

**Summer School on SLAM**  
**August 2006, Oxford**

# **Advanced EKF-SLAM: Improving Consistency**

**Juan Domingo Tardós**  
**University of Zaragoza, Spain**

[robots.unizar.es](http://robots.unizar.es)

# EKF-SLAM: Consistency

## 1. Introduction

## 2. Consistency of EKF-SLAM

## 3. Robocentric Mapping

## 4. Sum of Gaussians Filter

## 5. Application: rescue of avalanche victims

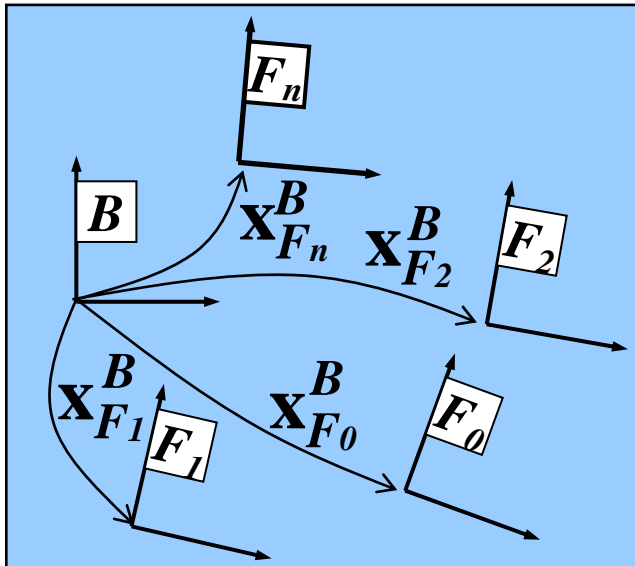
## 6. Conclusions

# EKF-SLAM approach

- Environment information related to a set of elements:

$$\mathcal{F} = \{B, F_0, F_1, \dots, F_n\} \quad F_0 = \text{Robot}$$

- represented by a map:  $\mathcal{M}_{\mathcal{F}}^B = (\hat{\mathbf{x}}_{\mathcal{F}}^B, \mathbf{P}_{\mathcal{F}}^B)$



$$\hat{\mathbf{x}}_{\mathcal{F}}^B = \begin{bmatrix} \hat{\mathbf{x}}_{F_0}^B \\ \vdots \\ \hat{\mathbf{x}}_{F_n}^B \end{bmatrix}$$

$$\mathbf{P}_{\mathcal{F}}^B = \begin{bmatrix} \mathbf{P}_{F_0 F_0}^B & \cdots & \mathbf{P}_{F_0 F_n}^B \\ \vdots & \ddots & \vdots \\ \mathbf{P}_{F_n F_0}^B & \cdots & \mathbf{P}_{F_n F_n}^B \end{bmatrix}$$

# EKF-SLAM

---

**Algorithm 1** SLAM:

---

$\mathbf{x}_0^B = \mathbf{0}; \mathbf{P}_0^B = \mathbf{0} \{Map\ initialization\}$

$[\mathbf{z}_0, \mathbf{R}_0] = \text{get\_measurements}$

$[\mathbf{x}_0^B, \mathbf{P}_0^B] = \text{add\_new\_features}(\mathbf{x}_0^B, \mathbf{P}_0^B, \mathbf{z}_0, \mathbf{R}_0)$

**for**  $k = 1$  to steps **do**

$[\mathbf{x}_{R_k}^{R_{k-1}}, \mathbf{Q}_k] = \text{get\_odometry}$

$[\mathbf{x}_{k|k-1}^B, \mathbf{P}_{k|k-1}^B] = \text{EKF\_prediction}(\mathbf{x}_{k-1}^B, \mathbf{P}_{k-1}^B, \mathbf{x}_{R_k}^{R_{k-1}}, \mathbf{Q}_k)$

$[\mathbf{z}_k, \mathbf{R}_k] = \text{get\_measurements}$

$\mathcal{H}_k = \text{data\_association}(\mathbf{x}_{k|k-1}^B, \mathbf{P}_{k|k-1}^B, \mathbf{z}_k, \mathbf{R}_k)$

$[\mathbf{x}_k^B, \mathbf{P}_k^B] = \text{EKF\_update}(\mathbf{x}_{k|k-1}^B, \mathbf{P}_{k|k-1}^B, \mathbf{z}_k, \mathbf{R}_k, \mathcal{H}_k)$

$[\mathbf{x}_k^B, \mathbf{P}_k^B] = \text{add\_new\_features}(\mathbf{x}_k^B, \mathbf{P}_k^B, \mathbf{z}_k, \mathbf{R}_k, \mathcal{H}_k)$

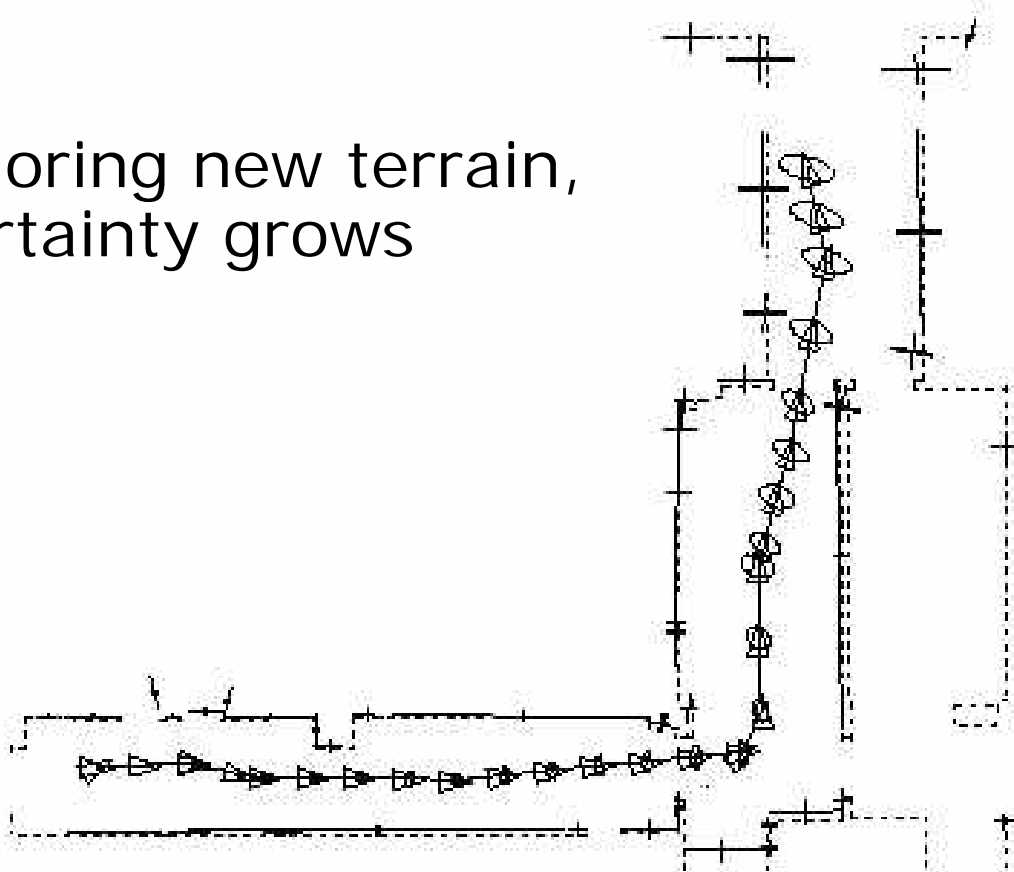
**end for**

---



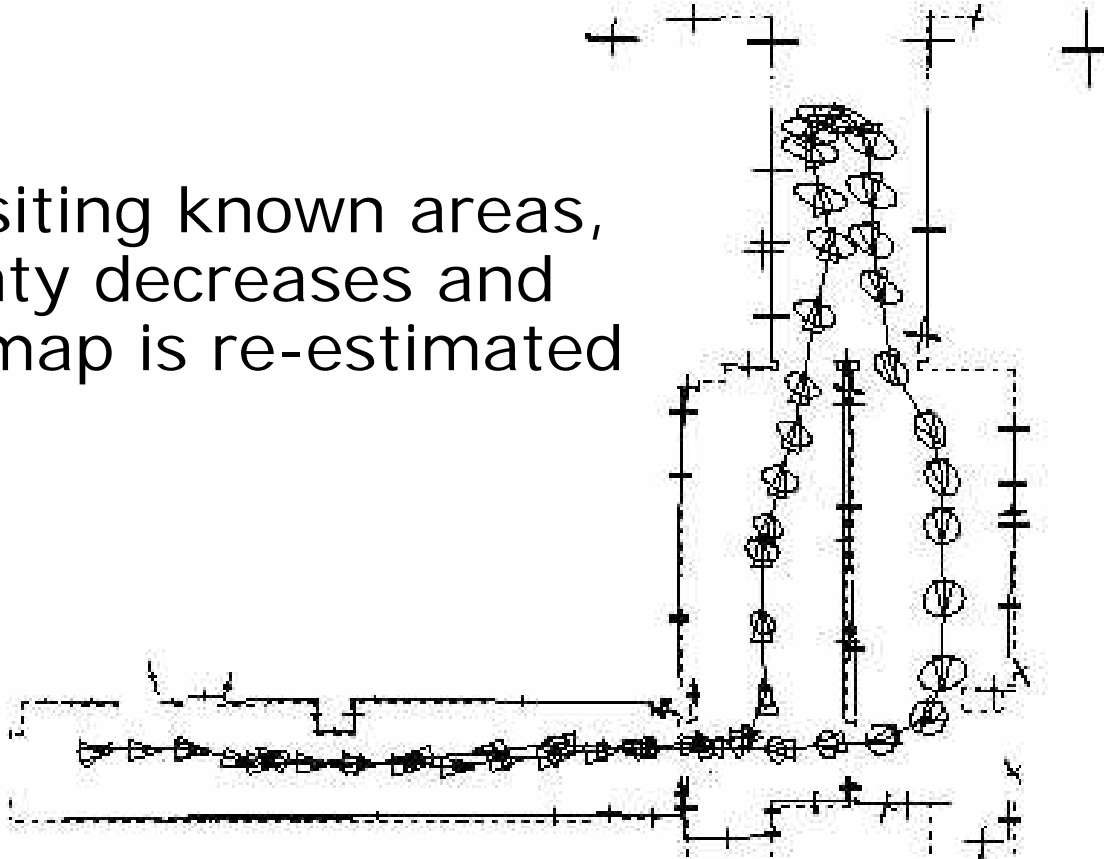
# EKF-SLAM: Exploration

When exploring new terrain,  
uncertainty grows



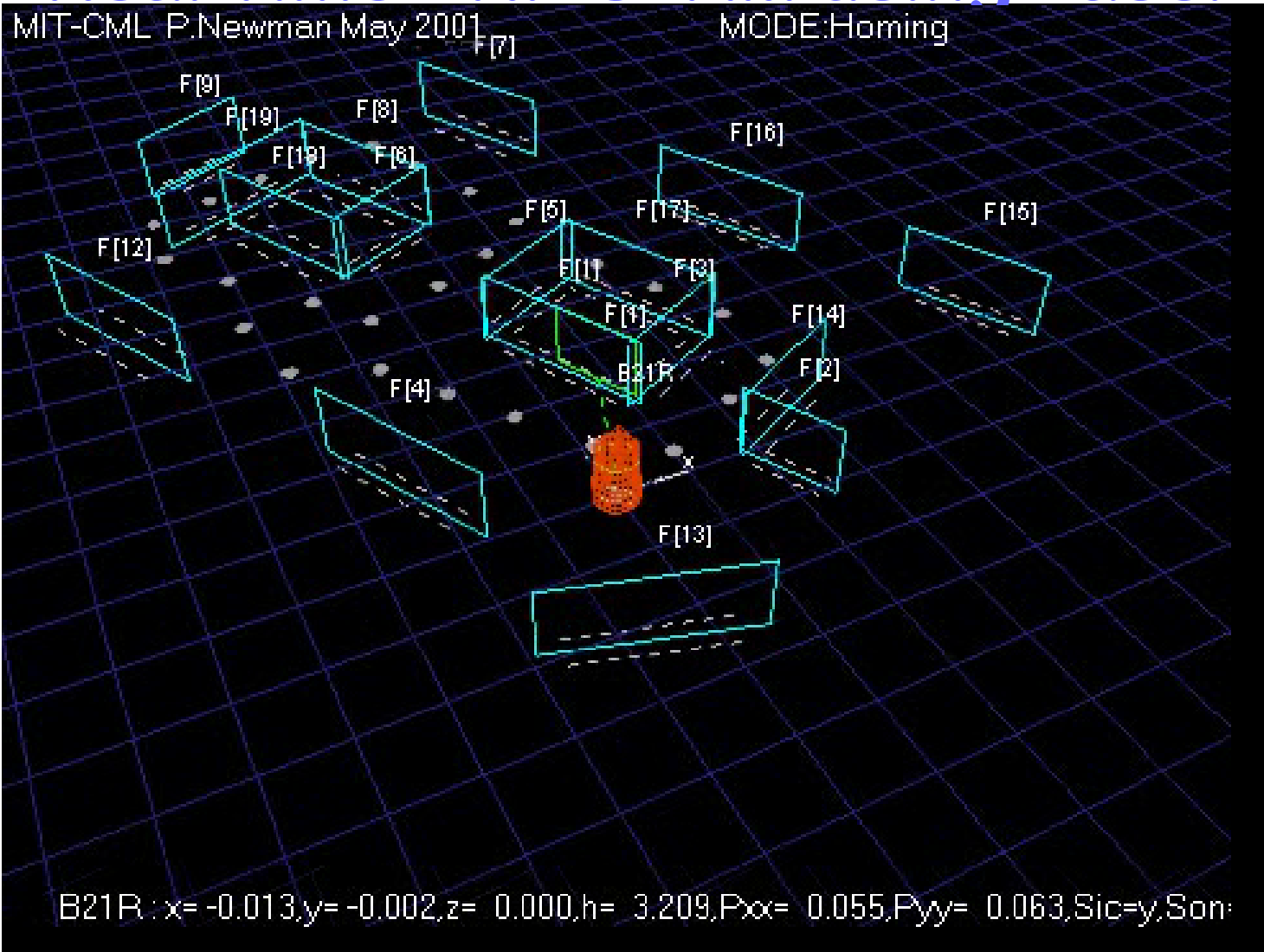
# EKF-SLAM: Loop closing (30m)

When revisiting known areas, uncertainty decreases and the whole map is re-estimated



J.A. Castellanos, J.M.M. Montiel, J. Neira and J.D. Tardós. The SPmap: A Probabilistic Framework for Simultaneous Localization and Map Building, IEEE Trans. Robotics and Automation, vol. 15 no. 5, pp. 948-953, Oct 1999.

# Real Time EKF-SLAM using Laser



P. M. Newman, J. J. Leonard, J. Neira and J.D. Tardós: Explore and Return: Experimental Validation of Real Time Concurrent Mapping and Localization. IEEE Int. Conf. Robotics and Automation, May 2002.

# EKF-SLAM: Consistency

1. Introduction

**2. Consistency of EKF-SLAM**

3. Robocentric Mapping

4. Sum of Gaussians Filter

5. Application: rescue of avalanche victims

6. Conclusions

# Consistency of EKF-SLAM

- Nice “convergence” properties of  $P_k^W$  (Dissanayake et al. 2001):
  - Landmark covariance decreases monotonically
  - In the limit, landmarks become fully correlated
  - In the limit, landmark covariance reaches a lower bound related to the initial vehicle covariance
- But SLAM is a non-linear problem
  - The inherent approximations due to linearizations can lead to **divergence (inconsistency)** of the EKF
    - » see for example (Jazwinski, 1970)

# EKF-SLAM: Robot Motion

$$\mathbf{x}_{R_k}^B = \mathbf{x}_{R_{k-1}}^B \oplus \mathbf{x}_{R_k}^{R_{k-1}}$$

Odometry model (white noise):

$$\begin{aligned}\mathbf{x}_{R_k}^{R_{k-1}} &= \hat{\mathbf{x}}_{R_k}^{R_{k-1}} + \mathbf{v}_k \\ E[\mathbf{v}_k] &= \mathbf{0} \\ E[\mathbf{v}_k \mathbf{v}_j^T] &= \delta_{kj} \mathbf{Q}_k\end{aligned}$$

EKF prediction:

$$\hat{\mathbf{x}}_{\mathcal{F}_{k|k-1}}^B = \begin{bmatrix} \hat{\mathbf{x}}_{R_{k-1}}^B \oplus \hat{\mathbf{x}}_{R_k}^{R_{k-1}} \\ \hat{\mathbf{x}}_{F_{1,k-1}}^B \\ \vdots \\ \hat{\mathbf{x}}_{F_{m,k-1}}^B \end{bmatrix} \quad \mathbf{F}_k = \begin{bmatrix} \mathbf{J}_{1 \oplus \left\{ \hat{\mathbf{x}}_{R_{k-1}}^B, \hat{\mathbf{x}}_{R_k}^{R_{k-1}} \right\}} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \mathbf{I} & & \vdots \\ \vdots & & \ddots & \\ \mathbf{0} & \cdots & & \mathbf{I} \end{bmatrix}$$

$$\mathbf{P}_{\mathcal{F}_{k|k-1}}^B = \mathbf{F}_k \mathbf{P}_{\mathcal{F}_{k-1}}^B \mathbf{F}_k^T + \mathbf{G}_k \mathbf{Q}_k \mathbf{G}_k^T \quad \mathbf{G}_k = \begin{bmatrix} \mathbf{J}_{2 \oplus \left\{ \hat{\mathbf{x}}_{R_{k-1}}^B, \hat{\mathbf{x}}_{R_k}^{R_{k-1}} \right\}} \\ \mathbf{0} \\ \vdots \\ \mathbf{0} \end{bmatrix}$$

Linearization

# EKF-SLAM: Map Update

Feature observations:

$$\mathbf{z}_k = \mathbf{h}_k(\mathbf{x}_{\mathcal{F}_k}^B) + \mathbf{w}_k$$

$$\mathbf{z}_k \simeq \mathbf{h}_k(\hat{\mathbf{x}}_{\mathcal{F}_{k|k-1}}^B) + \mathbf{H}_k(\mathbf{x}_{\mathcal{F}_k}^B - \hat{\mathbf{x}}_{\mathcal{F}_{k|k-1}}^B)$$

$$\mathbf{H}_k = \left. \frac{\partial \mathbf{h}_k}{\partial \mathbf{x}_{\mathcal{F}_k}^B} \right|_{(\hat{\mathbf{x}}_{\mathcal{F}_{k|k-1}}^B)} \quad \text{Linearization}$$

EKF map update:

$$\hat{\mathbf{x}}_{\mathcal{F}_k}^B = \hat{\mathbf{x}}_{\mathcal{F}_{k|k-1}}^B + \mathbf{K}_k(\mathbf{z}_k - \mathbf{h}_k(\hat{\mathbf{x}}_{\mathcal{F}_{k|k-1}}^B))$$

$$\mathbf{P}_{\mathcal{F}_k}^B = (\mathbf{I} - \mathbf{K}_k \mathbf{H}_k) \mathbf{P}_{\mathcal{F}_{k|k-1}}^B$$

$$\mathbf{K}_k = \mathbf{P}_{\mathcal{F}_{k|k-1}}^B \mathbf{H}_k^T (\mathbf{H}_k \mathbf{P}_{\mathcal{F}_{k|k-1}}^B \mathbf{H}_k^T + \mathbf{R}_k)^{-1}$$

$O(n^2)$

# The Consistency Problem

True map value

$$\mathbf{x}_k^W$$

EKF-SLAM estimation

$$\hat{\mathbf{x}}_k^W$$

$$\mathbf{P}_k^W$$

- An estimator is **consistent** if:

$$\begin{aligned} E \left[ \mathbf{x}_k^W - \hat{\mathbf{x}}_k^W \right] &= \mathbf{0} \\ E \left[ \left( \mathbf{x}_k^W - \hat{\mathbf{x}}_k^W \right) \left( \mathbf{x}_k^W - \hat{\mathbf{x}}_k^W \right)^T \right] &= \mathbf{P}_k^W \end{aligned}$$

Unbiased

The Mean Square Error  
matches the filter  
computed Covariance

- Pessimistic covariance is OK (but not too pessimistic)
- Optimistic covariance = Inconsistency = Filter divergence

# Consistency Testing

## 1. Normalized Estimation Error Squared NEES

$$D^2 = \left( \mathbf{x}_k^W - \hat{\mathbf{x}}_k^W \right)^T \left( \mathbf{P}_k^W \right)^{-1} \left( \mathbf{x}_k^W - \hat{\mathbf{x}}_k^W \right)$$

$$D^2 \leq \chi_{r,1-\alpha}^2$$

True map required  
→ Simulations

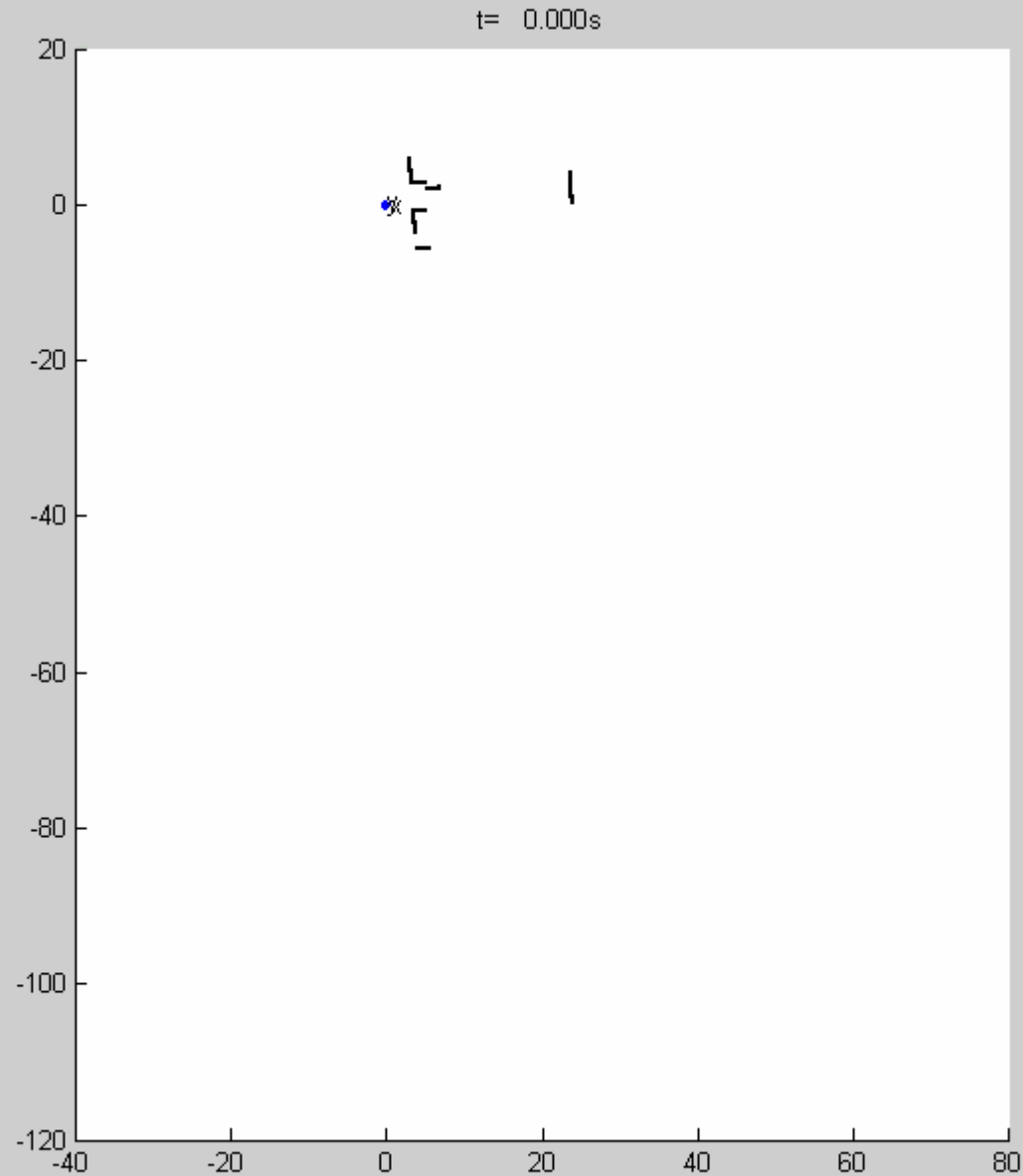
## 2. Innovation test (observation $i \rightarrow$ map feature $j$ )

$$D_{ij}^2 = \left( \mathbf{z}_i - \mathbf{h}_j(\hat{\mathbf{x}}_k^W) \right)^T \left( \mathbf{H}_j \mathbf{P}_k^W \mathbf{H}_j^T + \mathbf{R}_i \right)^{-1} \left( \mathbf{z}_i - \mathbf{h}_j(\hat{\mathbf{x}}_k^W) \right)$$

$$D_{ij}^2 \leq \chi_{d,1-\alpha}^2$$

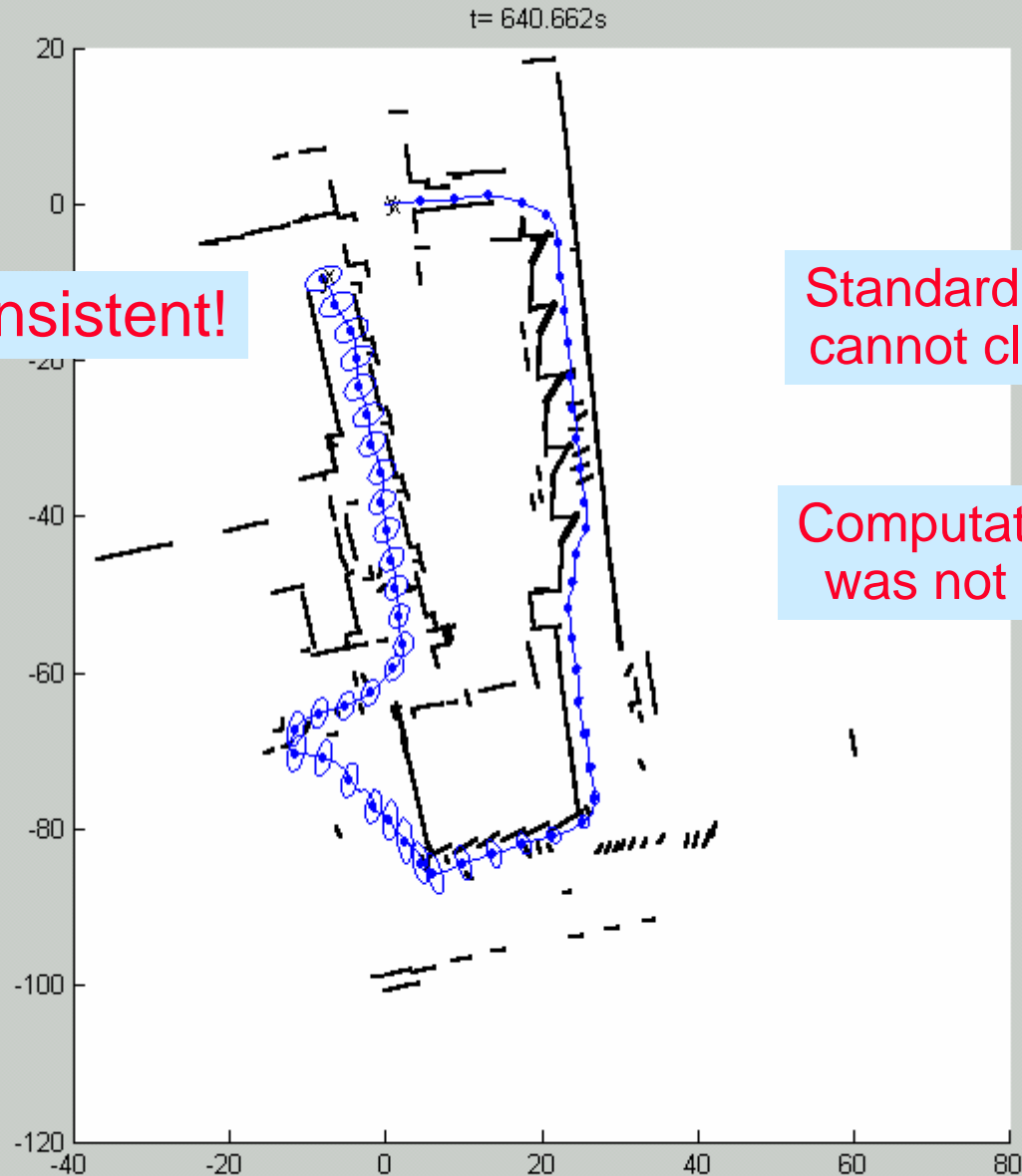
Critical when  
closing big loops

# EKF-SLAM: Real Example



# EKF-SLAM: Real Example

Inconsistent!



Standard data association cannot close a 250m loop

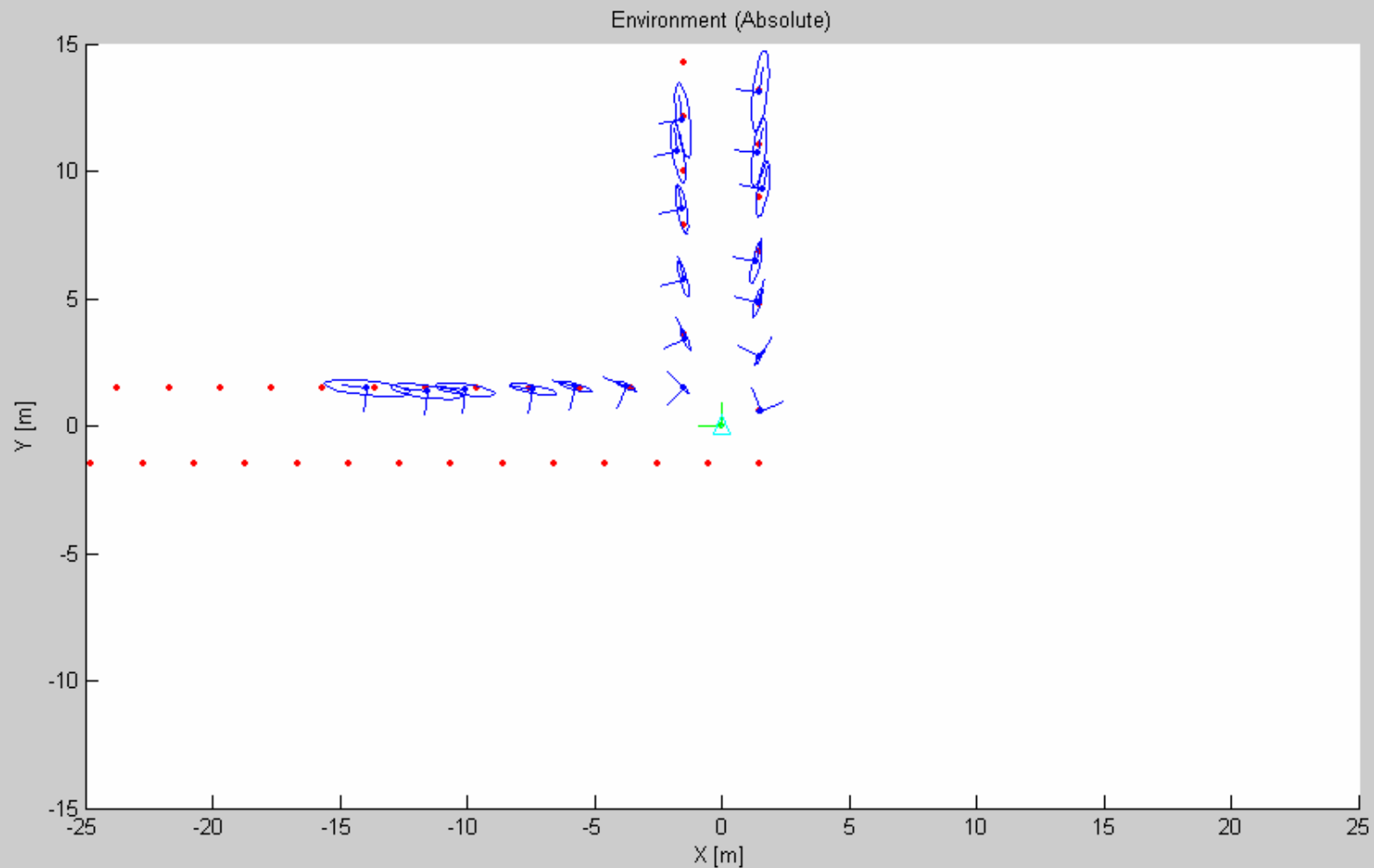
Computational complexity was not a problem here!

# EKF-SLAM: Simulation

- Simulation conditions
  - Perfect data association
  - Ideal odometry and measurement noise
    - » white, gaussian, known covariance
- Advantages of simulation:
  - Consistency can be tested against the true map
  - A simulation with noise=0 gives the theoretical map covariance (without linearization errors)

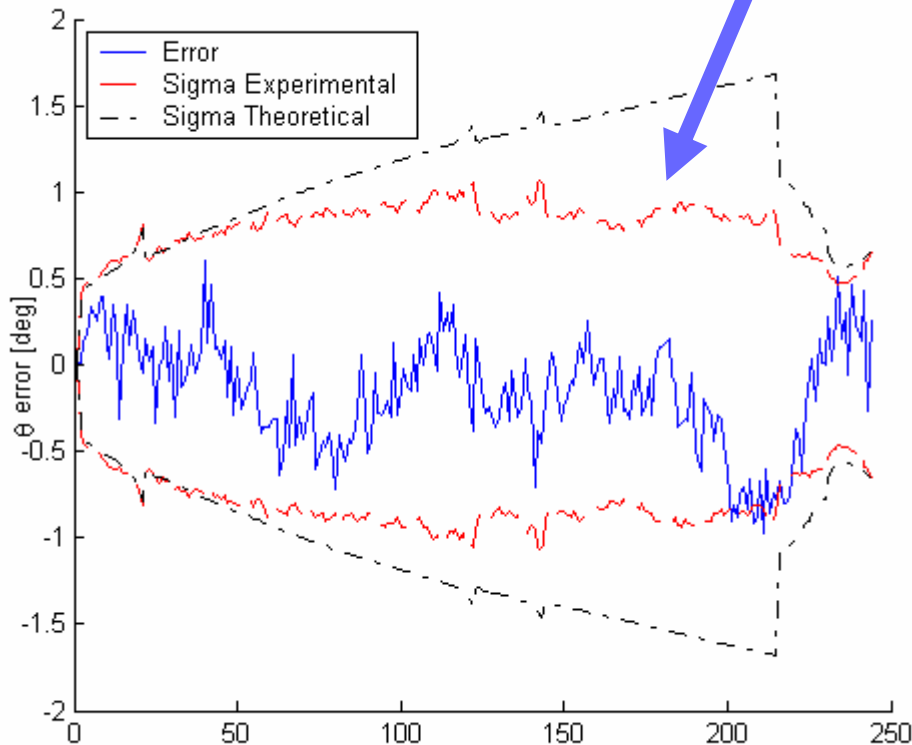
# EKF-SLAM: Simulation

Perfect data association  
and noise model



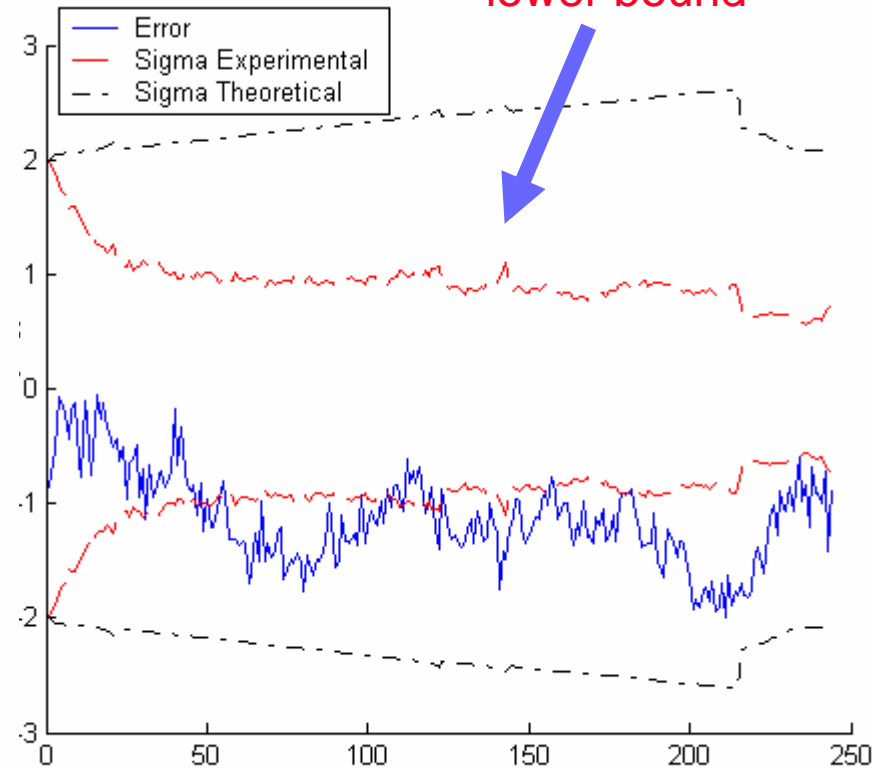
# EKF-SLAM: Covariance

Optimistic



Initial uncertainty = 0

Violates the lower bound



Initial uncertainty > 0

J.A. Castellanos, J. Neira, J.D. Tardós, Limits to the Consistency of EKF-based SLAM, 5th IFAC Symposium on Intelligent Autonomous Vehicles, Lisbon, July 2004

# EKF-SLAM: Consistency

1. Introduction
2. Consistency of EKF-SLAM
- 3. Robocentric Mapping**
4. Sum of Gaussians Filter
5. Application: rescue of avalanche victims
6. Conclusions

# Robocentric mapping

- Build the map relative to the robot
  - Uncertainty of the observed features is small, except for loop closing

$$\hat{\mathbf{x}}^R = \begin{bmatrix} \hat{\mathbf{x}}_W^R \\ \hat{\mathbf{x}}_{F_1}^R \\ \vdots \\ \hat{\mathbf{x}}_{F_n}^R \end{bmatrix}; \quad \mathbf{P}^R = \begin{bmatrix} \mathbf{P}_W^R & \cdots & \mathbf{P}_{WF_n}^R \\ \vdots & \ddots & \vdots \\ \mathbf{P}_{F_n W}^R & \cdots & \mathbf{P}_{F_n}^R \end{bmatrix}$$

# Robocentric mapping

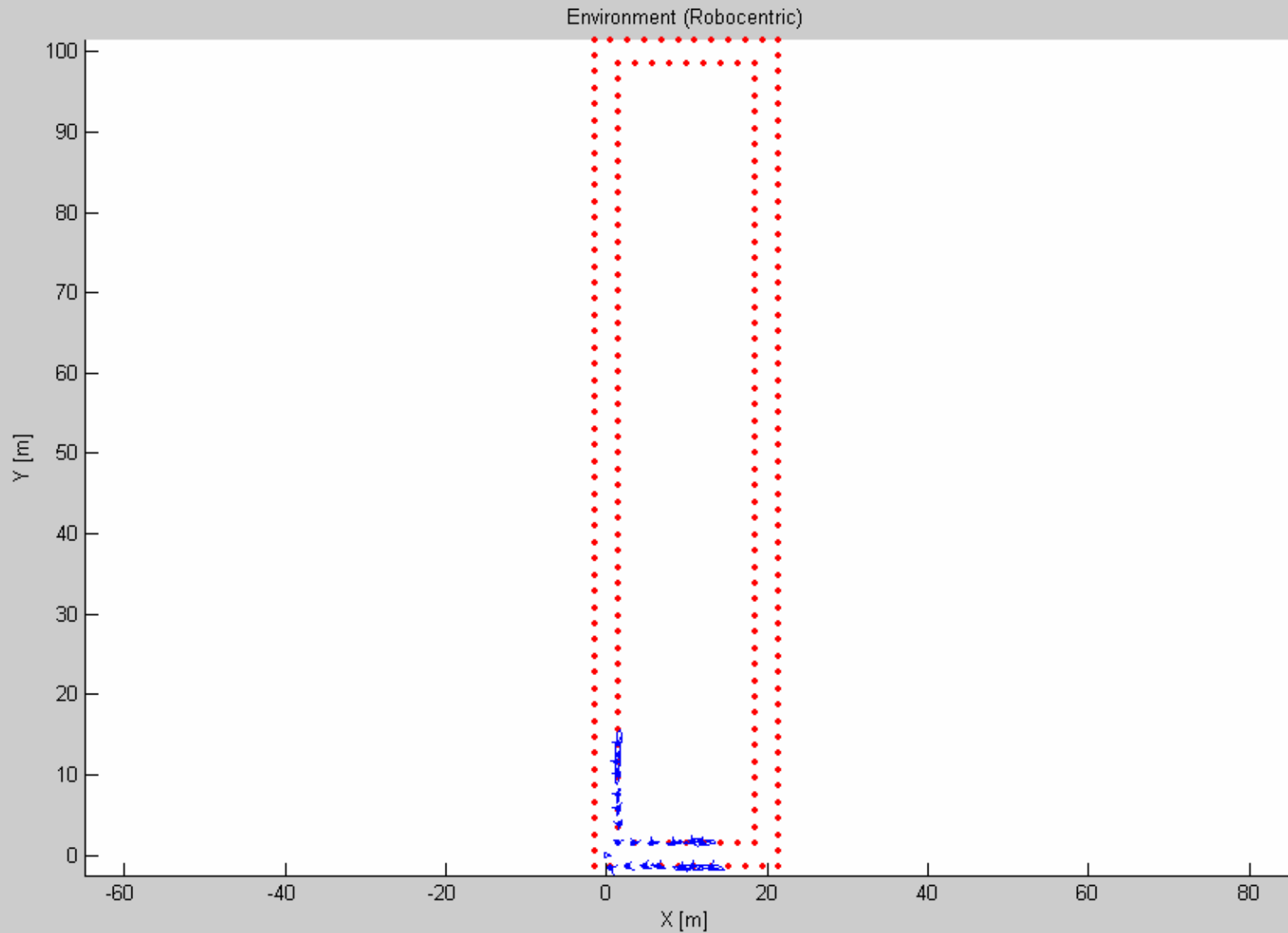
- Add the new motion to the map

$$\hat{\mathbf{x}}^{R_{k-1}} = \begin{bmatrix} \hat{\mathbf{x}}_W^{R_{k-1}} \\ \vdots \\ \hat{\mathbf{x}}_{F_n}^{R_{k-1}} \\ \hat{\mathbf{x}}_{R_k}^{R_{k-1}} \end{bmatrix}; \quad \mathbf{P}^{R_{k-1}} = \begin{bmatrix} \mathbf{P}_W^{R_{k-1}} & \cdots & \mathbf{P}_{WF_n}^{R_{k-1}} & 0 \\ \vdots & \ddots & \vdots & \vdots \\ \mathbf{P}_{F_n W}^{R_{k-1}} & \cdots & \mathbf{P}_{F_n}^{R_{k-1}} & 0 \\ 0 & \cdots & 0 & \mathbf{Q}_k \end{bmatrix}$$

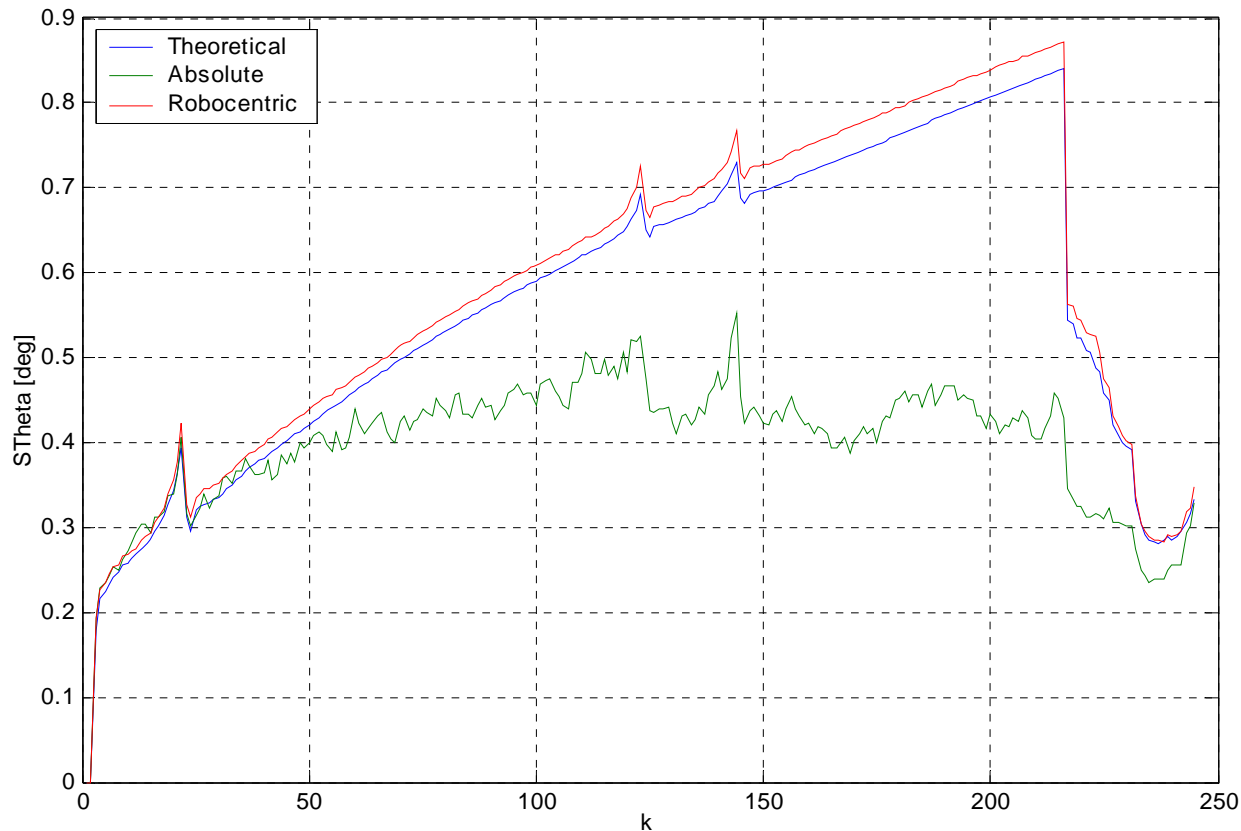
- Update map and motion using EKF
- Compose with updated motion (smaller linearization errors)

$$\hat{\mathbf{x}}_k^{R_k} = \begin{bmatrix} \ominus \hat{\mathbf{x}}_{R_k}^{R_{k-1}} \oplus \hat{\mathbf{x}}_{R_0}^{R_{k-1}} \\ \ominus \hat{\mathbf{x}}_{R_k}^{R_{k-1}} \oplus \hat{\mathbf{x}}_{F_1}^{R_{k-1}} \\ \vdots \\ \ominus \hat{\mathbf{x}}_{R_k}^{R_{k-1}} \oplus \hat{\mathbf{x}}_{F_n}^{R_{k-1}} \end{bmatrix}$$

# Robocentric mapping

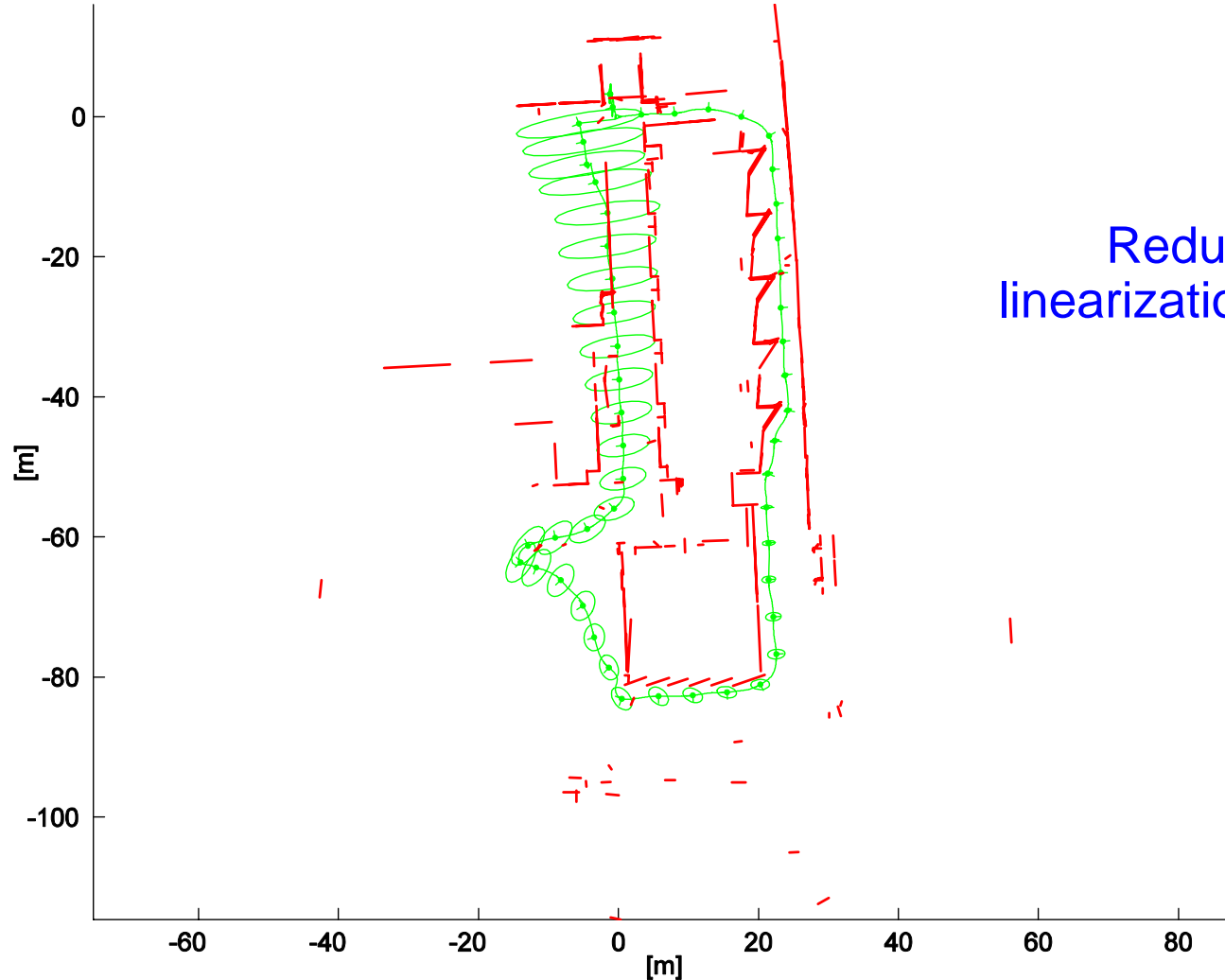


# Robocentric mapping



- Covariance is OK (slightly pessimistic)
- But there is some bias in the map → inconsistent

# Robocentric SLAM: real example



Reduces  
linearization errors

J. A. Castellanos, R. Martínez-Cantín, J. D. Tardós, J. Neira: "Robocentric Map Joining: Improving the Consistency of EKF-SLAM", Robotics and Autonomous Systems (to appear)

# EKF-SLAM: Consistency

1. Introduction
2. Consistency of EKF-SLAM
3. Robocentric Mapping
- 4. Sum of Gaussians Filter**
5. Application: rescue of avalanche victims
6. Conclusions

# Recursive Bayesian Estimation

$$\begin{aligned}
 p(\mathbf{x}_k | Z^{k-1}, U^{k-1}) & \stackrel{\text{Total Prob. Theor.}}{=} \int p(\mathbf{x}_k | \mathbf{x}_{k-1}, Z^{k-1}, U^{k-1}) p(\mathbf{x}_{k-1} | Z^{k-1}, U^{k-1}) d\mathbf{x}_{k-1} \\
 & \stackrel{\text{Markov + Causality}}{=} \int \underset{\substack{\uparrow \\ \text{Motion Model}}}{p(\mathbf{x}_k | \mathbf{x}_{k-1}, \mathbf{u}_{k-1})} \underset{\substack{\uparrow \\ \text{Information at k-1}}}{p(\mathbf{x}_{k-1} | Z^{k-1}, U^{k-1})} d\mathbf{x}_{k-1}
 \end{aligned}$$

$$\begin{aligned}
 \underset{\substack{\uparrow \\ \text{Información at k}}}{p(\mathbf{x}_k | Z^k, U^{k-1})} & \stackrel{\text{Bayes}}{=} \frac{p(\mathbf{z}_k | \mathbf{x}_k, Z^{k-1}, U^{k-1}) p(\mathbf{x}_k | Z^{k-1}, U^{k-1})}{p(\mathbf{z}_k | Z^{k-1}, U^{k-1})} \\
 & = \eta p(\mathbf{z}_k | \mathbf{x}_k, Z^{k-1}, U^{k-1}) p(\mathbf{x}_k | Z^{k-1}, U^{k-1}) \\
 & \stackrel{\text{Markov}}{=} \eta \underset{\substack{\uparrow \\ \text{Sensor Model}}}{p(\mathbf{z}_k | \mathbf{x}_k)} p(\mathbf{x}_k | Z^{k-1}, U^{k-1})
 \end{aligned}$$

# Recursive Bayesian Estimation

$$\begin{aligned} p_{k|k-1} &= \int p(\mathbf{x}_k | \mathbf{x}_{k-1}, \mathbf{u}_{k-1}) p_{k-1} d\mathbf{x}_{k-1} \\ p_k &= \eta p(\mathbf{z}_k | \mathbf{x}_k) p_{k|k-1} \end{aligned}$$

Prediction

Update

- No general closed-form solution
- If Linear-Gaussian  $\rightarrow$  Kalman Filter

$\mathbf{f}_k, \mathbf{h}_k$  lineales  
 $\mathbf{w}_k, \mathbf{v}_k$  AWGN indep.  $\left| \longrightarrow \right.$  All pdf are Gaussian

- Non-Linear case  $\rightarrow$  EKF
  - Linear approximation of motion and sensor models
  - Aproximates pdfs using the mean and covariance

# Estimation techniques for mobile robot localization

- Histogram Filter (Markov Localization)
  - State space split in cells
  - Aproximates pdf by a constant value in each cell

## ➡ EKF: Extended Kalman Filter

- Linear approximation of motion and sensor models
- Aproximates pdf using the mean and covariance

## ➡ Particle Filter (Monte-Carlo Localization)

- Pdf approximated by a set of samples

S. Thrun, W. Burgard, D. Fox: Probabilistic Robotics, MIT Press, 2005

# EKF

$$\begin{aligned} p_{k|k-1} &= \int p(\mathbf{x}_k | \mathbf{x}_{k-1}, \mathbf{u}_{k-1}) p_{k-1} d\mathbf{x}_{k-1} \\ p_k &= \eta p(\mathbf{z}_k | \mathbf{x}_k) p_{k|k-1} \end{aligned}$$

Prediction

Update

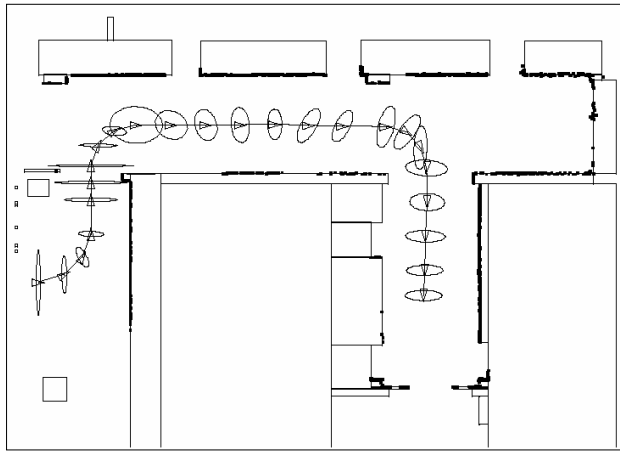
- Linear approximation of sensor and motion equations
- Approximates pdf using the mean and covariance
- Very efficient

$$\begin{aligned} \hat{\mathbf{x}}_{k|k-1} &= \mathbf{f}_k(\hat{\mathbf{x}}_{k-1|k-1}, \mathbf{u}_{k-1}) \\ P_{k|k-1} &= F_k P_{k-1|k-1} F_k^T + G_k Q_k G_k^T \\ \hat{\mathbf{x}}_{k|k} &= \hat{\mathbf{x}}_{k|k-1} + K_k (\mathbf{z}_k - \mathbf{h}_k(\hat{\mathbf{x}}_{k|k-1})) \\ P_{k|k} &= (I - K_k H_k) P_{k|k-1} \\ K_k &= P_{k|k-1} H_k^T (H_k P_{k|k-1} H_k^T + R_k)^{-1} \end{aligned}$$

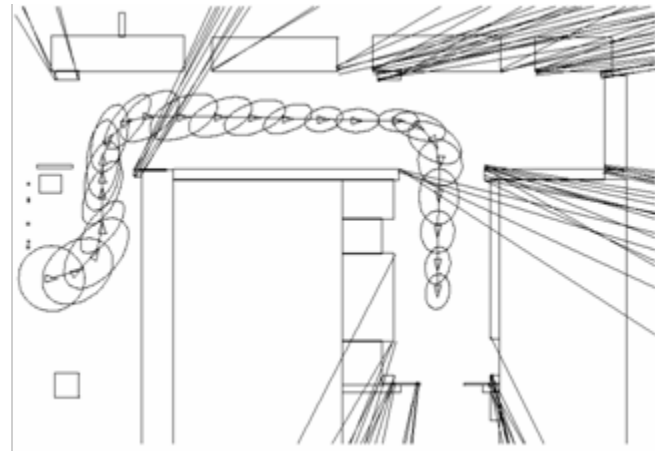
$$F_k = \left. \frac{\partial \mathbf{f}_k}{\partial \mathbf{x}} \right|_{(\hat{\mathbf{x}}_{k-1|k-1})}$$

$$H_k = \left. \frac{\partial \mathbf{h}_k}{\partial \mathbf{x}} \right|_{(\hat{\mathbf{x}}_{k|k-1})}$$

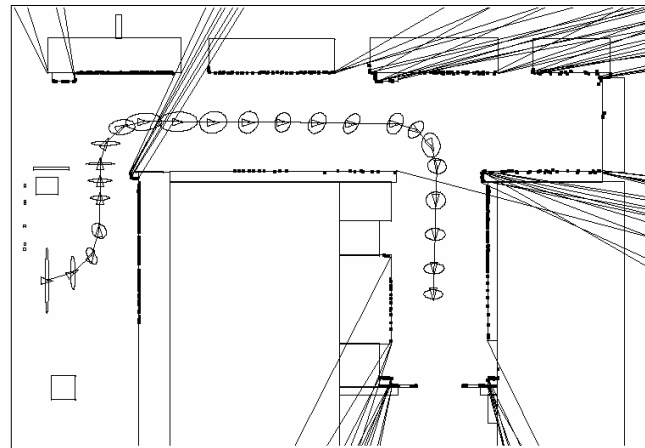
# Multisensor Localization using EKF



**Láser**



**Visión**



**Láser + Visión**

J. Neira, J.D. Tardós, J. Horn and G. Schmidt: Fusing Range and Intensity Images for Mobile Robot Localization, IEEE Trans. Robotics and Automation 15 (1), Feb 1999, pp 76-84.

# Particle Filter

$$\begin{aligned} p_{k|k-1} &= \int p(\mathbf{x}_k | \mathbf{x}_{k-1}, \mathbf{u}_{k-1}) p_{k-1} d\mathbf{x}_{k-1} \\ p_k &= \eta p(\mathbf{z}_k | \mathbf{x}_k) p_{k|k-1} \end{aligned}$$

Prediction

Update

- Approximates distribution by a set of samples
- It is important to maintain particle diversity

**for**  $i = 1$  **to**  $n$  **do**

sample  $\mathbf{x}_{k|k-1}^{(i)} \sim p(\mathbf{x}_k | \mathbf{x}_{k-1}^{(i)}, \mathbf{u}_{k-1})$

$w^{(i)} = p(\mathbf{z}_k | \mathbf{x}_{k|k-1}^{(i)})$

**end for**

**for**  $i = 1$  **to**  $n$  **do**

draw  $\mathbf{x}_{k|k}^{(i)}$  from  $\{\mathbf{x}_{k|k-1}^{(j)}\}$  with probability  $\propto w^{(j)}$

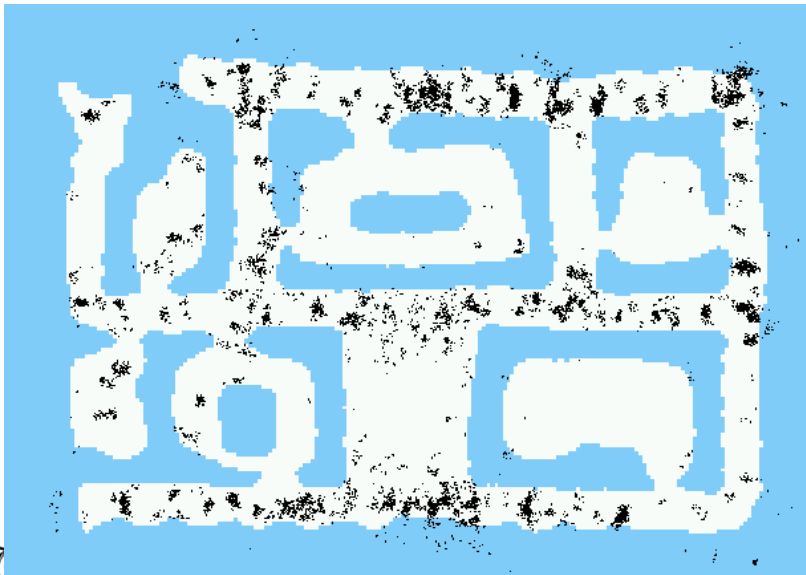
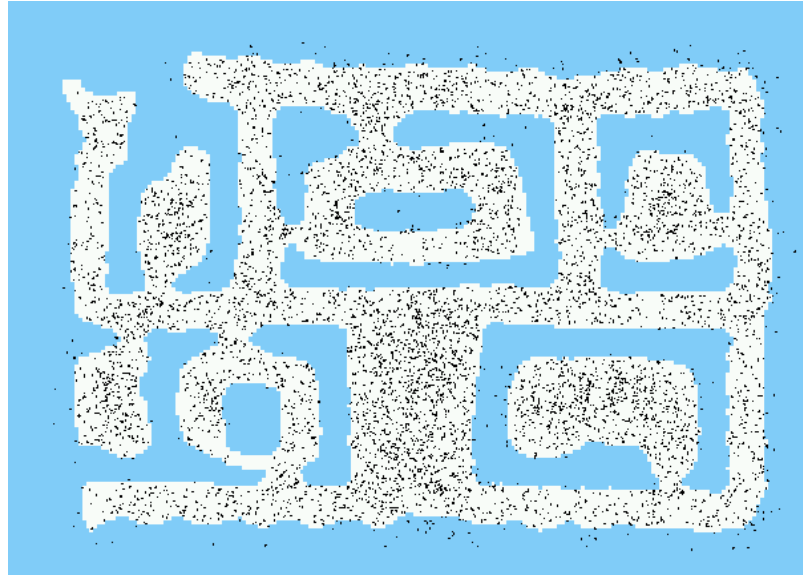
**end for**

Prediction

Weight update

Resampling

# Particle Filter (Thrun et al., 2005)



# Sum of Gaussians (SOGs)

D.L. Alspach, H.W. Sorenson: "Nonlinear bayesian estimation using Gaussian sum approximations", IEEE Trans. Automatic Control, vol. AC-17, pp 439-448, 1972

- Any pdf can be approximated by:

$$\hat{p}(\mathbf{x}) = \sum_{i=1}^n w^{(i)} N(\mathbf{x}; \bar{\mathbf{x}}^{(i)}, P^{(i)})$$

$$\sum_{i=1}^n w^{(i)} = 1 \quad , \quad w^{(i)} \geq 0$$

- Uniform convergence when  $n \rightarrow \infty$  and  $P^{(i)} \rightarrow 0$
- Used for Multi-Hypothesis tracking with EKF
- Used for sea bed modelling in underwater robotics

H. Durrant-whyte, S. Majumder, S. Thrun, M. de Battista, S. Scheduling: "A bayesian algorithm for simultaneous localisation and map building", Int. Symp. Robotics Research, 2003

# Sum of Gaussians Filter

$$p_{k|k-1} = \int p(\mathbf{x}_k | \mathbf{x}_{k-1}, \mathbf{u}_{k-1}) p_{k-1} d\mathbf{x}_{k-1}$$

$$p_k = \eta p(\mathbf{z}_k | \mathbf{x}_k) p_{k|k-1}$$

Prediction

Update

- Pdf approximated by a Sum of Gaussians (SoG)
- Each Gaussian is updated with an EKF

for  $i = 1$  to  $N$  do

$$F_k = \left. \frac{\partial \mathbf{f}_k}{\partial \mathbf{x}} \right|_{(\hat{\mathbf{x}}_{k-1|k-1}^{(i)})}$$

$$\hat{\mathbf{x}}_{k|k-1}^{(i)} = \mathbf{f}_k(\hat{\mathbf{x}}_{k-1|k-1}^{(i)}, \mathbf{u}_{k-1})$$

$$P_{k|k-1}^{(i)} = F_k P_{k-1|k-1}^{(i)} F_k^T + G_k Q_k G_k^T$$

$$H_k = \left. \frac{\partial \mathbf{h}_k}{\partial \mathbf{x}} \right|_{(\hat{\mathbf{x}}_{k|k-1}^{(i)})}$$

$$K_k = P_{k|k-1}^{(i)} H_k^T (H_k P_{k|k-1}^{(i)} H_k^T + R_k)^{-1}$$

$$\hat{\mathbf{x}}_{k|k}^{(i)} = \hat{\mathbf{x}}_{k|k-1}^{(i)} + K_k (\mathbf{z}_k - \mathbf{h}_k(\hat{\mathbf{x}}_{k|k-1}^{(i)}))$$

$$P_{k|k}^{(i)} = (I - K_k H_k) P_{k|k-1}^{(i)}$$

$$w_k^{(i)} = \eta w_{k-1}^{(i)} N(\mathbf{z}_k; \mathbf{h}_k(\hat{\mathbf{x}}_{k|k-1}^{(i)}), H_k P_{k|k-1}^{(i)} H_k^T + R_k)$$

end for

EKF Prediction

EKF Update

Weight Update

# SOG Filter .vs. EKF

- Advantages of SOG Filter
  - Allows to represent multimodal distributions
  - Motion and measurement equations are linearized in different points for each Gaussian
    - » If each Gaussian is small, the linear approximation is precise
    - » More adequate than EKF for highly non-linear systems

# SOGs .vs. Particle Filter

- SOG Filter

$$w_k^{(i)} = \eta w_{k-1}^{(i)} N(\mathbf{z}_k; \mathbf{h}_k(\hat{\mathbf{x}}_{k|k-1}^{(i)}), H_k P_{k|k-1}^{(i)} H_k^T + R_k)$$

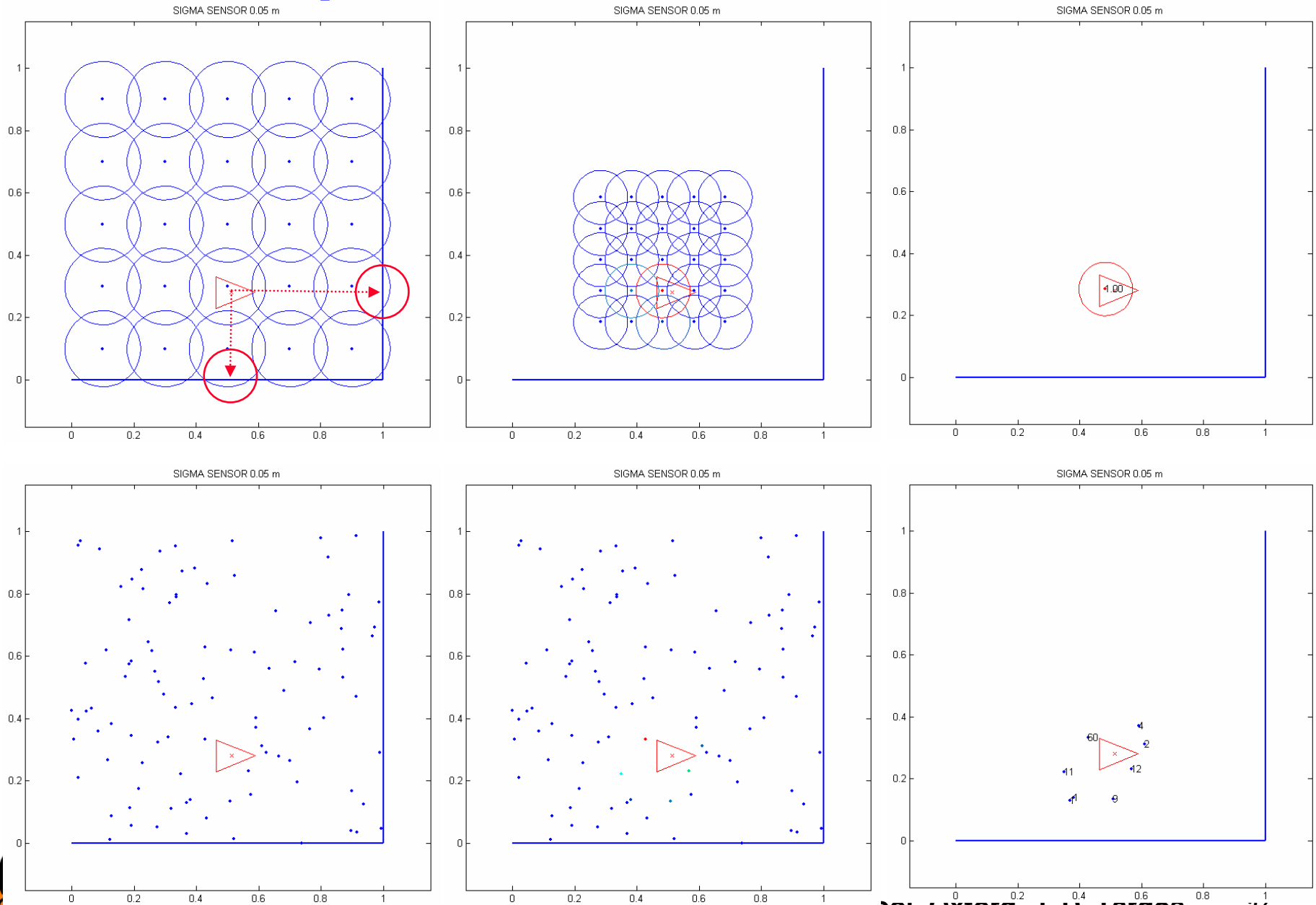
- EKF takes into account the prediction and sensor uncertainty
- Each Gaussian covers a portion of the state space

- Particle Filter

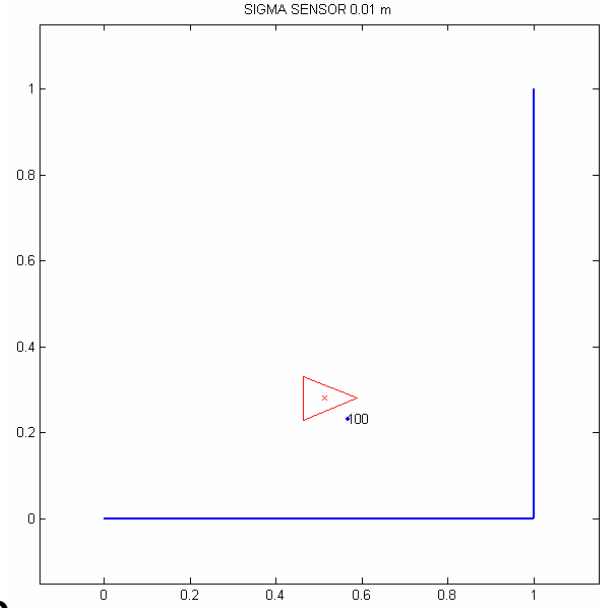
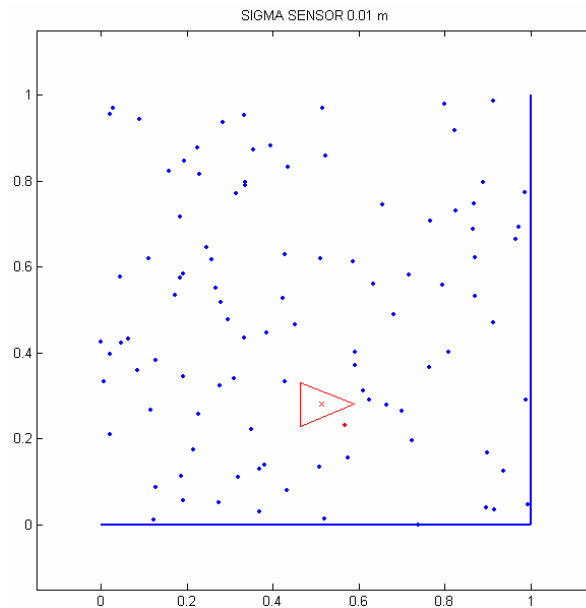
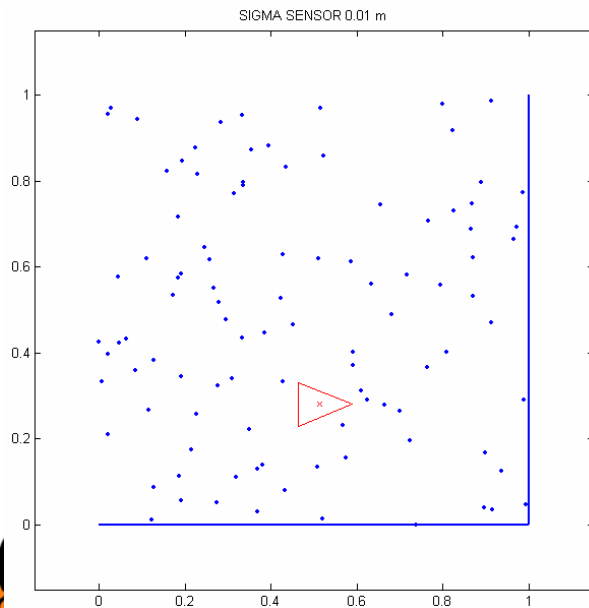
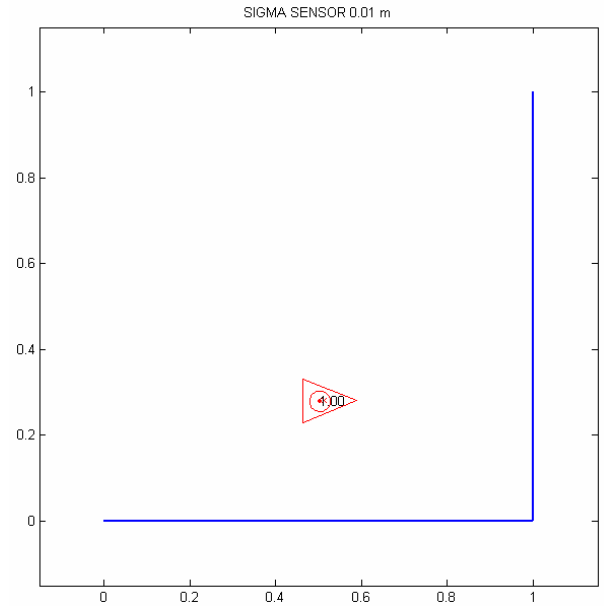
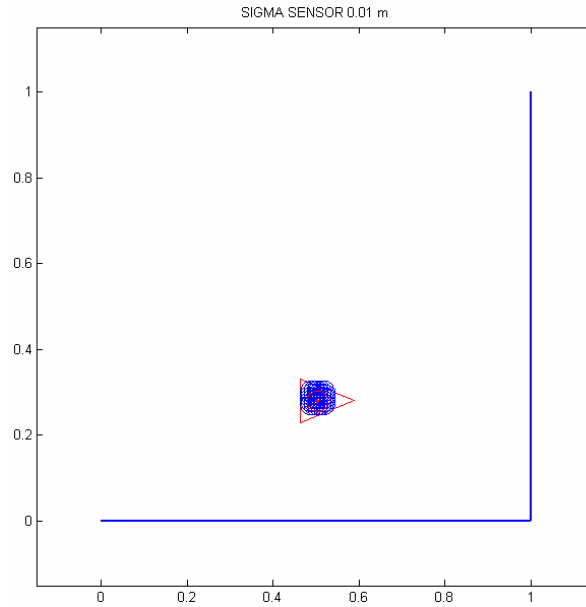
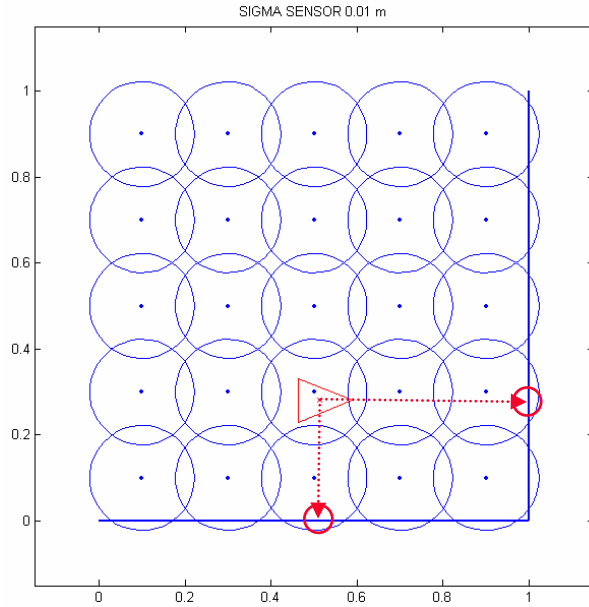
$$w_k^{(i)} = p(\mathbf{z}_k | \mathbf{x}_{k|k-1}^{(i)}) = N(\mathbf{z}_k; \mathbf{h}_k(\mathbf{x}_{k|k-1}^{(i)}), R_k)$$

- Each particle has no uncertainty
- If no particle is close enough to the true location, all particles obtain zero weight → “particle depletion”
- Works better with bad sensors:  $R_k \gg H_k P_{k|k-1}^{(i)} H_k^T$ 
  - » Simple solution: Inflate sensor noise
  - » More complex solution: Mixture proposal distribution
  - » See (Thrun et al., 2005) for a discussion

# Example: SOGs .vs. Particle Filter



# Example: SOGs .vs. Particle Filter

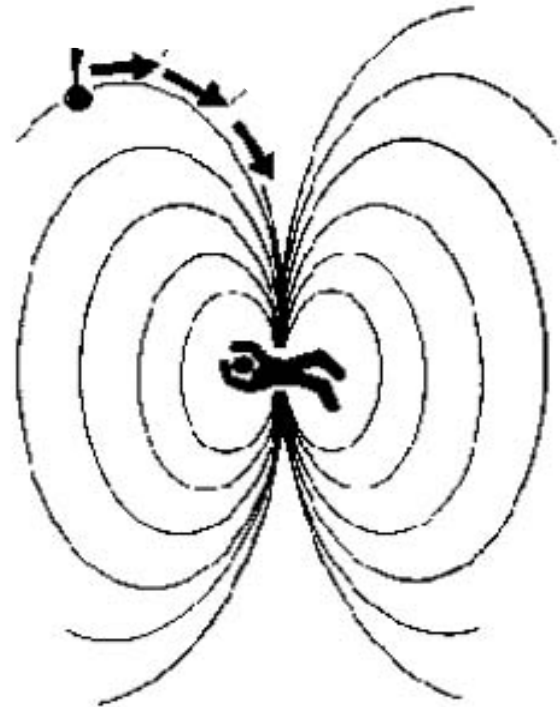


# EKF-SLAM: Consistency

1. Introduction
2. Consistency of EKF-SLAM
3. Robocentric Mapping
4. Sum of Gaussians Filter
- 5. Aplication: rescue of avalanche victims**
  - Classical search techniques
  - Our approach: 3D Robocentric SLAM
  - Solution using Sum of Gaussians
  - Simulation results
6. Conclusions

# Localization of avalanche victims

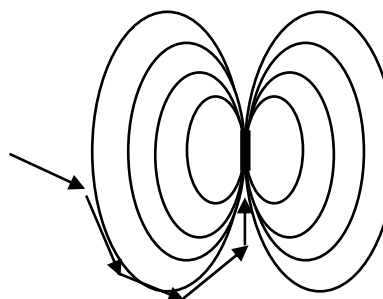
- Survival rate:
  - After 15 minutes: 93%
  - After 45 minutes: 25%
- ARVA: Appareil de Recherche de Victimes d'Avalanches
  - Magnetic field at 457 KHz (standard EN 300 718)



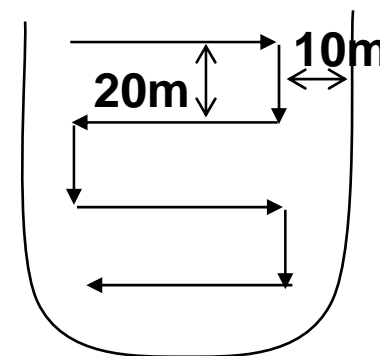
# ARVA: Classical Search

- Primary Search
  - Detect first signal (~20-40m)
- Secondary Search
  - Approach victim (~2-3m)
- Tertiary Search
  - Precise victim location
- Search body with the snow probe
- Shovel the snow

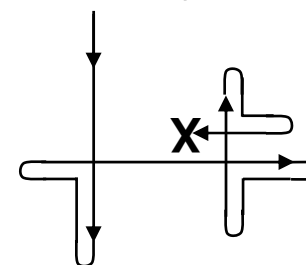
Secondary Search



Primary Search



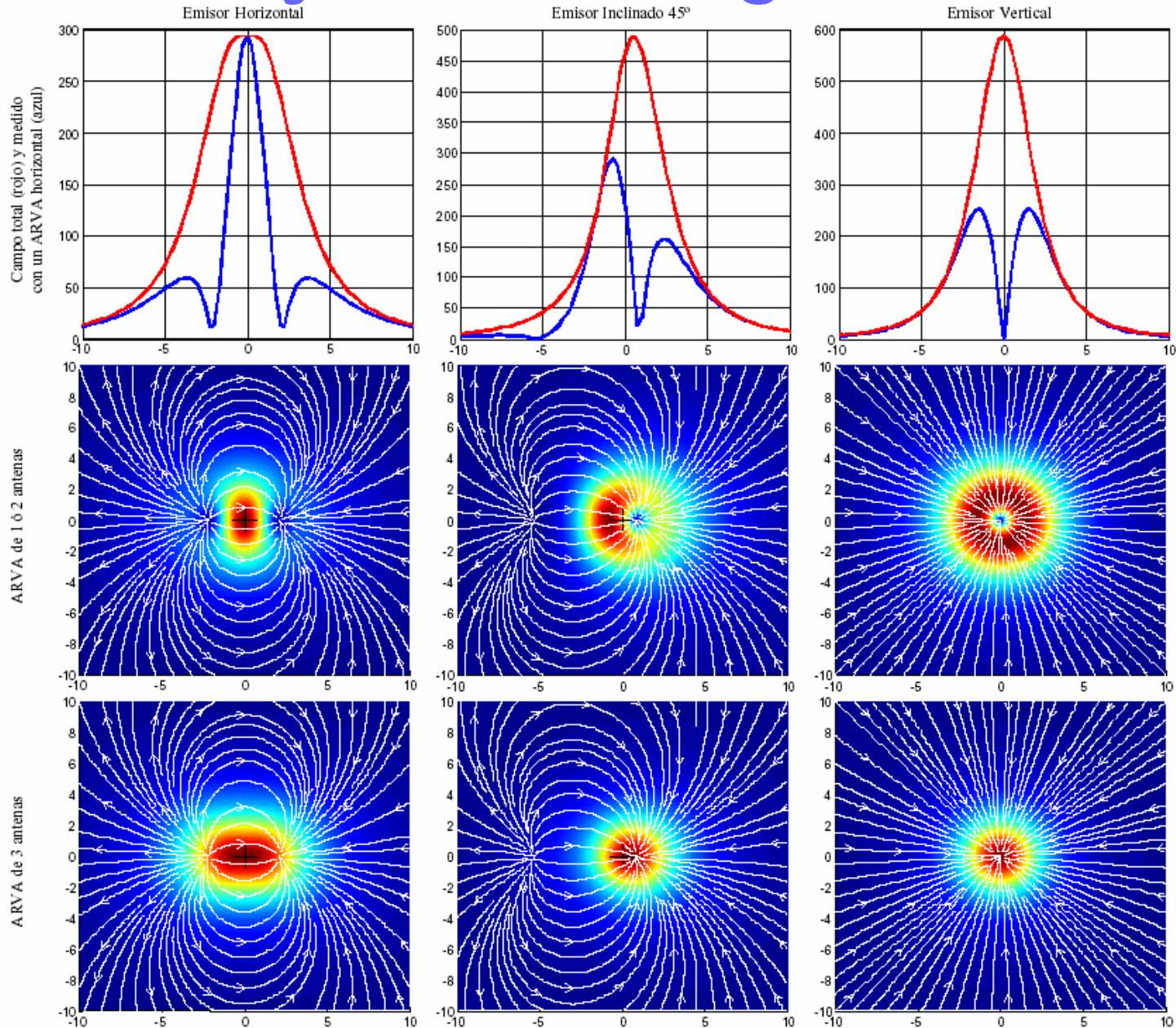
Tertiary Search



Require good training

Some novice users are  
unable to find the victim

# Tertiary Search: 2 signal maxima



# Approach: 3D SLAM problem

- State of receiver and transmitter (victim):

$$\mathbf{x}_r = \begin{bmatrix} x_r \\ y_r \\ z_r \\ \psi_r \\ \theta_r \\ \phi_r \end{bmatrix} \quad \mathbf{x}_t = \begin{bmatrix} x_t \\ y_t \\ z_t \\ m_x \\ m_y \\ m_z \end{bmatrix}$$

$$\mathcal{M} = (\hat{\mathbf{x}}, \mathbf{P})$$

$$\hat{\mathbf{x}} = \begin{bmatrix} \hat{\mathbf{x}}_r \\ \hat{\mathbf{x}}_{t_1} \\ \vdots \\ \hat{\mathbf{x}}_{t_n} \end{bmatrix} \quad \mathbf{P} = \begin{bmatrix} \mathbf{P}_{rr} & \cdots & \mathbf{P}_{rt_n} \\ \vdots & \ddots & \vdots \\ \mathbf{P}_{t_nr} & \cdots & \mathbf{P}_{t_nt_n} \end{bmatrix}$$

# Measurement equation

- Theoretical magnetic field:

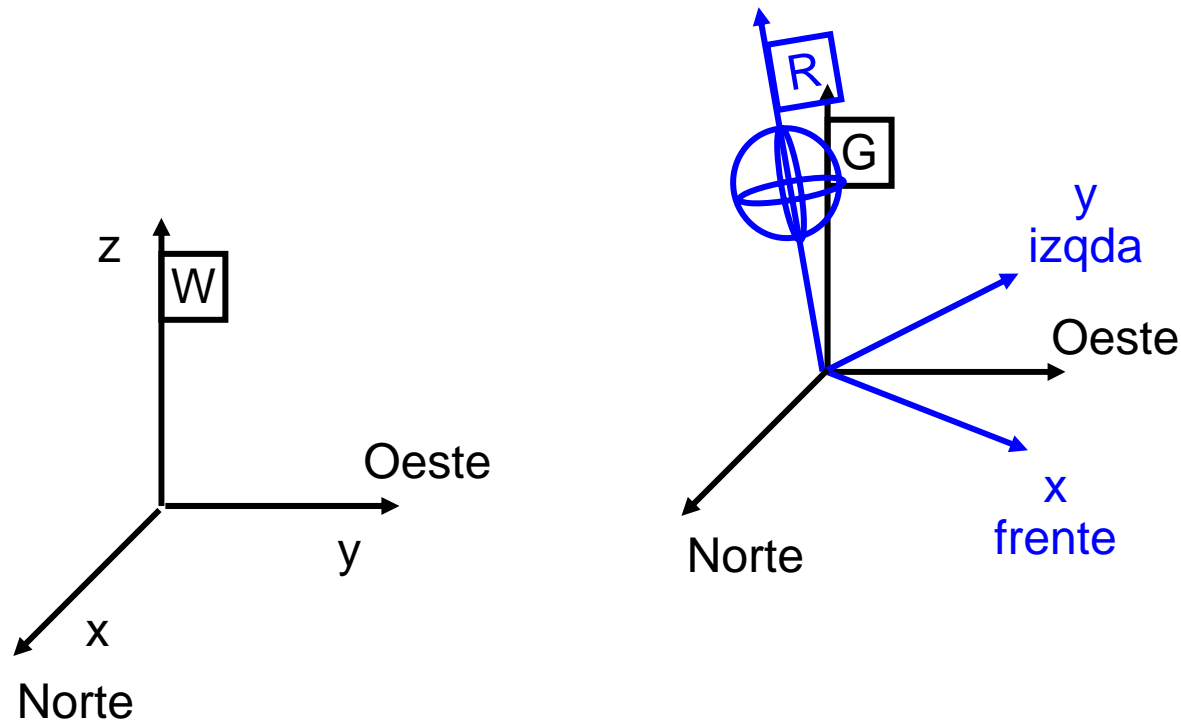
$$\mathbf{h}_k(\mathbf{x}_r, \mathbf{x}_t) = \begin{bmatrix} H_x \\ H_y \\ H_z \end{bmatrix} = \frac{1}{4\pi r^3} \left[ \frac{3(\mathbf{m}\mathbf{r})\mathbf{r}}{r^2} - \mathbf{m} \right]$$

$$\mathbf{r} = \begin{bmatrix} x_r \\ y_r \\ z_r \end{bmatrix} - \begin{bmatrix} x_t \\ y_t \\ z_t \end{bmatrix} \quad \text{Very non-linear}$$

- Plus white Gaussian noise:

$$\begin{aligned} \mathbf{z}_k &= \mathbf{h}_k(\mathbf{x}_k) + \mathbf{w}_k \\ \mathbf{w}_k &\sim N(0_{3 \times 1}, R_k) \end{aligned} \quad R_k = \begin{pmatrix} \sigma_x^2 & 0 & 0 \\ 0 & \sigma_y^2 & 0 \\ 0 & 0 & \sigma_z^2 \end{pmatrix}$$

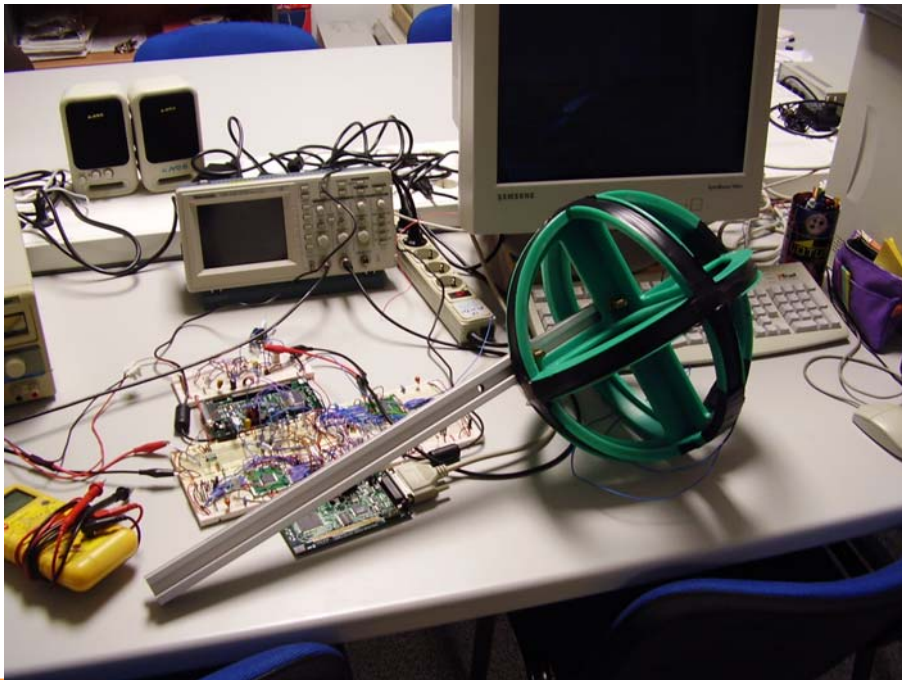
# 3D robocentric SALM



- Map built relative to reference G, that moves with the rescuer's antenna R, without turning

# Prototype

- 3D antenna
- GPS
- Orientation sensor
- Backpack with PC
- Hand-held interface



# First Solution: Non-linear optimization

- Simplified 2D case with known rescuer location
- Seed difficult to obtain: 2 measurements, no closed form solution known.
- Non-linear quadratic optimization
  - MATLAB lsqnonlin: Levenberg-Marquardt

$$\min_{\mathbf{x}_t} f(\mathbf{x}_t) = f_1(\mathbf{x}_t) + \sum_i (\mathbf{z}_i - \mathbf{h}_i(\mathbf{x}_t))^T R_i^{-1} (\mathbf{z}_i - \mathbf{h}_i(\mathbf{x}_t))$$

$$f_1(\mathbf{x}_t) = \left( \frac{m_{nominal} - m}{(m_{max} - m_{min})/4} \right)^2$$

- Emission power defined by norm (EN 300 718)

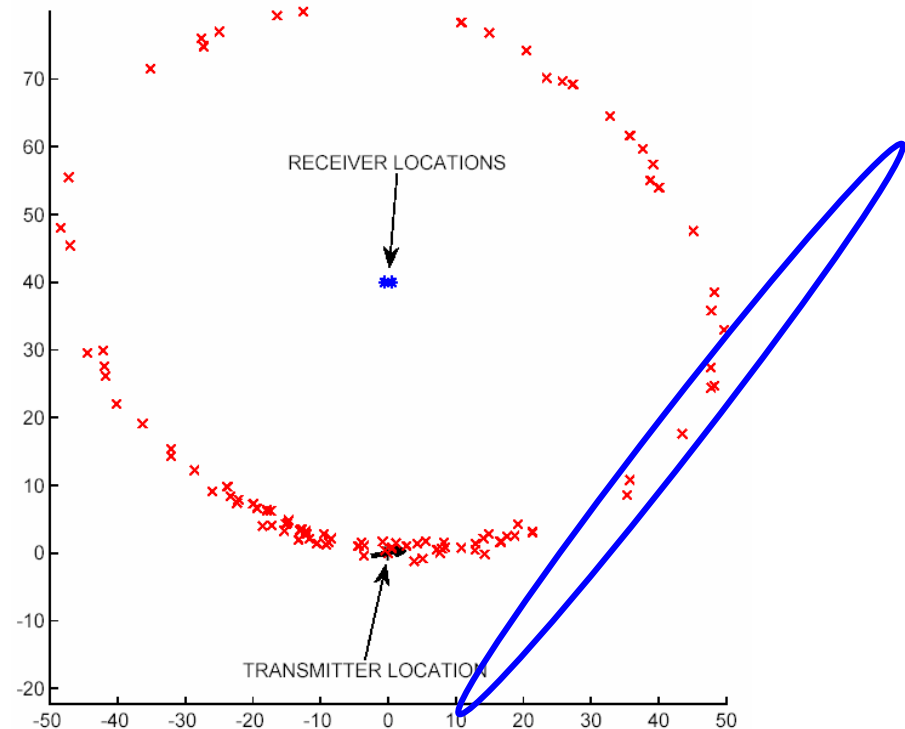
# First Solution: Non-linear optimization

Monte-Carlo simulation (100 runs):

- 2 measurements with 1m baseline
- Using true value as seed

Conclusions:

- Slow convergence
- pdf is NOT Gaussian
- EKF would not work



# Second solution: Histogram Filter

$$\begin{aligned} p_{k|k-1} &= \int p(\mathbf{x}_k | \mathbf{x}_{k-1}, \mathbf{u}_{k-1}) p_{k-1} d\mathbf{x}_{k-1} \\ p_k &= \eta p(\mathbf{z}_k | \mathbf{x}_k) p_{k|k-1} \end{aligned}$$

Prediction

Update

- Discretize search space in cells
- Approximates pdf by a constant value in each cell

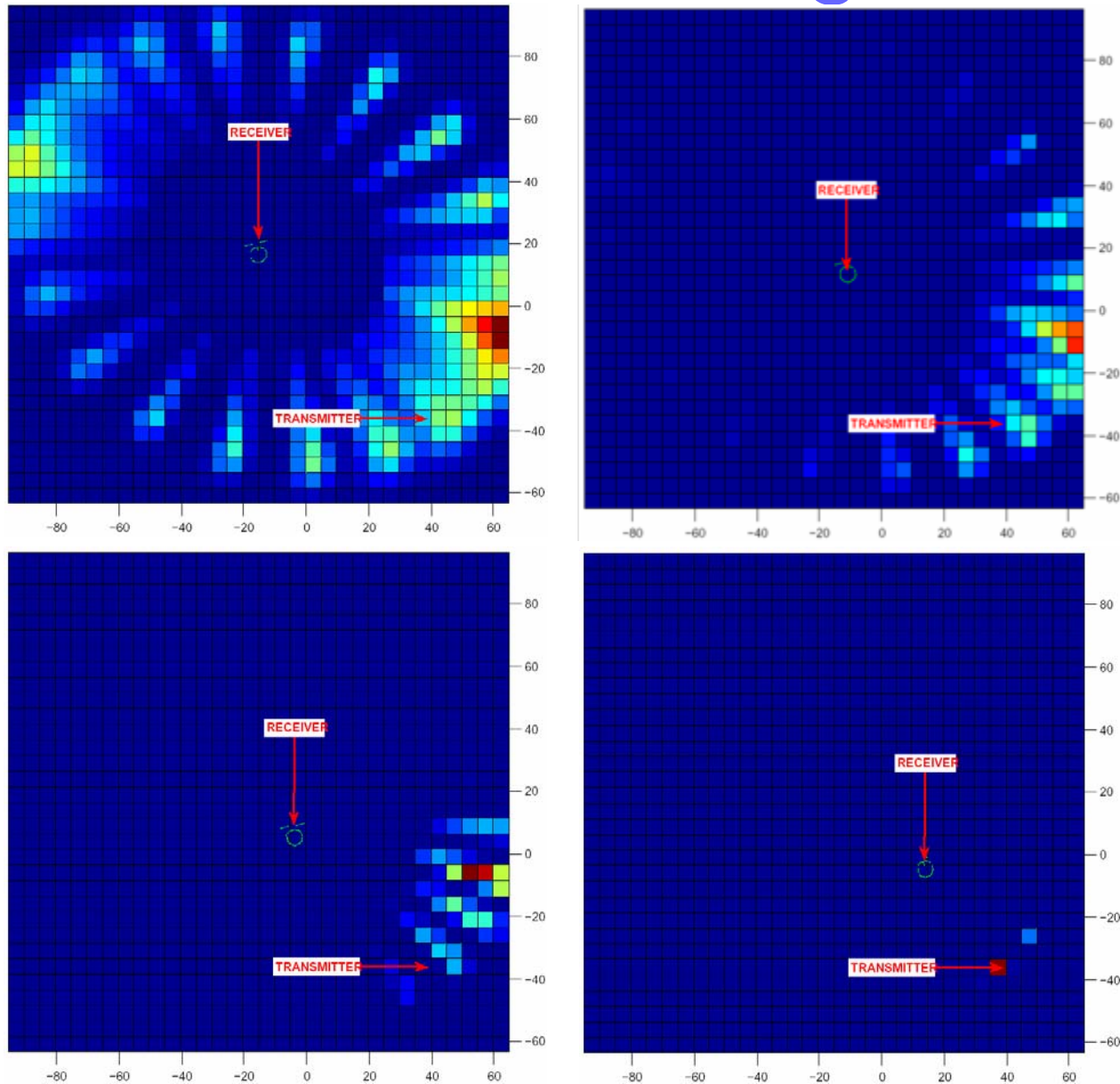
**for**  $i = 1$  to  $n$  **do**

$$p_{k|k-1}^{(i)} = \sum_j p(\mathbf{x}_k^{(i)} | \mathbf{x}_{k-1}^{(j)}, \mathbf{u}_{k-1}) p_{k-1}^{(j)} d\mathbf{x}_{k-1}^{(j)}$$

$$p_k^{(i)} = \eta p(\mathbf{z}_k | \mathbf{x}_k^{(i)}) p_{k|k-1}^{(i)}$$

**end for**

# Second solution: Histogram Filter



# Second solution: Histogram Filter

- It works, but:
  - Number of cells (50m x 50m x 5m x orientation x m)
  - High computational cost
  - Close to the victim, the magnetic field changes quickly

$$p(\mathbf{z}_k | \mathbf{x}_k) \rightarrow 0$$

» Multi-resolution grid needed

# Final solution: Sum of Gaussians Filter

$$p_{k|k-1} = \int p(\mathbf{x}_k | \mathbf{x}_{k-1}, \mathbf{u}_{k-1}) p_{k-1} d\mathbf{x}_{k-1}$$

$$p_k = \eta p(\mathbf{z}_k | \mathbf{x}_k) p_{k|k-1}$$

Prediction

Update

- Pdf approximated by a Sum of Gaussians SoG
- Each Gaussian is updated with an EKF

for  $i = 1$  to  $N$  do

$$F_k = \left. \frac{\partial \mathbf{f}_k}{\partial \mathbf{x}} \right|_{(\hat{\mathbf{x}}_{k-1|k-1}^{(i)})}$$

$$\hat{\mathbf{x}}_{k|k-1}^{(i)} = \mathbf{f}_k(\hat{\mathbf{x}}_{k-1|k-1}^{(i)}, \mathbf{u}_{k-1})$$

$$P_{k|k-1}^{(i)} = F_k P_{k-1|k-1}^{(i)} F_k^T + G_k Q_k G_k^T$$

$$H_k = \left. \frac{\partial \mathbf{h}_k}{\partial \mathbf{x}} \right|_{(\hat{\mathbf{x}}_{k|k-1}^{(i)})}$$

$$K_k = P_{k|k-1}^{(i)} H_k^T (H_k P_{k|k-1}^{(i)} H_k^T + R_k)^{-1}$$

$$\hat{\mathbf{x}}_{k|k}^{(i)} = \hat{\mathbf{x}}_{k|k-1}^{(i)} + K_k (\mathbf{z}_k - \mathbf{h}_k(\hat{\mathbf{x}}_{k|k-1}^{(i)}))$$

$$P_{k|k}^{(i)} = (I - K_k H_k) P_{k|k-1}^{(i)}$$

$$w_k^{(i)} = \eta w_{k-1}^{(i)} N(\mathbf{z}_k; \mathbf{h}_k(\hat{\mathbf{x}}_{k|k-1}^{(i)}), H_k P_{k|k-1}^{(i)} H_k^T + R_k)$$

end for

EKF Prediction

EKF Update

Weight Update



# Final Solution: SOGs filter

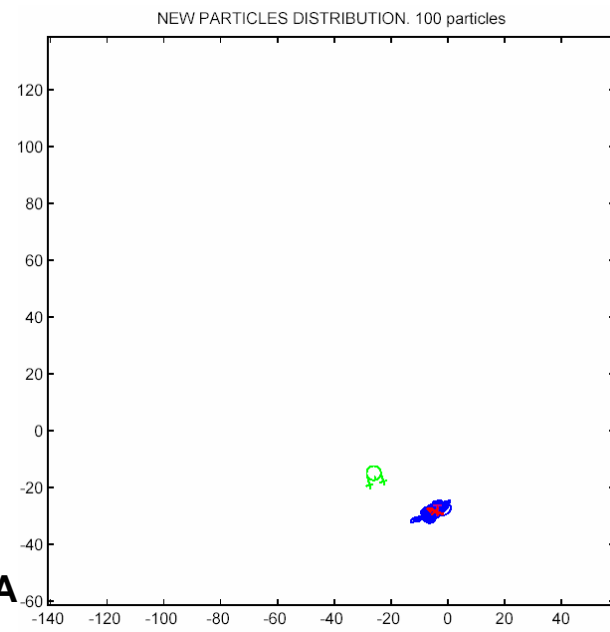
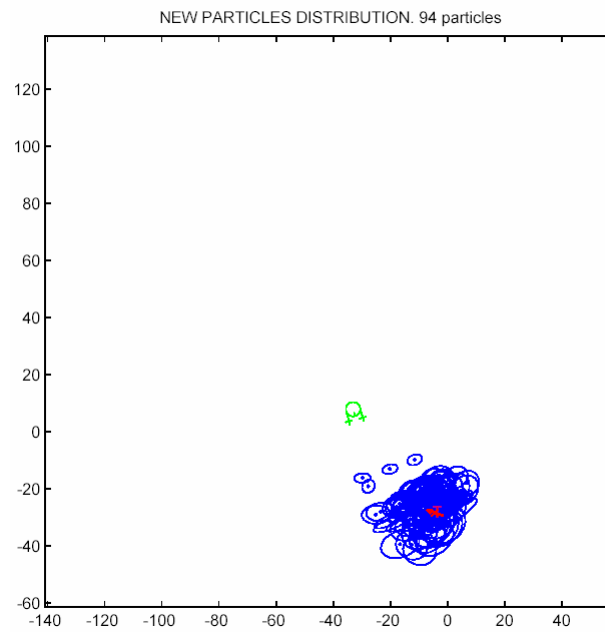
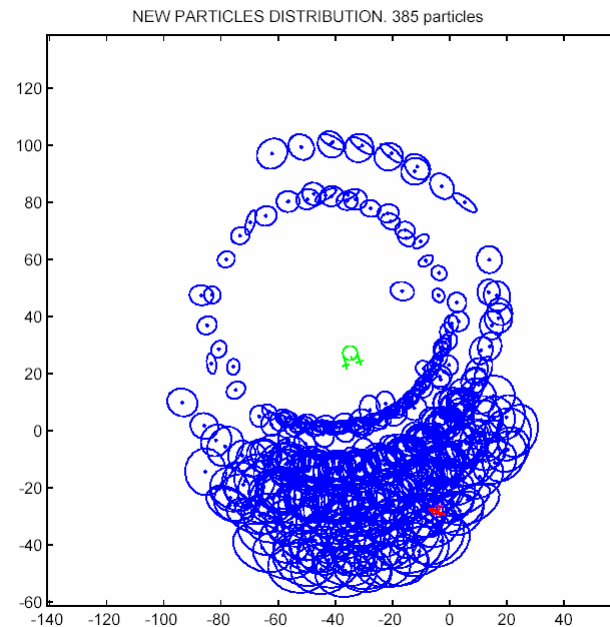
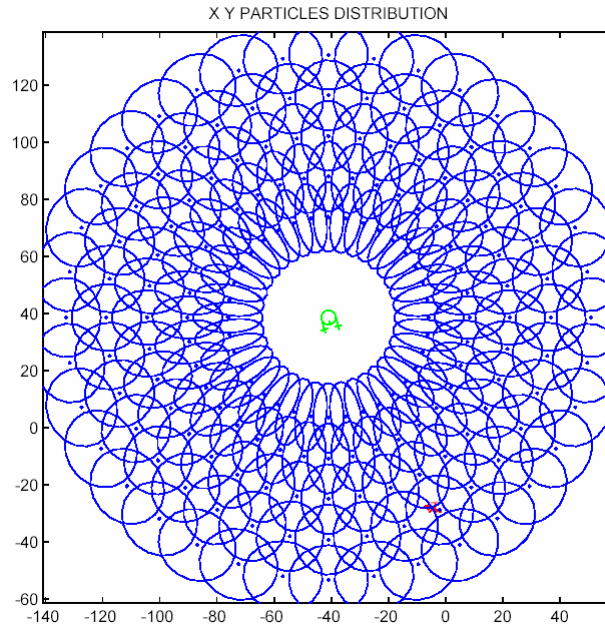
- With the first signal we can obtain the approximate distance  $r$  to the victim:

$$|H| = \frac{m}{4\pi r^3} \sqrt{1 + 3 \cos^2 \theta}$$

$$r_{min} = \left( \frac{m_{min}}{4\pi |H|} \right)^{\frac{1}{3}} \quad r_{max} = \left( \frac{2 \cdot m_{max}}{4\pi |H|} \right)^{\frac{1}{3}}$$

- Populate with Gaussians the area between  $r_{min}$  and  $r_{max}$
- Apply recursive estimation
  - Remove Gaussians with very small weight

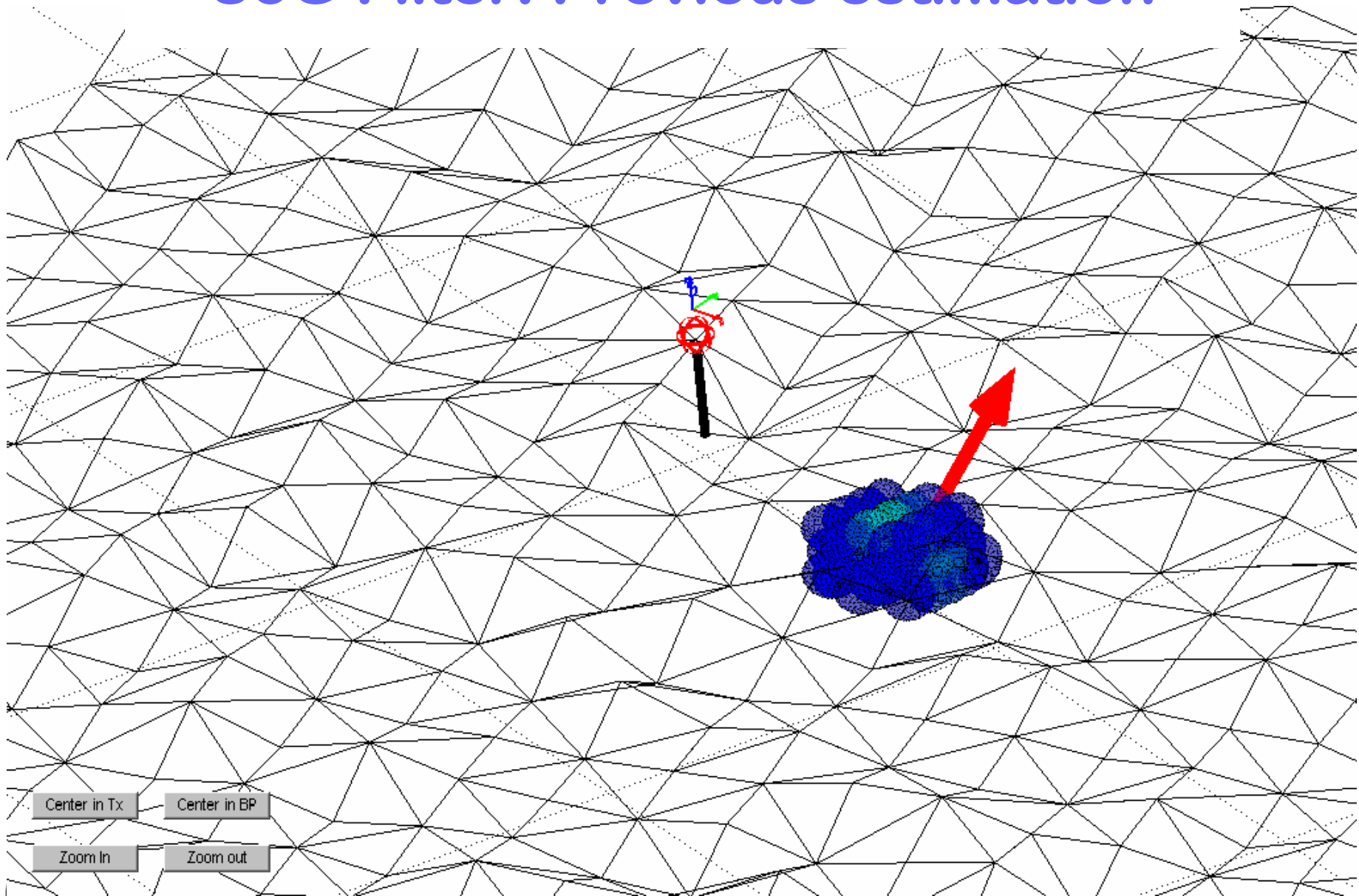
# Final Solution: SOGs filter (2D case)



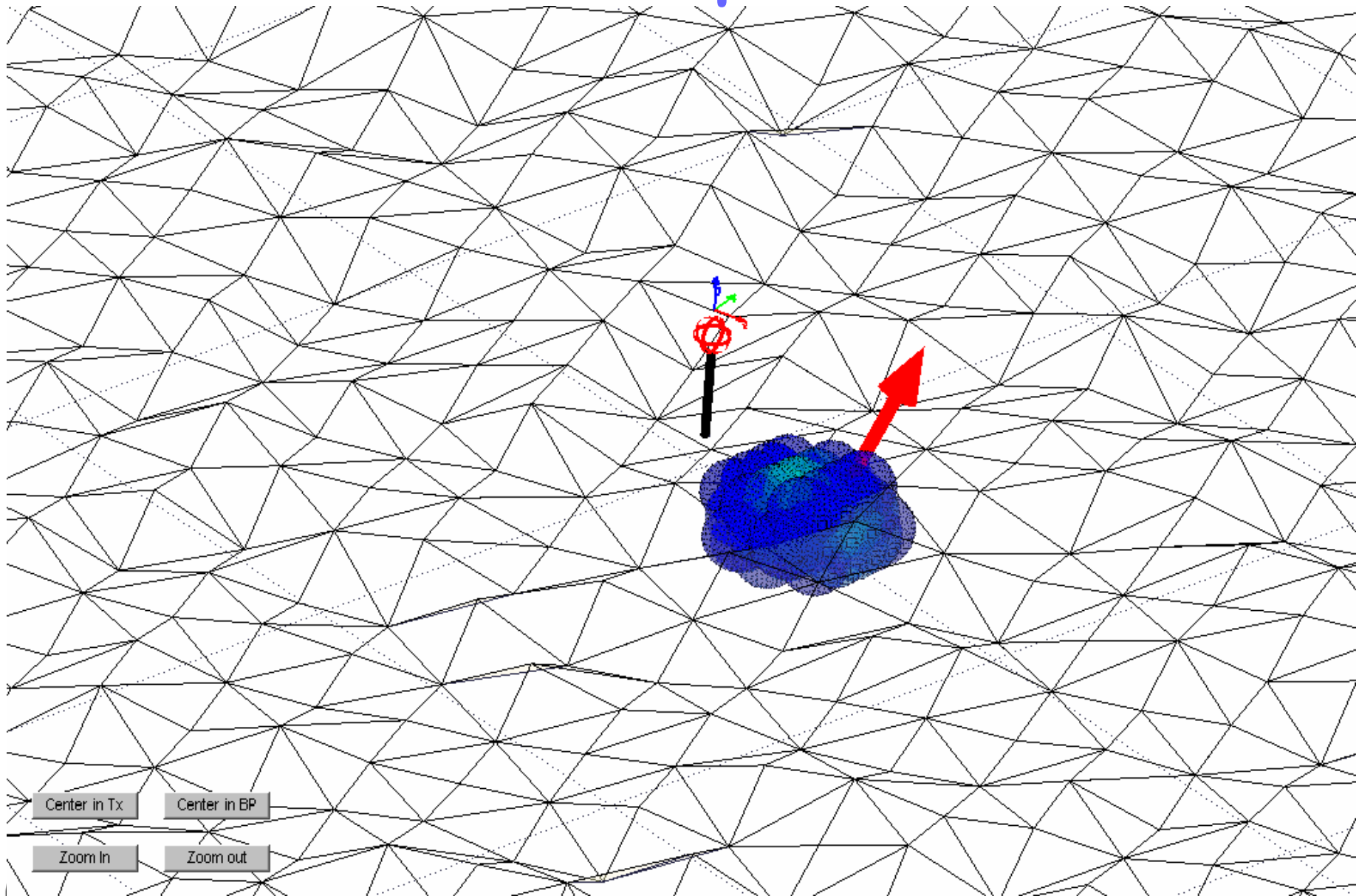
A

ardós

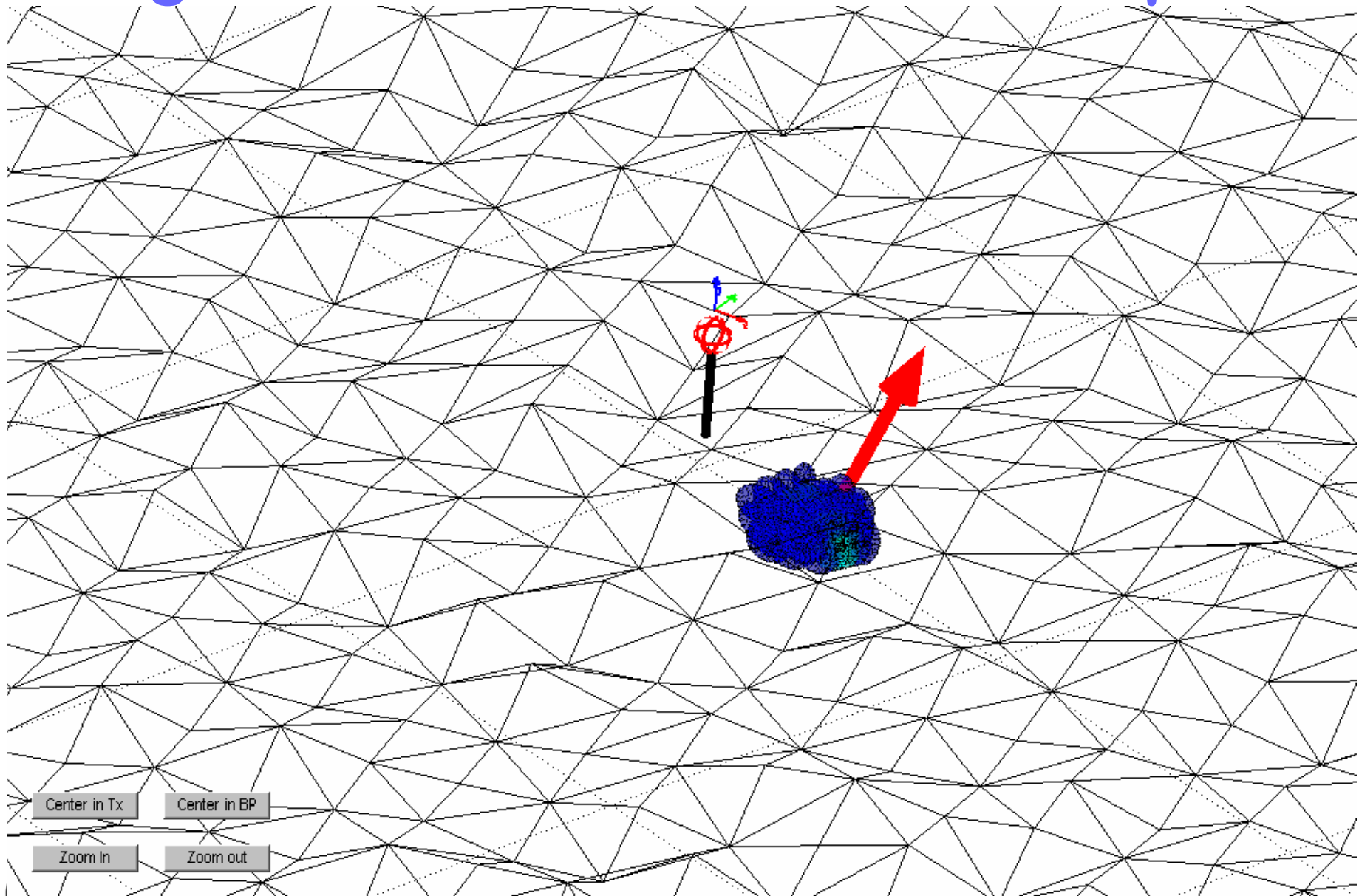
# SoG Filter: Previous estimation



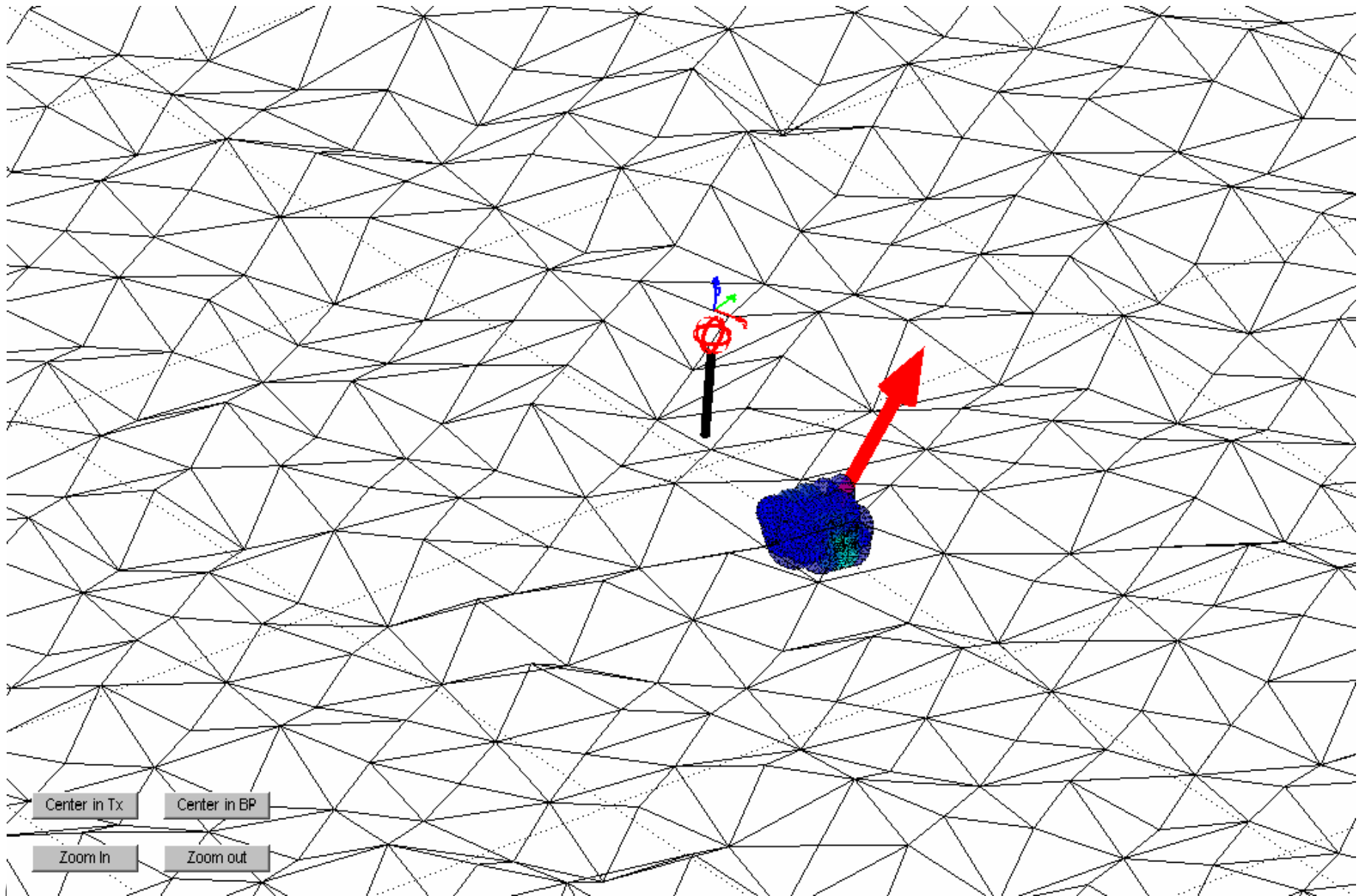
# Motion: EKF prediction



# Magnetic field measurement: EKF update



# Selection of Gaussians

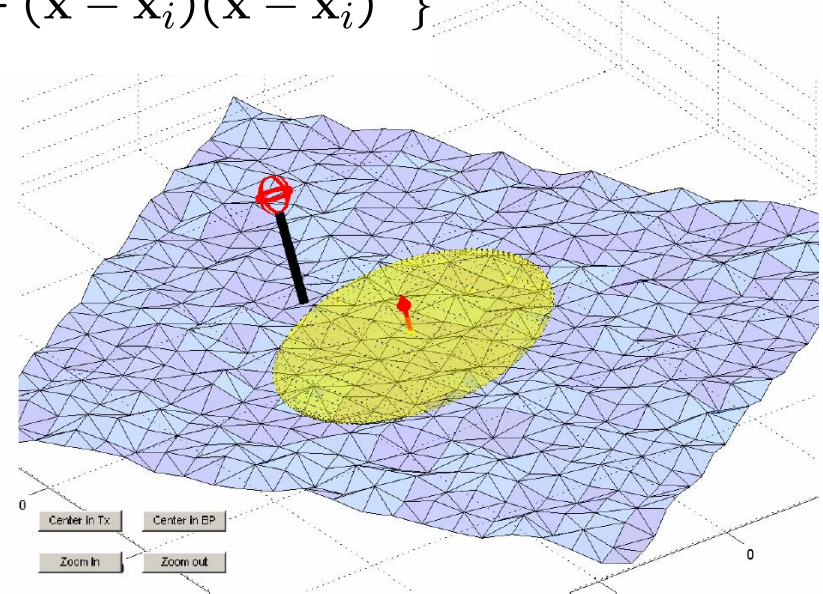
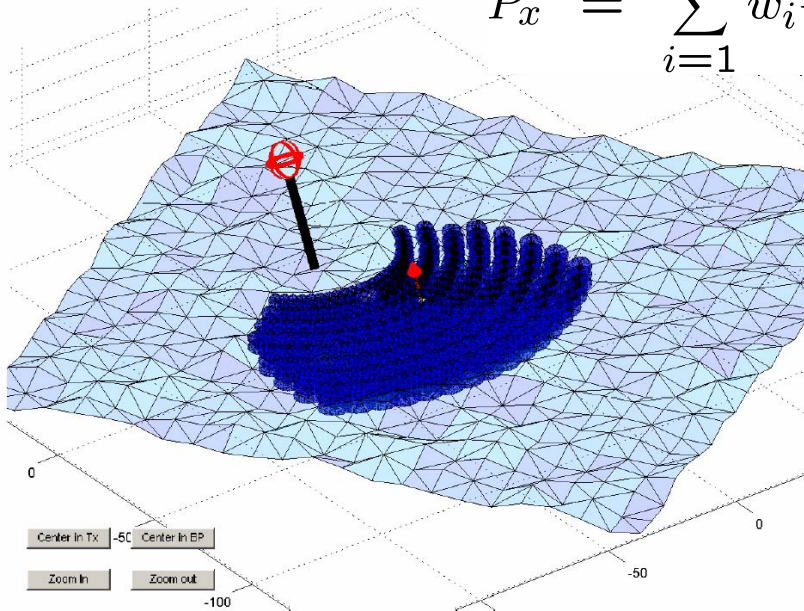


# Automated search

Mean and covariance of the SOGs

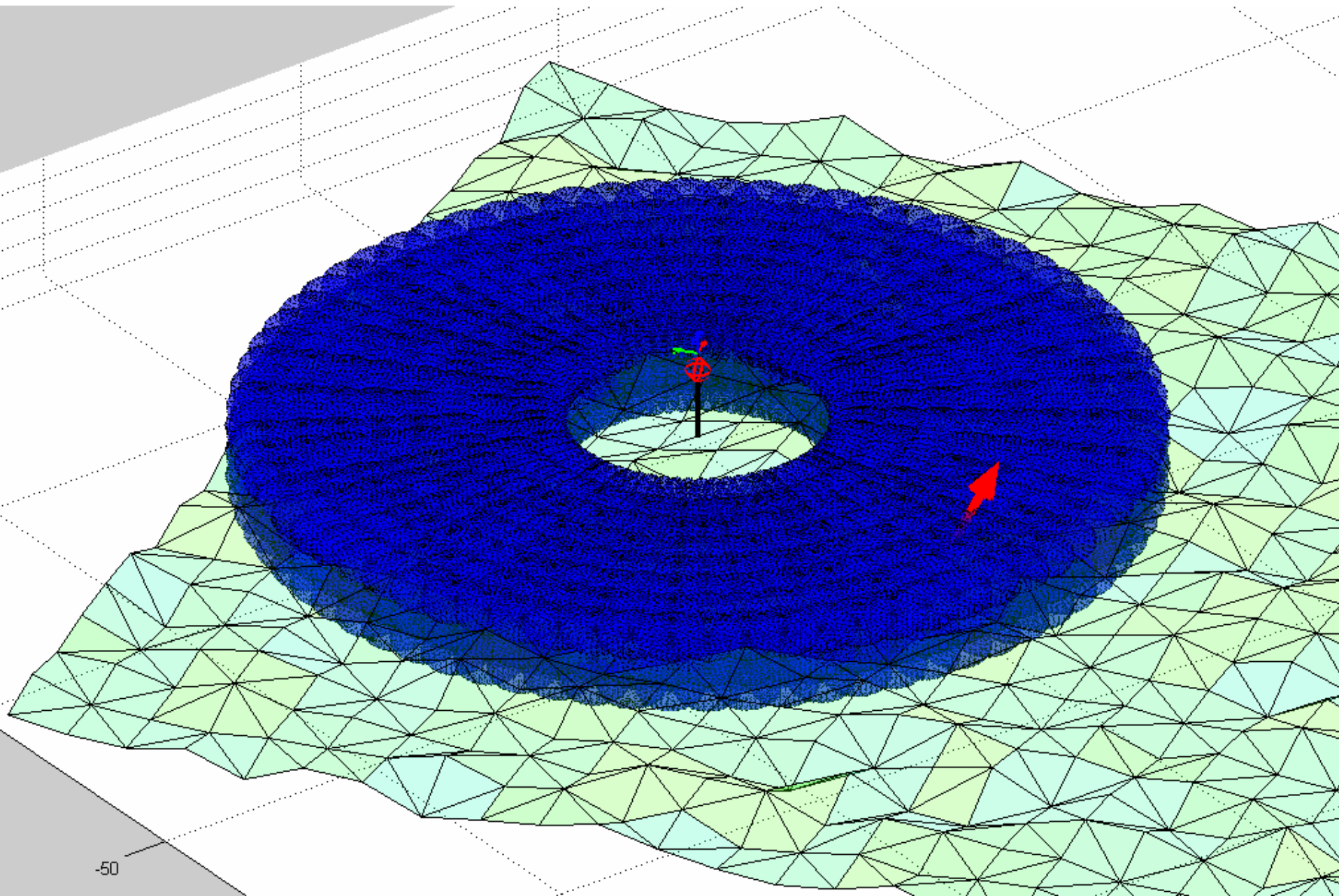
$$\bar{\mathbf{x}} = \sum_{i=1}^n w_i \bar{\mathbf{x}}_i$$

$$P_x = \sum_{i=1}^n w_i \{P_i + (\bar{\mathbf{x}} - \bar{\mathbf{x}}_i)(\bar{\mathbf{x}} - \bar{\mathbf{x}}_i)^T\}$$



- Initially, follow magnetic flux lines
- Later, move towards the mean of the SOGs

# Results (simulation)



# Results

- Fast localization using Sum of Gaussians
  - Classical search: 100s - 15 min (some novices give up)
  - Our proposal: 60-100s
  - Good precision: No need for tertiary search
- Work in progress
  - Prototype integration
  - Field tests next winter
  - INS to estimate rescuer motion
  - Detection of multiple close victims

# Why not Particle Filter?

```
for  $i = 1$  to  $n$  do
```

```
  sample  $\mathbf{x}_{k|k-1}^{(i)} \sim p(\mathbf{x}_k | \mathbf{x}_{k-1}^{(i)}, \mathbf{u}_{k-1})$ 
```

Prediction

```
   $w^{(i)} = p(\mathbf{z}_k | \mathbf{x}_{k|k-1}^{(i)})$ 
```

Weight update

```
end for
```

```
for  $i = 1$  to  $n$  do
```

```
  draw  $\mathbf{x}_{k|k}^{(i)}$  from  $\{\mathbf{x}_{k|k-1}^{(j)}\}$  with probability  $\propto w^{(j)}$ 
```

Resampling

```
end for
```

Vanilla particle filter:

- Easy to implement
- But does not work: loses particle diversity
- Carefull resampling improves a little
- Part of the state is static: adding artificial noise to improve diversity does not solve the problem  $\rightarrow$  RBPF

# RAO-Blackwellized Particle Filter

for  $i = 1$  to  $N$  do

Prediction

sample from  $\Psi_{k|k-1}^{(i)}, \mathbf{p}_{k|k-1}^{(i)} \sim p(\Psi_k, \mathbf{p}_k | \Psi_{k-1}^{(i)}, \mathbf{p}_{k-1}^{(i)}, {}^{R_{k-1}} \mathbf{x}_{R_k})$

$\hat{\mathbf{z}}_k^{(i)} = \mathbf{h}_k(\Psi_{k|k-1}^{(i)}, \mathbf{p}_{k|k-1}^{(i)}, \hat{\mathbf{m}}_{k-1}^{(i)})$

$H_{m_k} = \left. \frac{\partial \mathbf{h}_k}{\partial \mathbf{m}_k} \right|_{(\hat{\mathbf{m}}_{k-1}^{(i)})}$

$K_k = M_{k-1}^{(i)} H_{m_k}^T (H_{m_k} M_{k-1}^{(i)} H_{m_k}^T + R_k)^{-1}$

$\hat{\mathbf{m}}_k^{(i)} = \hat{\mathbf{m}}_{k-1}^{(i)} + K_k(\mathbf{z}_k - \hat{\mathbf{z}}_k^{(i)})$

$M_k^{(i)} = (I - K_k H_{m_k}) M_{k-1}^{(i)}$

$w_k^{(i)} = \eta w_{k-1}^{(i)} N(\mathbf{z}_k; \hat{\mathbf{z}}_k^{(i)}, H_{m_k} M_{k-1}^{(i)} H_{m_k}^T + R_k)$

Weight update

end for

for  $i = 1$  to  $N$  do

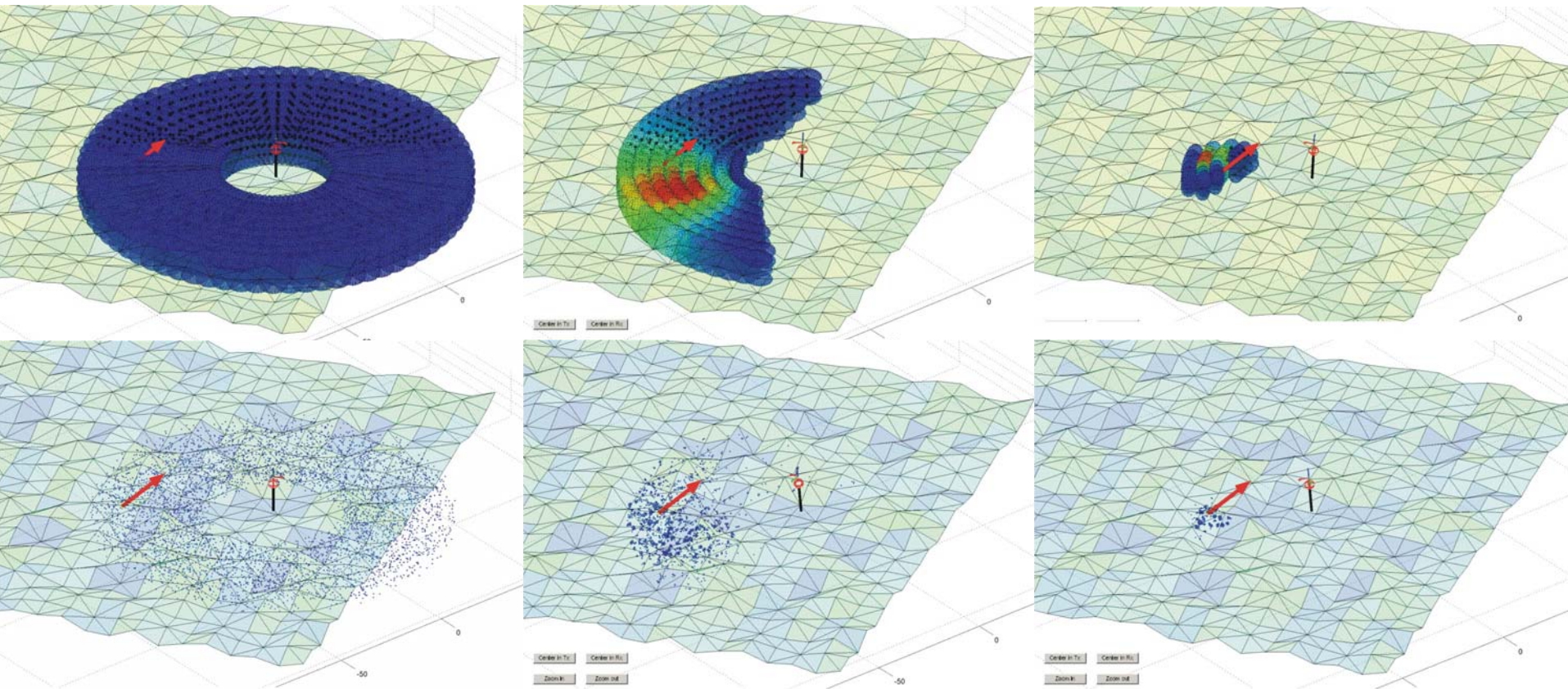
Resampling

draw  $\Psi_{k|k}^{(i)}, \mathbf{p}_{k|k}^{(i)}$  from  $\{\Psi_{k|k-1}^{(j)}, \mathbf{p}_{k|k-1}^{(j)}\}$  with probability  $\propto w_k^{(j)}$

end for

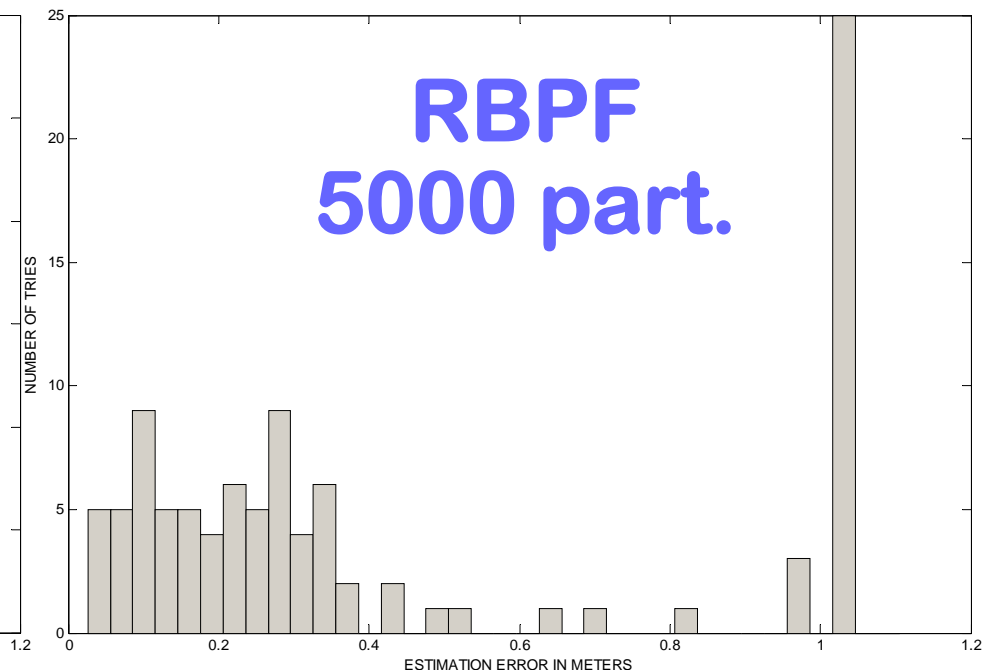
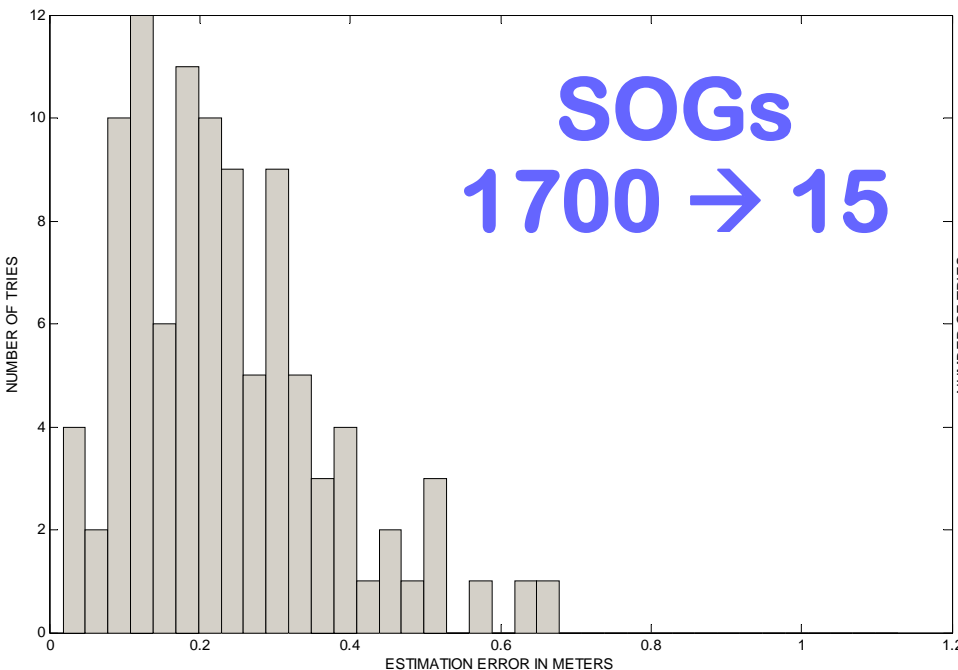


# SOGs .vs. RBPF



- Rao-Blackwellized Particle filter
  - 5000 particles, similar running time than SOGs
  - In 25% of the cases, error > 1 m
  - Problem: particle deprivation when close to the victim

# SOGs .vs. Particle Filter: 100 runs



- Similar running times for the first steps
- SOGs faster when close to the victim
- SOGs more reliable

Pedro Piniés, Juan D. Tardós, José Neira: Localization of avalanche victims using robocentric SLAM, IROS 2006, to appear.



# EKF-SLAM: Consistency

1. Introduction
2. Consistency of EKF-SLAM
3. Robocentric Mapping
4. Sum of Gaussians Filter
5. Application: rescue of avalanche victims
- 6. Conclusions**

# EKF-SLAM: inconsistency

- EKF-SLAM is only consistent for:
  - The linear case (1D robot )
  - Small scale maps ( $< 100\text{m}$ )
    - » Advice: Always use initial covariance = 0
- Inconsistency only becomes evident if:
  - Ground truth is available
  - Trying to close a big loop ( $> 100\text{m}$ )
- The inconsistency problem appears before the computational complexity problem !

# EKF-SLAM: improving consistency

- Robocentric mapping gives better consistency
- Local maps speed up EKF-SLAM and improve consistency (see talk tomorrow)
- SOGs filter splits non-linearities in small linear parts
  - Can it be fully applied to SLAM?

# Questions?

Juan Domingo Tardós  
University of Zaragoza, Spain  
[robots.unizar.es](http://robots.unizar.es)