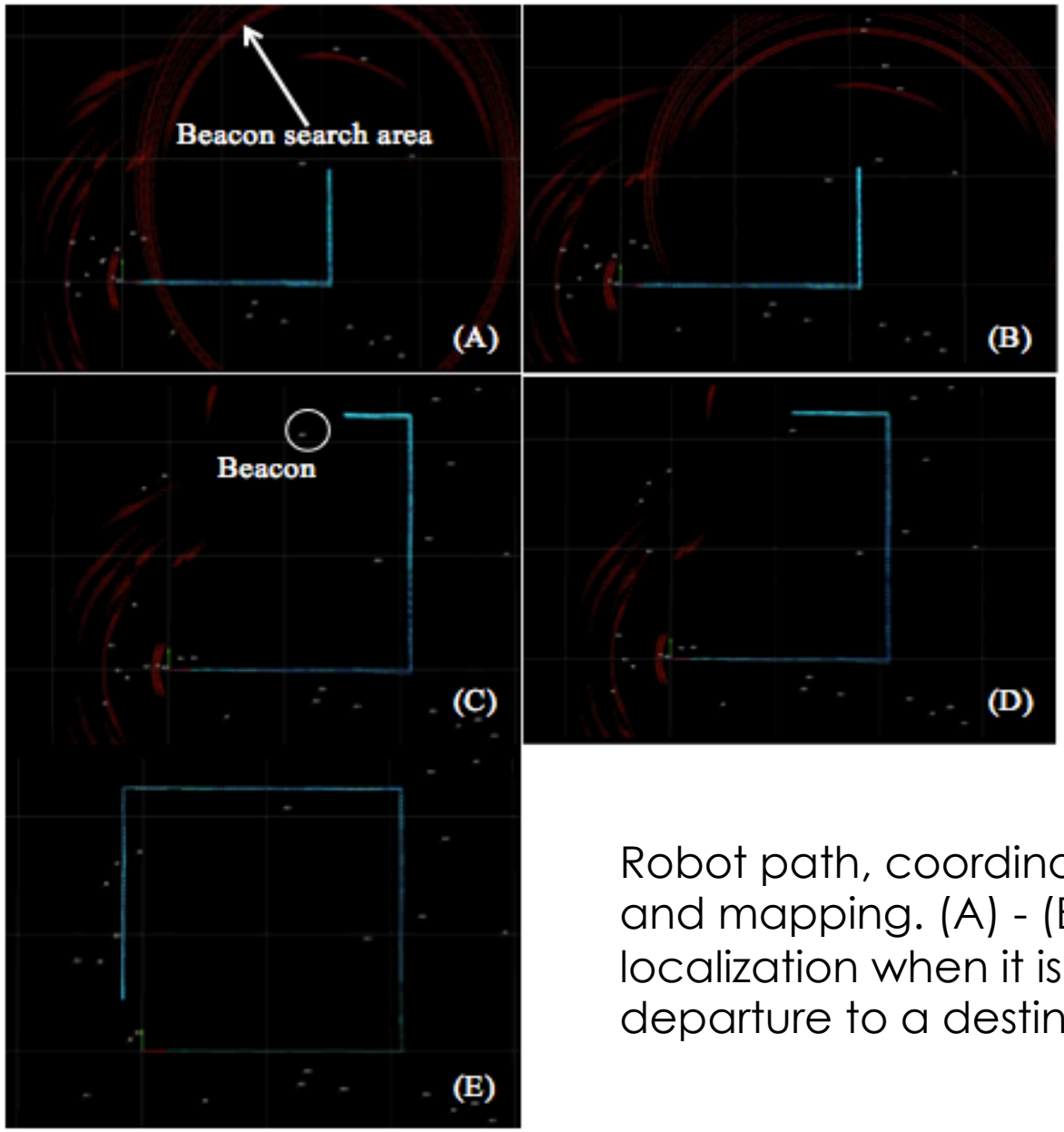
Reducing region of interest





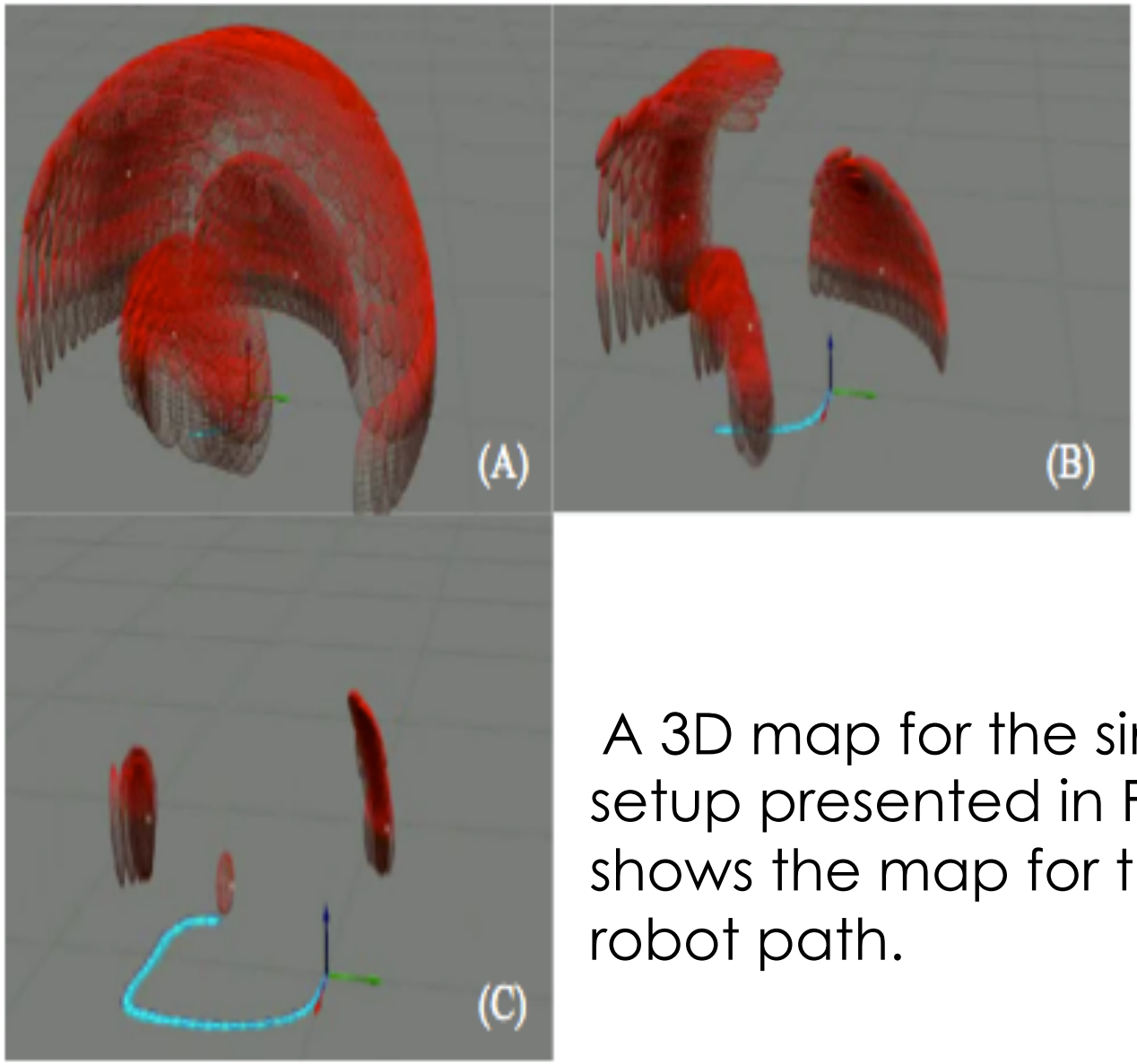
Removing ambiguity

Simulation result 2D



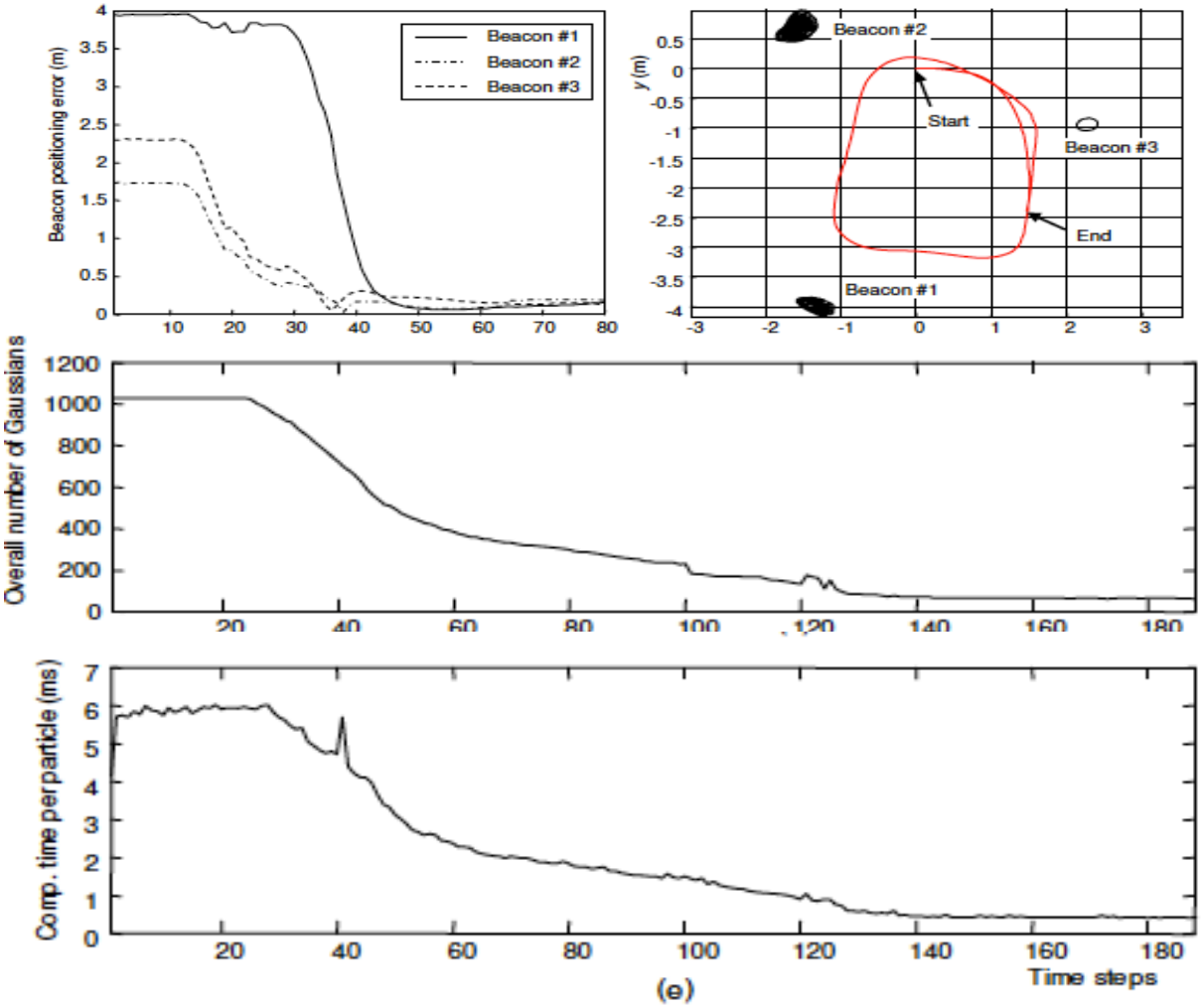
Robot path, coordinate of the bacon and mapping. (A) - (E) shows robot localization when it is moving from a departure to a destination.

Simulation result 3D

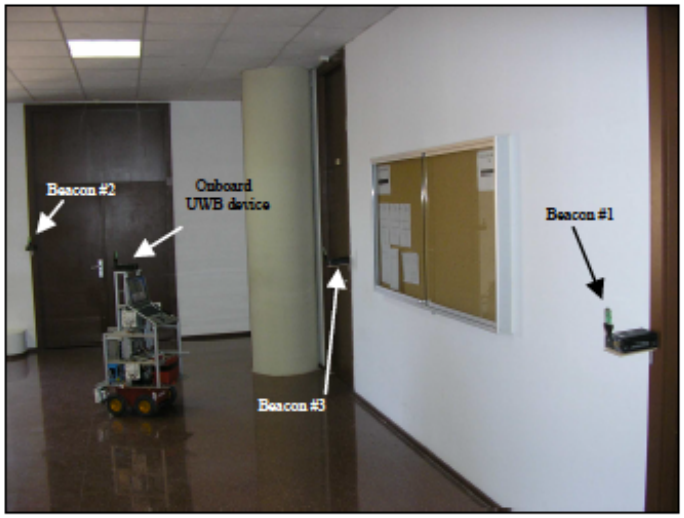


A 3D map for the simulation setup presented in Fig.3. (A)- (C) shows the map for three points in robot path.

Experimental result



Experimental data collected by real robot



Experimental result

Beacon coordinate		Ground truth (m)	Estimate (m)	Error (m)
#1	x	0	-0.059	0.059
	y	0	-0.278	0.278
	z	0.912	1.17	0.257
#2	x	-0.320	-0.392	0.072
	y	4.332	4.419	0.087
	z	1.374	1.214	0.159
#3	x	3.403	3.513	0.109
	y	2.802	2.740	0.062
	z	2.175	2.108	0.067

errors for the 3d map built from UWB data

Conclusion and future work

- The ambiguity removed from the problem
- The region quickly limited to the place where beacon is
- The number of Gaussian decreases exponentially which decrease the calculation time after time stamp k=50
- After K=50 step the calculation decreases to less than 1 ms which is huge advantage to the robot with limited hardware resources
- Practical experiment verifies the simulation results